CUBRID 9.0 릴리스 노트

1

릴리스 노트 공통

1.1 개정 내역

CUBRID 9.0 버전 릴리스 노트의 변경 사항은 아래와 같다.

작성 날짜	설명
2012년 10월	CUBRID 9.0 Beta 릴리스 (9.0.0.0478)

1.2 참고 문서

CUBRID 9.0 제품과 함께 배포되는 문서는 아래와 같다.

문서	설명
릴리스 노트	CUBRID 릴리스 버전의 특징 및 이전 버전에서 변경된 사항과 관련된 정보를 포함한다.
매뉴얼	CUBRID 소개, 시작, CSQL 인터프리터, SQL 설명서, 관리자 안내서, 성능 튜닝, API 레퍼런스, CUBRID 매니저, CUBRID 매니저 관리자 안내서를 포함한다.

1.3 버그 리포트 및 사용자 피드백 제공 방법

CUBRID 프로젝트에서는 사용자의 거침없는 버그 리포트와 솔직한 피드백을 기다리고 있으며, 아래 사이트에서 등록할 수 있다.

문서	설명
버그 리포트	CUBRID 오픈 소스 프로젝트: http://dev.naver.com/projects/cubrid/issue
사용자 피드백	CUBRID 오픈 소스 프로젝트: http://dev.naver.com/projects/cubrid/forum CUBRID 공식 사이트: http://www.cubrid.com

1.4 라이선스

CUBRID의 서버 엔진에는 GNU GPL v2 or later 가 적용되고 CUBRID 매니저 및 인터페이스(API)에는 BSD 라이선스가 적용된다. 보다 상세한 정보는 CUBRID 공식 사이트의 라이선스 가이드를 참고한다.

1.5 추가 정보

CUBRID에 관한 유용한 정보는 아래 사이트에서 찾을 수 있다.

정보	사이트
CUBRID 제품 정보	http://cubrid.com/zbxe/product
CUBRID 도구 정보	http://www.cubrid.org/wiki_tools
CUBRID 드라이버 정보	http://www.cubrid.org/wiki_apis
CUBRID 라이선스 정보	http://cubrid.com/zbxe/bbs_oss_guide/32249

1.6 드라이버 관련 주의 사항

현재 CUBRID가 지원하는 드라이버들은 JDBC, Node.js, CCI(CUBRID C API), PHP, PDO, Python, Perl, Ruby, ADO.NET, ODBC, OLE DB이며, JDBC와 Node.js, ADO.NET을 제외한 모든 드라이버들은 CCI 기반으로 개발되었으므로 CCI와 관련하여 변경된 사항은 CCI 기반 드라이버들에 영향을 미칠 수 있음에 주의한다.

1.7 버전 명 및 버전 스트링 규약

CUBRID 9.0 이후 버전 명 및 버전 스트링은 다음과 같이 표기한다.

- 버전 명: CUBRID x.x Patch x (major 버전, minor 버전, 필요한 경우에 patch 버전을 표기)
  - CUBRID 9.0 Patch 1 (또는 줄여서 CUBRID 9.0 P1)
- 버전 스트링: x.x.x.x (major 버전, minor 버전, patch 버전, build 번호를 표기)
  - 9.0.1.0012

CUBRID 9.0 이전 버전의 버전 명과 버전 스트링은 다음과 같이 표기한다.

- 버전 명: CUBRID 2008 Rx.x Patch x (major 버전을 2008로, minor 버전, patch 버전, build 번호 일부를 표기)
  - CUBRID 2008 R4.1 Patch 1
- 버전 스트링: 8.x.x.x (major 버전, minor 버전, patch 버전, build 번호를 표기)
  - 8.4.1.1001

1.8 기본 설치 및 설정

지원 플랫폼 및 설치 권장 사양

CUBRID 9.0 버전이 지원하는 플랫폼과 설치를 위한 하드웨어/소프트웨어 요구 사항은 아래 표와 같다.

지원 플랫폼	메모리 여유 공간	디스크 여유 공간	필요 소프트웨어
- Windows 32/64 Bit XP, 2003, Vista, Windows7 - Linux 계열 32/64 Bit Linux	2GB 이상	처음 설치 시 약 500MB의 디스크 공간이 필요하며, 하나의 DB를 기본 옵션으로 생성할 경우 약 1.5GB의 디스크 공간이 필요하다.[1]	

1 처음 설치 시 약 500MB의 디스크 공간이 필요하며, 하나의 DB를 기본 옵션으로 생성할 경우 약 1.5GB의 디스크 공간이 필요하다.

지원 플랫폼	메모리 여유 공간	디스크 여유 공간	필요 소프트웨어
kernel 2.4 및 glibc 2.3.4 이상		- JRE 또는 JDK 1.6 이상 Java 저장 프로시저를 사용하는 경우 필요	

2008 R4.0부터는 CUBRID 패키지 설치 시 CUBRID 매니저 클라이언트가 같이 설치되지 않는다. 따라서 CUBRID 매니저를 사용하려면 이를 추가로 설치해야 한다. CUBRID 설치 패키지는 http://ftp.cubrid.org 에서 받을 수 있다.

CUBRID 매니저 및 CUBRID 쿼리 브라우저 설치 패키지와 JDBC, PHP, ODBC, OLE DB 등의 드라이버들도 http://ftp.cubrid.org 에서 받을 수 있다.

CUBRID 엔진, 사용 도구 및 드라이버에 대한 자세한 정보는 http://www.cubrid.org를 참고한다.

버전 호환성과 운용성

응용 프로그램의 호환성

- 이전 버전의 JDBC, PHP, CCI API 등을 사용하는 응용 프로그램은 CUBRID 9.0 DB에 접근할 수 있다. 다만, JDBC, PHP, CCI 인터페이스에 추가/개선된 기능을 사용하기 위해서는 CUBRID 9.0 버전의 라이브러리를 링크하거나 드라이버를 사용해야 한다.
- 새로운 예약어 추가 및 일부 질의에 대한 스펙 변경으로 인해 질의 결과가 이전 버전과 다를 수 있으므로 주의한다. 보다 상세한 사항은 해당 버전 릴리스 노트의 "업그레이드 주의 사항"을 참고한다.
- GLO 클래스를 이용하여 개발된 응용은 BLOB, CLOB 타입에 맞는 응용 및 스키마로 변환하여 사용해야 한다.

CUBRID 매니저의 호환성

- CUBRID 매니저는 CUBRID 2008 R2.1 이상 버전의 서버에 대해서 하위 호환성을 보장하며, 각 서버 버전과 일치하는 CUBRID JDBC 드라이버를 사용한다. 하지만 CUBRID 매니저에서 제공하는 모든 기능을 제대로 사용하기 위해서는 CUBRID 서버 버전보다 높은 버전의 CUBRID 매니저를 사용해야 한다. CUBRID JDBC 드라이버는 CUBRID 설치 시 $CUBRID/jdbc 디렉터리에 포함되어 있다.[2]
- CUBRID 매니저의 Bit 버전과 JRE의 Bit 버전은 서로 동일해야 한다. 예를 들어, 64Bit 버전 DB 서버라도 CUBRID Manager 32Bit 버전을 사용한다면 JRE 또는 JDK 32Bit 버전을 설치해야 한다.
- CUBRID 2008 R2.2 이상 버전의 드라이버는 CUBRID 매니저에 기본으로 내장되어 있으며, cubrid.org에서 별도로 받을 수도 있다.

상호 운용성

- CUBRID DB 서버와 브로커 서버를 분리하여 운영하는 경우, 서버 장비의 운영 체제가 다르더라도 상호 운용성을 보장한다. 단, DB 서버의 Bit 버전과 브로커 서버의 Bit 버전은 서로 동일해야 한다. 예를 들어, Linux용 64Bit 버

---

2 $CUBRID는 CUBRID가 설치된 곳을 지정하는 Linux용 환경변수이며, Windows 환경에서는 %CUBRID%와 같은 형식으로 사용된다.

전 DB 서버는 Windows용 64Bit 버전 브로커 서버와 상호 운용이 가능하지만, 32Bit 버전 브로커 서버와는 상호 운용이 불가능하다.

1.9 설치 방법

제품 설치

Linux용 설치 패키지는 바이너리를 포함하는 스크립트, tar.gz 압축 파일, Linux RPM 패키지 형태로 제공되며, 설치 방법은 매뉴얼의 'CUBRID 시작> 설치와 실행> Linux에서의 설치와 실행'을 참고한다.

Windows용 설치 패키지는 설치 마법사를 이용하여 설치할 수 있다. 설치 방법은 매뉴얼의 'CUBRID 시작> 설치와 실행> Windows에서의 설치와 실행'을 참고한다.

이와 함께 Windows 환경에서 CUBRID 설치 디렉터리 경로에 공백을 포함하는 경우 정상 설치가 되지 않으므로 주의한다. 또한, **cubrid unloaddb**, **cubrid loaddb**, **cubrid backupdb** 등의 작업 대상 디렉터리 경로에도 공백을 포함할 수 없다.

CUBRID 환경 변수 및 OS 환경 변수 설정

CUBRID를 사용하기 위해서는 CUBRID 환경 변수와 관련 OS 환경 변수를 설정해야 한다. (매뉴얼의 'CUBRID 시작 > 환경 변수 설정 및 CUBRID 서비스 시작 > 환경 변수 설정' 참고) 특히, CUBRID 매니저와 Java 저장 프로시저를 사용하기 위해서는 Java 환경 변수를 설정해야 한다. (매뉴얼의 'CUBRID SQL 설명서 > Java 저장 함수/프로시저 > Java 저장 함수/프로시저 사용을 위한 환경 설정' 참고)

1.10 CUBRID 9.0 Beta로 업그레이드하는 방법

CUBRID 2008 R4.1에서 CUBRID 9.0 Beta로 업그레이드하기

CUBRID 2008 R4.1 버전 사용자는 CUBRID 9.0 Beta 버전을 설치한 후 기존의 환경 설정 파일에서 파라미터들의 값을 다음과 같이 변경해야 한다.

cubrid.conf

- single_byte_compare 파라미터는 더 이상 사용하지 않으므로 삭제해야 한다.
- intl_mbs_support 파라미터는 더 이상 사용하지 않으므로 삭제해야 한다.
- lock_timeout_message_type 파라미터는 더 이상 사용하지 않으므로 삭제해야 한다.

cubrid_ha.conf

- ha_apply_max_mem_size 파라미터의 값을 500 이상으로 설정한 사용자는 CUBRID 9.0 Beta 버전 이상으로 업그레이드한 이후 이 값을 500 이하로 설정해야 한다.

CURBID 2008 R4.0 이하 버전에서 CUBRID 9.0 Beta로 업그레이드하기

위의 변경 사항과 함께, CUBRID 2008 R4.1부터 기본값이 변경된 다음 파라미터들을 확인하여 변경한다.

cubrid.conf

- thread_stacksize의 기본값이 100K에서 1M으로 변경되었으므로, 이 값을 설정하지 않은 사용자는 CUBRID 관련 프로세스들의 메모리 사용량을 살펴볼 것을 권장한다.
- data_buffer_size의 최소값이 64K에서 16M으로 변경되었으므로, 이 값을 16M 미만으로 설정한 사용자는 16M 이상으로 설정해야 한다.

cubrid_broker.conf

- CCI_DEFAULT_AUTOCOMMIT의 기본값이 ON으로 변경되었으므로, 이를 설정하지 않은 응용 프로그램 사용자가 기존과 같은 자동 커밋 모드를 유지하고 싶다면 OFF로 설정해야 한다.
- APPL_SERVER_MAX_SIZE_HARD_LIMIT의 최소값이 1024M이며, APPL_SERVER_MAX_SIZE의 값을 설정한 사용자는 APPL_SERVER_MAX_SIZE_HARD_LIMIT의 값을 이보다 크게 설정할 것을 권장한다.

업그레이드 주의 사항

기존 환경 설정 파일 보관

- 이전 버전의 $CUBRID/conf 디렉터리의 환경 설정 파일(cubrid.conf, cubrid_broker.conf, cm.conf)과 $CUBRID_DATABASES 디렉터리의 DB 위치 정보 파일(databases.txt)을 보관한다.[3]

---

3 $CUBRID 또는 $CUBRID_DATABASES 형식의 환경변수는 Linux 환경에서 사용하는 형식이며, Windows 환경에서는 %CU

새로 추가된 예약어 검사

- CUBRID 설치 패키지에 포함 또는 http://ftp.cubrid.org 에서 배포되는 CUBRID 9.0 Beta 버전용 예약어 검출 스크립트인 check_reserved.sql을 이용하여 예약어 사용 여부를 검사할 수 있으며, 예약어로 지정된 식별자를 사용하고 있을 경우 식별자를 수정해야 한다. 매뉴얼의 'CUBRID SQL 설명서 > 식별자'를 참고한다.

DB 마이그레이션

- CUBRID 9.0 Beta는 CUBRID 2008 R4.x 및 그 이전 버전과 DB 볼륨이 호환되지 않으므로, cubrid unloaddb/loaddb 유틸리티를 사용하여 DB를 마이그레이션해야 한다.
- CUBRID 2008 R3.1부터 GLO를 지원하지 않으며 LOB 타입이 GLO 기능을 대체하게 되었으므로, GLO를 이용한 응용 및 스키마는 LOB 타입에 맞게 수정해야 한다. (아래의 GLO 클래스 사용자의 마이그레이션 참고)

복제 또는 HA 환경 재구성

- CUBRID 2008 R4.0부터는 복제 기능을 더 이상 지원하지 않으므로, 이전의 복제 기능을 사용하는 시스템에서는 DB 마이그레이션 이후 HA 환경으로 재구성할 것을 권장한다. 또한, CUBRID 2008 R2.0 및 R2.1에서 제공된 Linux Heartbeat 기반의 HA 기능을 사용하는 시스템도 보다 안정적인 운영을 위해 DB 마이그레이션 이후 CUBRID Heartbeat 기반의 HA 환경으로 재구성해야 한다. (아래의 HA 환경에서 DB 마이그레이션 절차 참고)
- HA 환경 구성은 매뉴얼의 '관리자 안내서 > CUBRID HA'를 참고하여 재설정해야 한다.

DB 마이그레이션 절차

CUBRID 9.0 Beta로 마이그레이션

- 아래의 표와 같이 DB 마이그레이션을 해야 한다.
- 또한, 기존의 GLO 클래스를 사용하고 있는 경우에는 추가 작업이 필요하다. (아래의 GLO 클래스 사용자의 마이그레이션 참고)
- 아래는 cubrid unloaddb/loaddb 유틸리티와 http://ftp.cubrid.org에서 별도 배포되는 check_reserved.sql 예약어 검출 스크립트를 이용하여 마이그레이션을 수행하는 방법이다. (매뉴얼의 '관리자 안내서 > 데이터베이스 마이그레이션' 참고)

단계	Linux 환경	Windows 환경
C1 단계: CUBRID Service 종료	% cubrid service stop	CUBRID Service Tray를 종료한다.
C2 단계: 예약어 검출 스크립트 실행	예약어 검출 스크립트가 위치하는 디렉터리에서 아래 명령을 실행한다. 검출 결과를 확인하여 마이그레이션 진행 또는 식별자 수정 작업을 진행한다. % csql -S -u dba -i check_reserved.sql testdb	
C3 단계:	이전 버전의 databases.txt 및 conf 디렉터리 내 설정 파일을 별도 디렉터리에 보관한다. (C3a) cubrid unloaddb 유틸리티를 실행하고 이때 생성된 파일을 별도 디렉터리에 보관한다. (C3b)	

BRID% 또는 %CUBRID_DATABASES%와 같은 형식으로 사용해야 한다.

단계	Linux 환경	Windows 환경
이전 버전 DB 언로드	% cubrid unloaddb -S testdb 기존 DB를 삭제한다. (C3c) % cubrid deletedb testdb	
		이전 버전의 CUBRID를 제거한다.
C4 단계: 새 버전 설치	"릴리스 노트 공통" > "설치 방법"을 참고한다.	
C5 단계: DB 생성 및 데이터 로딩	DB를 생성할 디렉터리로 이동한 후, DB를 생성한다. (C5a) % cd $CUBRID/databases/testdb % cubrid createdb testdb (C3b)에서 보관한 파일을 가지고 cubrid loaddb 유틸리티를 실행한다. (C5b) % cubrid loaddb -s testdb_schema -d testdb_objects -i testdb_indexes testdb	
C6 단계: 새 버전 DB 백업	% cubrid backupdb -S testdb	
C7 단계: CUBRID 환경 설정 및 CUBRID Service 구동	환경 설정 파일을 수정한다. 이때, (C3a)에서 보관한 이전 버전의 환경 설정 파일을 새 버전에 맞게 수정한다. (시스템 파라미터 설정은 주의 사항 및 관련 매뉴얼 참고) % cubrid service start % cubrid server start testdb	CUBRID Service Tray> [Service Start]를 선택하여 서비스를 시작한다. 명령 프롬프트 창에서 DB 서버를 구동한다. % cubrid server start testdb

GLO 클래스 사용자의 마이그레이션

- GLO 클래스를 사용하는 경우, CUBRID 2008 R3.1부터는 GLO 클래스를 지원하지 않으므로 BLOB 또는 CLOB 타입을 사용하도록 응용과 스키마를 변경해야 한다. 변경 작업이 용이하지 않다면 마이그레이션을 보류할 것을 권장한다.

HA 환경에서 DB 마이그레이션 절차

CUBRID 2008 R2.2 이상 버전에서 CUBRID 9.0 Beta로 HA 마이그레이션

- 아래는 브로커, 마스터 DB, 슬레이브 DB를 각각 별도 서버에 구축한 환경에서 현재 서비스를 중지하고 업그레이드를 수행하기 위한 가이드이다. 서비스 무정지 업그레이드 시나리오는 별도 가이드 문서를 참고한다.

단계	설명
H1~H6 단계: 마스터 노드에서 C1~C6 단계를 수행	마스터 노드에서 CUBRID 업그레이드 및 DB 마이그레이션을 수행하고, 새 버전의 DB를 백업한다.
H7 단계: 슬레이브 서버에 CUBRID 새 버전 설치	슬레이브 서버에서 이전 버전의 DB는 삭제하고, 새 버전을 설치한다.

단계	설명
	설치 방법은 "릴리스 노트 공통" > "설치 방법"을 참고한다.
H8 단계: 마스터 노드 백업본을 슬레이브 서버에서 복구	H6 단계에서 생성된 마스터 노드의 새 버전 DB 백업본(예: testdb_bk*)을 슬레이브 서버에서 복구한다. % 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 단계: HA 환경 재구성 후 HA모드 구동	마스터 및 슬레이브 서버에서 CUBRID 환경 설정 파일(cubrid.conf) 및 HA 환경 설정 파일(cubrid_ha.conf)을 설정한다. (관련 매뉴얼 참고) % 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 마스터 및 슬레이브 서버에서 HA모드로 DB를 구동한다. (관련 매뉴얼 참고) % cubrid heartbeat start
H10 단계: 브로커 서버에 새 버전 설치 및 브로커 구동	설치 방법은 "릴리스 노트 공통" > "설치 방법"을 참고한다. 브로커 서버에 있는 브로커를 시작한다. (관련 매뉴얼 참고) % cubrid broker start

CUBRID 2008 R2.0 또는 R2.1에서 CUBRID 9.0 Beta로 HA 마이그레이션

- CUBRID 2008 R2.0 또는 R2.1의 HA 기능을 사용하는 경우, 서버 버전 업그레이드, DB 마이그레이션을 수행하고 HA 환경을 새롭게 구축한 후 해당 버전에서 사용되었던 Linux Heartbeat 자동 시작 설정을 변경해야 한다. (Linux Heartbeat 패키지가 불필요한 경우 삭제한다.)
- 위의 H1~H10 단계를 수행한 후, 아래의 H11 단계를 수행한다.

단계	설명
H11 단계: 기존 Linux heartbeat 자동 시작 설정 변경	이하의 작업은 마스터 및 슬레이브 서버에서 root 계정으로 수행한다. [root@master ~]# chkconfig --del heartbeat // 슬레이브 서버에서 동일 작업 수행

2

CUBRID 9.0 Beta 릴리스 노트

2.1 개요

릴리스 노트 정보

본 문서는 CUBRID 9.0 Beta (빌드번호 9.0.0.0478)에 관한 유용한 정보를 포함한다. 릴리스 노트의 최신 버전은 CUBRID 릴리스 노트 사이트(http://release.cubrid.org/ko)에서 확인할 수 있다.

CUBRID 9.0 Beta 이전 버전에 대한 **자세한 내용은 CUBRID 2008 R4.1**의 최신 패치 버전 릴리스 노트를 참조한다.

릴리스 특징

CUBRID 9.0 Beta는 다양한 언어의 문자셋과 콜레이션을 지원하여 국제화를 위한 기능을 추가하고, 분석 함수, MERGE 문, DELETE/UPDATE의 JOIN 지원, ENUM 타입 지원 등 다양한 SQL 구문을 추가하여 사용 편의를 강화했다. 함수기반 인덱스, 필터링된 인덱스를 지원하며, 인덱스 스킵 스캔 최적화를 제공한다. 분할을 근본적으로 개선하였으며, 성능과 안정성을 크게 향상시켰다. 아울러 CUBRID SHARD 기능을 통해 대용량 데이터 처리 환경의 편의를 제공한다. 그리고 SysBench 벤치마크 기준으로 처리량(throughput)과 응답 시간 모두 3배 이상 향상되었으며, 기본 SELECT 성능이 약 1.6배 향상되었다. 그 외에 수 많은 크고 작은 버그와 성능 이슈 수정을 통해 제품을 안정화했다.

CUBRID 9.0 Beta 릴리스는 CUBRID 2008 R4.1 및 하위 버전의 수정 사항들을 모두 포함하며, 주요 특징은 다음과 같다.

다국어 지원

현지화에 적합한 데이터베이스 환경을 제공하기 위해 한국어, 영어, 일본어, 중국어, 베트남어, 캄보디아어, 터키어, 독일어, 스페인어, 프랑스어, 이탈리어어 등 다양한 국가 언어의 문자셋, 콜레이션, 캘린더 및 숫자 표기 규칙을 추가했다.

데이터베이스 sharding을 위한 미들웨어 기능 지원

다수의 장비로 수평 분할된 데이터베이스 환경을 용이하게 접근하기 위한 미들웨어인 CUBRID SHARD 기능을 제공한다. CUBRID SHARD 기능은 응용 프로그램이 여러 장비에 분산된 데이터베이스를 하나의 데이터베이스로 보이도록 단일 뷰(single view)를 제공하며, 이들을 인지하고 특정 데이터베이스를 접근할 필요없도록 투명성(transparency)을 제공한다.

OVER 절을 이용한 분석 함수 지원

특정 행 집합에 대해 다양한 통계를 얻을 수 있도록 OVER라는 새로운 분석 절을 함께 사용하는 분석 함수들을 추가했다.

INSERT, UPDATE, DELETE 질의를 하나로 통합하여 수행할 수 있는 MERGE 문 지원

하나 이상의 원본 테이블로부터 하나의 대상 테이블에 데이터를 입력, 수정 또는 삭제할 수 있는 MERGE 문을 추가했다.

UPDATE 문과 DELETE 문의 JOIN 지원

UPDATE 문과 **DELETE** 문에서 **JOIN**을 지원하게 되었다.

ENUM 타입 지원

열거형 문자열 상수들로 정의하는 **ENUM** 타입을 추가했다.

함수 기반 인덱스, 필터링된 인덱스, 인덱스 스킵 스캔 등 다양한 인덱스 기능 지원

인덱스를 구성하는 칼럼에 함수 표현을 포함하는 함수 기반 인덱스 기능을 제공하며, 인덱스 구성에 검색 조건을 포함하는 필터링된 인덱스 기능을 추가했다. 다중 컬럼 인덱스의 첫번째 컬럼이 명시되지 않아도 인덱스의 두 번째 컬럼부터 인덱스를 사용할 수 있는 인덱스 스킵 스캔(index skip scan) 최적화를 제공한다.

분할 기능 안정화, 성능 개선 및 PROMOTE 문 지원

분할 기능의 안정화와 성능 개선을 위해 근본적으로 개선하였다. 질의 컴파일 과정이 아니라 질의 수행 과정에서 분할 프루닝되도록 하여 성능을 개선하였고, 분할 키를 포함하는 기본 키 및 고유 인덱스를 분할 전체에 대해서 유지하지

않고 각 분할별로 유지되도록 변경하였다. 그외에 많은 안정성 개선 및 성능 개선이 있었다.

또한, 분할 테이블에서 특정 분할 부분을 일반 테이블로 승격시키는 **PROMOTE** 문을 추가했다.

HA 안정성 개선 및 운영 편이성 향상

데이터 및 스키마의 HA 복제 불일치 등 여러 안정성 이슈를 해결하였으며, HA 관리 프로세스들을 개별적으로 제어할 수 있도록 하였고 HA 운영 과정에서 노드의 동적 추가 및 삭제를 용이하게 수행할 수 있도록 개선하였다.

DEFALUT에 pseudo column 지원

칼럼의 **DEFAULT** 속성으로 **SYSDATE**, **USER** 등의 시스템 함수를 지정하면 질의 수행 시점을 기준으로 결과 값을 얻어올 수 있도록 했다.

CURSOR HOLDABILITY 지원

CURSOR HOLDABILITY가 기본으로 동작하게 하여 커밋 이후에도 커서가 유지되게 하여 결과 셋을 유지하면서 DML 커밋이 가능하게 했다.

VALUES 절 추가

주로 상수 값으로 구성된 테이블을 표현하기 위해 **VALUES** 문 이하 표현식에 명시된 행 값들을 출력하는 **VALUES** 문을 추가했다.

오류 메시지 개선

오류 발생 위치를 쉽게 찾아갈 수 있도록 오류 메시지를 강화하였고, 구체적이지 않은 오류 메시지들을 개선하여 오류 원인을 찾기 쉽게 했다.

전체 1800 여 개의 크고 작은 버그 및 기능들을 수정 또는 개선

SQL 함수, SQL 구문, 질의 계획, 인덱스, 트리거, 드라이버, 유틸리티 등에서 발견된 크고 작은 버그 및 기능들을 수정 또는 개선했다. 또한 질의 계획, 인덱스 스캔, 잠금, 삽입 및 삭제의 반복 수행, 메모리 누수, 디스크 사용량 등에서 발견된 이슈들을 수정 또는 개선했다. 아울러 HA 스키마 복제 및 데이터 복제 이슈들을 개선 또는 수정했다.

보다 자세한 변경 사항은 아래의 CUBRID 9.0 Beta에서 변경된 사항을 참고한다.

2.2 기능 추가

SQL

MERGE 문

하나 또는 그 이상의 원본으로부터 행들을 선택하여 하나의 테이블 또는 뷰로 갱신이나 삽입을 수행하기 위해 사용하는 MERGE 문을 추가했다. 대상 테이블 또는 뷰에 갱신할지 또는 삽입할지를 결정하는 조건을 지정할 수 있다.

```
MERGE INTO target_table tt USING source_table st
ON (st.a=tt.a AND st.b=tt.b)
WHEN MATCHED THEN UPDATE SET tt.c=st.c
WHEN NOT MATCHED THEN INSERT VALUES (st.a, st.b, st.c);
```

OVER 절을 이용하는 분석 함수(CUBRIDSUS-6112)

행들의 결과에 기반하여 집계 값을 계산하는 분석 함수를 추가했다. 추가된 분석 함수들은 **AVG, COUNT, MAX, MIN, RANK, ROW_NUMBER, STDDEV, STDDEV_POP, STDDEV_SAMP, SUM, VAR_POP, VAR_SAMP, VARIANCE, DENSE_RANK**이다. 분석 함수는 특정 행 집합에 대해 다양한 통계를 허용하기 위해 기존의 집계 함수들 일부에 OVER라는 새로운 분석 절이 함께 사용된다.

다음은 demodb에서 nation_code가 'AU'로 시작하는 국가에 대해 연도 별로 획득한 금메달 수와 해당 연도까지의 금메달 누적에 대한 평균 합계를 출력하는 예제이다.

```
SELECT host_year, nation_code, gold, AVG(gold) OVER (PARTITION BY nation_code ORDER BY
host_year) avg_gold
FROM participant WHERE nation_code LIKE 'AU%';
```

UPDATE JOIN, DELETE JOIN 문(CUBRIDSUS-5646)

하나 이상의 테이블에 대해 갱신하거나 삭제할 수 있는 **UPDATE JOIN, DELETE JOIN** 문을 추가했다.

```
UPDATE a_tbl INNER JOIN b_tbl ON a_tbl.id=b_tbl.rate_id
SET a_tbl.charge = a_tbl.charge * (1 + b_tbl.rate)
WHERE a_tbl.charge > 900.0;

DELETE a, b FROM tbl1 a INNER JOIN tbl2 b
WHERE a.idx=b.p_idx AND b.p_idx=5 AND b.flag=1;
```

ENUM 타입

열거형 문자열 상수들로 정의하는 **ENUM** 타입을 추가했다.

```
CREATE TABLE tbl (
    color ENUM('red', 'yellow', 'blue')
);
INSERT into tbl values ('yellow'), ('red'), (2), ('blue');
SELECT color FROM tbl ORDER BY color ASC;

 color
=====================
  red
  yellow
  yellow
  blue
```

UPDATE 문에 ORDER BY 절 명시(CUBRIDSUS-6605)

UPDATE 문의 **ORDER BY** 절에 따라 순서대로 갱신하는 것이 가능하도록 수정했다. 다음 질의에서 **UPDATE**문을 수행하면 a 칼럼의 값이 큰 순서대로 b의 값이 갱신된다.

```
CREATE TABLE t1 (a int, b int);
INSERT INTO t1 VALUES (1,1),(2,2),(3,3),(4,4);
SET @tmp=100;
UPDATE t1 SET b=(@tmp:=@tmp+1) ORDER BY a DESC;
```

VALUES 절(CUBRIDSUS-7553)

주로 임시로 실제 테이블을 생성하지 않고 상수 테이블을 생성할 때 사용하는 **VALUES** 절을 추가했다. **VALUES** 절은 **UNION ALL** 문을 연결하여 같은 결과를 얻을 수 있지만 훨씬 간편하다. 대개 **SELECT**, **UPDATE**, **DELETE** 질의 내에서 다른 테이블과 조인되는 형태로 많이 활용된다.

```
VALUES (1 AS col1, 'first' AS col2), (2, 'second'), (3, 'third'), (4, 'forth');

SELECT 1 AS col1, 'first' AS col2
UNION ALL
SELECT 2, 'second'
UNION ALL
SELECT 3, 'third'
UNION ALL
SELECT 4, 'forth';
```

SHOW CREATE TABLE 문(CUBRIDSUS-6996)

테이블을 생성하는 SQL 문을 출력해주는 **SHOW CREATE TABLE** 문을 추가했다.

```
SHOW CREATE TABLE tbl;

=== <Result of SELECT Command in Line 1> ===
TABLE CREATE TABLE
=========================================
'tbl' 'CREATE TABLE [tbl] ([id] INTEGER DEFAULT 0 NOT NULL, [phone] CHARACTER VARYING(10),
CONSTRAINT [pk_tbl_id] PRIMARY KEY ([id]))'
```

분할 PROMOTE 문(CUBRIDSUS-7629)

분할 테이블에서 사용자가 지정한 분할 일부를 일반 테이블로 승격시키는 **PROMOTE** 구문을 추가했다.

```
CREATE TABLE t(i int) PARTITION BY LIST(i) (
    partition p0 values in (1, 2, 3),
    partition p1 values in (4, 5, 6),
    partition p2 values in (7, 8, 9),
    partition p3 values in (10, 11, 12)
);

ALTER TABLE t PROMOTE PARTITION p1, p2;
```

테이블, 뷰, 트리거, 저장 함수 및 프로시저의 소유자를 변경하는 ALTER 문 (CUBRIDSUS-7922)

테이블, 뷰, 트리거, 저장 함수 및 프로시저의 소유자를 지정하는 **ALTER** 문을 추가했다.

```
ALTER TABLE test_tbl OWNER TO PUBLIC;
ALTER VIEW test_view OWNER TO PUBLIC;
ALTER TRIGGER test_trigger OWNER TO PUBLIC;
ALTER FUNCTION test_function OWNER TO PUBLIC;
ALTER PROCEDURE test_procedure OWNER TO PUBLIC;
```

CONNECT BY 절에 LEVEL 칼럼 지원(CUBRIDSUS-7526)

CONNECT BY 절에서 **LEVEL** 칼럼을 사용할 수 있게 되었다.

```
SELECT LEVEL FROM db_root CONNECT BY LEVEL <= 10;
```

LIMIT 절에 OFFSET 키워드 지원(CUBRIDSUS-7439)

LIMIT 절에 **OFFSET** 키워드를 사용할 수 있게 되었다. 아래 두 개의 질의는 같은 결과를 출력한다.

```
SELECT * FROM tab LIMIT 2, 1;
SELECT * FROM tab LIMIT 1 OFFSET 2;
```

INET_ATON 함수, INET_NTOA 함수(CUBRIDSUS-8230)

INET_ATON 함수, **INET_NTOA** 함수를 추가했다. **INET_ATON** 함수는 IPv4 주소를 입력하면 숫자 값을 반환하며, **INET_NTOA** 함수는 숫자를 입력하면 IPv4 주소 값을 문자열 형태로 반환한다.

```
SELECT INET_ATON('192.168.0.10');
inet_aton('192.168.0.10')
============================
3232235530

SELECT INET_NTOA(3232235530);
inet_ntoa(3232235530)
=====================
'192.168.0.10'
```

다국어

여러 국가의 문자셋과 콜레이션(CUBRIDSUS-7629)

다국어 지원을 위해 여러 국가의 로캘(문자셋과 콜레이션)을 지원하게 되었다. CUBRID에 추가된 로캘은 영어(en_US), 독일어(de_DE), 스페인어(es_ES), 프랑스어(fr_FR), 이태리어(it_IT), 일본어(ja_JP), 캄보디아어(km_KH), 한국어(ko_KR), 터키어(tr_TR), 베트남어(vi_VN), 중국어(zh_CN)이다.

이와 함께 다음의 다국어 관련 시스템 파라미터들을 추가했다.

파라미터 이름	설명
intl_check_input_string	입력되는 문자열이 사용하는 문자셋에 맞게 입력되는지에 대한 검사 여부를 설정. 기본값 no
string_max_size_bytes	문자열 함수 또는 연산에서 문자열 인자로 사용할 수 있는 최대 바이트 크기를 정의. 기본값 1,048,576 bytes

파라미터 이름	설명
unicode_input_normalization	입력할 유니코드를 결합된 상태로 저장할지 여부를 설정. 기본값 yes
unicode_output_normalization	저장된 유니코드를 분해된 코드로 출력할 것인지 여부를 설정. 기본값 no
use_locale_date_format	문자열을 날짜/시간 형식으로 변환하는 함수에서 입력 인자인 문자열에 대해 지역화된(localized) 날짜/시간 형식을 사용할 것인지 여부를 설정. 기본값 CUBRID_LANG 환경 변수
use_locale_number_format	문자열을 숫자로 또는 숫자를 문자열로 변환하는 함수들에서 입력 또는 출력되는 문자열에 숫자 형식을 부여할 것인지 여부를 설정. 기본값 CUBRID_LANG 환경 변수

또한, 문자셋이 유효한지에 대한 검사 여부를 설정하는 파라미터인 **intl_check_input_string**가 추가되었다. 이와 함께 **single_byte_compare, intl_mbs_support** 파라미터는 더 이상 사용되지 않게 되었다.

Sharding

데이터베이스 sharding을 위한 미들웨어(CUBRIDSUS-4996)

다수의 장비로 횡적 분할된 데이터베이스 환경을 용이하게 접근하기 위한 미들웨어인 CUBRID SHARD 기능을 제공한다. CUBRID SHARD는 다음과 같은 특징을 갖는다.

- 기존 응용의 변경을 최소화하기 위한 미들웨어 형태로서, 흔히 사용되는 JDBC와 CUBRID C API인 CCI 인터페이스를 이용하여 sharding된 데이터베이스를 투명하게 접근할 수 있다.
- 힌트를 이용하여 실제 질의 수행할 shard를 선택하는 방식으로 기존 사용하던 질의에 힌트를 추가하여 사용할 수 있다.
- CUBRID 이외에 MySQL을 backend shard DB로 하여 구성될 수 있다.
- 일부 트랜잭션의 특성을 보장한다.

인덱스

필터링된 인덱스 (CUBRIDSUS-6112)

특정 조건을 포함하는 필터링된 인덱스(filtered index)를 지원한다. 전체 인덱스에서 조건에 부합하는 일부 인덱스만 사용되므로 부분 인덱스(partial index)라고도 하며, 필요한 조건의 행만 인덱스를 생성하므로 인덱스의 갱신 부담이 적고 탐색 범위가 작아 검색 성능 향상에 도움이 된다. 필터링된 인덱스를 적용하여 질의를 처리하기 위해서는 반드시 USING INDEX 절에 해당 필터링된 인덱스를 명시해야 한다.

```
CREATE INDEX bugs_per_dev ON bugs(Author) WHERE Closed = 0;

SELECT * FROM bugs
WHERE Author = 'madden' AND Subject LIKE '%fopen%' AND Closed = 0
USING INDEX idx_open_bugs;
```

이와 함께, 메모리에 캐시하는 필터링된 인덱스 표현식의 최대 개수를 설정하는 **max_filter_pred_cache_entries** 파라미터를 추가했다. 기본값은 1000이다.

함수 기반 인덱스 (CUBRIDSUS-6112)

특정 함수의 결과 값을 기반으로하는 함수 기반 인덱스(function-based index)를 제공한다. 특정 함수를 통해 데이터를 정렬하거나 찾고 싶을 때 사용된다.

```
CREATE INDEX idx_upper_post ON posts_table(UPPER(keyword));
```

인덱스 스킵 스캔 (CUBRIDSUS-5646)

인덱스의 첫 번째 칼럼이 조건에 없어도 뒤따라오는 칼럼이 조건(주로 =)에 있으면 인덱스의 뒷 부분이 사용되는 것을 허용하는 인덱스 스킵 스캔(index skip scan) 최적화를 추가했다.

```
CREATE INDEX idx_t_gen_name on t (gender, name);
SELECT * from t WHERE name = 'SMITH';
```

드라이버

[JDBC] DatabaseMetaData의 메서드인 getDatabaseMajorVersion(), getDatabaseMinorVersion() 추가(CUBRIDSUS-7530)

JDBC DatabaseMetaData 인터페이스의 getDatabaseMajorVersion()과 getDatabaseMinorVersion() 메서드는 각각 메이저 버전 번호와 마이너 버전 번호를 반환한다.

[CCI] 이스케이프 문자열로 변환하는 cci_escape_string 함수(CUBRIDSUS-8940)

CUBRID 질의문에서 사용할 수 있는 이스케이프 문자열로 변환해주는 cci_escape_string 함수를 추가했다.

[CCI] 에러 메시지 출력용 버퍼를 입력 인자로 하는 연결 함수(CUBRIDSUS-5633)

오류 메시지 출력용 버퍼를 입력 인자로 하는 연결 함수인 cci_connect_ex(), cci_connect_with_url_ex()를 추가했다. 기존의 연결 함수는 오류 발생 시 하나의 오류 번호를 반환해서 상세한 오류가 무엇인지 알 수 없었으나, 수정 이후 오류 메시지 버퍼를 통해 상세 오류 번호를 확인할 수 있다.

```
T_CCI_ERROR error;

connection = cci_connect_ex ("localhost", 33000, "demodb", "dba", "pwd", &error);
connection = cci_connect_with_url_ex ("cci:cubrid:localhost:33000:demodb:::", "dba", "pwd",
&error);
```

시스템 카탈로그

HA 카탈로그 테이블에 칼럼 추가 및 삭제(CUBRIDSUS-5456)

복제 로그 재반영으로 인한 복제 불일치를 방지하고 보다 상세한 복제 로그 반영 상태 정보를 제공하기 위해, **db_ha_apply_info** 테이블에 칼럼들을 추가했다.

추가된 칼럼	설명
committed_lsa_pageid	마지막에 반영한 commit log lsa의 page id applylogdb 가 재시작해도 last_committed_lsa 이전 로그는 재반영하지 않음
committed_lsa_offset	마지막에 반영한 commit log lsa의 offset applylogdb 가 재시작해도 last_committed_lsa 이전 로그는

추가된 칼럼	설명
	재반영하지 않음
committed_rep_pageid	마지막 반영한 복제 로그 lsa의 pageid 복제 반영 지연 여부 확인
committed_rep_offset	마지막 반영한 복제 로그 lsa의 offset. 복제 반영 지연 여부 확인
append_lsa_page_id	마지막 반영 당시 복제 로그 마지막 lsa의 page id 복제 반영 당시, applylogdb 에서 처리 중인 복제 로그 헤더의 append_lsa 를 저장 복제 로그 반영 당시의 지연 여부를 확인
append_lsa_offset	마지막 반영 당시 복제 로그 마지막 lsa의 offset 복제 반영 당시, applylogdb 에서 처리 중인 복제 로그 헤더의 append_lsa 를 저장 복제 로그 반영 당시의 지연 여부를 확인
eof_lsa_page_id	마지막 반영 당시 복제 로그 eof lsa의 page id 복제 반영 당시, applylogdb 에서 처리 중인 복제 로그 헤더의 eof_lsa 를 저장 복제 로그 반영 당시의 지연 여부를 확인
eof_lsa_offset	마지막 반영 당시 복제 로그 eof lsa의 offset 복제 반영 당시, applylogdb 에서 처리 중인 복제 로그 헤더의 eof_lsa 를 저장 복제 로그 반영 당시의 지연 여부를 확인
final_lsa_pageid	applylogdb 에서 마지막으로 처리한 로그 lsa의 pageid 복제 반영 지연 여부 확인
final_lsa_offset	applylogdb 에서 마지막으로 처리한 로그 lsa의 offset 복제 반영 지연 여부 확인
required_page_id	log_max_archives 파라미터에 의해 삭제되지 않아야 할 가장 작은 log page id, 복제 반영 시작할 로그 페이지 번호
required_page_offset	복제 반영 시작할 로그 페이지 offset
log_commit_time	마지막 commit log 의 반영 시간

삭제된 칼럼은 다음과 같다.

삭제된 칼럼	설명
page_id	슬레이브 DB에 커밋된 복제 로그의 page
offset	슬레이브 DB에 커밋된 복제 로그의 offset

인덱스 카탈로그 테이블 및 뷰에 칼럼 추가

필터링된 인덱스 및 함수 기반 인덱스 기능이 추가됨에 따라 다음 카탈로그 테이블 및 뷰에 칼럼이 추가되었다.

_db_index 카탈로그 테이블에 다음과 같은 칼럼들이 추가되었다.

추가된 칼럼	설명
filter_expression	필터링된 인덱스의 조건
have_function	함수 기반 인덱스이면 1, 그렇지 않으면 0

DB_INDEX 카탈로그 뷰에 다음과 같은 칼럼들이 추가되었다.

추가된 칼럼	설명
filter_expression	필터링된 인덱스의 조건
have_function	함수 기반 인덱스이면 'YES', 그렇지 않으면 'NO'

_db_index_key 카탈로그 테이블에 다음 칼럼이 추가되었다.

추가된 칼럼	설명
func	함수 기반 인덱스의 함수 표현식

DB_INDEX_KEY 카탈로그 뷰에 다음 칼럼이 추가되었다.

추가된 칼럼	설명
func	함수 기반 인덱스의 함수 표현식

콜레이션 카탈로그 테이블 및 뷰 추가

다국어 지원에 따른 콜레이션 기능이 추가됨에 따라 다음 카탈로그 테이블 및 뷰가 추가되었다.

_db_collation 카탈로그 테이블이 추가되었다.

추가된 칼럼	설명
coll_id	콜레이션 ID
coll_name	콜레이션 이름
charset_id	문자셋 ID
built_in	제품 설치 시 콜레이션 포함 여부 (0: 포함 안 됨, 1: 포함)
expansions	확장 지원 여부(0: 지원 안함, 1: 지원)
contractions	확장 지원 여부(0: 지원 안함, 1: 지원)
checksum	콜레이션 파일의 체크섬
uca_strength	가중치 세기(weight strength)

DB_COLLATION 카탈로그 뷰가 추가되었다.

추가된 칼럼	설명
coll_id	콜레이션 ID
coll_name	콜레이션 이름

추가된 칼럼	설명
charset_name	문자셋 이름
is_builtin	설치 시 제품 내 포함 여부
has_expansions	확장 포함 여부
contractions	축소 포함 여부
uca_strength	가중치 세기(weight strength) (NOT APPLICABLE, PRIMARY, SECONDARY, TERTIARY, QUATERNARY, IDENTITY, UNKNOWN)

기타

클라이언트/서버 시스템 파라미터 중 일부에 세션 파라미터 개념(CUBRIDSUS-8193)

클라이언트/서버 시스템 파라미터들 중 일부에 대해 어느 한쪽의 값이 변경되면 서버와 클라이언트에 같은 값이 반영되게 하는 세션 파라미터 개념을 추가했다. 추가된 세션 파라미터는 **default_week_format, string_max_size_bytes, return_null_on_function_errors, alter_table_change_type_strict, plus_as_concat, compat_numeric_division_scale, use_locale_number_format, use_locale_date_format**이다.

실행 통계 정보에 다중 범위 스캔 최적화 수행 회수 항목(CUBRIDSUS-6163)

cubrid statdump 유틸리티와 **SHOW EXEC STATISTICS ALL**를 통해 확인할 수 있는 실행 통계 정보에 다중 범위 스캔 최적화(multi-range scan optimization)를 수행한 회수 항목을 추가했다.

2.3 동작 변경

본 장에서는 CUBRID 9.0에서 동작이 변경되어 기존 버전과 행위(behavior)가 달라진 이슈들을 설명한다. 대부분의 경우는 기존 버전의 오류나 애매한 동작을 수정하였기 때문에 달라진 것들이거나 기본 설정이 변경된 것들인데, 기존 버전과 같은 동작을 보이지 않는다는 측면에서 주의가 필요하다. 특히 브로커 응용 서버, 브로커 서버 및 CCI/JDBC 드라이버의 오류 번호 범위가 변경되었기 때문에 응용 프로그램에서 오류 이름이 아닌 오류 번호를 직접 사용한 경우에는 응용 프로그램 수정이 필요하다.

SQL

인덱스 추가, 삭제, 변경 시에 인덱스 이름을 명시하도록 변경(CUBRIDSUS-7761)

인덱스 추가, 삭제, 변경 시에 인덱스 이름을 반드시 지정하도록 변경했다. 인덱스 이름을 생략하면 오류가 발생된다.

칼럼의 DEFAULT 값으로 SYSDATE, USER 등의 함수를 지정하면 질의 수행 시점을 기준으로 결과 값을 얻어오도록 변경(CUBRIDSUS-4378)

칼럼의 **DEFAULT** 속성 값으로 **SYSTIMESTAMP, SYSDATE, SYSDATETIME, USER** 함수를 지정했을 때 이전 버전에서는 **DEFAULT** 값이 테이블 생성 시점의 함수 결과 값으로 고정되었으나, 질의가 수행될 때마다 결과 값을 계산하여 얻어오도록 변경했다.

```
CREATE TABLE t (ID int, col TIMESTAMP DEFAULT SYSTIMESTAMP);
ALTER TABLE t add column (uid STRING DEFAULT USER);
INSERT INTO t(ID) VALUES(1); -- col의 값은 질의 수행 시점에 SYSTIMESTAMP를 수행한 결과 값이 된다.
```

MINUTE, HOUR, SECOND, TIME_TO_SEC 함수들의 입력 인자가 시간 형태의 문자열이 아닌 경우를 허용하지 않도록 수정(CUBRIDSUS-7535)

MINUTE, HOUR, SECOND, TIME_TO_SEC 함수들과 같이 TIME 타입이 입력 인자인 경우 "YYYY-MM-DD", "MM/DD/YYYY"와 같은 날짜 형식의 문자열은 허용하지 않도록 수정했다.

```
// 수정 이후 아래와 같은 질의는 허용되지 않는다.
SELECT TIME_TO_SEC('2010-01-01');
CREATE TABLE foo(col TIME DEFAULT '2000-01-01');
```

TIME 타입의 값으로 유효하지 않은 시간을 바인딩하면 오류를 출력하도록 수정(CUBRIDSUS-7159)

TIME 타입의 값으로 "00:00:-1"과 같이 유효하지 않은 시간을 바인딩하면 -1을 무시하고 "00:00:00"으로 받아들였으나, 오류를 출력하도록 수정했다.

시리얼의 시작 값을 변경하면 수정된 시작 값부터 반환하도록 변경(CUBRIDSUS-8157)

시리얼의 시작값을 변경하면 수정된 시작 값 + 1부터 반환했으나 수정된 시작 값부터 반환하도록 변경했다.

```
ALTER SERIAL s1 START WITH 10;
SELECT s1.NEXTVAL;
10
```

칼럼의 숫자 타입에 저장된 숫자 값의 길이보다 작은 길이의 문자 타입으로 칼럼을 변경할 때 그 값이 문자 타입의 지정 길이를 초과하면 해당 길이만큼을 저장하도록 변경(CUBRIDSUS-8009)

칼럼의 숫자 타입에 저장된 숫자 값의 길이보다 작은 길이의 문자 타입으로 **ALTER TABLE ... CHANGE COLUMN...** 을 수행할 때, 해당 칼럼의 값이 문자 타입의 지정 길이를 초과하면 빈 문자열로 처리했으나 해당 길이만큼을 저장하도록 변경했다.

```
CREATE TABLE t1 (i1 INT);
INSERT INTO t1 VALUES (1),(-2147483648),(2147483647),(-2147483648),(2147483647);;
ALTER TABLE t1 CHANGE i1 s1 CHAR(4);
```

내림차순 인덱스 스캔 도중 인터럽트 발생 시 ER_INTERRUPTED 오류를 반환하도록 변경 (CUBRIDSUS-7316)

내림차순 인덱스 스캔 도중 인터럽트 발생 시 ER_DESC_ISCAN_ABORTED 오류를 반환했으나, ER_INTERRUPTED 오류를 반환하도록 변경했다.

SHOW COLUMNS 문의 출력 타입 문자열 변경(CUBRIDSUS-8533)

SHOW COLUMNS 문에서 **STRING**(n), **VARBIT**(n), **VARNCHAR**(n)로 출력되는 타입의 문자열들이 각각 **VARCHAR**(n), **BIT VARYING**(n), **NCHAR VARYING**(n)으로 변경되었다.

SHOW COLUMNS 문 결과 출력 시 특정 타입의 DEFALUT 값 출력에 포함된 홑따옴표 제거(CUBRIDSUS-5921)

테이블 정보를 출력하는 **SHOW COLUMNS** 문 수행 시 **CHAR**나 **DATETIME** 타입 등의 **DEFAULT** 값 출력에 홑따옴표가 포함되었으나 이를 제외하도록 수정했다.

응용 프로그램에서 데이터베이스에 DBA로 접속한 이후 재접속 시 패스워드가 일치하지 않아도 정상 접속되는 문제 수정(CUBRIDSUS-7192)

응용 프로그램에서 데이터베이스에 DBA로 접속한 이후에는 DBA 또는 다른 사용자로 재접속할 때 패스워드가 일치하지 않아도 접속에 성공하는 문제를 수정했다.

login() 메서드의 동작 변경(CUBRIDSUS-6307)

CSQL 인터프리터를 DBA로 수행한 경우 패스워드 확인없이 다른 사용자로 연속해서 login()이 허용되었으나, DBA가 아닌 사용자로 login()을 수행한 이후에는 패스워드 확인없이 다른 사용자로 login() 수행이 허용되지 않도록 변경했다.

```
csql -u dba demodb
CALL login ('test1', '') ON CLASS db_user; -- dba가 test1에 login()하므로 패스워드 확인 없이 허용
```

```
CALL login ('test2', '') ON CLASS db_user; -- test1으로 login()한 이후 test2로 login() 시
패스워드 확인 없이는 허용되지 않음
```

드라이버

[JDBC, CCI] HOLDABLE CURSOR가 기본으로 동작하도록 변경(CUBRIDSUS-8609)

JDBC, CCI에서 커밋 이후에도 커서를 유지하도록 기본 동작을 변경하였다. 커서를 닫지 않으면 서버 메모리 사용량이 늘어나게 되기 때문에 사용이 완료된 커서는 반드시 닫아주어야 한다.

[JDBC] NUMERIC 타입의 이름을 DECIMAL이 아니라 NUMERIC으로 반환(CUBRIDSUS-8387)

DatabaseMetaData.getColumns() 메서드가 **NUMERIC** 타입의 이름을 DECIMAL로 반환하던 것을 NUMERIC으로 반환한다.

```
//수정 이전 버전에서는 Hibernate를 이용해서 엔티티 간 매핑 설정을 할 때 NUMERIC 타입의 칼럼을
지정하면
Caused by: org.hibernate.HibernateException: Wrong column type in mytbl_map for column col2.
Found: decimal, expected: numeric(19,0)"
와 같은 오류가 발생했다.

@ManyToMany
@JoinTable(name="mytbl",
joinColumns={@JoinColumn(name="col1", columnDefinition="varchar(255)")},
inverseJoinColumns={@JoinColumn(name="col2", columnDefinition="numeric(19,0)")})
private Set<MyGroup> accessMyGroups;
```

[JDBC] getColumnClassName() 메서드의 NUMERIC 타입 칼럼에 대한 JAVA 반환 타입 변경 (CUBRIDSUS-7532)

ResultSetMetaData.getColumnClassName() 메서드가 **NUMERIC** 타입 칼럼에 대해 기존의 java.lang.Double대신 java.math.BigDecimal을 반환하도록 수정했다.

[CCI] cci_get_db_parameter() 함수가 반환하는 잠금 타임아웃 값의 단위를 밀리초로 변경(CUBRIDSUS-7538)

cci_get_db_parameter() 함수가 반환하는 잠금 타임아웃 값을 초 단위에서 밀리초(msec) 단위로 변경했다.

[CCI] cci_connect_with_url() 함수의 URL 속성에서 autocommit 속성 제거(CUBRIDSUS-7306)

cci_connect_with_url() 함수의 URL 속성에서 autocommit 속성을 제거했다.

기본 설정 변경

레플리카 노드에서는 항상 보관 로그를 삭제하도록 변경(CUBRIDSUS-8556)

레플리카 노드에서는 보관 로그 삭제를 위해서는 시스템 파라미터 **force_remove_log_archives**의 설정 값을 항상 yes로 변경해야 했다. 설정을 하지 않았을 경우에 불필요한 보관 로그가 계속 쌓이면서 문제를 야기시킬 수 있었는데, 9.0 Beta부터는 레플리카 노드는 **force_remove_log_archives**의 설정 값과 상관 없이 항상 보관 로그를 삭제하도록 변경했다.

설치할 때 보관 로그 파일의 최대 개수를 설정하는 시스템 파라미터의 초기 값을 0으로 설정하도록 수정(CUBRIDSUS-6603)

CUBRID를 설치할 때 설치되는 기본 cubrid.conf 파일에 "log_max_archives=0" 설정이 추가되었다.

log_max_archives의 값이 0이면 보관 로그 파일을 보관하지 않으므로, 보관 로그 파일이 디스크 공간을 차지하지는 않지만 저장 매체 장애(media failure)가 발생하면 원하는 시점으로의 복구가 불가능할 수 있다. 저장 매체 장애에 대비하여 데이터베이스를 복구할 수 있도록 하기 위해서는 백업 주기 등을 감안하여 이 파라미터 값을 적절하게 설정해야 한다.

복제 로그 반영 프로세스의 메모리 사용량이 500MB를 넘지 않도록 변경(CUBRIDSUS-6068)

HA 환경에서 복제 로그 반영 프로세스의 메모리 사용량이 500MB를 넘으면 복제 불일치가 발생할 수 있었으나, 복제 로그 반영 프로세스의 사용량이 500MB를 넘지 않도록 수정했다. 이전 버전에서 cubrid_ha.conf의 **ha_apply_max_mem_size** 값을 500 이상으로 설정한 사용자는 9.0 Beta 버전 이상으로 업그레이드한 이후 500 이하로 변경해야 한다.

기타

오류 번호 범위 변경(CUBRIDSUS-7666)

브로커 응용 서버(CAS), 브로커 서버, CCI 드라이버, JDBC 드라이버의 오류 번호 범위를 변경했다. CAS는 -10000부터 -10999, 브로커 서버는 -11000부터 -11999, CCI는 -20000부터 -20999, JDBC는 -21000부터 -21999 범위를 에러 번호로 사용한다.

응용 프로그램에서 오류 이름이 아닌 오류 번호를 직접 명시하여 사용한 경우에는 응용 프로그램 수정이 필요하다.

lock_timeout값이 -1이나 0일 때 브로커 응용 서버(CAS) 로그와 lockdb 유틸리티의 출력 형식 변경(CUBRIDSUS-8915)

시스템 파라미터인 **lock_timeout** 값이 -1(infinite wait)또는 0(no wait)으로 설정되어 있을 때, CAS 로그와 **lockdb** 유틸리티의 출력 형식을 각각 그 의미대로 "Infinite wait", "No wait"로 출력하도록 변경했다.

2.4 개선 및 오류 수정

성능 및 리소스

SET 절에 서로 다른 상수를 명시한 UPDATE 문들이 질의 계획 캐시에 별개로 캐시되지 않도록 개선(CUBRIDSUS-8511)

UPDATE 문의 **SET** 절에 명시된 상수 값이 다르면 해당 질의들이 질의 계획 캐시에서 같은 패턴임에도 별개로 간주되었었으나, 이들 상수들이 호스트변수로 자동 치환되도록 하여 하나의 패턴만 유지되도록 했다.

ORDER BY 절과 LIMIT 절이 있는 같은 패턴의 질의들이 질의 계획 캐시에 다른 계획으로 별개로 캐시되지 않도록 개선(CUBRIDSUS-8813)

ORDER BY 절과 LIMIT 절이 있는 같은 패턴의 질의는 LIMIT 절의 상수 값이 다르더라도 질의 계획 캐시에 같은 계획으로 저장되도록 수정했다.

INSERT ON DUPLICATE KEY UPDATE 문의 수행 성능 개선(CUBRIDSUS-8287)

INSERT ON DUPLICATE KEY UPDATE 문의 대상 테이블에 기본 키를 포함하여 고유 키가 두 개 이상 존재하는 경우의 수행 성능을 개선했다.

```
CREATE TABLE x (a INT PRIMARY KEY, b INT, c INT, d INT, UNIQUE(b), UNIQUE(c));
CREATE SERIAL s;
INSERT INTO x VALUES (s.NEXT_VALUE, 0, 0, 0) ON DUPLICATE KEY UPDATE d = d+1;
```

커버링 인덱스 스캔 성능 개선(CUBRIDSUS-7466)

잠금 에스컬레이션 성능 개선(CUBRIDSUS-5698)

레코드 잠금이 일정 개수를 초과하면 테이블 잠금으로 변환하는 작업을 수행하는 잠금 에스컬레이션(lock escalation)의 성능을 개선했다. **lock_escalation** 파라미터의 값이 5000 이고 100개의 분할을 가진 테이블에 100만 개의 레코드를 입력할 때 3.5배 성능이 향상되었다.

키 잠금과 관련하여 롤백 시 데이터 불일치 현상 수정 및 DML 성능 개선(CUBRIDSUS-7080)

키 잠금(key locking) 방식을 수정하여 인덱스가 있는 행에 **INSERT하는** 작업의 롤백 등에서 발생할 수 있는 데이터 불일치 현상을 수정했고, 인덱스가 있는 행에 대한 **INSERT**, **DELETE** 및 **SELECT**의 처리 성능을 개선했다.

UNION ALL 질의문의 수행 성능 개선(CUBRIDSUS-8130)

UNION ALL 질의문의 결과를 만들 때 앞쪽 질의문의 중간 결과를 복사하지 않고 재사용하여 성능을 개선했다.

```
// 다음의 예에서 t1 테이블의 건수가 클수록 수정으로 인한 성능 개선폭이 증가한다.
SELECT * FROM t1 UNION ALL SELECT * FROM t2;
```

DATE_ADD 함수의 성능 개선(CUBRIDSUS-7464)

테이블 DROP과 CREATE를 반복할수록 CREATE 시간이 오래 걸리는 문제(CUBRIDSUS-7288)

테이블 **DROP**과 **CREATE**를 반복할수록 **CREATE** 시간이 오래 걸리는 문제를 수정했다. 참고로 수정 이전 버전에서도 테이블 생성 시 아래의 예와 같이 **REUSE_OID** 옵션을 지정하면 이러한 현상이 발생하지 않았다.

```
CREATE TABLE reuse_tbl (a INT PRIMARY KEY) REUSE_OID;
```

INSERT와 DELETE 반복 시 수행 시간이 점차 느려지는 현상(CUBRIDSUS-7654)

INSERT와 **DELETE**를 반복 수행하면 수행 시간이 점차 느려지는 현상을 수정했다.

DESC, SHOW INDEX, SHOW COLUMNS 문 수행 성능 개선(CUBRIDSUS-5868)

TRUNCATE 문을 빈번하게 수행한 이후 INSERT를 비롯한 접근 성능이 현저히 떨어지는 현상(CUBRIDSUS-6499)

TRUNCATE 문을 빈번하게 수행한 이후에 **INSERT** 성능이 현저히 떨어지는 현상을 수행했다.

페이지 크기를 넘는 레코드들에 대해 UNION 수행 결과 건수가 1건인 경우 메모리 누수 현상(CUBRIDSUS-7158)

페이지 크기를 넘는 오버플로우 레코드들에 대해 **UNION** 질의 수행 결과 건수가 1건인 경우 발생하는 메모리 누수(memory leak) 현상을 수정했다. **UNION ALL** 질의는 메모리 누수 현상이 발생하지 않았다.

하나의 인덱스 키에 중복되는 레코드 값이 많아진 이후 수행되는 INSERT에 대해 볼륨 사용량이 급격히 증가하는 현상(CUBRIDSUS-8375)

하나의 인덱스 키에 중복되는 레코드 값이 많아 오버플로우 OID 레코드가 생성되면, 이후 해당 키보다 작은 값이 입력될 때 항상 새로운 페이지에 키가 입력되어 인덱스 볼륨 사용량이 급격히 증가하는 문제를 수정했다.

같은 칼럼에 두 개 이상의 외래 키를 정의하고 테이블을 DROP한 이후 일부 공간이 재사용되지 못하는 오류(CUBRIDSUS-8256)

하나의 칼럼에 이름만 다른 외래 키를 두 개 이상 정의하고 테이블을 **DROP**하면 해당 테이블이 사용하던 일부 공간이 재사용되지 못하는 오류를 수정했다.

```
CREATE TABLE foo (a INT, PRIMARY KEY (a));
CREATE TABLE bar (a INT,
CONSTRAINT con1 FOREIGN KEY(a) REFERENCES foo (a),
CONSTRAINT con2 FOREIGN KEY(a) REFERENCES foo (a));
-- INSERT records
DROP TABLE bar;
DROP TABLE foo;
```

INSERT ... SELECT 문으로 대량의 레코드를 입력하면 서버 프로세스의 메모리 사용량이 급격히 증가하는 문제(CUBRIDSUS-8736)

INSERT ... SELECT 문을 통해서 대량(예를 들어 200만건 정도)의 레코드를 입력할 때에 서버 프로세스의 메모리 사용량이 급격히 증가하는 문제를 수정했다.

INSERT 문의 값으로 질의문 또는 함수를 인자로 사용한 CONCAT_WS 함수를 입력하는 경우 발생하는 오류(CUBRIDSUS-6206)

INSERT 문의 값으로 질의문 또는 함수를 인자로 사용한 CONCAT_WS 함수를 입력하는 경우 "ERROR: Cannot eva luate ' concat_ws('a', cast(SYS_DATE as varchar))'" 오류가 발생하는 문제를 수정했다.

```
INSERT INTO tbl VALUES (1,concat_ws('a',SYS_DATE()));
```

OR 절에 부질의가 포함된 질의 수행 시 장시간 소요되는 현상(CUBRIDSUS-6031)

OR 절에 부질의(subquery)가 포함된 질의 수행 시 장시간 소요되는 현상을 수정했다.

```
SELECT col2 FROM tab0
WHERE (A AND B) OR (col3 IN (SELECT i FROM t WHERE X AND Y OR Z AND W) AND D);
```

DELETE FROM ALL 문이 LOB 파일을 삭제하지 못하는 오류(CUBRIDSUS-5596)

DELETE FROM ALL 문으로 상속 계층에 대해서 삭제할 때 해당 테이블 및 이를 상속받는 테이블들의 LOB 파일이 삭제되지 않는 오류를 수정했다.

```
DELETE FROM ALL parent_tbl;
```

인덱스 생성 시 임시 볼륨 사용량 최적화(CUBRIDSUS-5528)

인덱스 생성 과정에서 더 이상 사용되지 않는 페이지를 반납하여 임시 볼륨을 필요 이상으로 많이 사용하지 않도록 수정했다.

일시적 임시 볼륨 크기 최적화(CUBRIDSUS-5639)

일시적 임시 볼륨 추가 시 필요 이상의 큰 볼륨을 생성하지 않도록 수정했다.

특정 상황에서 인덱스 볼륨 여유 공간이 있어도 일반 볼륨을 생성하는 문제(CUBRIDSUS-5595)

인덱스 생성 시 여유 공간이 가장 많은 인덱스 볼륨 파일의 여유 공간이 인덱스 생성에 필요한 전체 공간의 25%보다 작으면, 인덱스 볼륨을 사용하지 않고 일반 볼륨을 생성하는 문제를 수정했다.

백그라운드 보관 로그 파일에서 정상 상황임에도 불구하고 파일 I/O sync. 오류 메시지를 잘못 출력하는 현상 수정(CUBRIDSUS-8163)

백그라운드 보관 로그 파일에서 정상 상황임에도 불구하고 다음과 같은 파일 I/O sync. 오류 메시지(에러 코드: -599)를 잘못 출력하는 현상을 수정했다.

```
An I/O error occurred while synchronizing state of volume
"/home/cubrid/database/testdb/testdb_lgar_t".... Bad file descriptor
```

UPDATE STATISTICS 문 수행으로 인해 다른 트랜잭션들이 장시간 대기하는 현상(CUBRIDSUS-6981)

UPDATE STATISTICS 문 수행 도중에 인덱스 페이지에 대한 래치(latch)를 오래 유지하지 않도록 하여 다른 트랜잭션들이 장시간 대기하지 않도록 수정했다.

연산자와 SQL 함수

날짜/시간 함수에서 입력 인자로 YYYYMMDDH 형식을 지원하게 됨(CUBRIDSUS-8622)

TIME, **TO_DATETIME**과 같은 날짜/시간 함수에서 입력 인자로 YYYYMMDDH 형식을 지원하도록 수정했다.

```
SELECT TIME('1104209');

time('1104209')
======================
'09:00:00'

SELECT TO_DATETIME('1104209','YYMMDDH');

to_datetime('1104209', 'YYMMDDH', 'en_US')
=========================================
09:00:00.000 AM 04/20/2011
```

ADDTIME 함수가 잘못된 결과를 반환하는 문제(CUBRIDSUS-8568)

ADDTIME 함수가 잘못된 결과를 반환하는 문제를 수정했다.

```
SELECT ADDTIME('2012-02-02','9:9:9');
```

INT 타입의 최소값이 저장된 칼럼에 % 연산, BIT_AND 함수 혹은 BIT_OR 함수 수행 시 오버플로우 오류(CUBRIDSUS-6203)

INT 타입의 최소값(-2147483648)이 저장된 칼럼에 % 연산, **BIT_AND** 함수 또는 **BIT_OR** 함수를 수행하면 "ERROR: Overflow occurred in ..." 오류가 발생하는 문제를 수정했다.

```
INSERT INTO tbl VALUES (-2147483648);
SELECT i%1009 FROM tbl;
SELECT BIT_AND(i) FROM tbl;
```

TRIM 함수가 잘못된 결과를 출력하는 오류(CUBRIDSUS-6591)

TRIM 함수가 지정한 삭제 대상 문자열보다 한 글자를 더 삭제하는 오류를 수정했다. 다음 예와 같이 "foook"에서 "foo"를 **TRIM**하면 "ok"를 출력해야 한다.

```
SELECT TRIM('foo' FROM 'foook');
```

INDEX_CARDINALITY 함수의 입력 인자로 소문자인 테이블 이름만 인식하는 오류(CUBRIDSUS-6264)

INDEX_CARDINALITY 함수의 입력 인자로 소문자인 테이블 이름만 인식하는 오류를 수정했다.

CONCAT 함수와 CONCAT_WS 함수의 입력 인자가 CHAR 타입만 있는 질의 수행 시 오류(CUBRIDSUS-6524)

CONCAT 함수와 **CONCAT_WS** 함수의 입력 인자가 **CHARACTER** 타입만 주어진 경우 "ERROR: No error message available." 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

```
CREATE TABLE t1 (a int , b char (20) );
INSERT INTO t1 VALUES (-1, '');
-- below statement will throw "ERROR: No error message available." message.
SELECT CONCAT(b) FROM t1;
```

CLOB_TO_CHAR 함수 수행 시 오류(CUBRIDSUS-6520)

CLOB 칼럼이 있는 테이블에 고유 인덱스를 스캔하는 **REPLACE** 문을 수행하고 커밋한후에 **CLOB_TO_CHAR** 함수를 수행하면 "ERROR: External file "xxxx" was not found." 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

```
SELECT id, CLOB_TO_CHAR(text) FROM tbl ORDER BY id;
```

TIMEDIFF 함수의 입력 인자가 DATE 타입인 질의 수행 시 오류(CUBRIDSUS-8692)

TIMEDIFF 함수의 입력 인자가 **DATE** 타입인 경우 "ERROR: Conversion error in time format." 오류 메시지와 함께 수행에 실패하는 현상을 수정했다.

```
SELECT TIMEDIFF(TO_DATE('2012-12-2'), TO_DATE('2012-11-2'));
```

CONCAT_WS 함수가 LEFT OUTER JOIN의 조건으로 존재할 때 조인 결과를 적게 출력하는 현상(CUBRIDSUS-6590)

CONCAT_WS 함수가 **LEFT OUTER JOIN**의 조건으로 존재할 때 **LEFT OUTER JOIN** 질의를 최적화하는 과정의 오류로 인해 **CONCAT_WS** 함수의 입력 인자 값 하나만 NULL이어도 함수의 결과 값을 항상 NULL로 간주하여 조인 결과를 적게 출력하는 현상을 수정했다.

```
SELECT * FROM t1 LEFT JOIN t2 ON t1.id = t2.id WHERE CONCAT_WS(' ', t1_name, t2_name) LIKE
'%In%';
```

DATE_ADD 함수의 INTERVAL 값을 호스트 변수로 사용하면 질의 수행 시 서버가 비정상 종료되는 현상(CUBRIDSUS-7610)

질의를 **PREPARE**할 때 **DATE_ADD** 함수의 **INTERVAL** 값을 호스트 변수로 사용하면, **EXECUTE**할 때 서버가 비정상 종료되는 현상을 수정했다. 이와 함께 **DATE_ADD** 함수의 **INTERVAL** 단위에 따라 입력 값으로 **INTERGER** 타입만 가능했던 동작을 **VARCHAR**도 가능하게 수정했다.

```
PREPARE s FROM 'SELECT DATE_ADD(?, INTERVAL ? YEAR_MONTH)';
EXECUTE s USING '2010-01-01', 1;
EXECUTE s USING '2010-01-01', '1-1';
```

집계 함수의 인자 안에 DISTINCTROW가 존재하는 질의가 동작하지 않는 문제(CUBRIDSUS-7414)

집계 함수의 인자 안에 **DISTINCTROW**가 존재하는 경우, "Syntax error: unexpected 'DISTINCTROW'" 오류를 출력하면서 해당 질의가 동작하지 않는 문제를 수정했다.

일부 SQL 함수에서 모든 인자가 호스트 변수일 때 값을 바인딩하면서 질의 수행에 실패하는 현상(CUBRIDSUS-7521)

NULLIF, LEAST, GREATEST 등 일부 SQL 함수에서 모든 인자가 호스트 변수일 때 값을 바인딩하면서 **DOUBLE**로 타입 변환을 시도하게 되어 질의 수행에 실패하는 현상을 수정했다.

```
preStmt = conn.prepareStatement("select nullif (?, ?)");
preStmt.setString(1, "A");
preStmt.setString(2, "a");
rs = preStmt.executeQuery();
```

PREPARE 문에서 DEFAULT 지정 인자를 생략한 DECODE 함수 수행 시 오류(CUBRIDSUS-9134)

PREPARE 문에서 **DECODE** 함수 수행 시 **DEFAULT** 값을 지정하는 입력 인자를 생략하거나 **NULL** 값이 주어지면 "ERROR: Attribute "val" cannot be made NULL." 오류 메시지와 함께 수행에 실패하는 현상을 수정했다.

```
PREPARE stmt1 FROM 'UPDATE foo SET del_ts = 100, val=DECODE(name,?,val + ?) WHERE name IN (?)';
EXECUTE stmt1 USING 'seo', 1, 'seo';
```

SELECT 리스트에 INSERT 함수 혹은 ELT 함수가 있고 그 뒤에 값이 오는 질의 수행 시 오류 (CUBRIDSUS-6577)

SELECT 리스트에 **INSERT** 함수 혹은 **ELT** 함수가 있고 그 뒤에 값이 오면 "ERROR: System error (query result) in ./../src/parser/query_result.c" 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

```
SELECT INSERT('test',2,1,'hi'), 5;
SELECT ELT(2, 1), 5;
```

부질의 내에 집계 함수가 있는 질의문에 USING INDEX 구문을 포함하면 비정상 종료되는 현상(CUBRIDSUS-8057)

SELECT 리스트의 부질의(sub-query) 내에 집계 함수를 포함하는 질의문에 **USING INDEX** 구문을 포함하면 비정상 종료되는 현상을 수정했다.

네트워크 장애나 HA 환경에서 절체된 상황에서 LAST_INSERT_ID 함수와 같은 세션 연산이 계속 실패할 수 있는 현상(CUBRIDSUS-7669)

정상적인 동작을 위해서는 브로커 응용 서버(CAS) 간에 세션 ID를 공유하면 안 되지만, 네트워크 장애나 HA 환경에서 절체 등으로 인해 세션 ID를 공유하게 된 브로커 응용 서버들 중 하나가 먼저 종료하면 남은 브로커 응용 서버에 접속되어 있는 응용 프로그램에서 세션 연산 수행이 계속 실패하는 현상을 수정했다. **LAST_INSERT_ID** 함수, **PREPARE** 구문, **SET**으로 정의한 사용자 세션 변수, **ROW_COUNT** 함수 등이 세션 연산에 해당된다.

SQL 문

SELECT ALL/DISTINCT 상수 표현이 가능해짐(CUBRIDSUS-6080)

"**SELECT ALL** 상수"와 "**SELECT DISTINCT** 상수"가 가능하도록 수정했다.

```
SELECT ALL 1;
SELECT DISTINCT 1;
```

아울러, 칼럼을 괄호로 감싸거나 상수를 DISTINCT하여 집계 함수를 수행하는 것이 가능하도록 수정했다.

```
SELECT SUM(DISTINCT(i)) FROM t;
SELECT SUM(DISTINCT 4) FROM t;
```

FROM 절의 유도 테이블(derived table) 이름 생략이 가능해짐(CUBRIDSUS-6546)

반드시 명시해야 했던 **FROM** 절의 유도 테이블 이름을 생략할 수 있도록 개선했다.

```
SELECT * FROM (SELECT sysdate FROM db_root);

// 수정 이전 구문
FROM (subquery) [ AS ] derived_table_name [( column_name [ {, column_name } ... ] )]
// 수정 이후 구문
FROM (subquery) [ [ AS ] derived_table_name [( column_name [ {, column_name } ... ] )] ]
```

사용하지 않는 예약어 제거(CUBRIDSUS-6250)

ALIAS, TYPE, VIRTUAL, TEST, WAIT 등 사용하지 않는 예약어(reserved word)는 제거하여, 해당 예약어를 테이블 이름이나 칼럼 이름 등의 식별자로 사용할 수 있게 개선했다.

```
CREATE TABLE TYPE (test INT);
```

DO 문에 SELECT 질의문의 입력이 가능해짐(CUBRIDSUS-6528)

DO 문에 **SELECT** 질의문의 입력이 가능하도록 수정했다. DO 문은 SQL 문장을 수행만 할뿐 수행한 후에 결과집합을 반환하지는 않는다.

```
DO (SELECT count(*) FROM athlete);
```

UTF-8 문자셋을 테이블 이름이나 칼럼 이름 등의 식별자로 사용 가능해짐(CUBRIDSUS-7227)

UTF-8 문자셋을 테이블 이름이나 칼럼 이름 등의 식별자로 사용 가능하도록 개선했다. 자세한 사항은 매뉴얼의 다국어 지원을 참고하면 된다.

컬럼 이름 변경 시 기존 컬럼 이름과 새 컬럼 이름 사이에 TO도 사용할 수 있게 됨(CUBRIDSUS-7477)

컬럼 이름 변경 시 **AS**와 함께 **TO**도 사용할 수 있도록 추가했다.

```
CREATE TABLE t (a int);
ALTER TABLE t RENAME COLUMN a TO b;
```

IN, NOT IN 조건의 스칼라 부질의에, ORDER BY 절이 사용되거나 SELECT 리스트의 칼럼 개수가 여러 개인 경우 잘못된 질의 결과를 출력하는 현상(CUBRIDSUS-7700)

IN, NOT IN 조건에 포함된 스칼라 부질의(scalar subquery)에 **ORDER BY** 절이 사용되거나 해당 질의에 대한 **SELECT** 리스트의 칼럼 개수가 여러 개인 경우 잘못된 질의 결과를 출력하는 현상을 수정했다.

```
// 수정 이전 버전에서 스칼라 부질의에 ORDER BY 절이 사용되면 질의 결과가 항상 0건이 되었다.
SELECT * FROM tbl WHERE col IN (SELECT col FROM tbl2 ORDER BY b);

// 수정 이전 버전에서 스칼라 부질의에 대한 SELECT 리스트의 칼럼이 두 개 이상 사용되면
문법(semantic) 오류가 발생해야 하나, 질의 결과가 0건이 되었다.
SELECT * FROM tbl WHERE col IN (SELECT a, b FROM tbl2);
SELECT * FROM tbl WHERE col NOT IN (select a,b from tbl2);
```

IN 절의 호스트 변수 타입이 날짜/시간인 경우 정상 수행되지 않는 현상(CUBRIDSUS-7916)

IN 절의 호스트 변수 타입이 날짜/시간인 경우 정상 수행되지 않는 현상을 수정했다. 수정 이전 버전에서 바인딩할 값의 타입이 날짜/시간 타입인 경우 아래의 Q1은 정상 동작했으나, Q2는 데이터가 삭제되지 않았다.

```
DELETE FROM tbl WHERE d = ?;     -- Q1
DELETE FROM tbl WHERE d IN (?);  -- Q2
```

GROUP BY 절과 연관 부질의, SELECT 리스트에 없는 칼럼으로 정렬하는 ORDER BY가 같이 있으면 잘못된 결과를 출력할 수 있는 현상(CUBRIDSUS-8640)

GROUP BY 절과 연관 부질의(correlated subquery) 그리고 **ORDER BY** 절에 **SELECT** 리스트에 없는 칼럼이 지정되었을 때 잘못된 결과를 출력하는 현상을 수정했다.

```
SELECT (SELECT f1.a FROM foo f1 WHERE f1.b=f2.b) as t
FROM foo f2
WHERE f2.b >= 1 and f2.b < 10
GROUP BY f2.c
ORDER BY f2.c;
```

GROUP BY 절의 WITH ROLLUP 수정자 지정 시 수행 결과가 잘못될 수 있는 문제(CUBRIDSUS-6518)

GROUP BY 절의 **WITH ROLLUP** 수정자를 지정하면 수행 결과가 잘못될 수 있는 문제를 수정했다.

```
// 수정 이전 버전에서 다음 질의 수행 시 ROLLUP된 결과 값이 출력되지 않았다.
SELECT a FROM t1 GROUP BY A WITH ROLLUP;

// 수정 이전 버전에서 다음 질의 수행 시 ROLLUP된 결과 레코드의 a 칼럼 값이 NULL인데 a>1 이므로 출력되지 않아야 하나 출력되었다.
SELECT a, COUNT(*) FROM t1 GROUP BY a WITH ROLLUP HAVING a>1;
```

부질의의 SELECT 리스트 형식이 "*, 칼럼"인 경우 "*" 뒤의 칼럼들을 출력하지 않는 문제(CUBRIDSUS-6589)

부질의의 **SELECT** 리스트 형식이 "*, 칼럼"인 경우 "*" 뒤에 명시된 칼럼들을 출력하지 않는 문제를 수정했다.

```
SELECT b FROM (SELECT *, 'hello' AS b FROM t1) t;
```

ESCAPE 절에 명시된 이스케이프 문자가 LIKE 패턴의 마지막 문자인 경우에 오류(CUBRIDSUS-6849)

ESCAPE 절에 명시된 이스케이프 문자가 **LIKE** 패턴의 마지막 문자인 경우에 "System error" 오류 메시지와 함께 질의 수행에 실패하는 문제를 수정했다.

```
CREATE TABLE foo(a char(10));
SELECT * FROM foo WHERE a LIKE 'ab' ESCAPE 'b';
ERROR: System error (db_compress_like_pattern) in ../../src/optimizer/query_rewrite.c (line: 3291)
```

SELECT 리스트에 *를 지정한 질의문과 칼럼 명을 명시한 질의문을 UNION할 때 오류(CUBRIDSUS-6588)

SELECT 리스트에 와일드카드(*)를 지정한 질의문과 칼럼 명을 명시한 질의문을 UNION 등 문장 집합 연산자로 연결한 경우 칼럼 개수가 같음에도 불구하고 "ERROR: The number of columns, 0, in the left query do not match the number of columns, N, in the right query." 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

```
(SELECT * FROM t1) UNION (SELECT a, b FROM t2 );
```

질의의 비교 조건 절에 호스트 변수가 있는 경우 잘못된 결과 출력(CUBRIDSUS-5843)

질의의 비교 조건 절에 호스트 변수가 있는 경우, 해당 호스트 변수에 입력되는 값의 타입이 변환되어 기대 결과와 다르게 결과가 출력되는 문제를 수정했다.

```
// 수정 이전 버전에서도 호스트 변수가 아닌 상수를 쓰는 경우에는 결과값이 정상 출력되었다.

SELECT * FROM foo WHERE a > 2.5;

a
=============
3
4
5

// 칼럼 a의 타입은 INT일 때, 수정 이전 버전에서는 입력되는 값이 2.5이면 INT인 3으로 변환되어 비교되는 오류가 발생했다.

PREPARE stmt FROM 'SELECT * FROM foo WHERE a > ?';
EXECUTE stmt USING 2.5;

a
=============
4
5
```

UNION과 LIMIT 절을 포함한 부질의가 있는 질의 수행 시 잘못된 결과를 출력(CUBRIDSUS-6596)

UNION과 LIMIT 절을 포함한 부질의가 있는 질의를 수행하면 잘못된 결과를 출력하는 문제를 수정했다.

```
CREATE TABLE t1 (a INT);
INSERT INTO t1 VALUES (1);

--수정 이전 버전에서는 다음 질의에 대해 0건을 출력했다.
SELECT * FROM ((SELECT a from t1) UNION (SELECT a from t1) LIMIT 1) s1;
```

UNION 절이 있는 부질의가 WHERE 절 조건에 사용되는 질의 수행 시 오류(CUBRIDSUS-6530)

UNION 절이 있는 부질의가 **WHERE** 절 조건에 존재하면 "ERROR: '(select t1.i from t1 t1)<>0' is not union compatible with '(select t2.i from t2 t2)'." 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

```
SELECT * FROM t1 WHERE EXISTS ((SELECT i FROM t1) UNION ALL (SELECT i FROM t2));
```

병합 조인 방식으로 OUTER JOIN 질의 수행 시 잘못된 질의 결과를 출력하는 오류(CUBRIDSUS-5703)

병합 조인(merge join) 방식으로 **OUTER JOIN** 질의를 수행하면 잘못된 질의 결과를 출력하는 오류를 수정했다.

```
SELECT /*+ USE_MERGE */ *
FROM tab t
LEFT OUTER JOIN idx i on t.t = i.t
LEFT OUTER JOIN col c on i.i = c.i AND c.c = t.c;
```

ROWNUM과 ORDERBY_NUM()이 같이 사용된 질의에서 지정된 ROWNUM의 행 개수보다 ORDERBY_NUM()의 행 개수가 크면 결과 개수가 잘못되는 현상(CUBRIDSUS-6676)

ROWNM으로 행 개수를 제한한 후 정렬된 결과에서 **ORDERBY_NUM**()으로 행 개수를 제한 출력하는 조건에서 지정된 **ROWNUM**의 행 개수보다 **ORDERBY_NUM**()의 행 개수가 크면 잘못된 개수의 결과를 출력하는 현상을 수정했다.

```
SELECT *
FROM foo f, bar b
WHERE f.a > 0 AND f.a = b.a AND ROWNUM <=4
ORDER BY f.a FOR ORDERYBY_NUM()<=10;
```

TIMESTAMP 타입에서 오버플로우 오류 처리가 잘못되는 문제(CUBRIDSUS-6004)

TIMESTAMP 타입에 대해 아래와 같은 질의에서 오버플로우 오류 처리가 되지 않는 문제를 수정했다. 수정된 버전에서는 아래의 질의를 수행하면 오버플로우 오류를 반환한다.

```
SELECT timestamp'01/19/2038 12:14:07 pm' - CAST(-32768 as smallint);
```

ANY, SOME 한정자의 인자로 ORDER BY 절이 있는 부질의 수행 시 오류(CUBRIDSUS-7799)

ANY, SOME 한정자의 인자로 **ORDER BY** 절이 있는 부질의를 수행하면 "ERROR: Aggregate function must have 1 argument: min(t2.id, t2.a)." 오류가 발생하는 문제를 수정했다.

```
SELECT * FROM t1 WHERE id > ANY(SELECT id FROM t2 ORDER BY a);
```

다중 질의를 한 번에 PREPARE한 이후 반복 실행하면 첫 번째 수행만 정상 수행되는 현상(CUBRIDSUS-7455)

다중 질의를 한 번에 **PREPARE**한 이후 반복 실행하면 첫 번째 수행만 정상 수행되고, 두 번째 수행부터는 오류가 발생하는 현상을 수정했다.

```
String MULTI_SELECT = "SELECT A FROM T1 WHERE A = ?; UPDATE T1 SET A = 2 WHERE A = 2; SELECT A,
B FROM T1 WHERE A = ?; SELECT A, B, A AS C FROM T1 WHERE A = ?;";
PreparedStatement p = c.prepareStatement(MULTI_SELECT);
...
while(...)
{
...
p.execute();
...
}
```

EXISTS 조건과 FOR ORDERBY_NUM() BETWEEN 조건이 있는 질의에서 key limit 최적화가 잘못 적용되는 문제(CUBRIDSUS-9198)

EXISTS 조건과 **FOR ORDERBY_NUM**() **BETWEEN** 조건이 있는 질의에서 적용되지 않아야 할 key limit 최적화가 적용되면서 잘못된 결과 건수를 출력하는 문제를 수정했다.

```
SELECT cd, tcd, nm
FROM a
WHERE EXISTS (SELECT 1 FROM b
              WHERE a.cd = b.cd
              AND (b.no = 10000 OR b.uno = 10000))
ORDER BY a.nm
FOR ORDERBY_NUM() BETWEEN 1 AND 50;
```

UNION 질의에서 앞쪽 VARCHAR 타입 칼럼의 크기가 뒤쪽 칼럼의 크기보다 작을 때 발생하는 오류(CUBRIDSUS-9148)

UNION 질의에서 앞쪽 **VARCHAR** 타입 칼럼의 크기가 뒤쪽 칼럼의 크기보다 작으면 "ERROR: Execute: Query execution failure #1336." 오류 메시지와 함께 수행에 실패하는 현상을 수정했다.

```
CREATE TABLE u1 (a varchar(1));
CREATE TABLE u2 (a varchar(2));
INSERT INTO u1 values ('1');
INSERT INTO u2 values ('22');

SELECT a FROM (SELECT a FROM u1 UNION ALL SELECT a FROM u2) t(a);
```

PREPARE 구문으로 CASE 또는 DECODE 문 수행 시 오류(CUBRIDSUS-6847)

PREPARE 구문으로 **CASE** 또는 **DECODE** 문 수행 시 모든 인자들이 호스트변수로 주어지는 등 타입을 결정할 수 없을 때, "ERROR: Semantic: System error (generate var) in ../../src/parser/xasl_generation.c" 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

```
PREPARE st FROM 'SELECT CASE WHEN col=? THEN ? ELSE ? END FROM tbl;';
```

'%' 문자를 LIKE 문의 이스케이프 문자로 사용할 때의 오동작(CUBRIDSUS-7211)

'%' 문자를 LIKE 문의 이스케이프 문자로 사용하면 오동작하는 오류를 수정했다.

```
// 다음 질의 수행 시 첫번째 문자는 '%'이며, 두번째 문자 '_'는 아무 문자나 하나가 오고, 세번째 문자 이후는 'cab'인 문자열이 검색된다.

SELECT * FROM foo WHERE a LIKE '%%_cab' escape '%';
```

NUMERIC 타입이 허용하는 범위 내의 피연산자로 나누기 연산을 수행함에도 불구하고 발생하는 오버플로우 오류(CUBRIDSUS-6506)

NUMERIC 타입의 피연산자로 나누기 연산을 수행할 때, 입력 인자 값이 **NUMERIC** 타입이 허용하는 범위 내의 숫자임에도 불구하고 "ERROR: Data overflow on data type numeric" 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

```
// 수정 이전 버전에서는 다음의 경우에도 오류를 발생했다.

SELECT 9/1.23456789012111111111;
SELECT -9/1.23456789012111111111;
```

NUMERIC 타입 이외의 숫자 타입, 날짜/시간 타입에 대한 precision과 scale 값 수정(CUBRIDSUS-6967)

NUMERIC 타입 이외의 숫자 타입, 날짜/시간 타입들은 precision과 scale 값이 모두 0이었으나, 각 타입마다 precision과 scale 값을 반환하도록 수정했다. 예를 들어, **INT**는 최대 10자리 숫자이므로 precision은 10이고 scale은 0이다. **DATETIME**은 "hh:mi:ss.fff mm/dd/yyyy"로 나타내므로 precision은 23이고 scale은 3이다.

GROUP BY 절에 상수 지정하여 질의 수행 시 오류(CUBRIDSUS-6268)

SELECT '' AS group_key FROM tbl GROUP BY group_key와 같이 **GROUP BY** 절에 상수를 지정하면 "ERROR: xxxxxxxx in sort spec is out of range." 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

NOT과 단항 연산자 +, -를 이용한 조건 질의 수행 시 오류(CUBRIDSUS-6040)

NOT과 단항 연산자 -를 이용하여 다음과 같은 조건 질의 수행 시 "ERROR: 'unknown opcode' operator is not defined on types integer and integer." 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

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

아울러 질의문에 단항 연산자 + 와 - 를 같이 사용하는 경우 발생하는 오류를 수정했다.

```
SELECT * FROM tab0 WHERE col0 IN (+ - col0);
```

ORDER BY 절을 포함한 뷰에 대해 특정 컬럼만 조회하는 SELECT 질의를 수행하면 서버 프로세스가 비정상 종료될 수 있는 현상(CUBRIDSUS-7140)

```
CREATE VIEW va AS SELECT code, name, gender, nation_code FROM athlete ORDER BY nation_code;
SELECT code, name FROM va;
```

CHAR 타입 칼럼에 INT 타입을 바인딩하여 INSERT INTO ... SELECT ? ... 질의 수행 시 오류(CUBRIDSUS-6563)

INSERT INTO ... SELECT ? FROM ... 질의 수행 시 **CHAR** 타입 칼럼에 **INT** 타입을 바인딩하면 "ERROR: A domain conflict exists on attribute *noname*" 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

```
CREATE TABLE t (a CHAR(1));
PREPARE s FROM 'INSERT INTO t SELECT ? FROM db_root';
EXECUTE s USING 1;
```

IN 또는 EXISTS 표현식의 한 쪽 조건이 조건식으로 된 SELECT 리스트를 가진 부질의인 경우 서버 프로세스가 비정상 종료되는 문제(CUBRIDSUS-6482)

IN 또는 **EXISTS** 표현식의 조건이 단순 비교, **ALL, BETWEEN, LIKE, ISNULL** 등 조건식으로 된 **SELECT** 리스트인 경우 서버 프로세스가 비정상 종료되는 문제를 수정했다.

```
SELECT * FROM t1 WHERE v IN (SELECT (1 = 1));
SELECT * FROM t1 WHERE v IN (SELECT (aaa' LIKE 'bbb'));
SELECT * FROM t1 WHERE EXISTS (SELECT (1 < ALL{1,3,4}));
```

스키마 변경 이전으로 부분 롤백 수행 이후 USING INDEX 절을 포함한 질의 수행 시 오류(CUBRIDSUS-6458)

스키마를 변경하기 전으로 부분 롤백(partial rollback)을 수행한 이후 **USING INDEX** 절을 포함한 질의를 수행하면 "ERROR: Execute: Query execution failure #10842." 오류 메시지와 함께 질의 수행에 실패하는 문제를 수정했다.

```
;autocommit off
CREATE TABLE t (id INTEGER, textlabel VARCHAR(255), description VARCHAR(4096));
CREATE INDEX i_t_id_text ON t(id, textlabel);
COMMIT;
SAVEPOINT sp4;
TRUNCATE t;
SELECT * FROM t WHERE id > -1 USING INDEX i_t_id_text(+);
ROLLBACK TO SAVEPOINT sp4;
SELECT * FROM t WHERE id > -1 USING INDEX i_t_id_text(+);
```

CASE 연산식에 정의되지 않은 칼럼을 사용하거나 서버 프로세스 종료 후 LAST_INSERT_ID 함수 호출 시 CSQL, CAS가 비정상 종료되는 현상(CUBRIDSUS-5759)

CASE 연산식에 정의되지 않은 칼럼이 있는 경우에 CSQL, CAS 등이 비정상 종료되는 현상을 수정했다.

```
// 아래 구문에서 a는 정의되지 않은 칼럼이다.

UPDATE tbl SET col1 = (CASE WHEN EXISTS (SELECT * FROM tbl2 WHERE LENGTH(a) > 0)
                            THEN (SELECT col2 FROM tbl2 WHERE colx='1')
                            ELSE (SELECT col1 FROM tbl2 WHERE colx='1') END)
```

아울러 서버 프로세스 종료 후 **LAST_INSERT_ID** 함수 호출 시 CSQL, CAS 등이 비정상 종료될 수 있는 문제를 수정했다.

중첩 깊이가 128을 초과하는 부질의를 수행하면 응용 프로그램이 비정상 종료되는 현상(CUBRIDSUS-7826)

중첩 깊이가 128을 초과하는 부질의(subquery)를 수행하면 응용 프로그램이 비정상 종료되는 현상을 수정했다.

```
SELECT * FROM (SELECT * FROM (... (SELECT 1) ...) ...);
```

DESCRIBE 문 수행 시 STRING 타입에 대해 길이를 잘못 출력하는 오류(CUBRIDSUS-6432)

테이블 정보를 출력하는 **DESCRIBE** 문을 수행하면 **STRING** 타입의 길이(precision)를 -1로 잘못 출력했으나 이를 출력하지 않도록 수정했다.

참고로, **STRING**(n)으로 출력되는 것은 **VARCHAR**(n) 타입을 의미한다.

```
csql> DESCRIBE test_tbl;

Field Type Null Key Default Extra
=======================================
's_name' 'CHAR(1)' 'YES' '' NULL ''
'f_name' 'STRING(30)' 'YES' '' NULL ''
'name' 'STRING' 'YES' '' NULL ''
```

DECIMAL(p, s) 타입에서 허용 범위를 벗어나는 p와 s의 입력이 가능한 문제(CUBRIDSUS-6505)

DECIMAL(p, s) 타입의 정밀도 p와 스케일 s를 벗어나는 값의 입력이 허용되는 오류를 수정했다.

```
CREATE TABLE t1 (col1 decimal (5, 2));

// 수정 이전 버전에서는 아래의 질의 수행에 성공하여 1000.00이라는 값이 저장되는 오류가
존재했다.
INSERT INTO t1 VALUES (999.999);
```

테이블 생성 시 DEFAULT 표현식에 대한 문법 검사 없이 수행되는 문제(CUBRIDSUS-6761)

테이블 생성 시 **DEFAULT** 표현식에 대한 문법 검사가 되도록 수정했다. 아래의 예에서 수정 이전에는 **SYSTIMESTAMP** 뒤의 - 9999의 입력을 무시하고 테이블이 생성되는 현상이 발생했으나, 수정 이후에는 문법에 맞지 않기 때문에 오류를 출력한다.

```
CREATE TABLE foo(a TIMESTAMP DEFAULT SYSTIMESTAMP - 9999);

Invalid DEFAULT clause. 'sys_timestamp ' cannot be used in a nested expression.
```

호스트 변수들 간의 곱셈/나눗셈 연산 수행 시 바인딩되는 문자열이 숫자 타입으로 자동 형 변환이 되지 않는 오류(CUBRIDSUS-5506)

호스트 변수들 간의 곱셈/나눗셈 연산 수행 시 바인딩되는 문자열이 숫자 타입으로 자동 형 변환이 되지 않는 오류를 수정했다.

```
CREATE TABLE t1 (i1 integer);
PREPARE st FROM 'INSERT INTO t1(i1) VALUES (? * ?)';
EXECUTE st USING '4', '2.2';
```

UNIQUE 키 위반으로 INSERT 문 수행이 실패한 이후 LAST_INSERT_ID 함수의 값이 잘못 출력될 수 있는 현상(CUBRIDSUS-6450)

UNIQUE 키 위반으로 **INSERT** 문 수행이 실패한 이후 **LAST_INSERT_ID** 함수의 결과 값이 잘못 출력될 수 있는 현상을 수정했다.

```
CREATE TABLE t1 (k INT PRIMARY KEY AUTO_INCREMENT, a INT UNIQUE);
INSERT INTO t1 (a) VALUES (1);
INSERT INTO t1 (a) VALUES (2);

-- below sql will fail because of unique constraint in a column.
INSERT INTO t1 (a) VALUES (1);

-- below should return an old value(2) because of failed insert operation.
SELECT LAST_INSERT_ID();
```

유효하지 않은 ALTER COLUMN 문 수행 시 오류를 출력하지 않는 문제(CUBRIDSUS-6759)

유효하지 않은 **ALTER COLUMN ... SET DEFAULT** 문 수행 시 오류를 출력하지 않는 문제를 수정했다. 다음 예에서 칼럼 a는 **TIMESTAMP** 컬럼이므로 **DEFAULT** 값으로 'aaa'와 같은 문자열을 사용할 수 없다.

```
CREATE TABLE foo(a TIMESTAMP);
ALTER TABLE foo ALTER COLUMN a SET DEFAULT 'aaa';
```

BIT 타입 칼럼에 setBytes() 메서드로 값을 설정하여 INSERT하면 잘못된 데이터가 저장되는 문제(CUBRIDSUS-6628)

JDBC 응용 프로그램에서 **BIT** 타입 칼럼에 setBytes() 메서드로 호스트 변수의 값을 설정하여 입력하면 값이 잘못되어 저장되는 현상을 수정했다.

INSERT ... ON DUPLICATE KEY UPDATE 문 수행 시 기본 키에 NULL 값이 입력될 수 있는 문제(CUBRIDSUS-6448)

아래의 예와 같이 **INSERT ... ON DUPLICATE KEY UPDATE** 문을 수행하면 기본 키에 NULL 값이 입력되는 문제를 수정했다.

```
CREATE TABLE t1(id int AUTO_INCREMENT NOT NULL, c CHAR(1) NOT NULL,
                counter int NOT NULL DEFAULT 1, PRIMARY KEY(id), UNIQUE KEY(c));
INSERT INTO t1 (id, c) VALUES (NULL, 'a'), (NULL, 'a')
ON DUPLICATE KEY UPDATE id = null, counter = counter + 1;
```

INSERT ... SELECT 구문 수행 시 UNIQUE 칼럼의 값으로 NULL이 입력되는 경우 COUNT(*) 질의의 결과가 잘못 출력되는 현상(CUBRIDSUS-8338)

INSERT ... SELECT 구문 수행시 **UNIQUE** 칼럼의 값으로 NULL이 입력되는 경우 **COUNT**(*) 질의의 결과가 잘못되는 현상을 수정했다. **COUNT**(칼럼 이름)과 같이 칼럼 이름을 명시하여 **COUNT**를 수행하는 경우에는 이전 버전에서도 정상 동작했다.

```
CREATE TABLE t1(id INT AUTO_INCREMENT, mgrid INT UNIQUE, dummy INT);
INSERT INTO t1(dummy) VALUES (1);
INSERT INTO t1(dummy) SELECT dummy FROM t1;
INSERT INTO t1(dummy) SELECT dummy FROM t1;

SELECT COUNT(*) FROM t1;
```

AUTO_INCREMENT 칼럼으로 리스트 분할된 테이블에 INSERT 수행 시 오류(CUBRIDSUS-6522)

AUTO_INCREMENT 칼럼으로 리스트 분할된 테이블에 **INSERT**를 수행하면 "ERROR: Appropriate partition does not exist." 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

```
CREATE TABLE t1 (a int AUTO_INCREMENT PRIMARY KEY)
PARTITION BY LIST(a) (PARTITION p0 VALUES IN (1, 2));
INSERT INTO t1 VALUES (NULL),(NULL);
```

한 칼럼에 대한 제약조건이 100개 이상인 테이블에 INSERT 혹은 UPDATE 수행 시 서버 프로세스가 비정상 종료되는 현상(CUBRIDSUS-6437)

한 개의 칼럼에 대한 제약조건, 예를 들어, 외래 키 제약조건이 100개 이상인 테이블에 **INSERT** 혹은 **UPDATE** 수행 시 서버 프로세스가 비정상 종료되는 현상을 수정했다.

Windows 64 비트 환경에서 BLOB, CLOB 타입에 문자열 INSERT 시 오류(CUBRIDSUS-6111)

Windows 64 비트 환경에서 **BLOB**, **CLOB** 타입에 문자열을 **INSERT**하면 "ERROR: Cannot coerce '123' to type clob." 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

```
CREATE TABLE alltypes(blob_col BLOC, clob_col CLOB);
INSERT INTO alltypes VALUES ('123', '123')
```

REPLACE 문과 DELETE 문이 각각 다른 트랜잭션으로 수행될 때 데드락 현상이 발생할 수 있는 문제(CUBRIDSUS-5016)

REPLACE 문과 **DELETE** 문이 각각 다른 트랜잭션으로 수행될 때 데드락 현상이 발생할 수 있는 문제를 수정했다.

```
CREATE TABLE t1(a int);
CREATE UNIQUE INDEX ON t1 (a);
INSERT INTO t1 VALUES (1),(4),(7);

// set autocommit off
T1: REPLACE t1 (a) VALUE (6);
T2: REPLACE t1 (a) VALUE (7);
T1: DELETE FROM t1 WHERE a = 6;
T2: REPLACE t1 (a) VALUE (5);
```

REPLACE 문 사용 시 고유 키 위반 오류가 발생할 수 있는 문제(CUBRIDSUS-5786)

REPLACE 문 사용 시 고유 키 위반(unique key violation) 오류가 발생할 수 있는 문제를 수정했다.

```
CREATE TABLE t1(id1 VARCHAR(10) UNIQUE);
INSERT INTO t1 VALUES ('a'),('b'),('f'),('k');
COMMIT;

T1: DELETE FROM t1 WHERE id1 = 'f';
T1: INSERT INTO t1 VALUES ('g');
T2: INSERT INTO t1 VALUES ('f');
T2: COMMIT;
T1: ROLLBACK;
T3: REPLACE INTO t1 SET id1 = 'f';
```

DELETE 문에서 FROM 키워드의 생략이 가능해짐(CUBRIDSUS-6547)

DELETE 문에서 **FROM** 절 이하의 테이블이 하나인 경우 키워드의 생략이 가능하도록 수정했다.

```
DELETE tbl;
```

하나의 테이블에 UPDATE를 수행하는 동안 다른 스레드가 SELECT를 수행하면 잘못된 결과를 출력하는 현상(CUBRIDSUS-8460)

하나의 테이블에 **UPDATE**를 수행하는 동안 트랜잭션 격리 수준이 커밋되지 않은 읽기(uncommitted read)인 다른 스레드가 **SELECT**를 수행하면 잘못된 결과를 출력할 수 있는 문제를 수정했다.

UPDATE와 SELECT가 동시에 수행되는 상황에서 서버 에러 로그에 -46번 오류가 기록되는 현상(CUBRIDSUS-8347)

동일한 테이블에 대해 고립 수준이 커밋되지 않은 읽기(uncommitted read)인 **SELECT**와 **UPDATE**가 함께 수행되는 상황에서 “Internal error: slot 17 on page 166272 of volume xxx is not allocated."과 같은 -46번 오류가 발생할 수 있는 문제가 존재했으나, 해당 오류가 발생하더라도 질의는 정상적으로 진행되므로 오류가 아닌 경고를 출력하도록 수정했다.

INSERT 문을 반복 수행하는 도중에 다른 트랜잭션에 의한 테이블 잠금 발생 시에 트랜잭션 일관성 보장되지 못하는 문제(CUBRIDSUS-8303)

INSERT 문을 **PREPARE** 한 후에 반복적으로 수행하는 트랜잭션이 첫번째 실행 이후 커밋한 후에 다른 트랜잭션이 해당 테이블에 X-lock을 설정한 후에 다시 **INSERT**를 수행하게 될 때, 해당 테이블에 대한 IX-lock 부재로 인해 잠금 처리가 정확하게 되지 않아 결국 트랜잭션들의 일관성이 보장될 수 없는 문제를 수정했다.

이 문제는 2008 R4.1 이상 버전에서 발생할 수 있다.

다른 트랜잭션에 의해 삭제된 테이블에 대한 SELECT 질의가 오류를 출력하지 않고 0건을 반환하는 문제(CUBRIDSUS-7389)

다른 트랜잭션이 삭제한 테이블에 대해 **SELECT** 질의를 수행하면 해당 테이블을 찾을 수 없다는 오류를 출력해야 하지만 0 건을 반환하는 문제를 수정했다.

클릭 카운터로 인해 교착 상태가 발생하면 서버 프로세스가 동작을 멈출(hang) 수 있는 문제(CUBRIDSUS-7378)

클릭 카운터(click counter) 함수로 인해 교착 상태가 발생하면 가끔 서버 프로세스가 이를 감지하지 못하고 동작을 멈출 수 있는 문제를 수정했다. 클릭 카운터로 인해 교착 상태가 발생하면 클릭 카운터의 업데이트만 무시되고, 나머지 질의들은 정상적으로 진행되어야 한다.

두 개 이상의 트랜잭션이 동시에 일시적 임시 볼륨을 생성하거나 확장하려고 할 때 latch timeout 오류 발생(CUBRIDSUS-6667)

두 개 이상의 트랜잭션이 동시에 일시적 임시 볼륨(temporary temp volume)을 생성하거나 확장하려고 할 때 "LATCH ON PAGE(12345|0) TIMEDOUT" 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

CREATE TABLE ... LIKE 문으로 REUSE_OID 테이블을 복사 생성할 때 REUSE_OID 속성 복사가 누락되는 오류(CUBRIDSUS-7371)

CREATE TABLE ... LIKE 문으로 **REUSE_OID** 테이블을 복사 생성할 때, 새로 생성된 테이블에 **REUSE_OID** 속성이 누락되는 오류를 수정했다.

```
CREATE TABLE src_tbl (i INT) REUSE_OID;
CREATE TABLE dest_tbl LIKE src_tbl;
```

AUTO_INCREMENT 속성이 설정된 칼럼에 ALTER 문으로 DEFAULT 속성이 추가되는 문제(CUBRIDSUS-6407)

AUTO_INCREMENT 속성과 **DEFAULT** 속성이 같은 칼럼에 동시에 설정될 수 없음에도 불구하고 **ALTER** 문으로 **DEFAULT** 속성을 추가할 수 있었던 오류를 수정했다.

```
CREATE TABLE tbl (x INT AUTO_INCREMENT);
ALTER TABLE tbl ALTER COLUMN x SET DEFAULT 100;

-- 수정한 버전에서는 다음의 오류를 출력한다.
SHARED, DEFAULT and AUTO_INCREMENT cannot be defined with each other.
```

ALTER SERIAL 문으로 시리얼의 캐시 옵션 설정을 변경하지 못하는 문제(CUBRIDSUS-7120)

ALTER SERIAL 문으로 캐시 옵션을 활성화 또는 비활성화하지 못하는 문제를 수정했다.

```
CREATE SERIAL s2 START WITH 5 INCREMENT BY 6 CACHE 5;
ALTER SERIAL s2 NOCACHE;
```

기본 키 칼럼에 DEFAULT NULL 제약조건 부여가 가능한 문제(CUBRIDSUS-7348)

기본 키 칼럼에 **DEFAULT NULL** 제약조건을 부여하지 못하도록 수정했다.

```
// 이전 버전에서는 아래 질의가 수행되는 문제가 존재한다.
CREATE TABLE t(a INT PRIMARY KEY DEFAULT NULL);
```

상위 클래스와 서브 클래스에 둘 다 기본 키 선언이 가능한 문제(CUBRIDSUS-9207)

상위 클래스에 기본 키가 있으면 서브 클래스는 이미 상위 클래스의 기본 키를 상속받으므로 기본 키를 선언할 수 없어야 하나, 이를 허용했던 문제를 수정했다.

```
CREATE TABLE parent (a INT PRIMARY KEY);
CREATE TABLE children as subclass of parent (b INT PRIMARY KEY);
```

REUSE_OID 옵션으로 생성한 테이블을 DROP하는 경우 공간이 반환되지 않을 수 있는 현상(CUBRIDSUS-8140)

REUSE_OID 옵션을 가지는 테이블을 **DROP**하는 도중 서버에서 내부적으로 오류가 발생하는 경우에 해당 공간이 반환되지 못하는 문제를 수정했다.

호스트 변수에 값을 바인딩할 때 두 번째 바인딩하는 값의 타입이 첫 번째와 다르면 오류가 발생하는 문제(CUBRIDSUS-7377)

호스트 변수에 값을 바인딩할 때 두 번째 바인딩하는 값의 타입이 첫 번째 타입과 다르면 첫 번째 타입으로 바인딩하려고 시도하면서 오류가 발생될 수 있는 문제를 수정했다. 예를 들어 '**SELECT** ?' 질의에 처음에는 1을 바인딩하고, 이후에 'A'를 바인딩하면 'A'를 **INTEGER** 타입으로 바인딩하려고 시도하면서 오류가 발생했다.

호스트 변수 포함한 뷰에 대한 질의 수행 시에 서버 프로세스가 비정상 종료되는 현상(CUBRIDSUS-6767)

호스트 변수를 포함한 뷰에 대한 질의 수행 시에 서버 프로세스가 비정상 종료되는 현상을 수정했다. 호스트 변수를 포함한 DDL은 허용되지 않는다.

```
CREATE TABLE tree(node INT,parentnode INT,name VARCHAR(30));
INSERT INTO TREE VALUES (1, NULL,'1'),(2, 1,'2'),(3, 1,'3'),(4,2,'4');
PREPARE stmt FROM 'CREATE VIEW v as SELECT parentnode,node,name FROM tree WHERE node < ? START
WITH parentnode IS NULL CONNECT BY parentnode=PRIOR node ORDER BY node';
EXECUTE stmt USING 3;
SELECT * FROM v;
```

뷰 생성 시 DEFAULT 값이 뷰에 복사되지 않는 오류(CUBRIDSUS-6772)

CREATE VIEW AS SELECT ... 문 수행 시 원본 테이블의 **DEFAULT** 값이 뷰에 복사되지 않는 오류를 수정했다.

```
CREATE TABLE foo(a INT DEFAULT 0);
CREATE VIEW voo AS SELECT * FROM foo;
INSERT INTO voo VALUES(DEFAULT);
```

뷰를 구성하는 질의에 FROM 절이 없는 경우 SELECT 질의 수행 시 오류 (CUBRIDSUS-6592)

뷰를 구성하는 질의에 **FROM** 절이 없으면 **SELECT** 질의 수행 시 "ERROR: There are more attributes in class v1 than columns in the query specification." 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

```
CREATE VIEW v1 AS (SELECT 1 a) UNION (SELECT 1 a);
SELECT * FROM v1;
```

뷰 생성 또는 뷰 변경 시 SELECT 리스트에 복잡한 수식이 나타나는 경우 질의 수행에 실패할 수 있는 문제(CUBRIDSUS-8970)

뷰 생성 또는 뷰 변경 시 **SELECT** 리스트에 복잡한 수식이 포함된 경우 질의 수행에 실패할 수 있는 문제를 수정했다. 다음 예처럼 **SELECT** 리스트 중 복잡한 수식에 대해 칼럼 별칭(alias)을 명시하지 않으면 "SQRT(CAST(100 as double))"이 칼럼 별칭이 되는데, 이와 같은 수식을 식별자로 허용하지 않으므로 식별자를 감싸는 부호(큰따옴표, 대괄호 혹은 백틱)가 필요하다. 수정 이후에는 질의 수행 시 내부적으로 식별자를 감싸는 부호를 추가하여 해당 현상이 발생하지 않는다.

```
CREATE VIEW vw AS SELECT SQRT(CAST(100 as double));
// 위의 질의를 수행하면 내부적으로 아래와 같이 수행된다.
CREATE VIEW vw AS SELECT SQRT(CAST(100 as double)) AS [SQRT(CAST(100 as double))];
```

수정 이전 버전의 HA 환경에서는 마스터 노드에서 해당 현상이 발생하지 않게 질의문을 작성하더라도, 슬레이브 노드에서 함수 내의 인자 값을 CAST하도록 질의문이 재작성되면서 해당 현상이 발생했다.

```
// 아래의 질의문을 수행하면 슬레이브 노드에서는 위 예제의 첫번째 질의 형태로 재작성된다.
CREATE VIEW vw AS SELECT SQRT(100);
```

db_user, db_trigger 시스템 카탈로그 테이블의 레코드를 사용자가 임의로 변경할 수 있는 오류 (CUBRIDSUS-8690)(CUBRIDSUS-8692)

사용자가 임의로 **db_user** 와 **db_trigger** 시스템 카탈로그 테이블의 레코드를 임의로 변경할 수 있는 오류를 수정했다.

자신을 참조하는 외래 키가 있는 테이블에 값 입력 시 오류(CUBRIDSUS-6216)

자신을 참조하는 외래 키가 있는 테이블에 값을 입력하면 "ERROR: The constraint of the foreign key 'fk_pkfk_b' is invalid." 오류 메시지와 함께 질의 수행에 실패하는 문제를 수정했다.

칼럼의 DEFAULT 값 크기가 데이터베이스 페이지 크기를 초과하는 테이블 생성 시 서버 프로세스가 비정상 종료되는 현상(CUBRIDSUS-6510)

칼럼의 **DEFAULT** 값 크기가 데이터베이스 페이지 크기를 초과하는 테이블을 생성하면 서버 프로세스가 비정상 종료되는 현상을 수정했다.

사용자 이름의 길이가 32 bytes를 초과하면 해당 사용자로 로그인할 수 없는 문제(CUBRIDSUS-6633)

사용자 이름의 길이가 32 bytes를 초과하면 이후 해당 사용자로 접속하지 못하는 문제가 발생했다. 수정된 버전에서는 DB 사용자 계정 생성 시 이름의 길이를 최대 32 bytes로 제한했다.

```
CREATE USER a12345678901234567890123456789012345678901234567890;

$ csql -u a12345678901234567890123456789012345678901234567890123456
ERROR: User "a123456789012345678901234567890123456789012345" is invalid.
```

CONNECT BY 절이 START WITH 절 뒤에 오는 것을 허용하게 됨(CUBRIDSUS-6548)

계층 질의문에서 **CONNECT BY** 절이 **START WITH** 절 뒤에 오는 것도 허용하도록 수정했다.

```
SELECT *
FROM a
CONNECT BY PRIOR id = pid
START WITH id = 1;
```

ORDER SIBLINGS BY 절을 가진 계층 질의가 부질의로 사용되면 잘못된 질의 결과를 출력하는 문제(CUBRIDSUS-7748)

ORDER SIBLINGS BY 절이 포함된 계층 질의가 부질의로 사용되면 질의 최적화기에서 질의가 잘못 재작성되어 질의 결과가 잘못되는 문제를 수정했다.

```
SELECT *
FROM tbl
WHERE id IN (SELECT id
             FROM tbl
             WHERE yn = 'Y'
             START WITH id = '100002'
             CONNECT BY NOCYCLE PRIOR id = pid
             ORDER SIBLINGS BY sort_col);
```

ROWNUM 조건이 있는 계층 질의에서 질의 결과의 개수가 틀린 현상(CUBRIDSUS-6365)

ROWNUM 조건이 있는 계층 질의가 인덱스 스캔으로 처리될 때 질의 결과의 개수가 **ROWNUM**조건에 의해 제한되지 못하는 현상을 수정했다.

```
SELECT oid
FROM foo
WHERE ROWNUM = 1
START WITH oid = '1234567'
CONNECT BY PRIOR poid = oid;
```

일시적 임시 볼륨을 사용하는 질의 수행 중 DB 서버 재시작 시 서버 프로세스가 비정상 종료될 수 있는 문제(CUBRIDSUS-5684)

일시적 임시 볼륨을 사용하는 질의를 수행하는 도중에 데이터베이스를 재시작하면 서버 프로세스가 비정상 종료될 수 있는 문제를 수정했다.

자동 볼륨 증가 도중 실패한 경우 조치 후에도 질의 수행에 실패하는 문제(CUBRIDSUS-7216)

자동 볼륨 증가 도중 서버 프로세스가 비정상 종료되어 자동 볼륨 증가에 실패한 중간 파일이 있으면 서버 재시작 등의 조치 후에도 질의 수행에 실패하는 문제를 수정했다.

트리거 수행에 실패하는 현상(CUBRIDSUS-7187)

다음 예와 같이 트리거를 수행하는 경우 오류를 발생하거나 응용 프로그램이 비정상 종료되면서 수행에 실패하는 현상을 수정했다.

Case 1) 이전 버전에서 다음과 같은 질의를 수행하면 오류가 발생했다.

```
CREATE TABLE t1(a INT);
INSERT INTO t1 VALUES(1), (2), (3), (4);
CREATE TRIGGER TRIGG1 BEFORE UPDATE ON t1 EXECUTE DELETE FROM t1;

UPDATE t1 SET a=6;
ERROR: Error evaluating action for "trigg1", Accessing deleted object 0|1100|16.
```

Case 2) 이전 버전에서 다음과 같은 질의를 수행하면 응용 프로그램이 비정상 종료되었다.

```
CREATE TABLE t1(a INT, b INT);
INSERT INTO t1 VALUES (2,2), (3,3);
CREATE VIEW v1 AS SELECT * FROM t1;
CREATE TRIGGER tri_t1_before_update1 BEFORE UPDATE ON t1 EXECUTE DELETE FROM v1;
CREATE TRIGGER tri_t2_before_update2 BEFORE UPDATE ON t1 EXECUTE DELETE FROM t1;
```

```
--Test: crash happens.
UPDATE t1 SET t1.a=10;
```

다른 사용자가 생성한 테이블에 트리거가 있으면 해당 테이블에 대한 질의 오류가 무시되는 현상 (CUBRIDSUS-7336)

다른 사용자가 생성한 테이블에 트리거가 있는 경우 해당 테이블에 대한 질의 수행에 실패해도 오류가 발생하지 않고, 이후 트랜잭션이 종료될 때 "ERROR: Internal system failure: no more specific information is available." 오류가 발생하는 문제를 수정했다.

트리거에 의한 UPDATE/INSERT 질의가 실패했음에도 불구하고 트리거를 유발한 질의가 롤백되지 않는 문제(CUBRIDSUS-7239)

자동 커밋 모드가 OFF일 때 트리거에 의한 **UPDATE/INSERT** 질의가 실패했음에도 불구하고, 트리거를 유발한 질의가 롤백되지 않는 문제를 수정했다. 아래 예에서, Q6을 수행하면 t2 테이블의 trig1 트리거가 수행되는데, 이 때 t1 테이블에 **int** 타입의 값을 입력할 수 없으므로 에러가 발생하고 이 트리거를 유발한 Q6번은 롤백된 상태여야 한다. 즉, Q7 수행 이후에 t1과 t2 테이블의 레코드 건수는 각각 1, 0건이어야 한다. 이전 버전에서는 에러가 발생하면 trig1 트리거를 유발한 Q6 질의가 롤백되지 않는 문제가 존재했다.

```
;autocommit off

CREATE TABLE t1(col1 date); -- Q1
CREATE TABLE t2(col2 int); -- Q2
CREATE TRIGGER trig1 AFTER INSERT ON t2 EXECUTE INSERT INTO t1(col1) VALUES(obj.col2); -- Q3
COMMIT; -- Q4

INSERT INTO t1(col1) VALUES ('2012-04-30'); -- Q5
INSERT INTO t2(col2) VALUES (1); -- Q6

ERROR: Error evaluating action for "trig1", Execute: Cannot coerce obj.cold2 to type date.

COMMIT; -- Q7
```

트리거 내에서 CLOB_TO_CHAR 함수를 호출하면 NULL을 반환하는 오류 수정(CUBRIDSUS-7246)

트리거 내에서 **CLOB_TO_CHAR** 함수를 호출하면 **NULL**을 반환하는 오류를 수정했다.

질의 계획 및 최적화

질의 계획 캐시에 설정한 제한 개수보다 많은 질의 계획이 저장되는 문제(CUBRIDSUS-8619)

시스템 파라미터 **max_plan_cache_entries**를 통해 질의 계획 캐시(plan cache)에 설정한 제한 개수를 초과하여 질의 계획이 저장되는 문제를 수정했다. 이 문제로 인해서 서버의 메모리 및 임시 볼륨 사용량이 꾸준히 증가하는 현상이 있었다.

질의 최적화 과정에서 ORDER BY 절이 제거될 때 질의문에 ORDERBY_NUM()이 있으면 질의 결과가 0건인 문제(CUBRIDSUS-6060)

질의 최적화 과정에서 불필요한 **ORDER BY** 절을 제거하는데 이 때 orderby_num()은 그대로 남아있어 값이 항상 0으로 출력되는 오류를 수정했다.

```
SELECT ORDERBY_NUM(), history.*
FROM history
WHERE host_year = '2004'
ORDER BY host_year FOR ORDERBY_NUM() BETWEEN 1 AND 10;
```

질의 계획 캐시를 사용 안 하도록 설정하고 PREPARE 문 수행 시 응용 프로그램이 비정상 종료되는 현상(CUBRIDSUS-8094)

시스템 파라미터 **max_plan_cache_entries**의 값을 -1로 설정하여 질의 계획 캐시를 사용 안하는 경우에 **PREPARE** 문을 수행하면 응용 프로그램이 비정상 종료되는 현상을 수정했다.

```
CREATE TABLE t(a INT);
INSERT INTO t VALUES (1), (2), (3);
PREPARE STMT FROM 'SELECT COUNT(?) FROM t';
EXECUTE STMT USING 1;
```

질의 계획 정보 출력 이후 질의 수행 시 잘못된 결과를 출력할 수 있는 문제(CUBRIDSUS-7818)

;plan simple 명령어를 실행하여 질의 계획을 출력하면 입력 값에 대응되는 칼럼의 타입으로 입력 값이 변환되어 잘못된 결과를 출력할 수 있는 문제를 수정했다.

아래의 예제에 대해서 수정 이전 버전에서는 계획 정보를 출력하면서 2.3이라는 입력 값이 2로 변환되어 서버에 전달되어 결국 2가 결과에 포함되지 못하는 문제가 있었다..

```
CREATE TABLE foo (col INT);
INSERT INTO foo VALUES (1),(2);
;plan simple
```

```
SELECT * FROM foo WHERE col < 2.3;
```

질의 계획 캐시를 사용하지 않는 질의에서 테이블 이름을 변경한 후 질의 수행 시 이전 테이블에 대해서 질의가 수행되는 문제(CUBRIDSUS-7637)

테이블 이름을 변경한 후 질의 수행 시 해당 이름을 가지는 새 테이블이 아니라 변경된 이전 테이블에 대하여 질의가 수행되는 문제를 수정했다. 이전 버전에서는 **INSERT**문은 질의 계획 캐시를 사용하지 않으므로 항상 문제가 발생했으며, 나머지 질의의 경우 시스템 파라미터 **max_plan_cache_entries**의 값을 -1로 설정하여 질의 계획 캐시 기능을 끄는 경우에 문제가 발생했다.

```
// insert의 예
INSERT INTO tbl VALUES (...);
RENAME TABLE tbl AS tbl_old;
RENAME TABLE tbl2 AS tbl;
// 아래 질의 수행 시 새로운 테이블이 아닌 tbl_old에 값이 INSERT됨.
INSERT INTO tbl VALUES (...);
```

특정 OUTER JOIN 질의에 대해서 커버링 인덱스 스캔 질의 실행 시에 발생할 수 있는 오류(CUBRIDSUS-7868)

OUTER JOIN을 포함한 특정 질의에 대해서 커버링 인덱스 스캔 질의 실행 계획이 잘못되어 실행 과정에서 "Query execution failure #10946." 오류가 발생하는 문제를 수정했다.

```
SELECT foo.obj_id, foo.h_id
FROM foo
INNER JOIN table_j ON foo.host_id = table_j.host_id
LEFT OUTER JOIN table_d ON foo.s_id = table_d.svc_no
LEFT OUTER JOIN table_g ON foo.g_id = table_g.svr_grp_no
WHERE table_d.svc_no = foo.s_id
AND foo.s_id = '2152';
```

IN 절에 호스트 변수가 있으면 질의 실행 계획이 출력되지 않는 오류(CUBRIDSUS-6382)

IN 절이 (?, ?, ...) 형태의 호스트 변수를 포함하면 질의 실행 계획이 출력되지 않는 오류를 수정했다.

```
csql> ;plan detail
csql> SELECT * FROM tbl WHERE id IN (?, ?, ?);
```

AND와 OR 조건이 여러 개 결합된 질의 수행 시 조건 일부가 누락되어 잘못된 결과를 출력할 수 있는 문제(CUBRIDSUS-9193)

AND와 **OR** 조건이 여러 개 결합된 질의 수행 시 조건 일부가 누락된 결과를 출력할 수 있는 문제를 수정했다.

```
SELECT *
FROM it, p
WHERE
(
  p_pkey = l_pkey
  and p_br = 'Br12'
  and p_ct in ('CS', 'BX', 'PK', 'PG')
  and l_qty >= 1 and l_qty <= 1 + 10
  and p_sz between 1 and 5
  and l_sm in ('A', 'RG')
  and l_st = 'DIP'
)
OR
(
  p_pkey = l_pkey
  and p_br = 'Br12'
  and p_ct in ('MG', 'MB', 'MPK', 'MPC')
  and l_qty >= 10 and l_qty <= 10 + 10
  and p_sz between 1 and 10
  and l_sm in ('A', 'RG')
  and l_st = 'DIP'
)
OR
(
  p_pkey = l_pkey
  and p_br = 'Br12'
  and p_ct in ('LG', 'LB', 'LPK', 'LPC')
  and l_qty >= 20 and l_qty <= 20 + 10
  and p_sz between 1 and 15
  and l_sm in ('A', 'RG')
  and l_st = 'DIP'
);
```

AUTO_INCREMENT가 포함된 칼럼 또는 테이블이 삭제될 때 관련 질의문이 질의 계획 캐시에서 삭제되지 않는 문제(CUBRIDSUS-7872)

AUTO_INCREMENT가 포함된 칼럼 또는 테이블이 삭제될 때 관련 질의문이 질의 계획 캐시에서 삭제되지 않아 이후 해당 **AUTO_INCREMENT**와 같은 이름으로 시리얼을 생성하여 해당 시리얼로 질의를 수행하면 "ERROR: Cannot fetch serial object." 오류 메시지와 함께 수행에 실패하는 현상을 수정했다.

```
CREATE TABLE tbl (a INT AUTO_INCREMENT);
SELECT tbl_ai_a.NEXT_VALUE;
DROP TABLE tbl;

CREATE SERIAL tbl_ai_a;
SELECT tbl_ai_a.NEXT_VALUE;
```

질의 수행 없이 계획만 생성하도록 최적화 수준을 설정하고 REPLACE 문 수행 시 응용 프로그램이 비정상 종료되는 현상(CUBRIDSUS-6614)

OPTIMIZATION LEVEL을 질의 수행 없이 계획만 생성하도록 2, 258 혹은 514 중 하나로 설정하고 **REPLACE** 문을 수행하면 응용 프로그램이 비정상 종료되는 현상을 수정했다.

```
SET OPTIMIZATION LEVEL 2;

DROP TABLE t;
CREATE TABLE t( col1 INTEGER UNIQUE, col2 VARCHAR(128));
INSERT INTO t(col1, col2) VALUES (17, 'operators list');
REPLACE t(col1, col2) VALUES (17, 'personnel list');
```

뷰 재생성 이후 해당 뷰의 UPDATE 수행에 실패하는 현상(CUBRIDSUS-6942)

뷰를 재생성한 이후 **UPDATE**할 때 이전과 같은 **UPDATE** 문을 수행했을 경우, 이전 질의 계획을 사용하여 **DROP**했던 뷰를 참조하면서 **UPDATE**에 실패하는 현상을 수정했다.

```
CREATE TABLE foo(a int);
INSERT INTO foo VALUES(3);

CREATE VIEW v AS SELECT * FROM foo WHERE a < 2;
UPDATE v SET a = 3;
DROP VIEW v;

CREATE VIEW v AS SELECT * FROM foo WHERE a < 2;
UPDATE v SET a = 3;
```

ORDER BY 최적화나 GROUP BY 최적화가 가능한 질의 계획들 중 최선을 선택하도록 수정(CUBRIDSUS-6957)

ORDER BY 최적화(skip ORDER BY) 또는 **GROUP BY** 최적화(skip GROUP BY)가 가능한 질의 계획들 중에서 더 나은 계획이 있음에도 불구하고 전체 인덱스 스캔(full index scan)을 선택하는 문제를 개선했다.

LIMIT 0인 경우 곧바로 결과를 반환하도록 수정(CUBRIDSUS-7420)

LIMIT 0을 포함한 질의는 곧바로 결과를 반환하도록 수정했다. 이전 버전에서는 질의를 수행한 후에 **LIMIT** 절을 평가하므로 경우에 따라서 질의 처리에 많은 시간이 소요되었다.

```
SELECT CAST(dt_col AS DATE)
FROM article
WHERE id = '001' AND dt_col < TO_DATE('20120201', 'YYYYMMDD')
ORDER BY CAST(dt_col AS DATE) DESC
LIMIT 0;
```

분할

분할 테이블에 로컬 인덱스 사용할 수 있도록 개선(CUBRIDSUS-7629)

시스템 내부적으로 정의된 조건에 따라 분할 테이블의 각 분할마다 독립적으로 인덱스를 구성하는 로컬 인덱스를 생성할 수 있도록 수정했다. 모든 외래 키와 모든 비고유 인덱스는 로컬 인덱스이며, 고유 인덱스는 분할 키가 고유 인덱스에 속하는 경우에만 로컬 인덱스이다. 이전 버전에서는 기본 키를 비롯한 고유 인덱스는 모두 글로벌 인덱스로만 관리되었지만, 9.0 Beta부터는 분할 키가 포함되면 로컬 인덱스로 관리된다. 로컬 인덱스를 이용하면 글로벌 인덱스를 이용하는 경우보다 성능이 향상된다.

분할 테이블의 INSERT 성능 향상(CUBRIDSUS-6018)

분할 테이블에 대한 **INSERT** 성능을 향상시켰다.

분할 테이블에 TRUNCATE 문을 허용하도록 개선(CUBRIDSUS-6871)

분할 테이블에 대해 **TRUNCATE** 문을 허용하도록 개선했다.

자동 커밋 모드가 OFF이면 영역 분할 테이블의 SELECT 결과가 잘못 출력되는 오류(CUBRIDSUS-7127)

자동 커밋 모드가 OFF일 때, 영역 분할 테이블에 **INSERT** 하는 도중에 **SELECT**하고 추가로 **INSERT**한 후 다시 **SELECT**하면, 추가로 **INSERT**했던 레코드가 출력되지 않는 현상을 수정했다.

```
csql> ; autocommit off
CREATE TABLE foo (id1 BIGINT) PARTITION BY RANGE(id1) (
  PARTITION p1 VALUES LESS THAN (3),
  PARTITION p2 VALUES LESS THAN (5),
  PARTITION p3 VALUES LESS THAN (8)
);
INSERT INTO foo VALUES (1);

// 수정 이전 버전에서는 Q1 수행 후 Q2 수행하면 Q1 수행 이전에 INSERT한 1건의 레코드만 출력되는 현상이 나타났다.
SELECT * FROM foo; -- Q1
INSERT INTO foo VALUES (1);
INSERT INTO foo VALUES (1);
SELECT * FROM foo; -- Q2
```

영역 분할 테이블에서 SELECT 수행 시 IS NULL OR IS NOT NULL 조건이 존재하면 잘못된 결과를 출력하는 문제(CUBRIDSUS-4575)

영역 분할 테이블에서 **SELECT** 수행 시 **IS NULL OR IS NOT NULL** 조건이 포함되면 잘못된 결과를 출력하는 문제를 수정했다.

```
SELECT *
FROM tbl
WHERE a IS NULL OR a IS NOT NULL;
```

분할 테이블에서 OUTER JOIN 수행 시 잘못된 질의 결과를 출력하는 현상(CUBRIDSUS-6888)

분할 테이블에서 **OUTER JOIN** 수행 시 잘못된 질의 결과를 출력하는 현상을 수정했다.

```
CREATE TABLE pt1(i int,j int) PARTITION BY HASH(i) PARTITIONS 4;
CREATE TABLE pt2(i int,j int) PARTITION BY HASH(i) PARTITIONS 4;

INSERT INTO pt1 VALUES (1,1),(2,2),(3,3);
INSERT INTO pt2 VALUES (1,1);
CREATE INDEX idx ON pt2(i);
```

```
// 아래 질의 수행 시 수정 이전 버전에서는 첫번째 레코드가 "1, 1, NULL, NULL"로 잘못 출력되었다.
SELECT * FROM pt1 LEFT JOIN pt2 ON pt1.i=pt2.i USING INDEX idx(+);

i j i j
=============================================
1 1 1 1
2 2 NULL NULL
3 3 NULL NULL
```

분할 테이블에서 외래 키 생성 시 제약 조건을 검사하지 않는 문제(CUBRIDSUS-6916)

분할 테이블에서 외래 키 생성 시 제약 조건을 검사하지 않는 문제를 수정했다.

```
CREATE TABLE t1(i INT PRIMARY KEY);
INSERT INTO t1 VALUES(1);
CREATE TABLE t2(i int,j int) PARTITION BY HASH(i) PARTITIONS 4;
INSERT INTO t2 VALUES (2,2);
ALTER TABLE t2 ADD CONSTRAINT FOREIGN KEY t2(i) REFERENCES t1(i);
```

AUTO_INCREMENT 칼럼을 분할 키로 하여 해시 분할한 테이블에 데이터 입력 시 하나의 분할 테이블에만 입력되는 문제(CUBRIDSUS-5622)

AUTO_INCREMENT 칼럼을 분할 키로 하여 해시 분할한 테이블에 데이터를 입력하면 하나의 분할 테이블에만 입력되는 문제를 수정했다.

```
CREATE TABLE t(i INT AUTO_INCREMENT) PARTITION BY HASH(i) PARTITIONS 3;
INSERT INTO t(i) VALUES (NULL), (NULL), (NULL), (NULL), (NULL), (NULL), (NULL), (NULL);
```

리스트 분할 테이블에서 분할 키 값을 함수로 입력하면 분할 프루닝이 동작하지 않는 문제(CUBRIDSUS-5777)

리스트 분할 테이블에서 분할 키 값을 함수로 입력하면 질의 결과에 포함되지 않는 분할을 검색 조건에서 제외시켜주는 분할 프루닝(partition pruning)이 동작하지 않는 문제를 수정했다.

하위 테이블에서 분할 테이블을 생성하는 문제(CUBRIDSUS-8703)

하위 테이블(child table)의 분할 키로 사용한 칼럼이 상위 테이블(parent table)에 있음에도 불구하고 상위 테이블의 삭제가 가능한 문제가 존재했으나, 이러한 오류를 근본적으로 방지하기 위해 하위 테이블에서 분할 테이블을 생성할 수 없도록 수정했다.

CLOB 칼럼을 가지는 분할 테이블에 대해서 생성된 뷰로 레코드를 삭제하지 못하는 현상(CUBRIDSUS-8216)

CLOB 칼럼을 가지는 분할 테이블에 대해서 생성된 뷰로 레코드 삭제하는데 오류가 발생하는 현상을 수정했다.

```
CREATE TABLE t (i INT, c CLOB) PARTITION BY HASH(i) PARTITIONS 4;
INSERT INTO t SELECT ROWNUM, '1' FROM db_class LIMIT 40;
CREATE VIEW v AS SELECT * FROM t;

// 수정 이전 버전에서 다음 질의를 수행하면 "ERROR: Semantic: ..." 오류가 발생한다.
DELETE FROM v;
```

HA 환경에서 분할 테이블 생성 시 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-7724)

HA 환경에서 분할 테이블을 생성할 때 분할 테이블 생성문이 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
CREATE TABLE t1 (a int PRIMARY KEY)
PARTITION BY HASH (a) PARTITIONS 2;
```

분할 테이블에서 INSERT한 레코드가 트리거의 UPDATE 문으로 다른 파티션으로 이동해야 할 때 발생하는 오류(CUBRIDSUS-7520)

분할 테이블에서 **INSERT**한 레코드가 트리거의 **UPDATE**문으로 인해 **INSERT**한 분할(partition)과 다른 곳으로 이동하지 못하면서 "ERROR: Error evaluating action for "tr2", Not allowed access to partition." 오류가 발생하는 현상을 수정했다.

```
CREATE TABLE emp (store_id INT NOT NULL)
PARTITION BY RANGE (store_id) (
  PARTITION P1 VALUES LESS THAN (10),
  PARTITION P2 VALUES LESS THAN (20));
CREATE TRIGGER tr2 AFTER INSERT ON emp EXECUTE UPDATE emp SET store_id = store_id + 10 WHERE
store_id = obj.store_id;

INSERT INTO emp VALUES (5);
```

인덱스가 있는 분할 테이블을 생성, 롤백한 이후 재생성을 시도하면 실패하는 현상(CUBRIDSUS-7560)

인덱스가 있는 분할 테이블을 생성하고 롤백한 이후 같은 분할 테이블을 재생성하려고 시도하면 "ERROR: Partition failed." 오류를 출력하면서 실패하는 현상을 수정했다.

분할 테이블의 데이터 파일을 가지고 loaddb 수행 시 발생하는 오류(CUBRIDSUS-5815)

분할 테이블을 포함하는 데이터 파일을 가지고 loaddb 수행 시 분할 테이블 위치에 데이터가 존재하면 "Partitioned failed" 오류 메시지와 함께 수행에 실패하는 현상을 수정했다.

```
// 분할 테이블
CREATE TABLE record (
  host_year INTEGER NOT NULL,
  event_code INTEGER NOT NULL,
  score VARCHAR(20))
PARTITION BY RANGE (host_year)
(PARTITION before_1996 VALUES LESS THAN (1996),
 PARTITION after_1996 VALUES LESS THAN MAXVALUE);

// loaddb 용 데이터 파일. class [record] 이하에 데이터가 존재할 수 있도록 수정되었다.
%class [record] ([host_year] [event_code] [score])
1992 20288 '22.3'
2000 20101 '04:04.5'
2004 20341 '272.5'
```

영역 분할 재정의 수행 시 기존 분할에 LESS THAN MAXVALUE 조건이 있으면 재정의에 실패하는 오류(CUBRIDSUS-6379)

영역 분할 재정의 수행 시 기존 분할에 **LESS THAN MAXVALUE** 조건이 있으면 재정의에 실패하는 문제를 수정했다.

```
CREATE TABLE t1(id int)
PARTITION BY RANGE (id)
(PARTITION l10 VALUES LESS THAN (10),
 PARTITION l100 VALUES LESS THAN (100),
 PARTITION lar VALUES LESS THAN MAXVALUE);

ALTER TABLE t1 REORGANIZE PARTITION l100, lar
INTO (PARTITION lar VALUES LESS THAN MAXVALUE);
```

리스트 분할 테이블에서 IN 조건절에 대해 분할 프루닝 오류(CUBRIDSUS-9189)

리스트 분할 테이블에서 **IN** 조건절에 2개 이상의 분할이 포함되면 분할 프루닝 과정이 무한 반복될 수 있는 오류를 수정했다.

외래 키가 있는 분할 테이블에서 DELETE CASCADE가 동작하지 않는 현상(CUBRIDSUS-8085)

DELETE CASCADE 동작을 지정한 외래 키가 있는 분할 테이블이 존재할 때, 기본 키가 있는 테이블의 레코드를 삭제하면 외래 키가 있는 분할 테이블 쪽에서 "ERROR: The instance having the foreign key 'f' cannot be dropped. " 에러가 발생하는 현상을 수정했다.

```
CREATE TABLE dp(i INT PRIMARY KEY);
INSERT INTO dp SELECT ROWNUM FROM db_class LIMIT 40;

CREATE TABLE f(i INTEGER , orderdatekey INTEGER NOT NULL)
PARTITION BY range(i) (
  PARTITION p0 VALUES LESS THAN (200), PARTITION p1 VALUES LESS THAN (400),
  PARTITION p2 VALUES LESS THAN (600), PARTITION p3 VALUES LESS THAN MAXVALUE);

ALTER TABLE f ADD CONSTRAINT FOREIGN KEY f(i) REFERENCES dp(i) ON DELETE CASCADE;
INSERT INTO f SELECT ROWNUM , ROWNUM FROM db_class LIMIT 40;

-- record (4) should be deleted in both table dp and f;
DELETE FROM dp WHERE i = 4;
```

AUTO_INCREMENT 칼럼으로 영역 분할된 테이블이 있는 데이터베이스를 백업한 이후 SELECT 수행 시 서버 프로세스가 비정상 종료되는 현상(CUBRIDSUS-6390)

AUTO_INCREMENT 칼럼으로 영역 분할된 테이블이 있는 데이터베이스를 백업한 이후 **SELECT**를 수행하면 서버 프로세스가 비정상 종료되는 현상을 수정했다.

```
CREATE TABLE u1(id INT AUTO_INCREMENT,v VARCHAR(5))
PARTITION BY RANGE(id)
  (PARTITION l5 VALUES LESS THAN (5),
   PARTITION l10 VALUES LESS THAN (10),
   PARTITION l20 VALUES LESS THAN (20),
   PARTITION l100 VALUES LESS THAN (100));
INSERT INTO u1 VALUES (NULL,''),(6,''),(NULL,'');

$ cubrid backupdb testdb
```

AUTO_INCREMENT 칼럼으로 영역 분할된 테이블에 질의 수행 시 서버 프로세스가 비정상 종료되는 문제(CUBRIDSUS-6373)

AUTO_INCREMENT 칼럼으로 영역 분할된 테이블에 질의를 수행하면 서버 프로세스가 비정상 종료되는 문제를 수정했다.

```
CREATE TABLE u (id INT AUTO_INCREMENT,v VARCHAR(5))
PARTITION BY RANGE (id)
(PARTITION l5 VALUES LESS THAN (5),
 PARTITION l10 VALUES LESS THAN (10),
 PARTITION l20 VALUES LESS THAN (20),
 PARTITION l100 VALUES LESS THAN (100));

INSERT INTO u VALUES (0,';jf;dfj iouer'),(5,' fdfd'),(10,'dfd '),(15,'fdf'),(20,'a');
```

테이블의 분할에 대한 통계 정보가 업데이트되지 않아 인덱스 스캔이 되지 않는 현상 (CUBRIDSUS-7741)

UPDATE STATISTICS 문으로 분할 테이블의 통계 정보 업데이트 시에 테이블의 각 분할에 대한 통계 정보가 업데이트되지 않는 오류로 인해, 검색 질의 수행 시 인덱스 스캔이 되지 않는 현상을 수정했다.

자동 커밋 모드가 OFF일 때 분할 테이블에서 값을 삭제한 이후 OID로 해당 레코드를 검색하면 값이 출력되는 현상(CUBRIDSUS-2183)

자동 커밋 모드가 OFF일 때 분할 테이블에서 값을 삭제한 이후 OID(Object ID)로 해당 레코드를 검색하면 삭제 이전의 값이 출력되는 현상을 수정했다.

```
csql> ;autocommit off

CREATE TABLE tbl (id INT)
PARTITION BY RANGE (id + 1)
(PARTITION p0 VALUES LESS THAN (2),
 PARTITION p1 VALUES LESS THAN MAXVALUE);

INSERT INTO tbl VALUES (-1);
INSERT INTO tbl values (3);
INSERT INTO tbl values (99) into :xx;

SELECT :xx.id FROM db_root;
COMMIT;

SELECT :xx.id FROM db_root;
DELETE FROM tbl WHERE id = 99;
SELECT :xx.id FROM db_root;

// 수정 이전 버전에서는 위에서 레코드를 삭제했는데도 아래 질의를 수행하면 삭제 이전의 값이
출력된다.
```

```
SELECT :xx.id FROM db_root;
```

ALTER TABLE ... PARTITION 문으로 테이블 재분할 시 정상 수행되지 않는 문제(CUBRIDSUS-1408)

ALTER TABLE ... PARTITION 문으로 테이블 재분할을 수행하면 기존의 분할 정보가 제거되지 않고 유지되는 문제를 수정했다.

```
CREATE TABLE participant2
(host_year INT, nation CHAR(3), gold INT, silver INT, bronze INT)
PARTITION BY RANGE (host_year)
 (PARTITION before_2000 VALUES LESS THAN (2000),
  PARTITION before_2008 VALUES LESS THAN (2008));

ALTER TABLE participant2 REORGANIZE PARTITION before_2000 INTO (
  PARTITION before_1996 VALUES LESS THAN (1996),
  PARTITION before_2000 VALUES LESS THAN (2000)
);
```

loaddb 유틸리티로 많은 분할을 가진 테이블의 생성을 시도하면 분할이 생성되지 않는 문제(CUBRIDSUS-5833)

loaddb 유틸리티로 많은 분할을 가진 테이블을 생성하려고 시도하면 정상적으로 분할이 생성되지 않는 문제를 수정했다.

```
ALTER CLASS [col_day]
PARTITION BY LIST ( [yyyymmdd] )
(PARTITION [p_20110701] VALUES IN ('20110701'),
 PARTITION [p_20110702] VALUES IN ('20110702'),
 PARTITION [p_20110703] VALUES IN ('20110703'),
 PARTITION [p_20110704] VALUES IN ('20110704'),
 PARTITION [p_20110705] VALUES IN ('20110705'),
 PARTITION [p_20110706] VALUES IN ('20110706'),
 PARTITION [p_20110707] VALUES IN ('20110707'),
 ....
 PARTITION [p_20111231] VALUES IN ('20111231'));
```

인덱스

인덱스 생성 후 데이터를 삭제한 테이블에서 MAX 함수나 ORDER BY DESC를 사용한 질의 수행 시 발생하는 오류(CUBRIDSUS-7988)

테이블에 데이터가 존재하는 상태에서 인덱스를 생성하고 데이터를 모두 삭제한 이후, 이 테이블에 **MAX** 함수나 **ORDER BY DESC** 절을 사용하는 등 인덱스의 마지막 리프 페이지를 찾는 질의를 수행하면 "ERROR: An I/O error occurred while reading page 65536 of volume (null).... Bad file descriptor" 오류가 발생하는 현상을 수정했다.

```
CREATE TABLE tb2 (col1 INT PRIMARY KEY, col2 VARCHAR(16));
INSERT INTO tb2 VALUES (1, '1');
CREATE INDEX i_tb2 ON tb2(col1, col2);
DELETE FROM tb2;
SELECT * FROM tb2 ORDER BY 1 DESC,2 DESC;
```

PREFIX 인덱스가 생성된 칼럼에서 집계 함수 수행 시 오동작(CUBRIDSUS-6579)

PREFIX 인덱스가 생성된 칼럼에서 집계 함수를 수행하면 집계 대상 칼럼의 타입이 **CHAR**인 경우 "ERROR: No error message available." 오류가 발생하고, **VARCHAR**인 경우 잘못된 결과를 출력하는 현상을 수정했다.

```
CREATE TABLE t1 (id int PRIMARY KEY, b char(16), INDEX i1(b(4)));
INSERT INTO t1 VALUES (1, 'xxxxbbbb'), (2, 'xxxxaaaa');
-- 수정 이전 버전에서 다음 질의 수행 시 오류 발생
SELECT MAX(b) FROM t1;

CREATE TABLE t1 (id int PRIMARY KEY, b varchar(16), INDEX i1(b(4)));
INSERT INTO t1 VALUES (1, 'xxxxbbbb'), (2, 'xxxxaaaa');
-- 수정 이전 버전에서 다음 질의 수행 시 'xxxx'가 결과 값으로 나옴
SELECT MAX(b) FROM t1;
```

인덱스 생성 시 동일한 키를 가지는 인덱스가 생성되어 있으면 B-Tree를 공유하게 됨(CUBRIDSUS-6430)

인덱스 생성 시 동일한 키의 인덱스가 이미 생성되어 있으면 인덱스를 생성할 수 없다는 오류가 발생했으나, 물리적으로 인덱스를 중복하여 생성하지 않고 기존 인덱스를 공유하도록 변경했다. 아래 예에서 외래 키를 생성하는 b 칼럼에는 이미 인덱스가 정의되어 있어 외래 키를 위한 B-Tree를 별도로 생성할 필요가 없다.

```
CREATE TABLE t1
(
  a INT PRIMARY KEY,
```

```
  b INT,
  KEY idx(b),
  CONSTRAINT fk FOREIGN KEY (b) REFERENCES t1 (a)
);
```

검색 조건에 클래스 OID로 검색하는 조건이 있고 그외 나머지 조건들은 모두 커버링 인덱스 조건을 만족할 때 서버가 비정상 종료되는 오류(CUBRIDSUS-7585)

검색 조건에 클래스 OID로 검색하는 조건이 있고 그 외 나머지 조건들은 모두 커버링 인덱스 조건(인덱스에 **SELECT** 리스트의 칼럼과 **WHERE** 조건의 칼럼을 모두 포함)을 만족할 때 질의 수행 과정에서 서버가 비정상 종료되는 오류를 수정했다.

```
CREATE TABLE a (class_of OBJECT, class_name VARCHAR(200));
CREATE TABLE c (class_name VARCHAR(200), i INT);
CREATE INDEX idx ON c (class_name);
INSERT INTO a VALUES (INSERT INTO c VALUES('aa', 1), 'aa');

SELECT c.class_name
FROM a, c
WHERE a.class_of=c AND c.class_name=a.class_name;
```

USING INDEX 절이 유효하지 않은 경우 응용 프로그램이 비정상 종료될 수 있는 현상 (CUBRIDSUS-7672)

다음 예와 같이 **USING INDEX** 절이 유효하지 않은 경우 응용 프로그램이 비정상 종료될 수 있는 현상을 수정했다.

```
CREATE TABLE t1 (a INT, b CHAR(3), c INT);
CREATE TABLE t2 (a INT, b CHAR(3), c INT);
CREATE INDEX i1 ON t1 (a, c);
CREATE INDEX i2 ON t2 (a, c);

SELECT * FROM (t1,t2) s WHERE c=9 USING INDEX i1;
```

외래 키 생성 시 CONSTRAINT ... REFERENCES ... 구문 사용이 가능해짐(CUBRIDSUS-6556)

외래 키 생성 시 이전 버전에서는 **FOREIGN KEY REFERENCES ...** 형태의 구문만 사용 가능했으나, **CONSTRAINT constraint_이름 REFERENCES ...**도 사용 가능하도록 수정했다.

```
CREATE TABLE tbl (id INT CONSTRAINT pk_id PRIMARY KEY);

CREATE TABLE tbl2 (id INT FOREIGN KEY REFERENCES tbl(id));
```

```
CREATE TABLE tbl3 (id INT CONSTRAINT fk_id REFERENCES tbl(id));
```

ALTER ... CHANGE COLUMN 문 수행 시 기본 키 칼럼의 NOT NULL 제약 조건이 제거되는 문제(CUBRIDSUS-7966)

ALTER ... CHANGE COLUMN 문을 수행하면 기본 키 칼럼의 **NOT NULL** 제약 조건이 제거되는 문제를 수정했다.

```
CREATE TABLE t1_add_default (a INT PRIMARY KEY, b INT);
ALTER TABLE t1_add_default CHANGE COLUMN a a INT DEFAULT 2;
INSERT INTO t1_add_default(a, b) VALUES(NULL, 1);
```

다수의 트랜잭션이 오버 플로우 키가 존재하는 인덱스에 대한 입력/삭제를 동시에 수행하는 도중, 한 트랜잭션의 롤백으로 인해 다른 트랜잭션이 실패하는 오류(CUBRIDSUS-8838)

오버 플로우 키가 존재하는 인덱스에 여러 트랜잭션이 동시에 입력/삭제를 수행하는 환경에서, 한 트랜잭션이 롤백되면 다른 트랜잭션들에서 "Query execution failure" 에러가 발생할 수 있는 현상을 수정했다.

오버플로우 키가 seperator로 사용된 인덱스에 대해 통계 정보를 갱신하면 서버 프로세스가 비정상 종료되는 문제(CUBRIDSUS-8865)

HA 기능 및 HA 데이터 복제

컬럼 타입 변경 이후 복제 불일치 문제(CUBRIDSUS-7386)

트랜잭션 복제 로그 반영 프로세스가 재시작하면서, 스키마 변경 이전 로그를 재반영하여 복제 불일치가 발생하는 문제를 해결하였다.

매우 긴 트랜잭션을 끝까지 반영하지 못하는 문제(CUBRIDSUS-7638)

트랜잭션 복제 로그 반영 프로세스가 매우 긴 트랜잭션 반영 시 무한 재시작하며 끝까지 반영하지 못하는 문제를 해결하였다.

새 명령어들 추가(CUBRIDSUS-5468)

데이터베이스의 트랜잭션 로그 복사/복제 로그 반영 프로세스를 구동/정지하는 **cubrid heartbeat copylogdb/applylogdb <start|stop> db_name peer_host** 명령이 추가되었다.

로그 다중화가 불가능한 상태(노드간 데이터 동기화가 불가능한 상태)에서 단독으로 서버 프로세스만 실행되는 것을 방지하기 위해, HA 모드로 설정된 데이터베이스 서버 프로세스를 **cubrid server start/stop** 명령으로 시작/정지할 수 없도록 했다. 수정 이전 버전에서 사용되었던 **act, deact, deregister** 명령은 더 이상 사용되지 않는다.

DELETE 복제 성능 개선(CUBRIDSUS-6436)

HA 복제에서 **DELETE** 복제 성능을 개선하여 초당 복제 건수가 이전 버전 대비 약 1.7배 향상되었다.

트랜잭션 복제 로그 반영 프로세스의 CPU 사용량을 줄이도록 개선(CUBRIDSUS-6118)

정상 상황일 때 슬레이브 노드에 발생 가능한 오류에 대해 오류 수준을 Warning으로 변경(CUBRIDSUS-8937)

HA 환경에서 슬레이브 노드에 "Internal error: fetching deallocated pageid 0 of volume /CUBRID/databases/testdb_lgat" 오류는 정상 상황일 때 발생할 수 있으므로 오류 수준을 Error에서 Warning으로 변경했다.

시스템 파라미터 값의 구분자로 ","와 ":" 두 가지 모두 사용 가능하도록 개선(CUBRIDSUS-5647)

ha_node_list, ha_replica_list, ha_db_list, ha_copy_sync_mode, ha_ping_hosts 등 HA 관련 시스템 파라미터의 값들에 대한 구분자로 콤마(',')와 콜론(':') 두 가지 모두 사용할 수 있도록 개선했다.

마스터 노드가 슬레이브 노드의 연결을 대기 중인 to-be-active 상태에서 TCP 연결이 끊어지면 오류 처리하지 않는 문제(CUBRIDSUS-7154)

HA 환경에서 마스터 노드가 슬레이브 노드의 연결을 대기 중인 to-be-active 상태에서 연결을 요청한 클라이언트의 TCP 연결이 끊어지는 경우에도 마스터 서버 프로세스(cub_master)에 연결 대기 정보가 남아있는 문제가 존재했으나, 이러한 경우 오류 처리하도록 수정했다.

트리거가 존재하는 경우 트리거에 의해 변경된 데이터가 복제 반영되었음에도 불구하고 슬레이브 노드에서 다시 트리거를 수행하는 문제(CUBRIDSUS-8165)

마스터 노드의 트리거 동작 결과가 슬레이브 노드에 복제되었음에도 불구하고 슬레이브 노드에서 동일한 트리거 동작이 반복 수행되어 질의 결과가 잘못될 수 있는 문제를 수정했다.

핑 호스트가 설정된 HA 환경에서 슬레이브 서버와 핑 호스트 모두 연결 불가능한 상황이 발생하면 상황 복구 이후에도 응용 프로그램이 마스터 서버에 연결 불가능한 현상(CUBRIDSUS-7183)

핑 호스트(ping host)가 설정된 HA 환경에서 슬레이브 서버와 핑 호스트 모두 마스터 서버와 연결 불가능한 상황이 발생하면 연결 가능하도록 복구한 이후에도 응용 프로그램이 마스터 서버에 연결 불가능한 현상을 수정했다. 핑 호스트가 존재하는 HA 구성에서는 마스터 서버가 핑 호스트로 연결이 되지 않으면 마스터의 역할을 유지할 수 없다고 판단하기 때문에 슬레이브로 역할 변경을 하게 되고, 응용 프로그램은 읽기 전용으로만 사용이 가능하게 된다.

failover 후 마스터가 된 노드에서 CUBRID HA 기능을 정지하면 이후 HA 기능을 구동했을 때 비정상 종료되는 현상(CUBRIDSUS-8906)

failover 후 마스터가 된 노드에서 **cubrid heartbeat stop**으로 CUBRID HA 기능을 정지하면 이후 **cubrid heartbeat start**로 HA 기능을 구동했을 때 비정상 종료되는 현상을 수정했다.

레플리카 노드 설정을 삭제한 후 reload 명령을 수행해도 레플리카 노드가 구성 정보에 남아있는 문제(CUBRIDSUS-8107)

cubrid_ha.conf의 **ha_replica_list**를 삭제한 이후 **cubrid heartbeat reload** 명령을 수행했음에도 불구하고 레플리카 노드가 구성 정보에서 삭제되지 않고 남아있는 문제를 수정했다.

테이블의 분할 조건에 음수가 포함된 경우 조건이 양수로 바뀌는 문제(CUBRIDSUS-8461)

HA 환경에서 테이블의 분할 조건에 음수가 포함된 경우 조건이 양수로 바뀌는 문제를 수정했다. 수정 이전 버전에서 아래의 예를 수행하면 분할 조건 (-2, 1)이 (2, 1)로 잘못 생성되었다.

```
CREATE TABLE t1 (a INT AUTO_INCREMENT(-2,3) PRIMARY KEY)
PARTITION BY LIST(a)
(PARTITION p0 VALUES IN (-2, 1));
```

슬레이브 노드에 쓰기 요청 시 실패한 요청을 무한히 재시도하는 현상(CUBRIDSUS-6266)

슬레이브 노드에 **INCR/DECR** 함수, **UPDATE** 등 쓰기 요청 시 쓰기에 실패했다는 오류 메시지를 출력해야 하나 응용 프로그램에서는 실패한 요청을 무한히 재시도하고, CSQL 인터프리터에서는 "ERROR: Your transaction has been aborted by the system due to server failure or mode change." 메시지를 두 번 출력한 후 CSQL 인터프리터를 종료하는 현상을 수정했다.

슬레이브 노드에서 페이지 버퍼 크기를 초과하는 칼럼으로 정렬하는 SELECT 질의 수행에 실패하는 현상(CUBRIDSUS-9272)

HA 환경의 슬레이브 노드에서 페이지 버퍼 크기(기본 16K)를 초과하는 칼럼으로 정렬하는 **SELECT** 질의를 수행하면 "Attempted to update the database when updates are disabled." 오류 메시지와 함께 실패하는 현상을 수정했다.

to-be-active 상태의 서버에 클라이언트가 수 차례 접속을 시도한 상태에서 서버의 종료를 시도하면 서버 프로세스가 비정상 종료되는 현상(CUBRIDSUS-7756)

HA 환경에서 to-be-active 상태의 서버에 csql 등의 클라이언트가 cubrid.conf의 **max_clients** 파라미터 개수 이상 접속 시도를 반복한 상태에서 서버의 정상 종료를 시도하면 서버 프로세스가 비정상 종료되는 현상을 수정했다.

서버 프로세스 장애 이후 재시작된 상태에서 HA 기능 종료를 수행하면 HA 관리 프로세스가 비정상 종료될 수 있는 문제(CUBRIDSUS-6878)

HA 환경에서 서버 프로세스(cub_server) 장애 이후 프로세스가 재시작된 상태에서 **cubrid heartbeat stop**을 수행하면 HA를 관리하는 마스터 서버(cub_master) 프로세스가 비정상 종료될 수 있는 문제를 수정했다.

HA 기능 정지 도중 트랜잭션 복제 로그 반영 프로세스가 비정상 종료될 수 있는 현상(CUBRIDSUS-6974)

cubrid heartbeat stop으로 HA 기능 정지 도중 트랜잭션 복제 로그 반영 프로세스가 비정상 종료될 수 있는 현상을 수정했다.

슬레이브 노드의 복제 재구축 후 트랜잭션 복제 로그 반영 프로세스가 비정상 종료될 수 있는 문제(CUBRIDSUS-9166)

HA 환경에서 슬레이브 노드의 복제 재구축 후 반영해야 할 보관 로그의 로그 페이지를 찾지 못하는 오류로 인해 트랜잭션 복제 로그 반영 프로세스가 비정상 종료될 수 있는 문제를 수정했다.

ASYNC 모드로 동작하는 트랜잭션 복제 로그 복사 프로세스와 연결되는 서버 프로세스에서 ETIMEDOUT 오류는 정상 동작임에도 오류 로그를 기록하는 문제(CUBRIDSUS-6248)

ASYNC 모드로 동작하는 트랜잭션 복제 로그 복사 프로세스와 연결되는 서버 프로세스에서 "pthread_cond_wait() failed." (ETIMEDOUT) 오류 발생 시 오류 로그를 기록하지 않도록 수정했다.

ETIMEDOUT 오류는 서버 프로세스에서 트랜잭션 로그를 기록하는 스레드가 동작 대기 시간을 초과하면서 발생하는 오류이며 ASYNC 모드에서 이 오류가 발생하는 것은 정상임에도 불구하고, 이전 버전에서는 이를 오류 로그 파일에 기록하여 불필요한 디스크 I/O를 초래했다.

shell prompt가 길면 HA 복제 스크립트가 실패하는 문제(CUBRIDSUS-7524)

슬레이브 노드 재구성 스크립트인 ha_make_slavedb.sh에서 다른 노드의 환경 변수 체크 결과를 전달받는 부분이 실패하는 문제를 수정했다.

HA 스키마 복제

WITH CHECK OPTION 절이 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-8367)

HA 환경에서 뷰의 **WITH CHECK OPTION** 절이 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
CREATE TABLE vc_tb(a INT PRIMARY KEY);
CREATE VIEW fbo_view AS SELECT * FROM vc_tb WHERE a > 5 WITH CHECK OPTION;
```

DROP TABLE 문의 IF EXISTS 절이 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-8901)

HA 환경에서 **DROP TABLE** 문의 **IF EXISTS** 절이 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
DROP TABLE IF EXISTS tbl1;
```

뷰의 칼럼으로 논리 표현식이 포함된 경우 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-7927)

HA 환경에서 뷰의 **SELECT** 리스트에 논리 표현식이 포함된 경우 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
CREATE VIEW v(a,b,c) AS SELECT a, b, (b = 'aaa') c FROM t;
```

뷰가 마스터 노드와 슬레이브 노드에서 서로 다르게 보이는 문제(CUBRIDSUS-8366)

마스터 노드에서 생성한 뷰를 슬레이브 노드에서 시스템 카탈로그 테이블 **db_vclass**로 출력하면 **WHERE** 조건 일부에 괄호가 추가되어 보이는 문제를 수정했다.

```
CREATE TABLE vc_tb1(a INT PRIMARY KEY);
// 수정 이전 버전에서 위의 뷰를 슬레이브 노드의 db_vclass에서 출력하면 WHERE 조건이 괄호로 감싸였다.
CREATE VIEW vtb1(a INT) AS SELECT * FROM vc_tb1 WHERE a >= 1;
```

CREATE .... AS SELECT 문으로 테이블 생성 시 슬레이브 노드에 잘못된 칼럼 순서로 복제될 수 있는 문제(CUBRIDSUS-8071)

HA 환경에서 **CREATE .... AS SELECT** 문으로 테이블 생성 시 슬레이브 노드에 잘못된 칼럼 순서로 복제될 수 있는 문제를 수정했다.

```
// 마스터 노드에서 다음을 수행
CREATE TABLE t2 (a INT PRIMARY KEY, b INT, c INT)
AS SELECT t1.c AS a, t1.a AS b, t1.b AS c FROM t1;

// 슬레이브 노드에서 다음과 같이 칼럼 순서가 잘못 복제됨
CREATE TABLE [t2] ( [a] INTEGER, [b] INTGER, [c] INTEGER, PRIMARY KEY ([a]) )
AS SELECT [t1].[c], [t1].[a], [t1].[b] FROM [t1];
```

인덱스를 CREATE/DROP하고 테이블을 DROP 하면 슬레이브 노드에서 -414번 오류가 발생하는 현상(CUBRIDSUS-8634)

하나의 트랜잭션에서 인덱스를 **CREATE/DROP**하고 테이블을 **DROP**한 후 커밋하면 슬레이브 노드에서 정상 상황임에도 불구하고 "ERROR CODE = -414 ... Unknown class identifier" 오류가 발생하는 현상을 수정했다.

SELECT 문을 동반한 DDL 문이 슬레이브 노드에 복제되지 않는 문제(CUBRIDSUS-8992)

HA 환경에서 **CREATE TABLE ... SELECT, CREATE VIEW ... SELECT** 와 같이 **SELECT** 문을 동반한 DDL 문이 복제되지 않는 문제를 수정했다.

```
CREATE TABLE t1 (a INT PRIMARY KEY, b INT)
AS SELECT * FROM (SELECT 1 AS a, (SELECT 1) b FROM db_root) t;
```

인덱스 생성 시 슬레이브 노드에 반영되지 않는 문제(CUBRIDSUS-6818)

HA 환경에서 인덱스를 생성할 때 인덱스 생성문이 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
CREATE TABLE tab(a INT AUTO_INCREMENT(1,1) PRIMARY KEY, c VARCHAR(32));
CREATE INDEX i_tab_c ON tab(c);
```

생성하려는 인덱스의 이름이 예약어인 경우 슬레이브 노드에 반영되지 않는 현상 수정(CUBRIDSUS-8054)

HA 환경에서 생성하려는 인덱스의 이름이 예약어인 경우 "Internal system failure: [t2,'create class [t2] ([a] integer, [b] integer, index none ([a])) '] In line 1, column 44 before ' ([a])) '" 오류 메시지와 함께 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
CREATE TABLE t2(a INT, b INT, INDEX "none" (a));
```

AUTO_INCREMENT 필드가 있는 테이블에 대해 TRUNCATE 문 수행 시 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-7810)

HA 환경에서 **AUTO_INCREMENT** 필드가 있는 테이블에 대해 **TRUNCATE** 문을 수행하면 슬레이브 노드에서 "Accessing Delete Object" 오류 메시지와 함께 슬레이브 노드에 반영되지 않는 현상을 수정했다.

AUTO_INCREMENT 칼럼이 있는 테이블의 DROP 이후에도 슬레이브 노드에 AUTO_INCREMENT에 의한 시리얼이 남아있는 문제(CUBRIDSUS-8885)

HA 환경의 마스터 노드에서 **AUTO_INCREMENT** 칼럼이 있는 테이블의 **DROP** 이후에도 슬레이브 노드에 **AUTO_INCREMENT**에 의한 시리얼이 남아있는 오류로 인해, 마스터 노드에서 같은 테이블 이름과 칼럼 이름으로 **AUTO_INCREMENT**를 재생성을 시도하면 슬레이브 노드에 테이블이 생성되지 않는 문제를 수정했다.

```
CREATE TABLE t1(id INT PRIMARY KEY AUTO_INCREMENT, a INT);
DROP TABLE t1;
CREATE TABLE t1(id INT PRIMARY KEY AUTO_INCREMENT, b INT);
```

AUTO_INCREMENT 칼럼 속성 또는 이름을 변경하면 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-7830)

HA 환경에서 **AUTO_INCREMENT** 칼럼 속성 변경 또는 **AUTO_INCREMENT** 칼럼 속성 추가와 함께 이름을 변경하면 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
// 속성 변경
CREATE TABLE foo (a INTEGER PRIMARY KEY AUTO_INCREMENT, b CHAR(10), c DATETIME);
ALTER TABLE foo MODIFY ATTRIBUTE a BIGINT AUTO_INCREMENT;

// 이름 변경
CREATE TABLE boo (i int PRIMARY KEY);
ALTER TABLE boo ADD COLUMN ai INT AUTO_INCREMENT, RENAME TO u, AUTO_INCREMENT = 100;
```

ALTER SERIAL 문이 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-8727)

HA 환경에서 **ALTER SERIAL** 문이 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
ALTER SERIAL s1 START WITH 2;
```

DROP SERIAL 문이 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-8673)

HA 환경에서 **DROP SERIAL** 문이 슬레이브 노드에 반영되지 않는 현상을 수정했다.

시리얼 생성 혹은 변경에서 NOCACHE 옵션 지정 시 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-9225)

HA 환경에서 **CREATE SERIAL** 또는 **ALTER SERIAL** 문에 **NOCACHE** 옵션이 지정되면 슬레이브 노드에 반영되지 않는 현상을 수정했다.

ALTER INDEX ... REBUILD 문이 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-8090)

HA 환경에서 **ALTER INDEX ... REBUILD** 문 수행 시 "log applier: failed to apply schema replication log. class: "-", schema: 'alter index i_t_b on () rebuild', internal error: -492." 오류 메시지와 함께 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
ALTER INDEX i_t_b REBUILD;
```

ALTER ... CHANGE COLUMN 문이 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-8023)

HA 환경에서 **ALTER ... CHANGE COLUMN** 문이 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
ALTER TABLE coo CHANGE col1 col1 INT AFTER col3;

// 위의 질의를 수행하면 슬레이브 노드에서 아래와 같은 오류가 발생한다.
log applier: failed to apply schema replication log. class: "coo", schema: "'alter class [coo]
change attribute [col1] [col1] integerafter [col3] '", internal error: -492.

ALTER TABLE t1 ALTER COLUMN a set DEFAULT 1;

// 위의 질의를 수행하면 슬레이브 노드에서 아래와 같은 오류가 발생한다.
"alter class [t1] alter column [b] set default default 1", this SQL statement is illegal. So
there is something wrong with the "parser_print_xxx" function.
```

ALTER ... CHANGE 문으로 칼럼의 DEFAULT 값을 변경하면 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-8021)

HA 환경에서 **ALTER ... CHANGE** 문으로 **DEFAULT** 값을 변경하면 "log applier: failed to apply schema replicati on log. class: "t1", schema: "'alter class [t1] alter column [a] set default default 1'", internal error: -492." 오류 메시지와 함께 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
ALTER TABLE t1 ALTER COLUMN a SET DEFAULT 1;
```

ALTER .... AUTO_INCREMENT 문이 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-8203)

HA 환경에서 **ALTER .... AUTO_INCREMENT** 문이 "log applier: failed to apply schema replication log. class: "t", schema: "'alter class [t] auto_increment = 5'", internal error: -1056." 오류 메시지와 함께 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
ALTER TABLE t AUTO_INCREMENT = 5;
```

ALTER 문으로 테이블 분할 시 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-5266)

HA 환경에서 **ALTER** 문으로 테이블 분할을 수행하는 경우 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
ALTER TABLE range_tbl
REORGANIZE PARTITION p2 INTO
(PARTITION p3 VALUES LESS THAN (25), PARTITION p4 VALUES LESS THAN (30));
```

PREFIX 칼럼에 DESC INDEX 생성 시 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-8980)

HA 환경에서 PREFIX 칼럼에 **DESC INDEX** 생성 시 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
ALTER TABLE c2 ADD INDEX i_c (c(5) DESC);
```

트리거, 시리얼, 저장 프로시저의 소유자가 슬레이브 노드에 잘못 반영되는 현상(CUBRIDSUS-8723)

트리거, 시리얼, 저장 함수 및 프로시저의 소유자가 슬레이브 노드에 항상 "DBA"로 잘못 반영되는 현상을 수정했다.

자동 커밋 모드가 OFF일 때 DROP TABLE IF EXISTS 문을 수행하면 슬레이브 노드에 반영되지 않는 현상(CUBRIDSUS-8582)

자동 커밋 모드가 OFF일 때 **DROP TABLE IF EXISTS** 문을 수행하면 "SYNTAX ERROR ... ERROR CODE = -493" 오류 메시지와 함께 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
DROP TABLE IF EXISTS m1;
CREATE TABLE m1(i INT,j INT PRIMARY KEY);
COMMIT;
```

분할 테이블이 슬레이브 노드에 반영되지 않을 수 있는 문제(CUBRIDSUS-8985)

HA 환경에서 분할 테이블의 파티션 데이터가 내부적으로 타입 변환이 일어나는 경우, 슬레이브 노드에 복제되지 않는 문제를 수정했다.

```
CREATE TABLE t1(i INT PRIMARY KEY) PARTITION BY RANGE (i) (
    PARTITION p1 VALUES LESS THAN (-2147483648),
    PARTITION p2 VALUES LESS THAN MAXVALUE
);
```

드라이버

[JDBC] 예외 오류 메시지 출력 시 연결 URL을 출력하도록 개선(CUBRIDSUS-7781)

JDBC 예외 오류 메시지를 출력할 때 연결 URL 정보도 같이 출력하도록 개선했다.

```
cubrid.jdbc.driver.CUBRIDException: jdbc:cubrid:localhost:33300:testdb:dba:********:
Syntax: Syntax error: unexpected END OF STATEMENT
```

[JDBC] iBatis에서 자바 저장 프로시저의 Resultset을 조회할 수 없는 문제(CUBRIDSUS-7383)

iBatis에서 자바 저장 프로시저를 호출한 결과 값을 칼럼 이름으로 resultMap을 구성하여 ArrayList로 전달받을 수 있게 되었다.

[JDBC] ResultSetMetaData.isAutoIncrement() 메서드가 정상 동작하지 않는 문제(CUBRIDSUS-7531)

JDBC의 ResultSetMetaData.isAutoIncrement() 메서드 호출 시 어떤 칼럼의 **AUTO_INCREMENT** 설정 여부와 무관하게 false를 반환했으나, **AUTO_INCREMENT** 설정 여부에 맞게 동작하도록 수정했다.

[JDBC] 응용 프로그램에서 브로커 로그에 연결 URL을 잘못 출력하는 오류(CUBRIDSUS-7956)

JDBC 응용 프로그램에서 브로커 로그에 연결 URL을 출력할 때 "?"를 추가로 잘못 출력하는 오류를 수정했다. 다음은 수정 이전 버전에서 브로커 로그 파일에 URL을 잘못 출력한 예이다. dba:: 뒤에 나타나는 "?" 는 하나만 출력되어야 한다.

```
jdbc:cubrid:10.0.0.1:33000:demodb:dba::??queryTimeout=60000001&connectTimeout=10000
```

[JDBC] Date나 Time 객체 값을 받을 때 밀리초가 0으로 리셋되지 않은 채로 반환되는 문제(CUBRIDSUS-7352)

java.sql.Date나 java.sql.Time 객체의 값이 밀리초 부분의 값이 0으로 리셋되지 않은 상태로 반환되는 문제를 수정했다.

```
ResultSet rs = connection.createStatement().executeQuery("select time '12:15:00'");
rs.next();
System.out.println(rs.getTime(1).getTime());
```

[JDBC] 응용 프로그램에서 음수년도의 입력을 허용하는 오류(CUBRIDSUS-8844)

JDBC 응용 프로그램에서 Datetime 데이터를 bind 할 때 허용 범위(1-9999)를 벗어난 년도의 입력을 허용하는 오류를 수정했다.

[JDBC] Connection이 종료되거나 DatabaseMetaData가 close된 이후 호출되는 DatabaseMetaData의 일부 메서드에서 NullPointerException이 발생하는 오류(CUBRIDSUS-7807)

JDBC의 Connection에서 DatabaseMetaData를 얻어온 후 Connection이 종료되거나 DatabaseMetaData가 close된 경우에 DatabaseMetaData의 getTables(), getColumns(), getTablePrivileges(), getPrimaryKeys(), getForeignKeys() 메서드를 호출하면 NullPointerException이 발생하는 오류를 수정했다.

[JDBC] 같은 Statement 객체를 반복 사용하여 여러 개의 질의를 수행하면 도중에 발생할 수 있는 예외를 처리하지 못하는 문제(CUBRIDSUS-9015)

JDBC 응용 프로그램에서 같은 Statement 객체를 반복 사용하여 여러 개의 질의를 수행하면 도중에 예외 발생 시 사용 중이던 서버 핸들을 종료하지 못하여 에러 코드 -26번이 반복하여 발생할 수 있는 문제를 수정했다.

```
public void run(ArrayList<String> sqlList, int max) throws SQLException {
  this.sqlList = sqlList;
  Connection conn = getConnection();
  Statement stmt = conn.createStatement();

  String sql;
  int i = 0;
  while (true) {
    sql = getNextLine();
    if (sql == null)
      break;
    try {
      stmt.execute(sql);
    } catch (Exception e) {
      log(e, i, sql);
    }
    i++;
  }
  stmt.close();
  conn.close();
}
```

[JDBC] /*+ RECOMPILE */ 힌트가 있는 SELECT 문을 여러 스레드에서 동시에 실행하면 "Statement Pooling" 오류가 발생하는 문제(CUBRIDSUS-7616)

JDBC 응용 프로그램에서 /*+ RECOMPILE */ 힌트가 있는 SELECT 문을 여러 스레드에서 동시에 실행하면 "cubrid.jdbc.driver.CUBRIDException: Statement Pooling" 오류가 발생하는 문제를 수정했다.

[JDBC] ResultSet.TYPE_SCROLL_SENSITIVE 옵션이 사용될 때 시스템 카탈로그 뷰에 대한 SELECT 질의에 실패하는 문제(CUBRIDSUS-7076)

JDBC의 ResultSet.TYPE_SCROLL_SENSITIVE 옵션이 사용될 때 db_class와 같은 시스템 카탈로그 뷰에 대한 SELECT 질의를 수행한 뒤 fetch를 수행하면 "Semantic: System error (generate attr) in ../../src/parser/xasl_generation.c" 오류가 발생하는 문제를 수정했다.

ResultSet.TYPE_SCROLL_SENSITIVE 옵션은 현재 지원하지 않으므로 ResultSet.TYPE_SCROLL_INSENSITIVE로 설정할 것을 권장한다.

[JDBC] 2.4E+2와 같은 과학적 표기법의 값을 BigDecimal로 바인딩하는데 실패하는 문제(CUBRIDSUS-7943)

JDBC 응용 프로그램에서 2.4E+2와 같은 과학적 표기법(scientific notation)으로 표현된 값을 BigDecimal로 바인딩하는데 실패하는 문제를 수정했다.

```
BigDecimal x = new BigDecimal("240.0");
x = x.stripTrailingZeros(); // 과학 형식의 값으로 지정
p.setBigDecimal(1, x);
```

[JDBC] prepare한 질의에 유효하지 않은 값을 바인딩하여 실행한 후 다시 유효한 값을 바인딩하여 실행하면 Exception이 발생하는 현상(CUBRIDSUS-7648)

JDBC 응용프로그램에서 prepare한 질의에 유효하지 않은 값을 바인딩하여 실행한 후 다시 유효한 값을 바인딩하여 실행하면 Exception이 발생하는 현상을 수정했다.

```
// 수정 이전 버전에서는 다음과 같이 Exception이 발생했다.
No error message available.
cubrid.jdbc.driver.CUBRIDException: No error message available.
at cubrid.jdbc.driver.CUBRIDConnection.createCUBRIDException(CUBRIDConnection.java:829)
at cubrid.jdbc.driver.CUBRIDStatement.checkExecuteError(CUBRIDStatement.java:941)
at cubrid.jdbc.driver.CUBRIDStatement.executeCoreInternal(CUBRIDStatement.java:830)
at cubrid.jdbc.driver.CUBRIDStatement.executeCore(CUBRIDStatement.java:791)
at cubrid.jdbc.driver.CUBRIDPreparedStatement.executeQuery(CUBRIDPreparedStatement.java:107)
at cubrid_262_error.main(cubrid_262_error.java:26)
```

[JDBC] 백슬래시를 이스케이프 문자로 사용하도록 설정하면 getColumns() 메서드 수행 시 발생하는 오류(CUBRIDSUS-8080)

시스템 파라미터 **no_backslash_escape**의 값을 no로 설정하여 백슬래시를 이스케이프 문자로 사용하도록 설정하면 getColumns() 메서드로 테이블의 칼럼 정보를 얻을 때 "unterminated string" 오류가 발생하는 현상을 수정했다.

[JDBC] 응용 프로그램에서 Statement.close()를 수행하기 전에는 Resultset.close()를 수행해도 결과 셋의 메모리를 해제하지 않는 문제(CUBRIDSUS-9206)

JDBC 응용 프로그램에서 Statement.close()를 수행하기 전까지는 Resultset.close()를 수행해도 결과 셋의 메모리를 해제하지 않는 문제로 인해 메모리 사용량이 증가할 수 있는 현상을 수정했다.

[CCI] Linux용 CCI 라이브러리는 파일만으로도 드라이버의 버전 확인이 가능하게 개선(CUBRIDSUS-6954)

Linux용 CCI 라이브러리는 파일만으로도 드라이버의 버전 확인이 가능하게 개선했다.

```
$ strings /home/usr1/CUBRID/lib/libcascci.so ¦ grep VERSION
VERSION=9.0.0.0478
```

참고로, cci_get_version(major, minor, patch) 함수를 사용해도 드라이버의 버전 확인이 가능하다.

[CCI] 디버깅용 로그를 기록하도록 설정하는 연결 URL 속성 추가(CUBRIDSUS-6377)

CCI 연결 URL에 디버깅용 로그 기록을 설정하는 기능을 추가했다. logSlowQueries와 slowQueryThresholdMillis는 슬로우 쿼리의 로그 기록을, logTraceApi는 CCI 함수가 호출될 때 각 함수의 시작과 끝을, logTraceNetwork은 CCI 함수의 Network 데이터 전송 내용을 로그 파일에 기록한다.

```
url = "cci:cubrid:localhost:33000:demodb:::?logSlowQueries=true&slowQueryThresholdMillis=1000&logTraceApi=true&logTraceNetwork=true"
```

아울러, CCI 연결 URL에 디버깅용 로그 파일의 경로를 지정하는 logBaseDir의 동작 방식을 수정했다. 이전 버전에서는 logBaseDir 값과 logFile이 같이 있으면 logBaseDir 프로퍼티를 무시했으나 수정 이후 경로를 포함한 파일 이름을 "logBaseDir/logFile"로 지정하도록 바뀌었다.

[CCI] cci_schema_info() 함수에서 트리거 정보를 얻을 때 테이블 이름을 지정했는데도 전체 데이터베이스의 트리거 정보를 출력하는 오류(CUBRIDSUS-7675)

cci_schema_info() 함수를 통해 트리거 정보를 얻을 때 테이블 이름을 지정했는데도 전체 데이터베이스의 트리거 정보를 출력하는 오류를 수정했다. 이와 함께, 트리거 정보를 얻을 때 입력할 테이블 이름 또는 칼럼 이름을 LIKE 절의 패턴 매칭으로 검색할 수 있는 기능을 추가했다.

```
req = cci_schema_info (conn, CCI_SCH_TRIGGER, "tbl%", NULL,
                       CCI_CLASS_NAME_PATTERN_MATCH, &error);
```

[CCI] cci_get_attr_type_str()의 반환 값이 잘못된 문제(CUBRIDSUS-7910)

cci_get_attr_type_str()의 반환 값에서 제일 마지막 문자가 잘리는 문제를 수정했다.

```
// 소스 예제
char *attr_infos[][2] =
{
  {"aa", "character(1)"},
  {NULL, NULL}
};

...
res = cci_get_attr_type_str(conn, class_name, attr_infos[i][0], buf, buf_size, &error);
...
fprintf(LOG_FD, "%s attr: %s\n\n", attr_infos[i][0], buf);

// 수정 이전 출력 화면
aa attr: character(1
```

[CCI] 서버 재시작 이후에도 cci_prepare() 함수를 호출하면 정상 수행되지 않고 여전히 오류가 발생하는 문제(CUBRIDSUS-8907)

데이터베이스 서버 종료로 인해 cci_prepare() 함수에서 오류가 발생한 이후 서버를 재시작해도 cci_prepare() 함수를 호출하면 정상 수행되지 않고 여전히 오류가 발생하는 문제를 수정했다.

[CCI] 잠금 타임아웃이 발생했을 때 CCI 함수에서 잘못된 오류 코드를 반환하는 문제(CUBRIDSUS-7226)

cci_prepare() 및 cci_execute() 함수에서 잠금 타임아웃(lock timeout)이 발생했을 때 오류 코드 -75를 오류 코드 -493으로 래핑(wrapping)하여 반환하던 것을 -75번 오류가 직접 전달되도록 수정했다.

[JDBC, CCI] 응용 프로그램에서 트랜잭션 진행 중 특정 시점 이후 일부 질의만 커밋되는 문제 수정(CUBRIDSUS-8563)

JDBC 또는 CCI 응용 프로그램에서 트랜잭션 진행 중 서버의 재시작, CAS의 강제 재시작 등 특정 시점 이후 일부 질의만 커밋되는 문제를 수정했다.

[CCI] cci_prepare_and_execute() 함수 실행 시 CAS가 재시작되면 오류가 발생할 수 있는 문제 (CUBRIDSUS-9278)

cci_prepare_and_execute() 함수 실행 시 재시작된 CAS에 재접속하는 루틴이 존재하지 않아 "Cannot communcation with broker" 오류가 발생하는 문제를 수정했다.

[CCI] 질의 타임아웃 값을 매우 작게 설정하면 cci_prepare_and_execute() 함수에서 질의 타임아웃이 발생하지 않는 현상(CUBRIDSUS-8102)

질의 타임아웃 값을 3밀리 초와 같이 아주 작게 설정하면 cci_prepare_and_execute() 함수에서 질의 타임 아웃이 발생하지 않는 현상을 수정했다. 이와 함께 CAS 로그에 보여지는 query timeout 설정값이 다음과 같이 밀리초 단위로 나타나도록 수정했다.

```
07/11 11:34:52.933 (2) set query timeout to 3000 milliseconds (from app)
```

[CCI] 동시 다중 스레드 실행 환경에서 cci_connect_with_url() 함수의 연결 URL에 호스트 명이 주어지면 응용 프로그램이 비정상 종료될 수 있는 현상(CUBRIDSUS-8318)

여러 개의 스레드가 동시에 실행되는 환경에서 cci_connect_with_url() 함수의 연결 URL에 IP 주소가 아닌 호스트 명이 주어지면 응용 프로그램이 비정상 종료될 수 있는 현상을 수정했다.

[CCI] cci_schema_info()의 수행 결과 페치 시 칼럼의 인덱스 여부에 대해 잘못된 정보를 출력하는 오류(CUBRIDSUS-7174)

cci_schema_info() 함수의 수행 결과를 페치(fetch)하는 경우 어떤 칼럼에 인덱스 존재 여부를 나타내는 값을 잘못 출력하는 오류를 수정했다.

[CCI] cci_get_result_info() 함수의 리턴 값 중 역 인덱스의 존재 여부에 대해 잘못된 정보를 출력하는 문제(CUBRIDSUS-7178)

cci_get_result_info()의 리턴 값 중 역(reverse) 인덱스의 존재 여부에 대해 잘못된 정보를 출력하는 문제를 수정했다.

```
col_info = cci_get_result_info(req, &cmd_type, &col_count);
if (col_info[i].is_reverse_index)
  printf("reverse_index");
```

[JDBC, CCI] 응용 프로그램에서 연결 URL이 잘못되었음에도 불구하고 오류가 발생하지 않는 문제 수정(CUBRIDSUS-7967)

JDBC/CCI 응용 프로그램에서 연결 URL이 잘못되는 경우 오류를 발생하도록 수정했다. 이 수정으로 인해, 수정 이전 버전에서는 DB 이름 뒤에 ":" 두 개가 생략될 수 있었으나, 수정 이후 버전에서는 이를 허용하지 않게 되었다.

```
URL=jdbc:CUBRID:192.168.0.1:33000:demodb:::?althosts=192.168.0.2:33000,192.168.0.3:33000
```

그리고, 수정 이후 Linux용 버전에서는 URL의 데이터베이스 이름에 "?"를 포함할 수 있게 되었다.

[JDBC, CCI] Windows용 CUBRID에서 JDBC 또는 CCI 응용 프로그램으로 VARCHAR 칼럼에 ""인 값을 바인딩하면 발생하는 precision 오류 수정(CUBRIDSUS-9306)

JDBC 또는 CCI 응용 프로그램으로 **VARCHAR** 타입 칼럼에 공백 문자열("")을 바인딩하면 발생하는 precision 오류를 수정했다. 이 문제는 Windows 버전에서만 발생하는 문제이다.

유틸리티

CSQL에서 질의문과 ; 다음에 주석이 존재하면 질의 수행에 실패하는 문제(CUBRIDSUS-6381)

CSQL 인터프리터에서 질의문과 세미콜론(;) 다음에 주석이 존재할 때 주석 제일 뒤에 세미콜론을 추가하지 않으면 질의 수행에 실패하는 문제를 수정했다.

```
SELECT 1; --
```

CSQL 세션 명령어를 UP 커서 키로 재입력 가능하도록 개선(CUBRIDSUS-8646)

CSQL 인터프리터에서 세션 명령어는 UP 커서 키로 재입력되지 않았으나, 이를 가능하도록 수정했다.

```
csql> ;sc tbl
// UP 커서 키로 방금 전 실행한 세션 명령어를 재입력할 수 있게 되었다.
csql> ;sc tbl
```

CSQL -i 옵션을 이용하여 파일로 질의문 입력 시 질의문이 매우 길거나 여러 라인으로 작성되면 질의문이 잘못 변경될 수 있는 문제(CUBRIDSUS-7848)

CSQL 인터프리터에서 -i 옵션을 이용하여 파일로 질의문을 입력할 때 질의문이 매우 길거나 여러 라인으로 작성되면, 긴 질의를 나누어 저장하는 버퍼의 끝 또는 라인의 끝에 나타나는 빈 문자열을 잘라냄으로 인해 질의문이 변경되어 의도하지 않은 값이 입력되거나 에러가 발생할 수 있는 문제를 수정했다.

CSQL에서 ;를 두번 입력해야 특정 질의가 수행되는 현상(CUBRIDSUS-5963)

시스템 파라미터인 **no_backslash_escapes**의 값을 no로 설정한 환경에서, "INSERT INTO t VALUES(1,'₩'')"와 같이 이스케이프 문자가 포함된 질의문을 CSQL 인터프리터에서 입력할 경우 세미콜론(;)을 한번 더 입력해야만 질의가 수행되는 문제를 수정했다.

CSQL에서 질의 계획 보기 수행 중 Ctrl+C한 후 스키마 내용을 조회하면 비정상 종료되는 현상(CUBRIDSUS-8456)

CSQL 인터프리터에서 ;info plan으로 질의 계획 보기 수행 중 Ctrl+C한 후 ;sc <table> 명령으로 스키마 내용을 조회하면 비정상 종료되는 현상을 수정했다.

데이터베이스 서버 종료 후 질의를 재차 수행 시 CSQL이 비정상 종료되는 현상(CUBRIDSUS-6256)

데이터베이스 서버 종료 후 CSQL 인터프리터에서 질의 수행 시 처음에는 "ERROR: Your transaction has been aborted by the system due to server failure or mode change."라는 정상 오류를 출력하나, 질의를 다시 수행하면 CSQL 인터프리터가 비정상 종료되는 현상을 수정했다.

Windows 용 CSQL에서 테이블 혹은 칼럼 명에 특수문자가 포함된 질의를 수행하면 CSQL 인터프리터가 비정상 종료되는 현상(CUBRIDSUS-9054)

Windows 버전의 CSQL 인터프리터에서 테이블 혹은 칼럼 명에 '%'와 같은 특수문자가 포함된 질의를 수행하면 CSQL 인터프리터가 비정상 종료되는 현상을 수정했다.

```
SELECT * FROM [as%];
```

broker_log_runner 실행 이후 출력 파일에 질의 수행 시간 포함(CUBRIDSUS-8237)

broker_log_runner 실행 이후 -o 옵션으로 지정한 출력 파일에 질의 수행 시간을 포함하도록 수정했다.

```
// broker_log_runner로 수행하면 prepare/execute/end tran 시간이 추가로 표시된다.
-------------- query -----------------
SELECT * FROM xx;
cci_prepare exec_time : 0.000
cci_execute exec_time : 0.000
cci_execute:1
---------- query plan --------------
 ...
---------- query result --------------
 ...
cci_end_tran exec_time : 0.000
```

```
// 수행중 에러가 발생하면 에러 메세지도 같이 표시된다.
-------------- query -----------------
INSERT INTO t1 VALUES (DATE '11/11/1994');
cci_prepare elapsed time : 0.000
replay_sqllog.txt:
server error : -493 Syntax: before ' VALUES (DATE '11/11/1994'); ' Unknown class "t1". insert into t1 values (date '11/11/1994')
cci_end_tran elapsed_time : 0.000
```

loaddb 유틸리티에서 --error-control-file 옵션을 사용하여 수행할 때의 성능 개선(CUBRIDSUS-7424)

loaddb 유틸리티 수행 시 --error-control-file 옵션이 설정되어 있을 때 데이터베이스 서버에 한 번에 하나의 행만 전송하도록 했으나 한 번에 여러 개의 행을 전송할 수 있도록 수정하여, 해당 옵션을 사용할 때의 성능이 개선되었다.

고유 키 위반 에러를 무시하도록 --error-control-file에 -670 에러 번호를 설정하고 한 테이블에 50만건의 데이터를 로딩하는 시험에서 고유 키 위반 에러 개수가 5만 건일 때, 수정 이전 대비 수정 이후 버전에서 약 1.5배의 성능이 향상되었다.

loaddb 유틸리티의 --error-control-file 옵션이 정상 동작하지 않는 문제(CUBRIDSUS-7484)

loaddb 유틸리티 수행 중 특정 에러는 무시하고 계속 진행하도록 --error-control-file 옵션을 설정했음에도 불구하고, 해당 에러가 발생하면 수행을 종료하는 문제를 수정했다.

```
cubrid loaddb -u dba -c 10000 -d tbl.ldb --error-control-file=error_lst testdb

Time: 04/04/12 20:05:47.066 - WARNING *** file ../../src/storage/heap_file.c, line 9678 CODE = -48 Tran = 1, EID = 16661
Accessing deleted object 2|607786|233.
```

loaddb 수행 중 서버에서 Warning이 발생하면 계속 수행하지 않고 종료하는 문제(CUBRIDSUS-6647)

loaddb 수행 중 서버에서 Warning이 발생하면 커밋되지 않은 레코드들을 모두 롤백하면서 **loaddb**의 수행을 종료하는 문제가 존재했으나, Warning 이후에도 계속 수행하도록 수정했다.

CLOB 칼럼이 있는 데이터 파일을 로드 시 loaddb 프로세스가 비정상 종료되는 문제(CUBRIDSUS-6330)

CLOB 칼럼이 있는 **unloaddb** 데이터 파일을 가지고 **loaddb** 유틸리티를 수행하면 **loaddb** 프로세스가 비정상 종료되는 문제를 수정했다.

alterdbhost 유틸리티의 오류 메시지가 잘못 출력되는 문제(CUBRIDSUS-7790)

cubrid alterdbhost 유틸리티에서 호스트 이름을 잘못 입력할 경우 오류 메시지가 "No error message available. "로 잘못 출력되었으나 상황에 맞는 메시지를 정상 출력하도록 수정했다.

```
$ cubrid alterdbhost --host=my_wrong_host_name
```

데이터베이스 생성 도중 ctrl+C로 작업 취소 시 비정상적인 오류를 출력하는 문제(CUBRIDSUS-8864)

cubrid createdb 실행 도중 ctrl+c를 눌러서 작업을 취소하면 "*** FATAL ERROR *** Internal error: logical log page -9 may be corrupted." 오류가 발생하는 문제를 수정했다.

cubrid heartbeat status 수행 시 출력되는 프로세스 상태 정보 변경(CUBRIDSUS-5212)

cubrid heartbeat status 수행 시 프로세스 상태 정보가 더 상세히 출력되도록 수정했다.

프로세스 현황	상태	출력 정보	
		수정 이전	수정 이후
cub_server가 HA 프로세스로 등록됨	standby	registered	registered_and_standby
cub_server가 heartbeat 리소스로 등록됨	to_be_standby	registered	registered_and_to_be_standby
cub_server가 heartbeat 리소스에 등록됨	to_be_active	registered_and_active	registered_and_to_be_active

DB 생성 직후 백업한 볼륨으로 -d 옵션을 사용하여 복구 시도 시 실패하는 현상(CUBRIDSUS-7527)

DB 생성 직후 **cubrid backupdb**를 수행하여 백업한 볼륨에 -d 옵션으로 백업 시간을 지정하여 cubrid restoredb를 실행하면 실패하는 현상을 수정했다.

```
cubrid backupdb -S demodb
csql -S demodb -c "create table x"
cubrid restoredb -d backuptime demodb

FATAL ERROR ***
No error message available.
Please consult error_log file = /home1/user1/CUBRID/log/demodb_restoredb.err for additional information
... ABORT/EXIT IMMEDIATELY ...<<<---
```

서버 프로세스 강제 종료 후 데이터베이스 복구 도중 서버 프로세스가 멈추는(hang) 현상(CUBRIDSUS-7217)

장시간 트랜잭션 수행 중에 서버 프로세스를 강제 종료한 후, 재시작하여 데이터베이스 복구 도중 멈추는 현상을 수정했다.

브로커 상태를 주기적으로 출력하는 옵션이 오동작하는 오류(CUBRIDSUS-6413)

cubrid broker status 명령에서 -s 옵션으로 초를 설정했음에도 불구하고 1초로 설정한 경우를 제외하고는 설정한 값대로 동작하지 않는 오류를 수정했다.

CS 모드로 cubrid checkdb 유틸리티 수행 시 DB 서버 프로세스가 비정상 종료될 수 있는 문제(CUBRIDSUS-7434)

중복 값이 매우 많은 비고유(non-unique) 인덱스가 존재하는 상황에서 CS 모드로 **cubrid checkdb** 유틸리티 수행 시 DB 서버 프로세스가 비정상 종료될 수 있는 문제를 수정했다.

CUBRID Manager에서 질의 편집기 수행 종료 이후, cubrid.conf에서 설정한 잠금 타임아웃 값이 줄어들 수 있는 오류(CUBRIDSUS-7462)

CUBRID Manager에서 질의 편집기를 수행하면 잠금 타임아웃이 내부적으로 1초로 설정되는데, 이 질의 편집기를 종료하면 이것과 연결되었던 CAS는 시스템 파라미터인 **lock_timeout_in_secs**에서 설정한 값으로 잠금 타임아웃을 재설정한다. 이때 해당 CAS의 잠금 타임아웃이 설정 값의 1/1000로 줄어드는 오류로 인해 이 CAS에 연결되는 다른 응용 프로그램이 줄어든 잠금 타임아웃을 사용하는 문제를 수정했다. 이 CAS는 재시작될 때까지 줄어든 잠금 타임아웃을 유지한다.

단, **lock_timeout_in_secs**가 기본값인 -1로 설정된 경우 이 오류는 발생하지 않았다.

64-bit Windows용 CUBRID에서 압축 백업 수행 시 오류(CUBRIDSUS-9253)

64-bit Windows용 CUBRID에서 "**cubrid backupdb -z**"으로 압축 백업 시 "No error message available." 오류 메시지와 함께 수행에 실패하는 현상을 수정했다.

Windows용 CUBRID에서 2G 초과 파일을 포함하는 볼륨을 백업한 데이터베이스 복구에 실패하는 문제(CUBRIDSUS-7588)

Windows용 CUBRID에서 2G 초과 파일을 포함하는 볼륨을 백업한 데이터베이스를 복구하는 경우 "Trying to format disk volume "xxx" with an incorrect value [xxx] for number of pages.", "Restoredb cancelled or an error occurred." 오류 메시지와 함께 수행에 실패하는 문제를 수정했다.

CUBRID 유틸리티에서 나타나는 DB 사용자 이름이 항상 대문자로 출력되도록 수정(CUBRIDSUS-8198)

cubrid 유틸리티에서 나타나는 DB 사용자 이름이 항상 대문자로 출력되도록 수정했다.

오류 메시지

데이터베이스 로그 처리 중 치명적인 오류 발생 시 오류 로그 파일에 정보를 남기도록 개선(CUBRIDSUS-8652)

데이터베이스 로그를 처리하는 작업 중 발생하는 치명적인 오류에 대해 관련 정보를 오류 로그 파일에 남기도록 수정했다.

고유 키 위반 오류 메시지 출력 시 테이블, 인덱스 이름, 키 값을 출력하도록 개선(CUBRIDSUS-6885)

고유 키를 위반한 오류 메시지를 오류 로그에 출력할 때 테이블, 인덱스 이름, 키 값을 출력하도록 수정했다.

문법 에러(syntax error)나 의미 에러(semantic error) 발생 시에 에러 발생 위치를 좀 더 쉽게 찾아갈 수 있도록 에러 메시지 개선

잘못된 구문으로 시리얼 생성 시 출력되는 오류 메시지 개선(CUBRIDSUS-7119)

시리얼 이름으로 예약어를 사용하거나 범위를 벗어난 값을 사용하는 등 잘못된 구문으로 시리얼 생성 시 출력되는 오류 메시지에서 잘못된 부분의 위치를 가리킬 수 있도록 수정했다.

```
CREATE SERIAL s START WITH 5 INCREMENT BY 3 cache -10;

-- 수정 이전 버전의 오류 메시지
In line 1, column 52,
ERROR: invalid create serial
CREATE SERIAL identifier {START WITH integer} {INCREMENT BY integer}
serial_min_max {CYCLE|NOCYCLE} {CACHE unsigned_integer|NOCACHE}

-- 수정 이후 버전의 오류 메시지
ERROR: In line 1, column 52 before '10; '
Syntax error: unexpected '-', expecting UNSIGNED_INTEGER
```

인덱스 이름이 잘못된 경우 출력되는 오류 메시지 개선 (CUBRIDSUS-7122)

인덱스를 생성할 때 출력되는 인덱스 이름이 잘못된 경우 출력되는 오류 메시지에 오류 원인이 되는 인덱스 이름을 출력하도록 수정했다.

```
CREATE TABLE t(i int);
CREATE INDEX INDEX ON t(i);
ERROR: In line 1, column 14 before ' on t(i); '
```

```
Syntax error: unexpected 'index'
```

데이터베이스 구동 시 호스트 IP 주소가 잘못된 경우 이를 확인할 수 있는 에러 메시지를 출력하도록 개선(CUBRIDSUS-6189)

데이터베이스 구동 시 호스트 이름에 매핑되는 IP 주소가 잘못된 경우 이를 확인할 수 있는 에러 메시지를 서버 에러 로그 파일에 출력하도록 수정했다.

DDL문 방지 파라미터 혹은 WHERE 절이 있는 질의 방지 파라미터를 yes로 설정하면 상황이 구체적으로 나타나도록 오류 메시지 개선(CUBRIDSUS-8096)

DDL문 방지 파라미터인 **block_ddl_statement** 혹은 **WHERE** 절이 있는 질의 방지 파라미터인 **block_where_statement**를 yes로 설정하면 "Authorization failure."라는 오류 메시지를 출력했으나, 상황이 구체적으로 나타나도록 오류 메시지를 수정했다.

```
DDL statement is not allowed by configuration (block_ddl_statement=yes).
Statement without WHERE clause is not allowed by configuration (block_nowhere_statement=yes).
```

분할 테이블 이름을 이미 존재하는 다른 분할 테이블 이름으로 RENAME할 때 잘못된 오류 메시지(CUBRIDSUS-7176)

분할 테이블 이름을 이미 존재하는 다른 분할 테이블 이름으로 **RENAME**할 때 오류 메시지의 분할 테이블 이름이 잘못 출력되는 문제를 수정했다.

```
CREATE TABLE pt1(i int) PARTITION BY hash(i) PARTITION 5;
CREATE TABLE pt2(i int) PARTITION BY hash(i) PARTITION 5;
RENAME pt1 AS pt2;
-- 수정 이전 버전에서 위의 질의 수행 시 분할 테이블 이름을 출력했다.
ERROR: Class "pt2__p__p0" already exists
-- 수정 이후 다음과 같이 테이블 이름을 정상 출력한다.
ERROR: Class "pt2" already exists
```

2008 R2.2 CCI 드라이버로 접속 시 비정상적인 에러코드와 메시지(CUBRIDSUS-8924)

2008 R2.2 CCI 드라이버로 데이터베이스 서버에 접속하면 에러코드와 메시지를 비정상적으로 출력하는 문제를 수정했다.

```
// 정상 오류 메시지의 예
execute error - err_code : -670, err_msg : Operation would have caused one or more unique
constraint violations.
```

```
// 비정상 오류 메시지의 예
execute error - err_code : -2, err_msg : yyybOperation would have caused one or more unique
constraint violations
```

ADDDATE, SUBDATE 함수에 적합하지 않은 인자 입력 시 부적절한 오류 메시지(CUBRIDSUS-8437)

ADDDATE, **SUBDATE** 함수에 적합하지 않은 인자를 입력하면 출력하는 상황에 맞지 않는 오류 메시지를 수정했다.

```
SELECT ADDDATE('1991-01-01 00:00:00', -10000000);
// 수정 이전 버전에서는 아래와 같이 부적절한 오류 메시지를 출력했다.
ERROR: before ' , -10000000); '
No error message available.

SELECT SUBDATE(SYSDATE, 9999999);
// 수정 이전 버전에서는 아래와 같이 부적절한 오류 메시지를 출력했다.
ERROR: Execute: Query execution failure #931.
```

SHARED 속성 칼럼에 인덱스 생성 시 부적절한 오류 메시지(CUBRIDSUS-7132)

SHARED 속성 칼럼에 인덱스 생성 시 출력하는 상황에 맞지 않는 오류 메시지를 수정했다.

```
CREATE TABLE t1(i INT SAHRED 10);
CREATE INDEX ON t1(i);

// 수정 이전 에러 메시지
ERROR: before ' ); ' i is not defined.

// 수정 이후 에러 메시지
ERROR: Cannot create index on attribute "i", defined with a shared value.
```

유효하지 않은 최대값 또는 최소값으로 시리얼 생성 시도 시 잘못된 오류 메시지(CUBRIDSUS-7150)

유효하지 않은 최대값 또는 최소값으로 시리얼 생성을 시도하면 출력하는 잘못된 오류 메시지를 수정했다.

```
// 수정 이전 버전에서 무효한 최대값인 경우 "Minimum value is invalid" 라고 오류를 잘못
출력했다.
CREATE SERIAL s START WITH 3 INCREMENT BY 1 MAXVALUE 1;

// 수정 이전 버전에서 무효한 최소값인 경우 "Maximum value is invalid" 라고 오류를 잘못
```

```
출력했다.
CREATE SERIAL s START WITH 3 INCREMENT BY 1 MINVALUE 5;
```

잠금 타임아웃 발생 시 서버 오류 메시지에 한 트랜잭션의 종료를 대기 중이던 전체 트랜잭션 정보를 출력하도록 개선(CUBRIDSUS-7272)

잠금 타임아웃 오류 메시지에 해당 트랜잭션의 종료를 대기 중이던 다른 트랜잭션 정보를 모두 출력하도록 개선했다. 이와 함께, 잠금 타임아웃의 메시지 출력 포맷을 설정하던 **lock_timeout_message_type** 파라미터는 더 이상 사용하지 않는다.

```
Your transaction (index 2, user1@host1|9808) timed out waiting on IX_LOCK lock on class tbl.
You are waiting for user(s) user1@host1|csql(9807), user1@host1|csql(9805) to finish.
```

잠금 대기 중에 질의 타임아웃이 발생했음에도 불구하고 잠금 타임 아웃 오류가 발생하는 문제(CUBRIDSUS-7215)

잠금 대기 중에 질의 타임아웃(query timeout)이 발생했음에도 불구하고 질의 타임아웃 오류가 발생하지 않고 잠금 타임아웃(lock timeout) 오류가 발생하는 문제를 수정했다.

설정, 빌드 및 설치

특정 오류 번호에 대해 콜 스택 출력 여부를 설정하는 시스템 파라미터의 동적 변경을 허용하도록 개선(CUBRIDSUS-5717)

데이터베이스 구동 중에 특정 오류 번호에 대해 콜 스택(call stack) 출력 여부를 설정하는 시스템 파라미터인 **call_stack_dump_activation_list, call_stack_dump_deactivation_list**의 동적 변경을 허용하도록 개선했다. 또한, 치명적 에러(fatal error)들은 **call_stack_dump_activation_list**에 별도의 설정이 없어도 기본적으로 콜 스택 정보가 출력되도록 수정했다.

cub_master, cub_broker 프로세스의 유닉스 도메인 소켓 파일의 위치를 바꿀 수 있도록 수정(CUBRIDSUS-6910)

Linux용 CUBRID에서 **cub_master**, **cub_broker** 프로세스 수행 시 생성되는 유닉스 도메인 소켓(unix domain socket) 파일의 위치를 **CUBRID_TMP** 환경 변수를 통해 변경할 수 있도록 수정했다. **cub_master**의 유닉스 도메인 소켓 파일의 위치는 기존에는 /tmp였으며 수정 이후에도 **CUBRID_TMP** 환경 변수를 설정하지 않으면 기존 위치와 같다. **cub_broker**의 유닉스 도메인 소켓 파일의 위치는 기존에는 $CUBRID/var/CUBRID_SOCK이었으며, 수정 이후에도

CUBRID_TMP 환경 변수를 설정하지 않으면 기존 위치와 같다. **CUBRID_TMP** 환경 변수를 통해 다음과 같은 두 가지 문제를 회피할 수 있다.

1. **cub_broker**용 유닉스 도메인 소켓 파일을 저장하는 $CUBRID/var/CUBRID_SOCK 경로의 최대 길이는 108인데, CUBRID의 설치 경로가 길어서 경로 길이가 108을 넘는 경우.
2. /tmp 디렉터리의 파일을 관리 상의 목적 등으로 임의로 삭제하는 경우.

32bit 버전에서 시스템 파라미터인 data_buffer_pages의 최대값을 2G로 제한(CUBRIDSUS-6257)

32bit 버전에서 시스템 파라미터인 **data_buffer_pages**의 최대값을 2G로 제한하도록 수정했다.

인덱스 스캔 관련 OID 버퍼 파라미터 설정 값의 크기가 1K일 때 데이터베이스를 구동하지 못하는 현상(CUBRIDSUS-8958)

인덱스 스캔 관련 OID 버퍼 파라미터인 **index_scan_oid_buffer_size** 설정 값의 크기가 1K일 때 데이터베이스를 구동하지 못하는 현상을 수정했다.

데이터베이스 서버 프로세스의 버퍼 크기 설정 값이 장비의 물리적 메모리 용량보다 크면 데이터베이스 생성에 실패하는 현상(CUBRIDSUS-7763)

데이터베이스 서버 프로세스가 메모리 내에 캐시하는 데이터 버퍼의 크기를 설정하는 시스템 파라미터인 **data_buffer_size**의 값이 물리적 메모리 용량보다 크더라도 데이터베이스를 생성할 수 있도록 수정했다.

시스템 파라미터인 data_buffer_pages의 크기를 시스템의 가용 크기보다 크게 설정하면 서버 프로세스가 비정상 종료되는 현상(CUBRIDSUS-6350)

시스템 파라미터인 **data_buffer_pages**의 크기를 시스템의 가용 크기보다 크게 설정한 후 구동하면 서버 프로세스가 비정상 종료되는 현상을 수정했다.

시스템 파라미터인 ha_node_list 에 구분자인 @이 없으면 HA 시작 시 서버 프로세스가 비정상 종료되는 문제(CUBRIDSUS-6474)

HA 환경에서 cubrid_ha.conf의 **ha_node_list**에 그룹 이름과 멤버 호스트 이름을 구분하는 @이 없으면 **cubrid heartbeat start** 수행 시 서버 프로세스가 비정상 종료되는 문제를 수정했다.

Windows 용 CUBRID에서 ha_mode 파라미터의 값을 yes 설정하면 CUBRID 서비스 종료를 시도해도 종료되지 않는 문제(CUBRIDSUS-6982)

Windows 용 CUBRID에서 **ha_mode** 파라미터의 값을 yes 설정하면 **cubrid service stop** 을 수행해도 CUBRID 서비스가 종료되지 않는 문제를 수정했다.

참고로, HA 기능은 Windows에서는 지원되지 않는다.

cubrid_broker.conf 파일 없이 브로커 유틸리티를 수행하면 잘못된 오류 메시지를 출력하는 문제(CUBRIDSUS-8932)

cubrid_broker.conf 파일 없이 브로커를 시작하면 브로커의 구동에 실패하면서 "The socket path is too long (>108): /home/CUBRID/var/CUBRID_SOCK/"이라는 비정상적인 오류 메시지를 출력했으나 정상적인 오류 메시지를 출력하도록 수정했다.

cubrid_broker.conf 파일 없이 **cubrid broker stauts, cubrid broker stop**를 수행하면 오류 메시지를 "cubrid broker is not running."으로 잘못 출력했으나, 정상적인 오류 메시지를 출력하도록 수정했다.

데이터베이스 서버에 동시 접속을 허용하는 최대 접속 개수를 10000개로 늘림(CUBRIDSUS-7233)

데이터베이스 서버에 동시 접속을 허용하는 최대 개수를 설정하는 **max_clients** 시스템 파라미터의 최대값 제한을 1024에서 10000으로 늘렸다. 참고로, 클라이언트의 DB 접속 여부에 관계 없이 **max_clients**의 개수를 크게 설정할수록 메모리 사용량이 증가하므로 이에 주의하도록 한다.

브로커 파라미터 값이 잘못되었을 때 부적절한 오류 메시지를 출력하는 문제(CUBRIDSUS-9280)

브로커 파라미터 값이 잘못되었을 때 "Error: can't find cubrid_broker.conf"라는 부적절한 오류 메시지를 출력하는 문제를 수정했다.

브로커 파라미터의 SLOW_LOG 설정을 동적으로 변경하면 각 CAS의 SQL_LOG 설정이 변경되는 문제(CUBRIDSUS-7592)

broker_changer로 SLOW_LOG 설정을 동적으로 변경하면 각 CAS의 SQL_LOG 설정이 변경되는 문제를 수정했다.

```
broker_changer query_editor SLOW_LOG ON
```

브로커 접속을 제한하는 ACCESS_CONTROL_FILE 파일의 작성 형식 확장(CUBRIDSUS-5449)

브로커에 접속하는 응용 클라이언트를 제한하는 **ACCESS_CONTROL_FILE** 파일에 데이터베이스 이름과 같은 데이터베이스 사용자에 대하여 여러 개의 IP 목록 파일을 한 줄 또는 여러 줄에 작성할 수 있도록 했다.

```
dbname:dbuser1:IPread.txt, IPwrite.txt
dbname:dbuser2:IPread.txt
dbname:dbuser2:IPwrite.txt
dbname:dbuser2:IPexternal.txt
```

TMPDIR 환경 변수 설정 시 Linux sh 패키지로 CUBRID 설치에 실패하는 문제(CUBRIDSUS-8005)

TMPDIR 환경 변수의 값이 설정되어 있으면 Linux sh 패키지를 이용하여 CUBRID를 설치하는데 실패하는 문제를 수정했다.

RPM 패키지의 의존성 오류(CUBRIDSUS-7809)

RPM 패키지 설치 시 의존성(dependencies) 오류를 수정했다.

CUBRID RPM 빌드 시 반드시 필요한 라이브러리를 모두 검사하도록 수정(CUBRIDSUS-7611)

CUBRID RPM 빌드 시 필요한 라이브러리 중 누락된 일부를 검사 대상에 포함하도록 수정했다.

Windows 버 전 설치 후 "제어판 > 프로그램 제거" 화면의 버전 명에 빌드 번호 명기(CUBRIDSUS-9282)

Windows 버전 설치 후 "제어판 > 프로그램 제거" 화면의 버전 명에 빌드 번호를 명기했다. 수정 이전에는 9.0.0과 같이 출력되었으나 수정 이후 9.0.0.xxxx와 같이 출력된다.

Ubuntu에서 CUBRID 소스를 빌드할 때 Warning 이 발생하는 현상(CUBRIDSUS-7812)

Ubuntu에서 configure를 수행할 때 automake 1.11.3을 쓰게 되면서 Warning이 발생하는 현상을 수정했다.

기타

CAS에 보낸 SIGTERM이 다른 응용 프로그램에 전달될 수 있는 문제(CUBRIDSUS-6693)

CAS 재시작을 위해 브로커가 CAS에 보낸 SIGTERM 시그널이 다른 응용 프로그램에 전달될 수 있는 문제를 수정했다.

질의 수행 시 영향 받는 행의 개수를 잘못 출력하는 경우(CUBRIDSUS-7135)

질의 수행 시 영향을 받는 행의 개수를 잘못 출력하는 경우가 존재했으나 이를 수정했다.

```
INSERT INTO t1 SELECT ROWNUM FROM db_class;
// 수정 이전 1 출력
1 row affected.
// 수정 이후 정상 출력
```

```
52 rows affected.
```

2.5 주의 사항

9.0 Beta 주의 사항

9.0 Beta 이전 버전과 DB 볼륨이 호환되지 않음(CUBRIDSUS-5238)

9.0 Beta 이전 버전과 DB 볼륨이 호환되지 않으므로 **cubrid unloaddb/loaddb**를 이용하여 데이터베이스를 마이그레이션해야 한다. 보다 자세한 사항은 매뉴얼 및 본 릴리스 노트의 CUBRID 9.0 Beta로 업그레이드하는 방법 절을 참고한다.

이전 버전부터의 주의 사항

2008 R4.1 버전부터 CCI_DEFAULT_AUTOCOMMIT 의 기본값이 ON으로 바뀜(CUBRIDSUS-5879)

2008 R4.1 버전부터 CCI 인터페이스로 개발된 응용 프로그램의 자동 커밋 모드에 영향을 주는 브로커 파라미터인 **CCI_DEFAULT_AUTOCOMMIT**의 기본값이 ON으로 변경되었다. 따라서 CCI 및 CCI로 개발된 인터페이스(PHP, ODBC, OLE DB 등) 사용자는 응용 프로그램의 자동 커밋 모드가 이에 적합한지 살펴보아야 한다.

2008 R4.0 버전부터 페이지 단위의 옵션 및 파라미터가 볼륨 크기 단위로 바뀜(CUBRIDSUS-5136)

cubrid createdb 유틸리티의 DB 볼륨 크기 및 로그 볼륨 크기를 지정할 때 페이지 단위를 사용하는 옵션들(-p, -l, -s)은 제거될 예정이므로, 2008 R4.0 Beta 이후 새로 추가된 옵션들(--db-volume-size, --log-volume-size, --db-page-size, --log-page-size)을 사용한다.

cubrid addvoldb 유틸리티의 DB 볼륨 크기를 지정하는 경우에도 페이지 단위를 사용하지 않고 2008 R4.0 Beta 이후 새로 추가된 옵션(--db-volume-size)을 사용한다.

페이지 단위의 시스템 파라미터들은 추후 제거될 예정이므로 바이트 단위의 새로운 시스템 파라미터를 사용할 것을 권장한다. 관련 시스템 파라미터들에 대한 내용은 아래를 참고한다.

2008 R4.0 Beta 이전 사용자는 DB 볼륨 크기 설정 시 주의(CUBRIDSUS-4222)

2008 R4.0 Beta 버전부터 DB 생성 시 데이터 페이지 및 로그 페이지의 크기 기본값이 4KB에서 16KB로 변경되었으므로, DB 볼륨을 페이지 개수로 지정하여 생성하는 경우 볼륨의 바이트 크기가 기대와 다를 수 있음에 주의한다. 아무런 옵션도 주지 않을 경우 이전 버전에서는 4KB의 페이지 크기로 100MB의 DB 볼륨을 생성했으나, 2008 R4.0 버전부터는 16KB의 페이지 크기로 512MB의 DB 볼륨을 생성하게 된다.

그리고, DB 볼륨의 생성 가능한 최소 크기를 20MB로 제한했으므로 이보다 작은 크기의 DB 볼륨은 생성할 수 없다.

2008 R4.0 이전 버전의 일부 시스템 파라미터들의 기본값 변경(CUBRIDSUS-4095)

2008 R4.0부터 다음 시스템 파라미터들의 기본값이 변경되었다.

DB 서버가 허용하는 동시 연결 개수를 설정하는 **max_clients**의 기본값, 인덱스 페이지 생성 시 향후 업데이트를 대비하여 확보하는 여유 공간 비율을 설정하는 **index_unfill_factor**의 기본값이 변경되었으며, 바이트 단위 시스템 파라미터의 기본값이 기존 페이지 단위 시스템 파라미터의 기본값보다 커져서 별도의 설정을 하지 않는 경우 더 많은 메모리를 사용하게 되었다.

기존 시스템 파라미터	추가된 시스템 파라미터	기존 기본값	변경된 기본값 (단위: 바이트)
max_clients	-	50	100
index_unfill_factor	-	0.2	0.05
data_buffer_pages	data_buffer_size	100M(페이지 크기=4K)	512M
log_buffer_pages	log_buffer_size	200K(페이지 크기=4K)	4M
sort_buffer_pages	sort_buffer_size	64K(페이지 크기=4K)	2M
index_scan_oid_buffer_pages	index_scan_oid_buffer_size	16K(페이지 크기=4K)	64K

또한, **cubrid createdb**로 DB 생성 시 데이터 페이지 크기와 로그 페이지 크기의 최소값이 1K에서 4K로 변경되었다.

시스템 파라미터를 잘못 설정하면 DB 서비스, 유틸리티 및 응응 프로그램이 구동되지 않음(CUBRIDSUS-5375)

cubrid.conf 또는 cubrid_ha.conf에 정의되지 않은 시스템 파라미터를 설정하거나, 페이지 단위의 시스템 파라미터와 바이트 단위의 시스템 파라미터가 동시에 사용되거나, 시스템 파라미터 값이 허용 범위를 벗어나면 이와 관련된 DB 서비스, 유틸리티 및 응응 프로그램이 구동되지 않는다.

CUBRID 32bit 버전에서 data_buffer_size에 2G를 초과하는 값을 설정하면 DB 구동에 실패함(CUBRIDSUS-5349)

CUBRID 32bit 버전에서 **data_buffer_size**가 2G를 초과하는 값으로 설정되는 경우 DB 구동에 실패한다. 32bit 버전에서는 OS의 한계로 인해 설정값이 2G를 초과할 수 없음에 주의한다.

2008 R3.0 이하 버전에서 사용하던 GLO 클래스 지원 중단(CUBRIDSUS-3826)

CUBRID 2008 R3.0 이하 버전은 glo(Generalized Large Object) 클래스를 사용하여 Large Object를 처리했으나, CUBRID 2008 R3.1 이상 버전 glo 클래스를 제거하고 BLOB, CLOB 타입(이하 LOB)을 지원한다. (매뉴얼의 'CUBRID SQL 설명서 > 데이터 타입 > BLOB/CLOB 데이터 타입' 참고)

기존의 glo 클래스 사용자는 다음과 같이 작업할 것을 권장한다.

- GLO 데이터를 파일로 저장한 후 어플리케이션 및 DB 스키마에서 GLO를 사용하지 않도록 수정한다.
- unloaddb, loaddb 유틸리티를 이용하여 DB 마이그레이션을 한다.
- 변경한 어플리케이션에 맞게 파일을 LOB 데이터로 로딩하는 작업을 수행하도록 한다.
- 수정한 어플리케이션이 정상 동작하는지 확인한다.

참고로, **cubrid loaddb** 유틸리티는 GLO 클래스를 상속받거나 GLO 클래스 타입을 가진 테이블을 로딩하려는 경우 "Error occurred during schema loading" 오류 메시지와 함께 데이터 로딩을 중지한다.

GLO 클래스의 지원 중단에 따라 각 인터페이스 별로 삭제한 함수는 다음과 같다.

인터페이스	삭제한 함수
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

Windows Vista 이상 버전에서 CUBRID 유틸리티를 사용한 서비스 제어 시 권장 사항(CUBRIDSUS-4186)

Windows Vista 이상 버전에서 **cubrid** 유틸리티를 사용하여 서비스를 제어하려면 명령 프롬프트 창을 관리자 권한으로 구동한 후 사용하는 것을 권장한다.

명령 프롬프트 창을 관리자 권한으로 구동하지 않고 **cubrid** 유틸리티를 사용하는 경우 UAC(User Account Control

) 대화 상자를 통하여 관리자 권한으로 수행될 수 있으나 수행 결과 메시지를 확인할 수 없다.

Windows Vista 이상 버전에서 명령 프롬프트 창을 관리자 권한으로 구동하는 방법은 다음과 같다.

- [시작> 모든 프로그램> 보조 프로그램> 명령 프롬프트]에서 마우스 오른쪽 버튼을 클릭한다.
- [관리자 권한으로 실행(A)]을 선택하면 권한 상승을 확인하는 대화 상자가 활성화되고, “예”를 클릭하여 관리자 권한으로 구동한다.

JDBC에서 연결 정보를 URL 스트링으로 입력하는 경우 물음표를 반드시 명시(CUBRIDSUS-3217)

JDBC에서 URL 스트링으로 연결 정보를 입력하는 경우 이전 버전에서는 물음표(?)를 입력하지 않더라도 속성(PROPERTY) 정보가 적용되었으나, CUBRID 2008 R3.0부터는 문법에 따라 반드시 물음표를 명시해야 하고 이를 생략할 경우 에러를 출력한다. 또한, 연결 정보 중 USERNAME과 PASSWORD가 없더라도 반드시 콜론(:)을 명시해야 한다.

```
URL=jdbc:CUBRID:127.0.0.1:31000:db1:::althosts=127.0.0.2:31000,127.0.0.3:31000  - 에러 처리
URL=jdbc:CUBRID:127.0.0.1:31000:db1:::?althosts=127.0.0.2:31000,127.0.0.3:31000  - 정상 처리
```

마스터 프로세스와 서버 프로세스 간 프로토콜 변경 및 두 개 버전을 동시에 운영하는 경우 포트 설정 필요(CUBRIDSUS-3564)

마스터 프로세스(**cub_master**)와 서버 프로세스(**cub_server**) 간 통신 프로토콜 변경으로 인해 CUBRID 2008 R3.0 이상 버전의 마스터 프로세스는 하위 버전의 서버 프로세스와 통신할 수 없고, 하위 버전의 마스터 프로세스도 2008 R3.0 이상 버전의 서버 프로세스와 통신할 수 없다. 따라서, 이미 하위 버전이 설치되어 있는 환경에서 새 버전을 추가 설치하여, 두 개 버전의 CUBRID를 동시에 운영하는 경우 각각 서로 다른 포트를 사용하도록 cubrid.conf의 **cubrid_port_id** 시스템 파라미터를 수정해야 한다.

DB 이름에 @를 포함할 수 없음(CUBRIDSUS-2828)

DB 이름에 @이 포함되는 경우 호스트 이름이 명시된 것으로 해석될 수 있으므로 이를 방지하기 위하여 **cubrid createdb**, **cubrid renamedb**, **cubrid copydb** 유틸리티 실행 시 DB 이름에 @를 포함할 수 없도록 수정했다.

CUBRID 소스 빌드 후 실행 시, 매니저 서버 프로세스 관련 오류 발생(CUBRIDSUS-3553)

사용자가 직접 빌드하여 설치하는 경우, CUBRID와 CUBRID 매니저를 각각 빌드하여 설치해야 한다. 만약, CUBRID 소스만 checkout하여 빌드 후 **cubrid service start** 또는 **cubrid manager start**를 실행하면, cubrid manager server is not installed라는 오류가 발생한다.

CUBRID 2008 릴리스 노트

3

릴리스 노트 공통

3.1 기본 설치 및 설정

지원 플랫폼 및 설치 권장 사양

CUBRID 2008 제품을 지원하는 플랫폼과 설치를 위한 하드웨어/소프트웨어 요구 사항은 아래 표와 같다.

지원 플랫폼	메모리 여유 공간	디스크 여유 공간	필요 소프트웨어
- Windows 32/64 Bit XP, 2003, Vista, Windows7 - Linux 계열 32/64 Bit Linux kernel 2.4 및 glibc 2.3.4 이상	1G 이상	2GB 이상[1]	- JRE 또는 JDK 1.6 이상 Java 저장 프로시저를 사용하는 경우 필요

2008 R4.0부터는 CUBRID 패키지 설치 시 CUBRID 매니저 클라이언트가 같이 설치되지 않는다. 따라서 CUBRID 매니저를 사용하려면 이를 추가로 설치해야 한다. CUBRID 설치 패키지는 http://ftp.cubrid.org에서 받을 수 있다.

CUBRID 매니저 및 CUBRID 쿼리 브라우저 설치 패키지와 JDBC, PHP, ODBC, OLE DB 등의 드라이버들도 http://ftp.cubrid.org에서 받을 수 있다.

CUBRID 엔진, 사용 도구 및 드라이버에 대한 자세한 정보는 http://www.cubrid.org를 참고한다.

1 처음 설치 시 약 500MB의 디스크 공간이 필요하며, 하나의 DB를 기본 옵션으로 생성할 경우 약 1.5GB의 디스크 공간이 필요하다.

버전 호환성과 운용성

응용 프로그램의 호환성

- 이전 버전의 JDBC, PHP, CCI API 등을 사용하는 응용 프로그램은 CUBRID의 최신 버전 데이터베이스에 접근할 수 있다. 다만, JDBC, PHP, CCI 인터페이스에 추가/개선된 기능을 사용하기 위해서는 CUBRID 2008 R4.1 버전의 라이브러리를 링크하거나 드라이버를 사용해야 한다.
- 새로운 예약어 추가 및 일부 질의에 대한 스펙 변경으로 인해 질의 결과가 이전 버전과 다를 수 있으므로 주의한다.
- GLO 클래스를 이용하여 개발된 응용은 BLOB, CLOB 타입에 맞는 응용 및 스키마로 변환하여 사용해야 한다.

CUBRID 매니저의 호환성

- CUBRID 매니저는 CUBRID 2008 R2.1 이상 버전의 서버에 대해서 하위 호환성을 보장하며, 각 서버 버전과 일치하는 CUBRID JDBC 드라이버를 사용한다. 하지만 CUBRID 매니저에서 제공하는 모든 기능을 제대로 사용하기 위해서는 CUBRID 서버 버전보다 높은 버전의 CUBRID 매니저를 사용해야 한다. CUBRID JDBC 드라이버는 CUBRID 설치 시 $CUBRID/jdbc 디렉터리에 포함되어 있다.[2]
- CUBRID 매니저의 Bit 버전과 JRE의 Bit 버전은 서로 동일하여야 한다. 예를 들어, 64Bit 버전 DB 서버라도 CUBRID Manager 32Bit 버전을 사용한다면 JRE 또는 JDK 32Bit 버전을 설치하여야 한다.
- 2008 R2.2 이상 버전의 드라이버는 CUBRID 매니저에 기본으로 내장되어 있으며, http://www.cubrid.org에서 별도로 받을 수도 있다.

2008 R4.0부터는 CUBRID 패키지 설치 시 CUBRID 매니저 클라이언트가 같이 설치되지 않는다. 따라서 CUBRID 매니저를 사용하기 위해서는 이를 추가로 설치해야 한다.

상호 운용성

CUBRID DB 서버와 브로커 서버를 분리하여 운영하는 경우, 서버 장비의 운영 체제가 다르더라도 상호 운용성을 보장한다. 단, DB 서버의 Bit 버전과 브로커 서버의 Bit 버전은 서로 동일하여야 한다. 예를 들어, Linux용 64Bit 버전 DB 서버는 Windows용 64Bit 버전 브로커 서버와 상호 운용이 가능하지만, 32Bit 버전 브로커 서버와는 상호 운용이 불가능하다.

---

2 $CUBRID는 CUBRID가 설치된 곳을 지정하는 Linux용 환경변수이며, Windows 환경에서는 %CUBRID%와 같은 형식으로 사용된다.

3.2 추가 정보

참고 문서

CUBRID 2008 제품과 함께 배포되는 문서는 아래와 같다.

문서	설명
릴리스 노트	CUBRID 릴리스 버전의 특징 및 이전 버전에서 변경된 사항과 관련된 정보를 포함한다.
매뉴얼	CUBRID 소개, 시작, CSQL 인터프리터, SQL 설명서, 관리자 안내서, 성능 튜닝, API 레퍼런스, CUBRID 매니저, CUBRID 매니저 관리자 안내서를 포함한다.

버그 리포트 및 사용자 피드백 제공 방법 안내

CUBRID 프로젝트에서는 사용자의 거침없는 버그 리포트와 솔직한 피드백을 기다리고 있으며, 아래 사이트에서 등록할 수 있다.

문서	설명
버그 리포트	CUBRID 오픈 소스 프로젝트: http://dev.naver.com/projects/cubrid/issue
사용자 피드백	CUBRID 오픈 소스 프로젝트: http://dev.naver.com/projects/cubrid/forum CUBRID 공식 사이트: http://www.cubrid.com

추가 정보 안내

CUBRID에 관한 유용한 정보는 아래 사이트에서 찾을 수 있다.

정보	사이트
CUBRID 제품 정보	http://cubrid.com/zbxe/product
CUBRID 라이선스 정보	http://cubrid.com/zbxe/bbs_oss_guide/32249
CUBRID 사용자 문서	http://cubrid.com/zbxe/developer
CUBRID 교육 서비스	http://cubrid.com/zbxe/education_overview

라이선스 안내

CUBRID의 서버 엔진에는 GNU GPL v2 or later 가 적용되고 CUBRID 매니저 및 인터페이스(API)에는 BSD 라이선스가 적용된다. 보다 상세한 정보는 CUBRID 공식 사이트의 라이선스 가이드를 참고한다.

3.3 드라이버 관련 주의 사항

현재 CUBRID가 지원하는 드라이버들은 JDBC, Node.js, CCI(CUBRID C API), PHP, PDO, Python, Perl, Ruby, ADO.NET, ODBC, OLE DB이며, JDBC와 Node.js, ADO.NET을 제외한 모든 드라이버들은 CCI 기반으로 개발되었으므로 CCI와 관련하여 변경된 사항은 CCI 기반 드라이버들에 영향을 미칠 수 있음에 주의한다.

4

CUBRID 8.4.3 릴리스 노트

4.1 개요

릴리스 노트 정보

본 문서는 CUBRID 2008 R4.3 버전에 관한 유용한 정보를 포함한다. 릴리스 노트의 최신 버전은 릴리스 정보 사이트(http://release.cubrid.org/ko)에서 확인할 수 있다.

CUBRID 2008 R4.3 이전 버전에 대한 자세한 내용은 CUBRID 2008 R4.1의 최신 패치 버전 릴리스 노트를 참조한다.

릴리스 노트 개정 내역

CUBRID 2008 R4.3 버전 릴리스 노트의 변경 사항은 아래와 같다.

작성 날짜	설명
2012년 11월	CUBRID 2008 R4.3 릴리스 (8.4.3.0146)

릴리스 특징

CUBRID 2008 R4.3은 CUBRID SHARD 기능을 통해 대용량 데이터 처리 환경의 편의를 제공한다. HA 복제 성능이 약 2배 정도 향상되었다. 또한, 잠금 대기를 유발하는 프로세스와 SQL 문을 쉽게 파악하기 위한 트랜잭션의 질의 수행 상태 출력, 지정 시간을 초과하는 질의문과 질의 실행 계획 정보 확인, HA 환경에서 복제 지연된 페이지 수와 지연 예상 시간을 포함하는 정보 출력 등과 같이 운영 편의성이 개선되었다. 그 외에 질의 수행, 드라이버 및 유틸리티 등에

서 100여 개의 오류를 수정했다.

CUBRID 2008 R4.3 릴리스는 CUBRID 2008 R4.1 패치 버전 및 그 이하 버전들의 수정 사항을 모두 포함한다.

CUBRID 2008 R4.3 버전의 주요 특징은 다음과 같다.

Sharding 기능 추가

다수의 장비로 수평 분할된 데이터베이스 환경을 용이하게 접근하기 위한 CUBRID SHARD 기능을 제공한다. CUBRID SHARD 기능은 응용 프로그램이 여러 장비에 분산된 데이터베이스를 하나의 데이터베이스로 보이도록 단일 뷰(single view)를 제공하며, 이들을 인지하고 특정 데이터베이스를 접근할 필요 없도록 투명성(transparency)을 제공한다.

HA 복제 성능 개선

HA 복제 성능을 개선하여 슬레이브 노드에서 처리량 기준으로 INSERT 성능이 약 2 배, UPDATE 성능이 약 1.4 배, DELETE 성능이 약 1.7 배 향상되었다.

상세 정보 출력으로 운영 편의성 개선

cubrid killtran 명령에 -q 옵션을 추가하여 잠금 대기를 유발하는 프로세스와 SQL 문을 쉽게 파악하기 위한 트랜잭션 정보들을 출력하게 했다. 그리고, 시스템 파라미터의 설정을 통해 지정 시간을 초과하는 질의문과 이에 대한 질의 실행 계획 정보를 로그 파일에 기록하게 했다. 또한, cubrid applyinfo 명령을 통해 HA 환경에서 복제 로그의 복사 지연 또는 반영 지연 정보를 출력하도록 개선했다.

100여 개의 오류 수정

SQL 구문, CCI 및 JDBC, loaddb, HA 등에서 100여 개의 오류를 수정했다.

보다 자세한 변경 사항은 아래의 CUBRID 2008 R4.3에서 변경된 사항을 참고한다.

데이터베이스 호환성

서버 버전을 하위 버전에서 업그레이드하는 경우, DB의 마이그레이션 작업을 수행해야 한다. 보다 상세한 사항은 CUBRID 2008 R4.3으로 업그레이드하는 방법을 참고한다.

CUBRID 2008 R4.3의 설치 방법

제품 설치

Linux용 설치 패키지는 바이너리를 포함하는 스크립트, tar.gz 압축 파일, Linux RPM 패키지 형태로 제공되며, 설치 방법은 매뉴얼의 'CUBRID 시작> 설치와 실행> Linux에서의 설치와 실행'을 참고한다.

Windows용 설치 패키지는 설치 마법사를 이용하여 설치할 수 있다. 설치 방법은 매뉴얼의 'CUBRID 시작> 설치와 실행> Windows에서의 설치와 실행'을 참고한다. 이와 함께 CUBRIDSUS-3267 Windows 환경에서 디렉터리 경로 설정 시 주의사항을 참고한다.

CUBRID 환경 변수 및 OS 환경 변수 설정

CUBRID를 사용하기 위해서는 CUBRID 환경 변수와 관련 OS 환경 변수를 설정해야 한다. (매뉴얼의 'CUBRID 시작 > 환경 변수 설정 및 CUBRID 서비스 시작 > 환경 변수 설정' 참고) 특히, CUBRID 매니저와 Java 저장 프로시저를 사용하기 위해서는 Java 환경 변수를 설정해야 한다. (매뉴얼의 'CUBRID SQL 설명서 > Java 저장 함수/프로시저 > Java 저장 함수/프로시저 사용을 위한 환경 설정' 참고)

CUBRID 2008 R4.3으로 업그레이드하는 방법

2008 R4.1에서 2008 R4.3으로 업그레이드하기

CUBRID 2008 R4.1 버전 사용자는 2008 R4.3를 설치한 후 기존의 데이터베이스와 환경 설정을 그대로 사용할 수 있다.

2008 R4.0에서 2008 R4.3으로 업그레이드하기

CUBRID 2008 R4.0, 2008 R4.0 Patch X 버전 사용자는 2008 R4.3을 설치한 후 기존의 환경 설정 파일에서 파라미터들의 값을 다음과 같이 변경해야 한다.

cubrid.conf

thread_stacksize의 기본값이 100K에서 1M으로 변경되었으므로 이 값을 설정하지 않은 사용자는 CUBRID 관련 프로세스들의 메모리 사용량을 살펴볼 것을 권장한다.

data_buffer_size의 최소값이 64K에서 16M으로 변경되었으므로 이 값을 16M 미만으로 설정한 사용자는 16M 이상으로 설정해야 한다.

cubrid_broker.conf

CCI_DEFAULT_AUTOCOMMIT의 기본값이 ON으로 변경되었으므로 이를 설정하지 않은 응용 프로그램 사용자가 기존과 같은 자동 커밋 모드를 유지하고 싶다면 OFF로 설정해야 한다.

APPL_SERVER_MAX_SIZE_HARD_LIMIT의 최소값이 1024M이며, **APPL_SERVER_MAX_SIZE**의 값을 설정한 사용자는 **APPL_SERVER_MAX_SIZE_HARD_LIMIT**의 값을 이보다 크게 설정할 것을 권장한다.

2008 R4.0 베타 이전 버전에서 2008 R4.3으로 업그레이드하기

CUBRID 2008 R4.0 Beta 버전 및 그 이전 버전에서 업그레이드하는 사용자는 아래의 CUBRID 2008 R4.3으로 업그레이드하는 방법을 참고하도록 한다.

업그레이드 주의 사항

기존 환경 설정 파일 보관

이전 버전의 $CUBRID/conf 디렉터리의 환경 설정 파일(cubrid.conf, cubrid_broker.conf, cm.conf)과 $CUBRID_DATABASES 디렉터리의 DB 위치 정보 파일(databases.txt)을 보관한다.[1]

새로 추가된 예약어 검사

CUBRID 설치 패키지에 포함 또는 http://ftp.cubrid.org 에서 배포되는 CUBRID 2008 R4.3용 예약어 검출 스크립트인 check_reserved.sql을 이용하여 예약어 사용 여부를 검사할 수 있으며, 예약어로 지정된 식별자를 사용하고 있을 경우 식별자를 수정해야 한다. (매뉴얼의 'CUBRID SQL 설명서 > 식별자' 참고)

DB 마이그레이션

CUBRID 2008 R4.3은 2008 R4.0 Beta 및 그 이전 버전과 DB 볼륨이 호환되지 않으므로, 이전 DB를 CUBRID 2008 R4.3으로 마이그레이션해야 한다. (아래의 DB 마이그레이션 절차참고)

2008 R4.0 Beta에서 마이그레이션하기 위해서는 다운로드 페이지에서 제공하는 **migrate_r40beta2ga** 유틸리티를 사용할 수 있다.

CUBRID 2008 R3.1부터 GLO를 지원하지 않으며 LOB 타입이 GLO 기능을 대체하게 되었으므로, GLO를 이용한 응용 및 스키마는 LOB 타입에 맞게 수정해야 한다. (아래의 GLO 클래스 사용자의 마이그레이션 참고)

2008 R3.1 및 그 이전 버전에서 마이그레이션하기 위해서는 **cubrid unloaddb/loaddb** 유틸리티를 사용한다.

복제 또는 HA 환경 재구성

2008 R4.0부터는 복제 기능을 더 이상 지원하지 않으므로, 이전의 복제 기능을 사용하는 시스템에서는 DB 마이그레이션 이후 HA 환경으로 재구성할 것을 권장한다. 또한, CUBRID 2008 R2.0, 2008 R2.1에서 제공된 Linux Heartbeat 기반의 HA 기능을 사용하는 시스템도 보다 안정적인 운영을 위해 DB 마이그레이션 이후 CUBRID Heartbeat 기반의 HA 환경으로 재구성해야 한다. (아래의 HA 환경에서 DB 마이그레이션 절차 참고)

HA 환경 구성은 매뉴얼의 '관리자 안내서 > CUBRID HA'를 참고하여 재설정해야 한다.

DB 마이그레이션 절차

2008 R3.x 및 그 이전 버전에서 2008 R4.3으로 마이그레이션

2008 R4.0 Beta 버전에서 마이그레이션

GLO 클래스 사용자의 마이그레이션

2008 R3.x 및 그 이전 버전에서 2008 R4.3으로 마이그레이션

2008 R3.x 및 그 이전 버전의 사용자는 아래의 표와 같이 2008 R4.3으로 DB 마이그레이션을 해야 한다.

또한, 기존의 GLO 클래스를 사용하고 있는 경우에는 추가 작업이 필요하다. (아래의 GLO 클래스 사용자의 마이그레

---

1 $CUBRID 또는 $CUBRID_DATABASES 형식의 환경변수는 Linux 환경에서 사용하는 형식이며 Windows 환경에서는 %CUBRID% 또는 %CUBRID_DATABASES%와 같은 형식으로 사용해야 한다.

이션 참고)

아래는 **cubrid unloaddb/loaddb** 유틸리티와 http://ftp.cubrid.org에서 별도 배포되는 check_reserved.sql 예약어 검출 스크립트를 이용하여 마이그레이션을 수행하는 방법이다. (매뉴얼의 '관리자 안내서 > 데이터베이스 마이그레이션'과 본 문서의 CUBRIDSUS-3826 GLO 클래스 지원 중단에 따른 주의 사항 참고)

단계	Linux 환경	Windows 환경
C1 단계: CUBRID Service 종료	% cubrid service stop	CUBRID Service Tray를 종료한다.
C2 단계: 예약어 검출 스크립트 실행	예약어 검출 스크립트가 위치하는 디렉터리에서 아래 명령을 실행한다. 검출 결과를 확인하여 마이그레이션 진행 또는 식별자 수정 작업을 진행한다. (허용되는 식별자는 관련 매뉴얼 참고) % csql -S -u dba -i check_reserved.sql testdb	
C3 단계: 이전 버전 DB 언로드	이전 버전의 databases.txt 및 conf 디렉터리 내 설정 파일을 별도 디렉터리에 보관한다. (C3a) cubrid unloaddb 유틸리티를 실행하고 이때 생성된 파일을 별도 디렉터리에 보관한다. (C3b) % cubrid unloaddb -S testdb 기존 DB를 삭제한다. (C3c) % cubrid deletedb testdb	
		이전 버전의 CUBRID를 제거한다.
C4 단계: 2008 R4.3 설치	본 문서에서 CUBRID 2008 R4.3의 설치 방법을 참고한다.	
C5 단계: DB 생성 및 데이터 로딩	DB를 생성할 디렉터리로 이동한 후, DB를 생성한다. (C5a) % cd $CUBRID/databases/testdb % cubrid createdb testdb (C3b)에서 보관한 파일을 가지고 cubrid loaddb 유틸리티를 실행한다. (C5b) % cubrid loaddb -s testdb_schema -d testdb_objects -i testdb_indexes testdb	
C6 단계: 새 버전 DB 백업	% cubrid backupdb -S testdb	
C7 단계: CUBRID 환경 설정 및 CUBRID Service 구동	환경 설정 파일을 수정한다. 이때, (C3a)에서 보관한 이전 버전의 환경 설정 파일을 새 버전에 맞게 수정한다. (시스템 파라미터 설정은 주의 사항 및 관련 매뉴얼 참고) % cubrid service start % cubrid server start testdb	CUBRID Service Tray> [Service Start]를 선택하여 서비스를 시작한다. 명령 프롬프트 창에서 DB 서버를 구동한다. % cubrid server start testdb

2008 R4.0 Beta 버전에서 마이그레이션

2008 R4.0 Beta 버전 사용자는 아래의 표와 같이 DB 마이그레이션을 해야 한다.

다운로드 페이지에서 제공되는 **migrate_r40beta2ga** 유틸리티를 이용하여 마이그레이션을 수행할 수 있다. 그러나 DB의 페이지 크기가 4K 미만인 볼륨은 **cubrid unloaddb/loaddb**를 이용하여 마이그레이션을 수행해야 한다. (200

8 R4.0 베타 이전 버전에서 2008 R4.3으로 업그레이드하기 참고)

마이그레이션 수행 중 실패하는 경우 백업해 놓았던 이전 버전의 DB를 복구(**cubrid restoredb**)하여 재시도한다.

단계	Linux 환경	Windows 환경
D1 단계: CUBRID Service 종료	% cubrid service stop	CUBRID Service Tray를 종료한다.
D2 단계: 이전 버전 DB 백업	이전 버전으로 복구하는 상황에 대비하기 위해 백업을 수행하고, 백업 파일을 별도 디렉터리(backup)에 보관한다. (D2a) % mkdir backup % cubrid backupdb -S -D backup testdb 이전 버전의 databases.txt 및 conf 디렉터리 내 설정 파일을 별도 디렉터리에 보관한다. (D2b)	
		이전 버전의 CUBRID를 제거한다.
D3 단계: 2008 R4.3 설치	본 문서의 CUBRID 2008 R4.3의 설치 방법을 참고한다.	
D4 단계: 마이그레이션 도구 실행	(D2b)에서 보관한 databases.txt를 2008 R4.3의 설치 디렉터리에 복사한다. (D4a) 아래와 같이 migrate_r40beta2ga 유틸리티를 실행한다. (D4b) % migrate_r40beta2ga testdb	
D5 단계: 새 버전 DB 백업	% cubrid backupdb -S testdb	
D6 단계: CUBRID 환경 설정 및 CUBRID Service 구동	보관한 이전 버전의 환경 설정 파일을 새 버전에 맞게 수정한다. (시스템 파라미터 설정은 주의 사항 및 관련 매뉴얼 참고) % cubrid service start % cubrid server start testdb	CUBRID Service Tray> [Service Start]를 선택하여 서비스를 시작한다. 명령 프롬프트 창에서 DB 서버를 구동한다. % cubrid server start testdb

GLO 클래스 사용자의 마이그레이션

GLO 클래스를 사용하는 경우, 2008 R3.1부터는 GLO 클래스를 지원하지 않으므로 BLOB 또는 CLOB 타입을 사용하도록 응용과 스키마를 변경해야 한다. 변경 작업이 용이하지 않다면 마이그레이션을 보류할 것을 권장한다.

HA 환경에서 DB 마이그레이션 절차

2008 R2.2 이상 버전에서 2008 R4.3으로 HA 마이그레이션

2008 R2.0 또는 2008 R2.1에서 2008 R4.3으로 HA 마이그레이션

2008 R2.2 이상 버전에서 2008 R4.3으로 HA 마이그레이션

아래는 브로커, 마스터 DB, 슬레이브 DB를 각각 별도 서버에 구축한 환경에서 현재 서비스를 중지하고 업그레이드를 수행하기 위한 가이드이다.

단계	설명
H1~H6 단계: 버전에 따라 마스터 노드에서 C1~C6 단계 또는 D1~D5 단계	마스터 노드에서 CUBRID 업그레이드 및 DB 마이그레이션을 수행하고, 2008 R4.3 DB를 백업한다.

단계	설명
를 수행	
H7 단계: 슬레이브 서버에 CUBRID 2008 R4.3 설치	슬레이브 서버에서 이전 버전의 DB는 삭제하고, 새 버전을 설치한다. 설치 방법은 본 문서의 CUBRID 2008 R4.3의 설치 방법을 참고한다.
H8 단계: 마스터 노드 백업본을 슬레이브 서버에서 복구	H6 단계에서 생성된 마스터 노드의 2008 R4.3 DB 백업본(예: testdb_bk*)을 슬레이브 서버에서 복구한다. % 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 단계: HA 환경 재구성 후 HA모드 구동	마스터 및 슬레이브 서버에서 CUBRID 환경 설정 파일(cubrid.conf) 및 HA 환경 설정 파일(cubrid_ha.conf)을 설정한다. (관련 매뉴얼 참고) % 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 마스터 및 슬레이브 서버에서 HA모드로 DB를 구동한다. (관련 매뉴얼 참고) % cubrid heartbeat start
H10 단계: 브로커 서버에 CUBRID 2008 R4.3 설치 및 브로커 구동	설치 방법은 본 문서의 CUBRID 2008 R4.3의 설치 방법고한다. 브로커 서버에 있는 브로커를 시작한다. (관련 매뉴얼 참고) % cubrid broker start

2008 R2.0 또는 2008 R2.1에서 2008 R4.3으로 HA 마이그레이션

CUBRID 2008 R2.0 또는 2008 R2.1의 HA 기능을 사용하는 경우, 서버 버전 업그레이드, DB 마이그레이션을 수행하고 HA 환경을 새롭게 구축한 후 해당 버전에서 사용되었던 Linux Heartbeat 자동 시작 설정을 변경해야 한다. (Linux Heartbeat 패키지가 불필요한 경우 삭제한다.)

위의 H1~H10 단계를 수행한 후, 아래의 H11 단계를 수행한다.

단계	설명
H11 단계:	이하의 작업은 마스터 및 슬레이브 서버에서 root 계정으로 수

단계	설명
기존 Linux heartbeat 자동 시작 설정 변경	행한다. [root@master ~]# chkconfig --del heartbeat // 슬레이브 서버에서 동일 작업 수행

복제 기능 사용 시 주의 사항

CUBRID 2008 R4.1 버전부터 복제 기능이 제거되었으므로, 기존의 복제 기능을 사용 중인 환경에서 이중화 환경을 구축하려면 HA 기능을 사용해야 한다. 서버 버전 업그레이드 및 DB 마이그레이션을 수행한 후, HA 환경을 새롭게 구축할 수 있다. HA 환경 구축과 관련하여, 매뉴얼의 '관리자 안내서 > CUBRID HA'를 참고한다.

4.2 기능 추가

Sharding

CUBRIDSUS-4996 데이터베이스 sharding을 위한 기능 추가

다수의 장비로 횡적 분할된 데이터베이스 환경을 용이하게 접근하기 위한 CUBRID SHARD 기능을 제공한다. CUBRID SHARD는 다음과 같은 특징을 갖는다.

- 기존 응용의 변경을 최소화하기 위한 미들웨어 형태로서, 흔히 사용되는 JDBC와 CUBRID C API인 CCI 인터페이스를 이용하여 sharding된 데이터베이스를 투명하게 접근할 수 있다.
- 힌트를 이용하여 실제 질의 수행할 shard를 선택하는 방식으로 기존 사용하던 질의에 힌트를 추가하여 사용할 수 있다.
- CUBRID 이외에 MySQL을 backend shard DB로 하여 구성될 수 있다.

SQL

CUBRIDSUS-8230 INET_ATON 함수, INET_NTOA 함수 추가

INET_ATON 함수, INET_NTOA 함수를 추가했다. INET_ATON 함수는 IPv4 주소를 입력하면 숫자 값을 반환하며, INET_NTOA 함수는 숫자를 입력하면 IPv4 주소 값을 문자열 형태로 반환한다.

```
SELECT INET_ATON('192.168.0.10');
inet_aton('192.168.0.10')
============================
3232235530

SELECT INET_NTOA(3232235530);
inet_ntoa(3232235530)
=====================
'192.168.0.10'
```

드라이버

CUBRIDSUS-8675 CCI, JDBC 응용 프로그램의 로드 밸런싱 기능 추가

CCI, JDBC의 연결 URL에 althosts를 포함하는 경우 응용 프로그램이 메인 호스트와 althosts에 지정한 호스트들에 임의의 순서로 연결하게 하는 기능을 추가했다. 아래 연결 URL의 예에서 loadBalance의 값을 true로 설정하는 경우 해당 기능이 동작된다.

```
jdbc:cubrid:host1:port1:demodb:::?althosts=host2:port2,host3:port3&loadBalance=true
```

CUBRIDSUS-9269 CCI 응용 프로그램에서 resultset을 닫는 함수 추가

CCI 드라이버는 resultset과 statement를 각각 닫는 메소드를 제공하는 JDBC 드라이버와 달리 이 둘을 모두 닫는 cci_close_req_handle 함수만 존재했으나 resultset을 닫는 cci_close_query_result 함수를 추가했다. 새로운 함수를 호출하지 않으면 statement를 닫을 때까지 resultset의 메모리를 유지하므로 메모리 사용량이 증가하는 현상이 발생할 수 있다.

수정 이후 버전에서도 cci_close_query_result 함수 호출 없이 cci_close_req_handle 함수를 호출하면 이전 버전과 마찬가지로 resultset과 statement를 모두 닫는다.

CUBRIDSUS-8940 이스케이프 문자열로 변환하는 cci_escape_string 함수 추가

CUBRID 질의문에서 사용할 수 있는 이스케이프 문자열로 변환해주는 cci_escape_string 함수를 추가했다.

모니터링

CUBRIDSUS-9501 트랜잭션의 질의 수행 상태를 출력하는 기능 추가

cubrid killtran에 트랜잭션의 질의 수행 상태를 출력하는 -q(--query-exec-info) 옵션을 추가했다.

```
% cubrid killtran -q testdb

Tran index  Process id  Program name  Query time   Tran timeWait for lock holder      SQL Text
------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------
      1(+)     22982   b1_cub_cas_1        0.00       0.00                    -1         ***
empty ***
      2(+)     22983   b1_cub_cas_2        1.80       1.80                     1         update
[ta] [ta] set [a]=5 wh
------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------
```

질의 수행 상태에는 다음 정보를 포함한다.

- Tran index: 트랜잭션 인덱스
- Process id: 클라이언트 프로세스 ID
- Program name: 클라이언트 프로그램 이름
- Query time: 수행중인 질의의 총 수행 시간(단위: 초)
- Tran time: 현재 트랜잭션의 총 수행 시간(단위: 초)
- Wait for lock holder: 현재 트랜잭션이 잠금(lock) 대기중이면 해당 잠금을 소유하고 있는 트랜잭션의 리스트
- SQL Text: 수행중인 질의문(최대 30자)

CUBRIDSUS-6987 지정한 시간을 초과하는 질의문과 질의 실행 계획 정보를 로그에 기록하게 하는 기능 추가

시스템 파라미터 sql_trace_slow_msec에 의해 지정한 시간을 초과하는 질의문의 질의 실행 계획 정보를 로그에 기록하게 하는 기능을 추가했다. 시스템 파라미터 sql_trace_execution_plan의 값이 yes이면 해당 SQL 문과 함께 질의 실행 계획, cubrid statdump 정보를 각각 서버 에러 로그 파일, 브로커 응용 서버(CAS) 로그 파일에 기록하며, cubrid plandump를 실행하면 해당 SQL 문과 질의 실행 계획을 출력한다.

단, 서버 에러 로그 파일에는 error_log_level 파라미터의 값이 NOTIFICATION인 경우에만 해당 정보를 기록한다.

CUBRIDSUS-6377 디버깅용 로그를 기록하도록 설정하는 연결 URL 속성 추가

CCI 연결 URL에 디버깅용 로그 기록을 설정하는 기능을 추가했다. logSlowQueries와 slowQueryThresholdMillis는 슬로우 쿼리의 로그 기록을, logTraceApi는 CCI 함수가 호출될 때 각 함수의 시작과 끝을, logTraceNetwork은 CCI 함수의 Network 데이터 전송 내용을 로그 파일에 기록한다.

```
url =
"cci:cubrid:localhost:33000:demodb:::?logSlowQueries=true&slowQueryThresholdMillis=1000&logTrac
eApi=true&logTraceNetwork=true"
```

아울러, CCI 연결 URL에 디버깅용 로그 파일의 경로를 지정하는 logBaseDir의 동작 방식을 수정했다. 이전 버전에서는 logBaseDir 값과 logFile이 같이 있으면 logBaseDir 프로퍼티를 무시했으나 수정 이후 경로를 포함한 파일 이름을 "logBaseDir/logFile"로 지정하도록 바뀌었다.

설정

CUBRIDSUS-9308 데이터베이스 서버 프로세스와 이에 접속하는 브로커 응용 서버 등의 클라이언트 프로세스가 정상 동작하는지 확인하는 과정의 수행 여부를 설정

데이터베이스 서버 프로세스(cub_server)와 이에 접속하는 클라이언트 프로세스가 정상 동작하는지 서로 확인하는 과정의 수행 여부를 설정할 수 있도록 check_peer_alive 시스템 파라미터를 추가했다. 클라이언트 프로세스에는 브로커 응용 서버(cub_cas) 프로세스, 복제 로그 반영 프로세스(copylogdb), 복제 로그 복사 프로세스(applylogdb), CSQL 인터프리터(csql) 등이 있다.

서버 프로세스와 클라이언트 프로세스는 접속이 이루어진 후 네트워크를 통해 데이터를 기다리는 중 오랫동안(예: 5초 이상) 응답을 받지 못하면 설정에 따라 상대방이 정상 동작하는지 확인하는 과정을 거친다. 서로 확인하는 과정에서 정상 동작하지 않는다고 판단되면 연결된 접속을 강제 종료한다.

ECHO(7) 포트가 방화벽(firewall) 설정으로 막혀있으면 서버 프로세스 또는 클라이언트 프로세스가 각각 서로의 상태를 확인할 때 상대방 프로세스가 종료된 것으로 오인할 수 있으므로, 이 파라미터를 none으로 설정하여 이 문제를 회피해야 한다.

4.3 변경된 동작

드라이버

CUBRIDSUS-9364 CCI 응용 프로그램에서 여러 개의 질의를 한 번에 수행한 결과의 배열에 대한 에러 처리 방식 수정

CCI 응용에서 여러 개의 질의를 한 번에 수행할 때 2008 R3.0부터 2008 R4.1 버전까지는 cci_execute_array 함수, cci_execute_batch 함수 또는 cci_execute_result 함수에 의한 질의 수행 결과들 중 하나만 에러가 발생해도 해당 질의의 에러 코드를 반환했으나, 2008 R4.3 버전부터는 전체 질의 개수를 반환하고 CCI_QUERY_RESULT_* 매크

로들을 통해 개별 질의에 대한 에러를 확인할 수 있도록 수정했다.

이와 관련하여 전체 질의 결과에서 실패한 특정 질의의 에러 번호를 확인할 수 있도록 CCI_QUERY_RESULT_ERR_NO 매크로를, 실패한 위치가 CAS인지 DBMS인지 알 수 있도록 CCI_QUERY_RESULT_RESULT 매크로의 반환 값에 에러 인식자(CAS 에러 -1, DBMS 에러 -2)를 추가했다.

보다 자세한 사항은 신규 주의 사항을 참고한다.

4.4 개선 및 오류 수정

성능 및 리소스

CUBRIDSUS-8114 HA 복제 성능 개선

HA 복제 성능을 개선하여 슬레이브 노드에서 처리량 기준으로 INSERT 성능이 약 2 배, UPDATE 성능이 약 1.4 배, DELETE 성능이 약 1.7 배 향상되었다.

CUBRIDSUS-7466 커버링 인덱스 스캔 성능 개선

커버링 인덱스 스캔 시 사용자의 인터럽트를 매번 체크하는 불필요한 루틴을 제거하여 성능을 개선했다.

CUBRIDSUS-7418 ORDER BY 절의 칼럼을 타입 변환해도 결과 순서에 영향이 없으면 ORDER BY 절 최적화를 적용하도록 개선

ORDER BY 절의 칼럼을 타입 변환해도 타입 변환 이전과 이후의 결과 순서가 동일한 경우, 정렬 작업을 수행하지 않고 인덱스에 의해 정렬된 값의 순서대로 가져오는 ORDER BY 절 최적화(skip order by)를 적용하도록 개선했다.

CUBRIDSUS-7909 INSERT와 DELETE 반복 시 수행 시간이 점차 느려지는 현상 개선

INSERT와 DELETE를 반복 수행하면 수행 시간이 점차 느려지는 현상을 개선했다.

CUBRIDSUS-7158 페이지 크기를 넘는 레코드들에 대해 UNION 수행 결과 건수가 1건인 경우 메모리 누수 현상 수정

페이지 크기를 넘는 오버플로우 레코드들에 대해 UNION 질의 수행 결과 건수가 1건인 경우 발생하는 메모리 누수(memory leak) 현상을 수정했다. UNION ALL 질의는 메모리 누수 현상이 발생하지 않았다.

CUBRIDSUS-8140 REUSE_OID 옵션으로 생성한 테이블을 DROP하는 경우 공간이 반환되지 않을 수 있는 현상 수정

REUSE_OID 옵션을 가지는 테이블을 DROP하는 도중 서버에서 내부적으로 오류가 발생하는 경우에 해당 공간이 반환되지 못하는 문제를 수정했다.

CUBRIDSUS-8287 INSERT ON DUPLICATE KEY UPDATE 문의 수행 성능 개선

INSERT ON DUPLICATE KEY UPDATE 문의 대상 테이블에 기본 키를 포함하여 고유 키가 두 개 이상 존재하는 경우의 수행 성능을 개선했다.

```
CREATE TABLE x (a INT PRIMARY KEY, b INT, c INT, d INT, UNIQUE(b), UNIQUE(c));
CREATE SERIAL s;
INSERT INTO x VALUES (s.NEXT_VALUE, 0, 0, 0) ON DUPLICATE KEY UPDATE d = d+1;
```

CUBRIDSUS-7661 LIMIT 절 질의 최적화 개선

LIMIT N으로 질의 수행 시 N+1번째 레코드까지 탐색한 후 질의가 종료되는 비효율을 개선하였다.

```
SELECT * FROM t1 WHERE a > 0 AND b = 1 LIMIT 3;
```

을 실행하면 기존에는 조건을 만족하는 4번째 레코드까지 탐색한 후 앞의 세 레코드를 리턴했으나 수정 후에는 세번째 레코드에서 바로 리턴된다.

CUBRIDSUS-9521 데이터베이스 볼륨 추가로 인한 시스템 운영 영향을 줄이기 위해 디스크 출력량을 제한하는 기능 추가

데이터베이스 볼륨 추가로 인한 시스템 운영 영향을 줄이기 위해 디스크 출력량을 제한하는 기능을 추가했다. 이 기능은 addvoldb 명령에 --max_writesize-in-sec 옵션을 사용하여 1초당 쓸 수 있는 최대 크기를 지정한다.

```
% cubrid addvoldb -C --db-volume-size=2G --max-writesize-in-sec=1M testdb
```

CUBRIDSUS-8256 같은 칼럼에 두 개 이상의 외래 키를 정의하고 테이블을 DROP한 이후 일부 공간이 재사용되지 못하는 오류 수정

하나의 칼럼에 이름만 다른 외래 키를 두 개 이상 정의하고 테이블을 DROP하면 해당 테이블이 사용하던 일부 공간이 재사용되지 못하는 오류를 수정했다.

```
CREATE TABLE foo (a INT, PRIMARY KEY (a));
CREATE TABLE bar (a INT,
                  CONSTRAINT con1 FOREIGN KEY(a) REFERENCES foo (a),
                  CONSTRAINT con2 FOREIGN KEY(a) REFERENCES foo (a));
```

```
-- INSERT records

DROP TABLE bar;
DROP TABLE foo;
```

CUBRIDSUS-7464 DATE_ADD 함수의 성능 개선

DATE_ADD 함수 내에서 불필요하게 호출되는 루틴을 제거하여 성능을 개선했다.

CUBRIDSUS-7288 테이블 DROP과 CREATE를 반복할수록 CREATE 시간이 오래 걸리는 문제 수정

테이블 DROP과 CREATE를 반복할수록 CREATE 시간이 오래 걸리는 문제를 수정했다. 참고로 수정 이전 버전에서도 테이블 생성 시 REUSE_OID 옵션을 지정하면 이러한 현상이 발생하지 않았다.

```
CREATE TABLE reuse_tbl (a INT PRIMARY KEY) REUSE_OID;
```

SQL

CUBRIDSUS-7610 DATE_ADD 함수의 INTERVAL 값을 호스트 변수로 사용하면 질의 수행 시 서버가 비정상 종료되는 현상 등 수정

질의를 PREPARE할 때 DATE_ADD 함수의 INTERVAL 값을 호스트 변수로 사용하면, EXECUTE할 때 서버가 비정상 종료되는 현상을 수정했다. 이와 함께 DATE_ADD 함수의 INTERVAL 단위에 따라 입력 값으로 INTERGER만 가능했던 동작을 VARCHAR도 가능하게 수정했다.

```
PREPARE s FROM 'SELECT DATE_ADD(?, INTERVAL ? YEAR_MONTH)';
EXECUTE s USING '2010-01-01', 1;
EXECUTE s USING '2010-01-01', '1-1';
```

CUBRIDSUS-6520 CLOB_TO_CHAR 함수 수행 시 오류 수정

CLOB 컬럼이 있는 테이블에 고유 인덱스를 스캔하는 REPLACE 문을 수행하고 커밋한 후에 CLOB_TO_CHAR 함수를 수행하면 "ERROR: External file ""/home/CUBRID/databases/testdb/lob/ces_153/tbl.000013252320301 80575_3434"" was not found." 오류 메시지와 함께 질의 수행에 실패하는 현상을 수정했다.

```
SELECT id, CLOB_TO_CHAR(text) FROM tbl ORDER BY id;
```

CUBRIDSUS-9553 STR_TO_DATE 함수가 밀리초를 항상 0으로 리턴하는 문제 수정

STR_TO_DATE 함수가 밀리초를 항상 0으로 리턴하는 문제를 수정했다.

```
SELECT STR_TO_DATE('2012-10-31 23:49:29.123', '%Y-%m-%d %H:%i:%s.%f');
```

CUBRIDSUS-7700 IN, NOT IN 조건의 스칼라 부질의에 ORDER BY 절이 사용되거나 SELECT LIST의 컬럼 개수가 여러 개인 경우 잘못된 질의 결과를 출력하는 현상 수정

IN, NOT IN 조건의 스칼라 부질의(scalar subquery)에 ORDER BY 절이 사용되거나 SELECT 리스트의 컬럼 개수가 여러 개인 경우 잘못된 질의 결과를 출력하는 현상을 수정했다.

```
// 수정 이전 버전에서 스칼라 부질의에 ORDER BY 절이 사용되면 질의 결과가 항상 0건이 되었다.
SELECT * FROM tbl WHERE col IN (SELECT col FROM tbl2 ORDER BY b);

// 수정 이전 버전에서 스칼라 부질의의 SELECT LIST에 컬럼이 두 개 이상 사용되면 문법(semantic) 오류가 발생해야 하나, 질의 결과가 0건이 되었다.
SELECT * FROM tbl WHERE col IN (SELECT a, b FROM tbl2);
SELECT * FROM tbl WHERE col NOT IN (select a,b from tbl2);
```

CUBRIDSUS-7195 WHERE 절에 "IN NULL" 조건 수행 시 오류가 발생하는 문제 수정

WHERE 절에 "IN NULL" 조건 수행 시 "ERROR: ' in ' operator is not defined on types varchar and null" 오류가 발생하는 문제를 수정했다. 아래 예와 같이 IN 조건의 결과가 NULL인 경우 수정 이전 버전에서 오류가 발생했다.

```
SELECT * FROM tab0 WHERE col2 IN (SELECT col2 FROM tab0 WHERE 1=0);
```

CUBRIDSUS-9193 AND와 OR 조건이 여러 개 결합된 질의 수행 시 조건 일부가 누락되어 잘못된 결과를 출력할 수 있는 문제 수정

AND와 OR 조건이 여러 개 결합된 질의 수행 시 조건 일부가 누락된 결과를 출력할 수 있는 문제를 수정했다.

```
SELECT *
FROM it, p
WHERE
(
  p_pkey = l_pkey
  and p_br = 'Br12'
  and p_ct in ('CS', 'BX', 'PK', 'PG')
  and l_qty >= 1 and l_qty <= 1 + 10
  and p_sz between 1 and 5
```

```
  and l_sm in ('A', 'RG')
  and l_st = 'DIP'
)
OR
(
  p_pkey = l_pkey
  and p_br = 'Br12'
  and p_ct in ('MG', 'MB', 'MPK', 'MPC')
  and l_qty >= 10 and l_qty <= 10 + 10
  and p_sz between 1 and 10
  and l_sm in ('A', 'RG')
  and l_st = 'DIP'
)
OR
(
  p_pkey = l_pkey
  and p_br = 'Br12'
  and p_ct in ('LG', 'LB', 'LPK', 'LPC')
  and l_qty >= 20 and l_qty <= 20 + 10
  and p_sz between 1 and 15
  and l_sm in ('A', 'RG')
  and l_st = 'DIP'
);
```

CUBRIDSUS-8338 INSERT ... SELECT 구문 수행 시 UNIQUE 칼럼의 값으로 NULL이 입력되는 경우 COUNT(*) 질의의 결과가 잘못 출력되는 현상

INSERT ... SELECT 구문 수행시 UNIQUE 칼럼의 값으로 NULL이 입력되는 경우 COUNT(*) 질의의 결과가 잘못되는 현상을 수정했다. COUNT(칼럼 이름)과 같이 칼럼 이름을 명시하여 COUNT를 수행하는 경우에는 이전 버전에서도 정상 동작했다.

```
CREATE TABLE t1(id INT AUTO_INCREMENT, mgrid INT UNIQUE, dummy INT);
INSERT INTO t1(dummy) VALUES (1);
INSERT INTO t1(dummy) SELECT dummy FROM t1;
INSERT INTO t1(dummy) SELECT dummy FROM t1;

SELECT COUNT(*) FROM t1;
```

CUBRIDSUS-9331 ORDER BY절을 포함한 뷰를 SELECT할 때 서버가 비정상 종료하는 현상 수정

ORDER BY 절을 포함한 뷰를 SELECT할 때 SELECT 리스트에 *을 사용하는 경우를 제외하고는 서버가 비정상 종료하는 현상을 수정했다.

```
CREATE VIEW au AS
SELECT
    tbla.a_id AS a_id,
    tbla.u_id AS u_id,
    tbla.a_date AS a_date,
    tblu.u_name AS u_name,
FROM
    tbla LEFT JOIN tblu ON tbla.u_id = tblu.u_id
ORDER BY tbla.a_date ASC;

SELECT u_name FROM au;
```

CUBRIDSUS-9336 DESC 문과 SHOW 문에서 테이블, 뷰 이름의 대소문자를 구분하지 않도록 수정

DESC 문과 SHOW TABLES, SHOW CREATE VIEW, SHOW INDEX 문에서 입력으로 주어지는 테이블, 뷰 이름의 대소문자를 구분하는 문제를 수정했다.

CUBRIDSUS-9049 뷰 재생성한 이후 해당 뷰의 UPDATE 수행에 실패하는 현상 수정

뷰를 재생성한 이후 UPDATE할 때 이전과 같은 UPDATE 문을 수행했을 경우, 이전 질의 계획을 사용하여 DROP했던 뷰를 참조하면서 UPDATE에 실패하는 현상을 수정했다.

```
CREATE TABLE foo(a int);
INSERT INTO foo VALUES (3);

CREATE VIEW v AS SELECT * FROM foo WHERE a < 2;
UPDATE v SET a=3;
DROP VIEW v;

CREATE VIEW v AS SELECT * FROM foo WHERE a < 2;
UPDATE v SET a=3;
```

CUBRIDSUS-7549,7669네트워크 장애나 HA 절체 또는 데이터베이스 서버의 재시작 이후 LAST_INSERT_ID와 같이 세션 데이터를 사용하는 작업 요청이 정상 처리되지 않을 수 있는 현상 수정

HA 절체(failover) 또는 데이터베이스 서버가 재시작되면 서로 다른 응용 클라이언트들이 같은 세션 ID를 공유하는 경우가 발생할 수 있었다. 이 경우 세션을 공유하는 응용 클라이언트 중 하나가 종료하면 다른 응용 클라이언트에서 세션 데이터에 의해 관리되는 사용자 정의 변수, PREPARE 문, LAST_INSERT_ID, ROW_COUNT 등의 요청이 정상 처리되지 않는 현상을 수정했다.

CUBRIDSUS-7916 IN 절의 호스트 변수 타입이 날짜/시간인 경우 정상 수행되지 않는 현상 수정

IN 절의 호스트 변수 타입이 날짜/시간인 경우 정상 수행되지 않는 현상을 수정했다. 수정 이전 버전에서 바인딩할 값의 타입이 날짜/시간 타입인 경우 아래의 Q1은 정상 동작했으나, Q2는 데이터가 삭제되지 않았다.

```
DELETE FROM tbl WHERE d = ?;     -- Q1
DELETE FROM tbl WHERE d IN (?);   -- Q2
```

CUBRIDSUS-7246 트리거 내에서 CLOB_TO_CHAR 함수를 호출하면 NULL을 반환하는 오류 수정

트리거 내에서 CLOB_TO_CHAR 함수를 호출하면 NULL을 반환하는 오류를 수정했다.

CUBRIDSUS-7420 LIMIT 0인 경우 곧바로 결과를 반환하도록 수정

LIMIT 0을 포함한 질의는 곧바로 결과를 반환하도록 수정했다. 이전 버전에서는 질의를 수행한 후에 LIMIT 절을 평가하므로 경우에 따라서 질의 처리에 많은 시간이 소요되었다.

```
SELECT CAST(dt_col AS DATE)
FROM article
WHERE id = '001' AND dt_col < TO_DATE('20120201', 'YYYYMMDD')
ORDER BY CAST(dt_col AS DATE) DESC
LIMIT 0;
```

CUBRIDSUS-7521 일부 SQL 함수에서 모든 인자가 호스트 변수일 때 값을 바인딩하면서 질의 수행에 실패하는 현상 수정

NULLIF, LEAST, GREATEST 등 일부 SQL 함수에서 모든 인자가 호스트 변수일 때 값을 바인딩하면서 DOUBLE로 타입 변환을 시도하게 되어 질의 수행에 실패하는 현상을 수정했다.

```
preStmt = conn.prepareStatement("select nullif (?, ?)");
preStmt.setString(1, "A");
preStmt.setString(2, "a");
rs = preStmt.executeQuery();
```

CUBRIDSUS-7377 호스트 변수에 값을 바인딩할 때 두 번째 바인딩하는 값의 타입이 첫 번째와 다르면 오류가 발생하는 문제 수정

호스트 변수에 값을 바인딩할 때 두 번째 바인딩하는 값의 타입이 첫 번째 타입과 다르면 첫 번째 타입으로 바인딩하려고 시도하면서 오류가 발생될 수 있는 문제를 수정했다. 예를 들어 'SELECT ?' 질의에 처음에는 1을 바인딩하고, 이후에 'A'를 바인딩하면 'A'를 INTEGER 타입으로 바인딩하려고 시도하면서 오류가 발생했다.

CUBRIDSUS-6250 사용하지 않는 예약어 제거

ALIAS, TYPE, VIRTUAL, TEST, WAIT 등 사용하지 않는 예약어(reserved word)는 제거하여, 해당 예약어를 테이블 이름이나 칼럼 이름 등의 식별자로 사용할 수 있게 개선했다.

식별자로 사용이 가능해진 기존 예약어들은 다음과 같다.

기존 예약어 목록				
ALIAS	ASYNC	CLUSTER	COMPLETION	DICTIONARY
EXCLUDE	LDB	OID	OPERATION	OPERATORS
OTHERS	PENDANT	PREORDER	PRIVATE	PROTECTED
PROXY	REGISTER	STRUCTURE	SYS_USER	TEST
THERE	TYPE	VIRTUAL	VISIBLE	WAIT

CUBRIDSUS-7389 다른 트랜잭션에 의해 삭제된 테이블에 대한 SELECT 질의가 오류를 출력하지 않고 0건을 반환하는 문제 수정

다른 트랜잭션이 삭제한 테이블에 대해 SELECT 질의를 수행하면 해당 테이블을 찾을 수 없다는 오류를 출력해야 하지만 0건을 반환하는 문제를 수정했다.

CUBRIDSUS-7378 클릭 카운터로 인해 교착 상태가 발생하면 서버 프로세스가 동작을 멈출(hang) 수 있는 문제 수정

클릭 카운터로 인해 교착 상태가 발생하면 가끔 서버 프로세스가 이를 감지하지 못하고 동작을 멈출(hang) 수 있는 문제를 수정했다. 클릭 카운터로 인해 교착 상태가 발생하면 클릭 카운터의 업데이트만 무시되고, 나머지 질의들은 정상적으로 진행되어야 한다.

CUBRIDSUS-8838 다수의 트랜잭션이 오버 플로우 키가 존재하는 인덱스에 대한 입력/삭제를 동시에 수행하는 도중, 한 트랜잭션의 롤백으로 인해 다른 트랜잭션이 실패하는 오류

오버 플로우 키가 존재하는 인덱스에 여러 트랜잭션이 동시에 입력/삭제를 수행하는 환경에서, 한 트랜잭션이 롤백되면 다른 트랜잭션들에서 "Query execution failure" 에러가 발생할 수 있는 현상을 수정했다.

CUBRIDSUS-7239 트리거에 의한 UPDATE/INSERT 질의가 실패했음에도 불구하고 트리거를 유발한 질의가 롤백되지 않는 문제 수정

자동 커밋 모드가 OFF일 때 트리거에 의한 UPDATE/INSERT 질의가 실패했음에도 불구하고, 트리거를 유발한 질의가 롤백되지 않는 문제를 수정했다. 아래 예에서, Q6을 수행하면 t2 테이블의 trig1 트리거가 수행되는데, 이때 t1 테이블에 int 타입의 값을 입력할 수 없으므로 에러가 발생하고 이 트리거를 유발한 Q6번은 롤백된 상태여야 한다. 즉, Q7 수행 이후에 t1과 t2의 레코드 건수는 각각 1건, 0건이어야 한다. 이전 버전에서는 에러가 발생하면 trig1 트리거를 유발한 Q6 질의가 롤백되지 않는 문제가 존재했다.

```
;autocommit off

CREATE TABLE t1(col1 date);  -- Q1
CREATE TABLE t2(col2 int);  -- Q2
CREATE TRIGGER trig1 AFTER INSERT ON t2 EXECUTE INSERT INTO t1(col1) VALUES(obj.col2);   -- Q3
COMMIT;   -- Q4

INSERT INTO t1(col1) VALUES ('2012-04-30');  -- Q5
INSERT INTO t2(col2) VALUES (1);  -- Q6

ERROR: Error evaluating action for ""trig1"", Execute: Cannot coerce obj.cold2 to type date.

COMMIT;  -- Q7
```

CUBRIDSUS-6667 두 개 이상의 트랜잭션이 동시에 일시적 임시 볼륨을 생성하거나 확장하려고 할 때 latch timeout 오류가 발생하는 문제 수정

두 개 이상의 트랜잭션이 동시에 일시적 임시 볼륨(temporary temp volume)을 생성하거나 확장하려고 할 때 "LATCH ON PAGE(12345|0) TIMEDOUT" 오류 메시지와 함께 질의 수행에 실패하는 문제를 수정했다.

CUBRIDSUS-7606 자동 커밋 모드에서 다중 질의를 한 번에 수행하면 자동 커밋되지 않는 문제 수정

자동 커밋 모드에서 다중 질의를 한 번에 수행하면, 예를 들어 "CREATE TABLE a(col int);INSERT INTO a VALUES (1);"와 같이 한 번에 여러 개의 질의를 수행하면 자동 커밋되지 않는 문제를 수정했다.

질의 계획 및 최적화

CUBRIDSUS-7818 질의 계획 정보 출력 이후 질의 수행 시 잘못된 결과를 출력할 수 있는 문제 수정

;plan simple 명령어를 실행하여 질의 계획을 출력하면 입력 값에 대응되는 칼럼의 타입으로 입력 값이 변환되어 잘못된 결과를 출력할 수 있는 문제를 수정했다.

아래의 예제에 대해서 수정 이전 버전에서는 계획 정보를 출력하면서 2.3이라는 입력 값이 2로 변환되어 서버에 전달되어 결국 2가 결과에 포함되지 못하는 문제가 있었다..

```
CREATE TABLE foo (col INT);
INSERT INTO foo VALUES (1),(2);
csql>;plan simple
SELECT * FROM foo WHERE col < 2.3;
```

CUBRIDSUS-7637 질의 계획 캐시를 사용하지 않는 질의에서 테이블 이름을 변경한 후 질의 수행 시 이전 테이블에 대해서 질의가 수행되는 문제 수정

테이블 이름을 변경한 후 질의 수행 시 해당 이름을 가지는 새 테이블이 아니라 변경된 이전 테이블에 대하여 질의가 수행되는 문제를 수정했다. 이전 버전에서는 INSERT문은 질의 계획 캐시를 사용하지 않으므로 항상 문제가 발생했으며, 나머지 질의의 경우 시스템 파라미터 max_plan_cache_entries의 값을 -1로 설정하여 질의 계획 캐시 기능을 끄는 경우에 문제가 발생했다.

```
// insert의 예
INSERT INTO tbl VALUES (...);
RENAME TABLE tbl AS tbl_old;
RENAME TABLE tbl2 AS tbl;
// 아래 질의 수행 시 새로운 테이블이 아닌 tbl_old에 값이 INSERT됨.
INSERT INTO tbl VALUES (...);
```

CUBRIDSUS-6492 LIKE 절에 상수가 있는 질의문이 플랜 캐시에 존재하는 경우 오류가 발생하는 문제 수정

LIKE 절에 상수가 있는 질의문을 가지고 PREPARE 수행 시 질의문이 플랜 캐시에 존재하면 "ERROR: Invalid XASL tree node content." 오류가 발생하는 문제를 수정했다.

```
CREATE TABLE t1 (a char (10) , unique(a));
INSERT INTO t1 VALUES ('a');
UPDATE t1 SET a='a ' WHERE a LIKE 'a ';
UPDATE t1 SET a='a ' WHERE a LIKE 'a ';
```

인덱스

CUBRIDSUS-7868 특정 OUTER JOIN 질의에 대해서 커버링 인덱스 스캔 질의 실행 시에 발생할 수 있는 오류 수정

OUTER JOIN을 포함한 특정 질의에 대해서 커버링 인덱스 스캔 질의 실행 계획이 잘못되어 실행 과정에서 "Query execution failure #10946." 오류가 발생하는 문제를 수정했다.

CUBRIDSUS-9314 다중 컬럼 인덱스에 DESC 컬럼이 존재하고 OR 조건이 있을 때 조건에 따라 질의 결과가 잘못되는 문제 수정

다중 컬럼 인덱스에 DESC 컬럼이 존재하고 인덱스 전체 키가 아닌 부분 키에 대한 OR 조건들이 주어지면, 질의 결과가 잘못되는 문제를 수정했다.

```
CREATE TABLE foo(col1 INT, col2 INT, col3 INT);
CREATE INDEX idx_foo ON foo(col1,col2 DESC, col3);
INSERT INTO foo VALUES(1,10,100);
INSERT INTO foo VALUES (1,11,100);
PREPARE s FROM 'SELECT col1,col2 FROM foo WHERE col1=? AND ((col2=? ANDcol3<?) OR col2>?);';
EXECUTE s USING 1, 10, 100, 10;
```

CUBRIDSUS-7988 인덱스 생성 후 데이터를 삭제한 테이블에서 MAX 함수나 ORDER BY DESC 를 사용한 질의 수행 시 발생하는 오류 수정

테이블에 데이터가 존재하는 상태에서 인덱스를 생성하고 데이터를 모두 삭제한 이후, 이 테이블에 MAX 함수나 ORDER BY DESC 절을 사용하는 등 인덱스의 마지막 리프 페이지를 찾는 질의를 수행하면 "ERROR: An I/O error occurred while reading page 65536 of volume (null).... Bad file descriptor" 오류가 발생하는 현상을 수정했다.

```
CREATE TABLE tb2 (col1 INT PRIMARY KEY, col2 VARCHAR(16));
INSERT INTO tb2 VALUES (1, '1');
CREATE INDEX i_tb2 ON tb2(col1, col2);
DELETE FROM tb2;
SELECT * FROM tb2 ORDER BY 1 DESC,2 DESC;
```

CUBRIDSUS-7585 검색 조건에 클래스 OID로 검색하는 조건이 있고 그 외 나머지 조건들은 모두 커버링 인덱스 조건을 만족할 때 서버가 비정상 종료되는 오류 수정

검색 조건에 클래스 OID로 검색하는 조건이 있고 그 외 나머지 조건들은 모두 커버링 인덱스 조건(인덱스에 SELECT 리스트의 컬럼과 WHERE 조건의 컬럼을 모두 포함)을 만족할 때 질의 수행 과정에서 서버가 비정상 종료되는 오류를 수정했다.

```
CREATE TABLE a (class_of object, class_name varchar(200));
CREATE TABLE c (class_name varchar(200), i int);
CREATE INDEX ON c (class_name);
INSERT INTO a VALUES (INSERT INTO c VALUES('aa', 1), 'aa');

SELECT c.class_name
FROM a, c
WHERE a.class_of=c AND c.class_name=a.class_name;
```

HA 기능

CUBRIDSUS-9525 복제 지연 정보를 출력하도록 개선

HA 환경에서 applylogdb 명령으로 트랜잭션 로그 복사와 트랜잭션 로그 반영 정보 출력 시 복제 지연 정보를 출력하도록 개선했다. 다음은 복제 지연 정보를 출력하는 예이다.

```
% cubrid applyinfo -L /home/cubrid/DB/testdb_nodeA -r nodeA -a -i 3 testdb

...

*** Delay in Copying Active Log ***
Delayed log page count         : 4
Estimated Delay                : 0 second(s)

 *** Delay in Applying Copied Log ***
Delayed log page count         : 1459
Estimated Delay                : 22 second(s)
```

CUBRIDSUS-8673 DROP SERIAL 문이 슬레이브 노드에 반영되지 않는 현상

HA 환경에서 DROP SERIAL 문이 슬레이브 노드에 반영되지 않는 현상을 수정했다.

CUBRIDSUS-9380 슬레이브 노드를 maintenance 모드로 변경하면 마스터 노드의 복제 로그 반영 프로세스가 재시작을 반복하는 문제 수정

HA 환경에서 슬레이브 노드를 maintenance 모드로 변경하면 마스터 노드의 복제 로그 반영 프로세스가 반복적으로 재시작하는 문제를 수정했다.

CUBRIDSUS-8885 마스터 노드에서 AUTO_INCREMENT 칼럼이 있는 테이블의 DROP 이후에도 슬레이브 노드에 AUTO_INCREMENT에 의한 시리얼이 남아있는 문제 수정

HA 환경의 마스터 노드에서 AUTO_INCREMENT 칼럼이 있는 테이블의 DROP 이후에도 슬레이브 노드에 AUTO_INCREMENT에 의한 시리얼이 남아있는 오류로 인해, 마스터 노드에서 같은 테이블 이름과 칼럼 이름으로 AUTO_INCREMENT를 재생성하면 슬레이브 노드에서 테이블 생성이 되지 않는 문제를 수정했다.

```
CREATE TABLE t1(id INT PRIMARY KEY AUTO_INCREMENT, a INT);
DROP TABLE t1;
CREATE TABLE t1(id INT PRIMARY KEY AUTO_INCREMENT, a INT);
```

CUBRIDSUS-8582 자동 커밋 모드가 OFF이고 DROP TABLE IF EXISTS 문 수행 시 슬레이브 노드에 오류가 발생하는 문제 수정

HA 환경에서 자동 커밋 모드가 OFF이고 DROP TABLE IF EXISTS 문을 수행하면 슬레이브 노드의 오류 로그에 "SYNTAX ERROR ... ERROR CODE = -493" 오류 메시지가 발생하는 문제를 수정했다.

```
DROP TABLE IF EXISTS m1;
CREATE TABLE m1(col1 INT,col2 INT PRIMARY KEY);
COMMIT;
```

CUBRIDSUS-8556 레플리카 노드에서는 항상 보관 로그를 삭제하도록 변경

레플리카 노드에서는 보관 로그 삭제를 위해서는 시스템 파라미터 force_remove_log_archives의 설정 값을 항상 yes로 변경해야 했다. 설정을 하지 않았을 경우에 불필요한 보관 로그가 계속 쌓이면서 문제를 야기시킬 수 있었는데, 2008 R4.3 부터 레플리카 노드는 force_remove_log_archives의 설정 값과 상관 없이 항상 보관 로그를 삭제하도록 변경했다.

CUBRIDSUS-8937 정상 상황일 때 슬레이브 노드에 발생 가능한 오류에 대해 오류 수준을 Warning으로 변경

HA 환경에서 슬레이브 노드에 "Internal error: fetching deallocated pageid 0 of volume /CUBRID/databases/testdb_lgat" 오류는 정상 상황일 때 발생할 수 있으므로 오류 수준을 Error에서 Warning으로 변경했다.

CUBRIDSUS-9166 슬레이브 노드의 복제 재구축 후 트랜잭션 복제 로그 반영 프로세스가 비정상 종료될 수 있는 문제 수정

HA 환경에서 슬레이브 노드의 복제 재구축 후 반영해야 할 보관 로그의 로그 페이지를 찾지 못하는 오류로 인해 트랜잭션 복제 로그 반영 프로세스가 비정상 종료될 수 있는 문제를 수정했다.

CUBRIDSUS-7524 shell prompt가 길면 HA 복제 스크립트가 실패하는 문제 수정

슬레이브 노드 재구성 스크립트인 ha_make_slavedb.sh에서 다른 노드의 환경 변수 체크 결과를 전달받는 부분이 실패하는 문제를 수정했다.

CUBRIDSUS-7756 to-be-active 상태의 서버에 클라이언트가 수 차례 접속을 시도한 상태에서 서버의 종료를 시도하면 서버 프로세스가 비정상 종료되는 현상 수정

HA 환경에서 to-be-active 상태의 서버에 csql 등의 클라이언트가 cubrid.conf의 max_clients 개수 이상 접속 시도를 반복한 상태에서 서버의 정상 종료를 시도하면 서버 프로세스가 비정상 종료되는 현상을 수정했다.

CUBRIDSUS-7154 마스터 노드가 슬레이브 노드의 연결을 대기 중인 to-be-active 상태에서 TCP 연결이 끊어지면 오류 처리하도록 수정

HA 환경에서 마스터 노드가 슬레이브 노드의 연결을 대기 중인 to-be-active 상태에서 연결을 요청한 클라이언트의 TCP 연결이 끊어지는 경우에도 마스터 서버 프로세스(cub_master)에 연결 대기 정보가 남아있는 문제가 존재했으나, 이러한 경우 오류 처리하도록 수정했다.

CUBRIDSUS-9649 ALTER ... CHANGE COLUMN 문이 슬레이브 노드에 반영되지 않는 현상

HA 환경에서 ALTER ... CHANGE COLUMN 문이 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
ALTER TABLE t1 CHANGE i2 i0 INTEGER FIRST;
```

CUBRIDSUS-8203 ALTER TABLE ... AUTO_INCREMENT 문이 슬레이브 노드에 반영되지 않는 현상 수정

HA 환경에서 ALTER TABLE ... AUTO_INCREMENT 문이 "log applier: failed to apply schema replication log. class: "t", schema: "'alter class [t] auto_increment = 5'", internal error: -1056." 오류 메시지와 함께 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
ALTER TABLE t AUTO_INCREMENT = 5;
```

CUBRIDSUS-7622 AUTO_INCREMENT 컬럼을 DROP 할 때 슬레이브 노드에 반영되지 않는 현상 수정

HA 환경에서 AUTO_INCREMENT 컬럼을 DROP 할 때 "log applier: failed to apply schema replication log. class: "t1", schema: "'alter class [t1] drop [c]'", internal error: -48." 오류 메시지와 함께 슬레이브 노드에 반영되지 않는 현상을 수정했다.

CUBRIDSUS-7810 AUTO_INCREMENT 칼럼이 있는 테이블에 대해 TRUNCATE 문 수행 시 슬레이브 노드에 반영되지 않는 현상 수정

HA 환경에서 AUTO_INCREMENT 칼럼이 있는 테이블에 대해 TRUNCATE 문을 수행하면 슬레이브 노드에서 "Accessing Delete Object" 오류 메시지와 함께 슬레이브 노드에 반영되지 않는 현상을 수정했다.

CUBRIDSUS-7830 AUTO_INCREMENT 칼럼 속성 또는 이름을 변경하면 슬레이브 노드에 반영되지 않는 현상 수정

HA 환경에서 AUTO_INCREMENT 칼럼 속성 변경 또는 AUTO_INCREMENT 칼럼 속성 추가와 함께 이름을 변경하면 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
// 속성 변경
CREATE TABLE foo (a INTEGER PRIMARY KEY AUTO_INCREMENT, b CHAR(10), c DATETIME);
ALTER TABLE foo MODIFY ATTRIBUTE a BIGINT AUTO_INCREMENT;

// 이름 변경
CREATE TABLE boo (i int PRIMARY KEY);
ALTER TABLE boo ADD COLUMN ai INT AUTO_INCREMENT, RENAME TO u, AUTO_INCREMENT = 100;
```

CUBRIDSUS-8420 테이블에 자동 증가 칼럼 추가 후 테이블 이름을 변경하면 해당 칼럼이 슬레이브 노드에 반영되지 않는 현상 수정

HA 환경에서 테이블에 자동 증가 칼럼 추가 후 테이블 이름을 변경하면, 해당 칼럼이 슬레이브 노드에 반영되지 않는 현상을 수정했다.

```
CREATE TABLE t (i int PRIMARY KEY);
ALTER TABLE t ADD COLUMN ai INT AUTO_INCREMENT, RENAME TO u, AUTO_INCREMENT = 100;
```

CUBRIDSUS-8634 인덱스를 CREATE/DROP하고 테이블을 DROP 하면 슬레이브 노드에서 -414번 오류가 발생하는 현상 수정

하나의 트랜잭션에서 인덱스를 CREATE/DROP하고 테이블을 DROP한 후 커밋하면 슬레이브 노드에서 정상 상황임에도 불구하고 "ERROR CODE = -414 ... Unknown class identifier" 오류가 발생하는 현상을 수정했다.

드라이버

CUBRIDSUS-6954 Linux용 CCI 라이브러리는 파일만으로도 드라이버의 버전 확인이 가능하도록 개선

Linux용 CCI 라이브러리는 파일만으로도 드라이버의 버전 확인이 가능하도록 개선했다.

```
$ strings /home/usr1/CUBRID/lib/libcascci.so ¦ grep VERSION
VERSION=8.4.3.0001
```

참고로, cci_get_version(major, minor, patch) 함수를 사용해도 드라이버의 버전 확인이 가능하다.

CUBRIDSUS-7657 CCI 응용 프로그램에서 statement_pooling을 사용하지 않을 때 datasource를 반납해도 요청 핸들이 정리되지 않는 문제 수정

CCI 응용 프로그램에서 statement_pooling을 사용하지 않도록 설정했을 때 cci_datasource_release 함수를 통해 datasource를 반납해도 해당 연결에 대한 요청 핸들이 모두 닫히지 않는 문제를 수정했다.

CUBRIDSUS-9435 CCI 응용 프로그램에서 T_CCI_ERROR 구조체 포인터를 사용하는 함수에 포인터 값으로 NULL을 입력하면 응용 프로그램이 비정상 종료하는 현상 수정

CCI 응용 프로그램에서 cci_datasource_release 등 T_CCI_ERROR 구조체 포인터를 사용하는 함수에 포인터 값으로 NULL을 입력하면 응용 프로그램이 비정상 종료하는 현상을 수정했다.

CUBRIDSUS-7910 CCI 응용 프로그램에서 cci_get_attr_type_str() 함수의 반환 값이 잘못된 문제 수정

cci_get_attr_type_str() 함수의 반환 값에서 제일 마지막 문자가 잘리는 문제를 수정했다.

```
// 소스 예제
char *attr_infos[][2] =
{
  {"aa", "character(1)"},
  {NULL, NULL}
};


...
res = cci_get_attr_type_str(conn, class_name, attr_infos[i][0], buf, buf_size, &error);
...
fprintf(LOG_FD, "%s attr: %s\n\n", attr_infos[i][0], buf);

// 수정 이전 출력 화면
aa attr: character(1
```

CUBRIDSUS-7226 CCI 응용 프로그램에서 잠금 타임아웃이 발생했을 때 CCI 함수에서 잘못된 오류 코드를 반환하는 문제 수정

cci_prepare() 및 cci_execute() 함수에서 잠금 타임아웃(lock timeout)이 발생했을 때 오류 코드 -75를 오류 코드 -493으로 래핑(wrapping)하여 반환하던 것을 -75번 오류가 직접 전달되도록 수정했다.

CUBRIDSUS-5633 CCI 응용 프로그램에서 에러 메시지 출력용 버퍼를 입력 인자로 하는 연결 함수 추가

오류 메시지 출력용 버퍼를 입력 인자로 하는 연결 함수인 cci_connect_ex(), cci_connect_with_url_ex()를 추가했다. 기존의 연결 함수는 오류 발생 시 하나의 오류 번호를 반환해서 상세한 오류가 무엇인지 알 수 없었으나, 수정 이후 오류 메시지 버퍼를 통해 상세 오류 번호를 확인할 수 있다.

```
T_CCI_ERROR error;

connection = cci_connect_ex ("localhost", 33000, "demodb", "dba", "pwd", &error);
connection = cci_connect_with_url_ex ("cci:cubrid:localhost:33000:demodb:::", "dba", "pwd",
&error);
```

CUBRIDSUS-9397 CCI 응용 프로그램에서 같은 SQL을 두 번 prepare 했을 때 statement pool에서 statement가 제거되지 않는 현상 수정

한 DB 연결에서 같은 SQL로 두 번 prepare 했을 때 두 번째 statement를 닫으면 사용 중인 statement pool에서 해당 statement가 제거되지 않는 현상을 수정했다.

CUBRIDSUS-7306 cci_connect_with_url() 함수의 URL 문자열에서 사용하는 autocommit 속성 제거

cci_connect_with_url() 함수의 URL 문자열에서 사용하는 autocommit 속성을 제거했다.

CUBRIDSUS-8924 2008 R2.2 CCI 드라이버로 접속 시 비정상적인 에러코드와 메시지 수정

2008 R2.2 CCI 드라이버로 데이터베이스 서버에 접속하면 에러코드와 메시지를 비정상적으로 출력하는 문제를 수정했다.

```
// 정상 오류 메시지의 예
execute error - err_code : -670, err_msg : Operation would have caused one or more unique
constraint violations.

// 비정상 오류 메시지의 예
execute error - err_code : -2, err_msg : yyybOperation would have caused one or more unique
constraint violations
```

CUBRIDSUS-9361 Windows용 CCI 응용 프로그램에서 연결 대상 DB 서버가 없으면 무한 대기하는 현상 수정

CUBRIDSUS-9496 JDBC의 executeBatch 메서드가 실행에 실패하는 현상 수정

JDBC의 executeBatch 메서드 실행 시 "Cannot communicate with the broker or received invalid packet" 오류 메시지와 함께 실패하는 현상을 수정했다.

CUBRIDSUS-6148 자동 커밋 모드에서, 배열 내 여러 개의 질의문을 일괄적으로 처리하는 함수가 각 질의문을 수행할 때마다 커밋하도록 수정

자동 커밋 모드에서 cci_execute_array 함수와 cci_execute_batch 함수, 그리고 JDBC의 Statement.executeBatch 메서드, PreparedStatement.executeBatch 메서드 등이 배열 내 여러 개의 질의문을 일괄적으로 처리할 때 모든 질의문을 수행한 이후에 커밋했으나 각 질의문을 수행할 때마다 커밋하도록 수정했다.

CUBRIDSUS-7532 getColumnClassName 메서드의 NUMERIC 타입 컬럼에 대한 JAVA 반환 타입 반환 변경

ResultSetMetaData.getColumnClassName 메소드가 NUMERIC 타입 칼럼에 대해 기존의 java.lang.Double 대신 java.math.BigDecimal을 반환하도록 수정했다.

CUBRIDSUS-7076 ResultSet.TYPE_SCROLL_SENSITIVE 옵션이 사용될 때 시스템 카탈로그 뷰에 대한 SELECT 질의에 실패하는 문제 수정

JDBC의 ResultSet.TYPE_SCROLL_SENSITIVE 옵션이 사용될 때 db_class와 같은 시스템 카탈로그 뷰에 대한 SELECT 질의를 수행한 뒤 fetch를 수행하면 "Semantic: System error (generate attr) in ../../src/parser/xasl_generation.c" 오류가 발생하는 문제를 수정했다.

ResultSet.TYPE_SCROLL_SENSITIVE 옵션은 현재 지원하지 않으므로, Resultset.TYPE_SCROLL_INSENSITIVE로 설정할 것을 권장한다.

CUBRIDSUS-9399 백슬래시를 이스케이프 문자로 사용하도록 설정하면 getColumns 메서드 수행 시 발생하는 오류

시스템 파라미터 no_backslash_escape의 값을 no로 설정하여 백슬래시를 이스케이프 문자로 사용하도록 설정하면 getColumns 메서드로 테이블의 칼럼 정보를 얻을 때 "unterminated string" 오류가 발생하는 현상을 수정했다.

CUBRIDSUS-8387 JDBC 응용 프로그램에서 NUMERIC 타입의 이름을 DECIMAL이 아니라 NUMERIC으로 반환

DatabaseMetaData.getColumns 메서드가 NUMERIC 타입의 이름을 DECIMAL로 반환하던 것을 NUMERIC으로 반환한다.

```
//수정 이전 버전에서는 Hibernate를 이용해서 엔티티 간 매핑 설정을 할 때 NUMERIC 타입의 칼럼을 지정하면
Caused by: org.hibernate.HibernateException: Wrong column type in mytbl_map for column col2. Found: decimal, expected: numeric(19,0)"
와 같은 오류가 발생했다.

@ManyToMany
@JoinTable(name="mytbl",
joinColumns={@JoinColumn(name="col1", columnDefinition="varchar(255)")},
inverseJoinColumns={@JoinColumn(name="col2", columnDefinition="numeric(19,0)")})
private Set<MyGroup> accessMyGroups;
```

CUBRIDSUS-8041 JDBC에서 Connection 시도가 connectTimeout에 의해 중단될 경우 브로커의 Job Queue가 쌓이는 현상 수정

JDBC에서 Connection 시도가 connectTimeout에 의해 중단된 이후 네트워크 소켓이 닫히지 않은 상태로 계속 재접속을 시도할 경우 브로커의 Job Queue가 쌓이는 문제를 수정했다.

CUBRIDSUS-9205 JDBC에서 PreparedStatement 객체에 대해 setBoolean 메서드로 값을 바인딩할 때 BIT 타입의 값만 허용하는 문제

JDBC에서 PreparedStatement 객체에 대해 setBoolean() 메서드로 값을 바인딩할 때 BIT 타입의 값만 허용하는 문제가 존재했으나 BIT 타입의 값은 제외하되 SMALLINT, INTEGER, BIGINT, DECIMAL, REAL , DOUBLE, MONETARY 등 모든 숫자 타입 또는 CHAR, VARCHAR 등 모든 문자 타입의 값을 바인딩할 수 있도록 수정했다.

CUBRIDSUS-6685 JDBC 응용 프로그램에서 NULL이 들어있는 CLOB 컬럼의 값을 fetch하지 못하는 현상 수정

JDBC의 statement 객체를 생성할 때 ResultSet.TYPE_SCROLL_SENSITIVE 플래그를 ON으로 설정한 뒤, NULL이 들어있는 CLOB 컬럼에 대해 SELECT 질의를 요청하면 오류가 발생하는 현상을 수정했다.

대표적인 예로 CUBRID Manager는 위 플래그를 ON으로 설정하므로, NULL이 있는 CLOB 컬럼 값을 fetch하지 못하는 문제가 존재했다.

CUBRIDSUS-8844 JDBC 응용 프로그램에서 음수년도의 입력을 허용하는 오류 수정

JDBC 응용 프로그램에서 Datetime 데이터를 bind 할 때 허용 범위(1-9999)를 벗어난 년도의 입력을 허용하는 오류를 수정했다.

CUBRIDSUS-7956 JDBC 응용 프로그램에서 브로커 로그에 연결 URL을 잘못 출력하는 오류 수정

JDBC 응용 프로그램에서 브로커 로그에 연결 URL을 출력할 때 "?"를 추가로 잘못 출력하는 오류를 수정했다. 다음은 수정 이전 버전에서 브로커 로그 파일에 URL을 잘못 출력한 예이다. dba:: 뒤에 나타나는 "?" 는 하나만 출력되어야 한다.

```
jdbc:cubrid:10.0.0.1:33000:demodb:dba::??queryTimeout=60000001&connectTimeout=10000
```

CUBRIDSUS-7943 JDBC 응용 프로그램에서 2.4E+2와 같은 과학적 표기법의 값을 BigDecimal로 바인딩하는데 실패하는 문제 수정

JDBC 응용 프로그램에서 2.4E+2와 같은 과학적 표기법(scientific notation)으로 표현된 값을 BigDecimal로 바인딩하는데 실패하는 문제를 수정했다.

```
BigDecimal x = new BigDecimal("240.0");
x = x.stripTrailingZeros(); // 과학 형식의 값으로 지정
p.setBigDecimal(1, x);
```

CUBRIDSUS-7616 JDBC 응용 프로그램에서 /*+ RECOMPILE */ 힌트가 있는 SELECT 문을 여러 스레드에서 동시에 실행하면 "Statement Pooling" 오류가 발생하는 문제 수정

JDBC 응용 프로그램에서 /*+ RECOMPILE */ 힌트가 있는 SELECT 문을 여러 스레드에서 동시에 실행하면 "cubrid.jdbc.driver.CUBRIDException: Statement Pooling" 오류가 발생하는 문제를 수정했다.

유틸리티

CUBRIDSUS-9537 CSQL의 --line-output 설정을 CSQL 세션 명령으로 설정 가능하도록 변경하고, SQL 실행 시간을 기본으로 출력하도록 변경

CSQL의 --line-output 설정을 CSQL의 세션 명령으로 설정 가능하도록 변경했다.

```
csql> ;line-output on

csql> ;line-output off
```

또한, CSQL에서 SQL 수행 시 수행한 시간을 기본으로 출력하도록 변경했다.

CUBRIDSUS-9629 CSQL에서 특정 SQL 문장의 경우 문장 끝에 도달했음을 인식하지 못하는 현상 수정

CSQL에서 특정 SQL 문장의 경우 문장의 끝에 도달했음을 인식하지 못해 수행이 되지 않는 현상을 수정했다. 그 예로, 세미콜론(;) 뒤에 공백이나 탭, 또는 주석(--)이 포함된 질의문, 작은 따옴표 사이에 개행 문자가 들어간 질의문, 시스템 파라미터 no_backslash_escapes의 값을 no로 설정한 환경에서 INSERT INTO t VALUES(1,'₩''); 와 같이 이스케이프 문자가 포함된 질의문 등이 있다.

```
INSERT INTO t VALUES(1,'test');  --
INSERT INTO t VALUES(1,'
');
INSERT INTO t VALUES(1,'\'');
```

CUBRIDSUS-8957 CSQL을 독립 모드에서 실행할 때 세션 변수를 지원

CSQL을 독립 모드에서 실행하면 세션 변수를 지원하지 않았으나 지원하도록 수정했다.

CUBRIDSUS-8456 CSQL에서 질의 계획 보기 수행 중 Ctrl+C한 후 스키마 내용을 조회하면 비정상 종료되는 현상 수정

CSQL 인터프리터에서 ;info plan으로 질의 계획 보기 수행 중 Ctrl+C한 후 ;sc <table> 명령으로 스키마 내용을 조회하면 비정상 종료되는 현상을 수정했다.

CUBRIDSUS-7848 CSQL -i 옵션을 이용하여 파일로 질의문 입력 시 질의문이 매우 길거나 여러 라인으로 작성되면 질의문이 잘못 변경될 수 있는 문제

CSQL 인터프리터에서 -i 옵션을 이용하여 파일로 질의문을 입력할 때 질의문이 매우 길거나 여러 라인으로 작성되면, 긴 질의를 나누어 저장하는 버퍼의 끝 또는 라인의 끝에 나타나는 빈 문자열을 잘라냄으로 인해 질의문이 변경되어 의도하지 않은 값이 입력되거나 에러가 발생할 수 있는 문제를 수정했다.

CUBRIDSUS-9478 서버의 최대 접속 개수를 초과해도 CSQL에서 --sysadm으로 추가 접속이 가능하도록 수정

시스템 파라미터 max_clients 값에 의해 설정된 서버의 최대 접속 개수를 초과하더라도 CSQL에서 시스템 관리자 모드(--sysadm)로 단 하나의 추가 접속이 가능하도록 수정했다.

```
csql -u dba --sysadm testdb
```

CUBRIDSUS-9245 문자열 바인딩 값의 첫번째 문자로 ')'가 오면 broker_log_top 수행 시 비정상 종료할 수 있는 문제 수정

CUBRIDSUS-7462 CUBRID Manager에서 질의 편집기 수행 종료 이후에 cubrid.conf에서 설정한 잠금 타임아웃 값이 줄어들 수 있는 오류 수정

CUBRID Manager에서 질의 편집기를 수행하면 잠금 타임아웃이 내부적으로 1초로 설정되는데, 이 질의 편집기를 종료하면 이것과 연결되었던 CAS는 cubrid.conf의 lock_timeout_in_secs에서 설정한 값으로 잠금 타임아웃을 재설정한다. 이때 해당 CAS의 잠금 타임아웃이 설정 값의 1/1000로 줄어드는 오류로 인해, 이 CAS에 연결되는 다른 응용 프로그램이 줄어든 잠금 타임아웃을 사용하는 문제를 수정했다. 이 CAS는 재시작될 때까지 줄어든 잠금 타임아웃을 유지한다.

단, lock_timeout_in_secs가 기본값인 -1로 설정된 경우 이 오류는 발생하지 않았다.

CUBRIDSUS-7484 loaddb 명령의 --error-control-file 옵션이 정상 동작하지 않는 오류 수정

loaddb 수행 중 특정 에러는 무시하고 계속 진행하도록 --error-control-file 옵션을 설정했음에도 불구하고, 해당 에러가 발생하면 수행을 종료하는 오류를 수정했다.

```
cubrid loaddb -u dba -c 10000 -d tbl.ldb --error-control-file=error_lst testdb

Time: 04/04/12 20:05:47.066 - WARNING *** file ../../src/storage/heap_file.c, line 9678 CODE =
-48 Tran = 1, EID = 16661
Accessing deleted object 2|607786|233."
```

CUBRIDSUS-6647 loaddb 수행 중 서버에서 Warning이 발생하면 수행을 지속하지 않고 종료하는 문제 수정

loaddb 수행 중 서버에서 Warning이 발생하면 커밋되지 않은 레코드들을 모두 롤백하면서 loaddb의 수행을 종료하는 문제가 존재했으나, Warning 이후에도 계속 수행하도록 수정했다.

CUBRIDSUS-9109 loaddb 스키마 파일에 백슬래시("₩")가 마지막 글자인 문자열이 포함되면 loaddb가 실패하거나 잘못된 값으로 로딩되는 현상 수정

CUBRIDSUS-8864 데이터베이스 생성 도중 ctrl+C로 작업 취소 시 비정상적인 오류를 출력하는 문제

createdb 실행 도중 ctrl+c를 눌러서 작업을 취소하면 "*** FATAL ERROR *** Internal error: logical log page -9 may be corrupted." 오류가 발생하는 문제를 수정했다.

CUBRIDSUS-9337 백업 수행 시 체크포인트 오퍼레이션이 진행 중이면 백업 작업을 중단할 수 없는 문제 수정

백업을 수행할 때 체크포인트 오퍼레이션이 진행중이면 백업 작업에 대해서 인터럽트(Ctrl+C)를 해도 백업 작업이 중단되지 않는 문제를 수정했다.

CUBRIDSUS-6867 데이터베이스 종료 수행 시간이 체크포인트로 인해 1분 이상 걸리면 체크포인트를 취소하도록 수정

cubrid server stop 명령으로 데이터베이스 종료를 수행하는 시간이 1분 이상 걸리면 데이터베이스를 강제 종료하게 되어 있다. 셧다운 과정 중에 체크포인트(checkpoint) 오퍼레이션이 오래 걸리는 경우에도 데이터베이스 서버를 강제 종료함으로 인해서 이후 데이터베이스 재시작 시 복구 시간이 오래 걸릴 수 있었으나, 셧다운 과정 중에 타임아웃이 발생되면 체크포인트 과정을 취소하고 데이터베이스를 종료하도록 수정했다.

CUBRIDSUS-7434 CS 모드로 checkdb 명령 수행 시 DB 서버 프로세스가 비정상 종료될 수 있는 문제 수정

중복 값이 매우 많은 비고유(non-unique) 인덱스가 존재하는 상황에서 CS 모드로 checkdb 명령 수행 시 DB 서버 프로세스가 비정상 종료될 수 있는 문제를 수정했다.

오류 메시지

CUBRIDSUS-6885 고유 키 위반 오류 메시지 출력 시 테이블, 인덱스 이름, 키 값을 출력하도록 개선

고유 키를 위반한 오류 메시지를 오류 로그에 출력할 때 테이블, 인덱스 이름, 키 값을 함께 출력하도록 수정했다.

CUBRIDSUS-9458 Warning을 출력하도록 설정된 환경에서 DML 수행 시 warning 메시지를 출력하는 현상 수정

Warning을 출력하도록 설정된 환경에서 정상적인 INSERT/UPDATE/DELETE를 수행함에도 불구하고 warning 메시지를 출력하는 현상을 수정했다.

설정, 빌드 및 설치

CUBRIDSUS-7227 UTF-8 문자셋을 테이블 이름이나 칼럼 이름 등의 식별자로 사용 가능해짐

UTF-8 문자셋을 테이블 명이나 컬럼 명 등의 식별자로 사용 가능하도록 개선했다. 주의할 사항으로, 수정 이후 버전에서도 UTF-8 문자셋으로 구성된 문자열을 검색하는 경우 cubrid.conf의 single_byte_compare를 반드시 yes로 설정해야 한다.

CUBRIDSUS-6910 Linux용 CUBRID에서 cub_master, cub_broker 프로세스 수행 시 생성되는 유닉스 도메인 소켓 파일의 위치를 바꿀 수 있도록 개선

Linux용 CUBRID에서 cub_master, cub_broker 프로세스 수행 시 생성되는 유닉스 도메인 소켓(unix domain socket) 파일의 위치를 CUBRID_TMP 환경 변수를 통해 변경할 수 있도록 수정했다. cub_master의 유닉스 도메인 소켓 파일의 위치는 기존에는 /tmp였으며 수정 이후에도 $CUBRID_TMP 환경 변수를 설정하지 않으면 기존 위치와 같다. cub_broker의 유닉스 도메인 소켓 파일의 위치는 기존에는 $CUBRID/var/CUBRID_SOCK이었으며, 수정 이후에도 $CUBRID_TMP를 설정하지 않으면 기존 위치와 같다.

$CUBRID_TMP 환경 변수를 통해 다음 두 가지 문제를 회피할 수 있다.

- cubrid_broker용 유닉스 도메인 소켓 파일을 저장하는 $CUBRID/var/CUBRID_SOCK 경로의 최대 길이는 108인데, CUBRID의 설치 경로가 길어서 경로 길이가 108을 넘는 경우
- /tmp 디렉터리의 파일을 관리 상의 목적 등으로 임의로 삭제하는 경우

CUBRIDSUS-7233 데이터베이스 서버에 동시 접속을 허용하는 최대 접속 개수를 10000개로 늘림

데이터베이스 서버에 동시 접속을 허용하는 최대 개수를 설정하는 max_clients 시스템 파라미터의 최대값 제한을 1024에서 10000으로 늘렸다. 참고로, 클라이언트의 DB 접속 여부에 관계 없이 max_clients의 개수를 크게 설정할수록 메모리 사용량이 증가하므로 이에 주의하도록 한다.

CUBRIDSUS-9392 브로커 이름이 길면 응용 프로그램에서 브로커 접속시 브로커가 비정상 종료되는 문제 수정

브로커 이름의 길이가 32자 이상이면 응용 프로그램에서 브로커에 접속 시 브로커가 비정상 종료되었으나, 브로커 이름의 길이를 63자까지만 허용하고 또한 브로커 이름이 이보다 길면 브로커를 시작하지 않도록 수정했다.

CUBRIDSUS-8005 $TMPDIR 환경 변수 설정 시 Linux sh 패키지로 CUBRID 설치에 실패하는 문제 수정

$TMPDIR 환경 변수의 값이 설정되어 있으면 Linux sh 패키지를 이용하여 CUBRID를 설치하는데 실패하는 문제를 수정했다.

CUBRIDSUS-7809 RPM 패키지의 의존성 오류 수정

RPM 패키지 설치 시 의존성(dependencies) 오류를 수정했다.

CUBRIDSUS-7812 Ubuntu에서 CUBRID 소스를 빌드할 때 Warning 이 발생하는 현상 수정

Ubuntu에서 configure를 수행할 때 automake 1.11.3을 쓰게 되면서 Warning이 발생하는 현상을 수정했다.

기타

CUBRIDSUS-8351 브로커 및 데이터베이스 서버의 에러 로그를 임의로 삭제하면 재생성되지 않는 문제

브로커 및 데이터베이스 서버의 에러 로그를 임의로 삭제하면 재생성되도록 수정했다.

CUBRIDSUS-9450 브로커 응용 서버에서 인터럽트 발생 시 DB 서버에서 발생한 에러가 잘못 전달되는 문제 수정

브로커 응용 서버(CAS)에서 인터럽트가 발생할 때 DB 서버에서 발생한 에러가 다른 에러로 전달되는 문제를 수정했다.

CUBRIDSUS-8163 백그라운드 보관 로그 파일에 대해 파일 I/O sync. 오류 메시지를 잘못 출력하는 현상 수정

백그라운드 보관 로그 파일에 대해 다음과 같이 파일 I/O sync. 오류 메시지(에러 코드: -599)를 잘못 출력하는 현상을 수정했다.

```
An I/O error occurred while synchronizing state of volume
"/home/cubrid/database/testdb/testdb_lgar_t".... Bad file descriptor
```

4.5 주의 사항

신규 주의 사항

CUBRIDSUS-9364 CCI 응용 프로그램에서 여러 개의 질의를 한 번에 수행한 결과의 배열에 대한 에러 처리 방식 수정

CCI 응용에서 여러 개의 질의를 한 번에 수행할 때 2008 R3.0부터 2008 R4.1 버전까지는 cci_execute_array 함수, cci_execute_batch 함수 또는 cci_execute_result 함수에 의한 질의 수행 결과들 중 하나만 에러가 발생해도 해당 질의의 에러 코드를 반환했으나, 2008 R4.3 버전부터는 전체 질의 개수를 반환하고 CCI_QUERY_RESULT_* 매크로들을 통해 개별 질의에 대한 에러를 확인할 수 있도록 수정했다.

수정 이전 버전에서는 에러가 발생한 경우에도 배열 내 각각의 질의들의 성공 실패 여부를 알 수 없으므로, 이를 판단해야 한다.

```
...
char *query = "INSERT INTO test_data (id, ndata, cdata, sdata, ldata) VALUES (?, ?, 'A',
'ABCD', 1234)";
...
req = cci_prepare (con, query, 0, &cci_error);
...
error = cci_bind_param_array_size (req, 3);
...
error = cci_bind_param_array (req, 1, CCI_A_TYPE_INT, co_ex, null_ind, CCI_U_TYPE_INT);
...
n_executed = cci_execute_array (req, &result, &cci_error);

if (n_executed < 0)
{
  printf ("execute error: %d, %s\n", cci_error.err_code, cci_error.err_msg);

  for (i = 1; i <= 3; i++)
  {
    printf ("query %d\n", i);
    printf ("result count = %d\n", CCI_QUERY_RESULT_RESULT (result, i));
    printf ("error message = %s\n", CCI_QUERY_RESULT_ERR_MSG (result, i));
    printf ("statement type = %d\n", CCI_QUERY_RESULT_STMT_TYPE (result, i));
  }
}
...
```

수정 이후 버전에서는 에러가 발생하면 전체 질의가 실패한 것이며, 에러가 발생하지 않은 경우에 대해 배열 내 각 질의들의 성공 실패 여부를 판단한다.

```
...
char *query = "INSERT INTO test_data (id, ndata, cdata, sdata, ldata) VALUES (?, ?, 'A',
'ABCD', 1234)";
...
req = cci_prepare (con, query, 0, &cci_error);
...
error = cci_bind_param_array_size (req, 3);
...
error = cci_bind_param_array (req, 1, CCI_A_TYPE_INT, co_ex, null_ind, CCI_U_TYPE_INT);
...
n_executed = cci_execute_array (req, &result, &cci_error);
```

```
if (n_executed < 0)
{
  printf ("execute error: %d, %s\n", cci_error.err_code, cci_error.err_msg);
}
else {
  for (i = 1; i <= 3; i++)
  {
    printf ("query %d\n", i);
    printf ("result count = %d\n", CCI_QUERY_RESULT_RESULT (result, i));
    printf ("error message = %s\n", CCI_QUERY_RESULT_ERR_MSG (result, i));
    printf ("statement type = %d\n", CCI_QUERY_RESULT_STMT_TYPE (result, i));
  }
}
...
```

기존 주의 사항

CUBRIDSUS-5879 2008 R4.1 버전부터 CCI_DEFAULT_AUTOCOMMIT 의 기본값이 ON으로 바뀜

2008 R4.1 버전부터 CCI 인터페이스로 개발된 응용 프로그램의 자동 커밋 모드에 영향을 주는 브로커 파라미터인 CCI_DEFAULT_AUTOCOMMIT의 기본값이 ON으로 변경되었다. 따라서 CCI 및 CCI로 개발된 인터페이스(PHP, ODBC, OLE DB 등) 사용자는 응용 프로그램의 자동 커밋 모드가 이에 적합한지 살펴보아야 한다.

CUBRIDSUS-5238 2008 R4.3 버전은 2008 R4.0 Beta 및 그 이전 버전과 DB 볼륨이 호환되지 않음

- 2008 R4.0 버전 이상 사용자는 2008 R4.3과 DB 볼륨이 호환된다.
- 2008 R4.3 버전은 2008 R4.0 Beta 및 그 이전 버전과 DB 볼륨이 호환되지 않으므로 DB를 업그레이드하려면 DB 볼륨을 마이그레이션해야 한다.

- 2008 R3.x 및 이전 버전 사용자는 cubrid unloaddb/loaddb를 이용한다.

- 2008 R4.0 Beta 사용자는 다운로드 페이지에서 제공되는 migrate_r40beta2ga 유틸리티를 이용할 수 있으나, 페이지 크기가 4K 미만인 볼륨은 cubrid unloaddb/loaddb를 이용하여야 한다.

보다 자세한 사항은 DB 마이그레이션 절차를 참고한다.

CUBRIDSUS-5597 CCI, PHP, JDBC 연결 URL에서 구분자를 암호에 포함할 수 없음

CCI, PHP, JDBC 연결 URL에서 구분자로 사용되는 ? 또는 :을 암호에 포함할 수 없다. 다음은 암호에 ?가 있어 연결 URL로 사용할 수 없는 예이다.

```
// CCI, PHP의 경우
url = "cci:jdbc:CUBRID:192.168.0.10:33000:tdb:dba:12?:?charset=UTF-8";
// JDBC의 경우
url="jdbc:CUBRID:192.168.0.10:33000:tdb:dba:12?:?charset=UTF-8";
```

암호에 ? 또는 :을 포함한 경우에는 암호를 별도의 인자로 전달하여 사용할 수 있다.

```
// CCI의 경우
url="cci:CUBRID:192.168.0.10:33000:tdb:::?charset=UTF-8";
cci_connect_with_url(url, "dba", "12?");
// PHP의 경우
url="cci:CUBRID:192.168.0.10:33000:tdb:::?charset=UTF-8";
cubrid_connect_with_url(url, "dba", "12?");
// JDBC의 경우
url="jdbc:CUBRID:192.168.0.10:33000:tdb:::?charset=UTF-8";
conn = DriverManager.getConnection(url,"dba", "12?");
```

이 주의 사항은 모든 버전에 해당한다.

CUBRIDSUS-5136 페이지 단위의 옵션 제거 예정

createdb 명령의 DB 볼륨 크기 및 로그 볼륨 크기를 지정할 때 페이지 단위를 사용하는 옵션들(-p, -l, -s)은 제거될 예정이므로, 2008 R4.0 Beta 이후 새로 추가된 옵션들(--db-volume-size, --log-volume-size, --db-page-size, --log-page-size)을 사용한다.

addvoldb 명령의 DB 볼륨 크기를 지정하는 경우에도 페이지 단위를 사용하지 않고 2008 R4.0 Beta 이후 새로 추가된 옵션(--db-volume-size)을 사용한다.

CUBRIDSUS-4222 DB 볼륨 크기 설정 시 주의 사항

2008 R4.0 Beta 버전부터 DB 생성 시 데이터 페이지 및 로그 페이지의 크기 기본값이 4KB에서 16KB로 변경되었으므로, DB 볼륨을 페이지 개수로 지정하여 생성하는 경우 볼륨의 바이트 크기가 기대와 다를 수 있음에 주의한다. 아무런 옵션도 주지 않을 경우 이전 버전에서는 4KB의 페이지 크기로 100MB의 DB 볼륨을 생성하였으나, 2008 R4.0 버전부터는 16KB의 페이지 크기로 512MB의 DB 볼륨을 생성하게 된다.

그리고, DB 볼륨의 생성 가능한 최소 크기를 20MB로 제한하였으므로 이보다 작은 크기의 DB 볼륨은 생성할 수 없다.

CUBRIDSUS-4222 페이지 단위의 시스템 파라미터 제거 예정

페이지 단위의 시스템 파라미터들은 추후 제거될 예정이므로 바이트 단위의 새로운 시스템 파라미터를 사용할 것을 권장한다. 관련 시스템 파라미터들에 대한 내용은 아래를 참고한다.

CUBRIDSUS-4095 일부 시스템 파라미터들의 기본값 변경

2008 R4.0부터 다음 시스템 파라미터들의 기본값이 변경되었다.

DB 서버가 허용하는 동시 연결 개수를 설정하는 max_clients의 기본값, 인덱스 페이지 생성 시 향후 업데이트를 대비하여 확보하는 여유 공간 비율을 설정하는 index_unfill_factor의 기본값이 변경되었으며, 바이트 단위 시스템 파라미터의 기본값이 기존 페이지 단위 시스템 파라미터의 기본값보다 커져서 별도의 설정을 하지 않는 경우 더 많은 메모리를 사용하게 되었다.

기존 시스템 파라미터	추가된 시스템 파라미터	기존 기본값	변경된 기본값 (단위: 바이트)
max_clients	-	50	100
index_unfill_factor	-	0.2	0.05
data_buffer_pages	data_buffer_size	100M(페이지 크기=4K)	512M
log_buffer_pages	log_buffer_size	200K(페이지 크기=4K)	4M
sort_buffer_pages	sort_buffer_size	64K(페이지 크기=4K)	2M
index_scan_oid_buffer_pages	index_scan_oid_buffer_size	16K(페이지 크기=4K)	64K

또한, createdb로 DB 생성 시 데이터 페이지 크기와 로그 페이지 크기의 최소값이 1K에서 4K로 변경되었다.

CUBRIDSUS-5375 시스템 파라미터를 잘못 설정하면 DB 서비스, 유틸리티 및 응응 프로그램이 구동되지 않음

cubrid.conf 또는 cubrid_ha.conf에 정의되지 않은 시스템 파라미터를 설정하거나, 페이지 단위의 시스템 파라미터와 바이트 단위의 시스템 파라미터가 동시에 사용되거나, 시스템 파라미터 값이 허용 범위를 벗어나면 이와 관련된 DB 서비스, 유틸리티 및 응응 프로그램이 구동되지 않는다.

CUBRIDSUS-4524 복제 기능 제거

CUBRID 2008 R4.0 버전부터 복제 기능이 제거되었으므로, 이중화 환경을 구축하려면 HA 기능을 사용해야 한다. 서버 버전 업그레이드 및 DB 마이그레이션을 수행한 후, HA 환경을 새롭게 구축할 수 있다.

CUBRIDSUS-5228 CUBRID 매니저 설치 패키지 별도 제공

CUBRID 2008 R4.0부터는 CUBRID 매니저 설치 패키지를 별도로 제공하며, CUBRID 매니저를 사용하려면 CUBRID 패키지 설치 후 CUBRID 매니저를 별도로 설치해야 한다. .

CUBRIDSUS-5097 컬럼 크기보다 큰 문자열을 INSERT/UPDATE 할 때 문자열이 절삭되어 입력됨

컬럼 크기보다 큰 CHAR, VARCHAR, NCHAR, VARNCHAR 타입의 문자열을 INSERT/UPDATE하면 컬럼 크기를 초과하는 문자열 부분을 절삭한다.

CUBRIDSUS-5349 CUBRID 32bit 버전에서 data_buffer_size에 2G를 초과하는 값을 설정하면 DB 구동에 실패함

CUBRID 32bit 버전에서 **data_buffer_size**가 2G를 초과하는 값으로 설정되는 경우 DB 구동에 실패한다. 32bit 버전에서는 OS의 한계로 인해 설정값이 2G를 초과할 수 없음에 주의한다.

CUBRIDSUS-4059 VARCHAR 타입의 컬럼에서 값을 가져올 때 커버링 인덱스가 적용되는 경우 뒤에 따르는 공백 문자열이 무시됨

VARCHAR 타입의 컬럼에서 값을 가져올 때 커버링 인덱스가 적용되는 경우 뒤에 따라오는 공백 문자열은 절삭된다. 질의 수행 시 커버링 인덱스가 적용되면 질의 결과 값을 인덱스에서 가져오는데, 인덱스에는 뒤이어 나타나는 공백 문자열을 제거한 채로 값을 저장하기 때문이다. 이러한 현상을 원하지 않을 경우에는 NO_COVERING_IDX 힌트를 지정하면 된다.

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

-- 아래 질의는 커버링 인덱스가 적용되어 3개의 데이터가 모두 같은 조건으로 인식된다.
SELECT * FROM tab WHERE c='abcd ' USING INDEX i_tab_c(+);
c
======================
'abcd'
'abcd'
'abcd'
```

CUBRIDSUS-3757 HA 관련 주의 사항

CUBRID HA에서 트리거 및 자바 저장 프로시저를 사용할 경우 마스터 노드에서 이미 수행된 트리거 또는 자바 저장 프로시저를 슬레이브 노드에서 중복 수행하여 CUBRID HA 그룹 내의 노드 간 데이터 불일치가 발생할 수 있으므로, CUBRID HA에서는 트리거 및 자바 저장 프로시저를 사용하지 않도록 한다.

CUBRID HA는 복제 로그를 기반으로 CUBRID HA 그룹 내의 노드 간 데이터를 동기화하므로 복제 로그를 생성하지 않는 메소드를 사용하거나 CUBRID 매니저를 통해 NOT NULL 옵션 설정 작업 수행 시 CUBRID HA 그룹 내 노드 간 데이터 불일치가 발생할 수 있으므로, CUBRID HA는 메소드를 사용할 수 없고, CUBRID 매니저를 통해 작업할 수 없다.

CUBRIDSUS-5071 DB 백업/복구 시 LOB 타입 저장소는 복구되지 않음

CUBRID에서 LOB 타입의 데이터는 DB 볼륨이 아닌 별도의 저장소에 존재하므로 DB의 백업/복구 과정에 포함되지 않는다. 즉, DB의 백업 시 LOB 타입 저장소는 같이 백업되지 않으므로, 복구 시 LOB 타입 저장소는 복구되지 않는다. LOB 타입 저장소는 별도로 관리해야 한다.

CUBRIDSUS-3826 GLO 클래스 지원 중단에 따른 주의 사항

CUBRID 2008 R3.0 이하 버전은 **glo**(Generalized Large Object) 클래스를 사용하여 Large Object를 처리하였으나, CUBRID 2008 R3.1 이상 버전 **glo** 클래스를 제거하고 BLOB, CLOB 타입(이하 LOB)을 지원한다.

- 기존의 **glo** 클래스 사용자는 다음과 같이 작업할 것을 권장한다.
- GLO 데이터를 파일로 저장한 후 어플리케이션 및 DB 스키마에서 GLO를 사용하지 않도록 수정한다.
- DB 마이그레이션을 한다. (2008 R3.x 및 그 이전 버전에서 2008 R4.3으로 마이그레이션 참고)
- 변경한 어플리케이션에 맞게 파일을 LOB 데이터로 로딩하는 작업을 수행하도록 한다.
- 수정한 어플리케이션이 정상 동작하는지 확인한다.

참고로, cubrid loaddb 유틸리티는 GLO 클래스를 상속받거나 GLO 클래스 타입을 가진 테이블을 로딩하려는 경우 "Error occurred during schema loading" 오류 메시지와 함께 데이터 로딩을 중지한다.

GLO 클래스의 지원 중단에 따라 각 인터페이스 별로 삭제한 함수는 다음과 같다.

인터페이스	삭제한 함수
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 BLOB, CLOB 타입 사용 시 제약 사항

BLOB, CLOB 타입(이하 LOB)에 대하여 다음과 같은 제약 사항이 있으므로 사용에 주의한다.

- LOB 타입 컬럼 간 비교 연산(=, <>, IN, NOT IN 등)을 할 수 없으며, 이를 위해서는 문자열 또는 비트열로 타입을 변환한 후 사용해야 한다. 단, IS NULL, IS NOT NULL은 지원한다.
- PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL 제약 조건 또는 인덱스를 정의할 수 없다.
- 테이블 생성 및 수정 시 SHARED 속성을 정의할 수 없으며 DEFAULT 속성은 NULL 값에 대해서만 정의할 수 있다.
- DB에는 파일의 위치(LOB Locator)가 저장되고 데이터는 파일로 저장되는 구조이므로, 장애가 발생하여 특정 시점으로 복구할 때 LOB Locator와 LOB 데이터의 매핑이 유효하지 않아 에러가 발생할 수 있다.
- ALTER TABLE DROP 문을 사용하여 컬럼을 삭제하거나, DROP TABLE 문을 사용하여 테이블을 삭제하는 경우 LOB Locator만 삭제되고 LOB 컬럼이 참조하는 외부 파일 시스템의 LOB 파일은 삭제되지 않고 남아있다.
- CUBRID가 제공하는 API나 CUBRID 매니저, CSQL을 사용하지 않고 사용자 임의로 LOB 타입의 데이터 파일을 직접 수정하면 내용이 일치됨을 보장할 수 없다.

CUBRIDSUS-4186 Windows Vista 이상 버전에서 CUBRID 유틸리티를 사용한 서비스 제어 시 권장 사항

Windows Vista 이상 버전에서 cubrid 유틸리티를 사용하여 서비스를 제어하려면 명령 프롬프트 창을 관리자 권한으로 구동한 후 사용하는 것을 권장한다.

명령 프롬프트 창을 관리자 권한으로 구동하지 않고 cubrid 유틸리티를 사용하는 경우 UAC(User Account Control) 대화 상자를 통하여 관리자 권한으로 수행될 수 있으나 수행 결과 메시지를 확인할 수 없다.

Windows Vista 이상 버전에서 명령 프롬프트 창을 관리자 권한으로 구동하는 방법은 다음과 같다.

- [시작> 모든 프로그램> 보조 프로그램> 명령 프롬프트]에서 마우스 오른쪽 버튼을 클릭한다.
- [관리자 권한으로 실행(A)]을 선택하면 권한 상승을 확인하는 대화 상자가 활성화되고, “예”를 클릭하여 관리자 권한으로 구동한다.

CUBRIDSUS-3217 JDBC에서 연결 정보를 URL 스트링으로 입력하는 경우 물음표를 반드시 명시

JDBC에서 URL 스트링으로 연결 정보를 입력하는 경우 이전 버전에서는 물음표(?)를 입력하지 않더라도 속성(PROPERTY) 정보가 적용되었으나, CUBRID 2008 R3.0부터는 문법에 따라 반드시 물음표를 명시해야 하고 이를 생략할 경우 에러를 출력한다. 또한, 연결 정보 중 USERNAME과 PASSWORD가 없더라도 반드시 콜론(:)을 명시해야 한다.

```
URL=jdbc:CUBRID:127.0.0.1:31000:db1:::althosts=127.0.0.2:31000,127.0.0.3:31000  - 에러 처리
URL=jdbc:CUBRID:127.0.0.1:31000:db1:::?althosts=127.0.0.2:31000,127.0.0.3:31000  - 정상 처리
```

CUBRIDSUS-3564 마스터 프로세스와 서버 프로세스 간 프로토콜 변경 및 두 개 버전을 동시에 운영하는 경우 포트 설정 필요

마스터 프로세스(cub_master)와 서버 프로세스(cub_server) 간 통신 프로토콜 변경으로 인해 CUBRID 2008 R3.0 이상 버전의 마스터 프로세스는 하위 버전의 서버 프로세스와 통신할 수 없고, 하위 버전의 마스터 프로세스도 2008 R3.0 이상 버전의 서버 프로세스와 통신할 수 없다. 따라서, 이미 하위 버전이 설치되어 있는 환경에서 새 버전을 추가 설치하여, 두 개 버전의 CUBRID를 동시에 운영하는 경우 각각 서로 다른 포트를 사용하도록 cubrid.conf의 cubrid_port_id 시스템 파라미터를 수정해야 한다.

CUBRIDSUS-2828 DB 이름에 @를 포함할 수 없음

DB 이름에 @이 포함되는 경우 호스트 이름이 명시된 것으로 해석될 수 있으므로 이를 방지하기 위하여 createdb, renamedb, copydb 명령 수행 시 DB 이름에 @를 포함할 수 없도록 수정했다.

CUBRIDSUS-3267 Windows 환경에서 디렉터리 경로 설정 시 주의 사항

Windows 환경에서 CUBRID 설치 디렉터리 경로에 공백을 포함하는 경우 정상 설치가 되지 않으므로 주의한다. 또한, unloaddb, loaddb, backupdb 등의 작업 대상 디렉터리 경로에도 공백을 포함할 수 없다.

CUBRIDSUS-3553 CUBRID 소스 빌드 후 실행 시, 매니저 서버 프로세스 관련 오류 발생

사용자가 직접 빌드하여 설치하는 경우, CUBRID와 CUBRID 매니저를 각각 빌드하여 설치해야 한다. 만약, CUBRID 소스만 checkout하여 빌드 후 cubrid service start 또는 cubrid manager start를 실행하면, cubrid manager server is not installed라는 오류가 발생한다.

5

CUBRID 8.4.1 릴리스 노트

5.1 CUBRID 2008 R4.1 정보

릴리스 노트 정보

본 문서는 CUBRID 2008 R4.1 및 Patch 버전에 관한 유용한 정보를 포함한다. 릴리스 노트의 최신 버전은 CUBRID 오픈 소스 프로젝트 사이트(http://dev.naver.com/projects/cubrid)에서 확인할 수 있다.

릴리스 노트 개정 내역

CUBRID 2008 R4.1 버전의 릴리스 이후 릴리스 노트의 변경 사항은 아래와 같다.

작성 날짜	설명
2012년 10월	CUBRID 2008 R4.1 Patch 7 릴리스(8.4.1.7007)
2012 8월	CUBRID 2008 R4.1 Patch 6 릴리스(8.4.1.6004)
2012 7월	CUBRID 2008 R4.1 Patch 5 릴리스(8.4.1.5002)
2012 6월	CUBRID 2008 R4.1 Patch 4 릴리스(8.4.1.4001)
2012 6월	CUBRID 2008 R4.1 Patch 3 릴리스(8.4.1.3024)
2012 4월	CUBRID 2008 R4.1 Patch 2 릴리스(8.4.1.2032)
2012 2월	CUBRID 2008 R4.1 Patch 1 릴리스(8.4.1.1018)
2012 1월	CUBRID 2008 R4.1 릴리스(8.4.1.0564)

릴리스 특징

대량의 트랜잭션 부하 테스트 시 INSERT와 UPDATE 성능이 이전 버전 대비 70% 향상되었고, 다양한 SQL 함수를 추가하였으며 조건 검색 시 정규 표현식을 지원한다. 또한, 브로커 파라미터의 추가 및 브로커 모니터링이 개선되었다. 아울러 50여개의 크고 작은 버그들을 수정했다.

동작에 영향을 미치는 변경 중 특히 주의할 사항으로, 브로커 파라미터인 **CCI_DEFAULT_AUTOCOMMIT**의 기본값이 ON으로 변경되었다. 따라서 CCI 및 CCI로 개발된 PHP, ODBC, OLE DB 등의 드라이버 사용자가 자동 커밋 모드의 기본값을 2008 R4.1 이전 버전과 같이 유지하려면 **cubrid_broker.conf**의 **CCI_DEFAULT_AUTOCOMMIT** 값을 OFF로 설정해야 한다.

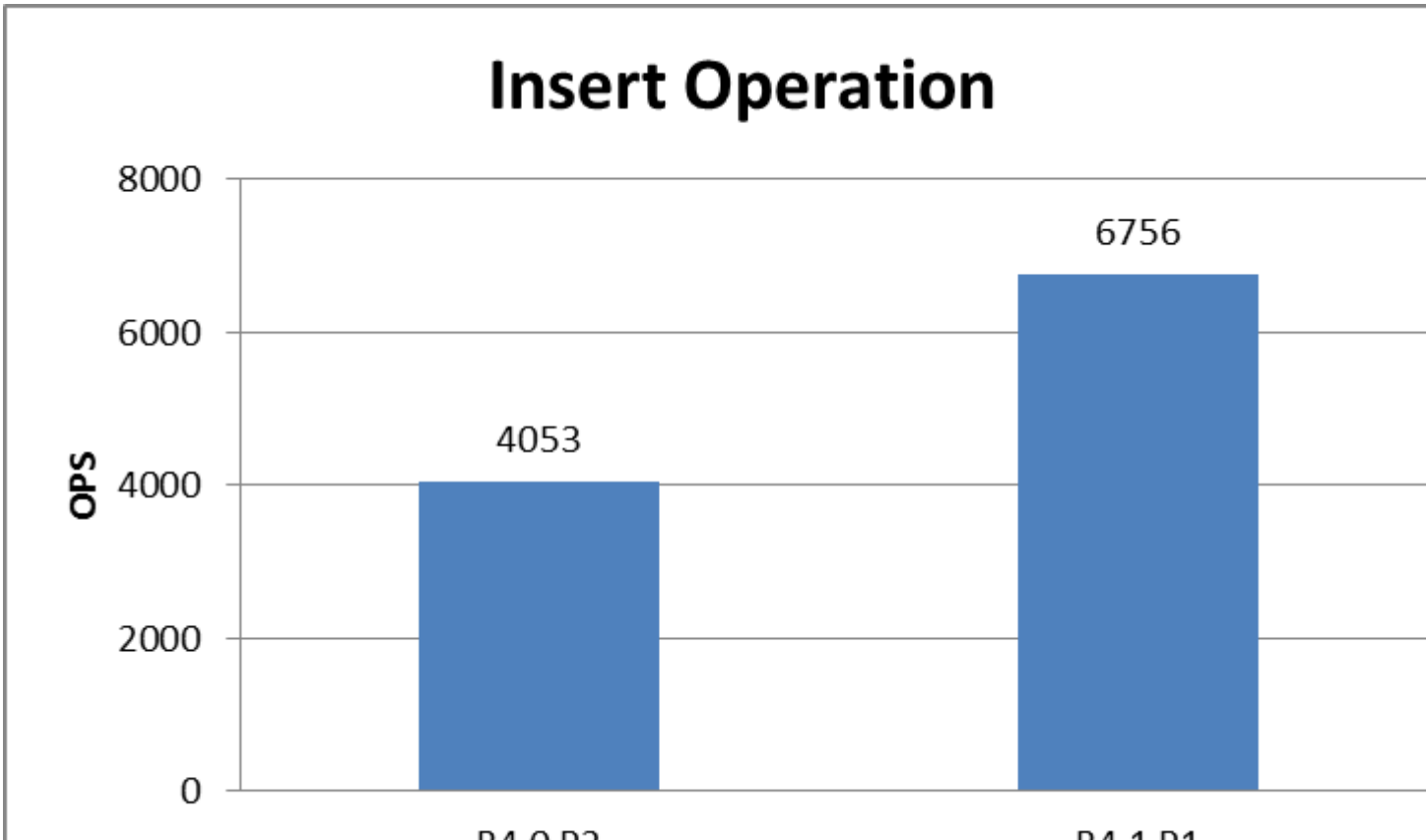
CUBRID 2008 R4.1 Patch 1은 특히, CUBRID 2008 R4.1에서 발견된 DB 복구 관련 오류를 포함하여, DB 엔진의 심각한 오류 및 CCI, JDBC의 오류들이 수정되었으므로, 2008 R4.1 사용자는 반드시 업그레이드해야 한다. 이외에도, PHP의 cubrid_query(), cubrid_execute() 함수를 개선하였으며, 응용 프로그램과 브로커 응용 서버(CAS) 사이의 메시지를 줄여, SELECT 위주의 워크로드에 대하여 이전 버전 대비 약 45%의 성능 향상을 보였다.

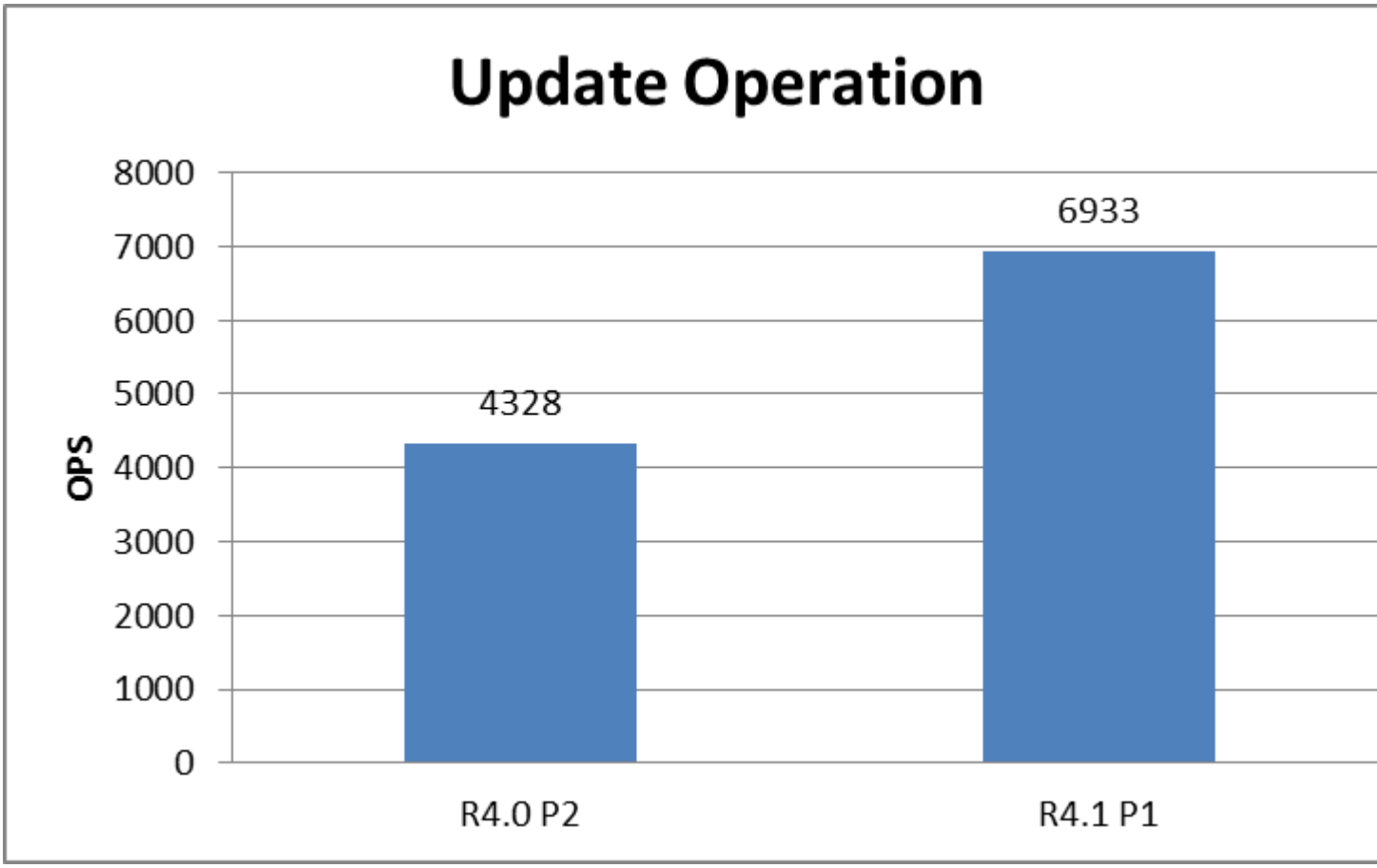
CUBRID 2008 R4.1 Patch 2 버전에서는 주로 서버 프로세스가 멈추거나(hang) 비정상 종료되는 오류 및 HA 안정성에 영향을 끼치는 오류들이 수정되었으므로, 이전 버전 사용자는 반드시 업그레이드해야 한다.

CUBRID 2008 R4.1 버전의 주요 특징은 다음과 같다.

대량의 트랜잭션 테스트에서 INSERT와 UPDATE 성능 70% 향상

트랜잭션 로그의 동시 처리, 메모리 버퍼의 디스크 쓰기, HA 복제 등에서 내부 구현을 개선했다. 그 결과, YCSB (https://github.com/brianfrankcooper/YCSB/wiki)에 기반한 대량의 트랜잭션 테스트 중 INSERT와 UPDATE에서 2008 R4.0 Patch 2 버전 대비 70%라는 매우 높은 성능 향상을 보였다. 성능 테스트와 관련한 자세한 내용은 "CUBRID 2008 R4.1 Patch1 QA Completion Report"를 참고한다. (아래 그림에서 Y축은 초당 질의 개수)





다양한 SQL 함수 및 정규 표현식 추가

ADDTIME, **ASCII**, **BIN**, **CONV**, **FIND_IN_SET**, **HEX** 등의 함수와 한 번의 호출로 여러 개의 시리얼 값을 얻을 수 있는 **SERIAL_NEXT_VALUE** 함수를 추가했다.

또한, **REGEXP** 연산자의 지원으로 정규 표현식을 이용한 검색이 가능해졌다.

브로커 파라미터 및 상태 정보 추가

브로커 파라미터 **MAX_QUERY_TIMEOUT**을 통해 프로그램의 질의 실행 시간을 제한할 수 있게 하고, **APPL_SERVER_MAX_SIZE_HARD_LIMIT**으로 브로커에서 CAS의 메모리 사용량이 초과되면 진행 중인 트랜잭션을 중단하고 CAS를 강제로 재시작할 수 있게 했다.

또한, **cubrid broker status** 유틸리티의 출력 정보에 브로커의 각 CAS ID에 대한 트랜잭션 시작 시간, 응용 프로그램 연결 회수와 재구동 회수를 추가했다.

50여 개의 발견된 버그 수정

SQL 함수, 인덱스, 질의문, DB 프로세스, HA 프로세스 등에서 발견된 50여 개의 버그를 수정했다.

자동 커밋 모드 기본값, IFNULL 등 NULL 조건 함수들의 반환 타입, 날짜 값으로 '0000-00-00' 지원 등의 동작 방식 변경

브로커 파라미터인 **CCI_DEFAULT_AUTOCOMMIT**의 기본값이 ON으로 변경되었다.

IFNULL, NVL, NVL2, COALESCE 함수들의 입력 인자 타입들이 서로 달라도 이를 수용하고 결과 타입으로 VARCHAR를 반환하게 되었다.

DATE, DATETIME, TIMESTAMP 타입에 '0000-00-00' 또는 '0000-00-00 00:00:00' 입력을 지원한다.

CUBRID 엔진, 사용 도구 및 드라이버에 대한 자세한 정보는 http://www.cubrid.org를 참고한다.

데이터베이스 호환성

서버 버전을 하위 버전에서 업그레이드하는 경우, DB의 마이그레이션 작업을 수행해야 한다. 보다 상세한 사항은 CUBRID 2008 R4.1로 업그레이드하는 방법을 참고한다.

CUBRID 2008 R4.1의 설치 방법

제품 설치

Linux용 설치 패키지는 바이너리를 포함하는 스크립트, tar.gz 압축 파일, Linux RPM 패키지 형태로 제공되며, 설치 방법은 매뉴얼의 'CUBRID 시작> 설치와 실행> Linux에서의 설치와 실행'을 참고한다.

Windows용 설치 패키지는 설치 마법사를 이용하여 설치할 수 있다. 설치 방법은 매뉴얼의 'CUBRID 시작> 설치와 실행> Windows에서의 설치와 실행'을 참고한다. 이와 함께 CUBRIDSUS-3267 Windows 환경에서 디렉터리 경로 설정 시 주의 사항을 참고한다.

CUBRID 환경 변수 및 OS 환경 변수 설정

CUBRID를 사용하기 위해서는 CUBRID 환경 변수와 관련 OS 환경 변수를 설정해야 한다. (매뉴얼의 'CUBRID 시작 > 환경 변수 설정 및 CUBRID 서비스 시작 > 환경 변수 설정' 참고) 특히, CUBRID 매니저와 Java 저장 프로시저를 사용하기 위해서는 Java 환경 변수를 설정해야 한다. (매뉴얼의 'CUBRID SQL 설명서 > Java 저장 함수/프로시저 > Java 저장 함수/프로시저 사용을 위한 환경 설정' 참고)

CUBRID 2008 R4.1로 업그레이드하는 방법

2008 R4.0에서 2008 R4.1로 업그레이드하기

CUBRID 2008 R4.0, 2008 R4.0 Patch X 버전 사용자는 2008 R4.1을 설치한 후 기존의 환경 설정 파일에서 파라미터들의 값을 다음과 같이 변경해야 한다.

cubrid.conf

thread_stacksize의 기본값이 100K에서 1M으로 변경되었으므로 이 값을 설정하지 않은 사용자는 CUBRID 관련 프로세스들의 메모리 사용량을 살펴볼 것을 권장한다.

data_buffer_size의 최소값이 64K에서 16M으로 변경되었으므로 이 값을 16M 미만으로 설정한 사용자는 16M 이상으로 설정해야 한다.

cubrid_broker.conf

CCI_DEFAULT_AUTOCOMMIT의 기본값이 ON으로 변경되었으므로 이를 설정하지 않은 응용 프로그램 사용자가 기존과 같은 자동 커밋 모드를 유지하고 싶다면 OFF로 설정해야 한다.

APPL_SERVER_MAX_SIZE_HARD_LIMIT의 최소값이 1024M이며, **APPL_SERVER_MAX_SIZE**의 값을 설정한 사용자는 **APPL_SERVER_MAX_SIZE_HARD_LIMIT**의 값을 이보다 크게 설정할 것을 권장한다(이 문서의 CUBRIDSUS-5956 CAS 프로세스의 메모리 사용량이 급격히 증가하는 경우를 차단하기 위한 방법 제공참고).

2008 R4.0 베타 이전 버전에서 2008 R4.1로 업그레이드하기

CUBRID 2008 R4.0 Beta 버전 및 그 이전 버전에서 업그레이드하는 사용자는 아래의 CUBRID 2008 R4.1로 업그레이드하는 방법을 참고하도록 한다.

업그레이드 주의 사항

기존 환경 설정 파일 보관

이전 버전의 $CUBRID/conf 디렉터리의 환경 설정 파일(cubrid.conf, cubrid_broker.conf, cm.conf)과 $CUBRID_DATABASES 디렉터리의 DB 위치 정보 파일(databases.txt)을 보관한다.[1]

새로 추가된 예약어 검사

CUBRID 설치 패키지에 포함 또는 http://ftp.cubrid.org에서 배포되는 CUBRID 2008 R4.1용 예약어 검출 스크립트인 check_reserved.sql을 이용하여 예약어 사용 여부를 검사할 수 있으며, 예약어로 지정된 식별자를 사용하고 있을 경우 식별자를 수정해야 한다. (매뉴얼의 'CUBRID SQL 설명서 > 식별자' 참고)

DB 마이그레이션

CUBRID 2008 R4.1은 2008 R4.0 Beta 및 그 이전 버전과 DB 볼륨이 호환되지 않으므로, 이전 DB를 CUBRID 2008 R4.1로 마이그레이션해야 한다. (아래의 DB 마이그레이션 절차 참고)

2008 R4.0 Beta에서 마이그레이션하기 위해서는 http://ftp.cubrid.org에서 제공하는 **migrate_r40beta2ga** 유틸리티를 사용할 수 있다.

CUBRID 2008 R3.1부터 GLO를 지원하지 않으며 LOB 타입이 GLO 기능을 대체하게 되었으므로, GLO를 이용한 응용 및 스키마는 LOB 타입에 맞게 수정해야 한다. (아래의 GLO 클래스 사용자의 마이그레이션 참고)

---

1 $CUBRID 또는 $CUBRID_DATABASES 형식의 환경변수는 Linux 환경에서 사용하는 형식이며 Windows 환경에서는 %CUBRID% 또는 %CUBRID_DATABASES%와 같은 형식으로 사용해야 한다.

2008 R3.1 및 그 이전 버전에서 마이그레이션하기 위해서는 **cubrid unloaddb/loaddb** 유틸리티를 사용한다.

복제 또는 HA 환경 재구성

2008 R4.0부터는 복제 기능을 더 이상 지원하지 않으므로, 이전의 복제 기능을 사용하는 시스템에서는 DB 마이그레이션 이후 HA 환경으로 재구성할 것을 권장한다. 또한, CUBRID 2008 R2.0, 2008 R2.1에서 제공된 Linux Heartbeat 기반의 HA 기능을 사용하는 시스템도 보다 안정적인 운영을 위해 DB 마이그레이션 이후 CUBRID Heartbeat 기반의 HA 환경으로 재구성해야 한다. (아래의 HA 환경에서 DB 마이그레이션 절차 참고)

HA 환경 구성은 매뉴얼의 '관리자 안내서 > CUBRID HA'를 참고하여 재설정해야 한다.

DB 마이그레이션 절차

2008 R3.x 및 그 이전 버전에서 2008 R4.1로 마이그레이션

2008 R3.x 및 그 이전 버전의 사용자는 아래의 표와 같이 2008 R4.1로 DB 마이그레이션을 해야한다.

또한, 기존의 GLO 클래스를 사용하고 있는 경우에는 추가 작업이 필요하다. (아래의 GLO 클래스 사용자의 마이그레이션 참고)

아래는 **cubrid unloaddb/loaddb** 유틸리티와 http://ftp.cubrid.org에서 별도 배포되는 check_reserved.sql 예약어 검출 스크립트를 이용하여 마이그레이션을 수행하는 방법이다. (매뉴얼의 '관리자 안내서 > 데이터베이스 마이그레이션'과 본 문서의 CUBRIDSUS-3826 GLO 클래스 지원 중단에 따른 주의 사항 참고)

단계	Linux 환경	Windows 환경
C1 단계: CUBRID Service 종료	% cubrid service stop	CUBRID Service Tray를 종료한다.
C2 단계: 예약어 검출 스크립트 실행	예약어 검출 스크립트가 위치하는 디렉터리에서 아래 명령을 실행한다. 검출 결과를 확인하여 마이그레이션 진행 또는 식별자 수정 작업을 진행한다. (허용되는 식별자는 관련 매뉴얼 참고) % csql -S -u dba -i check_reserved.sql testdb	
C3 단계: 이전 버전 DB 언로드	이전 버전의 databases.txt 및 conf 디렉터리 내 설정 파일을 별도 디렉터리에 보관한다. (C3a) cubrid unloaddb 유틸리티를 실행하고 이때 생성된 파일을 별도 디렉터리에 보관한다. (C3b) % cubrid unloaddb -S testdb 기존 DB를 삭제한다. (C3c) % cubrid deletedb testdb	
		이전 버전의 CUBRID를 제거한다.
C4 단계: 2008 R4.1 설치	본 문서에서 CUBRID 2008 R4.1의 설치 방법을 참고한다.	
C5 단계: DB 생성 및 데이터 로딩	DB를 생성할 디렉터리로 이동한 후, DB를 생성한다. (C5a) % cd $CUBRID/databases/testdb % cubrid createdb testdb (C3b)에서 보관한 파일을 가지고 cubrid loaddb 유틸리티를 실행한다. (C5b) % cubrid loaddb -s testdb_schema -d testdb_objects -i testdb_indexes testdb	

단계	Linux 환경	Windows 환경
C6 단계: 새 버전 DB 백업	% cubrid backupdb -S testdb	
C7 단계: CUBRID 환경 설정 및 CUBRID Service 구동	환경 설정 파일을 수정한다. 이때, (C3a)에서 보관한 이전 버전의 환경 설정 파일을 새 버전에 맞게 수정한다. (시스템 파라미터 설정은 주의 사항 및 관련 매뉴얼 참고) % cubrid service start % cubrid server start testdb	CUBRID Service Tray> [Service Start]를 선택하여 서비스를 시작한다. 명령 프롬프트 창에서 DB 서버를 구동한다. % cubrid server start testdb

2008 R4.0 Beta 버전에서 마이그레이션

2008 R4.0 Beta 버전 사용자는 아래의 표와 같이 DB 마이그레이션을 해야한다.

http://ftp.cubrid.org에서 제공되는 **migrate_r40beta2ga** 유틸리티를 이용하여 마이그레이션을 수행할 수 있다. 그러나 DB의 페이지 크기가 4K 미만인 볼륨은 **cubrid unloaddb/loaddb**를 이용하여 마이그레이션을 수행해야 한다. (2008 R4.0 베타 이전 버전에서 2008 R4.1로 업그레이드하기 참고)

마이그레이션 수행 중 실패하는 경우 백업해 놓았던 이전 버전의 DB를 복구(**cubrid restoredb**)하여 재시도한다.

단계	Linux 환경	Windows 환경
D1 단계: CUBRID Service 종료	% cubrid service stop	CUBRID Service Tray를 종료한다.
D2 단계: 이전 버전 DB 백업	이전 버전으로 복구하는 상황에 대비하기 위해 백업을 수행하고, 백업 파일을 별도 디렉터리(backup)에 보관한다. (D2a) % mkdir backup % cubrid backupdb -S -D backup testdb 이전 버전의 databases.txt 및 conf 디렉터리 내 설정 파일을 별도 디렉터리에 보관한다. (D2b)	
		이전 버전의 CUBRID를 제거한다.
D3 단계: 2008 R4.1 설치	본 문서의 CUBRID 2008 R4.1의 설치 방법을 참고한다.	
D4 단계: 마이그레이션 도구 실행	(D2b)에서 보관한 databases.txt를 2008 R4.1의 설치 디렉터리에 복사한다. (D4a) 아래와 같이 migrate_r40beta2ga 유틸리티를 실행한다. (D4b) % migrate_r40beta2ga testdb	
D5 단계: 새 버전 DB 백업	% cubrid backupdb -S testdb	
D6 단계: CUBRID 환경 설정 및 CUBRID Service 구동	보관한 이전 버전의 환경 설정 파일을 새 버전에 맞게 수정한다. (시스템 파라미터 설정은 주의 사항 및 관련 매뉴얼 참고) % cubrid service start % cubrid server start testdb	CUBRID Service Tray> [Service Start]를 선택하여 서비스를 시작한다. 명령 프롬프트 창에서 DB 서버를 구동한다. % cubrid server start testdb

GLO 클래스 사용자의 마이그레이션

GLO 클래스를 사용하는 경우, 2008 R3.1부터는 GLO 클래스를 지원하지 않으므로 BLOB 또는 CLOB 타입을 사용하도록 응용과 스키마를 변경해야 한다. 변경 작업이 용이하지 않다면 마이그레이션을 보류할 것을 권장한다.

HA 환경에서 DB 마이그레이션 절차

2008 R2.2 이상 버전에서 2008 R4.1로 HA 마이그레이션

2008 R2.0 또는 2008 R2.1에서 2008 R4.1로 HA 마이그레이션

2008 R2.2 이상 버전에서 2008 R4.1로 HA 마이그레이션

아래는 브로커, 마스터 DB, 슬레이브 DB를 각각 별도 서버에 구축한 환경에서 현재 서비스를 중지하고 업그레이드를 수행하기 위한 가이드이다. 서비스 무정지 업그레이드 시나리오는 별도 가이드 문서를 참고한다.

단계	설명
H1~H6 단계: 버전에 따라 마스터 노드에서 C1~C6 단계 또는 D1~D5 단계를 수행	마스터 노드에서 CUBRID 업그레이드 및 DB 마이그레이션을 수행하고, 2008 R4.1 DB를 백업한다.
H7 단계: 슬레이브 서버에 CUBRID 2008 R4.1 설치	슬레이브 서버에서 이전 버전의 DB는 삭제하고, 새 버전을 설치한다. 설치 방법은 본 문서의 CUBRID 2008 R4.1의 설치 방법을 참고한다.
H8 단계: 마스터 노드 백업본을 슬레이브 서버에서 복구	H6 단계에서 생성된 마스터 노드의 2008 R4.1 DB 백업본(예: testdb_bk*)을 슬레이브 서버에서 복구한다. % 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 단계: HA 환경 재구성 후 HA모드 구동	마스터 및 슬레이브 서버에서 CUBRID 환경 설정 파일(cubrid.conf) 및 HA 환경 설정 파일(cubrid_ha.conf)을 설정한다. (관련 매뉴얼 참고) % 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 마스터 및 슬레이브 서버에서 HA모드로 DB를 구동한다. (관련 매뉴얼 참고) % cubrid heartbeat start

단계	설명
H10 단계: 브로커 서버에 CUBRID 2008 R4.1 설치 및 브로커 구동	설치 방법은 본 문서의 CUBRID 2008 R4.1의 설치 방법을 참고한다. 브로커 서버에 있는 브로커를 시작한다. (관련 매뉴얼 참고) % cubrid broker start

2008 R2.0 또는 2008 R2.1에서 2008 R4.1로 HA 마이그레이션

CUBRID 2008 R2.0 또는 2008 R2.1의 HA 기능을 사용하는 경우, 서버 버전 업그레이드, DB 마이그레이션을 수행하고 HA 환경을 새롭게 구축한 후 해당 버전에서 사용되었던 Linux Heartbeat 자동 시작 설정을 변경해야 한다. (Linux Heartbeat 패키지가 불필요한 경우 삭제한다.)

위의 H1~H10 단계를 수행한 후, 아래의 H11 단계를 수행한다.

단계	설명
H11 단계: 기존 Linux heartbeat 자동 시작 설정 변경	이하의 작업은 마스터 및 슬레이브 서버에서 root 계정으로 수행한다. [root@master ~]# chkconfig --del heartbeat // 슬레이브 서버에서 동일 작업 수행

복제 기능 사용 시 주의 사항

CUBRID 2008 R4.1 버전부터 복제 기능이 제거되었으므로, 기존의 복제 기능을 사용 중인 환경에서 이중화 환경을 구축하려면 HA 기능을 사용해야 한다. 서버 버전 업그레이드 및 DB 마이그레이션을 수행한 후, HA 환경을 새롭게 구축할 수 있다. HA 환경 구축과 관련하여, 매뉴얼의 '관리자 안내서 > CUBRID HA'를 참고한다.

5.2 CUBRID 2008 R4.1 Patch 7에서 변경된 사항

추가된 기능

CUBRIDSUS-9262 DATASOURCE를 사용하는 CCI 응용 프로그램에서 prepared statement의 최대 값을 지정하는 속성 추가

DATASOURCE를 사용하는 CCI 응용 프로그램에서 statement pool에 유지할 prepared statement의 최대 값을 지정하는 max_open_prepared_statement 속성(property)을 추가했다. 기본값은 1000이다. 해당 값을 초과하면 가장 오래된 prepared statement는 해제되며 추후 재사용하면 다시 statement pool에 추가된다.

```
cci_property_set (properties, "max_open_prepared_statement", 1000);
```

수정된 사항

CUBRIDSUS-9286 EXISTS 조건과 FOR ORDERBY_NUM() BETWEEN 조건이 있는 질의에서 key limit 최적화가 잘못 적용되는 문제 수정

EXISTS 조건과 FOR ORDERBY_NUM() BETWEEN 조건이 있는 질의에서 적용되지 않아야 할 key limit 최적화가 적용되면서 잘못된 결과 건수를 출력하는 문제를 수정했다.

```
SELECT cd, tcd, nm
 FROM a
 WHERE EXISTS (SELECT 1 FROM b
               WHERE a.cd = b.cd
               AND (b.no = 10000 OR b.uno =10000))
 ORDER BY a.nm
 FOR ORDERBY_NUM() BETWEEN 1 AND 50;
```

CUBRIDSUS-9287 INSERT 문을 반복 수행하는 도중에 다른 트랜잭션에 의한 테이블 잠금 발생 시에 트랜잭션 일관성 보장되지 못하는 문제 수정

INSERT 문을 PREPARE 한 후에 반복적으로 수행하는 트랜잭션이 첫번째 실행 이후 커밋한 후에 다른 트랜잭션이 해당 테이블에 X-lock을 설정한 후에 다시 INSERT를 수행하게 될 때, 해당 테이블에 대한 IX-lock 부재로 인해 잠금 처리가 정확하게 되지 않아 결국 트랜잭션들의 일관성이 보장될 수 없는 문제를 수정했다.

CUBRIDSUS-9288 질의 계획 캐시에 설정한 제한 개수보다 많은 계획이 저장되는 현상 수정

시스템 파라미터 max_plan_cache_entries를 통해 질의 계획 캐시에 설정한 제한 개수보다 많은 계획이 저장되는 현상을 수정했다.

CUBRIDSUS-8435 UTF-8 문자셋 사용 시 LOWER("α") 결과의 문자가 깨지는 현상 수정

UTF-8 문자셋을 사용할 때 α와 같은 그리스 알파벳에 UPPPER, LOWER 등을 적용하면 비정상적인 문자를 출력하는 현상을 수정했다.

CUBRIDSUS-9430 JDBC 응용 프로그램에서 SELECT 리스트에 없는 칼럼으로 ORDER BY를 실행하면 잘못된 결과를 출력할 수 있는 문제 수정

JDBC 응용 프로그램에서 SELECT 리스트에 없는 칼럼으로 ORDER BY를 실행할 때, GROUP BY와 연관 부질의(correlated subquery)가 존재하면 잘못된 결과를 출력하는 문제를 수정했다.

```
SELECT (SELECT f1.a FROM foo f1 WHERE f1.b=f2.b) as ttt
FROM foo f2
WHERE f2.b >= 1 and f2.b < 10 GROUP BY f2.c ORDER BY f2.c;
```

CUBRIDSUS-9226 JDBC 응용 프로그램에서 Statement.close()를 수행하기 전에는 Resultset.close()를 수행해도 결과 셋의 메모리를 해제하지 않는 문제 수정

JDBC 응용 프로그램에서 Statement.close()를 수행하기 전까지는 Resultset.close()를 수행해도 결과 셋의 메모리를 해제하지 않는 문제로 인해 메모리 사용량이 증가할 수 있는 현상을 수정했다.

CUBRIDSUS-9251 DATASOURCE를 사용하는 CCI 응용 프로그램에서 statement의 요청 핸들을 닫아도 사용했던 결과 셋의 메모리를 해제하지 않는 문제 수정

DATASOURCE를 사용하는 CCI 응용 프로그램에서 statement의 요청 핸들을 닫아도 이전에 사용했던 statement를 재사용하기 전까지 결과 셋의 메모리를 해제하지 않는 문제를 수정했다.

CUBRIDSUS-9277 PHP 응용 프로그램에서 cubrid_connect_with_url 함수의 인자로 사용자 이름과 암호가 주어지지 않으면 public으로 접속되는 문제 수정

PHP 응용 프로그램에서 cubrid_connect_with_url 함수의 인자로 사용자 이름과 암호가 주어지지 않으면 연결 URL에 어떤 사용자가 주어졌음에도 불구하고 public 사용자로 접속되는 문제를 수정했다.

CUBRIDSUS-9278 cci_prepare_and_execute() 함수 실행 시 CAS가 재시작되면 오류가 발생할 수 있는 문제 수정

cci_prepare_and_execute() 함수 실행 시 CAS가 재시작되면 재시작된 CAS에 재접속하는 루틴이 존재하지 않아 "Cannot communcation with broker" 오류가 발생하는 문제를 수정했다.

CAS의 재시작은 CAS의 메모리 사용량이 증가할 때 이를 반환하기 위해 자동으로 발생한다.이 문제는 JDBC를 제외하고 PHP, ODBC, OLE DB 등 CCI 기반으로 개발된 드라이버들에도 영향을 미칠 수 있다.

CUBRIDSUS-9037 서버 재시작 이후에도 cci_prepare() 함수를 수행하면 정상 수행되지 않고 여전히 오류가 발생하는 문제 수정

데이터베이스 서버 종료로 인해 cci_prepare() 함수에서 오류가 발생한 이후 서버를 재시작해도 cci_prepare() 함수를 수행하면 정상 수행되지 않고 여전히 오류가 발생하는 문제를 수정했다.

CUBRIDSUS-9255 HA 환경에서 슬레이브 노드의 복제 재구축 후 복제 로그 반영 프로세스가 비정상 종료될 수 있는 문제 수정

HA 환경에서 슬레이브 노드의 복제 재구축 후 반영해야 할 보관 로그의 로그 페이지를 찾지 못하는 오류로 인해 applylogdb 프로세스가 비정상 종료될 수 있는 문제를 수정했다.

CUBRIDSUS-9367 64-bit Windows 버전에서 압축 백업 수행 시 오류 현상 수정

64-bit Windows 버전에서 **cubrid backupdb -z**으로 압축 백업 수행 시 "No error message available." 오류가 발생하는 현상을 수정했다.

CUBRIDSUS-9375 Windows용 CUBRID에서 2G 초과 파일을 포함하는 볼륨을 백업한 뒤 복구에 실패하는 문제 해결

Windows용 CUBRID에서 2G 초과 파일을 포함하는 볼륨을 백업하면 백업 볼륨이 잘못 생성되는 오류를 수정했다. 해당 백업 볼륨으로 복구를 시도하는 경우 "Restoredb cancelled or an error occurred." 오류 메시지와 함께 수행을 실패한다.

5.3 CUBRID 2008 R4.1 Patch 6에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-7938 검색 조건이 많으면 스캔 범위를 더 작게 하는 인덱스를 선택하도록 수정

검색 조건이 많은 경우 질의 계획에서 스캔 범위를 더 작게 하는 인덱스를 선택하도록 수정했다.

```
SELECT * FROM foo WHERE a = ? AND b = ? AND c = ? AND d = ? AND e = ? AND f = ? AND g = ? AND h
= ? ;
```

CUBRIDSUS-8524 CSQL 인터프리터의 질의 입력 파일 안에 ";", "," 혹은 "!"로 시작하는 라인이 질의의 문자열 값의 일부로 존재하면 세션 명령어로 잘못 인식하는 문제 수정

csql -i 옵션으로 수행할 질의의 입력 파일 안에 ";", "," 혹은 "!"로 시작하는 라인이 질의의 문자열 값의 일부로 존재하면 세션 명령어로 잘못 인식하는 문제를 수정했다.

```
// 이전 버전에서 아래 예와 같이 ";"이 오거나, 그대신 "," 또는 "!"가 오면 문제 발생.
INSERT INTO tbl VALUES('aaa
;
bbb');
```

CUBRIDSUS-8790 로깅 작업 중 치명적인 오류 발생 시 오류 로그 파일에 정보를 남기도록 수정

오류를 파일에 기록하는 로깅 작업 중 발생하는 치명적인 오류에 대해 관련 정보를 오류 로그 파일에 남기도록 수정했다.

CUBRIDSUS-8689 로그 페이지 크기보다 큰 로그 레코드를 쓸 때 서버 프로세스가 비정상 종료될 수 있는 현상 수정

로그 페이지 크기보다 큰 로그 레코드를 쓸 때 서버 프로세스가 비정상 종료될 수 있는 현상을 수정했다.

이 현상은 로그 레코드가 로그 페이지 크기(기본값 16K)보다 클 때 발생할 수 있다. DELETE 질의가 수행될 때 발생할 확률은 매우 낮고, 인덱스 키의 크기가 로그 페이지 크기보다 크고 HA를 사용할 경우 발생 가능성이 매우 높다. 수정 이전 버전에서는 이 현상으로 인해 서버가 비정상 종료된 후에도 정상적으로 재구동되었다.

CUBRIDSUS-8816 HA 환경에서 인덱스의 키를 로그 페이지 크기보다 크게 하는 경우 복제 불일치가 발생할 수 있는 현상 수정

HA 환경에서 인덱스의 키를 로그 페이지 크기보다 크게 하는 경우 마스터 로그 파일이 잘못 작성되어 복제 불일치가 발생할 수 있는 현상을 수정했다.

5.4 CUBRID 2008 R4.1 Patch 5에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-8290 CONNECT BY 절에 의한 계층 질의 수행 중 서버 프로세스가 비정상 종료될 수 있는 현상 수정

CONNECT BY 절에 의한 계층 질의 수행 중 질의 취소(query cancel) 오류가 발생하면 서버 프로세스가 비정상 종료될 수 있는 현상을 수정했다.

CUBRIDSUS-8074 CSQL 인터프리터에서 -i 옵션을 이용하여 파일로 입력한 질의문이 두 라인 이상으로 나누어지고 다음 라인의 첫 번째 문자에 ! 혹은 ;가 있으면 수행에 실패할 수 있는 문제 수정

CSQL 인터프리터에서 -i 옵션을 이용하여 파일로 입력한 질의문이 두 라인 이상으로 나누어지고 다음 라인의 첫 번째 문자에 ! 혹은 ;가 있으면 "ERROR: No such session command." 오류가 발생하는 문제를 수정했다.

```
INSERT INTO tbl VALUES ('aaa !');
```

CUBRIDSUS-8409 DB 생성 직후 백업한 볼륨에 -d 옵션을 사용하여 복구 시도 시 실패하는 문제 수정

DB 생성 직후 **cubrid backupdb**를 수행하여 백업한 볼륨에 **-d** 옵션으로 백업 시간을 지정하여 **cubrid restoredb**를 실행하면 실패하는 문제를 수정했다.

```
cubrid backupdb -S demodb
csql -S demodb -c "create table x"
cubrid restoredb -d backuptime demodb
FATAL ERROR ***
No error message available.
Please consult error_log file = /home1/user1/CUBRID/log/demodb_restoredb.err for additional
information
... ABORT/EXIT IMMEDIATELY ...<<<---
```

CUBRIDSUS-8408 브로커 파라미터의 SLOW_LOG 설정을 동적으로 변경하면 각 CAS의 SQL_LOG 설정이 변경되는 문제 수정

broker_changer로 SLOW_LOG 설정을 동적으로 변경하면 각 CAS의 SQL_LOG 설정이 변경되는 문제를 수정했다.

```
broker_changer query_editor SLOW_LOG ON
```

CUBRIDSUS-8124 HA 환경에서 복제 로그 파일이 삭제되지 않는 현상 수정

HA 환경에서 백업이 수행되는 노드의 맞은 편 노드(보통 슬레이브 노드에서 백업을 진행하므로 마스터 노드)에 있는 복제 로그 파일이 삭제되지 않는 현상을 수정했다.

5.5 CUBRID 2008 R4.1 Patch 4에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-8078 브로커 응용 서버 프로세스(CAS)의 통신 큐에 abort 메시지가 남게 되면 CAS가 비정상 종료될 수 있는 문제 수정

DB 서버 프로세스(cub_server)에서 CAS에 보낸 abort 메시지가 통신 큐에 남게 되면 CAS가 비정상 종료될 수 있는 문제를 수정했다. 참고로, abort 메시지는 HA에서 슬레이브 노드에 INSERT/UPDATE/DELETE와 같은 데이터 변경 작업을 방지하는 경우나 통신 프로토콜이 깨지는 경우 등에 발생할 수 있다.

5.6 CUBRID 2008 R4.1 Patch 3에서 변경된 사항

개선된 사항

CUBRIDSUS-7811 cubrid loaddb 유틸리티 실행 시 -t 옵션으로 테이블 이름을 주어 데이터 로딩이 가능하도록 개선

cubrid loaddb 유틸리티 실행 시 데이터 파일에 테이블 이름이 없더라도 **-t** 옵션에 테이블 이름을 주면 데이터 로딩이 가능하도록 개선했다.

```
cubrid loaddb -t tbl_name -d records_file testdb
```

수정된 사항

CUBRIDSUS-7780 인덱스 키 컬럼에 대해 OR조건들이 호스트 변수로 주어지고, '=' 조건이 존재하면, 잘못된 결과를 출력할 수 있는 문제 수정

인덱스 컬럼에 대해 OR조건들이 호스트 변수로 주어지고, '=' 조건이 존재하면, 조건들을 병합(merge)하는 과정의 오류로 인해 잘못된 결과를 출력 할 수 있는 문제를 수정했다.

```
PREPARE s FROM 'SELECT count( * ) FROM tbl WHERE col < ? OR col = ?';
```

CUBRIDSUS-7347 prepare된 SQL문의 호스트 변수에 NULL을 바인딩하고 나면 다음 번 수행 시 데이터를 잘못 해석하는 문제 수정

prepare된 SQL문의 호스트 변수에 NULL을 바인딩하고 나면 다음 번 수행 시 precision이나 scale을 잘못 계산하는 등, 데이터를 잘못 해석하는 문제를 수정했다.

```
// 수정 이전 버전에서 NUMBER(20,6)인 num 컬럼에 NUMERIC 11을 바인딩하면 0.000011로 입력되는
현상이 나타남
UPDATE tbl SET num = ? WHERE aa = ?
```

CUBRIDSUS-7157 WHERE 절에서 인덱스의 키 조건이 OR로 연결되고, 조건을 만족하는 범위가 키 전체이면 오류가 발생하는 문제 수정

WHERE 절에서 인덱스의 키 조건이 OR로 연결되고, 조건을 만족하는 범위가 키 전체이면 "Invalid XASL tree node content" 오류가 발생하는 문제를 수정했다.

```
// 수정 이전 버전에서 다음과 같이 OR 조건이 있고 a 컬럼에 대해 인덱스 스캔하면 오류가 발생함
CREATE TABLE t1(a INT PRIMARY KEY);
INSERT INTO t1(a) VALUES (1),(2),(3),(4),(5),(6),(7),(8),(9),(10)
PREPARE st FROM 'SELECT * FROM t1 WHERE a>? or a<?';
EXECUTE st USING 1,2;
```

CUBRIDSUS-6027 다중 컬럼 인덱스에 DESC 컬럼이 존재하고 OR 조건이 있을때 조건에 따라 질의 수행이 실패하는 문제 수정

다중 컬럼 인덱스에 DESC 컬럼이 존재하고 인덱스 전체 키가 아닌 부분 키에 대한 OR 조건들이 주어지면, 질의 실행 과정에서 "ERROR: Invalid XASL tree node content." 오류를 출력하면서 실패하는 문제를 수정했다.

```
CREATE TABLE tab0(pk INTEGER PRIMARY KEY, col0 INTEGER, col1 FLOAT);
CREATE INDEX idx_tab0_0 ON tab0 (col0 DESC,col1);
SELECT pk FROM tab0 WHERE (col0>1 OR col0<1);
```

CUBRIDSUS-8041 JDBC에서 Connection 시도가 connectTimeout에 의해 중단될 경우 브로커의 Job Queue가 쌓이는 현상 수정

JDBC에서 Connection 시도가 connectTimeout에 의해 중단된 이후 네트워크 소켓이 닫히지 않은 상태로 계속 재접속을 시도할 경우 브로커의 Job Queue가 쌓이는 문제를 수정했다.

5.7 CUBRID 2008 R4.1 Patch 2에서 변경된 사항

새로 추가된 기능

CUBRIDSUS-6706 내림차순 인덱스 스캔을 사용하지 않도록 하는 SQL 힌트 추가

내림차순 인덱스 스캔을 사용하지 않도록 하는 SQL 힌트인 NO_DESC_IDX를 추가했다.

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

CUBRIDSUS-6829 백업 수행으로 인한 디스크 I/O 부담을 줄이기 위해 백업 중간에 쉬게 하는 기능 추가

cubrid backupdb 수행 시 이로 인한 디스크 I/O 부담을 줄이기 위해 1MB의 파일을 읽을 때마다 중간에 지정한 시간만큼 쉬게 하는 옵션인 **--sleep-msecs**(단위: 밀리 초)를 추가했다.

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

개선된 사항

CUBRIDSUS-5940 HA 환경에서 레플리카 노드 구축이 용이하도록 복제 재구축 스크립트 개선

이전 버전에서는 슬레이브 노드에 대해서만 복제 재구축 스크립트(**ha_make_slavedb.sh**)를 사용할 수 있었으나, 레플리카 노드도 이 스크립트를 이용할 수 있도록 **ha_make_slavedb.sh**를 개선했다.

CUBRIDSUS-7043 다중 컬럼 인덱스의 key값에 NULL이 포함되어 있는 경우 불필요한 스캔이 수행되는 문제 개선

다중 컬럼 인덱스의 key값에 NULL이 포함되어 있으면 불필요한 데이터를 스캔하여 질의 부하량에 비해 CPU 사용률이 높아지는 문제를 개선했다.

```
CREATE INDEX i_tbl ON tbl (id1, id2, flag);
-- flag에 NULL 값이 포함되어 있음
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 INSERT INTO ... SELECT ... ORDER BY 구문 수행 시 인덱스가 있어도 "SKIP ORDER BY"가 되지 않는 문제 개선

INSERT INTO ... SELECT ... ORDER BY 구문 수행 시 인덱스에 의해 이미 정렬이 된 상태임에도 불구하고 ORDER BY를 추가적으로 수행하는 문제를 개선했다.

```
INSERT INTO tbl SELECT id, name FROM tbl WHERE id > 10000 ORDER BY id ASC;
```

CUBRIDSUS-6781 브로커 로그에 CCI 연결 URL을 사용자가 입력한대로 출력하도록 수정

브로커 로그에 CCI 연결 URL을 사용자가 입력한대로 출력하도록 수정했다. 수정 이전 버전에서는 아래 예와 같이 입력 URL에 있는 althosts 값을 출력하지 않았다.

```
입력 URL=cci:cubrid:10.0.10.89: 30000:cub01:::?althosts=10.0.10. 90:30000&rctime=60
수정 이전 브로커 로그 출력 URL=cci:cubrid:10.24.159.89:30000:cub01:::
수정 이후 브로커 로그 출력 URL= 입력 URL과 같음
```

수정된 사항

CUBRIDSUS-7001 오버플로우 키를 INSERT하고 롤백하면 오류가 발생할 수 있는 문제 수정

키의 크기가 커서 여러 페이지로 나뉘어 저장되는 오버플로우 키를 INSERT하고 롤백하면 메모리 참조 오류가 발생할 수 있는 문제를 수정했다.

```
CREATE TABLE tbl(id char (16000));
COMMIT;
INSERT INTO tbl VALUES ('x');
ROLLBACK;
```

CUBRIDSUS-7104 기본 키로 선언된 VARCHAR 타입 컬럼의 크기를 이전보다 크게 바꾸면 NOT NULL 제약 조건이 사라지는 문제 수정

"ALTER TABLE ~ CHANGE col"을 실행하여 기본 키로 선언된 VARCHAR 타입 컬럼의 크기를 이전보다 크게 바꾸면 NOT NULL 제약 조건이 사라지는 문제를 수정했다.

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

CUBRIDSUS-7106 HEX 함수에서 음수 입력 시 반환하는 결과 타입의 크기 정보가 잘못될 수 있는 오류 수정

HEX 함수에서 음수 입력 시 반환하는 결과 타입의 크기 정보(precision)가 잘못될 수 있는 오류를 수정했다.

```
stmt.executeUpdate("INSERT INTO tbl VALUES (-999999999999999)");
ResultSet r = stmt.executeQuery("SELECT HEX(N) FROM tbl");
ResultSetMetaData m = r.getMetaData();
m.getPrecision() // 이전 버전에서 잘못된 정보 가져옴
```

CUBRIDSUS-7217 서버 프로세스 비정상 종료 이후 재시작 도중 서버 프로세스가 멈추는 현상 수정

장시간 트랜잭션 수행 중에 서버 프로세스가 비정상 종료된 이후 재시작 과정 중에 멈추는(hang) 현상을 수정했다.

CUBRIDSUS-6991 서버 프로세스의 비정상 종료 이후 정상 복구되지 않는 문제 수정

서버 프로세스의 비정상 종료 이후 정상적으로 데이터베이스가 복구되지 않을 수 있는 문제를 수정했다.

CUBRIDSUS-7350 많은 트랜잭션들이 잠금을 획득하고 있는 상태에서 교착 상태가 발생하면 서버 프로세스가 비정상 종료되는 현상 수정

많은 트랜잭션들이 잠금을 획득하고 있는 상태에서 교착 상태(deadlock)가 발생하면 서버 프로세스가 비정상 종료되는 현상을 수정했다.

CUBRIDSUS-7216 자동 볼륨 추가 도중 실패한 경우 조치 후에도 질의 수행에 실패하는 등의 문제 수정

자동 볼륨 추가 도중 서버 프로세스가 비정상 종료되어 자동 볼륨 추가에 실패한 중간 파일이 서버 재시작 등의 조치 후에도 남아있어서 볼륨 추가가 안되고 질의 수행에 실패하는 등의 문제를 수정했다.

CUBRIDSUS-6933 계층 질의 수행 시 서버 프로세스가 비정상 종료될 수 있는 문제 수정

매우 큰 크기의 문자열을 지닌 컬럼에 대해 계층 질의 수행 시 서버 프로세스가 비정상 종료될 수 있는 문제를 수정했다.

CUBRIDSUS-7075 체크포인트 수행 시 서버 프로세스가 비정상 종료될 수 있는 문제 수정

체크포인트 수행 시 서버 프로세스가 비정상 종료될 수 있는 문제를 수정했다.

CUBRIDSUS-6811 여러 트랜잭션에서 동시에 DML 질의를 수행할 때 "Unknown key" 오류와 함께 서버 프로세스가 비정상 종료되는 문제 수정

여러 트랜잭션에서 동시에 DML 질의를 수행할 때 B+tree 인덱스와 관련하여 "Unknown key" 오류가 발생하면서 서버 프로세스가 비정상 종료되는 현상을 수정했다.

CUBRIDSUS-6946 다수의 DML 질의 수행이 같은 페이지를 변경하는 도중 롤백에 실패하는 문제 수정

하나의 페이지를 여러 트랜잭션이 동시에 입력, 삭제 또는 레코드 길이가 변경되도록 갱신하는 도중롤백을 수행하면 "Internal system failure: no more specific information is available." 오류가 발생하면서 롤백에 실패하는 문제를 수정했다.

CUBRIDSUS-7047 CCI 기반 응용 프로그램이 CAS와 DB 서버 간 연결이 끊어짐을 인지하지 못하는 문제 수정

CCI 기반 응용 프로그램[2](이하 A) - CAS(이하 B) - DB 서버 프로세스(이하 C)의 연결에서 B-C 연결이 끊어졌다가 재연결되는 경우, A가 이를 인지하지 못하여 C와 끊어진 기존의 A-B 연결을 유지함으로 인해 DB에 재연결하지 못하는 문제를 수정했다.

CUBRIDSUS-6998 CCI 기반 응용 프로그램이 DB 연결에 실패한 후 재연결 시도 시 해제된 메모리를 참조하는 오류 수정

CCI 기반 응용 프로그램이 cci_connect()를 사용하여 DB 연결에 실패한 후 재연결 시도 시 해제된 메모리를 참조하는 오류를 수정했다.

CUBRIDSUS-6679 인덱스 스캔하는 질의 수행 중 CAS가 비정상 종료될 수 있는 문제 수정

인덱스 스캔하는 질의 수행 중 통계 정보에 대한 메모리를 잘못 참조하는 문제로 인해 CAS가 비정상 종료될 수 있는 문제를 수정했다.

CUBRIDSUS-6788 CAS에서 CHANGE CLIENT가 발생할 때 새로 연결한 응용 클라이언트의 IP가 0.0.0.0으로 설정되는 문제 수정

CAS에서 CHANGE CLIENT가 발생할 때 새로 연결한 응용 클라이언트 IP가 0.0.0.0으로 설정되는 문제를 수정했다. 이전 버전에서 이 문제 발생 시 **cubrid broker status**를 통해 나타나는 응용 클라이언트 IP는 0.0.0.0으로 보이게 되고, 응용 클라이언트가 해당 CAS에 대해 query cancel을 요청하면 실패한다.

CUBRIDSUS-6781 cci_connect_with_url()에서 사용자/암호가 URL과 함수 인자로 동시에 주어진 경우 함수 인자를 사용하도록 수정

cci_connect_with_url()에서 사용자/암호가 URL과 함수 인자로 동시에 주어진 경우 함수 인자를 사용하도록 수정했다. 아래 예는 사용자와 암호로 "usr1"과 "pwd123"을 사용한다.

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

CUBRIDSUS-6338 CSQL 인터프리터에서 테이블 통계 정보 출력 시 타입 정보를 잘못 출력하는 오류 수정

CSQL 인터프리터에서 테이블 통계 정보 출력 시 smallint 타입을 UNKNOWN_TYPE으로 잘못 출력하는 오류를 수정했다.

---

2 CCI 인터페이스를 이용한 응용 프로그램 또는 CCI를 기반으로 개발된 드라이버들을 이용한 응용 프로그램. PHP, ODBC, OLE DB, Perl, Ruby, Python 등은 CCI를 기반으로 개발된 드라이버들이다. 따라서 CCI 기반 드라이버들을 사용하려면 CCI 동적 링크 라이브러리가 필요하다. 참고로, JDBC는 CCI 기반이 아니다.

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

CUBRIDSUS-7183 핑 노드가 설정된 HA 환경에서 슬레이브 서버와 핑 노드 모두 연결 불가능한 상황이 발생하면, 상황 복구 이후에도 응용 프로그램이 마스터 노드에 연결할 수 없는 현상 수정

시스템 파라미터 **ha_ping_hosts**에 의해 핑 노드가 설정된 HA 환경에서 슬레이브 노드와 핑 노드 모두 마스터 노드와 연결 불가능한 상황이 발생하면, 연결이 가능하게 복구한 이후에도 응용 프로그램이 마스터 노드에 연결할 수 없는 현상을 수정했다. 핑 노드가 존재하는 HA 구성에서 마스터 노드는 핑 노드로 연결되지 않으면 마스터의 역할을 유지할 수 없기 때문에 슬레이브로 역할 변경을 하게 되고, 응용 프로그램은 읽기 전용으로만 사용할 수 있게 된다.

CUBRIDSUS-6956 HA 환경에서 복제 반영 도중 특정 오류 발생 시 재시도하도록 수정

HA 환경에서 복제 반영 도중 특정 오류 발생 시 해당 작업을 재시도하도록 수정했다.

이와 함께, ha_applylogdb_ignore_error_list 파라미터와 ha_applylogdb_retry_error_list 파라미터를 추가하여 ha_applylogdb_ignore_error_list 파라미터에 설정된 에러가 발생하여도 이를 무시하고 복제 반영을 진행하고, ha_applylogdb_retry_error_list 파라미터에 설정된 에러가 발생하면 해당 작업을 반복적으로 재시도하도록 수정했다.

CUBRIDSUS-6918 HA 환경에서 기본 키로 지정된 단일 컬럼에 대해 내림차순 인덱스로 스캔하면 데이터가 출력되지 않는 현상 수정

HA 환경에서 기본 키로 지정된 단일 컬럼에 내림차순 인덱스를 추가하여 데이터 입력 후 해당 인덱스로 스캔하면, 데이터가 출력되지 않는 현상을 수정했다.

```
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 HA 환경에서 copylogdb가 이미 삭제된 오래된 보관 로그를 요청할 때 서버 프로세스가 비정상 종료되는 문제 수정

시스템 파라미터 **force_remove_archives**의 값이 yes로 설정된 경우, 서버 프로세스(**cub_server**)는 **copylogdb**가 복사하지 않은 트랜잭션 로그를 지울 수 있는데, 이후 **copylogdb**가 이 트랜잭션 로그를 요청할 때 서버 프로세스가 비정상 종료되는 문제를 수정했다. 이 경우 서버 프로세스는 **copylogdb**에 에러를 리턴하게 된다.

CUBRIDSUS-7163 HA 환경에서 복제 지연 중 failover 시 레플리카 노드에 발생하는 복제 불일치 현상 수정

HA 환경에서 레플리카 노드에는 마스터 노드의 복제 로그를 반영하는 **applylogdb** 프로세스와 슬레이브 노드의 복제 로그를 반영하는 **applylogdb** 프로세스가 각각 존재한다. 복제 지연이 있는 상태에서 failover가 발생하면, 두 개

의 **applylogdb**가 동시에 복제 로그를 반영하면서 복제 불일치가 발생할 수 있는 문제를 수정했다.

CUBRIDSUS-7164 HA 환경에서 마스터 서버 프로세스 정상 종료 시 그 이전에 생성한 모든 로그를 복사하도록 수정

HA 환경에서 마스터 서버 프로세스를 정상 종료하면 HA 로그 복제 모드가 ASYNC인 경우에도 프로세스 종료 이전에 생성한 모든 로그를 복사하도록 수정했다.

참고로, 수정된 버전에서 복제 지연으로 인해 복사할 로그 량이 많아 서비스 정상 종료 시간이 1분을 초과하는 경우 시스템 내부에서 서버 프로세스를 강제 종료하므로, 서비스가 재시작되는 시간이 DB 복구로 인해 길어질 수 있다.

CUBRIDSUS-6535 HA 환경에서 'ALTER TABLE... ADD COLUMN 예약어' 구문이 복제되지 않는 문제 수정

HA 환경에서 컬럼 이름이 예약어인 경우 ALTER TABLE 구문이 복제되지 않는 문제를 수정했다.

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

CUBRIDSUS-7387 HA 환경에서 로그 페이지 ID 값이 32비트 정수형의 최대값을 초과하면 applylogdb가 없는 로그 페이지를 무한히 반복하여 읽으려는 문제 수정

HA 환경에서 로그 페이지 ID 값이 32비트 정수형의 최대값을 초과하면 **applylogdb**가 존재하지 않는 로그 페이지를 무한히 반복하여 읽으려는 문제를 수정했다.

5.8 CUBRID 2008 R4.1 Patch 1에서 변경된 사항

새로 추가된 기능

CUBRIDSUS-6070 JDBC 연결 URL에 디버깅용 로그를 기록하도록 설정하는 기능 추가

JDBC 연결 URL에 **logOnException** 프로퍼티로 예외 처리를, **logSlowQueries** 프로퍼티와 **slowQueryThresholdMillis** 프로퍼티로 슬로우 쿼리를 로그 파일에 기록하도록 설정하는 기능을 추가했다.

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

개선된 사항

CUBRIDSUS-6648 CCI, PHP 인터페이스에서 한번의 호출로 PREPARE와 EXECUTE를 수행할 수 있게 개선하고, 요청 핸들 종료 함수 호출 시 메시지 전송횟수를 줄이도록 수정

CCI와 PHP에서 한 번의 호출로 PREPARE와 EXECUTE를 수행하도록 cci_prepare_and_execute()를 추가하고, PHP 인터페이스의 cubrid_query(), cubrid_execute()가 이를 사용하게 했다. 단, cubrid_execute()는 입력인자로 연결 객체와 질의문을 전달하는 경우에만 cci_prepare_and_execute()를 사용한다.

이와 함께, cci_close_req_handle()을 호출할 경우 종료 요청을 브로커 응용서버(CAS)에게 보내지 않고 다음 번 PREPARE시에 종료할 요청 핸들(request handle) 번호를 같이 보내줌으로써 메세지 전송 횟수를 줄이도록 수정했다. 또한, 이 수정으로 인해 자동 커밋 모드가 ON일 때 CAS가 재시작된 후 cci_close_req_handle()를 호출하면 -4 (COMMUNICATION ERROR) 오류를 반환하는 현상이 해결되었다.

SELECT 위주의 워크로드에 대하여 이전 버전 대비 약 45%의 성능 향상을 보였다.

CUBRIDSUS-6906 JDBC 드라이버의 성능 개선

JDBC 드라이버에서 헤더 정보를 버퍼에 모아서 한번에 전송하도록 수정하여 2008 R4.1 버전 JDBC 드라이버의 성능 저하를 개선했다.

수정된 사항

CUBRIDSUS-6678 서버 프로세스 재시작 시 DB 볼륨이 정상적으로 복구되지 못하는 오류 수정

서버 프로세스의 비정상 종료를 포함하여 종료 후 재시작 시에 DB 복구(recovery)가 비정상적으로 수행되어 DB 볼륨이 손상될 수 있는 오류를 수정했다.

CUBRIDSUS-6881 IFNULL/NVL/NVL2/COALESCE 함수에 호스트 변수를 사용하여 INSERT 수행 시 발생하는 오류 수정

IFNULL/NVL/NVL2/COALESCE 함수에 호스트 변수를 사용하여 INSERT 문을 수행하는 경우 "ERROR: Execute: Cannot evaluate ' ifnull(?:0 , 1)'" 오류가 발생하였으나 이를 수정했다.

CUBRIDSUS-6689 NOT 뒤에 단항 연산자 사용 시 'unknown opcode' 오류 수정

NOT 뒤에 단항(unary) 연산자를 사용하는 경우 "ERROR: 'unknown opcode' operator is not defined on types integer and integer." 오류가 발생하였으나 이를 수정했다.

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

CUBRIDSUS-6778 인덱스 스캔 도중 키 잠금을 획득/해제하는 과정에서 라이브락 현상이 발생할 수 있는 문제 수정

인덱스 스캔 도중 잠금(lock)을 해제했다가 획득하려는 시도를 무한 반복하는 라이브락(livelock) 현상이 발생할 수 있는 문제를 수정했다.

CUBRIDSUS-6914 CCI 응용이 DATASOURCE를 이용하여 DB에 연결한 상태에서 연결 종료 함수를 호출하는 경우 응용 프로그램이 비정상 종료될 수 있는 현상 수정

cci_datasource_create() 함수로 DATASOURCE를 생성하여 DB에 연결한 상태에서 **cci_disconnect**()를 호출하는 경우 응용 프로그램이 비정상 종료될 수 있는 현상을 수정했다.

CUBRIDSUS-6925 cci_set_query_timeout()로 설정한 질의 타임아웃 동작 오류 수정

cci_set_query_timeout() 함수로 설정한 질의 타임아웃이 정상적으로 동작하지 않는 문제를 수정했다.

CUBRIDSUS-6958 CCI, PHP 응용 프로그램에서 DB 연결 실패 시 failback이 되어도 첫 번째로 설정한 호스트에는 연결을 시도하지 않는 문제 수정

CCI 기반 응용 프로그램에서 DB 연결에 실패하면 연결 URL의 **althost** 앞에 설정한 첫 번째 호스트에는 연결을 시도하지 않는 문제로 인해, failback이 되어도 마스터 노드로 재연결하지 않는 현상이 존재하였으나 이 문제를 수정했다.

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

CUBRIDSUS-6753 인덱스에서 B+트리 노드의 분할/병합과 INSERT/DELETE/UPDATE가 동시에 일어날 때 "Unknown key" 오류 수정

인덱스에서 B+트리 노드의 분할/병합과 INSERT/DELETE/UPDATE가 동시에 일어날 때 "Unknown key '???' referenced in B+tree index {vfid: (40, 3), rt_pgid: 2520, key_type: character varying}" 오류가 발생하였으나 이를 수정했다.

CUBRIDSUS-6897 CCI 기반 응용이 두 개 이상의 브로커에 연결하는 환경에서, 어떤 질의가 5초 이상 수행되면 연결 오류가 발생할 수 있는 문제 수정

CCI 기반 응용 프로그램에서 여러 개의 쓰레드들이 두 개 이상의 브로커에 연결 할 때 적어도 하나의 쓰레드에서 질의가 5초 이상 수행되면 -4(COMMUNICATION ERROR) 오류가 발생할 수 있는 문제를 수정했다. 해당 오류 발생 시 응용 프로그램이 비정상 종료되거나 응용 프로그램에 의해 1G 이상 메모리가 불필요하게 할당되는 등의 이상 현상이 존재했다.

CUBRIDSUS-6779 응용 프로그램과 CAS 연결 직후 둘 사이의 상태 정보가 일치하지 않는 현상 수정

응용 프로그램이 CAS에 연결한 직후 **cubrid broker status** 명령으로 상태 정보를 확인하는 경우 트랜잭션 상태 정보(STATUS)가 CLOSE WAIT로 바뀌지 않고 CLIENT_WAIT인 현상을 수정했다.

CUBRIDSUS-6754 CAS의 개수를 100으로 설정하는 경우 브로커 구동에 실패하는 문제 수정

CAS의 개수를 설정하는 브로커 파라미터인 **MIN_NUM_APPL_SERVER**와 **MAX_NUM_APPL_SERVER**의 값을 100으로 설정하는 경우 브로커 구동에 실패하는 문제를 수정했다.

CUBRIDSUS-6364 CAS가 SQL 로그를 쓰는 도중 멈출(hang) 수 있는 현상 수정

CAS가 SQL 로그를 쓰는 도중 인터럽트 발생했을 때 멈출(hang) 수 있는 현상을 수정했다.

CUBRIDSUS-6889 HA 복제 반영 프로세스가 복제를 진행하지 못할 수 있는 문제 수정

HA 복제 반영 프로세스인 **applylogdb**에서 내부 오류로 인해 복제 반영을 정상적으로 진행하지 못하는 오류를 수정했다.

CUBRIDSUS-6905 JDBC에서 집합형 데이터 타입을 사용하면 CAS와 JDBC 응용 프로그램이 멈추는(hang) 현상 수정

JDBC에서 SET, MULTISET, LIST 등 집합형 데이터 타입을 사용하면 CAS와 JDBC 응용 프로그램이 멈추는(hang) 현상을 수정했다.

5.9 CUBRID 2008 R4.1에서 변경된 사항

새로 추가된 기능

CUBRIDSUS-5860 ADDTIME 함수 추가

날짜/시간 타입의 값에 시간 값을 더하는 ADDTIME 함수를 추가했다.

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

CUBRIDSUS-6265 ASCII 함수 추가

입력 문자를 ASCII 코드 값으로 반환하는 ASCII 함수를 추가했다.

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

CUBRIDSUS-5860 BIN 함수 추가

숫자를 이진 문자열로 표현하는 BIN 함수를 추가했다.

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

CUBRIDSUS-6233 CONV 함수 추가

숫자를 나타내는 입력 문자열을 진수가 다른 숫자로 변환하는 CONV 함수를 추가했다.

```
// 다음은 'a'인 16진수를 2진수로 바꾼다.
SELECT CONV('a',16,2);
'1010'
```

CUBRIDSUS-5860 FIND_IN_SET 함수 추가

쉼표(,)를 구분자로 하는 문자 집합에서 지정한 문자의 개수를 반환하는 FIND_IN_SET 함수를 추가했다.

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

CUBRIDSUS-6233 HEX 함수 추가

입력 인자가 문자열인 경우 각 문자에 해당하는 아스키 코드를 2자리 16진수 문자열로 반환하며, 입력인자가 숫자인 경우 10진수를 16진수 문자열로 변환하는 HEX 함수를 추가했다.

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

CUBRIDSUS-6190 현재의 시리얼 값과 하나 이상의 시리얼 값을 반환하는 함수들 추가

현재의 시리얼 값을 반환하는 SERIAL_CURRENT_VALUE(serial_name)와, 하나 이상의 시리얼 값을 반환하는 SERIAL_NEXT_VALUE(serial_name, number) 를 추가했다.

SERIAL_CURRENT_VALUE(serial_name)는 serial_name.current_value와 동일하게 동작한다.

CUBRIDSUS-5903 정규 표현식에 대한 연산자 추가

정규 표현식(regular expression)에 대한 연산자인 REGEXP를 추가했다. REGEXP에서 사용하는 패턴은 대소문자를 구분하지 않으며, 대소문자를 구분하려면 REGEXP BINARY를 사용한다.

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


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

CUBRIDSUS-5616 질의 실행 시간을 제한하는 브로커 파라미터 추가

브로커로 연결하는 응용 프로그램의 질의 실행 시간을 브로커 파라미터로 제한하기 위해 질의 타임아웃 값을 설정하는 **MAX_QUERY_TIMEOUT** 파라미터를 추가했다. **broker_changer** 유틸리티로 동적 변경이 가능하며, 값의 범위는 0~86400초(1일)이다.

CUBRIDSUS-6197 JDBC 연결 URL에 질의 타임아웃 시간을 설정하는 기능 추가

JDBC 연결 URL에 **queryTimeout** 프로퍼티를 사용하여 질의 수행에 대한 타임아웃 시간을 초 단위로 설정하는 기능이 추가되었다(기본값: 0, 무제한). 이 값은 DriverManager.setQueryTimeout() 메소드에 의해 변경될 수 있다.

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

CUBRIDSUS-6198 JDBC 연결 URL에 DB 연결 타임아웃 시간을 설정하는 기능 추가

JDBC 연결 URL에 **connectTimeout** 프로퍼티를 사용하여 DB 연결에 대한 타임아웃 시간을 초 단위로 설정하는 기능이 추가되었다(기본값: 0). DriverManager.setLoginTimeout() 메소드로도 이 값을 설정할 수 있으나, 연결 URL에 설정하면 해당 메소드로 설정하는 값은 무시된다.

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

CUBRIDSUS-5388 CAS의 상태를 출력할 때 트랜잭션 시작 시간을 추가

cubrid broker status -f 명령으로 CAS의 상태를 출력할 때 트랜잭션 시작 시간을 추가했다.

CUBRIDSUS-6199 CAS의 상태를 출력할 때 응용 프로그램이 CAS에 연결한 회수, CAS의 재구동 회수를 추가

cubrid broker status -f 명령으로 CAS의 상태를 출력할 때 응용 프로그램이 CAS에 연결한 회수, CAS의 재구동 회수를 추가했다. 이러한 CAS의 재구동 회수가 많은 경우, CAS의 메모리 사용량의 최대 크기를 지정하는 브로커 파라미터인 **APPL_SERVER_MAX_SIZE**의 설정 값을 현재보다 크게 변경하는 것을 고려할 수 있다.

CUBRIDSUS-6128 에러 심각성 수준이 NOTIFICATION일 때 교착상태가 발생하면 서버 에러 로그 파일에 잠금 관련 정보를 기록하는 기능 추가

에러 심각성 수준을 설정하는 시스템 파라미터인 **error_log_level**의 값을 NOTIFICATION으로 설정했을 때 교착 상태가 발생하면 서버 에러 로그 파일에 잠금 관련 정보를 기록하는 기능을 추가했다.

다음의 에러 로그 파일 정보에서 (1)은 교착상태를 유발한 테이블 이름을, (2)는 인덱스 이름을 나타낸다.

```
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
    ...
```

개선된 사항

CUBRIDSUS-5300 트랜잭션 로그의 동시 처리 성능 개선

여러 응용 프로그램들에 의해 동시에 트랜잭션 로그 버퍼를 사용하는 작업들이 수행되어야 하는 경우, 이들이 순서대로 진행되도록 잠금을 유지하는 구간(critical section) 내의 작업을 최소화하여, 대량의 트랜잭션 부하 테스트에서 특히 INSERT와 UPDATE에 대해 2008 R4.0 Patch 2 버전 대비 70% 의 성능이 향상되었다.

CUBRIDSUS-5687 메모리 버퍼의 더티 페이지를 디스크에 쓰는 방식을 수정하여 쓰기 성능 개선

메모리 버퍼의 더티 페이지를 디스크에 무작위로 쓰는 방식에서 일정 분량의 페이지를 모아 정렬하여 쓰도록 하여, 쓰기 트랜잭션의 성능이 개선되었다.

CUBRIDSUS-5297 현재의 날짜/시간 정보를 구하는 함수의 실행 성능 개선

SYSDATE, SYSTIME, SYSDATETIME, SYSTIMESTAMP 등 현재의 날짜/시간 정보를 구하는 함수 실행 시 이전 버전에서는 매번 서버에 값을 요청했으나 최근 1분 이내 요청된 값으로부터 현재 값을 계산하도록 하여 이들 함수의 실행 성능이 개선되었다.

CUBRIDSUS-6570 ORDER BY 컬럼에 대해 BETWEEN … AND 조건인 경우에도 다중 키 범위 조건의 정렬 수행 최적화가 가능하도록 개선

ORDER BY 절의 컬럼에 대해 동등(=) 조건인 경우에만 중간 값의 정렬을 진행하면서 결과를 수집하는 다중 범위 최적화(multi-range optimization)를 수행할 수 있었으나, 해당 컬럼에 대해 BETWEEN … AND … 조건인 경우에도 수행할 수 있도록 개선했다.

```
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 질의 실행 계획 캐시에 INSERT 질의가 포함되도록 개선

이전 버전에서는 SELECT, UPDATE, DELETE 질의만이 질의 실행 계획 캐시에 저장되었으나, INSERT 질의도 포함되도록 개선했다.

CUBRIDSUS-5850 시스템의 부하가 매우 큰 경우 트랜잭션 처리 응답 시간의 최저 값과 최고 값의 편차를 줄이도록 개선

시스템의 부하가 매우 큰 경우에도 일정한 성능을 유지하게 하기 위해 트랜잭션 처리 응답 시간의 최저 값과 최고 값의 편차를 줄이도록 개선했다.

CUBRIDSUS-5824 HA 환경에서 복제 지연 시 성능 개선

HA 환경에서 복제 지연 발생 시에 저장 로그(archive log) 볼륨 파일에 접근할 때 존재하던 오버헤드를 제거하여 성능을 개선했다.

수정된 사항

CUBRIDSUS-5882 날짜/시간 타입에 대해 '0000-00-00' 또는 '0000-00-00 00:00:00' 형식의 입력 지원

이전 버전에서는 DATETIME, TIMESTAMP 타입에서만 '0000-00-00 00:00:00' 형식의 입력을 지원하였으나 DATE 타입도 '0000-00-00' 형식의 입력을 지원하도록 수정했다. 그리고 이전 버전에서는 DATETIME, TIMESTAMP 타입에서 '0000-00-00 00:00:00'을 입력하면 최소값으로 변환되어 저장되었으나 '0000-00-00 00:00:00'이 저장되도록 수정했다.

이로 인해 TIMESTAMP 타입의 최소값이 '1970-01-01 00:00:00' (GMT)에서 '1970-01-01 00:00:01' (GMT)로 변경되었다.

보다 자세한 사항은 매뉴얼의 'CUBRID SQL 설명서 > 데이터 타입 > 날짜/시간 데이터 타입'을 참고한다.

CUBRIDSUS-5296 시리얼(SERIAL)을 포함하는 질의문의 질의 계획을 질의 실행 계획 캐시에 저장할 수 있도록 수정

시리얼을 포함하는 질의문의 질의 계획을 질의 실행 계획 캐시에 저장할 수 있도록 수정했다.

CUBRIDSUS-6138 ABS 등의 수치 연산 함수들에서 입력 인자가 호스트 변수이고 문자열을 바인딩하는 경우 질의 결과가 잘못되는 오류 수정

ABS, FLOOR, CEIL 등 수치 연산 함수들에서 입력 인자가 호스트 변수이고 문자열을 바인딩하는 경우 질의 결과가 잘못되는 오류를 수정했다.

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

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

CUBRIDSUS-5988 TO_CHAR 함수의 첫번째 인자가 호스트 변수이고 DATE 타입의 값이 바인딩되면 발생하는 오류 수정

TO_CHAR 함수의 첫번째 인자가 호스트 변수이고 DATE 타입의 값이 바인딩되면 "Semantic: Cannot coerce host var to type double" 오류가 발생하였으나 이를 수정했다.

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

CUBRIDSUS-6280 STR_TO_DATE 함수에서 '%r' 포맷 지정 시 PM 시각 문자열이 타입 변환이 되지 않거나 '12:XX:XX AM' 문자열이 '12:XX:XX PM'으로 잘못 변환되는 오류 수정

STR_TO_DATE 함수에서 '시:분:초 PM' 형태의 문자열에 '%r' 포맷을 지정하는 경우 타입 변환이 되지 않는 오류와, 오전 12시인 '12:XX:XX AM' 형태의 문자열에 '%r' 포맷을 지정하는 경우 오후 12시인 '12:XX:XX PM'으로 잘못 변환되는 오류를 수정했다.

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

CUBRIDSUS-6023 조건절에 AND .. OR 조건이 포함된 경우 OR의 왼쪽 항이 AND로 연결된 조건이면 질의 결과가 잘못될 수 있는 오류 수정

조건절에 AND .. OR 조건이 포함된 경우 OR의 왼쪽 항이 AND로 연결된 조건이고 이 조건이 항상 거짓이면 OR 조건 전체의 결과도 거짓으로 판단하여 질의 결과가 잘못될 수 있는 오류를 수정했다.

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

CUBRIDSUS-6231 NULL 입력 여부에 따라 결과 값을 정하는 SQL 함수들에서 입력 타입에 따른 결과 타입의 변환 규칙 변경

IFNULL, NVL, NVL2, COALESCE 와 같이 NULL 입력 여부에 따라 결과 값을 정하는 SQL 함수들에서 입력 타입에 따른 결과 타입의 변환 규칙을 변경했다. 입력 인자들 사이에서 우선순위가 가장 큰 타입이 결과 타입이 되며, 타입이 호환되지 않는 경우 VARCHAR 타입이 결과 타입이 된다.

변환 가능한 타입과 우선순위는 아래와 같다.

- CHAR < VARCHAR

- NCHAR < NCHAR VARYING

- BIT < VARBIT

- SHORT < INTEGER < BIGINT < NUMERIC < FLOAT < DOUBLE

- DATE < TIMESTAMP < DATETIME

예를 들어, 아래와 같이 COALESCE 함수에서 입력 인자의 타입인 INTEGER, DATE, TIME 사이에 서로 타입 변환이 될 수 없는 경우 이전 버전에서는 "ERROR: 'coalesce ' operator is not defined on types date and time." 오류가 발생했으나, 2008 R4.1 버전부터는 VARCHAR 타입의 결과 값을 반환한다.

```
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 커버링 인덱스로 WHERE 절에 IN 조건식이 있는 질의 수행 도중 발생하는 메모리 누수 현상 수정

커버링 인덱스(covering index)로 WHERE 절에 IN 조건식이 있는 질의 수행 도중 발생하는 메모리 누수 문제를 수정했다. 커버링 인덱스에 대한 자세한 내용은 매뉴얼의 'CUBRID SQL 설명서 > 질의 최적화 > 인덱스 활용 > 커버링 인덱스'를 참조한다.

CUBRIDSUS-6345 커버링 인덱스가 사용되며 연관 부질의가 존재하는 질의의 경우 잘못된 결과가 출력될 수 있는 오류 수정

질의 결과 집합에 대한 모든 데이터를 포함하는 커버링 인덱스(covering index)가 사용되며 연관 부질의(correlated subquery)가 존재하는 질의의 경우 잘못된 결과가 출력될 수 있는 오류를 수정했다. 이 오류는 2008 R4.0 버전 ~ 2008 R4.0 Patch 2 버전에서만 발생했다.

```
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 조인 질의문에 LIMIT 조건이 주어지고 인덱스를 이용한 LIMIT 최적화가 적용되면 잘못된 결과를 출력할 수 있는 문제 수정

조인 질의문에 LIMIT 조건이 주어지고 인덱스를 이용한 LIMIT 최적화가 적용되면, Nested Loop 조인의 외부 테이블과 내부 테이블에 각각 LIMIT 최적화를 적용하는 오류로 인해 기대보다 적은 수의 레코드를 출력하는 문제를 수정했다. 이 문제는 2008 R4.0 ~ R4.0 Patch 2 버전에서만 발생했다.

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

CUBRIDSUS-6322 계층 질의 수행 시 서버가 비정상 종료될 수 있는 문제 수정

레코드 개수가 많은 테이블에 대해 계층 질의를 수행하면 비정상 종료되는 문제를 수정했다.

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

CUBRIDSUS-6282 파티션 테이블 또는 부모 테이블에서 인덱스 관련 제약조건을 DROP 후 재생성하면 서버 프로세스가 비정상 종료되는 문제 수정

파티션 테이블 또는 자식 테이블을 가진 부모 테이블에서 PRIMARY KEY, UNIQUE, FOREIGN KEY 등 인덱스 관련 제약조건을 DROP 후 재생성하면 서버 프로세스가 비정상 종료되는 문제를 수정했다.

```
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 연관 부질의를 포함한 질의가 잘못된 결과를 출력하는 문제 수정

아래와 같은 연관 부질의를 포함한 질의가 잘못된 결과를 출력하는 문제를 수정했다.

```
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 INSERT 또는 UPDATE 도중 인덱스 변경과 관련하여 발생하는 오류 수정

INSERT 또는 UPDATE로 인해 인덱스 키 하나의 크기가 DB 볼륨 한 페이지 크기의 1/8을 초과할 때 발생할 수 있는 다음과 같은 오류를 수정했다.

```
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 서로 다른 DB 연결들이 하나의 세션 ID를 참조할 수 있는 오류 수정

서로 다른 DB 연결들이 하나의 세션 ID를 참조할 수 있는 오류로 인해 사용자 정의 변수, PREPARE 문, LAST_INSERT_ID, ROW_COUNT 등의 값이 잘못될 수 있는 문제를 수정했다. 다만, HA failover 시에는 수정 이후 버전에서도 해당 현상이 여전히 발생할 수 있다.

CUBRIDSUS-6054 "SELECT COUNT(DISTINCT 상수) FROM tbl"의 결과를 1건으로 출력하도록 수정

"SELECT COUNT(DISTINCT 상수) FROM tbl"의 결과로 이전에는 레코드 전체 개수를 출력하였으나 1건을 출력하도록 수정했다.

CUBRIDSUS-6370 다중 컬럼 인덱스를 구성하는 컬럼의 NULL 값을 UPDATE하면 오류가 발생하는 문제 수정

다중 컬럼 인덱스를 구성하는 컬럼에 존재하는 NULL 값을 UPDATE하면 다음의 오류가 발생하는 문제를 수정했다.

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

CUBRIDSUS-6347 PREPARE 실패 시 cubrid broker status의 에러 카운트를 증가하도록 수정

PREPARE 실패 시 **cubrid broker status**의 에러 카운트(ERR-Q)를 증가하도록 수정했다.

CUBRIDSUS-5879 CCI 인터페이스에서 자동 커밋 모드의 기본값이 ON이 되도록 수정

CCI 인터페이스에서 브로커의 **CCI_DEFAULT_AUTOCOMMIT** 파라미터를 통해 설정하는 자동 커밋 모드의 기본값이 ON이 되도록 수정했다.

CUBRIDSUS-6220 CCI 기반 응용 프로그램에서 어떤 질의 요청 핸들러(request handler)를 CLOSE 할 때 다른 질의 요청 핸들러에 대한 커밋에 영향을 주는 오류 수정

CCI 기반 응용 프로그램에서 어떤 질의 요청 핸들러(request handler) req1이 CLOSE되는 시점에 다른 질의 요청 핸들러 req2의 커밋에 영향을 주는 오류를 수정했다.

다음은 이전 버전에서 문제가 발생하였던 시나리오로, EXECUTE(req1) 이후에는 자동 커밋 모드가 OFF로 바뀌었음에도 불구하고 CLOSE(req1) 시점에 req1에 잘못 기록되었던 자동 커밋 모드를 참조하여 커밋이 수행되면서 req2의 변경내역도 반영되었다.

```
AUTOCOMMIT ON
  req1 = PREPARE
              EXECUTE(req1)
```

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

CUBRIDSUS-6327 CCI기반 응용 프로그램에서 PREPARE 함수 수행 중 오류가 발생하면 이후 트랜잭션 롤백을 실행해도 롤백되지 않는 현상 수정

CCI 기반 응용 프로그램에서 PREPARE 함수 수행 중 오류가 발생하면 이후 트랜잭션 롤백을 실행해도 롤백되지 않는 현상을 수정했다.

CUBRIDSUS-6366 Linux 환경의 CCI 기반 응용 프로그램에서 네트워크 소켓 fd 값이 1024를 초과하는 경우 비정상 동작하는 문제 수정

Linux 환경의 CCI 기반 응용 프로그램에서 네트워크 소켓 fd(file descriptor) 값이 1024를 초과하는 경우 CCI 인터페이스 구현에 사용된 네트워크 함수인 select()의 한계로 인해 응용 프로그램이 잘못된 메모리를 참조하게 되어, 비정상 종료 또는 통신 오류가 발생하는 문제를 수정했다.

CUBRIDSUS-6491 cci_datasource_borrow() 호출 시 연결을 가져오는 대기 시간까지 기다리지 않고 오류를 반환할 수 있는 문제 수정

cci_datasource_borrow() 호출 시 연결을 가져오기 위해 대기하는 최대 시간(max_wait)까지 기다리지 않고 "All connections are used"라는 오류를 반환할 수 있는 문제를 수정했다.

CUBRIDSUS-6606 CCI 인터페이스의 DATASOURCE 사용 시 statement 풀링을 설정하면 발생하는 메모리 누수 현상 수정

cci_datasource_create()를 호출하여 DATASOURCE를 생성할 때 statement 풀링 기능을 사용하도록 설정하면 발생하는 메모리 누수 현상을 수정했다.

CUBRIDSUS-6673 CCI 인터페이스로 개발된 응용에서 COMMUNICATION ERROR 발생 시 비정상 동작하는 현상 수정

CCI 인터페이스로 개발된 응용에서 "COMMUNICATION ERROR net_read_header()" 오류 발생 시 비정상 동작하는 현상이 존재하였으나 네트워크 함수 오류에 대한 검사를 추가하여 이 문제를 수정했다.

CUBRIDSUS-6318 JDBC에서 Java의 BigDecimal 클래스를 사용하여 BIGINT 타입의 값을 가져오는 경우 잘못된 값을 가져오는 오류 수정

JDBC에서 Java의 BigDecimal 클래스를 사용하여 BIGINT 타입의 값을 가져오는 경우 잘못된 값을 가져오는 오류를 수정했다.

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

CUBRIDSUS-6290 JDBC 드라이버에서 한 번의 요청으로 입력 가능한 데이터의 크기 제한을 제거

JDBC 드라이버에서 한 번의 요청(예를 들어 Java 응용 프로그램에서 execute 혹은 executeBatch와 같은 메소드를 통한 요청)으로 입력 가능한 데이터의 크기가 이전 버전에서는 100MB로 제한되었으나, 크기 제한이 제거되었다. 2008 R4.0 Patch 2 이하 버전에서는 크기 제한을 넘을 경우 다음의 오류가 발생했다.

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

CUBRIDSUS-6453 독립 모드로 실행한 CSQL 인터프리터에서 사용자 세션 변수로 PREPARE 문을 수행하면 비정상 종료하는 현상 수정

독립 모드(-S)로 실행한 CSQL 인터프리터에서 사용자 세션 변수를 사용하여 PREPARE 문을 수행하면 비정상 종료하는 현상을 수정했다.

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

CUBRIDSUS-6106 addvoldb, spacedb 유틸리티를 SA 모드로 수행할 때 16MB의 데이터 버퍼를 사용하도록 수정

cubrid addvoldb, spacedb 유틸리티를 SA 모드로 수행할 때 이전 버전에서는 512M의 데이터 버퍼를 사용하였으나, 16MB의 데이터 버퍼를 사용하도록 수정했다.

CUBRIDSUS-6235 CUBRID 매니저와 같은 응용 프로그램에서 장시간 동안 요청이 없는 경우에도 연결이 종료되지 않도록 수정

CUBRID 브로커가 설치된 장비 또는 방화벽 장비에서 일정 시간 동안 데이터의 전송이 없으면 소켓 연결을 종료하게 설정되어 있는 경우, 연결 종료 이후 재연결 과정에서 많은 시간이 소요된다. 이와 같이 설정된 장비에 연결한 CUBRID 매니저의 질의 편집기를 장시간 동안 방치했다가 질의를 수행하면, 결과를 출력하기 전까지 장시간이 소요되는 현상을 수정했다.

CUBRIDSUS-5472 질의 수행 중 인터럽트가 발생하는 경우 CSQL 또는 CAS가 비정상 종료되는 문제 수정

클라이언트/서버 모드로 실행한 CSQL 혹은 질의 실행 함수(cci_execute()/cubrid_execute())를 ASYNC 모드로 설정한 CCI/PHP 인터페이스 프로그램에서 질의 수행 중 인터럽트(ctrl+C, 또는 query cancel)가 발생하는 경우, CSQL 또는 CAS가 비정상 종료되는 문제를 수정했다.

CUBRIDSUS-5816 크기를 지정하는 CUBRID 유틸리티의 옵션 혹은 시스템 파라미터에 소수점 이하의 숫자 입력을 허용하도록 수정

크기를 지정하는 CUBRID 유틸리티의 옵션 혹은 시스템 파라미터에 소수점 이하의 숫자 입력을 허용했다.

```
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 인덱스가 삭제되는 작업 도중에 인터럽트가 발생하면 나타날 수 있는 인덱스 깨짐 현상 수정

인덱스가 삭제되는 작업 도중에 질의 타임아웃 등의 인터럽트가 발생하면 나타날 수 있는 인덱스 깨짐 현상을 수정했다.

CUBRIDSUS-6418 온라인 백업 수행 도중 DDL을 동시에 수행하면 서버 프로세스가 동작을 멈추는(hang) 문제 수정

온라인 백업 수행 도중 DDL을 동시에 수행하면 서버 프로세스가 동작을 멈추는(hang) 문제를 수정했다. 이 문제는 2008 R4.0 Patch 2 버전에서만 발생했다.

CUBRIDSUS-6279 인덱스 리프 노드(leaf node)의 이전 링크가 잘못되는 오류 수정

인덱스 병합(merge) 또는 분할(split) 시 리프 노드(leaf node)의 이전 링크가 잘못되어, 역방향 검색 결과가 잘못 출력되거나 임시 볼륨이 급격히 증가하는 문제를 수정했다. 이 문제는 2008 R4.0 GA 및 그 이후 버전에서 발생했다.

CUBRIDSUS-6001 다른 트랜잭션이 입력 도중 실패한 공간을 사용하기 위해 대기하게 되는 현상 수정

트랜잭션 A가 고유 키 위반(unique key violation)등으로 특정 레코드의 INSERT에 실패한 경우 트랜잭션이 종료되기 전까지 해당 슬롯의 잠금을 유지함으로 인해, 트랜잭션 B는 해당 슬롯의 잠금이 해제되기를 기다리는 현상이 존재하였으나, 트랜잭션 B가 새로운 슬롯을 찾아 대기 과정 없이 INSERT하도록 수정했다.

CUBRIDSUS-6396 UPDATE 도중 인터럽트가 발생하면 DB 볼륨 크래시(crash) 현상 수정

UPDATE 도중 인터럽트가 발생한 이후 인덱스 복구 과정에서 오류가 존재하여 DB 볼륨이 크래시되는 현상을 수정했다.

CUBRIDSUS-6720 부울리언 표현식 처리 과정에서 서버 프로세스의 메모리 누수 현상 수정

부울리언(Boolean) 표현식 처리 과정에서 서버 프로세스(cub_server)의 메모리 누수 현상을 수정했다.

CUBRIDSUS-6072 응용프로그램에서 트랜잭션을 종료하고 질의 요청 핸들을 CLOSE한 이후 일정 시간 동안 요청이 없는 경우 연결이 종료되는 문제 수정

응용프로그램에서 트랜잭션을 종료하고 질의 요청 핸들을 CLOSE한 이후, 일정 시간 동안 요청이 없으면 세션 타임아웃(브로커 파라미터인 SESSION_TIMEOUT으로 설정)이 발생하여 CAS와 응용 프로그램 간의 연결이 종료되는 문제를 수정했다.

CUBRIDSUS-6342 체크포인트 오류로 인해 서버가 비정상 종료된 이후 DB 복구에 실패할 수 있는 문제 수정

체크포인트 오류로 인해 서버가 비정상 종료되고, 이후 DB 복구에 실패할 수 있는 문제를 수정했다.

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

CUBRIDSUS-6513 서버 프로세스의 오류 로그에 소스의 파일 이름과 라인 번호가 항상 출력되도록 수정

서버 프로세스(**cub_server**)에서 기록하는 오류 로그에 소스의 파일 이름과 라인 번호가 항상 출력되도록 수정했다.

CUBRIDSUS-6240 SLOW SQL 로그 기록 시 질의 수행 시작 시각과 완료 시각이 모두 기록되도록 수정

SLOW SQL 로그 기록 시 질의 수행 완료 시점의 시각이 일괄적으로 기록되었으나, 질의 수행 시작 시각과 완료 시각이 모두 기록되도록 수정했다.

```
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 CAS 프로세스의 메모리 사용량이 급격히 증가하는 경우를 차단하기 위한 방법 제공

CAS 프로세스의 메모리 사용량이 급격히 증가하는 경우를 차단하기 위해 지정한 메모리 사용량을 초과하면 강제로 CAS 프로세스를 재구동하게 했다.

이와 관련하여 새로 추가된 브로커 파라미터는 **APPL_SERVER_MAX_SIZE_HARD_LIMIT**로서, CAS 프로세스가 이 파라미터에 의해 지정된 메모리 사용량을 초과하면 진행 중인 트랜잭션이 강제로 종료(롤백)되고 해당 CAS 프로세스가 재구동된다. 기본값은 1024MB이다. 이전 버전부터 있던 **APPL_SERVER_MAX_SIZE** 브로커 파라미터는 이와 비슷한 역할을 하지만, CAS가 이 파라미터에 의해 지정된 메모리 사용량을 초과해도 진행 중인 트랜잭션이 있으면 사용자에 의해 정상 종료(커밋 혹은 롤백)되기를 기다린 후 재구동된다는 점이 다르다.

일반적으로 메모리 사용량이 지정된 크기를 초과해도 어느 정도까지는 트랜잭션이 정상 종료되기를 기다리기 위해서 **APPL_SERVER_MAX_SIZE_HARD_LIMIT**의 값을 **APPL_SERVER_MAX_SIZE**의 값보다 크게 설정할 것을 권장한다.

CUBRIDSUS-5851 replica 노드에서는 ASYNC 모드로만 트랜잭션 로그를 복사하도록 수정

HA 환경에서 replica 노드인 경우, cubrid_ha.conf의 **ha_copy_sync_mode** 파라미터 설정과 관계 없이 ASYNC 모드로 트랜잭션 로그를 복사하도록 수정했다.

CUBRIDSUS-4817 HA 환경에서 한 트랜잭션 내 변경 레코드 개수가 매우 많은 경우 메모리 부족으로 인해 복제 반영이 되지 않는 문제 수정

HA 환경에서 한 트랜잭션 내 변경 레코드 개수가 매우 많은 경우 복제 로그를 반영하는 **applylogdb**의 메모리 사용량이 프로세스에 허용된 최대 메모리 사용량을 초과하면서 복제 반영이 되지 않는 문제를 수정했다.

CUBRIDSUS-6117 HA 환경에서 2008 R4.0 Patch2 버전 이후 발생한 applylogdb의 성능 퇴행 개선

HA 환경에서 2008 R4.0 Patch2 버전 이후 발생한 **applylogdb**의 성능 퇴행을 개선했다.

CUBRIDSUS-6172 HA 환경에서 failover 수행 이후, 슬레이브에서 마스터로 변경된 노드에 복제된 보관 로그 파일이 삭제되지 않고 계속 남아 있을 수 있는 현상 수정

HA 환경에서 failover 수행 이후, 슬레이브에서 마스터로 변경된 노드에 보관 로그(archive log) 파일의 유지 개수를 설정하는 시스템 파라미터인 **log_max_archives**를 설정하였음에도 불구하고, 복제된 보관 로그 파일이 삭제되지 않고 계속 남아 있을 수 있는 현상을 수정했다.

CUBRIDSUS-6183 HA 환경에서 applylogdb가 쓰기가 끝나지 않은 로그 페이지를 버퍼에 캐싱하여 읽고 반영하면서 오동작하는 문제 수정

HA 환경에서 **copylogdb** 프로세스에 의해 쓰기가 끝나지 않은 로그 페이지를 **applylogdb**가 페이지 버퍼에 캐싱한 후 이를 읽고 반영하면서 잘못된 로그 페이지 반영, 프로세스 비정상 종료 등 오동작할 수 있는 문제를 수정했다.

CUBRIDSUS-6193 HA 환경에서 마스터 노드 테이블의 기본 키를 변경한 직후 온라인 백업본으로 슬레이브 노드를 재구축하면 슬레이브 서버 프로세스가 비정상 종료될 수 있었으나 오류 처리하도록 수정

HA 환경에서 마스터 서버의 기본 키를 변경한 직후 온라인 백업본으로 슬레이브 노드를 재구축하는 경우, 기본 키 변경 이전의 로그를 기본 키 변경이 완료된 테이블에 반영하려고 시도하면서 슬레이브 서버 프로세스가 비정상 종료될 수 있었으나, 로그 반영 과정에서 기본 키 구성이 로그 레코드와 일치하지 않으면 오류로 처리하도록 수정했다.

CUBRIDSUS-5475 HA 환경의 슬레이브 노드에서 INCR/DECR 함수를 호출하면 해당 데이터를 업데이트할 수 없도록 수정

HA 환경의 슬레이브 노드에서 INCR/DECR 함수를 호출하면 이전 버전에서는 함수가 수행되면서 마스터 서버와 슬레이브 노드 간 데이터 불일치가 발생하였으나, 해당 데이터를 업데이트할 수 없도록 수정했다.

CUBRIDSUS-5939 HA 환경에서 standby 상태의 노드를 maintenance 상태로 변경할 때 진행 중이던 트랜잭션이 정상 종료될 때까지 대기하는 시간을 설정하는 기능 제공

HA 환경에서 **changemode** 유틸리티를 실행하여 standby 상태의 노드를 maintenance 상태로 변경할 때, 진행 중이던 트랜잭션이 정상 종료될 때까지 대기하는 시간을 설정하는 -t 옵션(생략 시 기본값: 5초)을 제공했다. 설정한 시간까지 트랜잭션이 진행 중이면 강제 종료 후 maintenance 상태로 변경하고, 설정한 시간 이내에 모든 트랜잭션이 정상 종료되면 즉시 maintenance 상태로 변경한다.

```
// 진행 중이던 트랜잭션의 수행 시간이 changemode 실행 시점 이후 10초를 넘으면 트랜잭션이
롤백되고, 노드 상태는 maintenance 상태로 변경된다.
cubrid changemode -m maintenance -t 10 demodb
```

CUBRIDSUS-6154 데이터 버퍼의 더티 페이지가 디스크에 저장되는 주기를 설정하는 시스템 파라미터의 값이 -1일 때의 동작 방식 변경

데이터 버퍼의 더티 페이지(dirty page)가 디스크에 저장되는 주기를 설정하는 시스템 파라미터인 **page_flush_interval_in_msecs**의 값이 -1인 경우 체크포인트 또는 페이지가 스왑(swap)되는 시점에만 더티 페이지가 디스크로 저장되었으나, 주기 값이 0인 경우와 동일하게 즉시 저장되도록 변경했다.

CUBRIDSUS-6133 교착상태를 감지하는 주기를 설정하는 시스템 파라미터 값의 입력 단위를 소수점 이하 한자리까지 가능하도록 변경

교착상태(deadlock)를 감지하는 주기를 설정하는 시스템 파라미터인 **deadlock_detection_interval_in_secs** 값의 입력 단위를 소수점 이하 한자리까지 가능하도록 변경했다. 최소값은 1초에서 0.1초로 변경되었다. 이 값은 0.1초 단위로 올림하여 동작한다. 즉, 입력값이 0.12초이면 0.2초를 입력한 것과 같이 동작한다.

CUBRIDSUS-6083 스레드의 스택 크기를 결정하는 기준 및 크기를 설정하는 시스템 파라미터의 기본값 변경

OS에 설정된 스레드의 스택 크기가 시스템 파라미터인 **thread_stacksize** 보다 큰 경우 이전 버전에서는 OS의 설정값에 의해 크기가 정해졌으나, 항상 **thread_stacksize**의 값에 의해 크기가 정해지도록 변경하였으며, **thread_stacksize**의 기본값을 100KB에서 1MB로 변경했다.

CUBRIDSUS-6668 데이터 캐싱 버퍼의 크기를 설정하는 시스템 파라미터의 최소값 변경

데이터 캐싱 버퍼의 크기를 설정하는 시스템 파라미터인 **data_buffer_size**의 최소값을 64K에서 16M로 변경했다.

CUBRIDSUS-6035 Linux 용 CUBRID 패키지를 기존 CUBRID 버전이 설치되어 있는 경로에 설치하면 파일들이 잘못 설치되는 오류 수정

Linux 용 CUBRID 패키지(.sh)를 기존 CUBRID 버전이 설치되어 있는 경로에 설치하는 경우 파일들이 잘못 설치되는 오류를 수정했다.

CUBRIDSUS-6061 JDK1.7로 JDBC 드라이버 소스를 빌드하지 못하는 문제 수정

JDK1.7로 JDBC 드라이버 소스를 빌드하지 못하는 문제를 수정했다. 참고로, JDK 1.7로 JDBC 드라이버 소스를 빌드하기 위해서는 ANT 1.8 이상의 버전이 필요하다.

CUBRIDSUS-5942 Windows 시스템을 재부팅한 이후 CUBRID Service Tray가 자동으로 구동될 수 있는 환경인지 CUBRID 설치 중에 미리 확인이 가능하도록 수정

Windows 시스템을 재부팅한 이후 CUBRID Service Tray가 자동으로 구동될 수 있는 환경인지 CUBRID 설치 중에 미리 검사하여, 문제 발견 시 주의 메시지를 출력하도록 수정했다. 시스템 재부팅 시 CUBRID Service Tray가 자동으로 구동되려면 "제어판 > 관리 도구 > 서비스"의 Task Scheduler가 시작된 상태에서 "관리 도구 > 작업 스케쥴러"에 CUBRID Service Tray가 등록되어야 한다.

CUBRIDSUS-6028 Windows 용 CUBRID 패키지 설치 후 시스템을 재부팅해야 cubrid 유틸리티가 실행되는 문제 수정

Windows 용 CUBRID 패키지 설치 후 시스템을 재부팅하기 전까지 CUBRID 관련 환경 변수가 시스템에 적용되지 않는 현상으로 인해 **cubrid** 유틸리티가 실행되지 않는 문제를 수정했다.

CUBRIDSUS-6278 Windows용 CCI 라이브러리(cascci.dll)를 사용한 응용 프로그램 수행 시 CRT DLL이 필요하지 않도록 수정

Windows용 CCI 라이브러리(cascci.dll)를 사용한 응용 프로그램 수행 시 CRT 동적 연결 라이브러리(C RunTime DLL)가 필요하지 않도록 수정했다.

5.10 주의 사항

2008 R4.1 신규 주의 사항

CUBRIDSUS-5879 2008 R4.1 버전부터 CCI_DEFAULT_AUTOCOMMIT 의 기본값이 ON으로 바뀜

2008 R4.1 버전부터 CCI 인터페이스로 개발된 응용 프로그램의 자동 커밋 모드에 영향을 주는 브로커 파라미터인 **CCI_DEFAULT_AUTOCOMMIT**의 기본값이 ON으로 변경되었다. 따라서 CCI 및 CCI로 개발된 인터페이스(PHP, ODBC, OLE DB 등) 사용자는 응용 프로그램의 자동 커밋 모드가 이에 적합한지 살펴보아야 한다.

CUBRIDSUS-5238 2008 R4.1 버전은 2008 R4.0 Beta 및 그 이전 버전과 DB 볼륨이 호환되지 않음

- 2008 R4.0 버전 이상 사용자는 2008 R4.1과 DB 볼륨이 호환된다.
- 2008 R4.1 버전은 2008 R4.0 Beta 및 그 이전 버전과 DB 볼륨이 호환되지 않으므로 DB를 업그레이드하려면 DB 볼륨을 마이그레이션해야 한다.
- 2008 R3.x 및 이전 버전 사용자는 cubrid unloaddb/loaddb를 이용한다.
- 2008 R4.0 Beta 사용자는 http://ftp.cubrid.org에서 제공되는 migrate_r40beta2ga 유틸리티를 이용할 수 있으나, 페이지 크기가 4K 미만인 볼륨은 cubrid unloaddb/loaddb를 이용하여야 한다.

보다 자세한 사항은 *DB 마이그레이션 절차*를 참고한다.

기존 주의 사항

CUBRIDSUS-5597 CCI, PHP, JDBC 연결 URL에서 구분자를 암호에 포함할 수 없음

CCI, PHP, JDBC 연결 URL에서 구분자로 사용되는 ? 또는 :을 암호에 포함할 수 없다. 다음은 암호에 ?가 있어 연결 URL로 사용할 수 없는 예이다.

```
// CCI, PHP의 경우
url = "cci:jdbc:CUBRID:192.168.0.10:33000:tdb:dba:12?:?charset=UTF-8";
// JDBC의 경우
url="jdbc:CUBRID:192.168.0.10:33000:tdb:dba:12?:?charset=UTF-8";
```

암호에 ? 또는 :을 포함한 경우에는 암호를 별도의 인자로 전달하여 사용할 수 있다.

```
// CCI의 경우
url="cci:CUBRID:192.168.0.10:33000:tdb:::?charset=UTF-8";
cci_connect_with_url(url, "dba", "12?");
// PHP의 경우
url="cci:CUBRID:192.168.0.10:33000:tdb:::?charset=UTF-8";
cubrid_connect_with_url(url, "dba", "12?");
// JDBC의 경우
url="jdbc:CUBRID:192.168.0.10:33000:tdb:::?charset=UTF-8";
conn = DriverManager.getConnection(url,"dba", "12?");
```

이 주의 사항은 모든 버전에 해당한다.

CUBRIDSUS-5136 페이지 단위의 옵션 제거 예정

cubrid createdb 유틸리티의 DB 볼륨 크기 및 로그 볼륨 크기를 지정할 때 페이지 단위를 사용하는 옵션들(-p, -l, -s)은 제거될 예정이므로, 2008 R4.0 Beta 이후 새로 추가된 옵션들(--db-volume-size, --log-volume-size, --db-page-size, --log-page-size)을 사용한다.

cubrid addvoldb 유틸리티의 DB 볼륨 크기를 지정하는 경우에도 페이지 단위를 사용하지 않고 2008 R4.0 Beta 이후 새로 추가된 옵션(--db-volume-size)을 사용한다.

CUBRIDSUS-4222 DB 볼륨 크기 설정 시 주의 사항

2008 R4.0 Beta 버전부터 DB 생성 시 데이터 페이지 및 로그 페이지의 크기 기본값이 4KB에서 16KB로 변경되었으므로, DB 볼륨을 페이지 개수로 지정하여 생성하는 경우 볼륨의 바이트 크기가 기대와 다를 수 있음에 주의한다. 아무런 옵션도 주지 않을 경우 이전 버전에서는 4KB의 페이지 크기로 100MB의 DB 볼륨을 생성하였으나, 2008 R4.0 버전부터는 16KB의 페이지 크기로 512MB의 DB 볼륨을 생성하게 된다.

그리고, DB 볼륨의 생성 가능한 최소 크기를 20MB로 제한하였으므로 이보다 작은 크기의 DB 볼륨은 생성할 수 없다.

CUBRIDSUS-4222 페이지 단위의 시스템 파라미터 제거 예정

페이지 단위의 시스템 파라미터들은 추후 제거될 예정이므로 바이트 단위의 새로운 시스템 파라미터를 사용할 것을 권장한다. 관련 시스템 파라미터들에 대한 내용은 아래를 참고한다.

CUBRIDSUS-4095 일부 시스템 파라미터들의 기본값 변경

2008 R4.0부터 다음 시스템 파라미터들의 기본값이 변경되었다.

DB 서버가 허용하는 동시 연결 개수를 설정하는 **max_clients**의 기본값, 인덱스 페이지 생성 시 향후 업데이트를 대비하여 확보하는 여유 공간 비율을 설정하는 **index_unfill_factor**의 기본값이 변경되었으며, 바이트 단위 시스템 파라미터의 기본값이 기존 페이지 단위 시스템 파라미터의 기본값보다 커져서 별도의 설정을 하지 않는 경우 더 많은 메모리를 사용하게 되었다.

기존 시스템 파라미터	추가된 시스템 파라미터	기존 기본값	변경된 기본값 (단위: 바이트)
max_clients	-	50	100
index_unfill_factor	-	0.2	0.05
data_buffer_pages	data_buffer_size	100M(페이지 크기=4K)	512M
log_buffer_pages	log_buffer_size	200K(페이지 크기=4K)	4M
sort_buffer_pages	sort_buffer_size	64K(페이지 크기=4K)	2M
index_scan_oid_buffer_pages	index_scan_oid_buffer_size	16K(페이지 크기=4K)	64K

또한, **cubrid createdb**로 DB 생성 시 데이터 페이지 크기와 로그 페이지 크기의 최소값이 1K에서 4K로 변경되었다.

CUBRIDSUS-5375 시스템 파라미터를 잘못 설정하면 DB 서비스, 유틸리티 및 응응 프로그램이 구동되지 않음

cubrid.conf 또는 cubrid_ha.conf에 정의되지 않은 시스템 파라미터를 설정하거나, 페이지 단위의 시스템 파라미터와 바이트 단위의 시스템 파라미터가 동시에 사용되거나, 시스템 파라미터 값이 허용 범위를 벗어나면 이와 관련된 DB 서비스, 유틸리티 및 응응 프로그램이 구동되지 않는다.

CUBRIDSUS-4524 복제 기능 제거

CUBRID 2008 R4.0 버전부터 복제 기능이 제거되었으므로, 이중화 환경을 구축하려면 HA 기능을 사용해야 한다. 서버 버전 업그레이드 및 DB 마이그레이션을 수행한 후, HA 환경을 새롭게 구축할 수 있다. HA 환경 구축과 관련하여, 매뉴얼의 '관리자 안내서 > CUBRID HA'를 참고한다.

CUBRIDSUS-5228 CUBRID 매니저 설치 패키지 별도 제공

CUBRID 2008 R4.0부터는 CUBRID 매니저 설치 패키지를 별도로 제공하며, CUBRID 매니저를 사용하려면 CUBRID 패키지 설치 후 CUBRID 매니저를 별도로 설치해야 한다. .

CUBRIDSUS-5097 컬럼 크기보다 큰 문자열을 INSERT/UPDATE 할 때 문자열이 절삭되어 입력됨

컬럼 크기보다 큰 CHAR, VARCHAR, NCHAR, VARNCHAR 타입의 문자열을 INSERT/UPDATE하면 컬럼 크기를 초과하는 문자열 부분을 절삭한다. (매뉴얼의 'CUBRID SQL 설명서 > 데이터 타입 > 문자열 데이터 타입' 참고)

CUBRIDSUS-5349 CUBRID 32bit 버전에서 data_buffer_size에 2G를 초과하는 값을 설정하면 DB 구동에 실패함

CUBRID 32bit 버전에서 **data_buffer_size**가 2G를 초과하는 값으로 설정되는 경우 DB 구동에 실패한다. 32bit 버전에서는 OS의 한계로 인해 설정값이 2G를 초과할 수 없음에 주의한다.

CUBRIDSUS-4059 VARCHAR 타입의 컬럼에서 값을 가져올 때 커버링 인덱스가 적용되는 경우 뒤에 따르는 공백 문자열이 무시됨

VARCHAR 타입의 컬럼에서 값을 가져올 때 커버링 인덱스가 적용되는 경우 뒤에 따라오는 공백 문자열은 절삭된다. 질의 수행 시 커버링 인덱스가 적용되면 질의 결과 값을 인덱스에서 가져오는데, 인덱스에는 뒤이어 나타나는 공백 문자열을 제거한 채로 값을 저장하기 때문이다. 이러한 현상을 원하지 않을 경우에는 NO_COVERING_IDX 힌트를 지정하면 된다. (매뉴얼의 'CUBRID SQL 설명서 > 질의 최적화 > 인덱스 활용 > 커버링 인덱스' 참고)

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

-- 아래 질의는 커버링 인덱스가 적용되어 3개의 데이터가 모두 같은 조건으로 인식된다.
SELECT * FROM tab WHERE c='abcd ' USING INDEX i_tab_c(+);
c
======================
'abcd'
'abcd'
'abcd'
```

CUBRIDSUS-3757 HA 관련 주의 사항

CUBRID HA에서 트리거 및 자바 저장 프로시저를 사용할 경우 마스터 노드에서 이미 수행된 트리거 또는 자바 저장 프로시저를 슬레이브 노드에서 중복 수행하여 CUBRID HA 그룹 내의 노드 간 데이터 불일치가 발생할 수 있으므로, CUBRID HA에서는 트리거 및 자바 저장 프로시저를 사용하지 않도록 한다.

CUBRID HA는 복제 로그를 기반으로 CUBRID HA 그룹 내의 노드 간 데이터를 동기화하므로 복제 로그를 생성하지 않는 메소드를 사용하거나 CUBRID 매니저를 통해 NOT NULL 옵션 설정 작업 수행 시 CUBRID HA 그룹 내 노드 간 데이터 불일치가 발생할 수 있으므로, CUBRID HA는 메소드를 사용할 수 없고, CUBRID 매니저를 통해 작업할 수 없다.

CUBRIDSUS-5071 DB 백업/복구 시 LOB 타입 저장소는 복구되지 않음

CUBRID에서 LOB 타입의 데이터는 DB 볼륨이 아닌 별도의 저장소에 존재하므로 DB의 백업/복구 과정에 포함되지 않는다. 즉, DB의 백업 시 LOB 타입 저장소는 같이 백업되지 않으므로, 복구 시 LOB 타입 저장소는 복구되지 않는다. LOB 타입 저장소는 별도로 관리해야 한다.

CUBRIDSUS-3826 GLO 클래스 지원 중단에 따른 주의 사항

CUBRID 2008 R3.0 이하 버전은 glo(Generalized Large Object) 클래스를 사용하여 Large Object를 처리하였으나, CUBRID 2008 R3.1 이상 버전 glo 클래스를 제거하고 BLOB, CLOB 타입(이하 LOB)을 지원한다. (매뉴얼의 'CUBRID SQL 설명서 > 데이터 타입 > BLOB/CLOB 데이터 타입' 참고)

기존의 glo 클래스 사용자는 다음과 같이 작업할 것을 권장한다.

- GLO 데이터를 파일로 저장한 후 어플리케이션 및 DB 스키마에서 GLO를 사용하지 않도록 수정한다.

- DB 마이그레이션을 한다. (2008 R3.x 및 그 이전 버전에서 2008 R4.1로 마이그레이션 참고)

- 변경한 어플리케이션에 맞게 파일을 LOB 데이터로 로딩하는 작업을 수행하도록 한다.

- 수정한 어플리케이션이 정상 동작하는지 확인한다.

참고로, **cubrid loaddb** 유틸리티는 GLO 클래스를 상속받거나 GLO 클래스 타입을 가진 테이블을 로딩하려는 경우 "Error occurred during schema loading" 오류 메시지와 함께 데이터 로딩을 중지한다.

GLO 클래스의 지원 중단에 따라 각 인터페이스 별로 삭제한 함수는 다음과 같다.

인터페이스	삭제한 함수
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 BLOB, CLOB 타입 사용 시 제약 사항

BLOB, CLOB 타입(이하 LOB)에 대하여 다음과 같은 제약 사항이 있으므로 사용에 주의한다.

- LOB 타입 컬럼 간 비교 연산(=, <>, IN, NOT IN 등)을 할 수 없으며, 이를 위해서는 문자열 또는 비트열로 타입을 변환한 후 사용해야 한다. 단, IS NULL, IS NOT NULL은 지원한다.
- PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL 제약 조건 또는 인덱스를 정의할 수 없다.
- 테이블 생성 및 수정 시 SHARED 속성을 정의할 수 없으며 DEFAULT 속성은 NULL 값에 대해서만 정의할 수 있다.
- DB에는 파일의 위치(LOB Locator)가 저장되고 데이터는 파일로 저장되는 구조이므로, 장애가 발생하여 특정 시점으로 복구할 때 LOB Locator와 LOB 데이터의 매핑이 유효하지 않아 에러가 발생할 수 있다.
- ALTER TABLE DROP 문을 사용하여 컬럼을 삭제하거나, DROP TABLE 문을 사용하여 테이블을 삭제하는 경우 LOB Locator만 삭제되고 LOB 컬럼이 참조하는 외부 파일 시스템의 LOB 파일은 삭제되지 않고 남아있다.
- CUBRID가 제공하는 API나 CUBRID 매니저, CSQL을 사용하지 않고 사용자 임의로 LOB 타입의 데이터 파일을 직접 수정하면 내용이 일치됨을 보장할 수 없다.

(매뉴얼의 'CUBRID SQL 설명서 > 데이터 타입 > BLOB/CLOB 데이터 타입' 참고)

CUBRIDSUS-4186 Windows Vista 이상 버전에서 CUBRID 유틸리티를 사용한 서비스 제어 시 권장 사항

Windows Vista 이상 버전에서 **cubrid** 유틸리티를 사용하여 서비스를 제어하려면 명령 프롬프트 창을 관리자 권한으로 구동한 후 사용하는 것을 권장한다.

명령 프롬프트 창을 관리자 권한으로 구동하지 않고 **cubrid** 유틸리티를 사용하는 경우 UAC(User Account Control) 대화 상자를 통하여 관리자 권한으로 수행될 수 있으나 수행 결과 메시지를 확인할 수 없다.

Windows Vista 이상 버전에서 명령 프롬프트 창을 관리자 권한으로 구동하는 방법은 다음과 같다.

- [시작> 모든 프로그램> 보조 프로그램> 명령 프롬프트]에서 마우스 오른쪽 버튼을 클릭한다.
- [관리자 권한으로 실행(A)]을 선택하면 권한 상승을 확인하는 대화 상자가 활성화되고, “예”를 클릭하여 관리자 권한으로 구동한다.

CUBRIDSUS-3217 JDBC에서 연결 정보를 URL 스트링으로 입력하는 경우 물음표를 반드시 명시

JDBC에서 URL 스트링으로 연결 정보를 입력하는 경우 이전 버전에서는 물음표(?)를 입력하지 않더라도 속성(PROPERTY) 정보가 적용되었으나, CUBRID 2008 R3.0부터는 문법에 따라 반드시 물음표를 명시해야 하고 이를 생략할 경우 에러를 출력한다. 또한, 연결 정보 중 USERNAME과 PASSWORD가 없더라도 반드시 콜론(:)을 명시해야 한다.

```
URL=jdbc:CUBRID:127.0.0.1:31000:db1:::althosts=127.0.0.2:31000,127.0.0.3:31000  - 에러 처리
URL=jdbc:CUBRID:127.0.0.1:31000:db1:::?althosts=127.0.0.2:31000,127.0.0.3:31000  - 정상 처리
```

CUBRIDSUS-3564 마스터 프로세스와 서버 프로세스 간 프로토콜 변경 및 두 개 버전을 동시에 운영하는 경우 포트 설정 필요

마스터 프로세스(**cub_master**)와 서버 프로세스(**cub_server**) 간 통신 프로토콜 변경으로 인해 CUBRID 2008 R3.0 이상 버전의 마스터 프로세스는 하위 버전의 서버 프로세스와 통신할 수 없고, 하위 버전의 마스터 프로세스도 2008 R3.0 이상 버전의 서버 프로세스와 통신할 수 없다. 따라서, 이미 하위 버전이 설치되어 있는 환경에서 새 버전을 추가 설치하여, 두 개 버전의 CUBRID를 동시에 운영하는 경우 각각 서로 다른 포트를 사용하도록 cubrid.conf의 **cubrid_port_id** 시스템 파라미터를 수정해야 한다.

CUBRIDSUS-2828 DB 이름에 @를 포함할 수 없음

DB 이름에 @이 포함되는 경우 호스트 이름이 명시된 것으로 해석될 수 있으므로 이를 방지하기 위하여 **cubrid createdb**, **cubrid renamedb**, **cubrid copydb** 유틸리티 실행 시 DB 이름에 @를 포함할 수 없도록 수정했다.

CUBRIDSUS-3267 Windows 환경에서 디렉터리 경로 설정 시 주의 사항

Windows 환경에서 CUBRID 설치 디렉터리 경로에 공백을 포함하는 경우 정상 설치가 되지 않으므로 주의한다. 또한, **cubrid unloaddb**, **cubrid loaddb**, **cubrid backupdb** 등의 작업 대상 디렉터리 경로에도 공백을 포함할 수 없다.

CUBRIDSUS-3553 CUBRID 소스 빌드 후 실행 시, 매니저 서버 프로세스 관련 오류 발생

사용자가 직접 빌드하여 설치하는 경우, CUBRID와 CUBRID 매니저를 각각 빌드하여 설치해야 한다. 만약, CUBRID 소스만 checkout하여 빌드 후 **cubrid service start** 또는 **cubrid manager start**를 실행하면, cubrid manager server is not installed라는 오류가 발생한다.

6

CUBRID 8.3.1 릴리스 노트

6.1 CUBRID 2008 R3.1 정보

릴리스 노트 개요

본 문서는 CUBRID 2008 R3.1 및 Patch 1, 2 버전에 관한 유용한 정보를 포함한다. 릴리스 노트의 최신 버전은 CUBRID 오픈 소스 프로젝트 사이트(http://dev.naver.com/projects/cubrid)에서 확인할 수 있다.

릴리스 노트 개정 내역

CUBRID 2008 R3.1 버전의 릴리스 이후 릴리스 노트의 변경 사항은 아래와 같다.

작성 날짜	설명
2011 4월	CUBRID 2008 R3.1 Patch 2 릴리스
2011 2월	CUBRID 2008 R3.1 Patch 1 릴리스
2010 12월	CUBRID 2008 R3.1 릴리스
2010 11월	CUBRID 2008 R3.1 Beta 릴리스

릴리스 특징

CUBRID 2008 R3.1 릴리스는 이미지 또는 긴 텍스트 등의 Large Object를 저장하는 BLOB, CLOB 타입을 추가하고, 이를 사용하는 JDBC 표준 메소드 및 CCI API, SQL 함수를 추가하였다. 또한, 서비스 운영 모니터링의 편의성을 위하

여 CUBRID 매니저 유틸리티에 HA 상태 모니터링 기능을 추가하였다.

CUBRID 2008 R3.1 Patch 1은 CUBRID 2008 R3.1에서 발견된 오류를 수정하고 기능/성능을 개선한 보완 버전이며, 아래와 같은 주요 사항을 포함하고 있다. 이번 버전에서는 특히 주기적으로 보관 로그를 생성하는 작업에서 특정 시점에 DB 서버 프로세스가 멈추는(hang) 오류를 수정하였으므로, 이전 버전의 사용자는 CUBRID 2008 R3.1 Patch 1 이상으로 업그레이드할 것을 적극 권장한다. (CUBRIDSUS-4645 참고)

CUBRID 2008 R3.1 버전의 주요 특징은 다음과 같다.

BLOB, CLOB 타입 추가

이미지 또는 긴 텍스트 등의 Large Object를 저장할 수 있는 BLOB, CLOB 타입을 지원한다. BLOB, CLOB 데이터는 데이터베이스 볼륨이 아닌 외부 파일 시스템에 파일 형태로 저장된다. JDBC 표준 메소드인 getBlob/setBlob, getClob/setClob을 지원하며 CLOB_FROM_FILE, BLOB_FROM_FILE 등의 SQL 함수를 지원하여 응용 개발에 편의성을 제공한다.

기존의 GLO 클래스를 이용하여 Large Object를 저장하는 방법은 이번 버전부터 지원하지 않는다. GLO 클래스 지원 중단에 따른 주의 사항은 CUBRIDSUS-3826을 참고한다.

HA 상태 모니터링 기능 추가

CUBRID 매니저의 HA 상태 모니터링 기능을 통해 데이터베이스 서버, 브로커, HA의 운영 상태 및 시스템 자원을 감시할 수 있도록 하여 데이터베이스의 운영 편의성을 보다 강화하였다.

데이터베이스 호환성

서버 버전을 업그레이드하는 경우, 2008 R3.0 사용자는 데이터베이스 마이그레이션을 하지 않아도 된다. 단, GLO 클래스를 사용하는 경우 데이터베이스의 마이그레이션 작업을 수행하여야 한다. 보다 상세한 사항은 데이터베이스 마이그레이션 절차를 참고한다.

CUBRID 2008 R3.1의 설치 방법

Linux에서 설치

Linux용 설치 패키지는 바이너리를 포함하는 스크립트, tar.gz 압축 파일, Linux RPM 패키지 형태로 제공되며, 설치 방법은 매뉴얼의 CUBRID 시작> 설치와 실행> Linux에서의 설치와 실행을 참고한다.

Windows에서 설치

Windows용 설치 파일이 제공되며, 설치 마법사를 이용하여 설치할 수 있다. 설치 방법은 매뉴얼의 CUBRID 시작> 설치와 실행> Windows에서의 설치와 실행을 참고한다.

Windows 환경에서 설치 디렉터리 설정에 관한 주의 사항은 CUBRIDSUS-3267을 참고한다.

CUBRID 환경 변수 및 OS 환경 변수 설정

CUBRID를 사용하기 위해서는 CUBRID 환경 변수와 관련 OS 환경 변수를 설정하여야 한다. (관련 매뉴얼 참고) 특히, CUBRID 매니저와 Java 저장 프로시저를 사용하기 위해서는 Java 환경 변수를 설정하여야 한다. (관련 매뉴얼 참고)

CUBRID 2008 R3.1로 업그레이드하는 방법

2008 R3.1 Beta 버전을 사용 중인 경우 업그레이드 방법

2008 R3.1 Beta 버전을 사용하는 사용자의 경우, 정식 버전으로의 업그레이드를 적극 권장하며 업그레이드 시 별도의 데이터베이스 마이그레이션 작업은 요구되지 않는다.

업그레이드 주의 사항

기존 환경 설정 파일 보관

이전 버전의 $CUBRID/conf 디렉터리의 환경 설정 파일(**cubrid.conf**, **cubrid_broker.conf**, **cm.conf**)과 $CUBRID_DATABASES 디렉터리의 데이터베이스 위치 정보 파일(**databases.txt**)을 보관한다. [1]

데이터베이스 마이그레이션

GLO 기능을 사용하지 않는 CUBRID 2008 R3.0 사용자는 마이그레이션을 하지 않아도 된다. CUBRID 2008 R3.1은 GLO를 지원하지 않으며 LOB 타입이 GLO 기능을 대체하게 되었으므로, GLO를 이용한 응용 및 스키마는 LOB 타입에 맞게 수정하여야 한다. (아래의 2008 R3.0에서 2008 R3.1로 마이그레이션 참고)

CUBRID 2008 R3.1은 2008 R2.x 이하 버전의 데이터베이스와 호환하지 않으므로, 이전 데이터베이스를 CUBRID 2008 R3.1로 마이그레이션하여야 한다. (아래의 2008 R2.x에서 2008 R3.1로 마이그레이션 참고)

복제 또는 HA 환경 재구성

이전 버전의 복제 기능을 사용하는 시스템에서는 보다 안정적인 운영을 위해 DB 마이그레이션 및 HA 환경으로 재구성할 것을 권장한다. 또한, CUBRID 2008 R2.0, 2008 R2.1에서 제공된 Linux Heartbeat 기반의 HA 기능을 사용하는 시스템도 보다 안정적인 운영을 위해 DB 마이그레이션 및 CUBRID Heartbeat 기반의 HA 환경으로 재구성할 것을 권장한다. (아래의 HA 환경에서 데이터베이스 마이그레이션 절차 참고)

---

1 $CUBRID 또는 $CUBRID_DATABASES 형식의 환경변수는 Linux 환경에서 사용하는 형식이며 Windows 환경에서는 %CUBRID% 또는 %CUBRID_DATABASES%와 같은 형식으로 사용해야 한다.

새로 추가된 예약어 검사

CUBRID 설치 패키지 또는 다운로드 페이지에서 배포되는 CUBRID 2008 R3.1용 예약어 검출 스크립트인 **check_reserved.sql** 을 이용하여 예약어 사용 여부를 검사 할 수 있으며, 예약어로 지정된 식별자를 사용하고 있을 경우 식별자를 수정해야 한다. (관련 매뉴얼 참고)

데이터베이스 마이그레이션 절차

2008 R3.0에서 2008 R3.1로 마이그레이션

GLO 클래스 사용 여부에 따라 다음 작업을 수행

GLO 클래스를 사용하지 않는 경우, 별도의 작업 없이 2008 R3.0의 데이터베이스 볼륨을 2008 R3.1에서 그대로 사용할 수 있다.

그러나 GLO 클래스를 사용하는 경우, 2008 R3.1부터는 GLO 클래스를 지원하지 않으므로 BLOB 또는 CLOB 타입을 사용하도록 응용과 스키마를 변경해야 한다. 변경 작업이 용이하지 않다면 마이그레이션을 보류할 것을 권장한다.

BLOB 또는 CLOB 타입을 사용하려면 다음 작업을 수행

2008 R3.1부터 지원하는 BLOB 또는 CLOB 타입을 사용하기 위해서는 아래의 예제와 같이 LOB 타입의 데이터를 저장할 디렉터리를 생성한 후, 이 경로를 $CUBRID_DATABASES/databases.txt에 추가하여야 한다.[2]

```
// lob 디렉터리 생성하기
$ cd $CUBRID_DATABASES/olddb
$ mkdir lob

// 생성한 lob 디렉터리 경로를 databases.txt에 lob-base-path로 추가하기
$ cd $CUBRID_DATABASES/
$ vi databases.txt
#db-name        vol-path                        db-host         log-path
lob-base-path
r30db          /home/CUBRID/databases/olddb localhost       /home/CUBRID/databases/olddb
file:/home/CUBRID/databases/olddb/lob
```

2008 R2.x에서 2008 R3.1로 마이그레이션

2008 R2.x 사용자는 아래의 표와 같이 2008 R3.1으로 데이터베이스 마이그레이션을 해야 한다.

또한, 기존의 GLO 클래스 사용 여부에 따라 필요한 추가 작업을 하도록 한다. (위의 2008 R3.0에서 2008 R3.1로 마이그레이션 참고)

아래는 **$CUBRID/bin/migrate_r30** 유틸리티 및 다운로드 페이지에서 별도 배포되는 **check_reserved.sql** 예약어

---

2 $CUBRID 또는 $CUBRID_DATABASES 형식의 환경변수는 Linux 환경에서 사용하는 형식이며 Windows 환경에서는 %CUBRID% 또는 %CUBRID_DATABASES%와 같은 형식으로 사용해야 한다.

검출 스크립트를 이용하여 마이그레이션을 수행하는 방법이다.[3]

다른 방법으로는 **cubrid unloaddb/loaddb** 유틸리티를 사용하여 마이그레이션을 수행할 수 있다. (관련 매뉴얼과 본 문서의 5. 주의 사항> CUBRIDSUS-3826을 참고)

단계	Linux 환경	Windows 환경
C1 단계: CUBRID Service 종료	% cubrid service stop	CUBRID Service Tray>[Exit]를 선택한다.
C2 단계: 예약어 검출 스크립트 실행	예약어 검출 스크립트가 위치하는 디렉터리에서 아래 명령을 실행한다. 검출 결과를 확인하여 마이그레이션 진행 또는 식별자 수정 작업을 진행한다. (허용되는 식별자는 관련 매뉴얼 참고) % csql -S -u dba -i check_reserved.sql <r22_db_name>	
C3 단계: 이전 버전 DB 백업	이전 버전으로 복구하는 상황에 대비하기 위해 백업을 수행하고, 백업 파일을 별도 디렉터리(R22_backup)에 보관한다. (C3a) % mkdir R22_backup % cubrid backupdb -S -D R22_backup <r22_db_name> 이전 버전의 databases.txt 및 conf 디렉터리 내 설정 파일을 별도 디렉터리에 보관한다. (C3b)	
		이전 버전의 CUBRID를 제거한다.
C4 단계: 2008 R3.1 설치	본 문서의 CUBRID 2008 R3.1의 설치 방법을 참고한다.	
C5 단계: 마이그레이션 도구 실행		CUBRID Service Tray> [CUBRID Server]>[Stop] 을 선택하여 서버를 종료한다.
	(C3b)에서 보관한 databases.txt를 2008 R3.1의 설치 디렉터리에 복사한다. (C5a) 아래와 같이 migrate_r30 유틸리티를 실행한다. (C5b) % migrate_r30 <db_name>	
C6 단계: 현재 버전 DB 백업	이전 데이터베이스 버전이 2008 R1.x인 경우에만 수행한다. 이전 버전이 2008 R2.x인 경우, 이 단계를 생략할 수 있다. % cubrid backupdb -S <db_name>	
C7 단계: CUBRID 환경 설정 및 CUBRID Service 구동	환경 설정 파일을 수정한다. 이때, (C3b)에서 보관한 이전 버전의 환경 설정 파일을 사용할 수 있다. % cubrid service start % cubrid server start <db_name>	CUBRID Service Tray> [CUBRID Server]-> [Start]를 선택하여 서버를 시작한다.

HA 환경에서 데이터베이스 마이그레이션 절차

2008 R2.2 이상 버전에서 2008 R3.1로 HA 마이그레이션

아래는 브로커, 마스터 DB, 슬레이브 DB를 각각 별도 서버에 구축한 환경에서 현재 서비스를 중지하고 업그레이드를 수행하기 위한 가이드이다. 서비스 무정지 업그레이드 시나리오는 별도 가이드 문서를 참고한다.

단계	설명
H1~H6 단계: 마스터 서버에서 C1~C6 단계를 수행	마스터 서버에서 CUBRID 업그레이드 및 데이터베이스 마이그레이션을 수행하고, 2008 R3.1 데이터베이스를 백업한다.
H7 단계: 슬레이브 서버에 CUBRID 2008 R3.1 설치	설치 방법은 본 문서의 CUBRID 2008 R3.1의 설치 방법을 참고한다.
H8 단계: 마스터 백업본을 슬레이브 서버에서 복구	H6에서 생성된 마스터 서버의 2008 R3.1 데이터베이스 백업본(예: testdb_bk*)을 슬레이브 서버에서 복구한다. % scp user1@master:~/DB/testdb/testdb_bk0v000 . % scp user1@master:~/DB/testdb/log/testdb_bkvinf ./log/. % cubrid restoredb testdb
H9단계: HA 환경 재구성 후 HA모드 구동	이 단계의 작업은 root 계정으로 수행한다. HA구동 스크립트(cubrid-ha) 및 환경 설정 파일(cubrid.conf)을 설정한다. (관련 매뉴얼 참고) 참고한다. 마스터 및 슬레이브 서버에서 HA모드로 DB를 구동한다. (관련 매뉴얼 참고) [root@master ~]# service cubrid-ha start [root@slave ~]# service cubrid-ha start
H10 단계: 브로커 서버에 CUBRID 2008 R3.1 설치 및 브로커 구동	설치 방법은 본 문서의 CUBRID 2008 R3.1의 설치 방법을 참고한다. 브로커 설정 후 브로커를 시작한다. (관련 매뉴얼 참고) % cat cubrid_broker.conf ... ACCESS_MODE=RW % cubrid broker start

2008 R2.0 또는 2008 R2.1에서 2008 R3.1로 HA 마이그레이션

CUBRID 2008 R2.0 또는 2008 R2.1의 HA 기능을 사용하는 경우, 해당 버전에서 사용되었던 Linux Heartbeat 패키지를 제거하고, 서버 버전 업그레이드, 데이터베이스 마이그레이션을 수행한 후 HA 환경을 새롭게 구축하여야 한다.

아래의 H0 단계를 수행한 후, 위의 H1~H10 단계를 수행한다.

단계	설명
H0 단계: HA 관련 서비스 종료 및 기존 Linux heartbeat 제거	구동 중인 브로커를 중지한다. % cubrid broker stop 이하의 작업은 마스터 및 슬레이브 서버에서 root 계정으로 수행한다. [root@master ~]# service heartbeat stop [root@master ~]# chkconfig --del heartbeat [root@master ~]# pkill -u user1 -f "cub_master"

단계	설명
	// 슬레이브 서버에서 동일 작업 수행

복제 기능 사용 시 주의 사항

CUBRID 2008 R3.0 이상 버전부터 복제 기능에 관한 개발은 중단(Outdated and Deprecated Feature)되었으므로, 복제 기능을 사용 중인 환경에서는 보다 안정적인 운영을 위하여 HA 기능을 사용하여 이중화 환경을 구축할 것을 권장한다. 서버 버전 업그레이드 및 데이터베이스 마이그레이션을 수행한 후, HA 환경을 새롭게 구축할 수 있다.

6.2 CUBRID 2008 R3.1 Patch 2에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-4905 컬럼 개수가 8개를 초과하는 다중 컬럼 인덱스의 인덱스 스캔 시 잘못된 질의 결과를 가져오는 오류 수정

다중 컬럼 인덱스에서 컬럼 개수가 8개를 초과하는 경우, 인덱스 스캔 시 잘못된 질의 결과를 가져오는 오류를 수정하였다.

CUBRIDSUS-5196 ORDER BY 컬럼에 대해 WHERE 절에서 호스트 변수를 이용하는 조건으로 사용되는 경우 ORDER BY 최적화 수정

ORDER BY 컬럼에 대해 WHERE 절에서 호스트 변수를 이용하는 조건으로 사용되는 경우, ORDER BY 최적화 과정에서 추가된 조건이 인덱스 스캔 범위로 선택되어 실행 속도가 저하되었다. 실행 속도 개선을 위해, ORDER BY 최적화를 위해 추가된 조건은 인덱스 스캔 조건으로 선택되지 않도록 수정하였다.

ORDER BY 절에 명시한 컬럼으로 구성된 인덱스가 존재하는 경우에는 정렬 과정을 생략할 수 있기 때문에 질의 최적화기는 가상의 조건을 추가한 후 최적의 실행 계획을 만든다. 아래 예의 경우, “req_ymdt BETWEEN -무한대 AND +무한대”의 조건이 추가된다.

```
-- 기존 버전에서는 다음과 같이 ORDER BY에 있는 req_ymdt 컬럼이 WHERE절에서 호스트 변수를
이용하는 조건으로 사용되는 경우, 실행 속도가 저하되었다.
SELECT key, id, messageType, serverIpAddr, req_ymdt
FROM info
WHERE req_ymdt >= TO_TIMESTAMP(?,'YYYY"-"MM"-"DD')
  AND req_ymdt < TO_TIMESTAMP(?,'YYYY"-"MM"-"DD') + 60*60*24
```

```
ORDER BY req_ymdt desc
```

CUBRIDSUS-5051 UPDATE 문에서 VARCHAR 타입 컬럼에 정의된 크기를 초과하는 문자열 바인딩 시 데이터가 크기를 초과하여 입력되는 문제 수정

UPDATE문에서 VARCHAR 타입 컬럼에 정의된 크기를 초과하는 문자열을 바인딩하는 경우, 정의된 크기를 초과하는 데이터를 포함하여 UPDATE되는 문제가 발생하였으나 정의된 크기를 초과하는 부분을 잘라내고 UPDATE하도록 수정하였다.

CUBRIDSUS-5019 OUTER JOIN 질의문에 ORDER BY 절이 있는 경우 질의 결과에서 NULL이 제외되는 오류 수정

OUTER JOIN 질의문에 ORDER BY 절이 있는 경우, 질의 재작성이 잘못되어 결과 값에서 NULL이 제외되는 오류를 수정하였다.

```
-- 결과에 NULL값이 포함되어 있다.
SELECT b.y, f.x FROM foo f RIGHT OUTER JOIN bar b ON f.y = b.y;
y x
=========================
1 1
2 NULL

-- ORDER BY 절이 있는 경우 결과에서 NULL이 제외되는 오류가 존재하였다.
SELECT b.y, f.x FROM foo f RIGHT OUTER JOIN bar b ON f.y = b.y ORDER BY f.x;
y x
=========================
1 1
```

CUBRIDSUS-5120 cubrid addvoldb 유틸리티와 데이터베이스 볼륨의 자동 증가가 동시 수행되는 경우 서버가 멈추는(hang) 오류 수정

cubrid addvoldb 유틸리티와 데이터베이스 볼륨의 자동 증가가 동시에 수행되는 경우, 데이터베이스 서버 프로세스가 동작을 멈추는(hang) 현상이 발생하였으나 이를 수정하였다.

CUBRIDSUS-4397 cubrid unloaddb 유틸리티에 테이블 별로 데이터 파일을 생성하는 옵션 추가

cubrid unloaddb 유틸리티를 수행하는 경우, 각 테이블 별로 데이터 파일 생성이 가능하도록 --datafile-per-class 옵션을 추가하였다.

CUBRIDSUS-5106 cubrid spacedb 유틸리티에서 합산 결과를 잘못 출력하는 오류 수정

데이터베이스 볼륨 파일의 크기가 매우 큰 경우, cubrid spacedb 유틸리티의 디스크 볼륨의 총 용량 합산 과정에서 오버플로우가 발생하여 합산 용량이 잘못 출력되는 오류를 수정하였다. 또한 temp 볼륨의 용량 합산 계산이 잘못된 오류를 수정하였다.

CUBRIDSUS-4991 디스크 볼륨 정보 캐시의 내부 정보가 잘못되는 경우 데이터베이스 재시작에 실패하는 오류 수정

디스크 볼륨 정보 캐시의 내부 정보가 잘못되는 경우, 데이터베이스 서버가 비정상 종료한 이후 재시작에 실패하는 오류를 수정하였다.

CUBRIDSUS-3433 브로커 및 데이터베이스 접속 보안 기능 추가

데이터베이스 사용자 또는 클라이언트 IP를 지정하여, 브로커 프로세스 또는 데이터베이스 서버 프로세스에 접속하는 것을 제어할 수 있도록 보안 관련 기능을 추가하였다. 이와 함께 다음의 명령어가 추가되었다.

```
// BRNAME에 해당하는(생략하면 전체) 브로커에 대한 현재 accesscontrol 설정을 화면에 출력cubrid broker acl status [BRNAME]

// BRNAME에 해당하는(생략하면 전체) 브로커에 대해 accesscontrol 설정을 reload. (다시 load) 에러 발생시 예전 설정으로 계속 동작cubrid broker acl reload [BRNAME]

// servername에 해당하는 서버에 대한 현재 ipcontrol설정을 dispalycubrid server acl status servername

// servername에 해당하는 서버에 대해 ipcontrol설정을 reload. 에러 발생시 예전 설정으로 계속 동작.
cubrid server acl reload servername
```

CUBRIDSUS-4957 인덱스와 데이터 사이에 불일치가 발생하는 경우 이를 인지할 수 있도록 에러 메시지 개선

인덱스와 데이터 사이에 불일치가 발생하는 경우, 이를 인지할 수 있도록 에러 메시지를 데이터베이스 서버/클라이언트의 에러 로그 파일에 남기도록 개선하였다.

트랜잭션 격리 수준(isolation level)이 “UNCOMMITED INSTANCE”인 경우, 일시적인 데이터 불일치가 발생할 수 있기 때문에 에러 레벨을 “NOTIFICATION”으로 출력하고, “UNCOMMITED INSTANCE” 이외의 트랜잭션 격리 수준에서는 에러 레벨을 “ERROR” 로 출력한다. 에러 레벨은 error_log_level 파라미터를 통해 변경할 수 있다.

에러 로그 출력의 예는 다음과 같다.

```
-- error_log_level= NOTIFICATION 인 경우 (isolation_level이 1 또는 2로서, UNCOMMITED INSTANCE 허용)
---- 데이터베이스 서버 에러 로그
Time: 03/15/11 15:20:31.804 - NOTIFICATION *** CODE = -545, Tran = 1, CLIENT = cdbs034.cub:csql(3926), EID = 3
Internal error: INDEX u_foo_i ON CLASS foo (CLASS_OID: 0|550|8). Key and OID: 0|600|16 entry on B+tree: 0|209|590 is incorrect. The object does not exist.

-- error_log_level= ERROR 인 경우 (isolation_level이 3 이상)
---- 데이터베이스 서버 에러 로그
Time: 03/15/11 15:14:35.907 - ERROR *** ERROR CODE = -545, Tran = 1, CLIENT = cdbs034.cub:csql(3776), EID = 1
Internal error: INDEX u_foo_i ON CLASS foo (CLASS_OID: 0|550|8). Key and OID: 0|600|2 entry on B+tree: 0|209|590 is incorrect. The object does not exist.
---- 클라이언트 에러 로그
ERROR: Internal error: INDEX u_foo_i ON CLASS foo (CLASS_OID: 0|550|8). Key and OID: 0|600|2 entry on B+tree: 0|209|590 is incorrect. The object does not exist.
```

CUBRIDSUS-4694 데이터베이스 서버 프로세스의 비정상 종료 후 특정 상황에서 데이터베이스 복구에 실패할 가능성이 있는 오류 수정

데이터베이스 서버 프로세스가 비정상 종료한 이후 특정 상황에서 데이터베이스 복구에 실패할 가능성이 있는 오류를 수정하였다.

CUBRIDSUS-4962 데이터베이스 임시 볼륨 공간이 추가로 필요하여 일반 볼륨으로 자동 확장 시 서버 프로세스가 비정상 종료하는 오류 수정

질의 처리 및 정렬(sorting)을 수행할 때 일시적으로 사용되는 임시 볼륨 파일보다 큰 공간이 필요하여 일반 볼륨으로 자동 확장되는 경우, 데이터베이스 서버 프로세스가 비정상 종료하는 오류가 발생하였으나 이를 수정하였다.

CUBRIDSUS-4975 브로커 프로세스가 클라이언트에 작업 할당했다는 메시지 전달 실패 시 60초간 다음 작업을 진행하지 못하는 문제 수정

브로커 프로세스가 브로커 응용 서버(CAS)에 작업 할당을 성공했다는 메시지를 클라이언트에 전달하지 못하는 경우, 60초간 대기 후 다시 동작하는 문제를 수정하였다. 60초는 소켓 타임아웃 시간이다.

수정 이후 버전에서는 브로커 서버가 브로커 서버가 직접 클라이언트로 메시지를 전송하지 않고 CAS가 전송하도록 하여 브로커 프로세스는 즉시 다음 작업을 수행할 수 있다.

CUBRIDSUS-4903 브로커 응용 서버(CAS)에서 트랜잭션을 커밋하는 경우 질의 결과 셋을 정리하도록 수정

브로커 응용 서버(CAS)에서 트랜잭션을 커밋하는 경우, 질의 결과 셋을 가리키는 주소를 정리하지 않아 이 주소가 다른 요청에 의해 사용되는 문제가 발생하였으나 브로커 응용 서버(CAS)에서 트랜잭션을 커밋하면서 질의 결과 셋을 정리하도록 수정하였다.

CUBRIDSUS-3985 HA 환경에서 HA 재구성 방법 개선

HA 환경에서 HA 재구성을 위한 스크립트(ha_make_slavedb.sh)를 제공하여 절차를 간소화하도록 하였다. 보다 자세한 설명은 $CUBRID/share/scripts/ha/README을 참고한다.

CUBRIDSUS-4971 HA 환경에서 복제 지연 발생 시 서버 프로세스가 비정상 종료 후 재시작에 실패할 수 있는 오류 수정

HA 환경에서 복제 지연이 발생하는 경우, 로그 전달 스레드와 트랜잭션 처리 스레드가 동시에 보관 로그 파일에 접근하면서 데이터베이스 서버 프로세스가 비정상 종료한 이후 재시작에 실패하는 경우가 존재하였으나 이를 수정하였다.

로그 전달 스레드는 데이터베이스 트랜잭션 로그를 복사하여 다른 서버로 전달하는 역할을 하며, 트랜잭션 처리 스레드는 사용자가 요청한 트랜잭션을 처리하는 역할을 한다.

CUBRIDSUS-4797 HA 환경에서 복제 지연 상태로 HA를 재시작하는 경우 복제 불일치가 발생하는 오류 수정

HA 환경에서 복제 지연이 있는 상태로 HA를 재시작하는 경우, 재시작된 copylogdb 프로세스가 내부 정보를 잘못 초기화하여 비정상 종료하면서 복제 불일치가 발생하는 오류를 수정하였다.

CUBRIDSUS-3984 HA 환경에서 에러 메시지 수정

HA 환경에서 발생하는 에러 메시지 내용을 상황에 맞도록 수정하고, 에러 메시지 관련 문제들을 수정하였다.

수정한 에러 메시지 관련 문제는 다음과 같다.

한글 에러 메시지 추가

LANG=ko_KR.utf8 또는 LANG=ko_KR.euckr 인 경우, backupdb를 수행하면 서버 core 발생하는 문제 수정

디버그 메시지 제거

CUBRIDSUS-3980 HA 환경에서 여러 대의 슬레이브 서버 구성 시 발생할 수 있는 복제 불일치에 대처할 수 있도록 개선

HA 환경에서 한 대의 마스터 서버와 여러 대의 슬레이브 서버를 구성하는 경우, 복제 불일치 발생이 가능한 상황에 대처하기 위해, 로그 복제 copylogdb 프로세스 및 applylogdb 프로세스가 복제 변경 사항을 로그에 반영하도록 하고 copylogdb 프로세스의 종료 시 시그널에 의한 처리 방식 및 복제 지연 시 동작 방식을 개선하였다.

CUBRIDSUS-3977 HA 환경에서 copylogdb 프로세스의 ASYNC 모드 동작 방식 개선

HA 환경에서 ASYNC 모드인 copylogdb 프로세스가 비정상 종료하는 경우, copylogdb 프로세스가 마스터 서버로부터 받은 트랜잭션은 디스크에 항상 반영(파일 동기화)할 수 있도록 개선하였다.

CUBRIDSUS-3986 HA 환경에서 트랜잭션 로그 복제 상태와 반영 상태를 출력하는 cubrid applyinfo 유틸리티 추가로 운영 편의성 개선

HA 환경에서 트랜잭션 로그 복제 상태와 반영 상태를 출력하는 cubrid applyinfo 유틸리티를 추가하여 운영 편의성을 개선하였다.

```
$ cubrid applyinfo
applyinfo: display CUBRID HA Apply information.
usage: cubrid applyinfo [OPTION] database-name

valid options:
  -r, --remote-host-name      remote host name; display remote node's active log information
  -a, --applied-info          display applied infomation
  -L, --copied-log-path=PATH  path of copied log volumes; display copied log information
  -p, --pageid=ID             page id; default : 0(active page)
  -v, --verbose               enable verbose status messages; default : disable
```

CUBRIDSUS-3983 HA 환경에서 하나의 트랜잭션 내 다수의 복제 로그 반영 중 발생하는 오류에 대한 처리 방법 개선

HA 환경에서 하나의 트랜잭션 내 다수의 복제 로그를 반영하는 중 오류가 발생하는 경우, 오류 발생 이후 트랜잭션 내 복제 로그를 무시하면서 트랜잭션의 일부만 반영되는 문제가 존재하였으나 오류가 발생하는 경우에도 모든 로그 반영을 시도하도록 개선하였다.

CUBRIDSUS-3180 HA 구동 스크립트(cubrid-ha)의 개선

HA 구동 스크립트인 cubrid-ha를 개선하여, 복제 로그 경로를 사용자가 지정 가능하도록 하였으며 명령어 실패 시 적절한 에러 메시지가 출력될 수 있도록 수정하였다.

CUBRIDSUS-3971 HA 환경에서 applylogdb 프로세스가 비정상 종료 후 재시작 시 복제 로그 반영이 누락될 가능성이 발생하지 않도록 개선

HA 환경에서 applylogdb 프로세스가 비정상 종료 후 재시작하는 경우, 복제 로그 반영이 누락될 가능성이 존재하였으나 이를 발생하지 않도록 개선하였다.

CUBRIDSUS-3885 HA 환경에서 cubrid changemode 유틸리티를 이용하여 HA 서버의 상태 변경 시 변경 가능한 상태를 제한하도록 수정

HA 환경에서 cubrid changemode 유틸리티를 이용하여 HA 서버의 상태를 변경하는 경우, standby에서 maintenance로 변경하거나 maintenance에서 standby로 변경하는 경우만 허용하도록 수정하였다.

CUBRIDSUS-3928 HA 환경에서 보관 로그 파일 개수가 log_max_archives를 초과하여도 슬레이브 서버에 전달되지 않은 로그는 보존하도록 수정

HA 환경에서 보관 로그 파일의 개수가 시스템 파라미터에서 설정한 log_max_archives 개수를 초과하는 경우, 슬레이브 서버에 전달되지 않은 로그가 삭제될 수 있었으나 이러한 로그는 보존하도록 수정하였다.

CUBRIDSUS-5209 HA 환경에서 applylogdb 프로세스의 메모리 누수 오류 수정

HA 환경에서 applylogdb 프로세스가 메모리 누수로 인하여 메모리를 과다 사용하게 되는 오류를 수정하였다.

6.3 CUBRID 2008 R3.1 Patch 1에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-4372 cubrid broker 유틸리티에 reset 기능 추가

HA 에서 failover 등으로 브로커 응용 서버(CAS)가 원하지 않는 데이터베이스 서버에 연결되었을 때, 기존 연결을 끊고 새롭게 연결할 수 있도록 cubrid broker 유틸리티에 reset 기능을 추가하였다.

예를 들어 Read Only 브로커가 active server와 연결된 후에는 standby server가 연결이 가능한 상태가 되더라도 자동으로 standby server와 재연결하지 않으며, "cubrid broker reset" 명령을 통해야만 기존 연결을 끊고 새롭게 standby server와 연결할 수 있다.

CUBRIDSUS-4321 브로커 파라미터인 SQL_LOG 값을 동적으로 변경하는 명령어에 특정 응용 서버(CAS) 파라미터만 변경할 수 있도록 옵션 추가

broker_changer 명령어로 브로커의 재시작 없이 특정 응용 서버(CAS)만 SQL_LOG 값을 변경하는 것이 가능하도록 아래의 형식과 같이 응용 서버 식별자(<cas_id>)를 지정할 수 있는 기능을 추가하였다. <cas_id>는 cubrid broker status 명령어에서 출력되는 ID이다.

```
broker_changer <broker_name> [<cas_id>] SQL_LOG <value>
```

참고로 broker_changer는 Linux 버전에서만 사용이 가능하다.

CUBRIDSUS-4736 broker_log_runner 명령어에서 질의 계획을 포함하도록 하는 옵션 추가

broker_log_runner 명령어에서 결과 파일을 저장하는 -o 옵션을 사용하는 경우, 결과 파일에 질의 계획을 포함하도록 하는 -Q 옵션을 추가하였다.

```
% broker_log_runner -I 192.168.1.10 -P 30000 -d demodb -o result -Q query_convert.in
```

CUBRIDSUS-4134 broker_log_top 명령어에서 시간 범위의 입력 방식 및 처리 성능 개선

broker_log_top으로 브로커의 로그를 확인할 때 시간 범위를 지정하는 옵션인 -F(시작 시간)와 -T(끝 시간)의 시간 형식에 밀리초(msec)까지 설정할 수 있도록 하였고, 시간 범위가 주어진 경우 처리 성능을 개선하였다.

참고로 시간 범위의 입력 형식은 “MM[/DD[hh[:mm[:ss[.msec]]]]]”을 따르며 []로 감싼 부분은 생략이 가능하다. 생략되면 DD는 01이, 나머지 부분은 0이 기본값으로 정해진다.

다음 예는 broker_log_top을 이용하여 지정한 시간 범위의 브로커 로그 분석 결과 파일을 생성하도록 한다.

```
# 밀리 초까지 검색 범위를 설정한다.
broker_log_top -F "01/19 15:00:25.000" -T "01/19 15:15:25.180" log1.log

# 아래의 옵션 값에서 시간 형식이 생략된 부분은 기본값 0으로 정해진다. 즉, -F "01/19
00:00:00.000" -T "01/20 00:00:00.000" 을 입력한 것과 같다.
broker_log_top -F "01/19" -T "01/20" log1.log
```

CUBRIDSUS-4440 호스트 변수에 잘못된 값을 바인딩하여 질의 수행에 실패하면 정상 값을 바인딩하여 재수행해도 계속 실패하는 오류 수정

플랜 캐시를 사용하지 않는 질의에서 호스트 변수에 잘못된 값을 바인딩한 뒤 질의 수행에 실패하는 경우, 이후 정상적인 값을 바인딩하고 수행해도 계속 실패하는 현상을 수정하였다.

참고로 데이터베이스 서버 기본 설정에서는 플랜 캐시를 사용하도록 max_plan_cache_entries 파라미터 값이 설정되어 있지만, INSERT 문에서는 설정 값과 무관하게 항상 플랜 캐시를 사용하지 않는다. (플랜 캐시와 관련하여 매뉴얼의 쿼리 캐시 관련 파라미터에서 max_plan_cache_entries 참고)

CUBRIDSUS-4656 JDBC의 getDriverVersion() 호출 시 잘못된 버전 번호를 반환하는 오류 수정

JDBC API인 getDriverVersion()을 호출하는 경우, 잘못된 버전 문자열을 반환하는 오류를 수정하였다.

CUBRIDSUS-4645 로그 페이지 버퍼의 제어 방식을 변경하여 DB 서버 프로세스의 멈춤 오류 수정

데이터베이스 시스템에서 실행되는 여러 쓰레드들이 로그 페이지 버퍼를 사용하고자 할 때 이를 제어하는 방식을 변경하여, 주기적으로 보관 로그를 생성하는 작업에서 로그 페이지 버퍼를 사용하려는 특정 시점에 DB 서버 프로세스가 동작을 멈추는(hang) 오류를 수정하였다.

CUBRIDSUS-4341 응용 프로그램의 연결 요청이 집중되는 경우 브로커 연결에 실패할 수 있는 오류 수정

응용 프로그램의 연결 요청이 브로커 프로세스(cub_broker)에 일시적으로 집중되는 경우, 브로커 연결에 실패할 수 있는 오류를 수정하였다.

CUBRIDSUS-4188 브로커의 SQL 로그 파일에 기록되는 질의 실행 시간의 오류 수정

브로커의 SQL 로그 파일에 기록되는 질의 실행 시간이 트랜잭션 시작 시점부터 질의가 끝난 시점까지의 시간으로 잘못 표시되는 오류를 수정하였다. 이 오류는 CUBRID 2008 R3.0부터 발생하였다.

CUBRIDSUS-4471 php 드라이버의 사용이 가능하도록 php 드라이버 소스 코드에서 glo 관련부분 제거

glo 관련 기능을 지원하지 않게 됨에 따라 contrib/php에 있는 php 드라이버 소스에서 glo와 관련된 코드를 삭제하여 php 드라이버의 사용이 가능하도록 수정하였다.

6.4 CUBRID 2008 R3.1에서 변경된 사항

새로 추가된 기능

CUBRIDSUS-3652 BLOB/CLOB 타입의 추가 및 관련 API 지원

이미지 또는 긴 텍스트 등의 Large Object를 저장할 수 있는 데이터 타입으로 BLOB/CLOB을 제공한다. BLOB은 바이너리 데이터를 저장하기 위한 타입이며 CLOB은 문자열 데이터를 저장하기 위한 타입이다.

BLOB/CLOB 타입의 데이터가 삽입되면 데이터는 외부 파일 시스템에 파일로 저장되고, 해당 파일의 경로 정보(**LOB** locator)는 CUBRID 데이터베이스에 저장된다.

BLOB/CLOB 데이터가 저장되는 최상위 디렉터리는 데이터베이스 생성 시 지정할 수 있으며 지정하지 않을 경우 데이터베이스 볼륨이 생성되는 경로의 lob 디렉터리 아래에 저장된다. (관련 매뉴얼 참고)

BLOB/CLOB 타입과 관련하여 다음의 표와 같은 SQL 함수, JDBC 메소드, CCI API가 추가되었다.

대상	목록	비고
SQL 함수	CLOB_TO_CHAR BLOB_TO_BIT CHAR_TO_CLOB BIT_TO_BLOB CHAR_TO_BLOB CLOB_FROM_FILE BLOB_FROM_FILE CLOB_LENGTH BLOB_LENGTH	관련 매뉴얼 참고
JDBC	CUBRIDConnection.createBlob ResultSet.setBlob ResultSet.getBlob	관련 매뉴얼 참고
CCI	cci_blob_new cci_blob_size cci_blob_write cci_blob_read cci_blob_free	관련 매뉴얼 참고

아래는 BLOB/CLOB 타입을 이용한 SQL 예제이다.

```
// 테이블 생성
CREATE TABLE lob_tbl (id int PRIMARY KEY, content CLOB, image BLOB);
// 데이터 입력
INSERT INTO lob_tbl VALUES (1, CHAR_TO_CLOB('This is a Dog'),
BLOB_FROM_FILE('/home1/data1/doc_image.jpg'));
// 데이터 조회
SELECT id, content, image FROM lob_tbl
 id  content     image
========================================================================
1  file:/data1/ces_522/lob_tbl.00001282208855807171_7329
                 file:/data1/ces_393/lob_tbl.00001282208855809474_7474
```

BLOB/CLOB 관련 제약사항은 본 문서의 CUBRIDSUS-3826 GLO 클래스 지원 중단에 따른 주의 사항을 참고한다.

CUBRIDSUS-434 JDBC/CCI에서 Foreign Key 정보를 얻을 수 있는 기능 추가

JDBC와 CCI API에 다음과 같이 Foreign Key 관련 정보를 얻을 수 있는 기능을 추가하였다.

대상	목록	비고
JDBC	**DatabaseMetaData클래스의 관련메소드지원** getImportedKeys getExportedKeys getCrossReference	관련 매뉴얼 참고
CCI	**cci_schema_info 함수의 관련 타입 확장** CCI_SCH_IMPORTED_KEYS CCI_SCH_EXPORTED_KEYS CCI_SCH_CROSS_REFERENCE	관련 매뉴얼 참고

CUBRIDSUS-3788 ODBC 드라이버에 스키마 정보를 얻을 수 있는 기능 추가

ODBC 드라이버에 다음과 같이 Primary Key, Foreign Key 등 스키마 관련 정보를 얻을 수 있는 기능을 추가하였다.

대상	목록	비고
ODBC	SQLForeignKeys SQLPrimaryKeys SQLProcedures SQLProcedureColumns SQLTablePrivileges	관련 매뉴얼 참고

CUBRIDSUS-4091 ODBC 드라이버에 UTF-8 문자셋 지원

ODBC 드라이버에 UTF-8 문자셋 지원 기능을 추가하여 UTF-8 문자열로 입력 및 조회가 가능하도록 개선하였다. UTF-8 이외에 다른 문자셋은 지정할 수 없으며, 아래의 예와 같이 연결 문자열에 “CHARSET=utf-8” 속성을 추가하는 방식으로 사용할 수 있다.

```
// 연결 문자열에 CHARSET 속성을 UTF-8로 설정
DRIVER=CUBRID Driver;UID=public;FETCH_SIZE=100;PORT=30000;SERVER=localhost;DB_NAME=demodb;DESCR
IPTION=cubrid_odbc;CHARSET=utf-8
```

CUBRIDSUS-3879, 4183 브로커의 상태 정보 출력 기능에 브로커 응용 서버(CAS) 정보 포함 옵션 추가

브로커 상태 정보 출력을 위한 cubrid broker status -b 명령 사용 시 -f 옵션을 추가로 사용하면 브로커 응용 서버(CAS) 정보를 포함하도록 개선하였다.

-f 옵션에 -I 옵션을 추가하면 지정한 초 동안 대기 상태인 응용 서버의 개수(Ns-W)와 수행 상태인 브로커 응용 서버(CAS)의 개수(Ns-B)를 출력한다. (-I 옵션을 생략하면 기본값은 1초이다.)

새로 추가한 브로커 응용 서버(CAS) 정보는 다음과 같다.

항목 이름	설명
T	실행 중인 응용 서버(CAS)의 전체 개수
W	현재 클라이언트 대기(Waiting) 상태인 응용 서버(CAS) 개수
B	현재 클라이언트 수행(Busy) 상태인 응용 서버(CAS) 개수
Ns-W	N초 동안 클라이언트 대기(Waiting) 상태였던 응용 서버(CAS) 개수
Ns-B	N초 동안 클라이언트 수행(Busy) 상태였던 응용 서버(CAS) 개수
CANCELED	브로커 구동 후 사용자 인터럽트로 취소된 질의 개수 (-s 옵션 추가 시 지정한 초 동안 누적된 개수)
ACCESS_MODE	브로커의 ACCESS_MODE 설정
SQL_LOG	브로커의 SQL_LOG 설정

```
// 브로커 상태 정보 출력 시 -f 옵션 추가. -l 옵션은 N초 동안의 Ns-W, Ns-B를 출력하도록 초를 설정
% cubrid broker status -b -f -l 2
@ cubrid broker status
NAME          PID    PSIZE PORT  AS(T W B 2s-W 2s-B) JQ REQ TPS QPS LONG-T LONG-Q ERR-Q
CANCELED
ACCESS_MODE SQL_LOG
================================================================================
query_editor 16784 56700 38000    5 0 0 0     0      0  0    0    0   0/60.0 0/60.0 0      0
          RW       ALL
```

개선된 사항

CUBRIDSUS-3761 CCI에서 응용과 브로커 간 네트워크 상태를 확인하는 방식 개선

CCI(C API)를 이용한 응용과 브로커 간의 네트워크 연결 상태를 주기적으로 검사하기 위하여 기존에는 Echo 포트를 사용하였으나, 브로커의 포트를 사용하도록 변경하였다.

CUBRIDSUS-3343 브로커와 JDBC 응용 프로그램 간 최대 응답 시간을 줄이도록 개선

브로커 응용 서버(CAS)와 응용 프로그램 사이에서 사용하는 네트워크 버퍼 크기를 줄여 JDBC 응용 프로그램의 최대 응답 시간이 줄어들도록 개선하였다.

JDBC 드라이버는 네트워크 버퍼를 통해 얻은 질의 수행 결과를 Java 객체로 생성하여 관리한다. 이때 생성된 객체가

차지하는 메모리 영역으로 인해 자주 JVM의 Full GC(Garbage Collection)가 발생하여 응답 시간이 늘어날 수 있으므로 네트워크 버퍼의 크기를 기존의 128K에서 16K로 줄여 Full GC의 발생 빈도가 낮아지도록 하였다.

이 수정은 CCI, PHP, ODBC 응용 프로그램에도 적용되지만, JDBC 응용 프로그램과 같이 성능 상의 이익이 발생하지는 않는다.

CUBRIDSUS-4027 브로커의 ACCESS_MODE를 동적으로 변경할 수 있도록 개선

broker_changer 유틸리티를 사용하여 특정 브로커의 ACCESS_MODE 값을 변경하고 해당 브로커는 자동으로 reset 되도록 개선하였다.

```
//ACCESS_MODE를 RO로 변경하고 브로커 reset
% broker_changer broker_name access_mode ro

//ACCESS_MODE를 RW로 변경하고 브로커 reset
% broker_changer broker_name access_mode rw
```

Windows 환경에서는 브로커 파라미터의 동적 변경 기능을 사용할 수 없다. (3. CUBRID 2008 R3.1에서 변경된 사항> CUBRIDSUS-3926 참고)

CUBRIDSUS-3835 보존할 보관 로그 파일의 최대 개수를 동적으로 변경할 수 있도록 개선

데이터베이스를 재시작하지 않고 보존할 보관 로그 파일의 최대 개수를 동적으로 변경할 수 있도록 개선하였다. 데이터베이스 구동 시에는 cubrid.conf에서 사용된 log_max_archives 파라미터의 값이 보존할 보관 로그 파일의 최대 개수로 설정된다.

아래의 예와 같이 csql을 이용하여 변경하는 경우에는 데이터베이스의 dba 계정으로 접속해야 한다.

```
//dba 계정으로 실행한 csql에서 log_max_archives 값을 동적으로 변경
csql>;set log_max_archives=5
```

CUBRIDSUS-3771 deadlock 및 lock timeout 발생 시 데이터베이스 서버의 에러 메시지 개선

데이터베이스 서버의 에러 로그 파일에 deadlock 발생 시 deadlock 메시지를 추가하고, lock timeout 발생 시 기존의 lock timeout 메시지에 lock timeout의 원인을 제공한 프로세스 이름과 프로세스 아이디를 출력하도록 개선하였다.

```
// deadlock 이 발생하는 경우, deadlock 메시지와 함께 이를 유발한 호스트/프로세스 정보 출력
Time: 08/23/10 12:06:11.215 - NOTIFICATION *** CODE = -993, Tran = 0, CLIENT =
(unknown):(unknown)(-1), EID = 7
deadlock cycle is detected. public@cdbs006.cub|csql(19502), public@cdbs006.cub|csql(19501),
```

```
public@cdbs006.cub|csql(19500).

// lock timeout이 발생하는 경우, 에러 메시지의 제일 끝에 프로세스 이름과 프로세스 아이디를 출력
Time: 08/23/10 12:06:11.215 - ERROR *** ERROR CODE = -967,Tran = 3, CLIENT =
cdbs006.cub:csql(19502), EID = 8
Your transaction (index 3, public@cdbs006.cub|19502) timed out waiting on IS_LOCK lock on class
tb1 because of deadlock. You are waiting for user(s) public@cdbs006.cub|csql(19500) to finish.
```

CUBRIDSUS-4011 Debian 패키지 빌드가 가능하도록 개선

Ubuntu를 포함한 Debian 계열 Linux에서 deb 패키지 빌드가 가능하도록 관련 파일을 추가하였다. 그리고, Ubuntu Linux의 최신 버전(10.10 maverick)에서 기본 제공하는 bison 2.4는 CUBRID 빌드 시 사용하는 bison 2.3과 호환되지 않기 때문에 bison 2.3의 소스를 CUBRID 소스 패키지에 포함하여 빌드 시 이를 이용하도록 개선하였다.

수정된 사항

CUBRIDSUS-3698 IF 함수의 인자에 NULL이 주어지면 항상 NULL을 반환하는 오류 수정

IF 함수에서 첫 번째 인자로 주어진 조건식에 컬럼이 포함되고 두 번째 혹은 세 번째 인자로 주어진 결과값에 NULL이 주어지면, 조건의 참, 거짓에 관계없이 항상 NULL을 반환하는 오류를 수정하였다.

```
CREATE TABLE t1( i1 INTEGER);
INSERT INTO t1 VALUES(1),(2);
SELECT i1,IF ( i1 > 1, i1, NULL) FROM t1; -- 첫 번째 레코드는 거짓, 두 번째 레코드는 참이나
모두 NULL을 반환
```

CUBRIDSUS-4047 RIGHT OUTER JOIN이 MERGE 조인으로 실행될 경우 결과가 잘못 나오는 오류 수정

아래의 예와 같이 RIGHT OUTER JOIN 질의를 MERGE 조인으로 실행하는 경우, 결과가 잘못 나오는 오류를 수정하였다.

```
select /*+ USE_MERGE */ * from foo right outer join bar on foo.i = bar.i
```

CUBRIDSUS-4193 LIKE 조건식에서 지정한 escape 문자가 검색 문자열 내에 존재하면 발생하는 오류 수정

LIKE 조건식에 ESCAPE 구문이 있는 경우, wildcard(_, %) 이외의 문자 앞에 escape 문자가 존재하면 오류가 발생하였으나, wildcard(_, %)를 포함한 모든 문자 앞에 escape 문자가 존재하면 뒤의 문자를 일반 문자로 처리하여 오류가 발생하지 않도록 수정하였다.

```
// escape 문자 \를 사용하여 질의를 수행하는 예. path 컬럼의 값이 "c:\path_"로 시작하는 레코드를 검색
SELECT * FROM tbl WHERE path LIKE 'c:\\path\_' ESCAPE '\';
```

CUBRIDSUS-4176 계층 질의문에서 출력할 컬럼의 타입이 CHAR 혹은 NUMERIC이면 발생하는 오류 수정

계층 질의문에서 출력할 컬럼의 타입이 CHAR 혹은 NUMERIC처럼 고정 길이를 지정하는 타입인 경우, Query execution failure #13511라는 에러 메시지가 발생하는 오류를 수정하였다.

```
// 데이터 생성
CREATE TABLE tbl (id VARCHAR(20), parent_id VARCHAR(20),use_yn CHAR(1));
INSERT INTO tbl VALUES('AFAC002', 'AFA','Y'), ('AFAC018', 'AFAC02','Y'), ('AFAC019',
'AFAC002','Y'), ('AFAC020', 'AFAC002','Y'), ('AFAC022', 'AFAC02','Y');

// 계층 질의문 수행
SELECT id, parent_id, use_yn FROM tbl
START WITH ID='AFAC002' CONNECT BY NOCYCLE PRIOR id=parent_id;

// 기존 버전에서 다음의 에러 발생
Execute: Query execution failure #13511
```

CUBRIDSUS-4177 계층 질의문에서 INLINE VIEW 외부에 WHERE 조건이 있을 경우 잘못된 결과를 출력하는 오류 수정

INLINE VIEW가 포함된 계층 질의문에 WHERE 절이 있는 경우, INLINE VIEW 외부의 WHERE 조건이 내부에 적용되어 잘못된 결과를 출력하는 오류를 수정하였다.

아래의 예를 보면, 기존 버전에서는 inline_view 외부에 WHERE id <> 1 이라는 조건이 inline_view 내부에 적용되어 inline_view의 결과에 id가 1인 레코드가 빠지게 되고, 이 결과를 가지고 START WITH id=1 인 조건을 수행하면서 최종 결과는 0건이 되는 오류가 존재한다.

```
// 테이블 생성 및 데이터 입력
CREATE TABLE tree(ID INT, MgrID INT, Name VARCHAR(32));
INSERT INTO tree VALUES (1,NULL,'Kim');
INSERT INTO tree VALUES (2,NULL,'Moy');
INSERT INTO tree VALUES (3,1,'Jonas');
INSERT INTO tree VALUES (4,1,'Smith');
INSERT INTO tree VALUES (5,2,'Verma');
INSERT INTO tree VALUES (6,2,'Foster');
INSERT INTO tree VALUES (7,6,'Brown');

// 계층 질의문 수행 시 WHERE id <> 1 조건이 inline_view 안에서 수행되던 오류를 수정함
SELECT id,mgrid,name
FROM (
SELECT id,mgrid,name FROM tree
)inline_view
WHERE id <> 1
START WITH id = 1
CONNECT BY PRIOR id = mgrid;

// 기존 버전에서 수행 시 결과 0건
// 수정된 버전에서 수행 시 결과 2건
3 1 Jonas
4 1 Smith
```

CUBRIDSUS-4178 계층 질의문에서 ORDER SIBLINGS BY 정렬이 잘못되는 오류 수정

계층 질의문에서 ORDER SIBLINGS BY에 의한 정렬을 수행하는 경우, 같은 계층의 레코드, 즉 SIBLINGS가 제대로 정렬이 되지 않는 오류를 수정하였다.

ORDER SIBLINGS BY <컬럼>에 의한 정렬은 최상위 레벨의 레코드에서 <컬럼> 값의 순서가 가장 먼저 오는 것을 출력하고 그 뒤를 이어 자식 레코드를 출력하는 깊이 우선 탐색을 할 때 같은 부모 레코드를 가지는 자식 레코드들을 <컬럼> 값의 순서대로 출력하도록 한다.

```
// ORDER SIBLINGS BY seq에 의해 정렬되는 예
// 같은 최상위 레벨의 레코드인 A01, A02, A03 중 seq 값이 가장 작은 A02를 먼저 출력하고,
// 이어서 A02의 자식 레코드를 한다. 이때 같은 레벨의 자식 레코드 중 seq값이 작은 A0201을 먼저 출력하고,
// A0202를 나중에 출력한다.

CREATE TABLE tbl(id VARCHAR(10), pid VARCHAR(15),  name VARCHAR(15) , seq INT);
```

```
INSERT INTO tbl VALUES('A01', 'A00', 'MENU01', 3);
INSERT INTO tbl VALUES('A02', 'A00', 'MENU02', 1);
INSERT INTO tbl VALUES('A0101', 'A01', 'MENU0101', 2);
INSERT INTO tbl VALUES('A0102', 'A01', 'MENU0102', 1);
INSERT INTO tbl VALUES('A0202', 'A02', 'MENU0202', 2);
INSERT INTO tbl VALUES('A0201', 'A02', 'MENU0201', 1);
INSERT INTO tbl VALUES('A03', 'A00', 'MENU03', 2);

SELECT id, pid, name, LEVEL, seq
FROM tbl
START WITH pid = 'A00'
CONNECT BY PRIOR id = pid
ORDER SIBLINGS BY seq;

id                    pid                   name                          LEVEL          seq
=============================================================================================
  'A02'                 'A00'                 'MENU02'                        1            1
  'A0201'               'A02'                 'MENU0201'                      2            1
  'A0202'               'A02'                 'MENU0202'                      2            2
  'A03'                 'A00'                 'MENU03'                        1            2
  'A01'                 'A00'                 'MENU01'                        1            3
  'A0102'               'A01'                 'MENU0102'                      2            1
  'A0101'               'A01'                 'MENU0101'                      2            2
```

CUBRIDSUS-4004 기본 키 생성 실패 후 COMMIT을 수행하면 해당 테이블 조회 시 에러가 발생할 수 있는 오류 수정

기본 키(Primary Key) 제약 조건을 추가하는 ALTER TABLE문이 실패한 상태에서 해당 트랜잭션을 커밋하면 서버에 캐시된 변경 내역을 불완전 롤백하게 되고, 이후 해당 테이블에 UPDATE문을 실행하면 데이터가 잘못 수정되는 오류가 발생한다. 이로 인해 데이터베이스 서버를 재시작한 이후 해당 테이블을 조회할 때 ERROR: Unknown representation identifier라는 에러 메시지가 출력되었으나, 이를 수정하였다.

CUBRIDSUS-3538 메타 데이터를 얻어오는 JDBC 메소드에서 ArrayIndexOutOfBoundsExceptions 예외가 발생하는 오류 수정

JDBC의 DatabaseMetaData 클래스 내에 정의된 getProcedure, getProcedureColumns, getColumns, getColumnPrivileges, getTablePrivileges, getImportedKeys, getExportedKeys, getCrossReference, getTypeInfo, getIndexInfo, getAttributes 메소드를 통해 얻은 ResultSet에서 사용하는 내부 정보가 잘못 기록되어, getMetaData 메소드를 호출하면 ArrayIndexOutOfBoundsExceptions 예외가 발생하는 경우가 있었으나, 이를 수정하였다.

CUBRIDSUS-3991 HA 환경에서 TRUNCATE TABLE 질의 수행 시 복제가 중단되는 오류 수정

HA 환경에서 마스터 서버에 TRUNCATE TABLE 문이 실행된 경우, 복제 정보 처리의 오류로 인해 슬레이브 서버의 applylogdb 프로세스가 비정상 종료되는 현상이 발생하였으나, 이를 수정하였다.

CUBRIDSUS-3911 HA 환경에서 온라인 백업 후 보관 로그 동기화 오류로 인해 복제가 중단되는 오류 수정

HA 환경에서 마스터 DB를 온라인 백업한 이후 복제가 지연되는 특정 상황에서 copylogdb 프로세스가 빈 보관 로그를 생성하는 오류로 인해, 특정 시점부터 copylogdb 프로세스가 반복적으로 재시작되면서 복제가 중단되는 문제가 발생하였으나 이를 수정하였다.

CUBRIDSUS-3910 HA 환경에서 applylogdb 프로세스가 재시작 후 비정상 종료되면서 데이터 불일치가 발생되는 오류 수정

HA 환경에서 데이터베이스의 페이지 크기(--page-size 옵션 사용)가 로그 페이지 크기(--log-page-size 옵션 사용)보다 크게 설정되어 생성된 경우, HA 운영 중에 로그 레코드가 미리 할당한 메모리 영역을 초과하여 기록되면서 로그를 슬레이브 DB로 반영할 때 applylogdb 프로세스가 비정상 종료되고, 이로 인해 데이터 불일치가 발생되는 오류가 있었으나 이를 수정하였다.

CUBRIDSUS-4295 HA 환경에서 인덱스 이름만을 명시한 DROP INDEX 구문이 슬레이브 DB에 반영되지 않는 오류 수정

HA 환경의 마스터 서버에서 아래의 예와 같이 테이블 명과 컬럼 명을 생략하고 인덱스 명만을 명시하여 DROP INDEX 구문을 수행하는 경우, 슬레이브 DB에 반영되지 않는 오류를 수정하였다.

```
// 인덱스 명만 명시하여 DROP INDEX 수행
DROP INDEX idx_abc;
```

CUBRIDSUS-4347 HA 환경에서 DB 연결을 설정하는 중에 발생하는 오류 수정

HA 환경에서 브로커 서버의 DB 호스트 접속 순서가 databases.txt에 host1:host2로 되어 있고 마스터 DB는 host1, 슬레이브 DB는 host2인 환경에서 브로커의 ACCESS_MODE=SO(Slave Only)로 설정되어 있는 경우, 슬레이브 DB를 찾는 과정에서 처음에 연결했던 host1은 슬레이브 DB가 아니므로 브로커 응용 서버(CAS)와 DB 서버 간의 연결을 종료하여야 하나, 연결이 종료되지 않고 남아 있는 오류를 수정하였다.

또한, HA 환경에서 DB 서버가 허용하는 최대 클라이언트 개수(cubrid.conf의 max_client)를 초과하는 경우, 브로커 서버의 databases.txt에 있는 순서에 따라 다음 호스트로 DB 연결을 시도하여야 하나 그렇지 않은 오류를 수정하였다.

CUBRIDSUS-3804 다중 동시 접속 및 삽입/삭제 질의 수행 시 특정 조건에서 연결이 종료되지 않는 오류 수정

다중으로 동시 접속하여 삽입 및 삭제 질의를 수행하는 경우, 응용 프로그램이 종료되지 않고 브로커 응용 서버(CAS) 중 일부는 연결 상태가 CLOSE_WAIT로 대기하는 현상이 발생하는 경우가 있었으나 이를 수정하였다. 이 현상은 브로커 설정 파일(cubrid_broker.conf)의 MIN_NUM_APPL_SERVER와 MAX_NUM_APPL_SERVER의 값이 서로 다르게 설정된 경우에 발생하였다.

CUBRIDSUS-4272 -327번 에러가 발생할 수 있는 오류 수정

동시 사용자 환경에서 다음과 같이 -327번 에러가 발생할 수 있는 오류를 수정하였다.

```
Time: 12/01/10 17:14:13.728 - ERROR *** ERROR CODE = -327, Tran = 5, EID = 3
Cannot create MOP with NULL OID.
```

CUBRIDSUS-4218 다량의 페이지를 가지는 내부 파일 생성 요구가 실패하는 오류 수정

인덱스 생성과 같이 한번에 많은 양의 페이지를 요구하는 작업을 수행하는 경우, 내부 파일 관리 맵이 잘못 되어 다음과 같은 에러를 발생시키면서 페이지 할당이 실패할 수 있는 오류를 수정하였다.

```
Internal error: fetching deallocated pageid -2146798529 of volume
"/home/qa/db/testdb/testdb_t32763".
```

CUBRIDSUS-4217 문자열 중간에 NULL이 포함된 데이터를 내보내기(unload)하는 경우 데이터 파일이 잘못 생성되는 오류 수정

VARCHAR, CHAR와 같은 문자 타입 컬럼이 있는 레코드 값의 중간에 ASCII CODE 값 0(즉, NULL)이 포함되어 있는 경우, unloaddb 유틸리티가 데이터 파일을 잘못 생성하여 이를 loaddb 유틸리티로 로딩하는데 실패하였으나, 데이터 파일을 정상적으로 생성하여 데이터 파일 로딩이 성공하도록 오류를 수정하였다.

CUBRIDSUS-4084 64bit Windows 시스템에 32bit CUBRID 설치 시 ODBC 드라이버가 등록되지 않는 오류 수정

64bit Windows 시스템에 32bit CUBRID를 설치하는 경우, ODBC 관리자에 32bit용 ODBC 드라이버가 등록되지 않는 오류가 있었으나, 이를 수정하였다.

CUBRIDSUS-3926 Windows의 명령 프롬프트 창에서 CUBRID 제어 시 정상 수행되지 않는 오류 수정

Windows의 명령 프롬프트 창에서 cubrid 유틸리티 명령어로 브로커나 서버 또는 매니저 서버를 시작/종료 시도하는 경우, SYSTEM 계정이 아닌 일반 사용자 계정으로 실행을 시도하면서 정상 동작하지 않는 오류가 있었으나, SYST

EM 계정으로 정상 수행이 가능하도록 수정하였다.

이로 인해 Windows Vista 이상 버전에서는 cubrid 유틸리티나 CUBRID Service Tray를 관리자 권한으로 구동하지 않는 경우 권한 상승 확인을 위한 UAC(User Account Control) 대화 상자가 활성화되고 이를 승인하여야 정상적인 제어가 가능하다.

cubrid 유틸리티를 사용하는 경우 명령 프롬프트 창을 관리자 권한으로 구동하는 것을 권장하며 구동 방법은 5.주의사항 > CUBRIDSUS-4186을 참고한다.

이 수정으로 인해 Windows 환경에서 브로커 파라미터의 동적 변경 기능이 동작하지 않는 제약 사항이 추가되었다. (5.주의사항 > CUBRIDSUS-3926 참고)

CUBRIDSUS-4015 영문 Windows에서 CUBRID Service Tray와 관련된 오류 메시지 출력 시 한글이 출력되는 오류 수정

한글 폰트가 설치되지 않은 영문 Windows에서 CUBRID Service Tray와 관련된 오류 메시지 출력 시 한글 메시지가 출력되어 메시지가 비정상으로 보였으나, 이 메시지를 영문으로 출력하도록 수정하였다.

6.5 CUBRID 매니저 2008 R3.1에서 변경된 사항

새로 추가된 기능

HA 모니터링 대시보드 기능 추가

운영 중인 데이터베이스 서버 및 브로커, HA의 상태를 모니터링 할 수 있는 모니터링 대시보드 기능을 추가하였다.

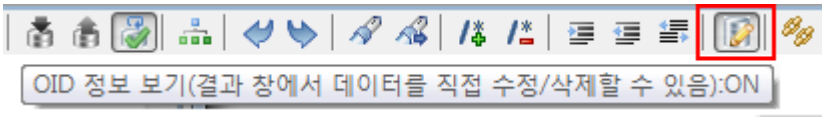
모니터링 대시보드를 통해 데이터베이스를 구동하고 있는 호스트, 데이터베이스 서버, 브로커의 실시간 운영 및 장애 현황을 감시할 수 있으며, 각각의 상세한 세부 정보도 별도의 창을 통해 볼 수 있다.

BLOB, CLOB, 길이가 100 이상인 BIT 타입의 입력 기능 추가

BLOB, CLOB, 그리고 길이가 100 이상인 BIT 타입의 데이터는 기존의 [데이터 내려받기 및 내보내기] 메뉴에서 데이터를 입력할 수 없어 [질의 편집기> 질의 결과> 상세 보기] 메뉴에서 입력할 수 있도록 추가하였다. 이와 같은 데이터는 질의 결과 창에 데이터를 직접 표시하지 않고 (BLOB), (CLOB), (BIT)와 같은 태그만 표시된다.

질의 편집기에서 데이터를 수정하려면, [질의 결과] 탭의 해당 컬럼에서 마우스 오른쪽 버튼을 클릭하고 [상세 보기]를 선택한 후 입력할 데이터 파일을 선택하면 된다.

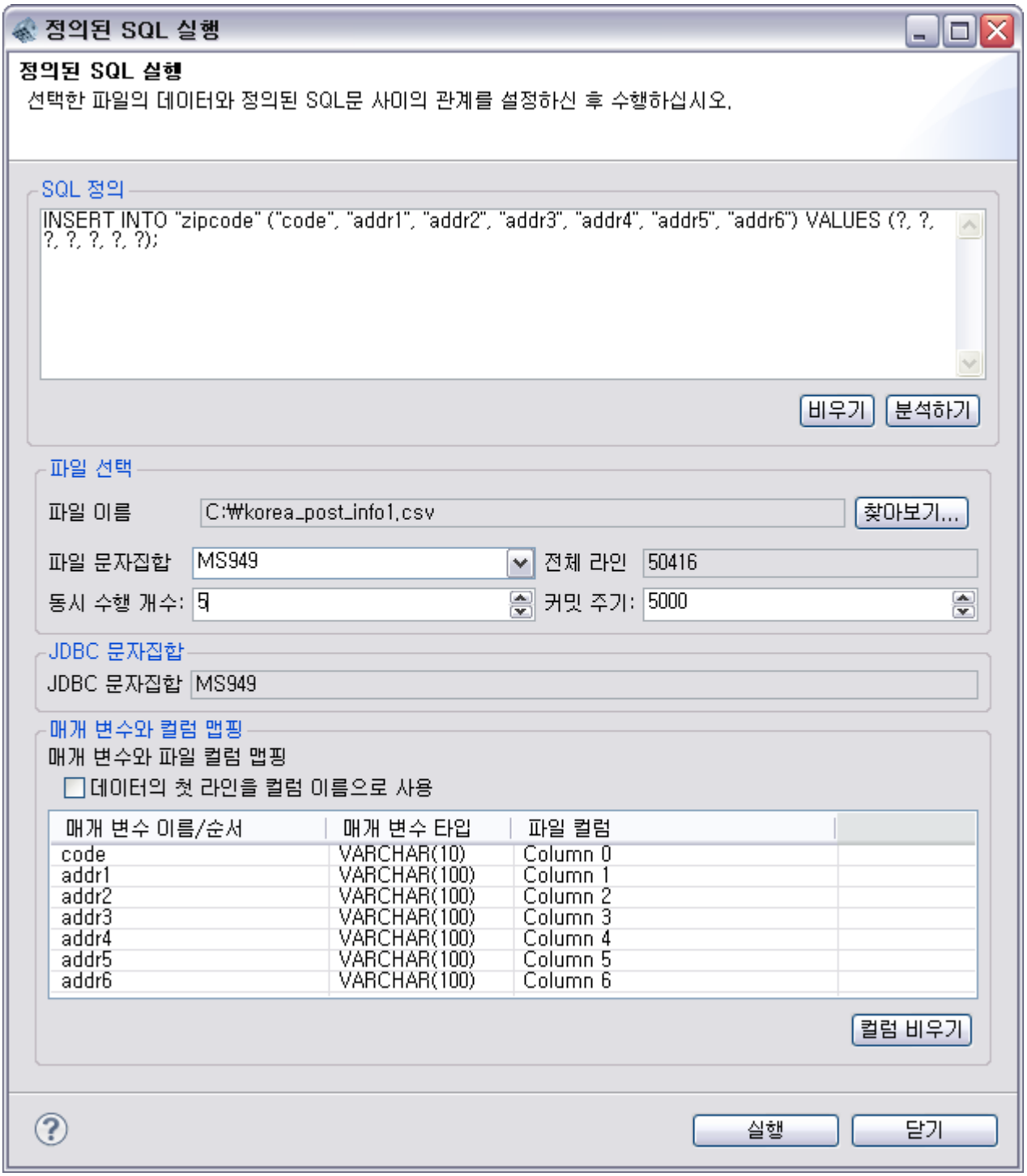
단, 해당 기능은 REUSE_OID 옵션을 사용하여 생성한 테이블에서는 사용할 수 없으며, [OID 정보 보기] 기능을 ON 했을 경우에 한해서만 지원한다. [OID 정보 보기]를 ON하는 방법은 아래의 그림과 같이 질의 편집기 툴 바에서 선택할 수 있으며, 기본값으로 설정하고자 할 때는 해당 호스트의 [속성> 질의 편집기] 옵션에서 [OID 정보 보기]를 선택한다.



정의된 SQL 실행 기능 추가

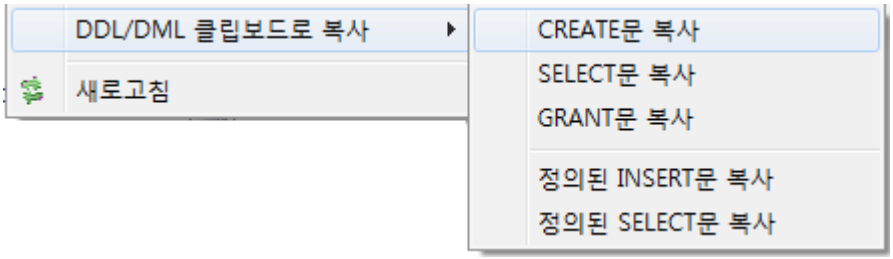
CUBRID 매니저에서 사용자가 작성한 정의된 SQL(Prepared Statement)문을 이용하여 데이터를 입력하고 조회할 수 있는 기능을 지원한다. 이를 통해 대량의 데이터를 좀 더 쉽고 빠르게 데이터베이스에 입력할 수 있으며 사용자가

문자 집합과 동시 수행 개수 그리고 커밋 주기 등을 직접 설정할 수 있다.



DDL/DML을 클립보드로 복사하는 기능 추가

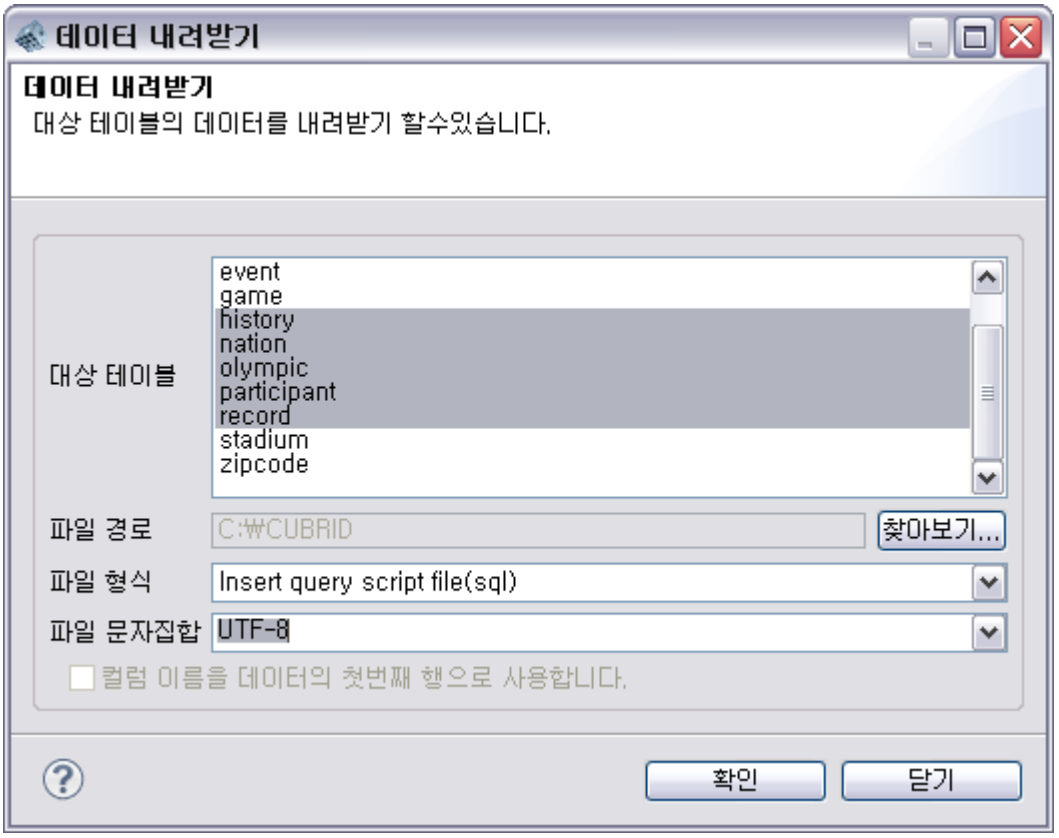
[DDL/DML 클립보드로 복사] 기능은 선택한 테이블의 DDL과 DML을 클립보드로 복사하여 다른 편집도구에서 사용하거나 저장할 수 있도록 지원한다. 제공하는 기능으로는 CREATE문, SELECT문, GRANT문과 Prepared Statement문으로 작성된 INSERT문과 SELECT문이 있다.



개선된 사항

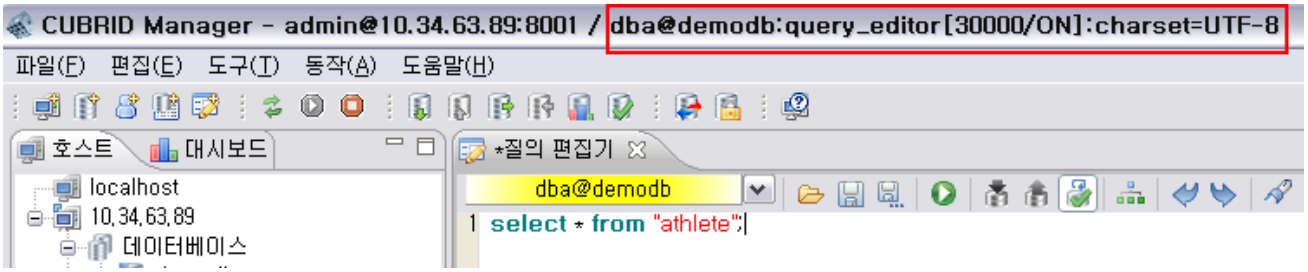
데이터 올리기/내려받기 할 때 문자 집합 설정 및 확인할 수 있도록 개선

데이터 올리기 및 내려받기에서 데이터의 문자 집합을 지정할 수 있도록 개선하였다. 뿐만 아니라, 다중 테이블을 선택하여 한번에 내려받을 수 있다.



질의 편집기에서 현재 사용중인 문자 집합을 확인할 수 있도록 개선

여러 개의 질의 편집기를 사용할 때 현재 편집기에서 사용중인 문자 집합 값을 쉽게 확인할 수 있도록, 매니저의 타이틀 바와 상태 바에 현재 질의 편집기에서 접속한 데이터베이스의 문자 집합 정보를 출력하도록 개선되었다.

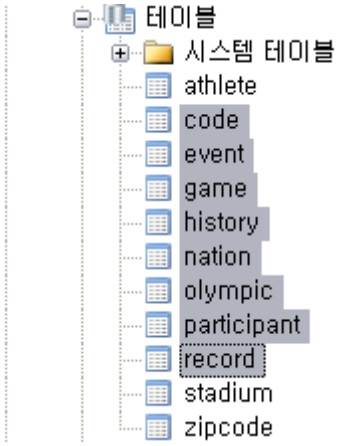


자동 볼륨 추가 설정 시 최대값을 사용자가 설정 할 수 있도록 개선

이전 버전은 자동 볼륨 추가 설정의 최대값이 2G로 내부 설정되어 있었으나, 최대값을 사용자가 설정할 수 있도록 개선하였다.

DELETE ALL, TRUNCATE TABLE 등을 여러 개의 테이블에 한번에 적용할 수 있도록 개선

테이블 탐색창에서 테이블을 다중 선택하여 DELETE ALL, TRUNCATE TABLE, 데이터 내려받기, 테이블 삭제, DDL/DML 클립보드로 복사 기능을 수행할 수 있도록 개선하였다.



브로커 상태 모니터 차트에서 SESSION과 ACTIVE SESSION이 구분되도록 개선

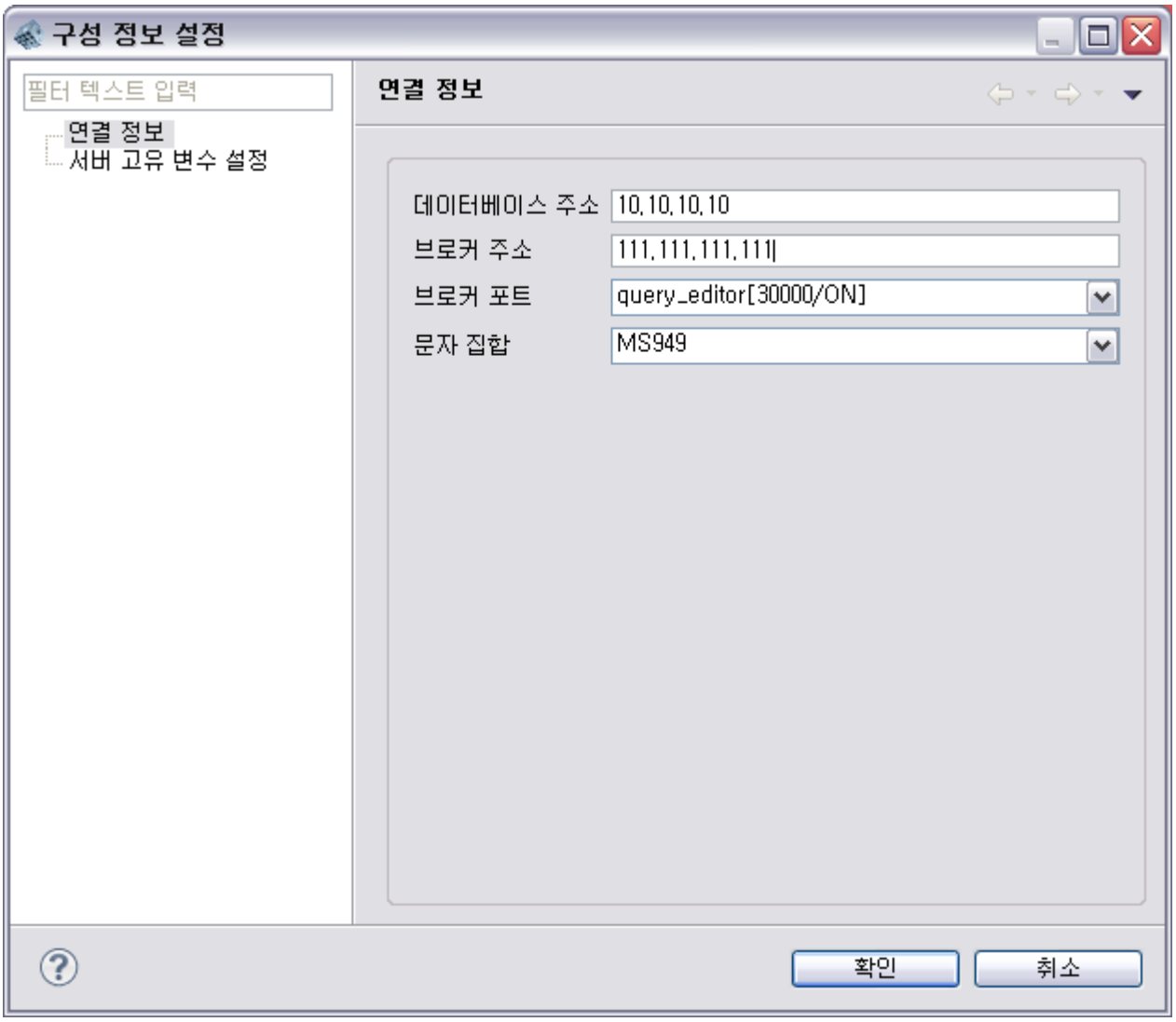
브로커 상태 모니터 차트 설정 시 기존에는 Active Session만 있던 항목을 Active Session과 Session 두 항목으로 구분되도록 개선하였다.

기존 버전에서 Active Session은 브로커 프로세스(이하 CAS) 상태가 IDLE이 아닌 모든 것을 의미하였으나, 2008 R3.1에서 Active Session은 CAS 상태가 BUSY인 것만 나타내며, Session은 CAS 상태가 IDLE이 아닌 모든 것을 의미하도록 수정하였다.

다중 네트워크 사용 환경에서 데이터베이스 서버 접속에 실패하는 문제 개선

CUBRID 매니저 클라이언트에서 접속할 때 사용하는 IP와 데이터베이스 서버에 접속할 때 사용하는 IP의 망(Network)이 분리되어 있는 경우, CUBRID 매니저에서 호스트 접속을 하면 서버 내부에서도 데이터베이스 접속을 같은 IP로 시도하게 되어 데이터베이스에 접근하지 못하는 문제가 발생하는데, 데이터베이스 서버에서 사용하는 IP를 호스트 속성에 추가하여 네트워크 망이 분리되어 있는 환경에서도 CUBRID 매니저를 정상 수행할 수 있도록 하였다.

호스트 > 데이터베이스 > 데이터베이스 명에서 마우스 오른쪽 버튼을 클릭하고, [속성] 메뉴를 선택하면 연결 정보에서 데이터베이스 주소를 설정할 수 있다.



수정된 사항

호스트 상태 모니터에서 메모리 정보 출력 시 물리적 메모리 외의 영역을 포함하는 오류 수정

호스트 상태 모니터의 사용 메모리 영역에 물리적으로 사용한 메모리 영역 외에 캐시와 버퍼 영역을 포함하여 출력하던 오류가 있었으나, 실제 물리적으로 사용한 메모리만 출력되도록 수정하였다.

Windows XP 서비스팩 3에서 CUBRID 매니저로 데이터베이스 시작 시 발생 가능한 오류 수정

서비스팩 3가 설치된 Windows XP에서 CUBRID 매니저로 데이터베이스를 시작하면 매우 드물게 프로세스 정보 수집 오류로 인해 “Database(demodb) is running in standalone mode” 라는 메시지가 발생하는 경우가 있었으나, 이를 수정하였다.

Windows 7에서 데이터베이스 상태 정보의 일부를 확인할 수 없는 오류 수정

Windows 7에서 CUBRID 매니저로 데이터베이스 상태 화면을 활성화 시키면 수직 스크롤 바가 생성되지 않아 DB 볼륨 정보 그래프를 볼 수 없었던 오류를 수정하였다.

문법에 맞지 않는 일부 질의에 대해 에러가 출력되지 않는 오류 수정

CUBRID 매니저의 질의 분석기 오류로 인해 SQL 문법에 맞지 않는 일부 질의가 입력된 경우 에러 메시지가 출력되지 않는 문제를 수정하였다.

6.6 주의 사항

CUBRID 2008 R3.1 사용 시 주의 사항

CUBRIDSUS-3826 GLO 클래스 지원 중단에 따른 주의 사항

CUBRID 2008 R3.0 이하 버전은 **glo**(Generalized Large Object) 클래스를 사용하여 Large Object를 처리하였으나, CUBRID 2008 R3.1은 **glo** 클래스를 제거하고 BLOB, CLOB 타입(이하 LOB)을 지원한다. (관련 매뉴얼 참고)

기존의 glo 클래스 사용자는 다음과 같이 작업할 것을 권장한다.

- GLO 데이터를 파일로 저장한 후 어플리케이션 및 DB 스키마에서 GLO를 사용하지 않도록 수정한다.
- 데이터베이스 마이그레이션을 한다. (본 문서의 데이터베이스 마이그레이션 절차 참고)
- 변경한 어플리케이션에 맞게 파일을 LOB 데이터로 로딩하는 작업을 수행하도록 한다.
- 수정한 어플리케이션이 정상 동작하는지 확인한다.

참고로, cubrid loaddb 유틸리티는 GLO 클래스를 상속받거나 GLO 클래스 타입을 가진 테이블을 로딩하려는 경우, Error occurred during schema loading 에러 메시지와 함께 데이터 로딩을 중지한다.

GLO 클래스의 지원 중단에 따라 각 인터페이스 별로 삭제한 함수는 다음과 같다.

인터페이스	삭제한 함수
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 BLOB, CLOB 타입 사용 시 제약 사항

- BLOB, CLOB 타입(이하 LOB)에 대하여 다음과 같은 제약 사항이 있으므로 사용에 주의한다.
- LOB 타입 컬럼 간 비교 연산(=, <>, IN, NOT IN 등)을 할 수 없으며, 이를 위해서는 문자열 또는 비트열로 타입을 변환한 후 사용해야 한다. 단, IS NULL, IS NOT NULL은 지원한다.
- PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL 제약 조건 또는 인덱스를 정의할 수 없다.
- 테이블 생성 및 수정 시, SHARED 속성을 정의할 수 없으며 DEFAULT 속성은 NULL 값에 대해서만 정의할 수 있다.
- 데이터베이스에는 파일의 위치(LOB Locator)가 저장되고 데이터는 파일로 저장되는 구조이므로, 장애가 발생하여 특정 시점으로 복구할 때 LOB Locator와 LOB 데이터의 매핑이 유효하지 않아 에러가 발생할 수 있다.
- ALTER TABLE DROP 문을 사용하여 컬럼을 삭제하거나, DROP TABLE 문을 사용하여 테이블을 삭제하는 경우 LOB Locator만 삭제되고 LOB 컬럼이 참조하는 외부 파일 시스템의 LOB 파일은 삭제되지 않고 남아있다.
- CUBRID가 제공하는 API나 CUBRID 매니저, csql을 사용하지 않고 사용자 임의로 LOB 타입의 데이터 파일을 직접 수정하면 내용이 일치됨을 보장할 수 없다.

(자세한 설명은 관련 매뉴얼 참고)

CUBRIDSUS-3926 Windows에서 브로커 파라미터의 동적 변경 제약 사항

Windows 환경에서는 broker_changer를 사용하여 브로커 파라미터를 동적으로 변경할 수 없으므로 주의한다. (본 문서의 3. CUBRID 2008 R3.1에서 변경된 사항> CUBRIDSUS-3926 참고)

```
// LONG_TRANSACTION_TIME 파라미터를 동적으로 변경하는 예
% broker_changer broker1 LONG_TRANSACTION_TIME 30.00
```

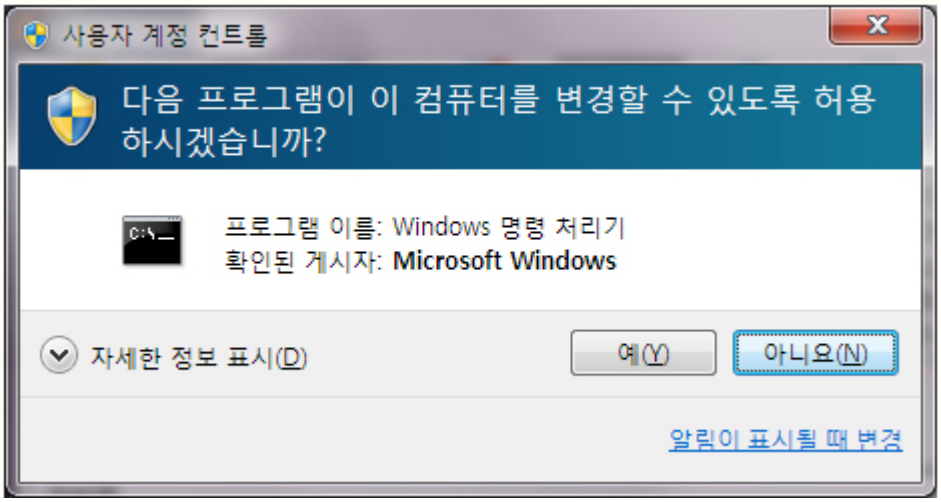
CUBRIDSUS-4186 Windows Vista 이상 버전에서 cubrid 유틸리티를 사용한 서비스 제어 시 권장 사항

Windows Vista 이상 버전에서 cubrid 유틸리티를 사용하여 서비스를 제어하려면 명령 프롬프트 창을 관리자 권한으로 구동한 후 사용하는 것을 권장한다.

명령 프롬프트 창을 관리자 권한으로 구동하지 않고 cubrid 유틸리티를 사용하는 경우 UAC(User Account Control) 대화 상자를 통하여 관리자 권한으로 수행될 수 있으나 수행 결과 메시지를 확인할 수 없다.

Windows Vista 이상 버전에서 명령 프롬프트 창을 관리자 권한으로 구동하는 방법은 다음과 같다.

1. [시작> 모든 프로그램> 보조 프로그램> 명령 프롬프트]에서 마우스 오른쪽 버튼을 클릭한다.
2. [관리자 권한으로 실행(A)]을 선택하면 아래와 같은 권한 상승을 확인하는 대화 상자가 활성화된다.



1. “예”를 클릭하면 명령 프롬프트 창이 관리자 권한으로 구동된다.

CUBRIDSUS-3217 JDBC에서 연결 정보를 URL 스트링으로 입력하는 경우 물음표를 반드시 명시

JDBC에서 URL 스트링으로 연결 정보를 입력하는 경우, 이전 버전에서는 물음표(?)를 입력하지 않더라도 속성(PROPERTY) 정보가 적용되었으나, CUBRID 2008 R3.0부터는 문법에 따라 반드시 물음표를 명시하여야 하고 이를 생략할 경우 에러를 출력한다. 또한, 연결 정보 중 USERNAME과 PASSWORD가 없더라도 반드시 콜론(:)을 명시하여야 한다.

```
URL=jdbc:CUBRID:127.0.0.1:31000:db1:::althosts=127.0.0.2:31000,127.0.0.3:31000  - 에러 처리
URL=jdbc:CUBRID:127.0.0.1:31000:db1:::?althosts=127.0.0.2:31000,127.0.0.3:31000  - 정상 처리
```

CUBRIDSUS-3564 마스터 프로세스와 서버 프로세스 간 프로토콜 변경 및 두 개 버전을 동시에 운영하는 경우 포트 설정 필요

마스터 프로세스(cub_master)와 서버 프로세스(cub_server) 간 통신 프로토콜 변경으로 인해 CUBRID 2008 R3.0 이상 버전의 마스터 프로세스는 하위 버전의 서버 프로세스와 통신할 수 없고, 하위 버전의 마스터 프로세스도 2008 R3.0 이상 버전의 서버 프로세스와 통신할 수 없다. 따라서, 이미 하위 버전이 설치되어 있는 환경에서 새 버전을 추가 설치하여, 두 개 버전의 CUBRID를 동시에 운영하는 경우, 각각 서로 다른 포트를 사용하도록 **cubrid.conf**의 **cubrid_port_id** 파라미터를 수정하여야 한다.

CUBRIDSUS-2828 데이터베이스 이름에 @를 포함할 수 없음

데이터베이스 이름에 @이 포함되는 경우 호스트 이름이 명시된 것으로 해석될 수 있으므로, 이를 방지하기 위하여 **cubrid createdb**, **cubrid renamedb**, **cubrid copydb** 유틸리티 실행 시 데이터베이스 이름에 @를 포함할 수 없도록 수정하였다.

CUBRIDSUS-3267 Windows 환경에서 디렉터리 경로 설정 시 주의 사항

Windows 환경에서 CUBRID 설치 디렉터리 경로에 공백을 포함하는 경우 정상 설치가 되지 않으므로 주의한다. 또한, DB 언로드/로드/백업 등의 작업 대상 디렉터리 경로에도 공백을 포함할 수 없다.

CUBRIDSUS-3553 CUBRID 소스 빌드 후 실행 시, 매니저 서버 프로세스 관련 오류 발생

사용자가 직접 빌드하여 설치하는 경우, CUBRID와 CUBRID 매니저를 각각 빌드하여 설치하여야 한다. 만약, CUBRID 소스만 checkout하여 빌드 후 **cubrid service** 또는 **cubrid manager**를 실행하면, cubrid manager server is not installed라는 오류가 발생한다.

CUBRID 매니저에서 [볼륨 자동 추가 기능 사용] 옵션을 선택하고 DB를 생성하는 경우 주의 사항

[볼륨 자동 추가 기능 사용] 옵션은 DB 볼륨이 사용자가 설정한 “여유 공간 비율" 값 이하가 될 경우 해당 볼륨을 자동으로 추가하는 기능이며, 이 옵션이 선택되면 여유 공간을 모니터링하기 위해 주기적(디폴트 5초)으로 **cubrid spacedb** 유틸리티를 수행하므로 에러 로그(<dbname>_spacedb.err)가 증가할 수 있다. 사용자는 **cm.conf** 파일의 **monitor_interval** 파라미터를 설정하여 모니터링 주기를 조정할 수 있다.

7

CUBRID 8.2.2 릴리스 노트

7.1 CUBRID 2008 R2.2 정보

릴리스 노트 개요

이 문서는 CUBRID 2008 R2.2 버전과 2008 R2.2 Patch 1~11 버전에 관한 유용한 정보를 포함한다. 릴리스 노트의 최신 버전은 CUBRID 오픈 소스 프로젝트 사이트(http://dev.naver.com/projects/cubrid)에서 확인할 수 있다.

릴리스 노트 개정 내역

CUBRID 2008 R2.2 버전의 릴리스 이후 릴리스 노트의 변경 사항은 아래와 같다.

작성 날짜	설명
2012 7월	CUBRID 2008 R2.2 Patch 11 릴리스
2011 10월	CUBRID 2008 R2.2 Patch 10 릴리스
2011 5월	CUBRID 2008 R2.2 Patch 9 릴리스
2011 2월	CUBRID 2008 R2.2 Patch 8 릴리스
2011 1월	CUBRID 2008 R2.2 Patch 7 릴리스
2010 12월	CUBRID 2008 R2.2 Patch 6 릴리스
2010 10월	CUBRID 2008 R2.2 Patch 5 릴리스
2010 9월	CUBRID 2008 R2.2 Patch 4 릴리스
2010 9월	CUBRID 2008 R2.2 Patch 3 릴리스
2010 7월	CUBRID 2008 R2.2 Patch 2 릴리스
2010 6월	CUBRID 2008 R2.2 Patch 1 릴리스

작성 날짜	설명
2010 4월	CUBRID 2008 R2.2 릴리스

릴리스 특징

CUBRID는 웹 서비스 응용 프로그램에서 요구하는 고성능, 안정성, 확장성 및 가용성을 보장하는 오픈소스 DBMS이다. 이번 CUBRID 2008 R2.2 릴리스의 특징은 아래와 같다.

INSERT 성능 개선

INSERT 연산이 집중되는 상황에서 I/O 부하가 분산되도록 관련 알고리즘을 수정하여 INSERT 성능을 개선하였다.

아래는 테이블 40 개에 대해 INSERT 연산을 3 시간 동안 수행한 초당 트랜잭션 처리 수(TPS)를 이전 버전과 비교한 그래프이다. 이러한 워크로드에서는 이전 버전인 2008 R2.0에 비해 TPS 성능이 10% 이상 향상되고 TPS가 0으로 급격히 감소되는 현상이 없어진 것을 확인할 수 있다.

공간 재사용률 증가

INSERT/DELETE가 반복되는 워크로드에서 페이지 공간이 충분히 재사용될 수 있도록 수정하였다.

아래는 130만 레코드(50만 pages)가 저장된 테이블에 대해 삭제 연산(DELETE 또는 DROP)을 수행한 후 다시 INSERT를 수행했을 때 각각 여유 페이지 수를 비교한 표이며, 이전 버전에 비해 공간 재사용률이 크게 증가된 것을 확인할 수 있다.

워크로드	CUBRID 2008 R2.1			CUBRID 2008 R2.2		
	삭제 후 (pages)	입력 후 (pages)	재사용률 (%)	삭제 후 (pages)	입력 후 (pages)	재사용률 (%)
INSERT DELETE ALL INSERT	275362	57213	**20.8**	275362	254850	**92.6**
INSERT DROP CREATE INSERT	275362	50746	**18.4**	275362	274483	**99.7**

HA 기능 강화

리소스 관리 및 장애 검출 기능을 CUBRID HA 기능에 포함하여, 이전 버전에서 사용되었던 Linux Heartbeat 패키지에 대한 의존성을 제거하고 사용자가 HA 환경 구성 및 관련 리소스 관리를 보다 쉽게 수행할 수 있도록 하였다.

데이터베이스 호환성

CUBRID 2008 R2.2로 업그레이드할 경우 이전 버전 데이터베이스를 마이그레이션해야 한다. 아래는 버전 별 데이터베이스 마이그레이션 필요 여부를 정리한 표이다.

From To	CUBRID 2008 R1.x	CUBRID 2008 R2.0	CUBRID 2008 R2.1	CUBRID 2008 R2.2
CUBRID 2008 R1.x	불필요	필요	필요	필요
CUBRID 2008 R2.0	필요	불필요	불필요	필요
CUBRID 2008 R2.1	필요	불필요	불필요	필요

CUBRID 2008 R2.2의 설치 방법

Linux에서 설치

Linux용 설치 패키지는 Linux RPM, tar.gz, 바이너리를 포함하는 스크립트로 제공되며, 설치 방법은 [매뉴얼> CUBRID 시작>설치와 실행>Linux에서의 설치와 실행]을 참고한다.

Windows에서 설치

Windows용 설치 파일이 제공되며, 설치 마법사를 이용하여 편리하게 설치할 수 있다. 설치 방법은 [매뉴얼> CUBRID 시작>설치와 실행>Windows에서의 설치와 실행]을 참고한다.

CUBRID 환경 변수 및 OS 환경 변수 설정

CUBRID를 사용하기 위해서는 CUBRID 환경 변수와 관련 OS 환경 변수를 설정하여야 한다. 특히, CUBRID 매니저와 Java 저장 프로시저를 사용하기 위해서는 Java 환경 변수를 설정하여야 한다. 설정 방법은 [매뉴얼> CUBRID 시작>꼭 알아 두어야 할 것>환경 변수 설정]을 참고한다.

CUBRID 2008 R2.2로 업그레이드하는 방법

업그레이드 주의 사항

기존 환경 설정 파일 보관

이전 버전의 $CUBRID/conf 디렉터리의 환경 설정 파일(**cubrid.conf**, **cubrid_broker.conf**, **cm.conf**)과 $CUBRID_DATABASES 디렉터리의 데이터베이스 위치 정보 파일(**databases.txt**)을 보관한다.

데이터베이스 마이그레이션

CUBRID 2008 R2.2 버전은 이전 버전의 데이터베이스와 호환되지 않으므로, 기존 데이터베이스를 CUBRID 2008 R2.2 버전으로 마이그레이션 하여야 한다. (이 문서의 데이터베이스 마이그레이션 절차를 참고)

파라미터 변경 사항 확인

CUBRID 2008 R2.2 버전에서 아래의 파라미터는 변경 사항이 있으므로, 기존 환경 설정 파일을 적용하는 경우 주의한다.

	파라미터 이름	관련 이슈
변경	INDEX_SCAN_OID_BUFFER_PAGES	CUBRIDSUS-2124
추가	MAX_FLUSH_PAGES_PER_SECOND, ADAPTIVE_FLUSH_CONTROL, SYNC_ON_NFLUSH	CUBRIDSUS-2216
	HA_PORT_ID, HA_NODE_LIST	CUBRIDSUS-2024
	SELECT_AUTO_COMMIT	CUBRIDSUS-1988
	ERROR_LOG_WARNING	CUBRIDSUS-1955
	PAGE_FLUSH_INTERVAL_IN_MSECS	CUBRIDSUS-2232

복제 또는 HA 환경 재구성

복제 또는 HA 기능을 사용하는 시스템에서는 데이터베이스 마이그레이션 이후 환경을 재구성해야 한다.

데이터베이스 마이그레이션 절차

CUBRID는 **<CUBRID 설치 디렉토리>/bin/migrate_r22** 유틸리티를 제공하며, 이를 이용하여 보다 쉽게 데이터베이스 마이그레이션을 수행할 수 있다. 다른 방법으로는 **cubrid unloaddb/loaddb** 유틸리티를 사용하여 마이그레이션을 수행할 수 있으며, 이는 [매뉴얼> 관리자 안내서> 데이터베이스 마이그레이션]을 참고한다.

마이그레이션 도구를 사용하여 CUBRID 2008 R2.2 버전으로 데이터베이스 마이그레이션을 수행하는 절차는 아래와 같다.

단계	Linux 환경	Windows 환경
C1 단계: CUBRID Service 종료	% cubrid service stop	CUBRID 서비스 트레이>[Exit] 선택
C2 단계: 이전 버전 DB 백업	2008 R2.2 운영 중 이전 버전으로 복구하는 상황에 대비하기 위해 백업을 수행하고, 백업 파일을 별도 디렉터리(R21_backup)에 보관한다.(C2a) % mkdir R21_backup % cubrid backupdb -S -D R21_backup <db_name> 이전 버전의 databases.txt 및 conf 디렉토리 내 설정 파일을 별도 디렉터리에 보	(C2a)와 (C2b) 수행 후, 이전 버전 CUBRID를 제거한다. 주의할 점은, 이전 버전의 데이터베이스가 삭제되지 않도록 “데이터베이스와 설정 파일을 모두 삭제하시겠습니까?”라는 요청 창에서 “아니오”를 반드시 선택한다.

단계	Linux 환경	Windows 환경
	관한다.(C2b)	
C3 단계: 2008 R2.2 버전 설치	본 문서의 CUBRID 2008 R2.2의 설치 방법을 참고한다.	동일
C4 단계: 마이그레이션 도구 실행	(C2b)에서 보관한 databases.txt를 2008 R2.2의 설치 디렉터리에 복사한다.(C4a) 아래와 같이 migrate_r22 유틸리티를 실행한다.(C4b) % migrate_r22 <db_name>	CUBRID 서비스 트레이> [CUBRID Server]>[Stop] 선택하여 서버 종료 후, (C4a)와 (C4b) 수행
C5 단계: 2008 R2.2 버전 DB 백업	데이터베이스 버전이 R1.x인 경우에만 수행한다. 이전 버전이 R2.0 또는 R2.1 버전인 경우, 이 단계를 생략할 수 있다. % cubrid backupdb -S <db_name>	동일
C6 단계: CUBRID 환경 설정 및 CUBRID Service 구동	환경 설정 파일을 수정한다. 이때, (C2b)에서 보관한 이전 버전의 환경 설정 파일을 사용할 수 있다. % cubrid service start % cubrid server start <db_name>	CUBRID 서비스 트레이> [CUBRID Server]-> [Start]

HA 환경에서 데이터베이스 마이그레이션 절차

노드 장애 검출 및 리소스 관리를 수행하는 CUBRID Heartbeat 기능이 지원되므로 이전 버전에서 사용되었던 Linux Heartbeat 패키지가 더 이상 사용되지 않고, 이로 인해 HA 환경 구성 및 동작 방식이 변경된다. 따라서, HA 기능을 사용 중인 경우 마스터 데이터베이스를 마이그레이션한 후 백업 파일을 이용하여 HA 환경을 새로 구축하여야 한다.

아래는 브로커, 마스터 DB, 슬레이브 DB를 각각 별도 서버에 구축한 환경에서 현재 서비스를 중지하고 업그레이드를 수행하는 시나리오에 대해 설명한다. 서비스 무정지 업그레이드 시나리오는 별도 가이드 문서를 참고한다.

HA 환경에서 데이터베이스 마이그레이션 절차는 다음과 같다.

단계	설명
H1 단계: HA 관련 서비스 종료 및 기존 Linux heartbeat 제거	% cubrid broker stop % service heartbeat stop % chkconfig --del heartbeat % pkill -u user1 -f "cub_master"
H2~H5 단계: 마스터 서버에서 C2~C5 단계를 수행	마스터 서버에서 CUBRID 업그레이드 및 데이터베이스 마이그레이션을 수행하고, 2008 R2.2 데이터베이스를 백업한다.
H6 단계: 슬레이브 서버에 CUBRID 2008 R2.2 버전 설치	설치 방법은 본 문서의 CUBRID 2008 R2.2의 설치 방법을 참고한다.
H7 단계: 마스터 백업본을 슬레이브 서버에서 복구	H5에서 생성된 마스터 서버의 2008 R2.2 데이터베이스 백업본(testdb_bk*)을 슬레이브 서버에서 복구한다. % scp user1@master:~/DB/testdb/testdb_bk0v000 .

단계	설명
	% scp user1@master:~/DB/testdb/log/testdb_bkvinf ./log/. % cubrid restoredb testdb
H8단계: HA 환경 재구성 후 HA모드 구동	HA구동 스크립트(cubrid-ha) 및 환경 설정 파일(cubrid.conf)을 설정한다. [매뉴얼> 관리자 안내서> CUBRID HA 환경 설정]을 참고한다. 마스터 서버 및 슬레이브 서버에서 HA모드로 DB를 구동한다. [root@master ~]# service cubrid-ha start [root@slave ~]# service cubrid-ha start [매뉴얼> 관리자 안내서> CUBRID HA모드 구동]을 참고한다.
H9 단계: 브로커 서버에 CUBRID 2008 R2.2 버전 설치 및 브로커 구동	설치 방법은 본 문서의 CUBRID 2008 R2.2의 설치 방법을 참고한다. 브로커 설정 후 브로커를 시작한다. [매뉴얼> 관리자 안내서>CUBRID HA 환경 설정]을 참고한다. % cat cubrid_broker.conf … ACCESS_MODE=RW % cubrid broker start

복제 재구성 방법

복제 기능을 사용 중인 경우, 마스터 데이터베이스를 마이그레이션한 후 백업 파일을 이용하여 복제 환경을 새로 구축한다. 복제를 구축하는 방법은 [매뉴얼> 관리자 안내서> 데이터베이스 복제를 참고한다.

7.2 CUBRID 2008 R2.2 Patch 11에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-8124 HA 환경에서 복제 로그 파일이 삭제되지 않는 현상 수정

HA 환경에서 백업이 수행되는 노드의 상대편 노드(보통 슬레이브 노드에서 백업을 진행하므로 마스터 노드)에 있는 복제 로그 파일이 삭제되지 않는 현상을 수정했다.

CUBRIDSUS-8415 임시 볼륨의 소진으로 인해 일시적 임시 볼륨을 사용하여야 하는 질의 수행에서 실패할 수 있는 오류 수정

매우 큰 크기의 인덱스 생성 등과 같은 정렬 작업에서 임시 볼륨의 소진으로 인해 일시적 임시 볼륨을 사용하여야 하는 질의 수행에서, 일시적 임시 볼륨의 파일 크기가 4G를 넘는 경우 실패할 수 있는 오류를 수정하였다.임시 볼륨은 질의 처리 및 정렬을 수행할 때 일시적으로 사용되는 볼륨으로 이 공간이 모두 소진되면 일시적 임시 볼륨을 생성하여 사용하게 된다.

7.3 CUBRID 2008 R2.2 Patch 10에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-5722 호스트 변수를 입력 인자로 가지는 일부 문자열 조작 함수들을 통해 CHAR 타입 컬럼의 UPDATE 실행 시 잘못된 값이 저장되는 오류 수정

호스트 변수를 입력 인자로 가지는 REPLACE, LTRIM, RTRIM, TRIM 함수의 결과를 CHAR 타입 컬럼에 대입하는 UPDATE문을 실행하면 잘못된 값이 저장되는 오류를 수정하였다.

```
UPDATE test_tbl SET test_str = RTRIM(?,'.')
```

CUBRIDSUS-5726 서버 접속 과정에서 오류 발생으로 접속에 실패한 상태를 처리하지 못하는 현상 수정

서버 접속 과정에서 접속에 실패한 상태를 처리하지 못하면서 Server refused client connection : max clients, (XXX), exceeded. 라는 서버 오류가 발생하는 현상을 수정하였다.

CUBRIDSUS-5390 브로커 응용 서버 프로세스(CAS)의 메모리 크기가 2G 초과 시 해당 프로세스가 재시작되지 않는 오류 수정

브로커 응용 서버(CAS) 프로세스의 메모리 크기가 2G를 초과하는 경우 cubrid broker status의 PSIZE 값이 음수로 출력되면서 CAS 프로세스가 재시작되지 않는 오류를 수정하였다.

CUBRIDSUS-5113 Windows 버전의 브로커와 응용 프로그램 사이에 통신이 오래 걸리는 경우 통신 오류가 발생하는 현상 수정

Windows 버전의 브로커와 응용 프로그램 사이에 통신이 오래 걸리는 경우 통신 오류가 발생하는 현상을 수정하였다.

CUBRIDSUS-5913 ASYNC 모드로 설정된 HA 환경에서 슬레이브 서버의 과부하가 마스터 서버의 질의 수행에 영향을 주지 않도록 수정

ASYNC 모드로 설정된 HA 환경에서 슬레이브 서버의 과부하가 마스터 서버의 질의 수행에 영향을 주어, 마스터 서버의 질의 수행 시간이 오래 걸리는 현상이 발생하지 않도록 수정하였다.

CUBRIDSUS-5911 HA 환경에서 마스터 서버와 슬레이브 서버의 스키마가 불일치하는 경우 applylogdb 프로세스의 메모리 사용량이 증가하는 오류 수정

HA 환경에서 마스터 서버와 슬레이브 서버의 스키마가 불일치하는 경우 복제 로그를 반영하는 applylogdb 프로세스가 수행에 실패하면서 해당 프로세스의 메모리 사용량이 증가하는 오류를 수정하였다.

CUBRIDSUS-5914 HA 환경에서 복제 로그 반영에 계속 실패하는 경우 applylogdb 프로세스의 메모리 사용량이 제한 크기보다 증가하는 문제 수정

HA 환경에서 복제 로그 반영에 계속 실패하는 경우 applylogdb 프로세스의 메모리 사용량을 검사하지 않아, HA 환경 설정 파라미터인 ha_apply_max_mem_size에서 설정한 제한 크기보다 증가해도 applylogdb 프로세스를 재시작하지 않는 문제를 수정하였다.

CUBRIDSUS-5915 HA 환경에서 쓰기가 완료되지 않은 복제 로그를 슬레이브 서버에 반영하려고 시도하면서 무한히 대기하는 현상 수정

HA 환경에서 copylogdb 프로세스에 의해 쓰기가 완료되지 않은 복제 로그 파일을 applylogdb 프로세스가 읽어 슬레이브 서버에 반영하려고 시도하면서 무한히 대기하는 현상을 수정하였다.

CUBRIDSUS-4748 HA 환경에서 네트워크 장애로 인한 split-brain 현상이 발생할 수 있는 문제 수정

HA 환경에서 정상 수행 중인 노드를 네트워크 장애로 인해 비정상 상태로 판단하면서 불필요한 장애 복구 절차를 수행하여 동시에 두 개의 마스터 노드가 존재하게 되는 split-brain 현상이 발생할 수 있는 문제를 수정하였다. split-brain 현상을 막기 위해서는 네트워크 상태를 감시할 필요가 있으며, 이를 위해 ha_ping_hosts 파라미터를 설정하여야 한다.

CUBRIDSUS-5997 Windows 64bit 환경에서 CUBRID 64bit 설치 도중 Microsoft Visual C++ 2008 재배포 가능 패키지 설치를 요청하는 알림 창이 잘못 팝업되는 오류 수정

Windows 64bit 환경에서 CUBRID 64bit 설치 도중 Microsoft Visual C++ 2008 재배포 가능 패키지가 설치되었음에도 불구하고 해당 패키지의 설치를 요청하는 알림 창이 잘못 팝업되는 오류를 수정하였다.

CUBRIDSUS-6063 REUSE_OID 테이블 옵션으로 생성한 테이블 재생성 시 장시간 소요되는 문제 개선

REUSE_OID 테이블 옵션으로 생성한 테이블을 재생성할 때 대량의 디스크 I/O로 인하여 장시간이 소요될 수 있는 문제를 개선하였다.

CUBRIDSUS-5964 많은 양의 볼륨 페이지를 사용하는 트랜잭션을 롤백하면 비정상 동작하는 오류 수정

CREATE INDEX 또는 INSERT .. SELECT 문과 같이 많은 양의 볼륨 페이지를 사용하는 트랜잭션을 롤백하면 잘못 작성된 롤백 정보로 인해 서버 프로세스가 무한 반복에 빠지는 등 비정상적으로 동작하는 오류를 수정하였다.

CUBRIDSUS-6037 error_log_level 시스템 파라미터가 NOTIFICATION인 경우 NOTIFICATION 메시지 이전에 발생한 오류 메시지가 지워지는 문제 수정

error_log_level 시스템 파라미터가 NOTIFICATION인 경우 NOTIFICATION 메시지 이전에 발생한 오류 메시지가 오류 스택(stack)에 보관되지 않아 지워지는 문제를 수정하였다.

7.4 CUBRID 2008 R2.2 Patch 9에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-4905 컬럼 개수가 8개를 초과하는 다중 컬럼 인덱스의 인덱스 스캔 시 잘못된 질의 결과를 가져오는 오류 수정

다중 컬럼 인덱스에서 컬럼 개수가 8개를 초과하는 경우 인덱스 스캔 시 잘못된 질의 결과를 가져오는 오류를 수정하였다.

CUBRIDSUS-5196 ORDER BY 컬럼에 대해 WHERE 절에서 호스트 변수를 이용하는 조건으로 사용되는 경우 ORDER BY 최적화 수정

ORDER BY 컬럼에 대해 WHERE 절에서 호스트 변수를 이용하는 조건으로 사용되는 경우 ORDER BY 최적화 과정에서 추가된 조건이 인덱스 스캔 범위로 선택되어 실행 속도가 저하되었다. 실행 속도 개선을 위해, ORDER BY 최적화를 위해 추가된 조건은 인덱스 스캔 조건으로 선택되지 않도록 수정하였다.

ORDER BY 절에 명시한 컬럼으로 구성된 인덱스가 존재하는 경우에는 정렬 과정을 생략할 수 있기 때문에 질의 최적화기는 가상의 조건을 추가한 후 최적의 실행 계획을 만든다. 아래 예의 경우, “req_ymdt BETWEEN -무한대 AND +무한대”의 조건이 추가된다.

```
-- 기존 버전에서는 다음과 같이 ORDER BY에 있는 req_ymdt 컬럼이 WHERE절에서 호스트 변수를
이용하는 조건으로 사용되는 경우, 실행 속도가 저하되었다.
SELECT key, id, messageType, serverIpAddr, req_ymdt
FROM info
WHERE req_ymdt >= TO_TIMESTAMP(?,'YYYY"-"MM"-"DD')
  AND req_ymdt < TO_TIMESTAMP(?,'YYYY"-"MM"-"DD') + 60*60*24
ORDER BY req_ymdt desc
```

CUBRIDSUS-5051 UPDATE 문에서 VARCHAR 타입 컬럼에 정의된 크기를 초과하는 문자열 바인딩 시 데이터가 크기를 초과하여 입력되는 문제 수정

UPDATE문에서 VARCHAR 타입 컬럼에 정의된 크기를 초과하는 문자열을 바인딩하는 경우 정의된 크기를 초과하는 데이터를 포함하여 UPDATE되는 문제가 발생하였으나, 정의된 크기를 초과하는 부분을 잘라내고 UPDATE하도록 수정하였다.

CUBRIDSUS-5019 OUTER JOIN 질의문에 ORDER BY 절이 있는 경우 질의 결과에서 NULL이 제외되는 오류 수정

OUTER JOIN 질의문에 ORDER BY 절이 있는 경우 질의 재작성이 잘못되어 결과 값에서 NULL이 제외되는 오류를 수정하였다.

```
-- 결과에 NULL값이 포함되어 있다.
SELECT b.y, f.x FROM foo f RIGHT OUTER JOIN bar b ON f.y = b.y;
y x
=========================
1 1
2 NULL

-- ORDER BY 절이 있는 경우 결과에서 NULL이 제외되는 오류가 존재하였다.
SELECT b.y, f.x FROM foo f RIGHT OUTER JOIN bar b ON f.y = b.y ORDER BY f.x;
```

```
y x
=======================
1 1
```

CUBRIDSUS-5088 DELETE 작업 철회로 인해 질의 실행 계획이 잘못될 수 있는 오류 수정

DELETE 작업 철회 시 통계 정보가 잘못 유지되어 질의 실행 계획이 잘못될 수 있는 오류를 수정하였다.

CUBRIDSUS-4715 CCI API 함수 일부에서 에러 메시지의 길이가 1024를 넘는 경우 발생하는 오류 수정

CCI API 함수 중 일부에서 에러 메시지의 길이가 지정한 에러 버퍼의 크기인 1024를 넘는 경우 메시지 출력에 실패하였으나, 에러 메시지의 길이가 1024를 넘는 부분을 제외하고 출력하도록 수정하였다.

CUBRIDSUS-5120 cubrid addvoldb 유틸리티와 데이터베이스 볼륨의 자동 증가가 동시 수행되는 경우 서버가 멈추는(hang) 오류 수정

cubrid addvoldb 유틸리티와 데이터베이스 볼륨의 자동 증가가 동시에 수행되는 경우 데이터베이스 서버 프로세스가 동작을 멈추는(hang) 현상이 발생하였으나 이를 수정하였다.

CUBRIDSUS-4397 cubrid unloaddb 유틸리티에 테이블 별로 데이터 파일을 생성하는 옵션 추가

cubrid unloaddb 유틸리티를 수행하는 경우, 각 테이블 별로 데이터 파일 생성이 가능하도록 --datafile-per-class 옵션을 추가하였다..

CUBRIDSUS-5200 cubrid spacedb 유틸리티에 요약 출력 기능 추가

cubrid spacedb 유틸리티에 데이터 볼륨(DATA), 인덱스 볼륨(INDEX), 일반 볼륨(GENERIC), 임시 볼륨(TEMP), 일시적 임시 볼륨(TEMP TEMP) 별로 전체 공간(total_pages), 사용 공간(used_pages), 빈 공간(free_pages), 볼륨 개수(volume_count)를 합산하여 출력하도록 -s(--summarize) 옵션을 추가하였다.

```
% cubrid spacedb -s demodb
Summarized space description for database 'demodb' with pagesize 16384. (log pagesize: 16384)

Purposetotal_pagesused_pagesfree_pagesvolume_count
------------------------------------------------------------------------------
DATA100039971
INDEX 2000319971
GENERIC500023647641
TEMP3000329971
```

```
TEMP TEMP0000
------------------------------------------------------------------------------
TOTAL11000245107554
```

CUBRIDSUS-5106 cubrid spacedb 유틸리티에서 합산 결과를 잘못 출력하는 오류 수정

데이터베이스 볼륨 파일의 크기가 매우 큰 경우 cubrid spacedb 유틸리티의 디스크 볼륨의 총 용량 합산 과정에서 오버플로우가 발생하여 합산 용량이 잘못 출력되는 오류를 수정하였다. 또한 temp 볼륨의 용량 합산 계산이 잘못된 오류를 수정하였다.

CUBRIDSUS-5284 cubrid compactdb 수행 시 간혹 내부 오류가 발생하여 데이터베이스 볼륨이 잘못될 수 있는 문제 수정

cubrid compactdb 수행(CUBRID 매니저에서는 “데이터베이스 공간 정리” 수행) 시 간혹 데이터베이스 볼륨의 헤더 정보가 잘못 작성되어 데이터베이스 볼륨이 잘못될 수 있는 문제를 수정하였다. cubrid checkdb(CUBRID 매니저에서는 “데이터베이스 검사” 수행)를 통해 이 문제가 발생하였는지 여부를 확인할 수 있다.

CUBRIDSUS-5199 cubrid statdump 유틸리티가 동시성을 보장하도록 개선

동시에 여러 사용자가 cubrid statdump 유틸리티를 통해 데이터베이스 서버 실행 통계 정보를 모니터링하는 경우에 잘못된 정보를 출력할 수 있는 오류를 수정하였다. 단, 각 통계 수치의 누적 값이 8바이트 크기 자료형의 표현 범위를 초과하는 경우에 해당 수치가 초기화된 후 누적됨을 주의해야 한다.

또한, CSQL 인터프리터에서 ;.history all (서버 전체 실행 통계 조회) 명령이 제거되었다. 이러한 명령어 대신에 cubrid statdump 유틸리티를 사용해야 한다.

CUBRIDSUS-5282 CS 모드에서 cubrid unloaddb 유틸리티 수행 시 원래의 레코드 개수보다 적은 개수가 언로드될 수 있는 오류 수정

CS 모드에서 cubrid unloaddb 유틸리티를 수행하는 경우 간혹 내부 버퍼 사용에 문제가 발생하여 원래의 레코드 개수보다 적은 개수가 언로드(unload)될 수 있는 오류를 수정하였다. 기존 버전의 사용자는 SA 모드에서 수행할 것을 권장한다.

CUBRIDSUS-4576 broker_log_runner 명령어에서 질의 계획을 포함하도록 하는 옵션 추가

broker_log_runner 명령어에서 결과 파일에 질의 계획을 포함하도록 하는 -Q 옵션을 추가하였다. -Q 옵션은 -o 옵션이 주어진 경우에만 유효하다.

```
% broker_log_runner -I 192.168.1.10 -P 30000 -d demodb -o result -Q query_convert.in
```

CUBRIDSUS-4991 디스크 볼륨 정보 캐시의 내부 정보가 잘못되는 경우 데이터베이스 재시작에 실패하는 오류 수정

디스크 볼륨 정보 캐시의 내부 정보가 잘못되는 경우 데이터베이스 서버가 비정상 종료한 이후 재시작에 실패하는 오류를 수정하였다.

CUBRIDSUS-4957 인덱스와 데이터 사이에 불일치가 발생하는 경우 이를 인지할 수 있도록 에러 로그 개선

인덱스와 데이터 사이에 불일치가 발생하는 경우, 이를 인지할 수 있도록 에러 메시지를 데이터베이스 서버/클라이언트의 에러 로그 파일에 남기도록 개선하였다.

트랜잭션 격리 수준(isolation level)이 “UNCOMMITED INSTANCE”인 경우, 일시적인 데이터 불일치가 발생할 수 있기 때문에 에러 레벨을 “NOTIFICATION”으로 출력하고, “UNCOMMITED INSTANCE” 이외의 트랜잭션 격리 수준에서는 에러 레벨을 “ERROR” 로 출력한다. 에러 레벨은 error_log_level 파라미터를 통해 변경할 수 있다.

에러 로그 출력의 예는 다음과 같다.

```
-- error_log_level= NOTIFICATION 인 경우 (isolation_level이 1 또는 2로서, UNCOMMITED INSTANCE 허용)
---- 데이터베이스 서버 에러 로그
Time: 03/15/11 15:20:31.804 - NOTIFICATION *** CODE = -545, Tran = 1, CLIENT = cdbs034.cub:csql(3926), EID = 3
Internal error: INDEX u_foo_i ON CLASS foo (CLASS_OID: 0|550|8). Key and OID: 0|600|16 entry on B+tree: 0|209|590 is incorrect. The object does not exist.

-- error_log_level= ERROR 인 경우 (isolation_level이 3 이상)
---- 데이터베이스 서버 에러 로그
Time: 03/15/11 15:14:35.907 - ERROR *** ERROR CODE = -545, Tran = 1, CLIENT = cdbs034.cub:csql(3776), EID = 1
Internal error: INDEX u_foo_i ON CLASS foo (CLASS_OID: 0|550|8). Key and OID: 0|600|2 entry on B+tree: 0|209|590 is incorrect. The object does not exist.
---- 클라이언트 에러 로그
ERROR: Internal error: INDEX u_foo_i ON CLASS foo (CLASS_OID: 0|550|8). Key and OID: 0|600|2 entry on B+tree: 0|209|590 is incorrect. The object does not exist.
```

CUBRIDSUS-4694 데이터베이스 서버 프로세스의 비정상 종료 후 특정 상황에서 데이터베이스 복구에 실패할 가능성이 있는 오류 수정

데이터베이스 서버 프로세스가 비정상 종료한 이후 특정 상황에서 데이터베이스 복구에 실패할 가능성이 있는 오류를 수정하였다.

CUBRIDSUS-4962 데이터베이스 임시 볼륨 공간이 추가로 필요하여 일반 볼륨으로 자동 확장 시 서버 프로세스가 비정상 종료하는 오류 수정

질의 처리 및 정렬(sorting)을 수행할 때 일시적으로 사용되는 임시 볼륨 파일보다 큰 공간이 필요하여 일반 볼륨으로 자동 확장되는 경우, 데이터베이스 서버 프로세스가 비정상 종료하는 오류가 발생하였으나 이를 수정하였다.

CUBRIDSUS-3433 브로커 및 데이터베이스 접속 보안 기능 추가

데이터베이스 사용자 또는 클라이언트 IP를 지정하여, 브로커 프로세스 또는 데이터베이스 서버 프로세스에 접속하는 것을 제어할 수 있도록 보안 관련 기능을 추가하였다. 이와 함께 다음의 명령어가 추가되었다.

```
// BRNAME에 해당하는(생략하면 전체) 브로커에 대한 현재의 접속 설정을 화면에 출력cubrid broker acl status [BRNAME]

// BRNAME에 해당하는(생략하면 전체) 브로커에 대해 접속 설정을 reload. 에러 발생시 예전 설정으로 계속 동작cubrid broker acl reload [BRNAME]

// servername에 해당하는 서버에 대한 현재의 허용 IP 설정을 dispalycubrid server acl status servername

// servername에 해당하는 서버에 대해 허용 IP 설정을 reload. 에러 발생시 예전 설정으로 계속 동작.
cubrid server acl reload servername
```

CUBRIDSUS-4903 브로커 응용 서버(CAS)에서 트랜잭션을 커밋하는 경우 질의 결과 셋을 정리하도록 수정

브로커 응용 서버(CAS)에서 트랜잭션을 커밋하는 경우 질의 결과 셋을 가리키는 주소를 정리하지 않아 이 주소가 다른 요청에 의해 사용되는 문제가 발생하였으나, 브로커 응용 서버(CAS)에서 트랜잭션을 커밋하면서 질의 결과 셋을 정리하도록 수정하였다.

CUBRIDSUS-3985 HA 환경에서 HA 재구성 방법 개선

HA 환경에서 HA 재구성을 위한 스크립트(ha_make_slavedb.sh)를 제공하여 절차를 간소화하도록 하였다. 보다 자세한 설명은 $CUBRID/share/scripts/ha/README을 참고한다.

CUBRIDSUS-4971 HA 환경에서 복제 지연 발생 시 서버 프로세스가 비정상 종료 후 재시작에 실패할 수 있는 오류 수정

HA 환경에서 복제 지연이 발생하면, 로그 전달 스레드와 트랜잭션 처리 스레드가 동시에 보관 로그 파일에 접근하면서 데이터베이스 서버 프로세스가 비정상 종료한 이후 재시작에 실패하는 경우가 존재하였으나 이를 수정하였다.

로그 전달 스레드는 데이터베이스 트랜잭션 로그를 복사하여 다른 서버로 전달하는 역할을 하며, 트랜잭션 처리 스레드는 사용자가 요청한 트랜잭션을 처리하는 역할을 한다.

CUBRIDSUS-4797 HA 환경에서 복제 지연 상태로 HA를 재시작하는 경우 복제 불일치가 발생하는 오류 수정

HA 환경에서 복제 지연이 있는 상태로 HA를 재시작하는 경우 재시작된 copylogdb 프로세스가 내부 정보를 잘못 초기화하여 비정상 종료하면서 복제 불일치가 발생하는 오류를 수정하였다.

CUBRIDSUS-3984 HA 환경에서 에러 메시지 수정

HA 환경에서 발생하는 에러 메시지 내용을 상황에 맞도록 수정하고, 에러 메시지 관련 문제들을 수정하였다.

수정한 에러 메시지 관련 문제는 다음과 같다.

한글 에러 메시지 추가

LANG=ko_KR.utf8 또는 LANG=ko_KR.euckr 인 경우, cubrid backupdb를 수행하면 서버 core 발생하는 문제 수정

디버그 메시지 제거

CUBRIDSUS-3980 HA 환경에서 여러 대의 슬레이브 서버 구성 시 발생할 수 있는 복제 불일치에 대처할 수 있도록 개선

HA 환경에서 한 대의 마스터 서버와 여러 대의 슬레이브 서버를 구성하는 경우 복제 불일치 발생이 가능한 상황에 대처하기 위해, copylogdb 프로세스 및 applylogdb 프로세스가 복제 변경 사항을 로그에 반영하도록 하고 copylogdb 프로세스의 종료 시 시그널에 의한 처리 방식 및 복제 지연 시 동작 방식을 개선하였다.

CUBRIDSUS-3977 HA 환경에서 copylogdb 프로세스의 ASYNC 모드 동작 방식 개선

HA 환경에서 ASYNC 모드인 copylogdb 프로세스가 비정상 종료하는 경우, copylogdb 프로세스가 마스터 서버로부터 받은 트랜잭션은 디스크에 항상 반영(파일 동기화)할 수 있도록 개선하였다.

CUBRIDSUS-3986 HA 환경에서 트랜잭션 로그 복제 상태와 반영 상태를 출력하는 cubrid applyinfo 유틸리티 추가로 운영 편의성 개선

HA 환경에서 트랜잭션 로그 복제 상태와 반영 상태를 출력하는 cubrid applyinfo 유틸리티를 추가하여 운영 편의성을 개선하였다.

```
$ cubrid applyinfo
applyinfo: display CUBRID HA Apply information.
usage: cubrid applyinfo [OPTION] database-name
```

```
valid options:
  -r, --remote-host-name      remote host name; display remote node's active log information
  -a, --applied-info          display applied infomation
  -L, --copied-log-path=PATH  path of copied log volumes; display copied log information
  -p, --pageid=ID             page id; default : 0(active page)
  -v, --verbose               enable verbose status messages; default : disable
```

CUBRIDSUS-3983 HA 환경에서 하나의 트랜잭션 내 다수의 복제 로그 반영 중 발생하는 오류에 대한 처리 방법 개선

HA 환경에서 하나의 트랜잭션 내 다수의 복제 로그를 반영하는 중 오류가 발생하면 오류 발생 이후 트랜잭션 내 복제 로그를 무시하면서 트랜잭션의 일부만 반영되는 문제가 존재하였으나, 오류가 발생하는 경우에도 모든 로그 반영을 시도하도록 개선하였다.

CUBRIDSUS-3180 HA 구동 스크립트(cubrid-ha)의 개선

HA 구동 스크립트인 cubrid-ha를 개선하여, 복제 로그 경로를 사용자가 지정 가능하도록 하였으며 명령어 실패 시 적절한 에러 메시지가 출력될 수 있도록 수정하였다.

CUBRIDSUS-3971 HA 환경에서 applylogdb 프로세스가 비정상 종료 후 재시작 시 복제 로그 반영이 누락될 가능성이 발생하지 않도록 개선

HA 환경에서 applylogdb 프로세스가 비정상 종료 후 재시작하는 경우 복제 로그 반영이 누락될 가능성이 존재하였으나, 이를 발생하지 않도록 개선하였다.

CUBRIDSUS-3885 HA 환경에서 cubrid changemode 유틸리티를 이용하여 HA 서버의 상태 변경 시 변경 가능한 상태를 제한하도록 수정

HA 환경에서 cubrid changemode 유틸리티를 이용하여 HA 서버의 상태를 변경하는 경우, standby에서 maintenance로 변경하거나 maintenance에서 standby로 변경하는 경우만 허용하도록 수정하였다.

CUBRIDSUS-3928 HA 환경에서 보관 로그 파일 개수가 log_max_archives를 초과하여도 슬레이브 서버에 전달되지 않은 로그는 보존하도록 수정

HA 환경에서 아카이브 로그 파일의 개수가 시스템 파라미터에서 설정한 log_max_archives 개수를 초과하는 경우 반영되지 않은 로그가 삭제될 수 있었으나, 이러한 로그는 보존하도록 수정하였다.

CUBRIDSUS-5209 HA 환경에서 로그 반영 프로세스의 메모리 누수 오류 수정

HA 환경에서 로그 반영 프로세스(applylogdb)가 메모리 누수(memory leak)로 인하여 메모리를 과다 사용하게 되는 오류를 수정하였다.

CUBRIDSUS-3981 HA 환경에서 split-brain 후 장애 복구 시 데이터 복제 불일치 오류 수정

HA 환경에서 split-brain 후 장애를 복구하는 경우 마스터 서버와 슬레이브 서버 간 데이터가 일치하지 않는 오류를 수정하였다. split-brain이란, HA 환경에서 마스터 서버가 죽은 것으로 판단한 슬레이브 서버가 마스터 서버로 역할을 바꾸었는데, 기존의 마스터 서버가 살아 있어서 둘다 마스터 서버의 역할을 하게 되는 상황을 가리키는 용어이다.

CUBRIDSUS-3973 HA 환경에서 applylogdb 프로세스의 페이지 버퍼 캐시 성능 개선

HA 환경에서 applylogdb 프로세스의 페이지 버퍼 캐시 알고리즘을 수정하여 로그 반영 속도를 개선하였다.

CUBRIDSUS-5270 HA 환경에서 cub_master 프로세스가 비정상 종료할 수 있는 오류 수정

cub_master 프로세스의 여러 스레드가 동시에 에러 메시지를 출력하는 과정 중에 비정상 종료할 수 있는 오류를 수정하였다.

CUBRIDSUS-4135 HA 환경에서 부하 분산 구성을 지원하도록 기능 추가

HA 환경에서 부하 분산 구성을 지원하도록 cubrid.conf의 ha_mode 파라미터에 replica 값을 추가하고, ha_replica_list 파라미터를 설정할 수 있도록 하였다. 이 기능은 2008 R4.0에 추가되면서 2008 R2.2 Patch9에도 패치된 것으로, 자세한 사항은 2008 R4.0의 매뉴얼을 참고한다.

CUBRIDSUS-5311 보관 로그 파일을 강제로 삭제하도록 할 수 있는 시스템 파라미터 추가

보관 로그를 강제로 삭제할 수 있는 시스템 파라미터가 추가되었다. 이 파라미터가 yes이면 항상 log_max_archives 파라미터의 설정대로 동작하며, no이면 보관 로그를 삭제하지 않고 유지한다. 그러나, 이 파라미터 값이 no이더라도 ha_mode=on이면 HA 프로세스에 의해 복제 반영이 완료된 보관 로그는 삭제될 수 있다.

이 파라미터는 2008 R4.0에 추가되면서 2008 R2.2 Patch9에도 패치된 것으로, 자세한 사항은 2008 R4.0의 매뉴얼을 참고한다.

파라미터 이름	설명
force_remove_log_archives	log_max_archives 에 지정된 수 만큼의 보관 로그만 남기고 나머지를 강제로 삭제할 지 여부를 지정

7.5 CUBRID 2008 R2.2 Patch 8에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-4134 broker_log_top 명령어에서 시간 범위의 입력 방식 및 검색 성능 개선

broker_log_top으로 브로커의 로그를 확인할 때 시간 범위를 지정하는 옵션인 -F(시작 시간)와 -T(끝 시간)의 시간 형식에 밀리초(msec)까지 검색이 가능하도록 하고, 검색 성능을 개선하였다.

참고로 시간 범위의 입력 형식은 “MM[/DD[hh[:mm[:ss[.msec]]]]]”을 따르며 []로 감싼 부분은 생략이 가능하다. 생략되면 DD는 01이, 나머지 부분은 0이 기본값으로 정해진다.

다음 예는 broker_log_top을 이용하여 지정한 시간 범위의 브로커 로그 분석 결과 파일을 생성하도록 한다.

```
# 밀리 초까지 검색 범위를 설정한다.
broker_log_top -F "01/19 15:00:25.000" -T "01/19 15:15:25.180" log1.log

# 아래의 옵션 값에서 시간 형식이 생략된 부분은 기본값 0으로 정해진다. 즉, -F "01/19
00:00:00.000" -T "01/20 00:00:00.000" 을 입력한 것과 같다.
broker_log_top -F "01/19" -T "01/20" log1.log
```

CUBRIDSUS-4321 브로커 파라미터인 SQL_LOG 값을 동적으로 변경하는 명령어에 특정 응용 서버(CAS) 파라미터만 변경할 수 있도록 옵션 추가

broker_changer 명령어로 브로커의 재시작 없이 특정 응용 서버(CAS)만 SQL_LOG 값을 변경하는 것이 가능하도록 아래의 형식과 같이 응용 서버 식별자(<cas_id>)를 지정할 수 있는 기능을 추가하였다. <cas_id>는 cubrid broker status 명령어에서 출력되는 ID이다.

```
broker_changer <broker_name> [<cas_id>] SQL_LOG <value>
```

참고로 broker_changer는 Linux 버전에서만 사용이 가능하다.

CUBRIDSUS-4440 호스트 변수에 잘못된 값을 바인딩하여 질의 수행에 실패하면 정상 값을 바인딩하여 재수행해도 계속 실패하는 오류 수정

플랜 캐시를 사용하지 않는 질의에서 호스트 변수에 잘못된 값을 바인딩한 뒤 질의 수행에 실패하는 경우, 이후 정상적인 값을 바인딩하고 수행해도 계속 실패하는 현상을 수정하였다.

참고로 데이터베이스 서버 기본 설정에서는 플랜 캐시를 사용하도록 max_plan_cache_entries 파라미터 값이 설정되어 있지만, INSERT 문에서는 설정 값과 무관하게 항상 플랜 캐시를 사용하지 않는다. (플랜 캐시와 관련하여 매뉴얼의 쿼리 캐시 관련 파라미터에서 max_plan_cache_entries 참고)

CUBRIDSUS-4233 주기적으로 보관 로그를 생성하는 작업 중에 로그 페이지가 일부 쓰여지지 않는 문제 수정

주기적으로 보관 로그를 생성하는 작업 중에 로그 볼륨에 로그 페이지가 일부 쓰여지지 않아 복구에 실패하는 문제를 수정하였다.

이러한 문제가 발생한 예로, cubrid createdb 유틸리티로 데이터베이스 페이지 크기(--page-size)와 로그 볼륨 페이지 크기(--log-page-size)를 다르게 생성한 데이터베이스에서 DB 서버 프로세스가 비정상 종료하는 경우, 데이터베이스의 복구가 잘못되어 SELECT 질의를 수행하면 오류가 발생할 수 있다.

CUBRIDSUS-4645 로그 페이지 버퍼의 제어 방식을 변경하여 DB 서버 프로세스의 멈춤 오류 수정

데이터베이스 시스템에서 실행되는 여러 쓰레드들이 로그 페이지 버퍼를 사용하고자 할 때 이를 제어하는 방식을 변경하여, 주기적으로 보관 로그를 생성하는 작업에서 로그 페이지 버퍼를 사용하려는 특정 시점에 DB 서버 프로세스가 동작을 멈추는(hang) 오류를 수정하였다.

7.6 CUBRID 2008 R2.2 Patch 7에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-4372 CUBRID broker 유틸리티에 reset 기능 추가

HA 에서 failover 등으로 브로커 응용 서버(CAS)가 원하지 않는 데이터베이스 서버에 연결되었을 때 명령을 통해 기존 연결을 끊고 새롭게 연결할 수 있도록 CUBRID broker 유틸리티에 reset 기능을 추가하였다.

예를 들어 Read Only 브로커가 active server와 연결된 후에는 standby server가 연결이 가능한 상태가 되더라도 자동으로 standby server와 재연결하지 않으며, "cubrid broker reset" 명령을 통해야만 기존 연결을 끊고 새롭게 standby server와 연결할 수 있다.

CUBRIDSUS-4183 브로커의 상태 정보 출력 기능에 ACCESS_MODE와 SQL_LOG 출력 추가

브로커 상태 정보 출력을 위한 cubrid broker status -b 명령 사용 시 -f 옵션을 추가로 사용하면 브로커 설정 파일(cubrid_broker.conf)의 ACCESS_MODE 파라미터와 SQL_LOG 파라미터의 설정 값을 추가로 출력하도록 하였다.

```
% cubrid broker status -b -f -l 2
@ cubrid broker status
NAME          PID    PSIZE PORT  AS(T W B 2s-W 2s-B) JQ REQ TPS QPS LONG-T LONG-Q ERR-Q
```

```
CANCELED
ACCESS_MODE SQL_LOG
================================================================================================
query_editor 16784 56700 38000    5 0 0 0     0      0  0    0    0   0/60.0 0/60.0 0      0
          RW       ALL
```

CUBRIDSUS-3343 브로커와 JDBC 응용 프로그램 간 최대 응답 시간을 줄이도록 개선

브로커 응용 서버(CAS)와 응용 프로그램 사이에서 사용하는 네트워크 버퍼 크기를 줄여 JDBC 응용 프로그램의 최대 응답 시간이 줄어들도록 개선하였다.

JDBC 드라이버는 네트워크 버퍼를 통해 얻은 질의 수행 결과를 Java 객체로 생성하여 관리한다. 이때 생성된 객체가 차지하는 메모리 영역으로 인해 자주 JVM의 Full GC(Garbage Collection)가 발생하여 응답 시간이 늘어날 수 있으므로 네트워크 버퍼의 크기를 기존의 128K에서 16K로 줄여 Full GC의 발생 빈도가 낮아지도록 하였다.

이 수정은 CCI, PHP, ODBC 응용 프로그램에도 적용되지만, JDBC 응용 프로그램과 같이 성능 상의 이익이 발생하지는 않는다.

CUBRIDSUS-3833 CUBRID addvoldb 유틸리티의 실행 성능 개선

2008 R2.2에서 추가된 플러시 제어(flush control) 기능으로 인해 CS 모드에서 CUBRID addvoldb 유틸리티의 실행이 느려지는 문제를 개선하였다. (CUBRIDSUS-2216 플러시 제어(flush control) 방식 구현 및 관련 파라미터 추가 참고)

CUBRIDSUS-4217 문자열 중간에 NULL 문자가 포함된 데이터를 내보내기(unload)하는 경우 데이터 파일이 잘못 생성되는 오류 수정

VARCHAR, CHAR와 같은 문자 타입 컬럼이 있는 레코드 값의 중간에 ASCII CODE 값 0(NULL 문자)이 포함되어 있는 경우, unloaddb 유틸리티가 데이터 파일을 잘못 생성하여 이를 loaddb 유틸리티로 로딩하는데 실패하였으나, 데이터 파일을 정상적으로 생성하여 데이터 파일 로딩이 성공하도록 오류를 수정하였다.

CUBRIDSUS-4341 응용 프로그램의 연결 요청이 집중되는 경우 브로커 연결에 실패할 수 있는 오류 수정

응용 프로그램의 연결 요청이 브로커 프로세스(cub_broker)에 일시적으로 집중되는 경우, 브로커 연결에 실패할 수 있는 오류를 수정하였다.

CUBRIDSUS-4347 HA 환경에서 DB 연결을 설정하는 중에 발생하는 오류 수정

HA 환경에서 브로커 서버의 DB 호스트 접속 순서가 databases.txt에 host1:host2로 되어 있고 마스터 DB는 host1, 슬레이브 DB는 host2인 환경에서 브로커의 ACCESS_MODE=SO(Slave Only)로 설정되어 있는 경우, 슬레이브 DB를 찾는 과정에서 처음에 연결했던 host1은 슬레이브 DB가 아니므로 브로커 응용 서버(CAS)와 DB 서버 간의 연결을 종료하여야 하나, 연결이 종료되지 않고 남아 있는 오류를 수정하였다.

또한, HA 환경에서 DB 서버가 허용하는 최대 클라이언트 개수(cubrid.conf의 max_client)를 초과하는 경우, 브로커 서버의 databases.txt에 있는 순서에 따라 다음 호스트로 DB 연결을 시도하여야 하나 그렇지 않은 오류를 수정하였다.

CUBRIDSUS-4295 HA 환경에서 인덱스 이름만을 명시한 DROP INDEX 구문이 슬레이브 DB에 반영되지 않는 오류 수정

HA 환경의 마스터 서버에서 아래의 예와 같이 테이블 명과 컬럼 명을 생략하고 인덱스 명만을 명시하여 DROP INDEX 구문을 수행하는 경우, 슬레이브 DB에 반영되지 않는 오류를 수정하였다.

```
// 인덱스 명만 명시하여 DROP INDEX 수행
DROP INDEX idx_abc;
```

CUBRIDSUS-4218 다량의 페이지를 가지는 내부 파일 생성 요구가 실패하는 오류 수정

인덱스 생성과 같이 한번에 많은 양의 페이지를 요구하는 작업을 수행하는 경우, 내부 파일 관리 맵이 잘못 되어 다음과 같은 에러를 발생시키면서 페이지 할당이 실패할 수 있는 오류를 수정하였다.

```
Internal error: fetching deallocated pageid -2146798529 of volume
"/home/qa/db/testdb/testdb_t32763".
```

CUBRIDSUS-4272 -327번 에러가 발생할 수 있는 오류 수정

동시 사용자 환경에서 다음과 같이 -327번 에러가 발생할 수 있는 오류를 수정하였다.

```
Time: 12/01/10 17:14:13.728 - ERROR *** ERROR CODE = -327, Tran = 5, EID = 3
Cannot create MOP with NULL OID.
```

CUBRIDSUS-4302 cci_disconnect() 함수를 호출하는 프로그램이 비정상 종료하는 오류 수정

ODBC 드라이버에 UTF-8 문자셋을 지원하기 위해 추가한 코드의 오류로 인해 CCI API 함수인 cci_disconnect() 를 호출하는 프로그램이 비정상 종료하였으나, 이를 수정하였다. (CUBRIDSUS-4091 ODBC 드라이버에 UTF-8 문자셋 지원 참고)

7.7 CUBRID 2008 R2.2 Patch 6에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-3879 브로커의 상태 정보 출력 기능에 응용 서버(CAS) 정보 포함 옵션 추가

브로커 상태 정보 출력을 위한 cubrid broker status -b 명령 사용 시 -f 옵션을 추가로 사용하면 응용 서버(CAS) 정보를 포함하도록 개선하였다.

-f 옵션에 -l 옵션을 추가하면 지정한 초 동안 대기 상태인 응용 서버의 개수(Ns-W)와 수행 상태인 브로커 응용 서버(CAS)의 개수(Ns-B)를 출력한다. (-l 옵션을 생략하면 기본값은 1초이다.)

새로 추가한 응용 서버(CAS) 정보는 다음과 같다.

항목 이름	설명
T	실행 중인 응용 서버(CAS)의 전체 개수
W	현재 클라이언트 대기(Waiting) 상태인 응용 서버(CAS)의 개수
B	현재 클라이언트 수행(Busy) 상태인 응용 서버(CAS)의 개수
Ns-W	N초 동안 클라이언트 대기(Waiting) 상태였던 응용 서버(CAS)의 개수
Ns-B	N초 동안 클라이언트 수행(Busy) 상태였던 응용 서버(CAS)의 개수
CANCELED	브로커 시작 이후 사용자 인터럽트로 인해 취소된 질의의 개수 (-s N 옵션과 함께 사용하면 N초 동안 누적된 개수)

```
// 브로커 상태 정보 실행 시 -f 옵션 추가. -l 옵션은 N초 동안의 Ns-W, Ns-B를 출력하도록 초를 설정
% cubrid broker status -b -f -l 2
@ cubrid broker status
NAME          PID    PSIZE PORT  AS(T W B 2s-W 2s-B) JQ REQ TPS QPS LONG-T LONG-Q ERR-Q
CANCELED
==========================================================================================
query_editor 16784 56700 38000    5 0 0 0     0      0  0    0    0   0/60.0 0/60.0 0      0
```

CUBRIDSUS-4091 ODBC 드라이버에 UTF-8 문자셋 지원

ODBC 드라이버에 UTF-8 문자셋 지원 기능을 추가하여 UTF-8 문자열로 입력 및 조회가 가능하도록 개선하였다. UTF-8 이외에 다른 문자셋은 지정할 수 없으며, 아래의 예와 같이 연결 문자열에 "CHARSET=utf-8" 속성을 추가하는 방식으로 사용할 수 있다.

```
// 연결 문자열에 CHARSET 속성을 UTF-8로 설정
DRIVER=CUBRID Driver;UID=public;FETCH_SIZE=100;PORT=30000;SERVER=localhost;DB_NAME=demodb;DESCR
IPTION=cubrid_odbc;CHARSET=utf-8
```

CUBRIDSUS-3771 deadlock 및 lock timeout 발생 시 데이터베이스 서버의 에러 로그 메시지 개선

데이터베이스 서버의 에러 로그 파일에 deadlock 발생 시 deadlock 메시지를 추가하고, lock timeout 발생 시 기존의 lock timeout 메시지에 lock timeout의 원인을 제공한 프로세스 이름과 프로세스 아이디를 출력하도록 개선하였다.

```
// deadlock 이 발생하는 경우, deadlock 메시지와 함께 이를 유발한 호스트/프로세스 정보 출력
Time: 08/23/10 12:06:11.215 - NOTIFICATION *** CODE = -993, Tran = 0, CLIENT =
(unknown):(unknown)(-1), EID = 7
deadlock cycle is detected. public@cdbs006.cub¦csql(19502), public@cdbs006.cub¦csql(19501),
public@cdbs006.cub¦csql(19500).

// lock timeout이 발생하는 경우, 에러 메시지의 제일 끝에 프로세스 이름과 프로세스 아이디를 출력
Time: 08/23/10 12:06:11.215 - ERROR *** ERROR CODE = -967,Tran = 3, CLIENT =
cdbs006.cub:csql(19502), EID = 8
Your transaction (index 3, public@cdbs006.cub¦19502) timed out waiting on IS_LOCK lock on class
tb1 because of deadlock. You are waiting for user(s) public@cdbs006.cub¦csql(19500) to finish.
```

CUBRIDSUS-3835 보존할 보관 로그 파일의 최대 개수를 동적으로 변경할 수 있도록 개선

데이터베이스를 재시작하지 않고 보존할 보관 로그 파일의 최대 개수를 동적으로 변경할 수 있도록 개선하였다.

데이터베이스 구동 시에는 cubrid.conf에서 사용된 log_max_archives 파라미터의 값이 보존할 보관 로그 파일의 최대 개수로 설정된다.

아래의 예와 같이 csql을 이용하여 변경하는 경우에는 데이터베이스의 dba 계정으로 접속해야 한다.

```
//dba 계정으로 실행한 csql에서 log_max_archives 값을 동적으로 변경
csql>;set log_max_archives=5
```

CUBRIDSUS-3532 작업해야 할 응용 클라이언트의 개수가 응용 서버(CAS)의 개수보다 많은 경우 연결 요청의 처리 오류 수정

작업해야 할 응용 클라이언트의 개수가 응용 서버(CAS)의 개수보다 많은 경우, 브로커(cub_broker)가 트랜잭션 처리가 끝난 응용 서버(CAS)에 응용 클라이언트의 새로운 연결 요청을 할당하지 못하는 오류를 수정하였다.

CUBRIDSUS-3032 브로커 설정 파일에서 KEEP_CONNECTION 파라미터의 값이 ON 인 경우 트랜잭션이 커밋되지 않는 오류 수정

브로커 설정 파일(cubrid_broker.conf)에서 KEEP_CONNECTION 파라미터의 값이 ON으로 설정되어 있는 경우, 클라이언트와 브로커간의 프로토콜 오류로 인하여 클라이언트에서 커밋을 요청한 트랜잭션이 데이터베이스에 반영되지 않는 오류를 수정하였다.

7.8 CUBRID 2008 R2.2 Patch 5에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-4004 ALTER TABLE 질의 실패 후 COMMIT을 수행하면 해당 테이블 조회 시 에러가 발생할 수 있는 오류 수정

기본 키 제약 조건을 추가로 정의하는 ALTER TABLE문이 정상 실행되지 않고 에러를 출력한 상태에서 해당 트랜잭션을 커밋한 직후 해당 테이블에 UPDATE문을 실행하여 데이터를 수정한 경우, 서버에 캐시된 변경 내역을 불완전 롤백 처리하는 오류로 인해 데이터베이스 서버를 재 시작한 이후 해당 테이블을 조회할 때 ERROR: Unknown representation identifier라는 에러 메시지가 출력되었으나, 이를 수정하였다.

CUBRIDSUS-3910 HA 환경에서 applylogdb 프로세스가 재시작 후 비정상 종료되면서 데이터 불일치가 발생되는 오류 수정

HA 구성된 데이터베이스의 페이지 크기(--page-size 옵션 사용)가 로그 페이지 크기(--log-page-size 옵션 사용)보다 크게 설정되어 생성된 경우, HA 운영 중에 로그 레코드가 미리 할당한 메모리 영역을 초과하여 기록되면서 로그를 슬래이브 DB로 반영할 때 applylogdb 프로세스가 비정상 종료되고 이로 인해 데이터 불일치가 발생되는 오류가 있었으나, 이를 수정하였다.

CUBRIDSUS-3911 HA 환경에서 온라인 백업 후 copylogdb 프로세스가 잘못된 보관 로그를 생성하면서 특정 시점부터 복제가 중단되는 오류 수정

HA 환경에서 마스터 DB를 온라인 백업한 이후 복제가 지연되는 특정 상황에서 copylogdb 프로세스가 빈 보관 로그를 생성하는 오류로 인해 특정 시점부터 copylogdb 프로세스가 반복적으로 재시작되면서 복제가 중단되는 문제가 발생하였으나, 이를 수정하였다.

CUBRIDSUS-4027 HA 환경에서 브로커의 ACCESS_MODE를 동적으로 변경할 수 있도록 broker_changer 유틸리티 개선

HA 환경에서 broker_changer 유틸리티를 사용하여 특정 브로커의 ACCESS_MODE 값을 변경하고 해당 브로커는 자동으로 reset 되도록 개선하였다.

```
//ACCESS_MODE를 RO로 변경하고 브로커 reset
broker_changer broker_name access_mode ro

//ACCESS_MODE를 RW로 변경하고 브로커 reset
broker_changer broker_name access_mode rw
```

CUBRIDSUS-3051 테이블 통계 정보를 갱신하는 작업이 정상 실행되지 않는 오류 수정

테이블 통계 정보를 갱신하는 UPDATE STATISTICS문을 실행하거나 cubrid checkdb 또는 cubrid optimizedb 유틸리티를 실행하는 경우, 통계 정보를 갱신할 테이블 ID를 가져오는 알고리즘의 오류로 인해 Unknown class identifier: 0|983040|4962와 같은 메시지가 출력되면서 정상 실행되지 않는 문제를 수정하였다.

7.9 CUBRID 2008 R2.2 Patch 4에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-1040, 1041 CCI, PHP API에서 OID 값을 사용하여 컬럼 값을 변경하고 커밋한 경우, 변경 내역이 반영되지 않는 오류 수정

CCI 라이브러리의 cci_oid_put( ) 또는 cci_oid( ) 함수를 사용하여 주어진 OID 값에 해당하는 컬럼 값을 갱신 또는 삭제하고 cci_tran_commit( )함수를 사용하여 해당 트랜잭션을 커밋하는 경우, 해당 커밋이 정상적으로 반영되지

않는 오류가 있었다. 이러한 오류는 CCI 라이브러리를 기반으로 구현된 PHP 라이브러리의 cubrid_put( ), cubrid_drop( ) 함수를 사용하는 경우에도 동일하게 발생하였으나 함께 수정되었다.

CUBRIDSUS-3792 SELECT문에서 부 질의를 포함하는 경우, 질의 결과 오류 수정

WHERE 절을 포함하는 부 질의가 SELECT문의 WHERE 절에 명시되는 경우, 부 질의를 내부적으로 재작성(rewrite)하는 알고리즘의 오류로 인해 잘못된 질의 결과가 출력되는 문제를 수정하였다.

```
//잘못된 결과를 출력한 질의 유형
SELECT * FROM tb1 A WEHRE (a, b, c) IN (SELECT a, b, c FROM tb1 B WHERE c=1);
```

CUBRIDSUS-3682 특정 상황에서 섹터가 반복적으로 할당되는 오류 수정

대량 데이터 저장 또는 인덱스 로딩이 수행된 이후 볼륨 내 여유 공간이 소진된 상황에서 페이지 할당을 요청했을 때, 섹터 할당이 반복적으로 발생하면서 아카이브 로그가 증가하는 오류가 발생하여 관련 알고리즘을 수정하였다.

CUBRIDSUS-3806 ha_mode=OFF인 환경에서 cubrid heartbeat list 명령 실행 시, 프로세스가 비정상 종료되는 오류 수정

HA 모드로 구동되지 않는 환경(ha_mode=off)에서 HA 노드 상태 확인을 위하여 cubrid-ha status 또는 cubrid heartbeat list 명령어를 실행하는 경우, cub_master 프로세스가 비정상 종료되는 오류를 수정하였다.

CUBRIDSUS-3968 ha_mode=to-be-active 인 서버가 active 모드로 전환되기 전에 클라이언트가 wake-up하는 오류 수정

HA 환경에서 to-be-active 모드인 서버에 접속하는 클라이언트가 정상적으로 대기하지 못하고 해당 서버 모드가 로그 반영을 완료하고 active로 전환되기 전에 wake-up하는 오류가 있었으나 이를 수정하였다.

CUBRIDSUS-3725 멀티 유저 환경에서 질의 실행 도중 CAS 프로세스가 비정상 종료되는 경우, 데이터 불일치 오류 수정

멀티 유저 환경에서 INSERT/UPDATE/DELETE 질의 실행 도중에 CSQL 또는 CAS 프로세스가 비정상 종료되는 경우, 이로 인해 발생한 인터럽트를 처리하는 알고리즘 오류로 인해 데이터 불일치가 발생하였으나 이를 수정하였다.

CUBRIDSUS-3800 CAS 프로세스가 서버 접속한 직후 비정상 종료되는 경우, 연결 핸들이 정리되지 않는 오류 수정

CSQL 또는 CAS 프로세스가 서버에 접속하고 질의가 실행되기 전에 해당 프로세스가 비정상 종료되는 경우, 트랜잭션이 정리되지 않은 상태로 서버와의 연결이 유지되는 오류가 있었으나 이를 수정하였다.

CUBRIDSUS-3853,3854 Big Endian 방식 시스템에서 ROUND 함수 및 MOD 함수에서 SMALLINT와 BIGINT 타입 값에 대한 연산 오류 수정

MOD( ) 함수 또는 ROUND( ) 함수의 인자로 BIGINT 또는 SMALLINT 타입 값이 주어진 경우 잘못된 결과 값을 반환하는 오류를 수정하였다. 이는 Big Endian 방식으로 값을 저장하는 운영 체제(예: AIX)에서만 재현되는 오류이며, Windows 및 Linux 환경에서는 정상적인 결과 값을 반환한다.

7.10 CUBRID 2008 R2.2 Patch 3에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-3627 크기가 큰 데이터를 연산하는 도중 발생되는 “all page buffers are fixed.” 오류 수정

데이터 크기가 크거나 질의에 관한 중간/최종 결과 리스트 파일의 크기가 큰 경우, 연산에 관한 임시 결과를 캐시할 메모리 버퍼 페이지가 모두 소진되어 "All page buffers are fixed." 오류가 발생되는 문제를 수정하였다.

CUBRIDSUS-3611 HA 환경에서 로그 페이지 크기가 4KB를 초과하는 경우, 마스터 서버 재시작 이후 active 모드로 변경되지 않는 오류 수정

HA 환경에서 로그 페이지 크기가 디폴트 값인 4KB보다 크게 설정되고, 마스터 서버 프로세스가 종료된 이후 재시작된 경우, applylogdb 프로세스의 초기화 작업 오류로 인해 마스터 서버가 to-be-active 모드에서 active 모드로 변경되지 않는 오류가 있었으나, 이를 수정하였다.

CUBRIDSUS-3617 HA 환경에서 브로커가 슬레이브 서버로 연결 시도 중 발생하는 "max clients exceeded" 오류 수정

HA 환경에서 브로커가 마스터 서버에서 슬레이브 서버로 연결을 시도하는 경우, 일부 클라이언트의 연결 리소스가 2개씩 사용되는 동작 오류로 인해 "max clients exceeded" 메시지가 출력되면서 연결 실패하는 문제를 수정하였다.

CUBRIDSUS-3639 HA 환경에서 트랜잭션 수행 도중 마스터 서버 프로세스 종료 시, 데이터 불일치 오류 수정

HA 환경에서 대량의 데이터가 지속적으로 마스터 DB로 입력되는 상황에서 마스터 서버의 cub_server 프로세스가 종료되는 경우, 종료 시점의 트랜잭션 로그가 슬레이브 DB로 복제되지 않는 오류를 수정하였다.

CUBRIDSUS-3650 HA 환경에서 대량 데이터 입력 시, 슬레이브 DB로 복제되지 않는 오류 수정

HA 환경에서 마스터 DB에 대량의 데이터가 입력되고 이로 인해 해당 페이지 락(page lock)이 설정되는 특정 상황에서 슬레이브 DB로 일부 데이터가 복제되지 않는 오류를 수정하였다.

CUBRIDSUS-3700 HA 환경에서 applylogdb 프로세스가 로그 반영 중 비정상 종료되는 오류 수정

HA 환경에서 슬레이브 DB의 applylogdb 프로세스가 복제 로그를 반영하는 도중 비정상 종료되고, cub_master 프로세스에 의해 무한히 재시작되는 오류를 수정하였다.

CUBRIDSUS-496 복제/HA 환경에서 마스터 DB에서 ALTER문을 사용하여 2개 이상의 컬럼을 삭제하는 경우, 슬레이브 DB에 반영되지 않는 오류 수정

복제 또는 HA 환경에서 ALTER TABLE tbl_name DROP col1, col2, col3;과 같은 질의문을 수행하여 마스터 DB에서 여러 개의 컬럼을 삭제한 경우, 변경된 스키마가 슬레이브 DB에 반영되지 않는 오류를 수정하였다.

CUBRIDSUS-3626 PHP 드라이버에서 cubrid_connect_with_url( ) 함수 추가 지원

PHP 드라이버에 cubrid_connect_with_url( ) 함수를 추가 지원하여, URL 인자에 데이터베이스 연결 정보를 명시할 수 있도록 하였다.

```
int cubrid_connect_with_url (char *url [, char *db_user, char *db_password ])
```

CUBRIDSUS-3645 Windows 환경에서의 PHP 드라이버 빌드 오류 수정

Windows 환경에서 PHP 드라이버가 정상적으로 빌드되지 않는 오류를 수정하였다.

CUBRIDSUS-3341 매니저 서버 프로세스의 메모리 누수 오류 수정

매니저 서버 프로세스(cm_common) 동작 중에 메모리가 누수(leak)되는 오류를 수정하였다.

CUBRIDSUS-3708 JDBC에서 날짜/시간 관련 타입 처리 방식 수정

JDBC에서 날짜/시간 관련 타입(TIME, TIMESTAMP, DATETIME, DATE) 연산을 수행하는 방식을 수정하여 성능을 개선시키고, TIME 타입에 대한 연산 수행 시 java.lang.NumberFormatException 오류가 발생되면서 정상 처리되지 않는 문제를 수정하였다.

CUBRIDSUS-3695 독립 모드로 수행 중인 CSQL가 강제 종료될 경우 트랜잭션이 비정상 롤백되는 오류 수정

CSQL에서 독립 모드(standalone)로 트랜잭션 수행 도중 커밋하지 않은 상태에서 CSQL 프로세스가 강제 종료되는 경우, 해당 트랜잭션을 정상적으로 롤백 처리하지 못하는 오류로 인해 ERROR: Internal system failure: no more specific information is available. 메시지가 출력되었으나 이를 수정하였다.

7.11 CUBRID 2008 R2.2 Patch 2에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-3523 CCI 및 PHP 응용 환경에서 cci.log 파일이 생성되는 문제 수정

CCI 라이브러리의 오류로 인해 CUBRID 실행 디렉토리 내에 cci.log 라는 불필요한 로그 파일이 생성되는 문제를 수정하였다.

CUBRIDSUS-3489 HA 환경에서 applylogdb 프로세스가 반복적으로 재 시작되며 복제 진행되지 않는 오류 수정

HA 환경에서 마스터 DB의 복제 로그 정보와 슬레이브 DB의 applylogdb 프로세스가 반영하려는 페이지 정보의 불일치로 인해 applylogdb 프로세스가 반복적으로 재시작되면서 복제가 진행되지 않는 오류가 있었으나, 이를 수정하였다. 이러한 오류는 쓰기 연산이 드물게 수행됨에도 불구하고 데이터베이스를 빈번하게 백업하는 환경에서 발생할 수 있다.

CUBRIDSUS-3491 멀티쓰레드 환경에서 REUSE_OID 테이블에 대해 INSERT/DELETE 연산을 반복 수행 시, 서버 오류가 출력되는 문제 수정

멀티쓰레드 환경에서 REUSE_OID 옵션이 적용된 테이블에 대해 INSERT 연산과 DELETE 연산을 반복적으로 수행하는 경우, 아래와 같은 서버 오류가 발생하는 문제를 수정하였다. 이러한 문제는 5개의 쓰레드가 INSERT 연산과 DELETE 연산을 200개씩 수행하는 환경에서 재현되었다.

```
Time: 06/30/10 16:14:08.309 - ERROR *** ERROR CODE = -45, Tran = 3, CLIENT =
cdbs005.cub:broker1_cub_cas_3(10674), EID = 1
Slot 2 on page 50 of volume "/DB/foo_x001" is allocated to an anchored record.  A new record
cannot be inserted here.
```

7.12 CUBRID 2008 R2.2 Patch 1에서 변경된 사항

수정/개선된 사항

CUBRIDSUS-3106 HA 환경에서 특정 커밋 로그 유실로 인해 복제가 중단되는 오류 수정

HA 환경에서 특정 아카이브 로그에 의도하지 않은 EOL(End-Of-Log)가 기록되는 오류로 인해 트랜잭션 커밋 로그가 유실되어 복제가 중단되는 현상이 발생하였으나, 이를 수정하였다.

발생 버전: CUBRID 2008 R2.1 R2.2

CUBRIDSUS-3399,3434,3435 HA 환경에서 applylogdb 프로세스의 CPU 사용률이 증가하고 복제가 중단되는 오류 수정

HA 환경에서 applylogdb 프로세스가 특정 구간의 로그를 반복적으로 반영하는 오류로 인해 CPU를 100% 점유하면서 복제가 중단되는 현상이 발생하였으나, 이를 수정하였다.

발생 버전: CUBRID 2008 R2.1 R2.2

CUBRIDSUS-3194 HA 환경에서 applylogdb 프로세스의 CPU 사용률이 증가하는 오류 수정

HA환경에서 복제 로그를 슬레이브 DB로 반영할 때 NULL 페이지를 처리하는 알고리즘의 오류로 인해 applylogdb 프로세스의 메모리가 지속적으로 증가하는 현상이 발생하였으나, 이를 수정하였다.

발생 버전: CUBRID 2008 R2.1 R2.2

CUBRIDSUS-3196 HA 환경에서 applylogdb 프로세스가 반복적으로 재 시작되는 현상 수정

HA환경에서 마스터 DB의 복제 로그 정보와 슬레이브 DB의 applylogdb 프로세스가 반영하려는 페이지 정보의 불일치로 인해 applylogdb 프로세스가 반복적으로 재 시작되면서 복제가 진행되지 않는 오류가 있었으나, 이를 수정하였다.

발생 버전: CUBRID 2008 R2.1 R2.2

CUBRIDSUS-3135 복제 환경에서 복제 로그 크기가 비정상적으로 증가하는 오류 수정

마스터 DB에서 복제가 지연되는 특정 상황에서 복제 로그가 슬레이브 DB로 복사된 이후에도 마스터 DB의 복제 로그 크기가 비정상적으로 증가하는 오류가 있었으나, 이를 수정하였다.

발생 버전: CUBRID 2008 R2.2 이하

CUBRIDSUS-2886 JDBC 를 통한 서버 접속 요청이 집중되는 환경에서 접속 실패하는 현상 수정

JDBC 드라이버를 통해 동시에 수십 개의 쓰레드가 집중적으로 서버에 접속하여 INSERT를 수행하는 환경에서 아래의 에러 메시지가 출력되면서 접속 실패하는 현상이 발생하였으나, 이를 수정하였다.

JDBC 에러 메시지

```
cubrid.jdbc.driver.CUBRIDException: Failed to connect to database server, 'address', on the
following host(s): localhost
at cubrid.jdbc.driver.CUBRIDConnection.prepare(CUBRIDConnection.java:676)
at cubrid.jdbc.driver.CUBRIDConnection.prepare(CUBRIDConnection.java:917)
at cubrid.jdbc.driver.CUBRIDConnection.prepareStatement(CUBRIDConnection.java:137)
at worker.addr_insert(foo_address.java:170)
at worker.run(foo_address.java:44)
```

브로커 에러 로그

```
Time: 04/29/10 19:36:10.258 - ERROR *** ERROR CODE = -353, Tran = -1, EID = 1
Cannot make connection to master server.... Resource temporarily unavailable

Time: 04/29/10 19:36:10.258 - ERROR *** ERROR CODE = -677, Tran = -1, EID = 2
Failed to connect to database server, 'address', on the following host(s): localhost
```

발생 버전: CUBRID 2008 R2.2 이하

CUBRIDSUS-3083 UNIQUE KEY 컬럼에 중복 값이 입력되는 오류 수정

단일 UPDATE문을 사용하여 다중 행의 컬럼 값을 변경하는 경우, UNIQUE 제약 조건이 정의된 컬럼에 중복된 값이 저장되는 오류를 수정하였다.

발생 버전: CUBRID 2008 R2.2

7.13 CUBRID 2008 R2.2에서 변경된 사항

새로 추가된 기능

CUBRIDSUS-2024 HA 기능에 노드 장애 검출 및 리소스 관리 기능 추가

기존의 HA 기능에서 노드 장애 검출 및 리소스 관리를 위한 기능을 추가함에 따라, 아래의 관련 파라미터가 추가되었다. 또한, HA 관련 리소스를 관리할 수 있는 cubrid heartbeat 유틸리티와 cubrid-ha 스크립트를 제공하여 보다 쉽게 HA 환경을 구축할 수 있도록 하였다.

아래는 이를 위해 추가된 HA 기능 관련 파라미터이다.

- HA_PORT_ID: HA 기능에 사용되는 메시지 송수신을 위한 UDP 포트를 설정하는 파라미터이다.
- HA_NODE_LIST: HA노드 그룹 식별자 및 그룹 내 멤버 노드를 지정하는 파라미터이다.

```
#cubrid.conf
[common]
...
HA_MODE = on
HA_PORT_ID = 41523
HA_NODE_LIST = group_id@server_s1:server_s2
```

- 관련 문서: 매뉴얼> 관리자 안내서> CUBRID HA

CUBRIDSUS-2475 OID 재사용을 위한 테이블 생성 옵션 추가

OID(object identifier)를 재사용할 수 있는 테이블 옵션(**REUSE_OID**)을 제공한다. REUSE_OID 옵션이 적용된 테이블에 대해 데이터가 삭제되면 해당 OID도 함께 삭제되고, 이후 삽입되는 새로운 데이터가 해당 OID를 재사용할 수 있다. 단, OID 재사용 테이블을 다른 테이블이 참조할 수 없고, 응용 프로그램에서 OID 재사용 테이블 내 객체들의 OID 값을 조회할 수 없다.

```
CREATE TABLE reuse_tbl (a INT PRIMARY KEY) REUSE_OID;
```

- 관련 문서: 매뉴얼> SQL 설명서> 데이터 정의> CREATE TABLE> 테이블 옵션

CUBRIDSUS-2216 플러시 제어(flush control) 방식 구현 및 관련 파라미터 추가

대량 데이터가 지속적으로 삽입되는 서비스에서 특정 시점에 I/O 부하가 집중되지 않도록 플러시 동작을 제어할 수 있는 기능을 지원한다. 아래는 이를 위해 추가된 파라미터이다.

ADAPTIVE_FLUSH_CONTROL: 현 시점의 워크로드에 따라 플러시 용량(flush capacity)을 주기적으로 조정하는 파라미터이며, 디폴트는 yes이다.

MAX_FLUSH_PAGES_PER_SECOND: 버퍼 내 페이지를 디스크로 플러시할 때, 최대 플러시 용량(flush capacity)을 설정하기 위한 파라미터이며, 디폴트 값은 10000(pages)이다.

SYNC_ON_NFLUSH: 데이터 페이지 및 로그 페이지를 플러시한 후, fsync를 수행하는 주기를 설정하는 파라미터이며, 디폴트 값은 200(pages)이다.

- 관련 문서: 매뉴얼> 성능 튜닝> 데이터베이스 서버 설정> 로깅 관련 파라미터

CUBRIDSUS-1961 시리얼 성능 개선을 위한 SERIAL CACHE 기능 추가

SERIAL 객체를 정의할 때 지정된 개수만큼 시리얼을 캐시할 수 있는 CACHE 옵션을 지원한다.

```
CREATE SERIAL order_no
START WITH 10000 INCREMENT BY 2 MAXVALUE 20000
CACHE 100;
```

- 관련 문서: 매뉴얼> SQL설명서> 시리얼(SERIAL) > CREATE SERIAL

CUBRIDSUS-1897 CCI에서 HA 기능 지원

CCI에서 HA 기능을 지원하기 위해 HA 연결 설정을 위한 cci_connect_with_url ( ) 함수를 제공한다.

```
cci_connect_with_url
("cci:cubrid:host:port:dbname:user:password:?althost=host1:port:host2:port&rctime=100", "dba",
"")
```

- 참고 사항: Active 노드 장애 발생 시 노드 장애 확인을 위해 일정 시간 응답을 대기하므로, standby 브로커와 최초 연결을 맺는 시점이 지연될 수 있다.
- 관련 문서: 매뉴얼> API 레퍼런스> CCI API> cci_connect_with_url

CUBRIDSUS-1988 CCI에서 SELECT 문에 대한 자동 커밋 지원

CCI에서 SELECT 질의에 대해 자동 커밋 기능을 지원하며, 이 기능을 위한 브로커 파라미터를 제공한다.

```
#cubrid_broker.conf
...
SELECT_AUTO_COMMIT = ON
...
```

- 참고 사항: 질의 요청 핸들 n개에 대해 모두 결과 셋을 fetch한 경우에만 자동 커밋된다.
- 관련 문서: 매뉴얼> API 레퍼런스> CCI API> cci_end_tran

CUBRIDSUS-1330 DB별로 HA 모드를 off할 수 있는 기능 추가

HA 환경 구성 시, [common] 섹션에 HA 모드를 설정하고, HA 모드로 구동하지 않을 데이터베이스에 대해 [@<database>] 섹션에서 파라미터(HA_MODE)를 비활성화할 수 있도록 수정하였다.

```
#cubrid.conf
[common]
...
HA_MODE = on

[@testdb1]
...
HA_MODE = off
```

- 관련 문서: 매뉴얼> 관리자 안내서> CUBRID HA

개선된 사항

CUBRIDSUS-1218 멀티호스트로 구성된 환경에서 유틸리티 수행 시 반드시 접속할 호스트를 명시하도록 변경

데이터베이스 위치 정보 파일(databases.txt)에 동일 DB에 대한 호스트가 여러 개 지정된 경우, 유틸리티 수행 시 대상 데이터베이스 이름 뒤에 접속할 호스트를 반드시 명시하여야 하며, 이를 생략할 경우 에러 메시지가 출력되도록 수정하였다.

```
[user1@localhost~]$ backupdb testdb@localhost
```

- 관련 문서: 매뉴얼> 관리자 안내서> CUBRID HA

CUBRIDSUS-2232 PAGE_FLUSH_INTERVAL_IN_MSECS 파라미터 추가

데이터 버퍼에 존재하는 더티 페이지(dirty pages)를 백그라운드에서 플러시(flush)하는 간격을 설정할 수 있으며, 이를 위한 파라미터를 제공한다.

PAGE_FLUSH_INTERVAL_IN_MSECS: 백그라운드 플러시 간격을 밀리초 단위로 설정하는 파라미터이다.

- 참고 사항: 이전 버전에서 제공된 백그라운드 플러시 관련 파라미터(**page_flush_thread_wakeup_interval_in**

_secs)는 삭제되었다.

- 관련 문서: 매뉴얼> 성능 튜닝> 데이터베이스 서버 설정> 로깅 관련 파라미터

CUBRIDSUS-2124 INDEX_SCAN_OID_BUFFER_PAGES 파라미터 값의 범위 확대

인덱스 스캔 성능을 향상시키기 위하여 관련 파라미터(**index_scan_oid_buffer_pages**) 값의 타입 (INT FLOAT) 및 최소 값(1 0.05)을 변경하여 설정 가능한 값의 범위를 확대하였다.

index_scan_oid_buffer_pages: 인덱스 스캔 수행 시 OID리스트가 임시 저장되는 버퍼 크기를 설정하는 파라미터이다.

- 관련 문서: 매뉴얼> 성능 튜닝> 데이터베이스 서버 설정> 메모리 관련 파라미터

CUBRIDSUS-2167 cubrid createdb 유틸리티에 로그 페이지 크기를 지정하는 옵션 추가

데이터베이스 생성 시 로그 페이지 크기를 설정할 수 있는 옵션(--log-page-size)을 추가하여, 데이터 페이지 크기와 다르게 설정할 수 있도록 하였다.

```
% cubrid createdb -s 8192 --log-page-size 4096 testdb
```

- 관련 문서: 매뉴얼> 관리자 안내서> 데이터베이스 관리> 데이터베이스 생성

CUBRIDSUS-2018 cubrid statdump의 출력 항목 및 옵션 추가

cubrid statdump 유틸리티 또는 csql 세션 명령어인 ;hist 수행 시 출력되는 모니터링 항목에 쿼리 수행 관련 정보를 추가하였다. 쿼리 수행 관련 정보는 SELECT/UPDATE/INSERT/DELETE 문의 수행 횟수와 스캔 및 조인 수행 횟수를 포함한다. 또한, 데이터베이스 서버의 누적된 실행 통계 정보를 출력하도록 하는 --cumulative (또는 -c) 옵션을 추가 제공한다.

```
% cubrid statdump  -i 5  --cumulative testdb
```

- 관련 문서: 매뉴얼> 관리자 안내서> 데이터베이스 관리> 데이터베이스 서버 실행 통계 정보 출력

CUBRIDSUS-1955 ERROR_LOG_WARNING 파라미터 제공

에러 메시지 중 심각성(severity) 수준이 비교적 낮은 WARNING 메시지가 에러 로그 파일에 출력되지 않도록 설정할 수 있으며, 이를 위한 파라미터(**ERROR_LOG_WARNING**)를 제공한다.

```
#cubrid.conf
...
error_log_level = notification
ERROR_LOG_WARNING = no
```

```
...
```

- 참고 사항: 디폴트 값이 no이므로, warning 메시지를 확인하려면 ERROR_LOG_WARNING=YES로 수정한다.
- 관련 문서: 매뉴얼> 성능 튜닝> 데이터베이스 서버 설정> 오류 메시지 관련 파라미터

CUBRIDSUS-1964 인터럽트 발생 시점에 출력되는 에러 메시지 추가

트랜잭션 수행 중에 인터럽트가 설정되는 경우 인터럽트가 실제 설정된 시점과 인터럽트가 발견된 시점에 각각 타임스탬프 값과 함께 해당 트랜잭션 정보를 포함하는 에러 메시지가 출력되도록 수정하였다.

```
Time: 02/16/10 03:30:29.466 - ERROR *** ERROR CODE = -981, Tran = 8, CLIENT =
db001.ccs:cub_user (297), EID = 250

Set interrupt to the transaction(7).

Time: 02/16/10 03:30:31.466 - ERROR *** ERROR CODE = -4, Tran = 7, CLIENT = db001.ccs:cub_user
(297), EID = 250

Has been interrupted.
```

CUBRIDSUS-2077 HA 환경에서 슬레이브 DB로 반영된 연산 횟수가 누적 카운트되도록 수정

applylogdb 프로세스에 의해 복제 로그가 반영되는 상태를 확인할 수 있는 db_ha_apply_info 테이블의 *_counter 컬럼 값이 applylogdb 프로세스가 재 시작되더라도 초기화되지 않도록 수정하여, 누적된 카운터 값을 확인할 수 있다.

- 관련 문서: 매뉴얼> CUBRID HA> HA관련 시스템 카탈로그

CUBRIDSUS-1963 기본 키가 설정된 경우, 기본 키 인덱스가 최우선 선택되도록 쿼리 플랜 생성 방식 변경

기본 키가 정의된 컬럼들이 모두 WHERE 절 내 등식 조건식에 명시되는 경우, 기본 키 인덱스가 최우선 선택되도록 쿼리 플랜 생성 방식을 수정하였다. 이전 버전에서는 동일한 상황에서 다른 인덱스가 선택되는 쿼리 플랜이 생성될 수 있었다.

- 관련 문서: 매뉴얼> SQL설명서> 쿼리 실행 계획 보기

CUBRIDSUS-2154 cubrid diagdb 유틸리티에 힙(heap) 정보 출력 기능 추가

cubrid diagdb 유틸리티 수행 시 -d 옵션 값을 9로 지정하여 힙 파일 내 페이지 공간 정보를 확인할 수 있다.

```
% cubrid diagdb -d 9 testdb
```

- 관련 문서: 매뉴얼> 관리자 안내서> 데이터베이스 관리> 데이터베이스 내부 정보 출력

수정된 사항

CUBRIDSUS-1967 복제 환경에서 데이터 불일치 오류 수정

매우 긴 트랜잭션이 복제되는 도중 repl_agent 프로세스가 재 시작되는 경우, I/O 에러가 발생되고 마스터 DB와 슬레이브 DB간 데이터가 일치하지 않는 오류를 수정하였다.

CUBRIDSUS-2029 HA 환경에서 데이터 불일치 오류 수정

HA 환경에서 applylogdb 프로세스가 비정상적으로 종료되는 경우, 마스터 DB의 일부 데이터가 누락되어 슬레이브 DB로 반영되지 않는 오류를 수정하였다.

CUBRIDSUS-2707, 2748 오버플로우 레코드의 삭제 연산 오류 및 applylogdb 프로세스가 복제 도중 비정상 종료되는 현상 수정

데이터 페이지 크기를 초과하는 오버플로우 레코드가 입력되는 경우, 해당 레코드에 대한 페이지 링크 관계가 잘못 생성된 채 저장되어 이로 인해 삭제 연산 시 Object buffer underflow while reading 오류가 출력되는 현상이 있었다. 또한, HA 환경에서 마스터 DB에 이러한 오버플로우 레코드가 입력되는 경우, applylogdb 프로세스의 동작 오류로 인해 복제 도중 applylogdb 프로세스가 비정상 종료되는 문제가 있었으나 이를 수정하였다.

CUBRIDSUS-2207 HA 환경에서 applylogdb 프로세스가 비정상 종료되는 현상 수정

HA환경에서 보관 로그(archive log)가 삭제된 경우, applylogdb 프로세스의 페이지 탐색 알고리즘의 오류로 인해 페이지 탐색 도중 에러를 출력하며 프로세스가 비정상 종료되는 오류를 수정하였다.

```
"Unable to mount log disk volume/file "/home/cubrid/CUBRID/databases/db_1/testdb_lgar000"
```

- 참고 사항: HA환경에서는 **log_max_archives** 파라미터에 설정된 개수만큼 보관 로그가 저장되고, 설정 개수를 초과하는 경우 보관 로그가 자동 삭제된다. 이 파라미터 값을 너무 작게 설정하는 경우, 필요한 보관 로그가 삭제됨에 따라 applylogdb 프로세스가 요청 페이지를 검색하지 못하는 상황이 발생될 수 있으므로, 이 값을 여유 있게 설정하는 것을 권장한다. 또한, HA환경에서는 보관 로그를 수동으로 삭제하지 않아야 하고, DB를 백업할 때에도 -r옵션(불필요한 보관 로그 삭제) 또는 -sp옵션(안전하게 삭제)를 사용하지 않도록 주의하여야 한다.

CUBRIDSUS-2123 HA 환경에서 마스터 DB의 multiple update 연산이 슬레이브 DB로 일부만 반영되는 오류 수정

HA 환경에서 마스터 DB의 테이블에 대해 기본 키의 변형이 없는 multiple update를 수행하는 경우, 슬레이브 DB에서는 마지막 행을 제외한 나머지 행에서 해당 컬럼 값이 업데이트되지 않는 오류를 수정하였다.

CUBRIDSUS-2468 JDBC에서 cubrid_broker reset 명령 수행 시 Read Only 브로커의 연결이 초기화되지 않는 오류 수정

HA 환경에서 JDBC에서 자동 커밋 기능을 사용하는 경우, cubrid_broker reset <broker_name> 명령을 수행해도 해당 Read Only 브로커의 연결이 초기화되지 않아 마스터 DB에 계속 접속되어 있는 오류를 수정하였다.

CUBRIDSUS-1811,2023 cubrid server stop시 서버가 종료되지 않고 무한 대기하는 현상 수정

DB 서버 (cub_server) 프로세스의 동작 오류로 인해 cubrid server stop 또는 cubrid service stop 명령을 수행하여도 서버가 정상적으로 종료되지 않는 오류를 수정하였다.

CUBRIDSUS-1938 특정 환경에서 CAS가 무한 대기하는 현상 수정

리소스 부하가 높은 환경에서 새로운 접속 요청을 처리하기 위해 CAS 응용 서버를 구동시키는 경우, 이를 담당하는 브로커 프로세스의 이상 동작으로 인해 CAS 응용 서버가 사용자 CPU 리소스를 과잉 점유하면서 무한 대기하는 오류를 수정하였다.

CUBRIDSUS-1670 특정 질의 수행 중 CAS가 무한 대기하는 현상 수정

복수 개의 부질의(subquery)를 포함하고 여러 개의 테이블에 접근하는 SELECT 문이 prepare된 이후 CAS 응용 서버가 무한 대기하는 현상이 발생되었으며, 관련 오류를 수정하였다.

- 우회 방안: 질의 중첩도가 낮아지도록 해당 질의문 수정

CUBRIDSUS-2112 INSERT 수행 시 데이터가 저장되는 페이지 검색 알고리즘의 오류 수정

데이터 삽입이 가능한 빈 공간을 검색하는 알고리즘의 오류로 인해 데이터 INSERT 수행 시 무한 루프가 발생되고 DB 서버가 비정상 종료되는 오류를 수정하였다.

- 우회 방안: 아래의 방안 중 하나를 채택하여 수행한다.

기존 버전에서 cubrid compactdb를 수행하여 페이지 공간 정보 업데이트

기존 버전에서 cubrid unload/load를 수행하여 DB를 재 구축

해당 테이블 재구성

CUBRIDSUS-2099 JDBC 수행 도중 브로커와 연결이 해제되는 오류 수정

Windows 환경에서 JDBC 응용 프로그램 또는 CUBRID 매니저에서 질의를 수행하는 도중에 Can't communicate with broker 라는 메시지를 출력하면서 연결 소켓이 해제되는 오류를 수정하였다.

- 참고 사항: Windows 환경에서 DB서버와 브로커 서버 사이의 네트워크가 서로 다른 경우에만 재현된다.

CUBRIDSUS-2434 새로 추가된 DEFAULT 속성 컬럼에 인덱스 설정 시 이미 입력되었던 레코드의 인덱스 키 값이 DEFAULT 값이 되도록 수정

테이블의 기존 컬럼(col1)과 새로 추가된 DEFAULT 속성 컬럼(col2)에 대해 멀티 컬럼 인덱스를 생성하는 경우, 컬럼 추가 이전에 삽입된 레코드의 col2컬럼 키 값이 DEFAULT값이 되도록 수정하였다.

이전 버전에서는 같은 상황에서 컬럼 추가 이전에 삽입된 레코드의 col2 키 값이 NULL로 저장되었기 때문에 인덱스 페이지가 손상되고 이로 인해 해당 데이터에 대한 UPDATE/DELETE질의 수행 시 Unknown key {'N', NULL} referenced in B+tree index 라는 오류 메시지가 출력되는 현상이 발생하였다.

CUBRIDSUS-2443 계층적 질의문에서 SYS_CONNECT_BY_PATH ( ) 함수의 대상 컬럼 값이 NULL인 경우 비정상 종료되는 문제 해결

계층적 질의문에서 제공하는 함수인 sys_connect_by_path ( ) 함수의 인자로 지정된 컬럼 값에 NULL이 있는 경우 이전 버전에서는 비정상 종료되는 문제가 있었으나, CUBRID 2008 R2.2에서는 공백 문자(empty string)을 반환하도록 수정하였다.

```
SELECT id, mgrid, name, SYS_CONNECT_BY_PATH(name,'/') FROM employee
START WITH mgrid IS NULL CONNECT BY PRIOR id = mgrid ORDER BY id;
;xr

id        mgrid  name            hierarchy
=========================================
1         NULL  Kim                 '/Kim'
2         NULL  Moy                 '/Moy'
3            1  NULL                '/Kim/'
4            2  NULL                '/Moy/'
5            4  NULL                '/Moy//'
```

CUBRIDSUS-2382 JDBC의 query cancel에 의해 수행 대기 중인 다음 질의문이 취소되는 오류를 수정

JDBC의 SELECT 질의 수행 도중 인터럽트가 설정되는 경우, 해당 질의의 다음 질의가 취소되는 오류를 수정하였다.

- 참고 사항: 서버가 결과 집합을 생성한 이후 인터럽트 쓰레드를 종료시키기 직전에 새로운 인터럽트가 요청되고, 이때 CAS가 다음 질의 요청을 대기 중인 경우에만 발생된다.

CUBRIDSUS-2253 JDBC에서 SELECT 질의에 대해 비정상 결과 집합을 반환하는 오류 수정

JDBC의 SELECT 질의 수행 시, 447번 에러가 출력되고 해당 질의 결과가 0으로 반환되는 오류를 수정하였다.

- 참고 사항: 서버가 SELECT질의 결과 리스트 파일 페이지를 읽는 순간에 인터럽트가 요청된 경우에만 재현된다.

CUBRIDSUS-1122 플랜 캐시 기능 사용시 이전 수행된 질의에 의해 다음 질의 결과 값의 타입이 변경되는 오류 수정

플랜 캐시 기능이 사용되는 경우, 타입이 다른 결과값이 반환되어야 하는 질의문을 동일한 질의문으로 판단하여 이전에 수행된 질의문에 대한 결과 값이 다음 질의문에 대해 반환되는 문제가 있었으나 관련 오류를 수정하여 의도한 결과 값이 출력되도록 하였다.

CUBRIDSUS-2125 NCHAR 타입이 동작하지 않는 오류 수정

NCHAR 타입이 환경 변수로 지정된 문자 세트와 무관하게 동작하는 오류를 수정하였다.

CUBRIDSUS-2041 질의문 내에 “;”가 포함되는 경우 loaddb 실패 오류 수정

cubrid loaddb 대상 스키마 파일에 포함된 질의문 중간에 “;”가 포함된 경우, 이를 질의문의 끝(end)으로 잘못 판단하여 스키마 로딩에 실패하는 오류를 수정하였다.

- 우회 방안: 질의 편집기 또는 CSQL인터프리터에서 스키마 파일을 로딩한다.

CUBRIDSUS-2543 Windows에서 loaddb 대상 스키마에 특정 컬럼 값이 비어 있는 레코드가 포함된 경우 에러 출력되는 문제 수정

Windows에서 특정 컬럼 값이 비어 있는 레코드가 포함된 스키마에 대해 cubrid unloaddb 수행 시 오류 메시지가 출력되면서 작업이 진행되지 않는 오류를 수정하였다.

```
"Line 25745:Missing attribute values. Expected 4, found 1."
```

- 참고 사항: Linux에서는 에러가 출력되지 않고 정상 수행된다.

CUBRIDSUS-2190 서버 측 INSERT 모드에서 변수 바인딩 오류 수정

Server-side INSERT 기능을 활성화하고 JDBC응용에서 바인드 변수를 사용한 INSERT문을 수행하는 도중에 DB 서버가 비정상 종료되는 오류를 수정하였다.

CUBRIDSUS-2242 서버 재 시작 이후 레코드 개수와 COUNT(*) 값의 불일치 오류 수정

트랜잭션이 롤백되고 아직 커밋되지 않은 상태에서 서버 프로세스가 재 시작된 경우, SELECT질의에 대한 결과 레코드의 수와 SELECT COUNT(*)의 반환 값이 서로 다르게 출력되는 오류를 수정하였다.

- 참고 사항: 인덱스 설정된 컬럼에 대해 연산을 수행하는 일부 집계 함수(COUNT(*), COUNT(*col*), COUNT(DISTINCT *col*), MIN(*col*), MAX(*col*))에서 롤백 전 상태의 결과 값이 반환된다.

CUBRIDSUS-2490, 2542, 2584 JDBC에서 SET 타입에 대한 연산 수행 시 메모리 누수 현상이 발생되는 오류 수정

JDBC에서 SET 타입을 포함한 질의가 수행될 때 메모리 누수 현상이 발생되어 서버 프로세스가 점유하는 메모리가 급격히 증가하는 오류를 수정하였다.

- 참고 사항 1: JDBC(sync모드), CCI(sync모드가 디폴트), CSQL(async모드가 디폴트)
- 참고 사항 2: 결과 셋 전부를 한번에 반환하는 sync모드에 대해서만 오류가 수정되었으므로, CSQL에서 해당 질의 수행 시 주의하여야 한다.

CUBRIDSUS-1900, 2193 C-API 및 JDBC에서 타입 변환(coerce) 수행 시 에러 출력되는 문제 수정

C-API의 db_value_coerce ( ) 함수를 호출하여 정수 데이터를 CHAR 타입으로 변환하는 경우, Data overflow error 오류가 발생되는 문제를 수정하였다. 또한, JDBC에서 CHAR타입 컬럼과 바인드 변수를 비교하는 조건절을 포함하는 prepared SELECT 문에 데이터를 바인딩하는 경우, cannot coerce host var to type char 오류가 발생되는 문제도 함께 수정하였다.

CUBRIDSUS-2513 계층적 질의가 부질의로 사용되는 SELECT문 수행 시, 비정상 종료되는 오류 수정

계층적 질의문이 다른 질의와 중첩되는 경우, 부모 행의 의사 컬럼(pseudo column) 초기값을 설정하지 못하는 오류로 인해 서버가 비정상 종료되는 오류를 수정하였다. 오류를 발생시킨 질의 패턴은 아래와 같다.

```
select aa.col1, aa.col2,
(select d.col3 from
   (select col4,substr(sys_connect_by_path(c.col3,','),2) from
        (select rownum rn, b.* from
    (select col5 from tmp_tbl a where col5 is not null) b
        )c
        start with c.rn=1
        connect by prior c.rn+1=c.rn
        order by col4 desc
```

```
    )d
    where rownum=1
) as disp_col3 from tmp_tbl aa;
```

CUBRIDSUS-1937 브로커 상태 정보 중 QPS와 LQS 값의 출력 오류 수정

cubrid broker status 명령어를 실행하여 출력되는 브로커 상태 정보 중 QPS(Query per Second)와 LQS(Long Query per Second) 항목이 음수 값으로 출력되는 오류를 수정하였다.

- 참고 사항: 복수 개의 브로커가 구동되는 환경에서 각각의 브로커마다 **MAX_NUM_APPL_SERVER** 파라미터 값이 다르게 설정된 경우에만 발생된다.

CUBRIDSUS-2148 get trigger 문이 동작하지 않는 오류 수정

GET TRIGGER 문이 수행되지 않는 오류를 수정하였다.

CUBRIDSUS-2285 Microsoft Visual C++ 2008 재배포 가능 패키지 SP1 버전 지원

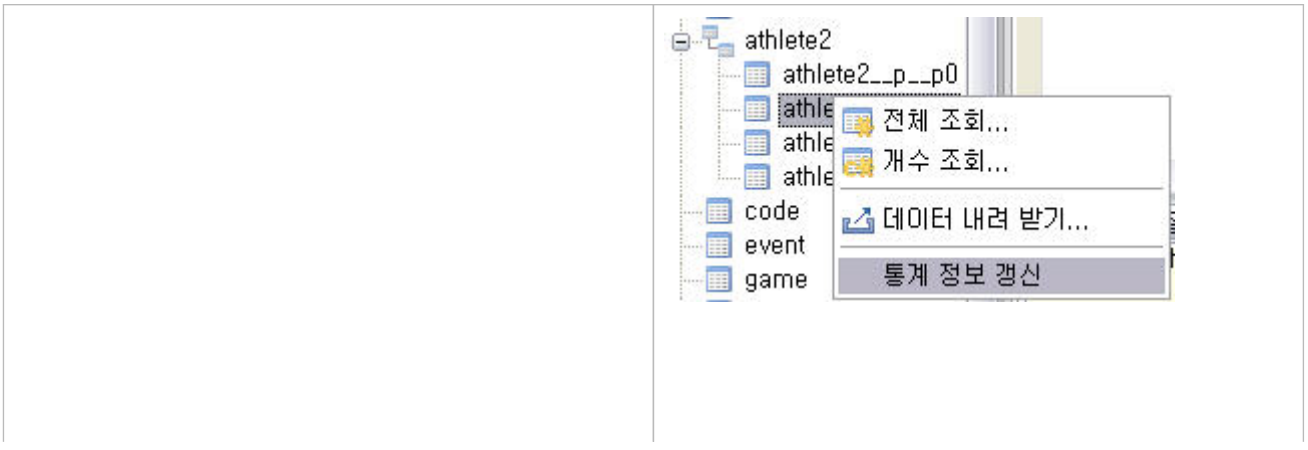
Microsoft Visual C++ 2008 재배포 가능 패키지의 SP1 버전을 추가로 지원하여 CUBRID 설치 마법사가 SP1 버전의 설치 여부를 인지할 수 있도록 수정하였다.

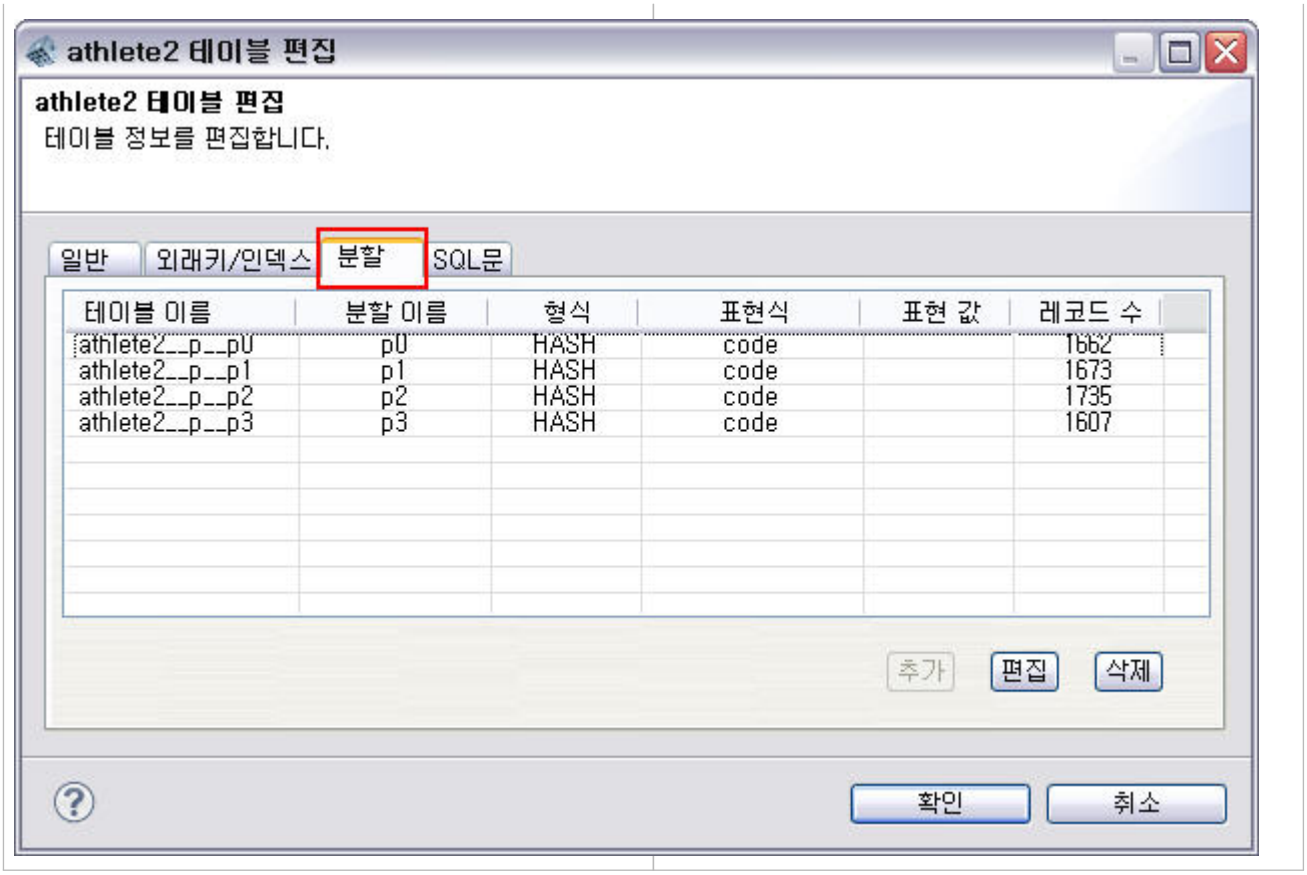
7.14 CUBRID 매니저 2008 R2.2에서 변경된 사항

새로 추가된 기능

테이블 추가/편집에 분할(Partition) 기능 추가 지원

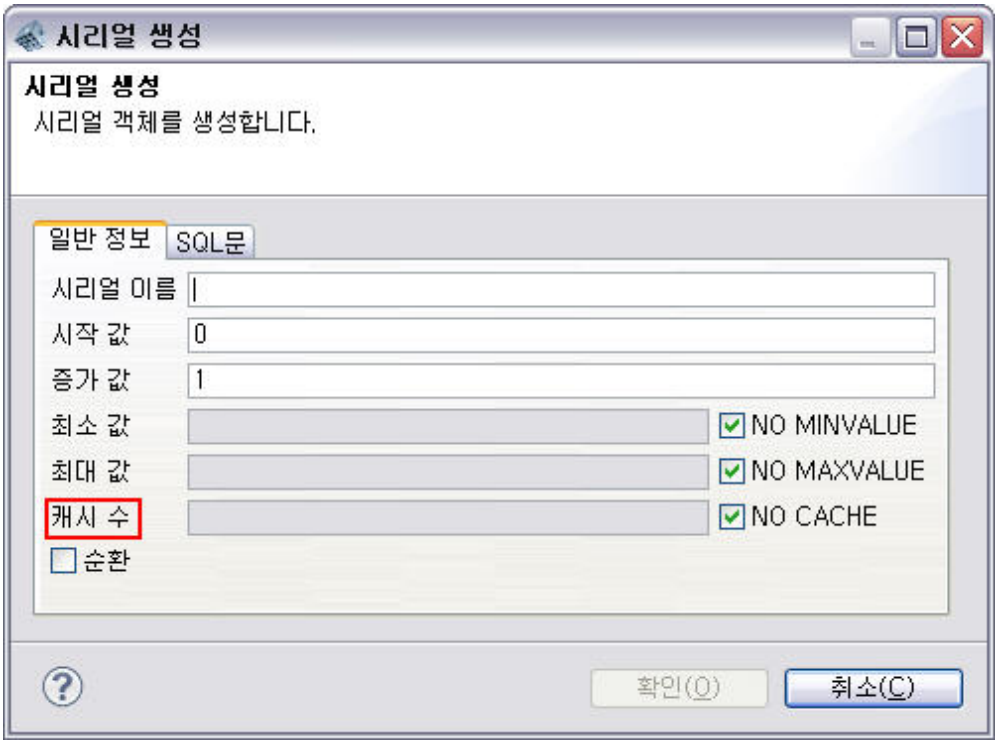
테이블 추가 및 편집 기능에서 분할 테이블의 추가, 편집, 삭제 기능을 지원하며, 분할 테이블에 대한 통계 정보 갱신은 테이블 팝업 메뉴를 통해 수행할 수 있다.





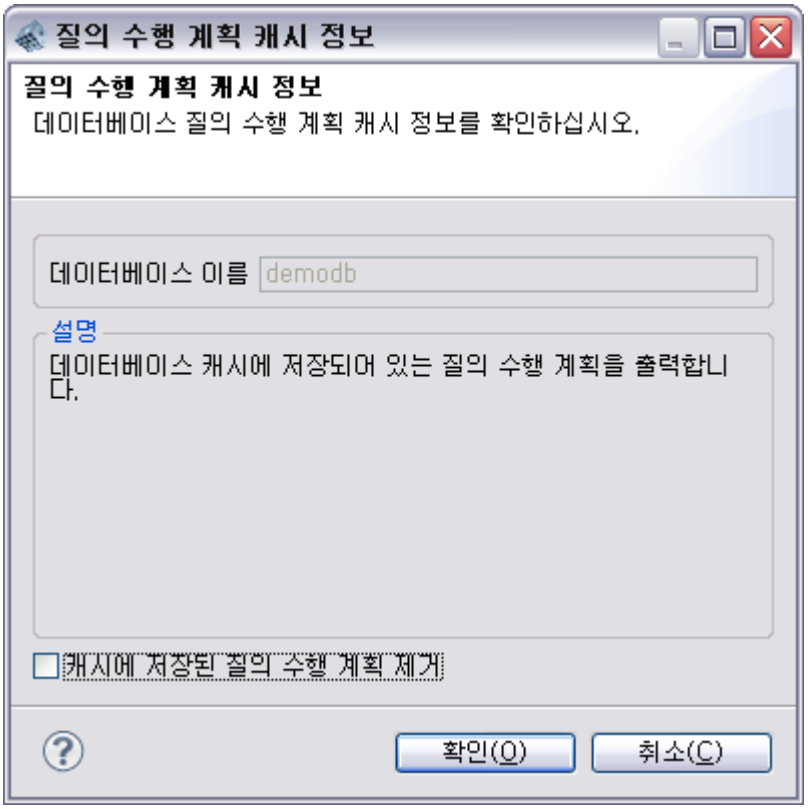
[시리얼 생성]에서 CACHE 기능 지원

시리얼 객체 정의 시 CACHE 옵션을 지정할 수 있다. 단, 이 옵션은 2008 R2.1 이하 서버에 대해서는 유효하지 않다.



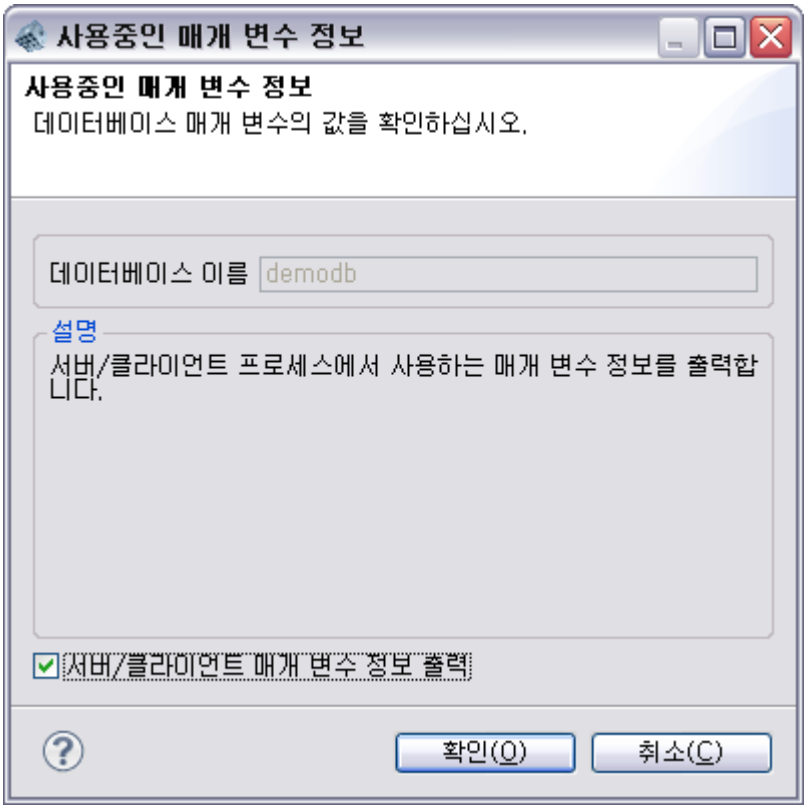
[질의 수행 계획 캐시 정보]에서 캐시 정보 확인 기능 지원

cubrid plandump 유틸리티를 매니저에서 지원하며, 서버에 저장(캐시)되어 있는 질의 수행 계획들의 정보를 출력할 수 있다. 질의 수행 계획 캐시 정보 기능은 데이터베이스 서버가 구동 중인 상태에서만 수행할 수 있다. [캐시에 저장된 질의 수행 계획 제거] 옵션을 선택하면, 질의 수행 계획 정보 출력 후 캐시를 초기화한다.



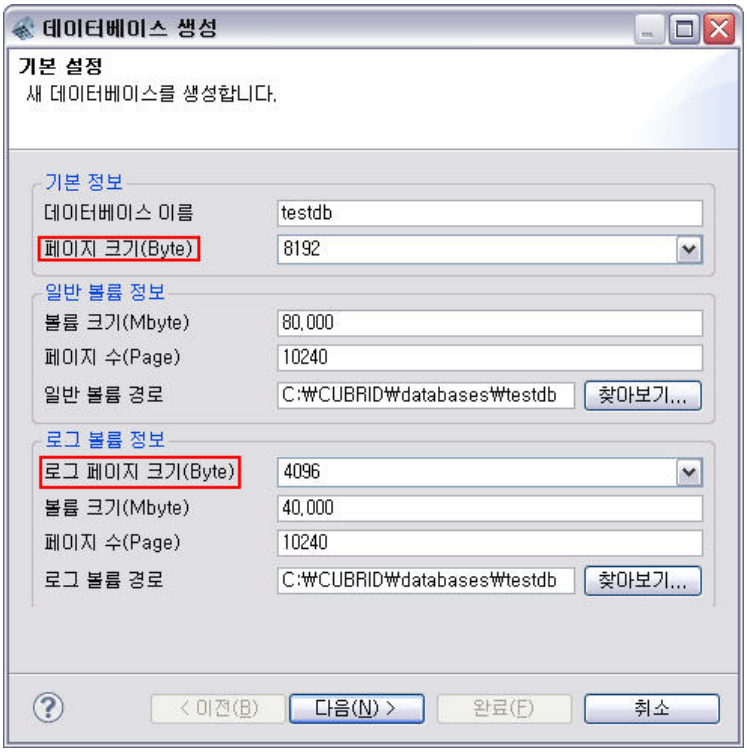
[사용 중인 매개 변수 정보] 보기 기능 지원

cubrid paramdump 유틸리티를 매니저에서 지원하며, 서버와 클라이언트에서 사용하는 매개 변수 정보를 출력한다. [서버/클라이언트 매개 변수 정보 출력] 옵션을 선택하면 사용 중인 서버/클라이언트 매개 변수 정보를 모두 출력하고, 옵션을 선택하지 않으면 서버 매개 변수 정보만 출력된다.



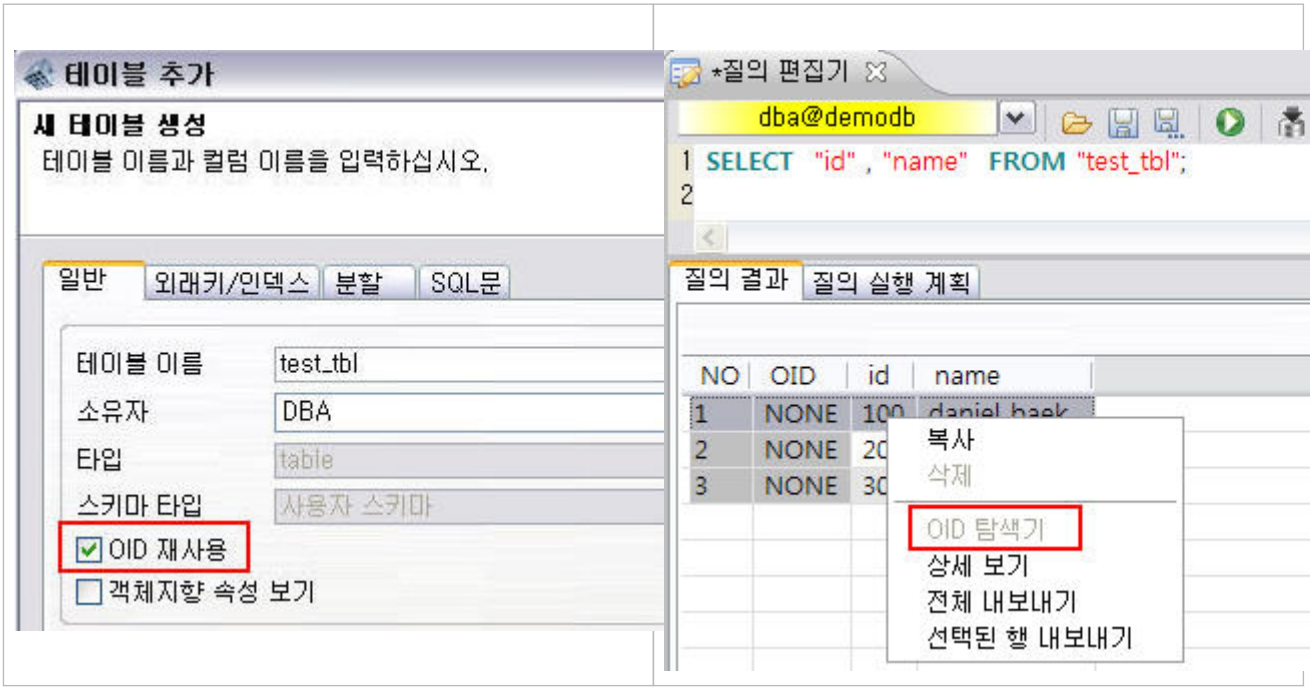
[데이터베이스 생성]에서 로그 페이지 크기 지정 기능 지원

cubrid createdb 유틸리티 수행 시 로그 페이지 크기를 지정할 수 있는 옵션(--log-page-size)이 지원됨에 따라, 이를 매니저의 [데이터베이스 생성] 마법사에서도 지정할 수 있다.



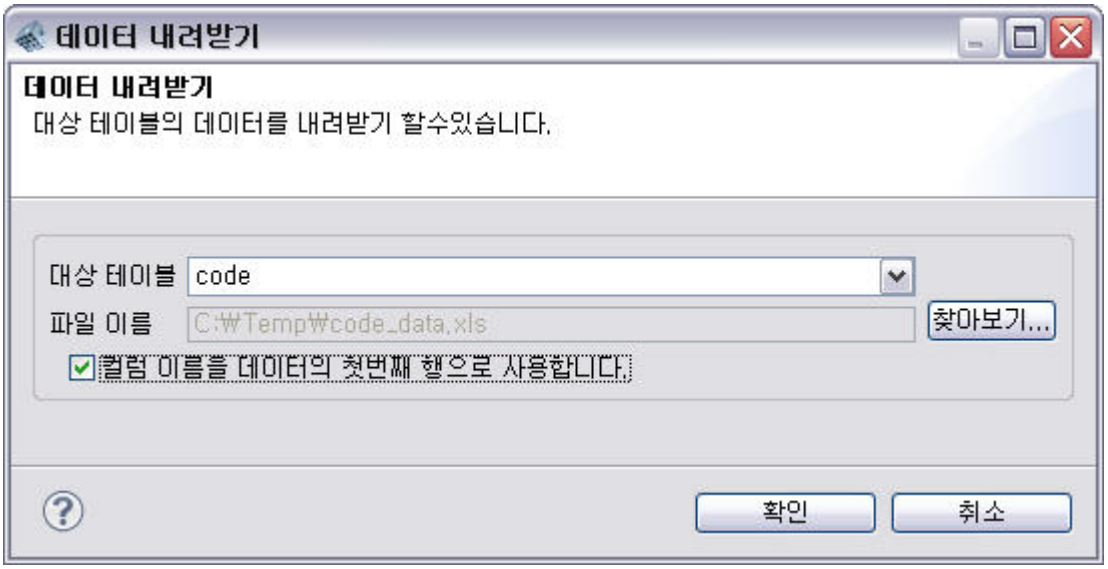
OID을 재사용하는 테이블 생성을 위한 REUSE_OID옵션 지원

테이블 생성 시 OID 재사용을 위한 테이블 옵션(REUSE_OID)이 지원됨에 따라, 매니저의 [테이블 추가] 메뉴에서 [OID 재사용] 옵션을 지정할 수 있다. OID 재사용 가능한 테이블에서는 OID값을 반환하지 않으므로, [OID 정보 보기] 옵션을 활성화하더라도 질의 결과 창에서 OID 탐색기가 동작하지 않으며, 질의 결과 데이터를 직접 수정할 수 없다.



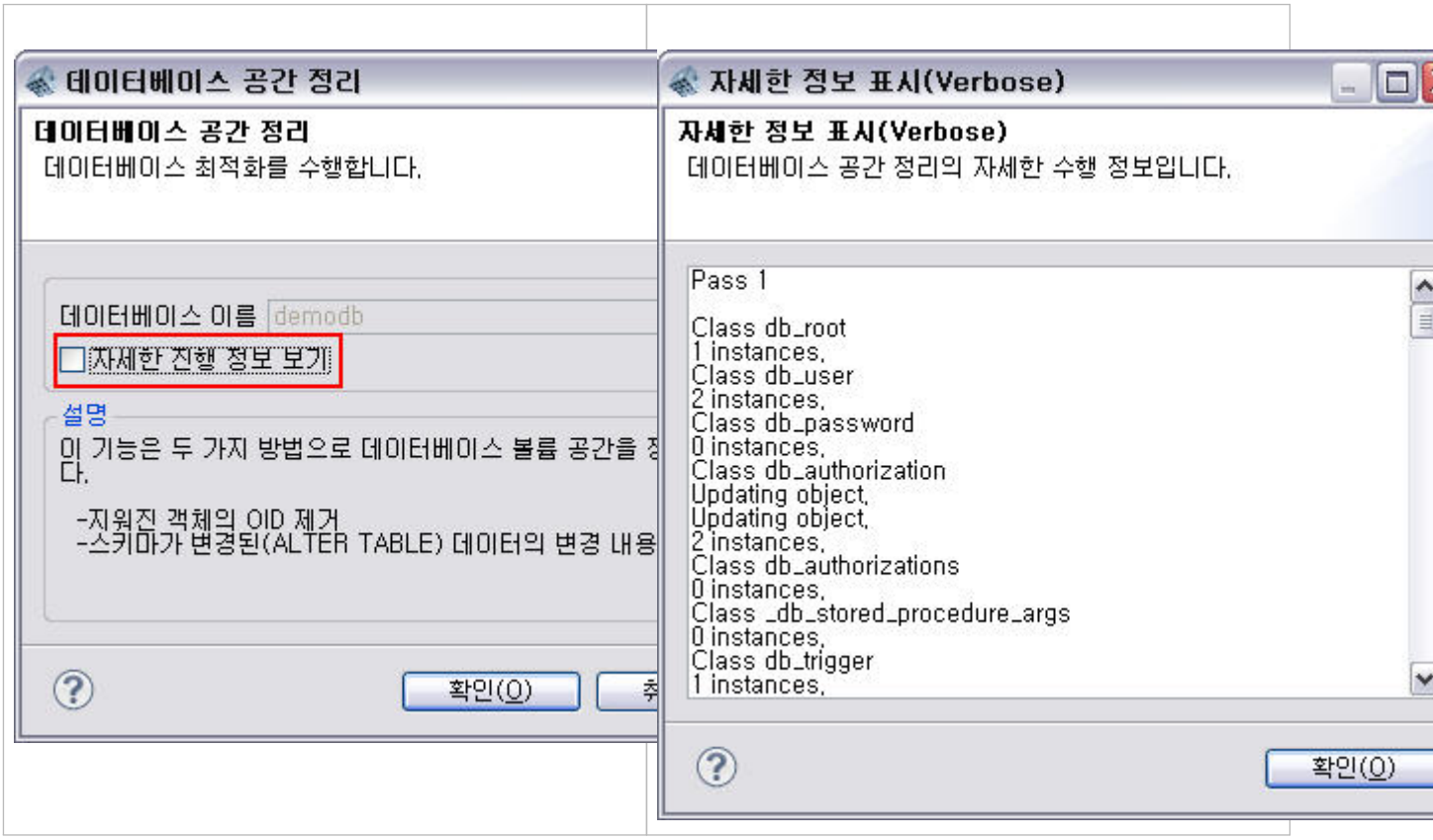
[데이터 내려받기] 수행 시 컬럼 이름 저장 기능 추가

[데이터 내려받기] 수행 시 테이블 내 존재하는 컬럼 이름을 대상 파일의 첫 번째 행으로 저장할 수 있는 기능을 지원한다.



[데이터베이스 공간 정리]에서 진행 정보 보기 기능 지원

cubrid compactdb 유틸리티를 수행하는 [데이터베이스 공간 정리]에서 [자세한 진행 정보 보기] 옵션을 지원한다.



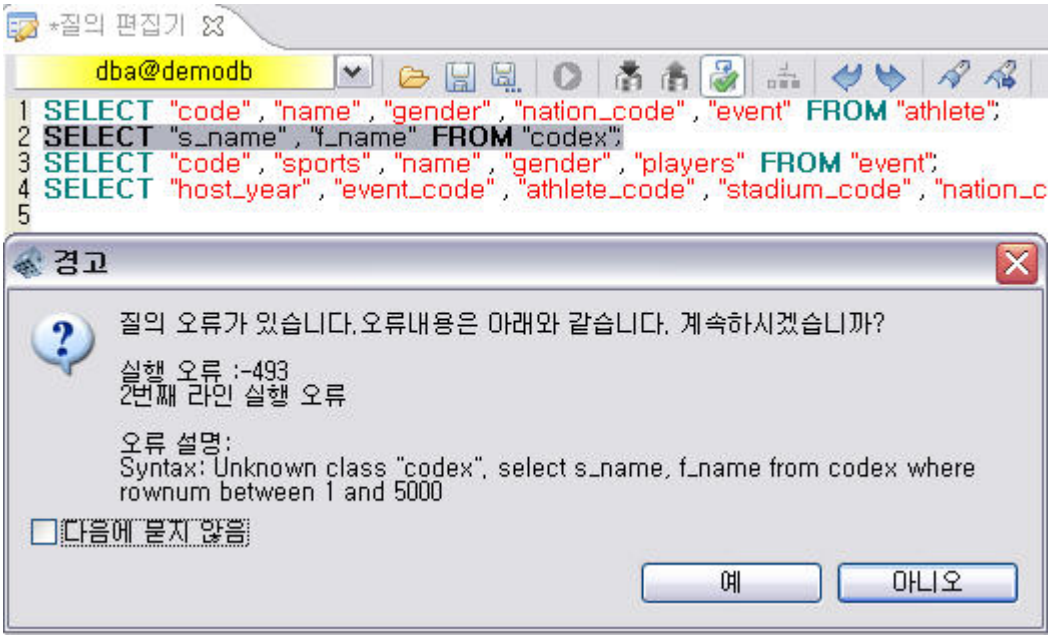
질의 자동화 수행 시 [질의 검사] 기능 지원

[질의 자동화]에서 추가할 질의를 검사하는 [질의 검사] 기능을 지원한다. 이를 통해 예약 수행될 질의에 문제가 있는지를 사전에 파악해 볼 수 있다.

개선된 사항

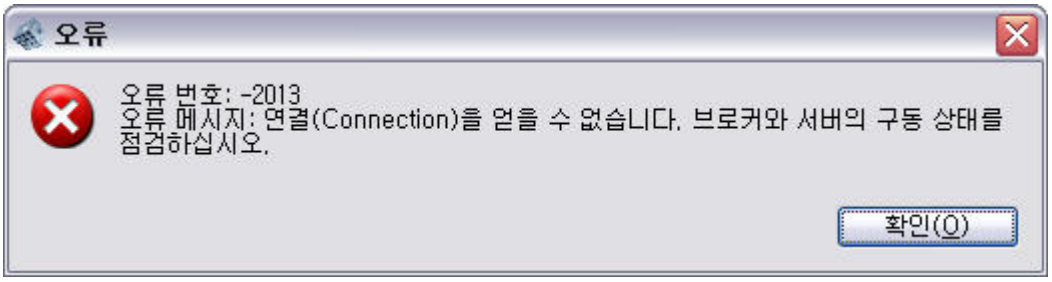
다중 질의 수행 시 중간에 오류가 발생해도 다음 질의를 수행할 수 있도록 개선

질의 편집기에서 다중 질의를 수행할 경우 중도에 오류가 발생해도 해당 질의를 제외하고 나머지 질의를 수행할 수 있는 기능이 아래 그림과 같이 지원된다. 질의 오류가 발견되는 경우 나머지 질의를 계속 수행할지, 중단할 것인지를 선택할 수 있다.



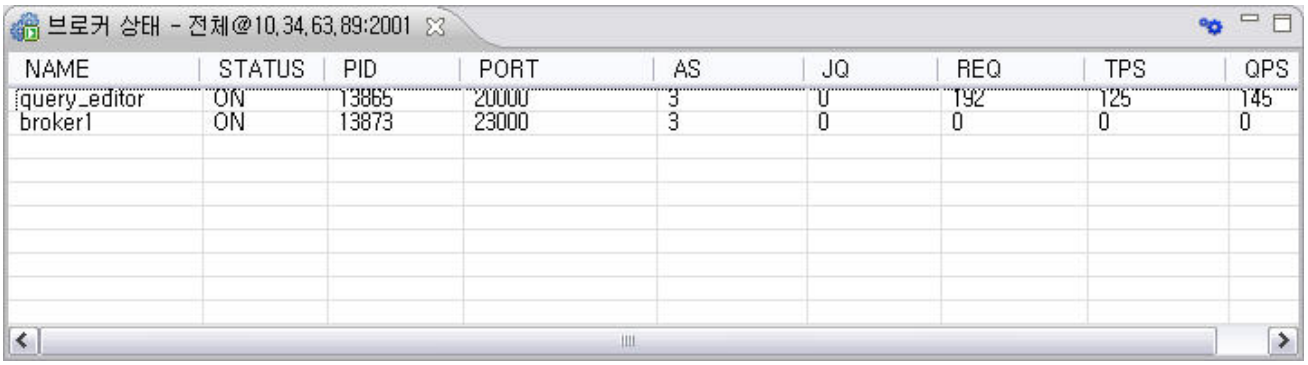
브로커와 연결 실패 시 JDBC에서 출력되는 오류 메시지 개선

브로커와 연결이 실패 또는 해제되는 경우, 이전 버전에서 “Version mismatch”라는 불명확한 메시지가 출력되었으나 아래와 같이 브로커에서 전달하는 오류 번호 및 메시지가 출력되도록 수정하였다.



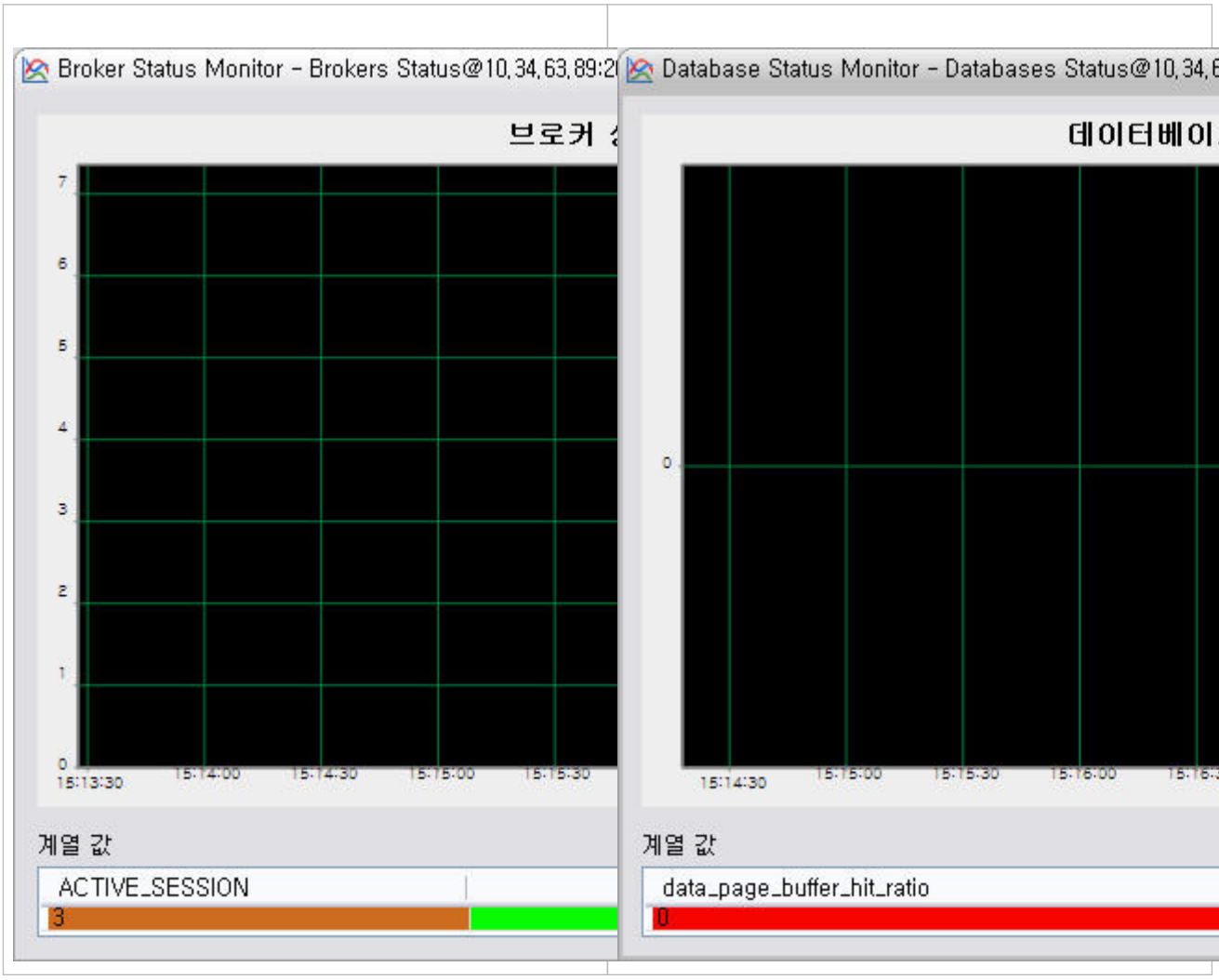
[브로커 상태] 정보 창에서 모니터링 항목 선택 기능 추가

[브로커 상태] 창에서 모니터링 항목을 사용자가 일부 항목만 선택할 수 있는 기능을 추가하였고, 이는 상단 바의 우측에 있는 설정 아이콘()을 선택하여 수행할 수 있다.



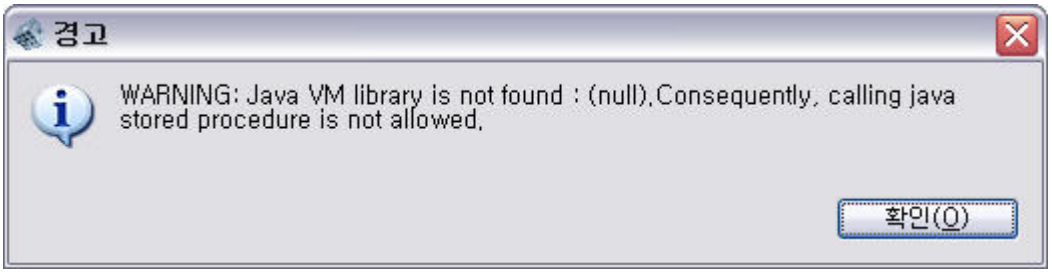
브로커 및 서버 상태의 모니터링 기능 개선

브로커 상태와 데이터베이스 상태를 하나의 차트를 통해 통합 모니터링할 수 있도록 지원한다. 상태 정보 창의 상단 바 우측에 있는 설정 아이콘( )을 통해 모니터링 항목 및 차트 계열을 설정할 수 있다.



java_stored_procedure 매개 변수 값이 YES이면서 Java 환경 설정이 되어 있지 않을 경우, 경고 메시지가 출력되도록 수정

java_stored_procedure 파라미터 값이 YES이면서 Java 환경 설정이 유효하지 않은 경우, Java 저장 프로시저를 수행할 수 없다는 경고 메시지가 출력되도록 수정하였다.



데이터베이스 서버와 매니저 클라이언트의 연결 상태를 주기적 점검하는 기능 추가

데이터베이스 서버와의 연결 상태를 주기적으로 점검하도록 하여, 장시간 해당 소켓을 사용하지 않는 경우에도 중간 방화벽에 의해 강제적으로 연결이 해제되지 않도록 개선하였다.

툴 바의 [새 질의 편집기]에서 포트 값 입력 시 유효 범위를 점검하도록 개선

툴 바의 [새 질의 편집기]를 통해 질의 편집기를 사용할 때 브로커 포트 값의 유효 범위를 점검하도록 수정하였다.

[테이블 추가/편집]에서 DEFAULT 속성 정의 시 타입 적합성을 점검하도록 개선

[테이블 추가], [테이블 편집]에서 컬럼에 DEFAULT 값 입력하는 경우 CUBRID가 지원하는 모든 데이터 타입에 대해 적합성을 점검하도록 수정하였다.

[데이터 내려받기] 수행 시 상태 창이 중복 출력되는 문제 개선

매니저의 진행 상태 창과 결과 메시지 창이 동시 출력되는 경우, 진행 상태 창이 하단의 상태 바 위치에서 출력되도록 수정하여 두 개의 창이 중복되어 출력되는 문제를 개선하였다.

[데이터베이스 삭제]시 해당 디렉터리에 파일이 존재하지 않으면 디렉터리까지 삭제할 수 있도록 개선

매니저에서 데이터베이스 생성 후 삭제 시 해당 디렉터리에 파일이 존재하지 않으면 디렉터리까지 삭제할 수 있도록 수정하였다.

데이터베이스 사용자의 비밀번호 변경 정책 변경

DBA 사용자에 의해서만 데이터베이스 사용자의 비밀번호 변경이 가능했던 정책을 변경하여, 데이터베이스 사용자가 자신의 비밀번호를 변경할 수 있도록 수정하였다.

수정된 사항

[테이블 추가/편집]에서 DEFAULT 값 출력 시 NULL과 공백 문자열('')이 구별되도록 수정

[테이블 추가], [테이블 편집] 창에서 디폴트 값 출력 시 NULL과 빈 문자열('')이 구분되지 않는 오류를 수정하여, 빈 문자열('')의 경우 따옴표가 출력되도록 하였다.

FLOAT 타입의 데이터에 대해 매니저와 CSQL의 출력 결과가 상이한 오류 수정

매니저의 질의 결과 창에서 출력되는 FLOAT 타입 데이터의 포맷이 CSQL에서 출력되는 데이터 포맷과 동일하도록 수정하였다.

질의 편집기에서 "//"와 같은 주석 처리 오류 수정

질의문 내 문자열에 "//"가 포함되는 경우, 이를 주석으로 간주하여 "--"로 변환하는 오류를 수정하여 매니저에서 해당 문자열이 정상 출력되도록 하였다.

다국어 테이블이 생성되지 않는 오류 수정

cubrid.conf의 intl_mbs_support 파라미터가 YES인 경우에도 매니저에서 다국어(multi-byte) 테이블이 생성되지 않는 오류를 수정하여, 해당 데이터베이스 [속성 > 연결 정보 > 문자 집합]에 있는 문자 집합으로 테이블을 생성할 수 있다. 즉, 해당 데이터베이스의 연결 정보에서 문자 집합을 UTF-8로 설정하면 UTF-8 문자로 테이블을 생성할 수 있다.

클라이언트와 서버의 프로토콜이 단절되는 오류 수정

클라이언트와 서버간의 통신 시 아무런 입력 정보 없이 태스크를 호출하면 프로토콜이 단절되는 오류를 수정하였다.

[기본설정> JDBC 드라이버]에서 기본 설정된 JDBC 드라이버를 제거해도 정보가 남아 있는 오류 수정

[기본설정> JDBC 드라이버]에서 기본 설정된 JDBC 드라이버를 제거한 이후에도 해당 드라이버 정보가 제거되지 않는 오류를 수정하였다.

엑셀 파일로부터 데이터를 DB로 로딩할 때, 특정 문자("GLO")가 있을 경우 NULL로 입력되는 오류 수정

엑셀 파일 내 데이터에 특정 문자("GLO")가 있을 경우, 해당 데이터를 GLO 객체로 잘못 분석하여 DB에 NULL로 저장되는 문제를 수정하였다.

편집 기능으로 시스템 트리거를 열 때 발생하는 오류 수정

매니저에서 편집 기능으로 시스템 트리거를 여는 경우, 수정 사항이 없음에도 확인 버튼이 활성화되는 오류 및 권한 실패 오류를 수정하였다.

[테이블 편집>SQL문 보기]에서 사용자가 지정한 기본 키(PK) 이름이 출력되지 않는 오류 수정

스크립트에 의한 테이블 생성 시 사용자가 기본 키(PK) 이름을 지정한 경우, 매니저의 [테이블 편집 > SQL문]에서 시스템에서 자동 부여한 이름이 출력되는 오류를 수정하여, 사용자가 지정한 기본 키(PK) 이름이 출력되도록 하였다.

데이터베이스 생성 경로에 공백이 포함된 경우 마법사가 진행되지 않도록 수정

데이터베이스가 생성되는 디렉터리 경로에 공백이 포함된 경우, [데이터베이스 생성] 마법사가 진행되지 않도록 수정하였다. 또한, C:₩, D:₩와 같은 루트 디렉터리가 데이터베이스 생성 경로로 지정되는 경우 오류 메시지가 출력되는 문제를 수정하였다.

데이터베이스 생성 마법사에서 페이지 크기 동기화 오류 수정

데이터베이스 생성 마법사에서 사용자가 전/후 단계를 이동하면서 데이터베이스 페이지 크기 값을 반복 설정하는 경우, 설정 값이 동기화되지 않는 오류를 수정하였다.

[데이터베이스 공간 정보 보기] 수행 후 데이터가 모두 로딩되기 이전에 창을 닫으면 발생하는 오류 수정

[데이터베이스 공간 정보 보기]를 수행한 후, 데이터가 모두 로딩되기 이전에 close하면 SWTException 오류가 발생되었으나, 이를 수정하였다.

[OID 정보 보기] 옵션을 선택한 후, 질의 결과 창에서 직접 데이터 수정 시 한글 깨짐 오류 수정

질의 결과 창에서의 엔터 키 값 처리 오류로 인해, 이전 버전에서는 결과 창에서 컬럼 데이터 중 한글을 수정할 때 깨짐 현상이 있었으나, 이를 수정하였다.

질의 편집기의 중간 분할 바의 위치가 사용자가 설정한 상태로 유지되도록 수정

이전 버전에서는 질의 편집기의 중간 분할 바가 고정 비율(3:7)로 유지되었으나, 이를 개선하여 사용자가 설정한 위치에 중간 분할 바가 유지되도록 하였다.

테이블 탐색기에서 분할 테이블 정보가 잘못 출력되는 오류 수정

분할 테이블의 분할 조건에 **MAXVALUE**가 포함되어 있을 경우, 테이블 탐색기의 노드 정보가 일반 테이블로 잘못 표시되는 오류가 있었으나, 이를 수정하였다.

매니저에 동일한 사용자 계정으로 다중 로그인한 경우, 뒤에 로그인한 사용자가 정상 수행할 수 있도록 수정

동일한 매니저 사용자 계정으로 다중 로그인하여 질의를 수행하는 경우, 이전 버전에서는 먼저 로그인한 사용자에 의해 변경된 정보로 인해 뒤에 로그인한 사용자의 질의 수행에 영향을 끼치는 문제를 수정하였다.

질의 편집기의 [검색 창의 검색 단위 설정] 옵션이 해제되어도 BETWEEN이 추가되는 오류 수정

질의 편집기의 [검색 창의 검색 단위 설정] 옵션이 해제되었음에도 BETWEEN 절이 추가되는 오류를 수정하였다.

질의 편집기 옵션의 폰트 설정이 적용되지 않는 오류 수정

질의 편집기 옵션에서 설정한 폰트가 질의 결과 창에만 적용되고, 편집 창에는 적용되지 않는 오류를 수정하였다.

DISTINCT/UNIQUE 키워드가 적용되지 않는 오류 수정

WHERE 절이 없는 SELECT 문에 DISTINCT 또는 UNIQUE 키워드가 포함되는 경우, 이전 버전에서는 키워드가 적용되지 않은 결과가 출력되는 문제를 수정하였다.

cubrid createdb 유틸리티로 생성한 데이터베이스가 매니저에 출력되지 않는 오류 수정

콘솔에서 cubrid createdb 유틸리티를 수행하여 데이터베이스를 생성하는 경우, 매니저의 왼쪽 트리를 새로고침한 이후에도 해당 데이터베이스가 트리에 표시되지 않는 오류를 수정하였다.

동일 데이터베이스 이름으로 데이터베이스 생성시 발생하는 오류 메시지 수정

데이터베이스 생성시 데이터베이스 이름이 중복되는 경우 이전 버전에서는 “서버에 연결할 수 없다”는 모호한 오류 메시지가 출력되었으나, 이를 수정하였다.

브로커가 하나도 없을 경우 추가가 안 되는 현상 수정

브로커 리스트에 브로커 정보가 하나도 없으면 브로커를 추가할 수 없었던 오류를 수정하였다.

7.15 주의 사항

CUBRID 2008 R2.2 사용 시 주의 사항

Windows 64bit 환경에서 CUBRID 32bit 설치 시 주의 사항

Windows 64bit 환경에서 CUBRID 32bit 설치 도중 Microsoft Visual C++ 2008 32bit 재배포 가능 패키지가 설치되었음에도 불구하고 해당 패키지의 설치를 요청하는 알림 창이 잘못 팝업되는 오류가 존재한다. Microsoft Visual C++ 2008 32bit 재배포 가능 패키지가 이미 설치되어 있는 경우 알림 창의 잘못된 정보를 무시하고 “아니오“ 버튼을 선택하여 CUBRID 32bit의 설치를 완료할 수 있다.

2008 R2.2 Patch 9 이전 버전 사용자는 cubrid compactdb 사용 금지

2008 R2.2 Patch 9 이전 버전에서는 cubrid compactdb 수행(CUBRID 매니저에서는 “데이터베이스 공간 정리” 수행) 시 간혹 볼륨의 헤더 정보가 잘못 수정되어 데이터베이스 볼륨이 잘못될 수 있는 문제가 존재하므로 사용하지 말아야 한다.

2008 R2.2 Patch9 이전 버전 사용자는 cubrid unloaddb 유틸리티를 SA 모드에서 사용할 것을 권장함

2008 R2.2 Patch9 이전 버전 사용자는 CS 모드에서 cubrid unloaddb 유틸리티를 수행하는 경우 간혹 내부 버퍼 사용에 문제가 발생하여 원래의 레코드 개수보다 적은 개수가 언로드될 수 있으므로, SA 모드에서 수행할 것을 권장한다.

복제 또는 HA 환경에서 외래 키가 정의된 테이블에 INSERT 수행 시 주의 사항

복제 또는 HA 환경에서 외래 키 제약 조건이 UNIQUE 인덱스 컬럼에 정의되고 트랜잭션 1과 2에서 각각 기본 키 테이블과 외래 키 테이블에 대해 INSERT를 수행하는 경우, 마스터 DB와 슬레이브 DB간 데이터 불일치가 발생될 수 있으므로 주의한다.

이전 버전의 백업 파일을 이용하여 복구 시도하는 경우 주의 사항

이전 버전의 **cubrid backupdb** 유틸리티로부터 생성한 백업 파일을 이후 버전의 **cubrid restoredb** 유틸리티를 사용하여 데이터베이스 복구를 시도하는 경우 동일한 버전을 사용하여야 한다는 에러를 출력하며 종료한다.

cubrid backupdb 유틸리티의 -r 옵션 사용 시 주의 사항

-r 옵션을 사용하여 증분 백업(백업 수준 1 또는 2)을 수행하는 경우 추후 데이터베이스의 정상 복구가 불가능할 수 있으므로, 전체 백업 수행 시에만 -r 옵션을 사용한다.

Windows 환경에서 CUBRID 설치 시 주의 사항

Windows 환경에서 CUBRID 설치 시 디렉터리 경로에 공백을 포함하는 경우 설치가 되지 않으므로 주의한다.

Windows 환경에서 CUBRID 환경 변수 설정 시 주의 사항

Windows 환경에서 CUBRID 환경 변수를 역슬래시 문자(“₩”)로 끝나는 문자열로 지정 시에는 CUBRID 관련 프로세스 구동 도중 오류가 발생될 수 있으므로 디렉터리의 마지막이 “₩”로 끝나지 않도록 주의하여야 한다.

Windows 환경에서 *.conf 파일에 파라미터 추가 시 주의 사항

Windows 환경에서 환경 설정 파일(***.conf**)에 파라미터를 추가하는 경우 라인의 끝에서 enter 키를 입력하여야 한다. 그렇지 않으면 개행 문자(“₩n”)가 없는 것으로 판단하여 해당 라인의 파라미터가 적용되지 않는다.

Windows 환경에서 cubrid service stop 명령 수행 시 주의 사항

Windows 환경에서는 “SYSTEM” 권한을 가진 사용자인 경우에만 **cubrid service stop** 명령이 정상 수행되므로, 관리자 또는 일반 사용자는 CUBRID 매니저 설치 후 작업 표시줄에 생성되는 CUBRID 서비스 트레이를 통해 프로세스를 종료한다.

CUBRID 매니저에서 [볼륨 자동 추가 기능 사용] 옵션을 선택하고 DB를 생성하는 경우 주의 사항

[볼륨 자동 추가 기능 사용] 옵션은 DB 볼륨이 사용자가 설정한 “여유 공간 비율" 값 이하가 될 경우 해당 볼륨을 자동으로 추가하는 기능이며, 이 옵션이 선택되면 여유 공간을 모니터링하기 위해 주기적(디폴트 5초)으로 **cubrid spacedb** 유틸리티를 수행하므로 에러 로그(<dbname>_spacedb.err)가 증가할 수 있다. 사용자는 **cm.conf** 파일의 **monitor_interval** 파라미터를 설정하여 모니터링 주기를 조정할 수 있다.