CUBRID 2008 R4.1 Patch 2
Release Notes

Table of Contents

1. Overview

Release Notes Information

This document contains information about CUBRID 2008 R4.1 Patch 2. For the latest version of the release notes, go to the CUBRID Open Source Project Website (http://www.cubrid.org/).

For a more detailed description of the patches issued prior to CUBRID 2008 R4.1 Patch 2, see the release notes of the latest CUBRID 2008 R4.0 Patch. Unless otherwise specified in this document, CUBRID 2008 R4.1 includes CUBRID 2008 R4.1 Patch 1.

Revision History

Changes in the release notes of CUBRID 2008 R4.1 are as follows:

Date	Description
April, 2012	CUBRID 2008 R4.1 Patch 2 (8.4.1.2032)
February, 2012	CUBRID 2008 R4.1 Patch 1 (8.4.1.1018)
January, 2012	CUBRID 2008 R4.1 (8.4.1.0564)

References7

The documents distributed with CUBRID 2008 R4.1 are as follows:

문서	설명
Release Notes	Describes which features are new or have been changed from the previous CUBRID version.
Manual	Describes Introduction to CUBRID, Starting CUBRID, CSQL Interpreter, SQL Guide, Administrator's Guide, Performance Tuning, API References, CUBRID Manager, and CUBRID Manager Administrator's Guide.

Bug Report and User Feedback

CUBRID welcomes your active participation in bug reporting and looks forward to your feedback. You can register your bug reports and feedback on the following Websites:

Document	Description
Bug Report	CUBRID Open Source Project: http://www.cubrid.org/bug_bash
User Feedback	CUBRID Open Source Project: http://www.cubrid.org/forum CUBRID Website: http://www.cubrid.org/

Additional Information

You can find useful information about CUBRID on the following Websites:

Information	Site
CUBRID Product Information	http://www.cubrid.org/about
CUBRID License Information	http://www.cubrid.org/license

CUBRID User Documents	http://www.cubrid.org/documentation
CUBRID Training Service	http://www.cubrid.org/blog/categories/cubrid-trainings-channel/

2. CUBRID 2008 R4.1 Information

Features of CUBRID 2008 R4.1 Patch 2

CUBRID 2008 R4.1 Patch 2 is downward compatible and contains additional modifications that will be explained below.

Users of the previous version are highly recommended to download CUBRID 2008 R4.1 Patch 2, as it fixes errors that cause processes to hang or abnormally terminate, or that affect the stability of HA.

Issues
CUBRIDSUS-6706 Added an SQL hint that does not use a descending index scan CUBRIDSUS-6829 Added a feature that enables idling in the middle of backup in order to reduce the disk I/O burden resulting from the process CUBRIDSUS-5940 Modified the replication script to make it easier to build the replica node in the HA environment CUBRIDSUS-7043 Fixed a problem in which unnecessary scanning would be executed when NULL is included in the key value of multiple column indexes CUBRIDSUS-7067 There was a problem in which "SKIP ORDER BY" would not be performed when executing the SELECT ... ORDER BY statement
CUBRIDSUS-6781 Modified the broker log so that it now displays the URL to a CCI connection as specified by the user CUBRIDSUS-7001 Fixed a problem in which an error might occur when overflow key is rolled back after INSERT CUBRIDSUS-7104 Fixed a problem in which the NOT NULL constraint would disappear when the size of the VARCHAR type column declared by using the default key is increased CUBRIDSUS-7106 Fixed an error in which an incorrect size for the result type might be returned when a negative value is passed to the HEX function CUBRIDSUS-7217 Fixed a problem in which a server process would hang during a restart after the server process was abnormally terminated
CUBRIDSUS-6991 Fixed a problem in which a server process would not be normally restored after an abnormal termination CUBRIDSUS-7350 Fixed a problem in which a server process would be abnormally terminated when a deadlock occurs while there are a substantial number of locked transactions CUBRIDSUS-7216 Fixed a problem in which the query would fail to be processed even when the user provided a solution to a failed auto-increment of volume CUBRIDSUS-6933 Fixed a problem in which a server process might be abnormally terminated while processing a hierarchical query CUBRIDSUS-7075 Fixed a problem in which a server process might be abnormally terminated while processing a checkpoint
CUBRIDSUS-6811 Fixed a problem in which a server process might be abnormally terminated with an "Unknown key" error when a DML task is performed concurrently CUBRIDSUS-6946 Fixed a problem in which a rollback would fail while multiple DML executions change the same page CUBRIDSUS-7047 Fixed a problem in which CCI-based applications would not detect connection failure between the CAS server and DB server CUBRIDSUS-6998 Fixed an error in which CCI-based applications would reference a block of released memory when attempting to reconnect to the DB after a failed connection CUBRIDSUS-6679 Fixed a problem in which CAS might be abnormally terminated when executing an index scanning query
CUBRIDSUS-6788 Fixed a problem in which the IP of a newly connected application client would be set to 0.0.0.0 when CHANGE CLIENT occurs in CAS CUBRIDSUS-6781 Fixed cci_connect_with_url() so that it now uses the function parameter when the user and password are passed as a URL and parameter simultaneously CUBRIDSUS-6338 Fixed an error in which an incorrect type would be displayed when displaying table statistics in the CSQL interpreter CUBRIDSUS-7183 Fixed a problem in which applications could not connect to the master node even after restoration, when both slave node and ping node are unable to establish connection in the HA environment where the ping node is configured CUBRIDSUS-6956 Modified the program so that it now retries replication when a certain error occurs in the HA environment
CUBRIDSUS-6918 Fixed a problem in which data would not display when scanned with an descending index for a single column that is specified as the primary key in the HA environment CUBRIDSUS-6717 Fixed a problem in which a server process would be abnormally terminated when copylogdb requested an

old achieve log that has already been deleted in the HA environment CUBRIDSUS-7163 Fixed a problem in which replication integrity would be compromised if a failover occurs while delaying the replication in the HA environment CUBRIDSUS-7164 Modified to copy all previously created logs when the master server process is normally terminated in the HA environment CUBRIDSUS-6535 Fixed a problem in which 'ALTER TABLE ... ADD COLUMN reserved word' statement would not replicate in the HA environment
CUBRIDSUS-7387 Fixed a problem in which the log page without applylogdb is read infinitely if the ID value of the log page exceeds the maximum value of 32-bit integer type in the HA environment

For more information, see What's New in CUBRID 2008 R4.1 Patch 2.

Features of CUBRID 2008 R4.1 Patch 1

CUBRID 2008 R4.1 Patch 1 is downward compatible and contains the additional modifications that will be explained below. CUBRID 2008 R.4.1 users are highly recommended to download CUBRID 2008 R4.1 Patch 1 because it fixes database engine errors including a critical error in database volumes, and interface errors about CCI and JDBC. On a SELECT statement-driven workload, performance has been increased by approximately 45% compared to the previous version. This has been achieved by enhancing cubrid_query() and cubrid_execute() of PHP.

Issues
CUBRIDSUS-6070 Added a feature that configures logs for debugging in connection URLs CUBRIDSUS-6648 Improved performance by allowing PREPARE and EXECUTE to be executed with a single call, and reduced the number of message transmissions required when calling the exit function of a request handle in the CCI and PHP interfaces CUBRIDSUS-6906 Improved the performance of JDBC drivers CUBRIDSUS-6678 Fixed a problem in which the data volume of a database would crash when restarting the database server process CUBRIDSUS-6881 Fixed an error that would occur when executing INSERT in IFNULL, NVL, NVL2, or COALESCE function using host variables
CUBRIDSUS-6689 Fixed the 'unknown opcode' error that would occur when using a unary operator after a NOT operator CUBRIDSUS-6778 Fixed a problem in which live lock would occur in the process of acquiring and releasing the key lock during an index scan CUBRIDSUS-6914 Fixed a problem in which the application would be abnormally terminated when calling the exit function while accessing the database by using DATASOURCE CUBRIDSUS-6925 Fixed a cci_set_query() behavior error CUBRIDSUS-6958 Fixed a problem in which a CCI interface-based application would not attempt to reestablish connection with the priority host when it has been failed back
CUBRIDSUS-6753 Fixed the 'Unknown key' error that would occur when separating or merging B+ tree nodes while executing INSERT, DELETE, or UPDATE in an index CUBRIDSUS-6897 Fixed a problem in which a connection error would occur when a query is being performed more than 5 seconds in an environment in which multiple threads are connected to more than two brokers CUBRIDSUS-6779 Fixed a problem in which the status information of the application and CAS would not match when they have just been connected to each other CUBRIDSUS-6754 Fixed a problem in which the broker would fail to initialize when the number of CASs is set to 100 CUBRIDSUS-6364 Fixed a problem in which CAS could hang while writing an SQL log
CUBRIDSUS-6889 Fixed a problem in which the HA replication reflection process would not replicate the source CUBRIDSUS-6905 Fixed a problem in which the CAS and application would hang when using collection data types in JDBC

For more information, see What's New in CUBRID 2008 R4.1 Patch 1.

Features of CUBRID 2008 R4.1

Performing test on large amount of transaction, performance of INSERT and UPDATE was improved up to 70%. Also various SQL functions are added, regular expressions have been supported for conditional search, broker parameters are added, and monitoring features have been improved. In addition, approximate 60 major or minor bugs have been fixed.

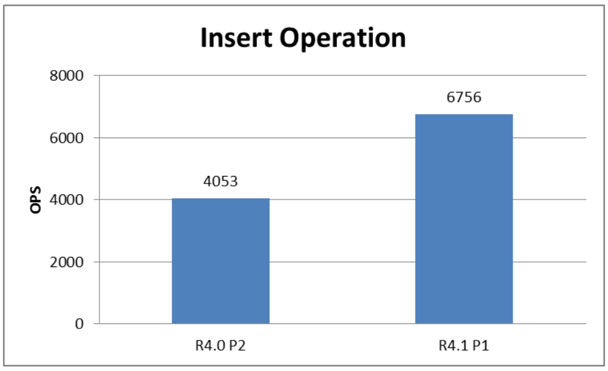
Please note that the value of the **CCI_DEFAULT_AUTOCOMMIT** broker parameter has been switched to ON by default. CCI and CCI-based driver (for PHP, ODBC and OLE DB) users should switch **CCI_DEFAULT_AUTOCOMMIT** to OFF to keep the previous version's auto commit mode.

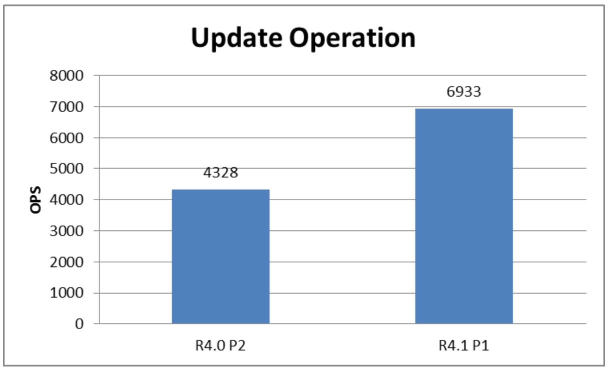
CUBRID 2008 R4.1 includes all modifications of previous CUBRID versions, up to CUBRID 2008 R4.0 Patch 2.

The key features of CUBRID 2008 R4.1 version are as follows:

Improved Performance of INSERT and UPDATE up to 70% While Performing Test on Large Amount of Transaction

Improved internal implementation to enhance performance when simultaneous processing transaction logs, writing memory buffers to a disk, and performing replication for HA. Consequently, performance of INSERT and UPDATE was dramatically improved up to 70% in comparison with 2008 R4.0 Patch 2 version while performing test on large amount of transaction based on YCSB (https://github.com/brianfrankcooper/YCSB/wiki). For more information on performance test, see "CUBRID 2008 R4.1 Patch1 QA Completion Report." (In the figure below, y-axis refers to the number of queries per second.)





Supports a Variety of SQL Functions and Regular Expressions

Added the ADDTIME, ASCII, BIN, CONV, FIND_IN_SET, HEX, and SERIAL_NEXT_VALUE (serial_name, number) functions, which can be used to obtain multiple serial values with a single call.

Added the REGEXP operator in order to allow conditions to be searched with regular expressions.

Added Some Broker Parameters and Status Information

The **MAX_QUERY_TIMEOUT** broker parameter is added to allow you to easily limit the query execution time of an application at the broker side, and the **APPL_SERVER_MAX_SIZE_HARD_LIMIT** broker parameter is added to force the restart of a broker application server (CAS) when its process exceeds the specified memory usage.

Implemented functionalities for the **cubrid broker status** utility so that it can output the transaction start time, the number of accesses from an application, and the number of restarts for each CAS.

Fixed about 60 New Bugs

Fixed about 50 bugs found in some SQL functions, indexes, queries, DB processes and HA processes.

Changed Operational Behaviors of Return Types of NULL Conditional Expressions (Default Value of Auto Commit Mode, IFNULL, etc.) and Supports the '0000-00-00' in a Date Value

Please note that the value of the CCI_DEFAULT_AUTOCOMMIT broker parameter has been switched to ON by default.

It now returns the VARCHAR type when the parameters of the IFNULL, NVL, NVL2, or COALESCE functions are passed in different types.

Now supports the '0000-00-00' format in the DATE type, and supports the '0000-00-00 00:00:00' the in the DATETIME and TIMESTAMP types.

For more information, see What's New in CUBRID 2008 R4.1.

Supported Platforms and System Requirements

The platforms supported by CUBRID 2008 R4.1 and hardware/software requirements for the installation are as follows:

Supported Platforms	Required Memory	Required Disk Space	Required Software
- Windows 32/64 Bit XP, 2003, Vista, Windows 7 - 32/64-bit Linux family Linux kernel version 2.4 and glibc version 2.3.4 or higher	1 GB or more	2 GB or more[1]	- JRE or JDK version 1.6 or higher Required when Java stored procedure is used.

Beginning with 2008 R4.0, CUBRID Manager Client is not automatically installed when installing the CUBRID package. For this reason, if you require CUBRID Manager you must install it separately. The CUBRID and the CUBRID Manager installation package can be downloaded from the download page.

Including CUBRID Query Browser, a variety of drivers such as PHP, ODBC, OLE, and DB can also be downloaded from the download page.

License

The GNU GPL v2 license or later applies to the CUBRID server engine, and the BSD license applies to CUBRID MANAGER and interfaces (APIs). For more information, see the License Policy on the CUBRID Website.

Version Compatibility and Operability

Application Compatibility

- Applications that use JDBC, PHP or CCI APIs from the earlier version of CUBRID can access the CUBRID 2008 R4.1 database. However, you must link the CUBRID 2008 R4.1 library or use the driver to use the added/improved features of JDBC, PHP or CCI interfaces.

[1] Requires a 500 MB of free disk space on the initial installation; requires approximately 1.5 GB of free disk space with a database creating with default options.

- Please note that query results may differ from those given in the earlier version because new reserved words have been added, and the specifications for some queries have been changed. For more information, see Cautions to Observe During Upgrade.
- An application that is developed by using the GLO class can be used after it is converted to an application or schema suitable to the BLOB or CLOB type.

CUBRID Manager Compatibility

Caution: Since 2008 R4.0 or higher, the CUBRID Manager client is not installed together when installing the CUBRID package. Therefore, you should install the CUBRID Manager additionally if you want to use it.

CUBRID Manager guarantees backward compatibility with the servers using CUBRID 2008 R2.1 or higher and uses the CUBRID JDBC driver that matches each server version. However, you must use a CUBRID Manager that is higher than CUBRID servers in version in order to utilize all the features of CUBRID Manager. The CUBRID JDBC driver is included in the $CUBRID/jdbc directory when CUBRID is installed.[2]

Drivers for 2008 R2.2 and higher versions are included in CUBRID Manager by default, which you can download separately from the cubrid.org Website.

Database Compatibility

When upgrading the server from the previous versions, you need to carry out database migration. For more information, see Database Migration Procedures.

Interoperability

If the CUBRID DB server and its broker server are operated separately, their interoperability is guaranteed, even when the operating systems are different. However, the bit version of a DB server must be identical to the bit version of a broker server. For example, the 64-bit DB server for Linux is interoperable with the 64-bit broker server for Windows, but it is not interoperable with a 32-bit broker server.

How to Install CUBRID 2008 R4.1

Product Installation

An installation package for Linux is provided as a script that includes a binary, tar.gz-compressed file and Linux RPM package format. For more information about installation, see Getting Started with CUBRID > Installing and Running CUBRID > Installing and Running on Linux.

The installation package for Windows can be installed using the installation wizard. For more information about installation, see Getting Started with CUBRID > Installing and Running CUBRID > Installing and Running on Windows. See also CUBRIDSUS-3267 Caution when setting a directory path in a Windows environment.

Configuring CUBRID and OS Environment Variables

Before you start CUBRID, you must configure the CUBRID and OS environment variables. See Getting Started with CUBRID > Configuring the Environment Variable and Starting CUBRID Service > Configuring the Environment Variable in the manual. It is important to configure the Java environment when using CUBRID Manager and Java stored

[2] $CUBRID is an environment variable for Linux that specifies the location where CUBRID has been installed. In Windows environments, $CUBRID is used in the same way as %CUBRID%.

procedures. For more information, see CUBRID SQL Guide > Java Stored Function/Procedure > Environment Configuration for Java Stored Functions/Procedure in the manual.

How to Upgrade to CUBRID 2008 R4.1

Upgrading from 2008 R4.0 to 2008 R4.1

Users who are using versions CUBRID 2008 R4.0 and 2008 R4.0 Patch X should install 2008 R4.1 and modify parameter values in the existing environment configuration file.

- cubrid.conf
  - Because the default value of thread_stacksize has been changed from 100K to 1M, it is recommended that users who have not configured this value check memory usage of CUBRID-associative processes.
  - Because the minimum value of data_buffer_size has been changed from 64K to 16M, users who have configured this value less than 16M must change the value equal to or greater than 16M.
- cubrid_broker.conf
  - Because the default value of **CCI_DEFAULT_AUTOCOMMIT** has been changed to ON, users who have not configured this value should change it to OFF if they want to keep auto commit mode.
  - The minimum value of **APPL_SERVER_MAX_SIZE_HARD_LIMIT** is 1024M. It is recommended that users configure the value greater than **APPL_SERVER_MAX_SIZE** if users configured **APPL_SERVER_MAX_SIZE** (see CUBRIDSUS-5956 Provided a solution to prevent CAS memory usage spikes in this document).

Upgrading from 2008 R4.0 Beta to 2008 R4.1

See “Upgrading from 2008 R4.0 to 2008 R4.1” section above.

Cautions to Observe During Upgrade

- Saving the Existing Configuration File
  - Archive the configuration files in the $CUBRID/conf directory (cubrid.conf, cubrid_broker.conf and cm.conf) and the DB location file (databases.txt) in the $CUBRID_DATABASES directory.[3]
- Checking New Reserved Words
  - You can check whether reserved words are being used or not by applying the CUBRID 2008 R4.1 reserved word detection script, **check_reserved.sql**, which is distributed through the CUBRID installation package or the download page. If the reserved words are being used as identifiers, the identifiers must be modified. *S*ee CUBRID SQL Guide > Identifier in the manual.
- DB Migration
  - It is necessary to migrate the previous version database to CUBRID 2008 R4.1, because CUBRID 2008 R4.1 is not compatible with database volumes whose version is 2008 R4.0 Beta or earlier. See Database Migration Procedures below.
  - To migrate from 2008 R4.0 Beta, use the **migrate_r40beta2ga** utility provided at the download page.

[3] Environment variables such as $CUBRID and $CUBRID_DATABASES are used in Linux. Windows uses environment variables such as %CUBRID% and %CUBRID_DATABASES% instead.

- CUBRID 2008 R3.1 and later don't support GLO and the LOB type replaces the GLO feature. For this reason, applications or schemas that use GLO must be modified to be compatible with LOB. See Migration for GLO Class Users below.
- Use the **cubrid unloaddb/loaddb** utility to migrate from the 2008 R3.1 or earlier versions to CUBRID 2008 R4.0.

- **Reconfiguring Environments for Replication or HA**
  - From 2008 R4.0, the replication feature is no longer supported; for this reason, it is recommended to reconfigure the DB migration and HA environment for systems in which previous replication versions are used. In addition, for systems that use Linux Heartbeat-based HA feature, which is provided in CUBRID 2008 R2.0 and 2008 R2.1, you must reconfigure to DB migration and the CUBRID Heartbeat-based HA environment for better operational stability. See Database Migration Procedures in HA Environment.
  - To reconfigure the HA environment configuration, see Administrator's Guide > CUBRID HA in the manual.

Database Migration Procedures

Migration from 2008 R3.x and Earlier Versions to 2008 R4.1

Migration from 2008 R4.0 Beta

Migration for GLO Class Users

- **Migration from 2008 R3.x and Earlier Versions to 2008 R4.1**
  - Users of 2008 R3.x and earlier versions must migrate the database to 2008 R4.1, as shown in the following table.
  - In addition, if an earlier version of GLO classes are being used, additional tasks are needed. See Migration for GLO Class Users below.
  - The following table shows how to perform the migration using the reserved word detection script, check_reserved.sql, which is separately distributed through the **cubrid unloaddb/loaddb** utility and the download page. See Administrator's Guide > Database Administration in the manual and CUBRIDSUS-3826 Cautions for stop of GLO class support in this document.

Step	Linux Environment	Windows Environment
Step C1: Stop CUBRID Service	% cubrid service stop	Stop CUBRID Service Tray.
Step C2: Execute the reserved words detection script.	Execute the following command in the directory where the reserved word detection script is located. Execute migration or identifier modification by checking the detection result (For the allowable identifier, see the *related manuals*). % csql -S -u dba -i check_reserved.sql testdb	
Step C3: Unload the earlier version of the DB	Store the databases.txt file and the configuration files under the conf directory of the earlier version in a separate directory (C3a). Execute the cubrid unloaddb utility and store the file generated at this point in a separate directory (C3b). % cubrid unloaddb -S testdb Delete the existing database (C3c). % cubrid deletedb testdb	
		Uninstall the earlier version of CUBRID.
Step C4: Install 2008 R4.1	See How to Install CUBRID 2008 R4.1 in this document.	
Step C5: Database creation and	Go to the directory where you want to create a database, and create one. (C5a). % cd $CUBRID/databases/testdb	

data loading	% cubrid createdb testdb Execute the cubrid loaddb utility with the stored files in (C3b). (C5b). % cubrid loaddb -s testdb_schema -d testdb_objects -i testdb_indexes testdb	
Step C6: Back up the new version of the DB	% cubrid backupdb -S testdb	
Step C7: Configure the CUBRID environment and start the CUBRID Service	Modify the configuration file. At this point, partially modify the configuration files from the earlier version stored in step (C3a) to fit the new version (For system parameter settings, see the cautions and related manuals). % cubrid service start % cubrid server start testdb	Start the service by selecting CUBRID Service Tray > [Service Start]. Start the database server from the command prompt. % cubrid server start testdb

- **Migration from 2008 R4.0 Beta**
  - 2008 R4.0 Beta users must migrate the database as shown in the table below.
  - Migration can be performed by using the **migrate_r40beta2ga** utility provided at the download page. However, when the page size of a database volume is less than 4K, migration should be performed by using **cubrid unloaddb/loaddb**. See Migration from 2008 R3.x and Earlier Versions to 2008 R4.1.
  - If the migration fails, retry by restoring the DB of the previous version (**cubrid restoredb**) that has been backed up.

Step	Linux Environment	Windows Environment
Step D1: Stop CUBRID Service	% cubrid service stop	Stop CUBRID Service Tray.
Step D2: Back up the existing DB	Back up the database in case you need to restore it later, and store the backup file in a separate directory (backup). (D2a) % mkdir backup % cubrid backupdb -S -D backup testdb Store the databases.txt file and the configuration files under the conf directory of the earlier version in a separate directory. (D2b)	
		Uninstall the earlier version of CUBRID.
Step D3: Install 2008 R4.1	See How to Install CUBRID 2008 R4.1 in this document.	
Step D4: Run the migration tool	Copy the databases.txt file stored in step (D2b) to the directory in which 2008 R4.1 will be installed. (D4a) Execute the migrate_r40beta2ga utility as shown below. (D4b) % migrate_r40beta2ga testdb	
Step D5: Back up the new version of the DB	% cubrid backupdb -S testdb	
Step D6: Configure the CUBRID environment and start the CUBRID Service	Partially modify the configuration files of the previous version saved in accordance with the new version (For system parameter settings, see the cautions and related manuals). % cubrid service start % cubrid server start testdb	Start the service by selecting CUBRID Service Tray > [Service Start]. Start the database server from the command prompt. % cubrid server start testdb

- **Migration for GLO Class Users**
  - If you use GLO classes, you must modify applications and schema in order to use BLOB or CLOB types, since GLO classes are not supported in 2008 R3.1. If this modification is not easy, it is not recommended to perform the migration.

Database Migration Procedures in HA Environment

HA migration from 2008 R2.2 or higher to 2008 R4.1

HA Migration from 2008 R2.0 or 2008 R2.1 to 2008 R4.1

- **HA migration from 2008 R2.2 or higher to 2008 R4.1**
  - In the scenario described below, the current service is stopped to perform an upgrade in an environment in which a broker, a master DB and a slave DB are operating on different servers. For a scenario in which an upgrade is performed without stopping the service, see the other guides.

Step	Description
Steps H1-H6: Perform steps C1-C6 or steps D1-D5 on the master node, depending on the version.	Run the CUBRID upgrade and database migration in the master node and back up the 2008 R4.1 database.
Step H7: Install CUBRID 2008 R4.1 in the slave node	Delete the previous version of the database from the slave node and install a new version. For more information about installation, see How to Install CUBRID 2008 R4.1 in this document.
Step H8: Restore the backup copy of the master node in the slave node.	Restore the 2008 R4.1 database backup copy (testdb_bk*) of the master node, which is created in step H6, to the slave node. % scp user1@master:$CUBRID/databases/databases.txt $CUBRID/databases/. % cd ~/DB/testdb % scp user1@master:~/DB/testdb/testdb_bk0v000 . % scp user1@master:~/DB/testdb/testdb_bkvinf . % cubrid restoredb testdb
H9 step: Reconfigure HA environment and start HA mode	In the master node and the slave node, set the CUBRID environment configuration file (cubrid.conf) and the HA environment configuration file (cubrid_ha.conf) (See the related manuals). % vi $CUBRID/conf/cubrid.conf ha_mode=on % vi $CUBRID/conf/cubrid_ha.conf [common] ha_port_id=59901 ha_node_list=cubrid-ha@master:slave ha_db_list=testdb Start the DB in HA mode in the master and slave nodes (See the related manuals). % cubrid heartbeat start
Step H10: Install CUBRID 2008 R4.1 in the broker server, and start the broker	For more information about installation, see How to Install CUBRID 2008 R4.1 in this document. Start the broker in the Broker server (See the related manuals). % cubrid broker start

- **HA Migration from 2008 R2.0 or 2008 R2.1 to 2008 R4.1**
    - If you are using the HA feature of CUBRID 2008 R2.0 or 2008 R2.1, you must upgrade the server version, migrate the database, set up a new HA environment, and then change the Linux Heartbeat auto start setting used in 2008 R2.0 or 2008 R2.1. If the Linux Heartbeat package is not needed, delete it.
    - Perform steps H1-H10 above, then perform step H11 below:

Step	Description
Step H11: Change the previous Linux heartbeat auto start settings	Perform the following task in the master and slave nodes from a root account. [root@master ~]# chkconfig --del heartbeat // Performing the same job in the slave node

Cautions when Using the Replication Feature

Since the replication feature from CUBRID 2008 R4.1 is deleted, you must use the HA feature in order to build a redundant environment in the existing environment using the replication feature. After performing a server version upgrade and a database migration, you can build a new HA environment. For building an HA environment, see Administrator's Guide > CUBRID HA in the manual.

3. What's New in CUBRID 2008 R4.1 Patch 2

New Features

CUBRIDSUS-6706 Added an SQL hint that does not use a descending index scan

NO_DESC_IDX, the SQL hint that does not use a descending index scan, has been added.

```
SELECT /*+ NO_DESC_IDX */ * FROM tbl WHERE col > 5 ORDER BY col DESC;
```

CUBRIDSUS-6829 Added a feature that enables idling in the middle of backup in order to reduce the disk I/O burden resulting from the process

--sleep-msecs (unit: msec), which is an idle option for a specified amount of time whenever a file with the size of 1MB is read in order to reduce a disk I/O burden resulting from executing **cubrid backupdb** has been added

```
cubrid backupdb --sleep-msecs=5 demodb
```

Enhanced Features

CUBRIDSUS-5940 Modified the replication script to make it easier to build the replica node in the HA environment

In earlier versions, users were only able to use the replication rebuilding script (ha_make_slavedb.sh) for slave nodes. In the updated version, ha_make_slavedb.sh has been improved so that replica nodes can use the script as well.

CUBRIDSUS-7043 Fixed a problem in which unnecessary scanning would be executed when NULL is included in the key value of multiple column indexes

Previously, there was a problem in which CPU usage would be increased more than the warrant query quality by scanning unnecessary data when NULL is included in the key value of multiple column indexes. This problem has been solved.

```
CREATE INDEX i_tbl ON tbl (id1, id2, flag);
-- A NULL value is included in flag
INSERT INTO tbl VALUES (10, 300, Y), (20, 500, NULL), …, (10000, 57000, NULL);
SELECT * FROM tbl WHERE id1 = 10 AND id2 = 300 AND flag='Y';
```

CUBRIDSUS-7067 There was a problem in which "SKIP ORDER BY" would not be performed when executing the SELECT ... ORDER BY statement

Previously, there was a problem in which ORDER BY would be redundantly performed when it was already sorted by indexes when executing the INSERT INTO ... SELECT ... ORDER BY statement. This problem has been solved.

CUBRIDSUS-6781 Modified the broker log so that it now displays the URL to a CCI connection as specified by the user

Modified the broker log so that it now displays the URL to a CCI connection as specified by the user. In earlier versions, the value of althosts in the input URL would not be displayed as shown below:

```
Input URL=cci:cubrid:10.0.10.89: 30000:cub01:::?althosts=10.0.10. 90:30000&rctime=60
Output URL before changes are made=cci:cubrid:10.24.159.89:30000:cub01:::
Output URL after changes are made = Same as the input URL
```

Fixed Errors

CUBRIDSUS-7001 Fixed a problem in which an error might occur when overflow key is rolled back after INSERT

Previously, a memory reference error might occur when an overflow key, a key that is too large to be stored in a single page, was rolled back after INSERT. This problem has been solved.

CUBRIDSUS-7104 Fixed a problem in which the NOT NULL constraint would disappear when the size of the VARCHAR type column declared by using the default key is increased

Previously, the NOT NULL constraint would disappear when the size of the VARCHAR type column that had been declared as the default key by executing "ALTER TABLE ~ CHANGE col" was increased. This problem has been solved.

```
CREATE TABLE tbl1 (col VARCHAR(10) PRIMARY KEY);
ALTER TABLE tbl1 CHANGE col col VARCHAR(20);
```

CUBRIDSUS-7106 Fixed an error in which an incorrect size for the result type might be returned when a negative value is passed to the HEX function

Previously, the HEX function might return an incorrect type of size (precision) when a negative value was passed to it. This problem has been fixed.

```
ResultSet r = s.executeQuery("SELECT HEX(N) FROM T2");
ResultSetMetaData m = r.getMetaData();
m.getPrecision() // Gets incorrect data from previous version.
```

CUBRIDSUS-7217 Fixed a problem in which a server process would hang during a restart after the server process was abnormally terminated

Previously, a server process would hang during a restart after it was abnormally terminated during a long transaction. This problem has been solved.

CUBRIDSUS-6991 Fixed a problem in which a server process would not be normally restored after an abnormal termination

Previously, transactions would not be rolled back normally after a server process was abnormally killed during a data INSERT. This problem has been solved.

CUBRIDSUS-7350 Fixed a problem in which a server process would be abnormally terminated when a deadlock occurs while there are a substantial number of locked transactions

Previously, server process would be abnormally terminated when a deadlock occurred while there were a substantial number of locked transactions. This problem has been solved.

CUBRIDSUS-7216 Fixed a problem in which the query would fail to be processed even when the user provided a solution to a failed auto-increment of volume

Previously, the query would fail to be processed even when the user provided a solution, such as restarting the server, to a failed auto-increment that was caused by an abnormal termination of the server process during an auto-increment. This problem has been solved.

Previously, the query would fail to be processed even when the user provided a solution, such as restarting the server, to a failed auto-increment of volume that was caused by an abnormal termination of the server process during an auto-increment because the corrupted file would not be deleted. This problem has been solved.

CUBRIDSUS-6933 Fixed a problem in which a server process might be abnormally terminated while processing a hierarchical query

Previously, a server process might be abnormally terminated when processing a hierarchical query on a column that contains an exceptionally large character string. This problem has been solved.

CUBRIDSUS-7075 Fixed a problem in which a server process might be abnormally terminated while processing a checkpoint

Previously, a server process might be abnormally terminated while processing a checkpoint. This problem has been solved.

CUBRIDSUS-6811 Fixed a problem in which a server process might be abnormally terminated with an "Unknown key" error when a DML task is performed concurrently

Previously, a server process might be abnormally terminated with an "Unknown key" error related to the B+tree index when the DML task is performed concurrently in multiple programs. This problem has been solved.

CUBRIDSUS-6946 Fixed a problem in which a rollback would fail while multiple DML executions change the same page

Previously, a rollback would fail and an "Internal system failure: no more specific information is available." error would occur when multiple transactions simultaneously access, delete or change the record length of the same page. This problem has been solved.

CUBRIDSUS-7047 Fixed a problem in which CCI-based applications would not detect connection failure between the CAS server and DB server

Previously, CCI-based applications[4] would not detect a connection failure between the CAS server and DB server, imparing the application-CAS server-DB server connection. This behavior prevented CCI-based applications from re-establishing the connection to the DB. This problem has been solved.

CUBRIDSUS-6998 Fixed an error in which CCI-based applications would reference a block of released memory when attempting to reconnect to the DB after a failed connection

Previously, CCI-based applications would reference a block of released memory when attempting to reconnect to the DB by using cci_connect(). This error has been fixed.

CUBRIDSUS-6679 Fixed a problem in which CAS might be abnormally terminated when executing an index scanning query

Previously, CAS might be abnormally terminated due to incorrect reference of memory for statistics information when executing an index scanning query. This problem has been solved.

[4] Application with CCI interface or with drivers based on CCI interface. PHP, ODBC, OLE DB, Perl, Ruby, Python etc. are drivers based on CCI interface. Drivers based on CCI interface require CCI dynamic link library. As a reference, JDBC is not a driver developed with CCI interface.

CUBRIDSUS-6788 Fixed a problem in which the IP of a newly connected application client would be set to 0.0.0.0 when CHANGE CLIENT occurs in CAS

Previously, a newly connected application client was set to 0.0.0.0 when CHANGE CLIENT occurred in CAS. When this problem occurred in earlier versions, the application client IP displayed through cubrid broker status would be shown as 0.0.0.0; if the application client requested a query cancel of the corresponding CAS, it would fail. This problem has been solved.

CUBRIDSUS-6781 Fixed cci_connect_with_url() so that it now uses the function parameter when the user and password are passed as a URL and parameter simultaneously

Fixed cci_connect_with_url() so that it now uses the function parameter when the user and password are passed as a URL and parameter simultaneously. The example below uses "usr1" and "pwd123" as the ID and password.

```
cci_connect_with_url("cci:CUBRID:192.168.0.1:33000:demodb:dba:pwd:", "usr1", "pwd123");
```

CUBRIDSUS-6338 Fixed an error in which an incorrect type would be displayed when displaying table statistics in the CSQL interpreter

Previously, smallint was erroneously displayed as UNKNOWN_TYPE when displaying table statistics in the CSQL interpreter. This error has been fixed.

```
csql> ;info stats tbl_name
```

CUBRIDSUS-7183 Fixed a problem in which applications could not connect to the master node even after restoration, when both slave node and ping node are unable to establish connection in the HA environment where the ping node is configured

Previously, applications could not connect to the master node even after a connection was restored, when both slave node and ping node were unable to establish connection in the HA environment where a ping node was configured with the ha_ping_hosts parameter. If the master node was not connected to a ping node in the HA structure where a ping node existed, the structure assumed that the master server could not maintain its role; thus, its role was changed to "slave" and the applications could be used as read-only.

CUBRIDSUS-6956 Modified the program so that it now retries replication when a certain error occurs in the HA environment

Modified the program so that it now retries replication when a certain error occurs in the HA environment.

The ha_applylogdb_ignore_error_list and ha_applylogdb_retry_error_list parameters are added to ignore an error that occurs in the ha_applylogdb_ignore_error_list parameter and proceed with replication reflection; in addition, it has been modified to retry the corresponding task if the configured error occurs in the ha_applylogdb_retry_error_list.

CUBRIDSUS-6918 Fixed a problem in which data would not display when scanned with an descending index for a single column that is specified as the primary key in the HA environment

Previously, data would not be displayed when they were scanned with a descending index for a single column that was specified as the primary key in the HA environment. This problem has been solved.

```
CREATE TABLE t (id INT PRIMARY KEY, name CHAR(10));
CREATE INDEX ridx1 ON t(id DESC);
INSERT INTO t VALUES(1, '1');
SELECT * FROM t WHERE id = 1 USING INDEX ridx(+);
```

CUBRIDSUS-6717 Fixed a problem in which a server process would be abnormally terminated when copylogdb requested an old achieve log that has already been deleted in the HA environment

Previously, if the value of system parameter force_remove_archives was set to yes, the server process (cub_server) could delete a transaction log that copylogdb did not copy. This might cause a problem in which cub_server would be abnormally terminated when copylogdb requested a transaction log. This problem has been solved. In this case, cub_server is not terminated. It returns an error to copylogdb.

CUBRIDSUS-7163 Fixed a problem in which replication integrity would be compromised if a failover occurs while delaying the replication in the HA environment

In the HA environment, there are two types of processes in a replica node: a applylogdb process that reflects a replication log of the master node and a applylogdb process that reflects a replication log of slave nodes. Previously, replication integrity would be compromised if a failover occurred while delaying the replication these tow applylogdb(s) reflect the replication log simultaneously in the HA environment. This problem has been solved.

CUBRIDSUS-7164 Modified to copy all previously created logs when the master server process is normally terminated in the HA environment

Modified to copy all the logs created before a process is terminated, even if the HA log replication mode is ASYNC when the master server process is normally terminated in the HA environment.

Note that the system will forcibly terminate the server process if it is still active one minute after the normally expected service end time due to a delay caused by an excessive amount of logs to be copied. This may cause a DB restoration process, consequently delaying the service restart.

CUBRIDSUS-6535 Fixed a problem in which 'ALTER TABLE ... ADD COLUMN reserved word' statement would not replicate in the HA environment

Fixed a problem in which the ALTER TABLE statement would not replicate when the column name was a reserved word in the HA environment.

```
ALTER TABLE "content" ADD COLUMN "size" smallint DEFAULT 1
```

CUBRIDSUS-7387 Fixed a problem in which the log page without applylogdb is read infinitely if the ID value of the log page exceeds the maximum value of 32-bit integer type in the HA environment

Fixed a problem in which the log page without **applylogdb** is read infinitely if the ID value of the log page exceeds the maximum value of 32-bit integer type in the HA environment.

4. What's New in CUBRID 2008 R4.1 Patch 1

New Features

CUBRIDSUS-6070 Added a feature that configures logs for debugging in connection URLs

A function was added to handle exceptions as the **logOnException** property of JDBC connection URLs and to write the **logSlowQueries** and **slowQueryThresholdMills** properties to the log file.

```
url = "jdbc:cubrid:localhost:33000:demodb:::?logOnException=true&logSlowQueries=true&slowQueryThresholdMillis=1000"
```

Enhanced Performance

CUBRIDSUS-6648 Improved performance by allowing PREPARE and EXECUTE to be executed with a single call, and reduced the number of message transmissions required when calling the exit function of a request handle in the CCI and PHP interfaces

Added cci_prepare_and_execute() to the CCI and PHP interfaces so that a single call would execute both PREPARE and EXECUTE, and made it available to the cubrid_query() and cubrid_execute() of the PHP interface. Note that cubrid_execute() uses cci_prepare_and_execute() only when it passes the connection object and query as the input parameters.

cci_close_req_handle() has been modified to reduce the number of message transmissions by sending the exit request and the request handle number to the next PREPARE, instead of CAS. This modification also effectively eliminated the case of returning the -4 (COMMUNICATION ERROR) error when it calls cci_close_req_handle() after CAS has been restarted with the auto-commit mode ON.

A test performed on SELECT statement-heavy applications showed that performance has increased by approximately 45% compared to the previous version.

CUBRIDSUS-6906 Improved the performance of JDBC drivers

Improved performance by modifying the JDBC drivers to collect header information in a buffer and send them in a single attempt; it results in improving the performance of JDBC drivers in CUBRID 2008 R4.1 version.

Changed Features

CUBRIDSUS-6678 Fixed a problem in which the data volume of a database would crash when restarting the database server process

Previously, the data volume of a database would crash being caused by an abnormal operation during performing recovery when restarting the database server process or the server process would abnormally terminate. These problems have been fixed.

CUBRIDSUS-6881 Fixed an error that would occur when executing INSERT in IFNULL, NVL, NVL2, or COALESCE function using host variables

Previously, executing the INSERT statement in the IFNULL, NVL, NVL2, or COALESE function using host variables would cause the "ERROR: Execute: Cannot evaluate ' ifnull(?:0 , 1)'" error. This error has been fixed.

CUBRIDSUS-6689 Fixed the 'unknown opcode' error that would occur when using a unary operator after a NOT operator

Previously, using a unary operator after a NOT operator would cause the "ERROR: 'unknown opcode' operator is not defined on types integer and integer." error. This problem has been fixed.

```
SELECT * FROM tab WHERE NOT - col0 = - col0;
```

CUBRIDSUS-6778 Fixed a problem in which live lock would occur in the process of acquiring and releasing the key lock during an index scan

Previously, livelock getting stuck in an infinite loop which releases and acquires lock during an index scan, would occur. This problem has been fixed.

CUBRIDSUS-6914 Fixed a problem in which the application would be abnormally terminated when calling the exit function while accessing the database by using DATASOURCE

Previously, the application would be abnormally terminated when generating DATASOURCE by using the cci_datasource_create() and call cci_disconnect() with it while accessing the database. This problem has been fixed.

CUBRIDSUS-6925 Fixed a cci_set_query() behavior error

Previously, the query timeout would not behave as intended when setting the timeout by using cci_set_query_timeout(). This problem has been fixed.

CUBRIDSUS-6958 Fixed a problem in which a CCI interface-based application would not attempt to reestablish connection with the priority host when it has been failed back

Previously, a CCI interface-based application would not attempt to reestablish connection with the host configured before the **althost** of the connection URL when it has been failed back, preventing it from reconnecting to the master node. This problem has been fixed.

```
cci:cubrid:10.0.0.1:30000:demodb::?althosts=10.0.0.2:30000
```

CUBRIDSUS-6753 Fixed the 'Unknown key' error that would occur when separating or merging B+ tree nodes while executing INSERT, DELETE, or UPDATE in an index

Previously, separating or merging B+ tree nodes while executing INSERT, DELETE, or UPDATE in an index would cause the "Unknown key 'abc10d5d-652a-4ab8-b546-589603b8cde8' referenced in B+tree index {vfid: (40, 3), rt_pgid: 2520, key_type: character varying}" error. This problem has been fixed.

CUBRIDSUS-6897 Fixed a problem in which a connection error would occur when a query is being performed more than 5 seconds in an environment in which multiple threads are connected to more than two brokers

Previously, the -4 (COMMUNICATION ERROR) error would occur when a query is being performed more than 5 seconds when multiple threads were connected to more than two brokers in CCI interface-based applications. This problem would consequently cause an abnormal termination of the application or an unnecessary allocation of more than 1G of memory. This problem has been fixed.

CUBRIDSUS-6779 Fixed a problem in which the status information of the application and CAS would not match when they have just been connected to each other

Previously, the transaction status information (STATUS) of CAS would not be changed from CLIENT_WAIT to CLOSE_WAIT when checking it by using the **cubrid broker status** command immediately after the application is connected to CAS. This problem has been fixed.

CUBRIDSUS-6754 Fixed a problem in which the broker would fail to initialize when the number of CASs is set to 100

Previously, the broker would fail to initialize when the values of **MIN_NUM_APPL_SERVER** and **MAX_NUM_APPL_SERVER**, both of which were parameters determining the number of CASs, were set to 100. This problem has been fixed.

CUBRIDSUS-6364 Fixed a problem in which CAS could hang while writing an SQL log

Fixed a problem in which CAS would hang while writing an SQL log.

CUBRIDSUS-6889 Fixed a problem in which the HA replication reflection process would not replicate the source

Previously, the HA replication reflection process would not replicate the source when the commit list of **applylogdb** was incorrect. This problem has been fixed.

CUBRIDSUS-6905 Fixed a problem in which the CAS and application would hang when using collection data types in JDBC

Previously, the CAS and application would hang when using collection data types such as SET, MULTISET, or LIST in JDBC. This problem has been fixed.

5. What's New in CUBRID 2008 R4.1

New Features

CUBRIDSUS-5860 Added the ADDTIME function

Added the ADDTIME function that is used to add the time value to the date/time type value.

```
SELECT ADDTIME('09:00:13 am', '9:33:17')
  '06:33:30 PM'
```

CUBRIDSUS-6265 Added the ASCII function

Added the ASCII function that is used to convert a character into the corresponding ASCII value.

```
SELECT ASCII('a');
97
```

CUBRIDSUS-5860 Added the BIN function

Added the BIN function that is used to convert a numeric value into a binary character string.

```
SELECT BIN(12);
'1100'
```

CUBRIDSUS-6233 Added the CONV function

Added the CONV function that is used to convert the number system used to represent a numeric value in character string into a different one.

```
// The following example shows how to convert hexadecimal 'a' into binary.
SELECT CONV('a',16,2);
'1010'
```

CUBRIDSUS-5860 Added the FIND_IN_SET function

Added the FIND_IN_SET function that is used to get the number of specified characters from a comma-separated character set as a return value.

```
SELECT FIND_IN_SET('1', '1,4,1,8');
 2
```

CUBRIDSUS-6233 Added the HEX function

Added the HEX function that is used to convert the ASCII code of each character in a character string into a 2-digit hexadecimal character string, or a decimal number into the corresponding hexadecimal number.

```
SELECT HEX('abc'), HEX(255);
'616263' 'FF'
```

CUBRIDSUS-6190 Added functions that is used to get the current serial value and the serial value that will occur after a designated number of serial values as a return

Added the SERIAL_CURRENT_VALUE (serial_name) function that is used to get the current serial value as a return and the SERIAL_NEXT_VALUE (serial_name, number) function that is used to get the resulting value of a specified increment of the serial value as a return.

SERIAL_CURRENT_VALUE(serial_name) works in the same manner as serial_name.current_value.

CUBRIDSUS-5903 Added an operator for regular expressions

Added the REGEXP operator for regular expressions. The patterns used in REGEXP are not case sensitive; to enable case sensitive, you should use REGEXP BINARY.

```
SELECT name FROM athlete where name REGEXP '^[a-d]';

name
======================
'Dziouba Irina'
'Dzieciol Iwona'
'Crosta Daniele'
'Bukovec Brigita'
'Abdullayev Namik'
```

CUBRIDSUS-5616 Added the broker parameter that is used to limit the query execution time

The **MAX_QUERY_TIMEOUT** parameter is added to set a query timeout value in order to limit the query execution time of an application connected to the broker. This parameter can be dynamically changed using the broker_changer utility, and the range of the value is 0-86,400 seconds (1 day).

CUBRIDSUS-6197 Added a function that sets the query execution timeout for the JDBC connection URL

A function has been added to set the query execution timeout of the JDBC connection URL using the **queryTimeout** property(default: 0 sec., infinite). This value can be modified using the DriverManager.setQueryTimeout() method.

```
url = "jdbc:cubrid:localhost:33000:demodb:::?rctime=600&queryTimeout=1"
```

CUBRIDSUS-6198 Added a function that configures database connection timeout value in JDBC connection URL

Added a function that configures database connection timeout value in seconds by using the **connectTimeout** property in JDBC connection URL (default value: 0). This value also can be configured by the DriverManger.setLoginTimeout() method; however, if this value is configured in connection URL, the value configured by the method is ignored.

```
url = "jdbc:cubrid:localhost:33000:demodb:::?rctime=600&connectTimeout=5"
```

CUBRIDSUS-5388 Added the transaction start time for each CAS when outputting the broker status information

Added the transaction start time to be used when outputting the status of CAS by using the **cubrid broker status -f** command.

CUBRIDSUS-6199 Added the number of CAS access from applications, and the number of CAS restarts

Added the number of CAS access from applications, and the number of CAS restarts. It is now possible to increase the value of the **APPL_SERVER_MAX_SIZE** broker parameter, which specifies the maximum memory usage of CAS, when the number of CAS restarts is high.

CUBRIDSUS-6128 Added a function that logs the lock-related information to the server error log file when a deadlock occurs, when the error severity is NOTIFICATION

A function has been added to log the lock-related information to the server error log file when a deadlock occurs while **error_log_level**, the system parameter that configures the error severity, is set to NOTIFICATION.

In the error log file information below, (1) indicates the name of a table that caused the deadlock and (2) indicates the name of an index.

```
demodb_20111102_1811.err


       ...
     OID = -532| 520| 1
(1) Object type: Index key of class ( 0| 417| 7) = tbl.
     BTID = 0| 123| 530
(2) Index Name : i_tbl_col1
     Total mode of holders = NS_LOCK, Total mode of waiters = NULL_LOCK.
     Num holders= 1, Num blocked-holders= 0, Num waiters= 0
     LOCK HOLDERS:
    Tran_index = 2, Granted_mode = NS_LOCK, Count = 1
    ...
```

Enhanced Performance

CUBRIDSUS-5300 Improved the simultaneous processing performance of transaction logs

For tasks that several applications process simultaneously by using a transaction log buffer, simultaneous processing performance is improved by minimizing the task within the section (critical section) that keeps the lock for sequential processing. Improved performance of INSERT and UPDATE up to 70% in comparison with 2008 R4.0 Patch 2 version while performing test on large amount of transaction.

CUBRIDSUS-5687 Improved write performance by modifying the method used to write the dirty pages of a memory buffer to a disk

The write transaction's performance was improved by changing the method from randomly writing the dirty pages of a memory buffer to a disk to writing them in an arranged manner once a certain number of them have been collected.

CUBRIDSUS-5297 Improved the performance of the functions that are used to get the current date/time information

The performance of the functions that calculate the current date/time information, such as SYSDATE, SYSTIME, SYSDATETIME, and SYSTIMESTAMP, was improved by allowing them to calculate the return value from a value that was requested within a minute, instead of requesting the server for the value every time they are called.

CUBRIDSUS-6570 Improved to optimize the sorting of multiple key range conditions for the ORDER BY column, even if the BETWEEN … AND condition is used

Previously, the multi-range optimization, in which the median values were sorted and the results were collected, was available only for columns of the ORDER BY clause with equal (=) conditions. Now, this feature is available even when the column condition is BETWEEN … AND.

In the example below, if the number of records specified in the LIMIT clause is equal to or less than the value of the multi_range_optimization_limit, multi-range optimization can be performed.

```
CREATE INDEX i_tbl_a_b_c ON tbl(a, b, c);
SELECT a, b, c
FROM tbl
WHERE a='1' AND b IN (1,2,3) AND
c BETWEEN 5 AND 10
USING INDEX i_tbl_a_b_c(+)
ORDER BY c LIMIT 10;
```

CUBRIDSUS-5295 Improved the performance by including the INSERT query in the query execution plan cache

In the previous version, only the SELECT, UPDATE, and DELETE queries were included in the query execution plan cache. Now, the INSERT query is included in the cache as well.

CUBRIDSUS-5850 Reduced the deviation between the maximum value and the minimum value of transaction processing response time when the system load is extremely large

To maintain a certain level of performance during heavy system load, the deviation between the maximum value and the minimum value of transaction processing response time was reduced.

CUBRIDSUS-5824 Improved performance when replication is delayed in the HA environment

Performance is improved by eliminating the overhead from accessing the archive log volume file when a delay of replication occurs in the HA environment.

Changed Features

CUBRIDSUS-5882 Add the '0000-00-00' or '0000-00-00 00:00:00' format as a date/time type

In the previous version, the '0000-00-00 00:00:00' type was allowed only for DATETIME and TIMESTAMP types. In this version, the '0000-00-00' type is supported as a DATE type. In addition, in the previous version, when '0000-00-00 00:00:00' was entered in the DATETIME and TIMESTAMP type, the value was converted to the minimum value and saved. Now, '0000-00-00 00:00:00' is saved as it is.

For this reason, the minimum value of the TIMESTAMP type has been changed from '1970-01-01 00:00:00' (GMT) to '1970-01-01 00:00:01' (GMT).

For more information, see CUBRID SQL Guide > Data Types > Date/Time Type in the manual.

CUBRIDSUS-5296 Modified the query plan query statement that includes SERIAL so that it can be stored in the cache

The query plan query statement that includes SERIAL has been modified so that it can be stored in the cache.

CUBRIDSUS-6138 Fixed a problem in which numerical operation functions would return incorrect query results when the input parameter is a host variable and a character string is bound to it

Previously, numerical operation functions such as ABS, FLOOR, CEIL would return incorrect query results when the input parameter is a host variable and a character string is bound to it. This problem has been fixed.

```
SELECT a FROM tbl WHERE a < ABS( ? );

setString (1, "1.5");
```

CUBRIDSUS-5988 Fixed a problem in which an error would occur when the first parameter of the TO_CHAR function is a host variable and a DATE type value is bound to it

Previously, an error reading "Semantic: Cannot coerce host var to type double" would occur when the first parameter of the TO_CHAR function was a host variable and a DATE type value was bound to it. This problem has been fixed.

```
SELECT * FROM tbl WHERE TO_CHAR(col,'yyyymmdd') = TO_CHAR(? ,'yyyymmdd')
```

CUBRIDSUS-6280 Fixed a problem in which the PM (time) character string would be left unconverted or the '12:xx:xx AM' character string would be erroneously converted to '12:xx:xx PM' when formatting with '%r' format in the STR_TO_DATE function

Previously, in the STR_TO_DATE function, applying the '%r' format to a character string in the 'HH:MM:SS PM' format would not trigger the type conversion; applying the '%r' format to a character string in the '12:xx:xx AM' format would cause it to be converted to '12:xx:xx PM' erroneously. These problems have been fixed.

```
SELECT STR_TO_DATE('11:41:10 PM', '%r');
SELECT STR_TO_DATE('12:41:10 AM', '%r');
```

CUBRIDSUS-6023 Fixed a problem in which a conditional clause in which the left of OR is followed by AND may yield incorrect query results

Previously, a conditional clause that had the left of OR followed by AND was considered false when the condition of AND was false, resulting in incorrect query results. This problem has been fixed.

```
SELECT * FROM tbl WHERE col0 BETWEEN 81 AND 23 OR col0 = 92;
```

CUBRIDSUS-6231 Changed the input-output type conversion rules of SQL functions in which the resultant value is determined by the presence or absence of a NULL input

Changed the input-output type conversion rules of SQL functions in which the resultant value is determined by the presence or absence of a NULL input. Now the convertible type with the highest priority among input arguments becomes the result type. Otherwise, it is VARCHAR.

The result types that can be converted and their priorities are as follows:

- CHAR < VARCHAR
- NCHAR < NCHAR VARYING
- BIT < VARBIT
- SHORT < INTEGER < BIGINT < NUMERIC < FLOAT < DOUBLE
- DATE < TIMESTAMP < DATETIME

Previously, if the COALESCE function could not convert types back and forth with input factor types such as INTEGER, DATE, or TIME, it would produce the "ERROR: 'coalesce ' operator is not defined on types date and time." error. Since CUBRID 2008 R4.1, the function returns a VARCHAR type value instead.

```
CREATE TABLE tbl(a INTEGER, b DATE, c TIME);
INSERT INTO tbl(a) values(1);
INSERT INTO tbl(b) values(SYSDATE);
INSERT INTO tbl(c) values(SYSTIME);
SELECT COALESCE (a, b, c) FROM tbl;
```

CUBRIDSUS-6033 Fixed a problem in which a memory leak would occur when processing a query containing an IN predicate in the WHERE clause by using a covering index

Fixed a problem in which a memory leak would occur when processing a query containing an IN predicate in the WHERE clause by using a covering index. For more information on covering indexes, see the CUBRID SQL Guide > Query Optimization > Using Indexes > Covering Indexes section in the manual.

CUBRIDSUS-6345 Fixed a problem in which an incorrect result might be output for queries with covering indexes and correlated subqueries

Previously, an incorrect result might be output when processing a query with covering indexes and correlated subqueries. This version-specific (CUBRID 2008 R4.0 - CUBRID 2008 R4.0 Patch 2) problem has been solved.

```
CREATE TABLE foo (i int, j int, k int);
CREATE INDEX ON foo (i, j);
SELECT count(*) FROM foo x, foo y WHERE x.i = y.i
AND EXIST(SELECT y.k FROM foo z WHERE z.i = x.i+1 and z.j = y.j+1)
```

CUBRIDSUS-6649 Fixed a problem in which an incorrect result would be output when the LIMIT expression in a join query is optimized.

Previously, in CUBRID 2008 R4.0 and R4.0 Patch 2, an incorrect result would be output when the LIMIT expression in a join query is optimized. This was caused by an erroneous LIMIT optimization of tables both in and outside of the Nested Loop join, consequently underestimating the number of records in those tables. This problem has been fixed.

```
SELECT * FROM x, y WHERE x.id > 10 AND x.id = y.id LIMIT 4;
```

CUBRIDSUS-6322 Fixed a problem in which the server might be abnormally terminated when processing a layered query

Previously, the server might be abnormally terminated when performing a layered query on a table that contains a large number of records. This problem has been solved.

```
SELECT * FROM tree1 CONNECT BY PRIOR id=PARENTID ORDER BY id
```

CUBRIDSUS-6282 Fixed a problem in which the server process would be abnormally terminated when an index-related constraint has been dropped and then recreated in a partition table or a parent table

Previously, the server process would be abnormally terminated when index-related constraints such as PRIMARY KEY, UNIQUE, and FOREIGN KEY are dropped and then recreated in a partition table or a parent table that has a child table. This problem has been solved.

```
CREATE TABLE bar (id int, name VARCHAR(200), PRIMARY KEY(id)) PARTITION BY RANGE(id) (PARTITION aa VALUES LESS THAN(2000), partition bbb VALUES LESS THAN MAXVALUE);
ALTER TABLE bar DROP CONSTRAINT pk_bar_id;
ALTER TABLE bar ADD CONSTRAINT PRIMARY KEY(name, id);
```

CUBRIDSUS-6352 Fixed a problem in which an incorrect result would be output by queries that contain correlated subqueries

Previously, queries containing the correlated subqueries listed below would output incorrect results. This problem has been solved.

```
SELECT i FROM t
    WHERE
```

```
        NOT EXISTS (
              SELECT 1
                FROM (SELECT t1.i
                        FROM t1
                      UNION
                      SELECT t2.i
                        FROM t2
                     ) a
                 WHERE t.i = a.i
        )
ORDER BY 1;
```

CUBRIDSUS-6404 Fixed a problem in which indexes would be erroneously converted while performing the INSERT or UPDATE operation

Previously, one of the following errors would occur when the size of an index exceeded one-eighth of a page as a result of an INSERT or UPDATE operation. This problem has been solved.

```
FATAL ERROR *** file /home1/cubrid/src/storage/btree.c, line 8481 ERROR
CODE = -2 Tran = 1, EID = 4
Internal system failure: no more specific information is available.
```

CUBRIDSUS-5808 Fixed a problem in which a session ID could be referenced by different DB connections

Previously, the value of user-defined variables, **PREPARE** statements, **LAST_INSERT_ID**, or **ROW_COUNT** could be incorrectly assigned due to an error in which a session ID could be referenced by multiple DB connections. This problem has been solved. Please note that this problem may persist in the case of HA failover.

CUBRIDSUS-6054 Modified the system so that it outputs only one instance of "SELECT COUNT (DISTINCT constant) FROM tbl" results

Previously, the result of "SELECT COUNT(DISTINCT function) FROM tbl" would output the total number of records. It now outputs only one instance of the results.

CUBRIDSUS-6370 Fixed a problem in which an error would occur when performing the UPDATE operation on the NULL value existing in the column that consists of multiple column indexes

Previously, performing the UPDATE operation on the NULL value existing in the column that consists of multiple column indexes would produce the error below. This problem has been solved.

```
Unknown key {NULL, '45', NULL, '45'} referenced in B+tree index {vfid: (151, 0), rt_pgid: 550, key_type: midxkey}
```

CUBRIDSUS-6347 Modified the system so that it now increments the ERR-Q of cubrid broker status by one when a PREPARE operation fails

Previously, the ERR-Q of the **cubrid broker status** did not increase when a PREPARE operation failed. Now, the count is correctly increased by one when such an operation fails.

CUBRIDSUS-5879 Enabled auto commit mode to ON by default in the CCI interface

Auto commit mode is now set to ON by default. This mode can be configured by using the **CCI_DEFAULT_AUTOCOMMIT** parameter of the broker in the CCI interface.

CUBRIDSUS-6220 Fixed a problem in which performing the CLOSE operation on a certain query request handler in a CCI interface-based application would affect the commit status of other query request handlers

Previously, performing the CLOSE operation on the req1 of a certain query request handler in a CCI interface-based application would affect the commit status of the req2 of a different query request handler. This problem has been solved.

The following is a case scenario that caused a problem in earlier versions. In this scenario, the system would refer to the req1 and its erroneous auto commit mode at the time of CLOSE(req1) despite auto commit mode being turned off after EXECUTE(req1). This caused req2 to be unexpectedly changed as well.

```
AUTOCOMMIT ON
  req1 = PREPARE
               EXECUTE(req1)
AUCOMMIT OFF
  req2 = PREPARE
                EXECUTE(req2)
CLOSE(req1)
```

CUBRIDSUS-6327 Fixed a problem in which transactions would not be rolled back once an error occurred during the PREPARE function call in an CCI interface-based application

Previously, transactions would not be rolled back once an error occurred during the PREPARE function call in an CCI interface-based application. This problem has been solved.

CUBRIDSUS-6366 Fixed a problem in which a CCI interface-based application for Linux would malfunction when the value of network socket file descriptors exceeds 1,024

Previously, a CCI interface-based application for Linux would malfunction when the value of network socket file descriptors exceeded 1,024. In this condition, the application would end up referring to an incorrect memory address due to the limitation of select(), a function used to implement the CCI interface. For this reason, the application would be abnormally terminated or a communication error would occur. This problem has been solved.

CUBRIDSUS-6491 Fixed a problem in which the cci_datasource_borrow function might immediately return an error without waiting for the connection to be established at the function call

Previously, the cci_datasource_borrow() might immediately return the "All connections are used" error without waiting for the connection to be established at the function call as specified in max_wait. This problem has been solved.

CUBRIDSUS-6606 Fixed a problem in which a memory leak would occur when statement pooling has been set by using the DATASOURCE in the CCI interface

Previously, a memory leak would occur when statement pooling has been set by using the DATASOURCE in the CCI interface. This problem has been fixed.

CUBRIDSUS-6673 CCI Fixed a problem in which an application that was developed with the CCI interface would malfunction when COMMUNICATION ERROR occurs

Previously, in an application that was developed with the CCI interface, "COMMUNICATION ERROR net_read_header()" would cause the application to malfunction. This problem has been solved by adding a functionality that can be used to verify functional errors of the network.

CUBRIDSUS-6318 Fixed a problem in which the BigDecimal class (JAVA) would retrieve an incorrect BIGINT value in JDBC

Previously, the BigDecimal class (JAVA) would retrieve an incorrect BIGINT value in JDBC. This problem has been solved.

```
BigDecimal nInt = rs.getBigDecimal(1);
```

CUBRIDSUS-6290 Removed the cap on the size of data that can be entered with a single request in the JDBC driver

Previously, the size of data that could be entered with a single request (for example, a request made by using methods such as execute or executeBatch in a JAVA application) in the JDBC driver was capped at 100MB. This cap has been removed. In CUBRID 2008 R4.0 Patch 2 and earlier versions, exceeding this cap would result in the following error:

```
Java.lang.ArrayIndexOutOfBoundsException : 1024
At cubrid.jdbc.jci.UOutputBuffer.writeByte(UOutputBuffer.java : 607)
```

CUBRIDSUS-6453 Fixed a problem in which the CSQL interpreter in standalone mode would be abnormally terminated when a PREPARE statement is executed with a user-defined session variable

Previously, the CSQL interpreter in standalone mode (-s) would be abnormally terminated when a PREPARE statement was executed with a user-defined session variable. This problem has been solved.

```
prepare st from 'select ?';
set @a='a';
execute st using @a;
```

CUBRIDSUS-6106 Modified the data buffer size to 16MB when running the addvoldb or spacedb utility in SA mode

Previously, the data buffer size was 512MB when running the **addvoldb** or **spacedb** utility in SA mode. The buffer size is now 16MB.

CUBRIDSUS-6235 Modified for applications such as CUBRID Manager to remain connected even when they have been idle for a long time

If the equipment in which CUBRID broker is installed or the firewall equipment is configured to disconnect the socket connection when no data has been transmitted for a certain amount of time, it takes a long time to reconnect the socket. For this reason, when a query was executed on a CUBRID Manager that had been idle for some time, it took a long time to print the result.

CUBRIDSUS-5472 Fixed a problem in which CSQL or broker CAS (Common Application Server) would be abnormally terminated when an interrupt occurs while processing a query

Previously, when an interrupt (Ctrl+C or query cancel) occurred while processing CSQL that had been executed in client/server mode or a query in the CCI/PHP interface program in which the query execution function (cci_execute()/cubrid_execute()) was set to ASYNC mode, the CSQL or broker CAS was abnormally terminated. This problem has been fixed.

CUBRIDSUS-5816 Modified the size-specifying options of the CUBRID utility and system parameters to accept numbers with decimals

Now size-specifying options of the CUBRID utility and system parameters take numbers with decimals.

```
cubrid createdb --db-volume-size=0.5G --log-volume-size=0.5G testdb
cubrid addvoldb -p data --db-volume-size=0.25G testdb

# cubrid.conf
db_volume_size=0.5G
log_volume_size=0.5G
data_buffer_size=0.5G
log_buffer_size=2.5M
sort_buffer_size=2.5M
```

CUBRIDSUS-6275 Fixed a problem in which indexes would become broken when an interrupt occurred while deleting them

Previously, indexes could be broken when an interrupt such as query timeout occurred while deleting them. This problem has been fixed.

CUBRIDSUS-6418 Fixed a problem in which the server process would hang when performing an online backup and a DDL simultaneously

Previously, the server process would hang when performing an online backup and a DDL simultaneously. This problem has been solved. This problem was reported only in the CUBRID 2008 R4.0 Patch 2 version.

CUBRIDSUS-6279 Fixed a problem in which an existing link of the index leaf node was incorrectly made

Previously, an existing link of the index leaf node was incorrectly made when merging or splitting indexes; this caused wrong results of reverse lookup or dramatic increases of temporary volumes.

This problem has been solved. This problem was reported in 2008 R4.0 GA and later versions.

CUBRIDSUS-6001 Fixed a problem in which the slot was not freed up for other transactions when a preceding transaction had failed to perform the INSERT operation on a specific record and was still progressing

Previously, if a transaction failed to perform the INSERT operation due to reasons such as unique key violation, the slot was locked until the transaction was ended. For this reason, other transactions had to wait for the slot to be unlocked. Now the other transactions occupy a different slot for their own INSERT operations.

As a result of this change, the performance of INSERT in the previously described situation has been partially improved.

CUBRIDSUS-6396 Fixed a problem in which database volumes would be crashed when the UPDATE operation was interrupted

Fixed the error where the database volume would be crashed if it was interrupted while being updated and had a problem restoring the index.

CUBRIDSUS-6720 Fixed a problem in which memory leak happens in server process in the processing of Boolean expressions

Fixed a problem in which memory leak happens in server process in the processing of Boolean expressions.

CUBRIDSUS-6072 Fixed a problem in which the connection would be terminated when no request has been made within a certain amount of time after the transaction is ended and the query request handle is closed in an application

Previously, the connection between CAS and an application would be terminated due to a session timeout when no request has been made within a certain amount of time after the transaction is ended and the query request handle is closed in an application. This problem has been solved.

CUBRIDSUS-6342 Fixed a problem in which database recovery would fail when a checkpoint error abnormally terminates the database

Previously, database recovery would fail when a checkpoint error abnormally terminated the database. This problem has been solved.

```
Unable to mount disk volume "/data/DB/demodb/log/demodb_lgar592".... No such file or directory
```

CUBRIDSUS-6513 Modified the error log of the server process so that it always outputs the name of the source file and the line number in the log

Previously, the server process (cub_server) error log would not always output the name of the source file and the line numbers. Now, it always outputs them.

CUBRIDSUS-6240 Fixed a problem in which only the query finish time would be recorded in the SLOW SQL log, when both query start time and finish time should be recorded

Previously, only the query finish time would be recorded in the SLOW SQL log. Now, both query start time and finish time are recorded in the log.

```
11/29 11:10:43.477 (10) execute srv_h_id 1 select 1 from db_class a, db_class b, db_class c, db_class d
11/29 11:10:45.076 (10) execute error:-4 tuple 0 time 1.634, EID = 8
```

CUBRIDSUS-5956 Provided a solution to prevent CAS memory usage spikes

Now a CAS process that exceeds the specified memory usage limit will be forcibly restarted in order to prevent memory usage spikes.

APPL_SERVER_MAX_SIZE_HARD_LIMIT broker parameter has been introduced to handle the restart. This parameter is used to specify a memory usage limit, which when reached will cause the transactions of any exceeding CAS process to be forcibly rolled back and restarted. The default value of this parameter is 1,024MB. The **APPL_SERVER_MAX_SIZE** broker parameter, which was present in earlier versions, plays a similar role. The only difference is that it prompts, rather than forces, the user to manually commit or roll back instead when a CAS process has exceeded the specified limit.

It is a common practice to configure the **APPL_SERVER_MAX_SIZE_HARD_LIMIT** value to be greater than that of **APPL_SERVER_MAX_SIZE** in order to allow transactions to be rolled back normally when it exceeds the specified memory usage limit.

CUBRIDSUS-5851 Modified replica nodes so that transaction logs can be copied only in ASYNC mode

For replica nodes in the HA environment, the transaction log can be copied in ASYNC mode, regardless of the **ha_copy_sync_mode** parameter of cubrid_ha.conf.

CUBRIDSUS-4817 Fixed a problem in which replication would fail in the HA environment due to insufficient memory when the number of modified records in a single transaction was exceptionally high

Previously, when there were too many modified records in a single transaction in the HA environment , the memory available to the **applylogdb** process that applied the replication logs was insufficient, consequently preventing replication from occurring. This problem has been fixed.

CUBRIDSUS-6117 For 2008 R4.0 Patch 2 and later versions, the performance regression of the applygdb process that was identified in the HA environment has been decreased

The performance regression of the **applylogdb** process in the HA environment, which was identified in 2008 R4.0 Patch2 and later versions, has reduced.

CUBRIDSUS-6172 Fixed a problem in which an archive log file copied to a node that has been switched from a slave to a master is not deleted after a failover in the HA environment

Previously, even when **log_max_archives**, the system parameter that configures the number of archive log files to be maintained in the node that has been switched from a slave to a master, was set, replicated archive log files were not deleted in the HA environment after a failover. This problem has been fixed.

CUBRIDSUS-6183 Fixed a problem in which the applylogdb process would be abnormally executed when it attempted to read and apply the log pages being written from the cache

Previously, a malfunction such as the incorrect application of log pages or abnormal termination of process might occur in the HA environment when the **applylogdb** process tried to apply the log pages that were being written right after the archive log had been created. This problem has been fixed.

CUBRIDSUS-6193 Fixed a problem in which the slave node process could be abnormally terminated when the primary key of the master node has been changed and the slave node is recreated with an online backup in the HA environment

Previously, the slave node process could be abnormally terminated when the user attempted to recreate the slave node with an online backup in the HA environment. This caused the slave node to be abnormally terminated when applying the previous log to the new table with the updated key. This problem has been solved by allowing the system to handle the situation as a log error.

CUBRIDSUS-5475 Modified the behavior of the INCR/DECR function so that it cannot update the data associated with the function call in a slave node in the HA environment

Previously, calling the INCR/DECR functions in a slave node would cause data inconsistency in the master and slave node in the HA environment. Now the data cannot be modified by these functions.

CUBRIDSUS-5939 Provided a feature that is used to set the timeout for normal termination for active transactions when changing the status of a node from 'standby' to 'maintenance' in the HA environment

It is now possible to set the timeout for normal termination for active transactions when changing the status of a node from 'standby' to 'maintenance' by using the **changemode** utility in the HA environment. This can be achieved by using the -t option. The default value is 5 seconds. With this option, the status of transactions that are still in progress when the timeout occurs will become 'maintenance' after a forced termination. Otherwise, the status of transactions immediately becomes 'maintenance' when they are normally terminated.

```
// When an transaction in progress exceeds 10 seconds from the point of executing changemod, the transaction will be rolled back and the node status becomes ‘maintenance’.
cubrid changemode -m maintenance -t 10 demodb
```

CUBRIDSUS-6154 Changed the operation behavior when the value of the system parameter that configures the cycle of saving the dirty pages of a data buffer to a disk is -1

Previously, when **page_flush_interval_in_msecs**, a system parameter that configures the cycle of saving the dirty pages of a data buffer to a disk, is -1, the dirty pages could be saved to the disk only when the checkpoint was reached or the page was swapped. This problem has been fixed, so that now the value of -1 works as if it were 0.

CUBRIDSUS-6133 Changed the system parameter that configures the cycle of detecting the deadlock status to take one decimal place

Changed **deadlock_detection_interval_in_secs**, the system parameter that configures the cycle of detecting the deadlock status, to take one decimal place. The minimum value is changed from 1 second to 0.1 seconds. This value will be rounded up at the second decimal point. For example, the value of 0.12 seconds is rounded up to 0.2 seconds.

CUBRIDSUS-6083 Changed the default value of a system parameter that configures the size and criteria of determining the stack size of a thread

Previously, when the stack size of a thread set in the stack OS was larger than the system parameter (**thread_stacksize**), the size was determined by the set value of the OS. Now, this size is determined by the **thread_stacksize** value, and the default value of the **thread_stacksize** is changed from 100 KB to 1 MB.

CUBRIDSUS-6668 Modified the minimum value of the system parameter that is used to determine the data caching buffer size

The minimum value of **data_buffer_size**, a parameter that is used to determine the size of data caching buffer, has been changed from 64K to 16M.

CUBRIDSUS-6035 Fixed a problem in which files would be incorrectly installed when a CUBRID package for Linux is installed in a directory with a different version of CUBRID had already been installed

Previously, the CUBRID package (.sh) for Linux would incorrectly install its files if a different version of CUBRID existed in the directory. This problem has been solved.

CUBRIDSUS-6061 Fixed a problem in which JDK version 1.7 could not build the JDBC driver source

Previously, JDK version 1.7 could not build the JDBC driver source. This problem has been solved. Please note that an ANT 1.8 or higher version is required to build JDBC driver source with JDK 1.7.

CUBRIDSUS-5942 Modified the CUBRID installer so that it checks whether or not the CUBRID Service Tray can be automatically started after a system reboot during the installation process in the Windows environment

Now, the CUBRID installer checks whether or not the CUBRID Service Tray can be automatically started after a system reboot during the installation process in the Windows environment and outputs a warning message if there is a problem that prevents the system from automatically starting the tray. For CUBRID Service Tray to run automatically after the OS has been rebooted, CUBRID Service Tray must be registered on "Administrative Tools > Task Scheduler" once the Task Scheduler has been started from "Control Panel > Administrative Tools > Services."

CUBRIDSUS-6028 Fixed a problem in which the cubrid utility would not run until the user reboots the system after the installation of a CUBRID package for Windows

Previously, the **cubrid** utility would not run until the user reboots the system after the installation of a CUBRID package for Windows. This was caused by CUBRID-related environmental variables not being applied to the system without a system reboot. This problem has been solved.

CUBRIDSUS-6278 No CRT DLL when executing applications using the Windows CCI library (cascci.dll)

Now executing applications using the CCI library (cascci.dll) for Windows does not require C RunTime DLL.

6. Cautions

New Cautions from 2008 R4.1

CUBRIDSUS-5879 The default value of CCI_DEFAULT_AUTOCOMMIT has been changed to ON since CUBRID 2008 R4.1

The default value for the **CCI_DEFAULT_AUTOCOMMIT** broker parameter, which affects the auto commit mode for application programs developed with CCI interface, has been changed to ON since CUBRID 2008 R4.1. As a result of this change, CCI-based interface(PHP, ODBC, OLE DB etc.) users should check whether or not the application's auto commit mode is suitable for this.

CUBRIDSUS-5238 The database volume of 2008 R4.1 is not compatible with those of 2008 R4.0 Beta and earlier versions

- Database volumes from 2008 R4.0 or later versions are compatible with that of 2008 R4.1.
- 2008 R4.1 is not backward compatible with database volumes of 2008 R4.0. Beta and other earlier versions. For this reason, the database volume must be migrated in order to upgrade the database.
    - 2008 R3.x or earlier version users must use **cubrid unloaddb/loaddb**.
    - 2008 R4.0 Beta users can use the **migrate_r40beta2ga** utility provided at the download page. However, for the volume with page size of less than 4K, the **cubrid unloaddb/loaddb** must be used instead.

For more information, see Database Migration Procedures.

Existing Cautions

CUBRIDSUS-5597 No identifier can be included in the password for CCI, PHP, and JDBC connection URL

? and : that are used as identifiers in CCI, PHP, or JDBC connection URLs cannot be included in the password. The following is an example of a password that is invalid to use as a connection URL because it contains ?:

```
// For CCI and PHP
url = "cci:jdbc:CUBRID:192.168.0.10:33000:tdb:dba:12?:?charset=UTF-8";
// For JDBC
url="jdbc:CUBRID:192.168.0.10:33000:tdb:dba:12?:?charset=UTF-8";
```

Passwords that contain ? or : may be passed as a separate parameter.

```
// For CCI
url="cci:CUBRID:192.168.0.10:33000:tdb:::?charset=UTF-8";
cci_connect_with_url(url, "dba", "12?");
// For PHP
url="cci:CUBRID:192.168.0.10:33000:tdb:::?charset=UTF-8";
cubrid_connect_with_url(url, "dba", "12?");
// For JDBC
url="jdbc:CUBRID:192.168.0.10:33000:tdb:::?charset=UTF-8";
conn = DriverManager.getConnection(url,"dba", "12?");
```

This applies to all versions.

CUBRIDSUS-5136 The page-unit options to be removed

The options (-p, -l, -s), which use page units to specify the database volume size and log volume size of the **cubrid createdb** utility, will be removed. Instead, the new options, added after 2008 R4.0 Beta (--db-volume-size, --log-volume-size, --db-page-size, --log-page-size), are used.

To specify the database volume size of the **cubrid addvoldb** utility, use the newly-added option (--db-volume-size) after 2008 R4.0 Beta instead of using the page unit.

CUBRIDSUS-4222 Cautions for setting the database volume size

From the 2008 R4.0 Beta version, the default value of data page size and log page size in creating the database was changed from 4 KB to 16 KB. If you specify the database volume to the page count, the byte size of the volume may differ from your expectations. If you did not set any options, a 100 MB-database volume with 4 KB-page size was created in the previous version. However, starting from 2008 R4.0, a 512 MB-database volume with 16 KB-page size is created.

In addition, the minimum size of the available database volume is limited to 20 MB. Therefore, a database volume less than this size cannot be created.

CUBRIDSUS-4222 The page-unit system parameter to be removed

It is recommended to use the new system parameters in bytes, because the page-unit system parameters will be removed. For details on the related system parameters, see the below.

CUBRIDSUS-4095 Changed the default values of some system parameters

Starting from 2008 R4.0, the default values of some system parameters have been changed.

Now, the default value of **max_clients**, which specifies the number of concurrent connections allowed by a DB server, and the default value of **index_unfill_factor** that specifies the ratio of reserved space for future updates while creating an index page, have been changed. Furthermore, the default values of the system parameters in bytes now use more memory when they exceed the default values of the previous system parameters per page.

Previous System Parameter	Added System Parameter	Previous Default Value	Changed Default Value (unit: byte)
max_clients	-	50	100
index_unfill_factor	-	0.2	0.05
data_buffer_pages	data_buffer_size	100M (page size=4K)	512M
log_buffer_pages	log_buffer_size	200K (page size=4K)	4M
sort_buffer_pages	sort_buffer_size	64K (page size=4K)	2M
index_scan_oid_buffer_pages	index_scan_oid_buffer_size	16K (page size=4K)	64K

In addition, when a database is created using **cubrid createdb**, the minimum value of the data page size and the log page size has been changed from 1K to 4K.

CUBRIDSUS-5375 Changed so that database services, utilities, and applications cannot be executed when the system parameter is incorrectly configured

A change has been made so that now the related database services, utilities, and applications are not executed when configuring system parameters that are not defined in cubrid.conf or cubrid_ha.conf, when the value of system parameters exceed the threshold, or when the system parameters per page and the system parameters in bytes are used simultaneously.

CUBRIDSUS-4524 Deleted replication feature

Since the replication feature has been deleted from CUBRID 2008 R4.0, you must use the HA feature in order to set up a redundant environment. After performing a server version upgrade and a database migration, you can build a new HA environment. For building an HA environment, see Administrator's Guide > CUBRID HA in the manual.

CUBRIDSUS-5228 Provided a separate CUBRID Manager installation package

Starting from CUBRID 2008 R4.0, the CUBRID Manager installation package is provided separately. To use CUBRID Manager, it must be separately installed after installing the CUBRID package.

CUBRIDSUS-5097 When performing INSERT/UPDATE with strings that are larger than the column size, the strings are truncated

When INSERT/UPDATE is executed to a CHAR, VARCHAR, NCHAR, or VARNCHAR type string that is larger than the column size, the string exceeding the column size is truncated. See CUBRID SQL Guide > Data Types > Character Strings in the manual.

CUBRIDSUS-5349 Database fails to start if the data_buffer_size is configured with a value that exceeds 2G in CUBRID 32-bit version

In the CUBRID 32-bit version, if the value of data_buffer_size exceeds 2G, the running database fails. Note that the configuration value cannot exceed 2G in the 32-bit version because of the OS limit.

CUBRIDSUS-4059 When retrieving values from VARCHAR type columns, trailing spaces are ignored if covering index is applied

When you retrieve values from VARCHAR type columns, trailing spaces are truncated if a covering index is applied, because the query results are retrieved from the index but the values are stored without the trailing spaces in the index when a covering index is applied during the execution of queries. If this is not desired, you just can specify the NO_COVERING_IDX hint. See CUBRID SQL Guide > Query Optimization > Using Indexes > Covering Index in the manual.

```
CREATE TABLE tab(c VARCHAR(32));
INSERT INTO tab VALUES('abcd'),('abcd '),('abcd ');
CREATE INDEX ON tab(c);

-- The query below shows that covering index is applied. In this case, three pieces of data are recognized as if they have the same conditions.

SELECT * FROM tab WHERE c='abcd ' USING INDEX i_tab_c(+);
c
======================
'abcd'
'abcd'
'abcd'
```

CUBRIDSUS-3757 HA-related cautions

If you use triggers and java stored procedures within CUBRID HA, the slave node redundantly executes the triggers or the java stored procedure that has already been executed by the master node. As this may result in data inconsistency among nodes in the CUBRID HA group, you must not use triggers and java stored procedures in CUBRID HA.

Because CUBRID HA synchronizes data among nodes within the CUBRID HA group based on the replication log, if you use a method that does not create a replication log or set a NOT NULL option in the CUBRID Manager, there may be data inconsistency among nodes within the CUBRID HA group. Therefore CUBRID HA cannot use methods and their tasks cannot be performed through the CUBRID Manager.

CUBRIDSUS-5071 LOB type storage is not restored when you backup/restore a database

CUBRID doesn't include the LOB type data in the database backup/restore procedure, because they don't exist in a database volume, but in a separate storage. That is, since the LOB type storage is not backed up together during database backup, LOB type storage is not restored during restoration. LOB type storage must be maintained separately.

CUBRIDSUS-3826 Cautions for stop of GLO class support

CUBRID 2008 R3.0 and earlier versions processed Large Objects with the **GLO** (Generalized Large Object) class, but the **GLO** class has been removed from CUBRID 2008 R3.1 and later versions. Instead, they support BLOB and CLOB (LOB from this point forward) data types. See CUBRID SQL Guide > Data Types > BLOB/CLOB Data Types in the manual.

IT IS RECOMMENDED FOR USERS WITH THE EXISTING GLO CLASSES TO WORK AS FOLLOWS:

After saving GLO data as a file, modify to not use GLO in any application and DB schema.

Migrate the database (Migration from 2008 R3.x and Earlier Versions to 2008 R4.1 in this document).

Perform tasks to load files into LOB data according to the modified application.

Verify the application that you modified operates normally.

For reference, if the **cubrid loaddb** utility loads a table that inherits the GLO class or has the GLO class type, it stops the data from loading by displaying an error message, "Error occurred during schema loading".

With the discontinued support of GLO class, the deleted functions for each interface are as follows:

Interface	Deleted Functions
CCI	cci_glo_append_data cci_glo_compress_data cci_glo_data_size cci_glo_delete_data cci_glo_destroy_data cci_glo_insert_data cci_glo_load cci_glo_new cci_glo_read_data cci_glo_save cci_glo_truncate_data cci_glo_write_data
JDBC	CUBRIDConnection.getNewGLO CUBRIDOID.loadGLO CUBRIDOID.saveGLO
PHP	cubrid_new_glo cubrid_save_to_glo cubrid_load_from_glo cubrid_send_glo

CUBRIDSUS-4172 Constraints when using the BLOB and CLOB types

Be careful when using the BLOB and CLOB types (hereinafter, LOB) because there are some constraints, which are as follows:

- You cannot perform comparison operations (=, <>, IN, NOT IN, etc.) between LOB type columns. To perform such an operation, the type must be converted to a string or a bit array. However, IS NULL and IS NOT NULL are supported.
- You cannot define the PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL constraints or an index.
- When creating or editing a table, you cannot define the SHARED property for them, and can define the DEFAULT property only for NULL values.
- Because the location of a file (LOB Locator) is saved into a database and the data itself is saved into a file, an error may occur because the mapping between the LOB Locator and the LOB data is not valid when recovering to a specific time after a fault.
- When deleting a column with the ALTER TABLE DROP statement or deleting a table with the DROP TABLE statement, only the LOB Locator is deleted, and the LOB file in the external file system to which the LOB column refers is not deleted.
- Consistency is not guaranteed if users directly modify data files of the LOB type without using csql, CUBRID Manager or APIs provided by CUBRID.

See CUBRID SQL Guide > Data Types > BLOB/CLOB Data Types in the manual.

CUBRIDSUS-4186 Recommendations for controlling services with the CUBRID utility in Windows Vista and higher

To control services using **cubrid** utility from Windows Vista and higher, it is recommended to start the command prompt window with administrative privileges.

If you don't start the command prompt window with administrative privileges and use the **cubrid** utility, you can still execute it with administrative privileges through the User Account Control (UAC) dialog box, but you will not be able to verify the resulting messages.

The procedures for starting the command prompt window as an administrator in Windows Vista and higher are as follows:

- Right-click [Start > All Programs > Accessories > Command Prompt].
- When [Execute as an administrator (A)] is selected, a dialog box to verify the privilege escalation is activated. Click "YES" to start with administrative privileges.

CUBRIDSUS-3217 Specifying a question mark when entering connection information as a URL string in JDBC

When entering connection information as a URL string in JDBC, property information would be applied even if you did not enter a question mark (?) in the earlier version. However, from CUBRID 2008 R3, you must include a question mark according to the grammar. Otherwise, an error will be displayed. In addition, you must specify colon (:) even if there is no username or password in the connection information.

```
URL=jdbc:CUBRID:127.0.0.1:31000:db1:::althosts=127.0.0.2:31000,127.0.0.3:31000 - handling as an error
URL=jdbc:CUBRID:127.0.0.1:31000:db1:::?althosts=127.0.0.2:31000,127.0.0.3:31000 - handling as a normal operation
```

CUBRIDSUS-3564 Port configuration is required if the protocol between the master and server processes is changed, or if two versions are running at the same time

Because the communication protocol between a master process (**cub_master**) and a server process (**cub_server**) has been changed, the master process of CUBRID 2008 R3.0 or later cannot communicate with the server process of a lower version, and the master process of a lower version cannot communicate with a server process of 2008 R3.0 version or later. Therefore, if you run two versions of CUBRID at the same time by adding a new version in an environment where a lower version has already been installed, you should modify the cubrid_port_id system parameter of cubrid.conf so that different ports are used by the different versions.

CUBRIDSUS-2828 @ cannot be included in the database name

If @ is included in a database name, it can be interpreted that a host name has been specified. To prevent this, a revision has been made so that @ cannot be included in a database name when running **cubrid createdb**, **cubrid renamedb** and **cubrid copydb** utilities.

CUBRIDSUS-3267 Caution when setting a directory path in a Windows environment

Note that if a directory path where you want to install CUBRID has any empty space in the Windows environment, you cannot install it successfully. In addition, an empty space cannot be included in the target directory paths such as the **cubrid unloaddb**, the **cubrid loaddb**, and the **cubrid backupdb**.

CUBRIDSUS-3553 A Manager Server process-related error occurs in the execution of the CUBRID source after its build

If users want to build the CUBRID source and install it themselves, they must build and install CUBRID and the CUBRID Manager, respectively. If you check out the CUBRID source only and run **cubrid service start** or **cubrid manager start**, the "cubrid manager server is not installed" error occurs.