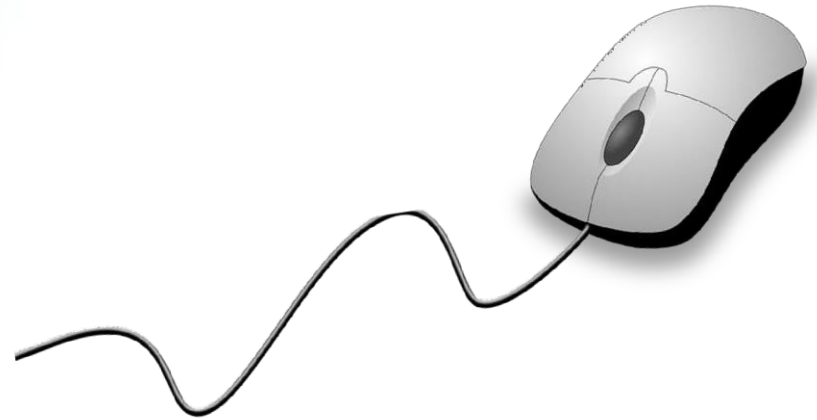


공개SW 솔루션 설치 & 활용 가이드

시스템SW > 데이터베이스



CUBRID™



제대로 배워보자

How to Use Open Source Software

Open Source Software Installation & Application Guide



오픈소스 소프트웨어 통합지원센터
Open Source Software Support Center



CONTENTS

1. 개요
2. 기능요약
3. 실행환경
4. 설치 및 실행
5. 기능소개
6. 활용예제
7. FAQ
8. 용어정리

1. 개요



<p>소개</p>	<ul style="list-style-type: none"> • CUBRID는 국산 공개SW 객체 관계형 데이터베이스 관리시스템 • CUBRID는 데이터베이스 서버, 브로커, CUBRID 매니저로 구성 • CUBRID는 인터넷 데이터 서비스에 최적화된 데이터베이스시스템이며, 다양한 기능 제공 		
<p>주요기능</p>	<ul style="list-style-type: none"> • SQL92 지원 • RDB modeling • Record locking • Online backup/recovery • HA (High Availability) • API : JDBC, PHP, ODBC, OLEDB, CCI 		
<p>대분류</p>	<ul style="list-style-type: none"> • 시스템SW 	<p>소분류</p>	<ul style="list-style-type: none"> • DBMS
<p>라이선스형태</p>	<ul style="list-style-type: none"> • BSD, GPL2 	<p>사전설치 솔루션</p>	<ul style="list-style-type: none"> • JDK 6.0 이상
<p>운영제제</p>	<ul style="list-style-type: none"> • LINUX5.0 이상, WINDOW 7 이상 	<p>버전</p>	<ul style="list-style-type: none"> • 10.1
<p>특징</p>	<ul style="list-style-type: none"> • community와 enterprise version 구분 없음 • 3-Tier Architecture (Application-Broker-DBMS) • 사용자/운영자를 위한 도구 CM, CMT 제공 • 전용 heartbeat network 없이 HA 구동 • HA만이 아닌 read 분산을 위한 replica 서버 구성 가능 		
<p>보안취약점</p>	<ul style="list-style-type: none"> • N/A 		
<p>개발회사/커뮤니티</p>	<ul style="list-style-type: none"> • 네이버 / 큐브리드 		
<p>공식 홈페이지</p>	<ul style="list-style-type: none"> • http://www.cubrid.com 		



2. 기능요약



기능 구분	CUBRID
SQL	SQL-92, SQL-99(ODB)
Data Type	Alpha-numeric, Large Object (CLOB, BLOB)
Modeling	RDB (table, column, RI)
API	JDBC, PHP, ODBC, OLEDB, C api, etc
Transaction	Record locking Online backup/recovery
Availability	HA (High Availability)

• 대용량 RDBMS

보편성, 확장성, 안정성

- DB/테이블: 개수 및 크기 무제한
- 64bit 지원

• 트랜잭션

- ACID 보장: commit, rollback, savepoint
- 다중 단위 잠금: 테이블, 레코드 단위

• 고가용성 기능

- HA (High Availability)
- 백업 및 복구
 - 온라인/오프라인 백업 지원
 - 전체백업, 증분백업, 시점 복구

• 다양한 응용 환경

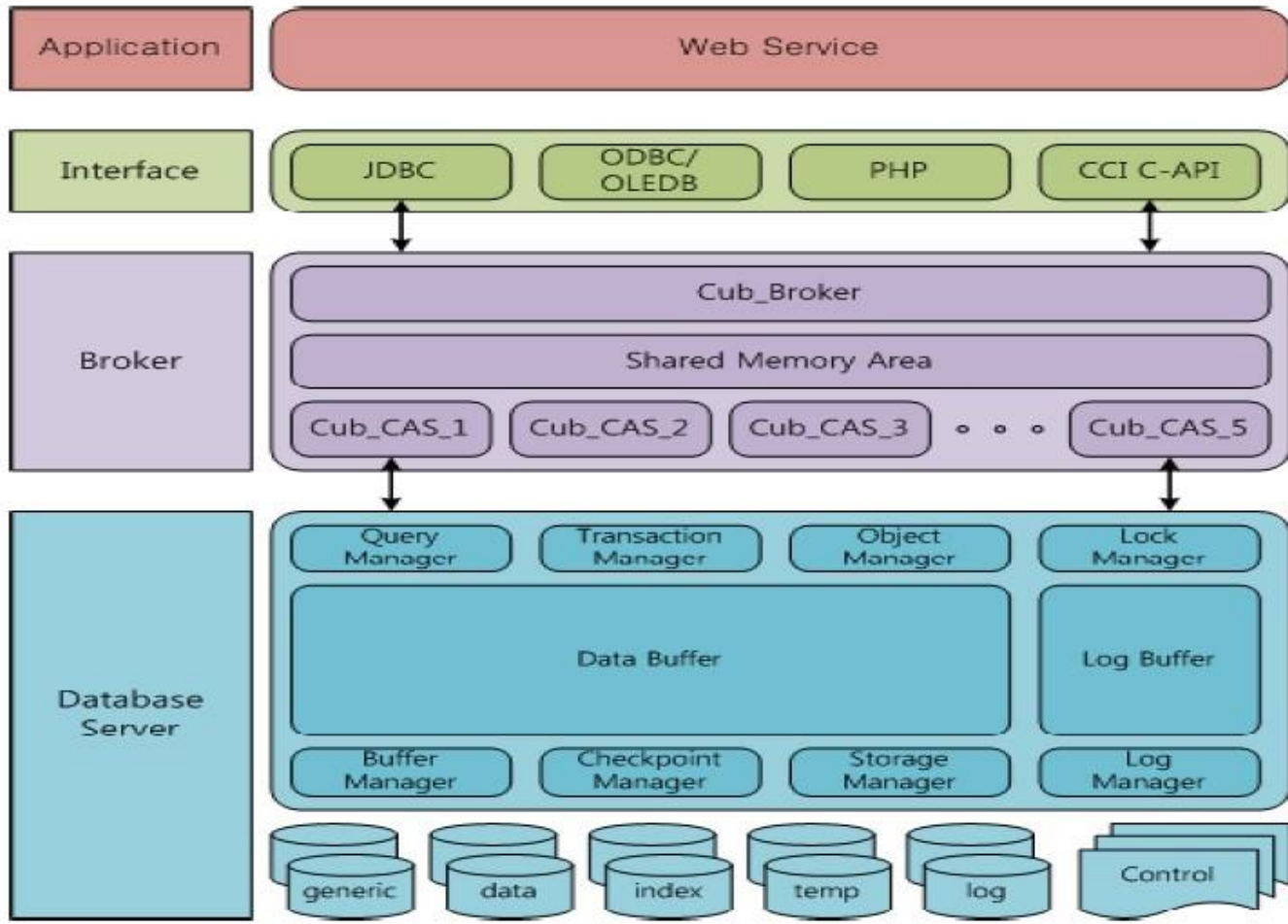
- JDBC, PHP, ODBC, OLEDB, Ruby, Python, C API

• CUBRID Manager

- 플랫폼에 독립적인 GUI 개발 및 운영 도구
- 통합 도구: 관리, 질의, 진단, 튜닝 등



2. 기능요약



[큐브리드 기본 아키텍처]





CUBRID 시스템 구조

- **Database Server**

- 데이터를 저장 및 관리하는 기능을 수행한다.
- Multi thread기반client/server 방식으로 동작한다.
- 사용자가 입력한 질의를 처리하고 DB 내의 객체를 관리한다.
- 잠금(LOCK)과 로깅(Logging) 기법을 이용해 다수사용자의 동시 트랜잭션을 지원한다.
- 운영에 필요한 백업 및 복구기능을 지원한다.

- **BROKER**

- CUBRID 전용미들웨어로 외부응용프로그램간의 통신을 중계하는 역할을 한다.
- 브로커는 커넥션풀링, 모니터링, 로그추적 및 분석기능을 제공한다.

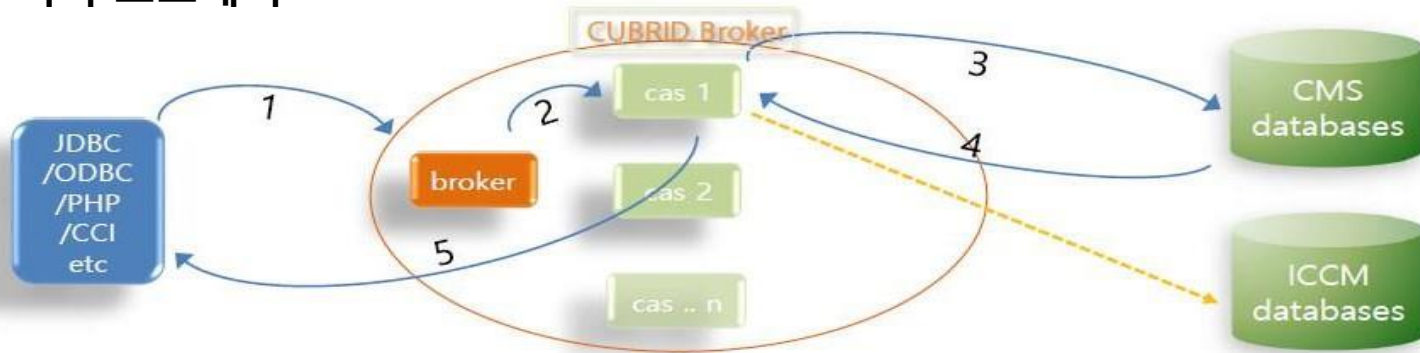
- **CUBRID Manager**

- GUI 환경에서Database Server와 Broker를 관리할 수 있는 관리자도구이다.
- CUBRID 설치시 기본으로 CUBRID Manager Server 모듈은 같이 설치된다.
- CUBRID Manager 관리자도구를 다운로드 받아 사용한다.





CUBRID 처리 프로세서



1. 응용(JAVA,PHP)에서 작업(질의)처리를 broker로 요청한다.
2. Broker는 관리하고 있는 CAS 프로세스에게 작업을 할당한다
 - 1) 질의의 작업처리는 CAS가 수행한다.(프로세스 단위로 작업을 수행함)
 - 2) 응용(JAVA,PHP)은 CAS에 직접 요청하는 것이 아니고, broker 에게 작업 요청해 CAS를 할당 받는다.
3. CAS는 데이터베이스에 연결하고 질의 처리를 요청한다.
 - 1) 응용(JAVA,PHP)의 컨넥션 URL 정보로 데이터베이스지정하며, 지정한 데이터베이스로 CAS가 연결한다.
 - 2) CAS는 데이터베이스 작업 후 연결 유지해 동일한 데이터베이스로 작업요청이 들어오면 기존 연결 재사용한다.
 - 3) CAS는 연결유지상태에서 다른 데이터베이스의 작업요청이 들어오면 현재연결을 끊고 새로이 연결한다.
 - 4) 데이터베이스 별로 broker를 할당해서 사용하는 것이 성능과 유지관리에 유리하다.
4. 데이터베이스는CAS에서 요청을 처리하고 결과를 해당 CAS에게 전달한다.
5. 데이터베이스의 작업결과(데이터)를 응용에 직접 전달한다.

※ CAS와트랜잭션

트랜잭션 처리중에는 하나의 CAS가 하나의 응용에 종속되며 트랜잭션 처리 종료 전까지는 다른 요청 받지 않는다.

- * CAS 프로세서 구동수 만큼에 동시트랜잭션이 처리된다.
- * 응용프로그램에서 select 도commit/rollback을 처리해야 한다.



3. 실행환경



- Linux 환경

- 공식지원 리눅스 버전

- 32bit: CentOS 4 이상, Fedora 4, Ubuntu 6.10 이상, openSUSE 11 이상, Gentoo 2007 이상, Asianux 2 이상, Debian 4 이상
- 64bit: CentOS 4 이상, Fedora 11, Ubuntu 9.04

- 호환 CPU: Intel x86, Intel EM64T, AMD64

- OS 버전 확인 : Linux Kernel 2.4과 glibc 2.3.4버전 이상만 지원한다.

- 확인방법 : `$> uname -a & rpm -q glibc`
- 결과 : Linux host_name **2.6.18-53.1.14.el5xen** #1 SMP Wed Mar 5 12:08:17 EST x86_64 x86_64 x86_64 GNU/Linux 7
Kernel 2.6

- 기본 라이브러리: Linux 버전에 상관없이 기본 라이브러리 확인이 필요하다.

- 확인방법 : `$> rpm -q ncurses & libgcrypt & libstdc++`

- 32/64bit 여부 : Linux bit를 확인하고 설치할 CUBRID 버전과 bit를 선택한다.

- 확인 방법 : `$> uname -a`
- 결과 : Linux host_name 2.6.18-53.1.14.el5xen #1 SMP Wed Mar 5 12:08:17 EST **x86_64 x86_64 x86_64** GNU/Linux 7
64bit OS

- 방화벽 설정

- Linux 서버 환경에서 CUBRID 사용하기 위해서 다음 포트는 기본적으로 오픈 한다.
- CUBRID Manager : 8001(CUBRID 9.3.2 이하 버전: 8001, 8002)
- 브로커: 30000, 33000 (질의편집기 : 30000, 응용개발 : 33000)



3. 실행환경



- Window 환경
 - 공식 지원 윈도우 버전
 - 32bit: Windows XP 32bit, Windows Vista 32bit, Windows 2003 server 32bit
 - 64bit: Windows Vista 64bit, (Windows 7)
 - 호환 CPU: Intel x86, Intel EM64T, AMD64
 - Windows 32bit & 64bit 지원
 - OS bit 확인 방법 : 내컴퓨터 ⑦ 속성 ⑦ 시스템에서 확인 가능.
 - 설치 유형 선택
 - 전체설치 : CUBRID서버와 명령행 도구 및 인터페이스 드라이버(JDBC, ODBC, OLEDB 등)가 설치된다.
 - 인터페이스 드라이버 설치 : 드라이버(JDBC, ODBC, OLEDB)만 설치된다.
 - 방화벽 설정
 - Windows 서버 환경에서 CUBRID를 사용하기 위해서 다음 포트는 기본적으로 오픈 한다.
 - CUBRID Manager : 8001(CUBRID 9.3.2 이하 버전: 8001, 8002)
 - 질의편집기 : 30000 ~ 30040
 - 응용개발 : 33000 ~ 33040



3. 실행환경



- Java 환경
 - CUBRID Manager client, CUBRID Query Browser, JAVA Stored Procedure 사용 시 JRE 1.6이상 버전이 설치되어 있어야 한다.



4. 설치 및 실행

세부 목차



1. 다운받기
2. Linux 설치
- 4.2 Window 설치



4. 설치 및 실행



4.1 다운받기

- WEB site
 - <http://cubrid.org>
 - <http://cubrid.com>
- FTP site
 - <http://ftp.cubrid.org>

로그인 비밀번호 CUBRID.ORG 🔍

PRODUCTS DOWNLOADS DOCUMENTS SERVICES & TRAINING FORUM PARTNERS NEWS COMPANY

DOWNLOADS

HOME / DOWNLOADS

CUBRID Engine

Platform	Download
CUBRID 10.1 (최신)	
Windows 32Bit	DOWNLOAD
Windows 64Bit	DOWNLOAD
Linux 64Bit	DOWNLOAD
CUBRID 9.3 (최신)	
Windows 32Bit	DOWNLOAD
Windows 64Bit	DOWNLOAD
Linux 32Bit	DOWNLOAD
Linux 64Bit	DOWNLOAD

Index of /

Name	Last modified	Size
CUBRID_Docs/	14-Jul-2013 18:16	-
CUBRID_Drivers/	14-Jul-2017 15:34	-
CUBRID_Engine/	04-Aug-2017 17:24	-
CUBRID_Repos/	18-Aug-2017 17:33	-
CUBRID_Tools/	17-Jan-2015 12:22	-
CUBRID_VMLimages/	20-Mar-2013 21:59	-
QA/	09-Apr-2014 17:33	-
cubrid_repo_settings/	17-Aug-2017 11:24	-
sites/	13-Jan-2014 16:18	-
README.txt	16-Apr-2011 17:53	2.0K

Apache/2.2.15 (CentOS) Server at ftp.cubrid.org Port 80

4. 설치 및 실행



2. Linux 설치

- **Linux 버전**

- CUBRID 설치 및 관리할 사용계정을 생성해서 제품설치를 권장한다.
- CUBRID DBMS Server와 Client(CUBRID 명령어, Broker)는 동일 버전에서만 완전한 호환을 지원한다.

```
root#> useradd cubrid

root#> su - cubrid

cubrid$> sh CUBRID-9.3.7.0008-linux.x86\_64.sh
Copyright (C) -2014 Search Solution Corporation. All rights reserved.
...
Do you agree to the above license terms? (yes or no) : yes
Do you want to install this software(CUBRID) to the default(/home/cubrid/CUBRID) directory? (yes or no) [Default: yes] : yes
Install CUBRID to '/home/cubrid/CUBRID' ...
Since CUBRID broker and server versions should match, please make sure that you are running the same version if you operate them in separate machines. For installation of CUBRID tools like Query Browser, Manager and Web Manager, please refer to http://www.cubrid.org/wiki\_tools.
Do you want to continue? (yes or no) [Default: yes] : yes

If you want to use CUBRID, run the following commands
% ./home/cubrid/.cubrid.sh
% cubrid service start

cubrid$> cubrid_rel
CUBRID 9.3 (9.3.7.0008) (64bit release build for linux_gnu)
```



4. 설치 및 실행



3. Window 설치

• Windows 버전

- CUBRID 설치는 administrator 계정을 권장한다.
- Default로 C:\CUBRID 위치에 CUBRID 제품이 설치된다.
- CUBRID Manager Client 사용을 위해 JRE 1.6 이상 버전을 설치한다.
- 설치유형 선택 시 전체설치 선택한다.
- 설치과정

1단계: 설치 디렉터리 지정

2단계: 설치 유형 선택

- 전체 설치 : CUBRID 서버와 명령행 도구 및 인터페이스 드라이버(JDBC, C API)가 모두 설치된다.
- 인터페이스 드라이버 설치 : 인터페이스 드라이버(JDBC, C API)만 설치된다. CUBRID 데이터베이스 서버가 설치된 컴퓨터에 원격 접근하여 개발하는 경우, 이 설치 유형을 선택할 수 있다.

3단계: 샘플 데이터베이스 생성

- ⑦ 샘플 데이터베이스를 생성하려면 약 300MB의 디스크 공간이 필요하다.

4단계: 설치 완료

- ⑦ 우측 하단 []모양에 CUBRID Service Tray가 나타난다.



5. 기능소개

세부 목차



1. CUBRID 서비스 구동/종료
2. CUBRID Server 구동/종료
3. CUBRID Broker 구동/종료
4. CUBRID Manager Sever 구동/종료
5. 데이터베이스 생성
6. CSQL 인터프리터
7. 백업과 복구



5. 기능소개

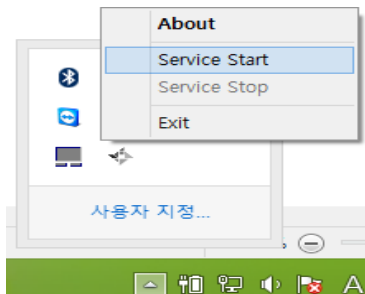


1. CUBRID 서비스 구동/종료(1/2)

- 서비스 구동
 - CUBRID 운영에 필요한 기본 프로세스 구동
 - CUBRID 사용자 계정으로 로그인 필요
 - master, broker, manager server 구동
 - database server 는 별도 구동, 또는 설정을 통하여 서비스 구동 시 같이 구동 가능
- 명령어

```
$ cubrid service start
@ cubrid master start
++ cubrid master start: success
@ cubrid broker start
++ cubrid broker start: success
@ cubrid manager server start
++ cubrid manager server start: success
```

- Windows
 - service 에 등록되어 부팅 시 구동(default)



5. 기능소개

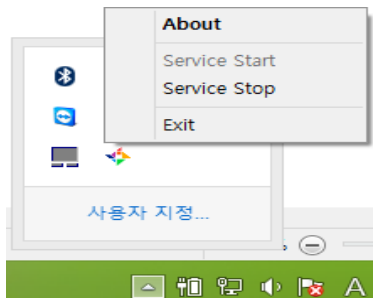


1. CUBRID 서비스 구동/종료(2/2)

- 서비스 종료
 - CUBRID 관련 모든 프로세스 종료
 - CUBRID 사용자 계정으로 로그인 필요
 - master, broker, manager server 및 database server 종료
- 명령어

```
$ cubrid service stop  
@ cubrid broker stop  
++ cubrid broker stop: success  
@ cubrid manager server stop  
++ cubrid manager server stop: success  
@ cubrid master stop  
++ cubrid master stop: success
```

- Windows
 - Exit 를 선택하면, 서비스 종료 후 tray 응용까지 종료됨.



5. 기능소개



2. CUBRID 서버 구동/종료(1/2)

- 데이터베이스 구동
 - 사용하는 데이터베이스 별 서버 구동
- 명령어
 - CUBRID 사용자 계정으로 로그인

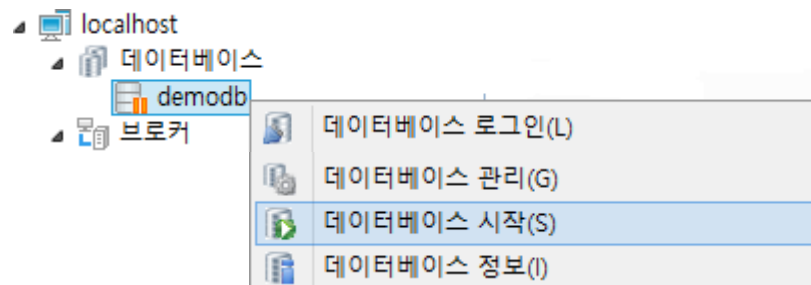
```
$ cubrid server start demodb  
@ cubrid server start: demodb
```

This may take a long time depending on the amount of recovery works to do.

CUBRID R9.2

```
++ cubrid server start: success
```

- Windows
 - CUBRID Manager Client에서 database dba 계정으로 로그인해야만 구동이 가능.



5. 기능소개



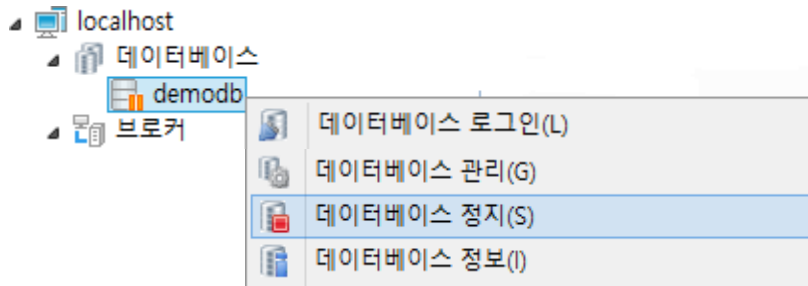
2. CUBRID 서버 구동/종료(2/2)

- 데이터베이스 종료
 - 사용하는 데이터베이스 별 서버 종료
- 명령어
 - CUBRID 사용자 계정으로 로그인

```
$ cubrid server stop demodb
@ cubrid server stop: demodb

Server demodb notified of shutdown.
This may take several minutes. Please wait.
++ cubrid server stop: success
```

- Windows
 - CUBRID Manager에서 database dba 계정으로 로그인



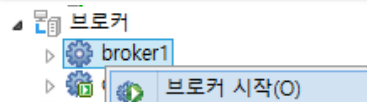
5. 기능소개



3. CUBRID Broker 구동/종료

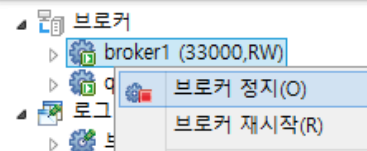
- 구동
 - CUBRID가 설치되어 있는 호스트의 브로커 구동.
 - CUBRID service 구동 시 자동으로 구동된다.

```
$ cubrid broker start
@ cubrid broker start
++ cubrid broker start: success
⑦ 이미 구동되어 있을 경우 아래와 같이 출력
++ cubrid broker is already running.
```



- 종료
 - CUBRID가 설치되어 있는 호스트의 브로커 종료.
 - CUBRID service 종료 시 자동으로 종료.

```
$ cubrid broker stop
@ cubrid broker stop
++ cubrid broker stop: success
⑦ 이미 종료되어 있을 경우
++ cubrid broker is not running.
```



5. 기능소개



4. CUBRID Manager Server 구동/종료

- 구동
 - CUBRID가 설치되어 있는 호스트의 Manager Server 구동.
 - CUBRID service 구동 시 자동으로 구동된다.

```
$ cubrid manager start
@ cubrid manager server start
++ cubrid manager server start: success
⑦ 이미 구동되어 있을 경우 아래와 같이 출력
++ cubrid manager server is already running.
```

- 종료
 - CUBRID가 설치되어 있는 호스트의 Manager Server 종료.
 - CUBRID service 종료 시 자동으로 종료.

```
$ cubrid manager stop
@ cubrid manager server stop
++ cubrid manager server stop: success
⑦ 이미 종료되어 있을 경우
++ cubrid manager server is not running.
```



5. 기능소개



5. DATABASE 생성(1/6)

- CUBRID engine을 설치한 후 사용할 데이터베이스를 생성한다.
 - CUBRID 설치 시 선택적으로 demodb를 생성할 수 있다.
 - demodb는 Sample Data를 가지고 있는 데이터베이스로 CUBRID 테스트 용도로 사용이 가능하다.
 - 프로젝트에 사용할 데이터베이스를 생성해 서비스할 것을 권장한다.

- 데이터베이스 생성
 - 데이터 보존 연한간 전체 데이터의 크기를 예측해 생성한다.
 - 데이터베이스의 생성된 크기는 줄일 수는 없으므로 주의해야 한다.

- 기본 정보
 - 페이지 크기 : 16384 bytes(디스크 I/O 단위이며, 성능상 무난한 크기임)
 - 볼륨 : 스키마나 데이터정보 등이 저장되는 공간
 - 32bit 사용시 한 개 볼륨의 최대 크기 : 2G
 - 범용 볼륨 : 스키마 및 메타정보 저장.
 - 로그 볼륨 : 데이터 변경에 관련된 정보 저장. 백업 복구 시 사용.
 - 데이터, 인덱스 볼륨 : 각 데이터, 인덱스 저장.
 - 템프 볼륨 : 질의 처리 및 정렬(sorting)을 수행할 때 중간, 최종 결과를 임시로 저장하는 공간

- 디스크 구성Tip
 - 일반적으로 디스크 RAID 구성은 1+0 또는 0+1이 안정성과 I/O 성능이 가장 좋다.
 - RAID 1+0, 0+1 구성이 불가능한 환경이라면 RAID-5 구성을 권장한다.



5. 기능소개



5. DATABASE 생성(2/6)

- 명령어 :createdb
 - 특수문자로 시작하는 데이터베이스 명은 사용할 수 없다.
 - 데이터베이스 명은 최대 17자까지만 사용할 수 있다.
 - DB 생성 명령어

\$ cubrid createdb [options] database_name locale_name.charset

- cubrid: CUBRID 서비스 및 데이터베이스 관리를 위한 통합 유틸리티이다.
- createdb: 새로운 데이터베이스를 생성하기 위한 명령어이다.
- database_name: 데이터베이스가 생성될 디렉터리 경로명을 포함하지 않고, 생성하고자 하는 데이터베이스의 이름을 고유하게 부여한다. 이 때, 지정한 데이터베이스 이름이 이미 존재하는 데이터베이스 이름과 중복되는 경우, CUBRID는 기존 파일을 보호하기 위하여 데이터베이스 생성을 더 이상 진행하지 않는다.
- locale_name: 데이터베이스에서 사용할 로캘 이름을 입력한다.
 - 큐브리드는 영어, 한국어, 터키어는 기본 로캘 라이브러리를 제공하며 이외 언어도 해당 로캘을 포함하는 로캘 라이브러리를 컴파일해서 사용할 수 있다.
 - 중국어(zh_CN), 독일어(de_DE), 프랑스(fr_FR), 스페인어(es_ES), 일본어(ja_JP)등과 언어를 사용할 경우 UTF-8로 설정하는 것을 권장한다.
- charset: 데이터베이스에서 사용할 문자셋을 입력한다. CUBRID에서 사용 가능한 문자셋은 iso88591, euckr, utf8이다.
 - locale_name이 en_US이고 charset을 생략하면 문자셋은 iso88591이 된다.
 - locale_name이 ko_KR이고 charset을 생략하면 문자셋은 utf8이 된다.
 - 나머지 locale_name은 charset을 생략할 수 없으며, utf8만 지정 가능하다.



5. 기능소개



5. DATABASE 생성(3/6)

- 명령어 : createdb [options]

옵션	옵션 전체 이름	설 명	초기값
-F	--file-path	초기 볼륨이 위치할 경로 지정	현재
-L	--log-path	로그 볼륨이 위치할 경로 지정	현재
-B	--lob-base-path	LOB파일이 저장될 위치 경로 지정	<File-path>/lob
-r	--replace	DB가 이미 존재하는 경우 기존 데이터베이스를 삭제하고 재생성함.	존재 시 에러 발생
	--db-volume-size	생성되는 데이터베이스 볼륨의 크기를 바이트 단위로 지정한다.	512M
	--db-page-size	데이터베이스 페이지의 크기를 바이트 단위로 지정한다.	16K
	--log-volume-size	로그 볼륨의 크기를 지정한다.	db_volume_size와 동일
	--log-page-size	로그 볼륨의 페이지 크기를 바이트 단위로 지정한다	db_page_size와 동일

- 사용 예제

```
cubrid createdb --db-volume-size=512M -F /data --log-volume-size=200M -L /log edudb ko_KR.utf8
```

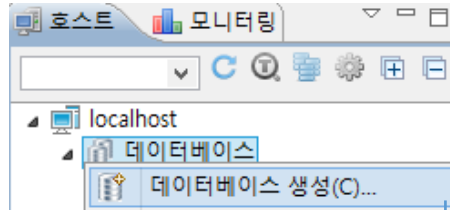


5. 기능소개

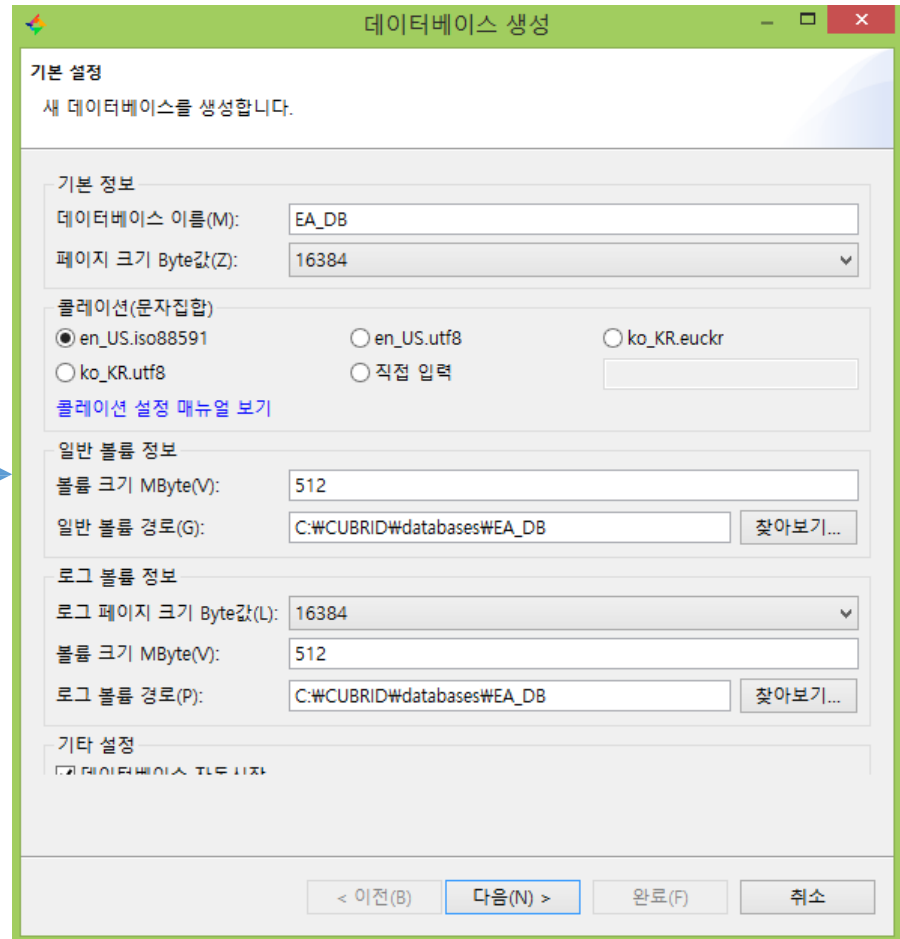


5. DATABASE 생성(4/6)

- CUBRID Manager를 이용한 DB생성



- 기본정보
 - 생성할 데이터베이스 이름
 - Page크기 (default 16K)
- 콜레이션(문자집합)
 - 데이터 문자 셋 선택
- 일반 볼륨 정보
 - 초기 생성 볼륨의 크기
 - Page 수(생성 볼륨 크기 환산 값)
 - 초기 볼륨 생성 위치
- 로그 볼륨 정보
 - page 크기(default는 DB page값과 동일)
 - 로그 볼륨 크기
 - 로그 볼륨 page 수
 - 로그 볼륨 생성 위치



5. 기능소개



5. DATABASE 생성(5/6)

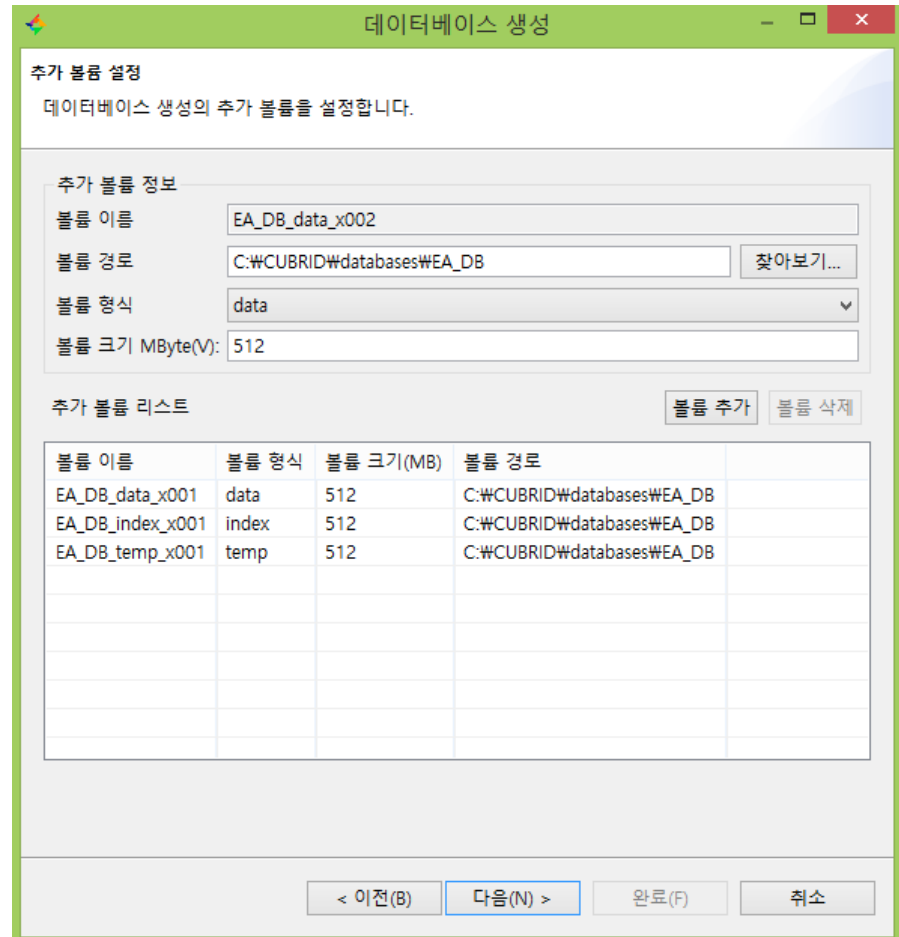
- 볼륨 추가
 - 용도별 볼륨을 분산하는 것이 성능상 효율적이다.
 - 운영상 적합한 크기를 예측해 사용 용도별 볼륨 확장 필요하다.

1. 추가 볼륨 정보

1. 추가할 볼륨의 이름을 기록
2. 생성될 볼륨 위치
3. 볼륨의 용도
4. 볼륨 크기
5. 볼륨 Page 수(볼륨 크기의 page환산 값)

2. 추가 볼륨 리스트

- Default로 추가되도록 설정된 볼륨으로 삭제하고 다시 추가하는 것이 가능. 리스트를 클릭 후 볼륨 삭제 버튼을 이용하여 제거 가능하다.



5. 기능소개



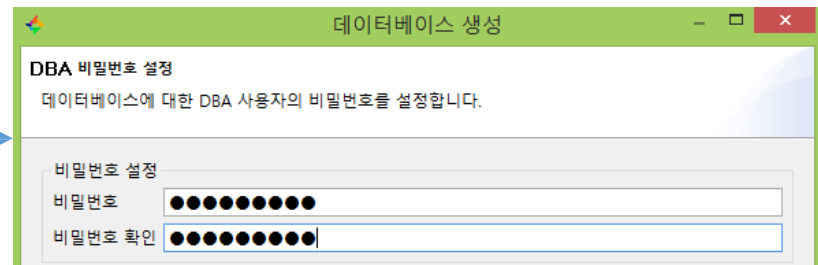
5. DATABASE 생성(6/6)

- 볼륨 자동 증가 설정(CUBRID Manager에서만 지원)
 - 각각의 볼륨의 여유 공간 부족 시 볼륨 자동 생성한다.
 - 확장 볼륨 기준(여유공간 비율) 및 자동 확장 볼륨 크기 지정한다.

1. 볼륨 형식

- 데이터, 인덱스 볼륨에 대하여 여유 공간 비율 설정 및 생성될 볼륨 크기를 기록한다. 확장 페이지 수는 볼륨 크기에 따라 자동 변환 된다.
- 데이터와 인덱스 볼륨에 대해서만 자동 볼륨 추가 설정이 가능하다.

- DBA 비밀번호 설정
 - DBA의 비밀번호를 설정한다.
 - Console작업 시 dba의 default 비밀번호는 없다.
- 데이터베이스설치 완료
 - 지금까지 설정한 내용들을 보여주고 DB설치 종료.



5. 기능소개



6. CSQL 인터프리터(1/2)

- 명령어 : csql

\$ csql [options] database_name

- CSQL 인터프리터는 CUBRID와 함께 설치되며, 대화형(interactive) 방식과 일괄 수행(batch) 방식으로 SQL 질의를 수행하고 수행 결과를 조회할 수 있는 프로그램이다.
- CSQL 인터프리터는 독립 모드(Standalone Mode), 클라이언트/서버 모드(Client/Server Mode), 시스템 관리자 모드(System admin Mode)를 제공한다.

옵션	인자	설명	기본값
-S -C		stand-alone, client-server mode 지정(On/Off-line)	-C
-u	DB 사용자	DB 접속 계정 정보	public
-p	'password'	DB 비밀번호 정보	없음
-i	File name	배치 모드에서 입력파일 이름 지정	없음
-c	'SQL'	CSQL 명령 시 SQL문을 직접 입력해 수행	없음
-e		CSQL 명령에서 수행한 SQL문장에 오류가 있어도 무시	없음
-o	File name	CSQL 수행한 SQL 수행 결과를 파일로 저장	없음
--no-auto-commit		CSQL 접속 시 auto commit off	ON



5. 기능소개



6. CSQL 인터프리터(2/2)

- CSQL 접속 방법
 - 현재 접속한 CUBRID 서버로 CSQL 접속방법

```
$ csql -C -u db_user -p 'qwe123' demodb@localhost
```

- 외부(원격) CUBRID 서버로 CSQL 접속방법

```
$ csql -C -u db_user -p 'qwe123' demodb@192.168.0.100
```

- 데이터베이스 정지 상태 CSQL 접속방법

```
$ csql -S -u db_user -p 'qwe123' demodb
```

- CSQL 옵션 사용방법
 - -i 옵션 사용방법은 수행할 질의를 파일명(infile)로 만들어 사용하는데 질의별로 마지막은 세미콜론(;) 붙여 입력

```
$ csql -u db_user -p 'qwe123' -i infile demodb@localhost
```

- -c 옵션을 이용하여 셸 상에서 하나 이상의 SQL 문을 직접 수행한다. 이 때, 각 문장은 세미콜론(;)으로 구분

```
$ csql -u db_user -p 'qwe123' -c 'select * from olympic;select * from stadium' demodb@localhost
```



5. 기능소개



7. 백업과 복구(1/8)

- 데이터베이스 백업 시 참고사항
 - 데이터베이스 백업은 데이터베이스의 이미지를 파일 등으로 덤프 하는 것이다.
 - OS 유틸리티(cp, sftp/ftp, scp등)를 사용한 데이터베이스 볼륨(파일) 백업은 권장하지 않는다.
 - 데이터베이스 백업 시점에 불필요한 로그 아카이브 볼륨들을 정리 할 수 있다.
 - 백업은 매일 받는 것을 권장하며, DBA 직접 수행하는 방법보다 새벽시간 자동백업을 설정하는 것이 좋다.
 - 백업 볼륨은 외부 저장장치에 백업하는 것이 안전하다.
 - 데이터베이스를 새로운 버전으로 migration했다면 새로 생성한 데이터베이스를 즉시 백업해야 한다. 이전 버전의 백업 볼륨은 새로운 버전 복구에 사용 불가능하기 때문이다.
 - 백업 볼륨을 이동하거나 이름을 변경하지 않으면 이후의 백업 시에 이전 볼륨을 덮어쓴다.
 - 백업 볼륨은 데이터베이스를 가장 최근의 상태로 복구하기 위해서 로그 볼륨과 함께 사용된다.
- 데이터베이스백업 정책
 - 백업할 데이터의선택
 - 데이터베이스의 전체 또는 일부만 백업 할 것인가?
 - 데이터 보존 기간은 얼마로 할 것인가?
 - 데이터베이스와 함께 백업되어야 할 다른 파일은 있는가?
 - 백업 방법(backupdb)
 - 사용 가능한 백업 툴 및 백업 장비
 - FULL/INCREMENTAL백업(0,1,2 level)
 - On-line/Off-line 백업
 - 백업 시기
 - 데이터베이스의 활동이 가장 적은 시기



5. 기능소개



7. 백업과 복구(2/8)

- 명령어 : backupdb

```
$ cubrid backupdb [options] database_name
```

- Disk와 tape 등의 미디어 지정이 가능하다.
- 백업이 완료되면 백업볼륨 및 백업정보 파일을 생성한다. (예, demodb_bk0v000, demodb_bkvinf)

옵션	인자	설명	기본값
-S -C		stand-alone, client-server mode 지정(On/Off-line)	-C
-D	filepath/device	볼륨이 저장될 경로 지정	log file 경로와 같음
-l	0,1,2	백업 레벨	0
-r		백업 후 불필요한 archive log 삭제	수행 않음
--no-check		백업 전에 데이터베이스 일관성 점검을 수행하지 않는다.	수행
-z		데이터베이스를 압축하여 백업 볼륨에 저장함	수행 않음
-t	integer	백업을 수행하는 thread 수	0<auto>
-e		백업 시 활성 로그 볼륨을 포함하지 않도록 설정 함	수행 않음

- 사용 예

```
$ cubrid backupdb -C -z -r -D /CUBRID/databases/demodb -l 0 demodb  
⑦ windows% cubrid backupdb -C -z -r -D C:\CUBRID\databases\demodb -l 0 demodb
```

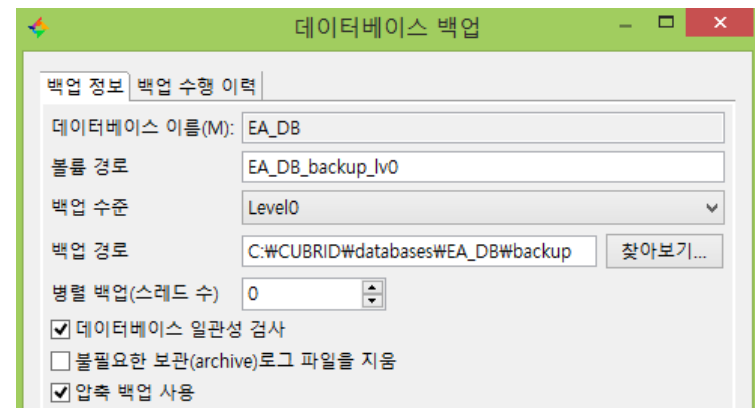
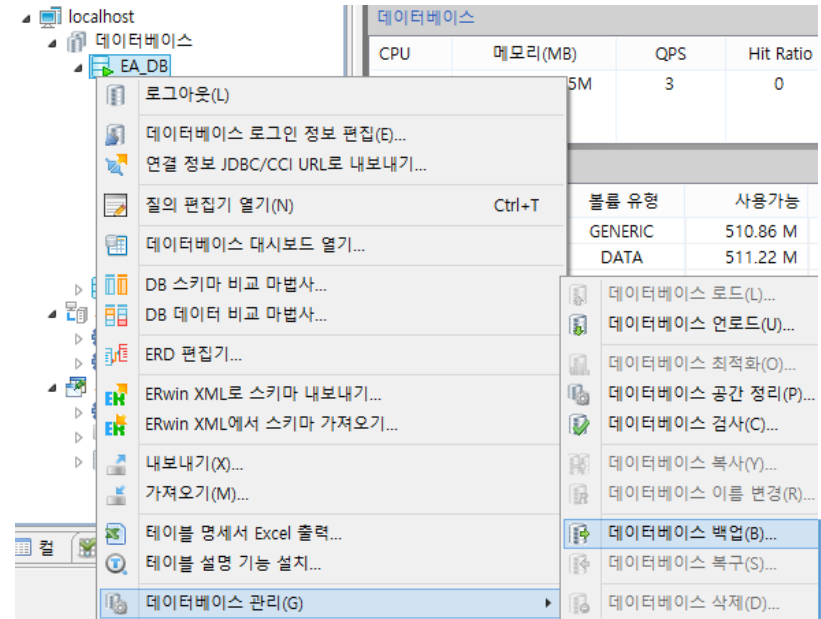


5. 기능소개



7. 백업과 복구(3/8)

- CUBRID Manager를 이용한 백업 방법
 1. 데이터베이스를 선택
 2. 백업 정보
 - 백업 볼륨의 이름을 명시한다
 - 백업 레벨을 선택한다. Full backup을 받은 적이 없는 경우 level 0만 표시된다.
 - 백업 경로를 선택한다.
 - 백업 작업을 수행할 thread개수를 설정한다.
 - 데이터베이스 일관성 검사(DB이상 유무 확인)
 - 보관로그 파일 지움(archive log 정리)
 - 압축 백업 사용(백업 파일을 압축하여 저장한다.)

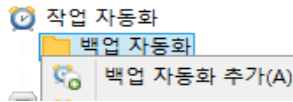


5. 기능소개



7. 백업과 복구(4/8)

- CUBRID Manager를 이용한 백업 자동화 설정



1. 백업 정보

- 백업 ID : 임의 값 설정
- 백업 level 선택 : 처음 백업 시 level 0만 가능
- 백업 경로 : 별도의 Disk 또는 media 권장

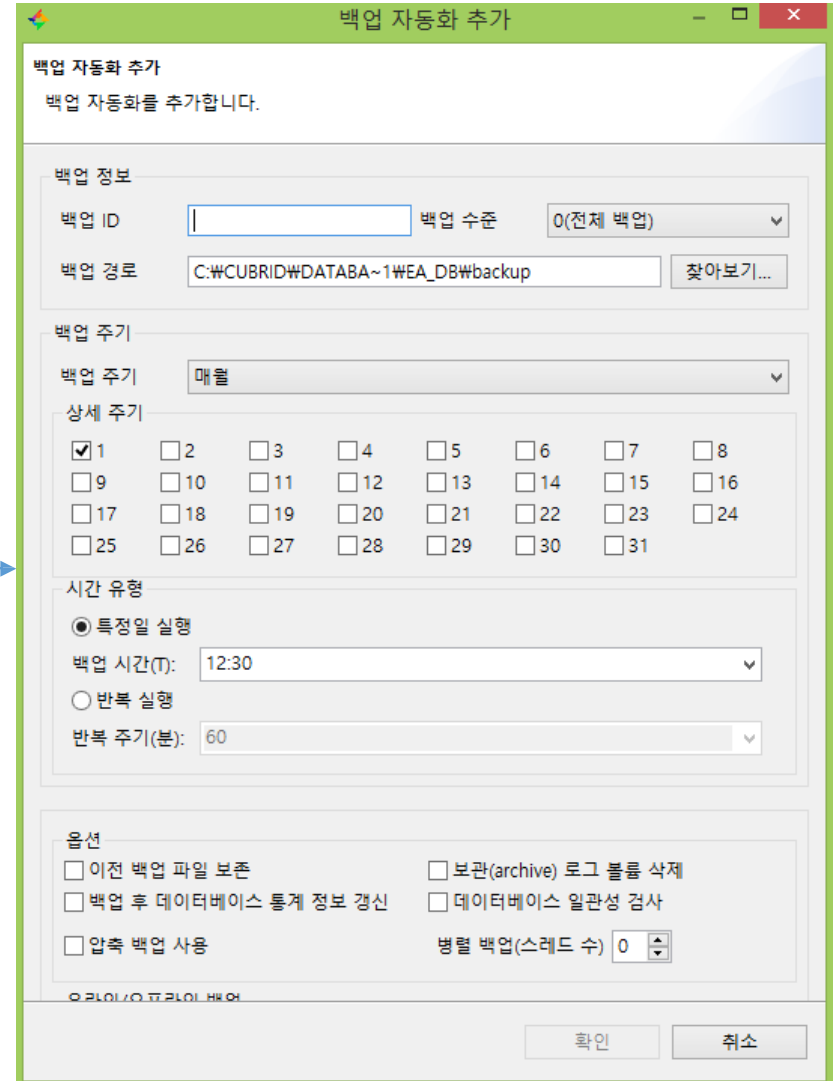
2. 백업 주기

- 백업 주기 : 디스크/백업파일 사이즈에 따라 선택
- 상세 주기 : 백업주기에 따라 날짜, 요일 등
- 백업 시간 : DB활동이 가장 적은 시기(시, 분)

3. 옵션

- 이전 백업 파일 보존 : 이전 단계 복구가 필요할 경우,
- - 보관 로그 볼륨 로그 삭제: archive log 정리 여부(권장)
- 백업 후 데이터베이스 통계 정보 갱신 : DB에 갱신 작업이 빈번하게 발생하는 경우 주기적인 작업이 필요
- 데이터베이스 일관성 검사 : 백업 시 DB 이상 유무를 확인,
- 압축 백업 사용 : 사용할 것을 권장
- 병렬 백업 : 백업 작업에 이용될 thread 수

4.온/오프라인 백업 : 오프라인 백업 시 DB가 정지 된 후 백업을 진행하고 DB가 구동된다.



5. 기능소개



7. 백업과 복구(5/8)

- 명령어 :restoredb

```
$ cubrid restoredb [options] database_name
```

옵션	인자	설명	기본값
-l	복구 레벨	복구 레벨을 지정한다(0, 1, 2)	0
-d	복구 시점	복구할 시점을 지정 형식) dd-mm-yyyy:hh:mm:ss 예) 21-12-2002:17:00:10. 또는 backuptime : 마지막 백업 시점으로 복구 시 사용	가장 최근 시점
-B	파일 패스	복구할 백업 볼륨이 존재하는 디렉터리나 장치 명을 지정	dbname_bkvinf 경로
-p		archive 로그가 없을 경우 무시하고 수행하여 사용자 인터페이스를 받지 않을 경우	
-u		데이터베이스 위치 파일 내에 지정된 경로로 데이터베이스와 로그 볼륨을 복구(databases.txt)	
--list		백업을 수행하지 않고 백업 볼륨 목록을 출력할 경우	



5. 기능소개



7. 백업과 복구(6/8)

- 데이터베이스 복구
 - 백업 시점으로 복구 또는 백업 이후 원하는 시점을 복구 가능하다.
 - `restoredb` 명령어 수행 시 `-d` 옵션을 사용하지 않으면 복구가능한 가장 최근 시점으로 자동 복구한다.
- 데이터베이스 복구방법
 - 최근 시간/시점으로 복구

```
$ cubrid restoredb -d 19-02-2015:17:39:00 demodb
```

- 백업 시간/시점으로 복구

```
$ cubrid -d backuptime restoredb demodb
```

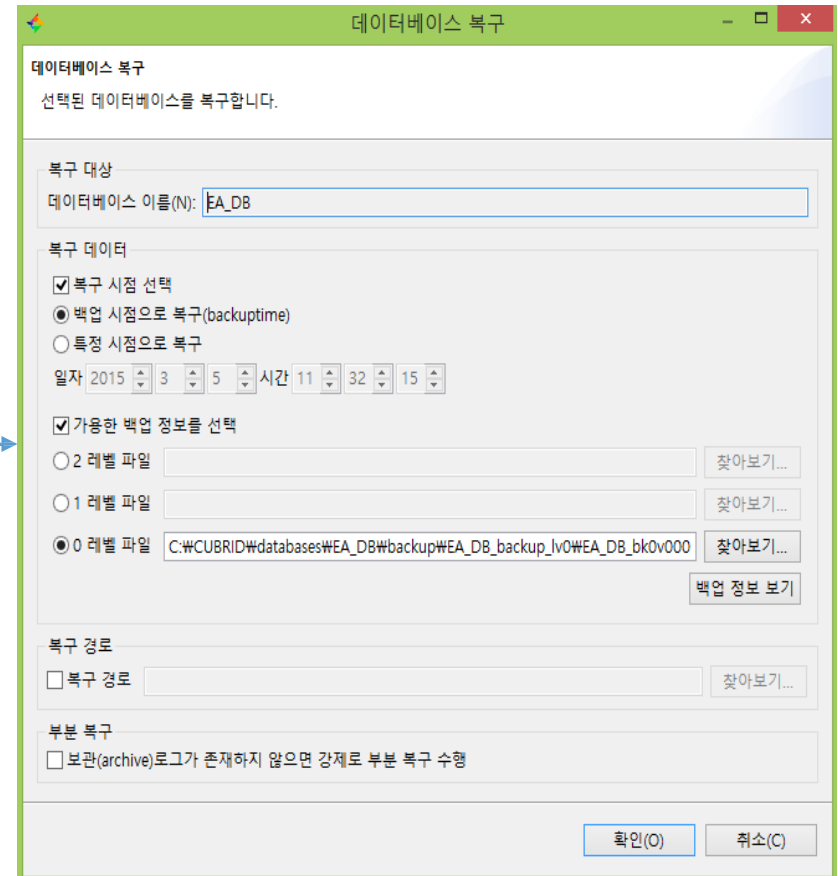
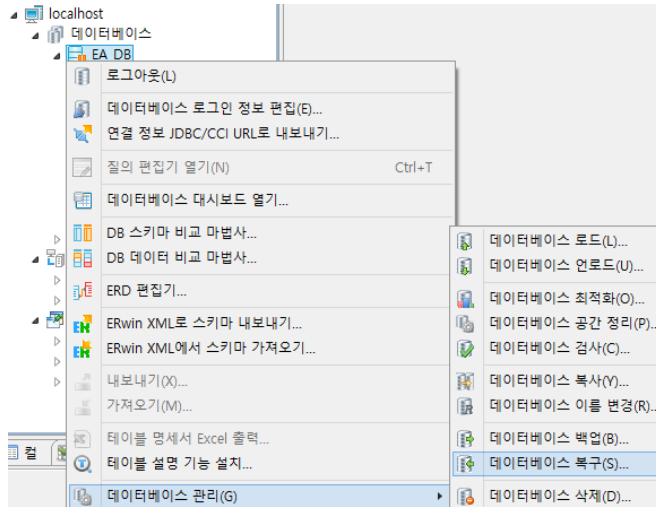


5. 기능소개



7. 백업과 복구(7/8)

- CUBRID Manager를 이용한 데이터 복구
 - DB Server 종료
 - 백업 시점에서의 복구 및 백업 이후 원하는 시점을 복구 가능하다.



1. 복구 대상 데이터베이스 이름
2. 복구 데이터
 - 복구 시점 선택 : 백업 시점 또는 특정 시점
 - 가용 가능한 백업 정보 선택 : <db_name>_bkvinf 파일에 백업의 경로가 등록되어 있을 경우 자동으로 검색 된다.
3. 복구 경로 : 백업을 복구할 위치를 지정할 수 있다.
4. 부분 복구 : archive log 부재 시 백업 볼륨만 가지고 복구



5. 기능소개



7. 백업과 복구(8/8)

- 데이터베이스 복구
 - 디스크 장애(mediafailure) 발생
 - H/W(물리적) 디스크 복구 이후 데이터베이스 볼륨(파일) 복구에 실패하거나 손상이 있을경우 backupdb로 받은 백업볼륨을 이용해서 restoredb 명령으로 복구한다.
 - 데이터베이스 특정시점 복원
 - Where 조건이 없는 delete/update로 인한 특정 시점 복원이 필요할 때 백업된 데이터베이스와 로그를 이용해서 복구한다. 이때 백업 볼륨과 로그볼륨(archive log, active log)은 반드시 필요한 위치에 있어야 한다.
- 로그 복구
 - 로그 복구 방법은 CUBRID Manager에서는 지원하지 않는 기능이다.
 - DB서버 및 데이터베이스가 비정상 종료된 경우 손상된 로그를 복구한다.
 - 로그 복구 시 시간이 많이 걸릴 수 있으며, 복구 완료 이전 강제 종료하면 손상이 더 심해질 수 있다.

```
$ csq! -S -u dba demodb
```

- 위의 명령 수행 중 에러 발생과 함께 실패되면 다음 명령 수행한다.

```
$ cubrid emergency_patchlog demodb
```

- 위 명령어 수행 후 csq! 접속 시 이상 없으면 복구 완료, 여전히 에러 발생하면 다음 명령 수행한다.

```
$ cubrid emergency_patchlog -r demodb
```

- 위 명령어 수행 후 csq! 접속 시 이상 없으면 복구 완료, 여전히 에러 발생하면 “백업” 을 이용한 복구 수행한다.



6. 활용예제

세부 목차



1. 연결설정(JDBC, ODBC, ADO.NET, PHP, CCI)
2. 데이터베이스 서버 구축 예제



6. 활용예제



6.1 연결설정(1/5)

JDBC 연결설정

- 사용환경
 - JDK1.6 이상
- JDBC driver
 - <CUBRID 설치 디렉토리>/jdbc/cubrid_jdbc.jar
- 연결설정

```
import java.sql.*;
import cubrid.jdbc.driver.*;

String url = "jdbc:cubrid:192.168.0.100:33000:demodb::";
String url_c = "jdbc:cubrid:192.168.0.100:33000:demodb::?charset=utf8";
String url_ha = "jdbc:cubrid:192.168.0.100:33000:demodb::?alhosts=192.168.0.101:33000&charset=utf8";

String db_user = "db_user";
String db_passwd = "passWd";

Class.forName("cubrid.jdbc.driver.CUBRIDDriver");
Connection conn = DriverManager.getConnection(url, db_user, db_passwd);
...
```



6. 활용예제



6.1 연결설정(2/5)

ODBC 연결설정

- 설치
 - 통합설치
 - 개별설치 ftp.cubrid.org
 - Unicode(UTF8), ANSI 문자셋 설치파일이 각각 있음

- 연결설정

```
// ANSI 문자셋
sConn = "driver={CUBRID Driver};server=localhost;port=33000;uid=db_user;pwd=passWd;db_name=demodb;"

// UNICODE ( UTF8 ) 문자셋
sConn = "driver={CUBRID Driver Unicode};server=localhost;CHARSET=utf-8;
port=33000;uid=db_user;pwd=passWd;db_name=demodb;"

//HA 접속 예제
sConn = "driver={CUBRID Driver Unicode};server=localhost;CHARSET=utf-8; port=33000;uid=db_user;pwd=pass
Wd;db_name=demodb;alhosts=192.168.0.218:33000;loginTimeout=600;"
```

- transection control

```
sConn.BeginTrans
... sConn.ComitTran
s sConn.RollbackTra
ns
```

- isolation level

```
sConn.IsolationLevel = ADODB.IsolationLevelEnum.adXactReadUncommitted
```



6. 활용예제



6.1 연결설정(3/5)

ADO.NET 연결설정

- 설치
 - 설치 본
 - 라이브러리
- 연결설정

```
using CUBRID.Data.CUBRIDClient;

namespace CUBRID.ADO.NET.DataProvider.MyClass
{
    ...
    sb = new CUBRIDConnectionStringBuilder();
    sb.User = "db_user" ;
    sb.Password = "passWd";
    sb.Database = "demodb";
    sb.Port = "33000"; sb.Server = "localhost";
    using (CUBRIDConnection conn = new CUBRIDConnection(sb.GetConnectionString()))
    {
        conn.Open();
    }
    ...
}
```



6. 활용예제



6.1 연결설정(4/5)

PHP 연결설정

- 설치
 - 빌드
 - 빌드 본 설치
- 연결설정

```
$conn = cubrid_connect("localhost", 33000, "demodb", "db_user", "passWd");  
$con = cubrid_connect_with_url("cci:CUBRID:localhost:33000:demodb:db_user:passWd");  
$con = cubrid_connect_with_url("cci:CUBRID:localhost:33000:demodb:db_user:passWd?autocommit=true");  
$con = cubrid_connect_with_url(  
    "cci:CUBRID:localhost:33000:demodb:db_user:passWd?althosts=192.168.0.101:33000&  
    login_timeout=5000&disconnect_on_query_timeout=true", "db_user", "passWd");
```



6. 활용예제



6.1 연결설정(5/5)

CCI 연결설정

- 설치
 - 통합설치
- 빌드 환경
 - header file : <CUBRID 설치 디렉토리>/include/cas_cci.h
 - library file : <CUBRID 설치 디렉토리>/lib/libcascci.so, windows의 경우: cascci.lib
- 배포
 - LINUX : <CUBRID 설치 디렉토리>/lib/libcascci.so
 - Windows : <CUBRID 설치 디렉토리>/bin/cascci.dll

```
T_CCI_ERROR cci_error;
con = cci_connect_with_url("cci:CUBRID:localhost:33000:demodb:db_user:passWd:", "db_user", "passWd");
if (con < 0) {
    printf("ERR: cci connect failure\n");
    return -1;
}
...
req = cci_prepare (con, query, 0, &cci_error);
error = cci_execute (req, 0, 0, &cci_error);
if (error < 0) {
    printf ("execute error: %d, %s\n", cci_error.err_code, cci_error.err_msg);
}
do {
    error = cci_fetch (req, &cci_error);
    ...
} while ((res = cci_cursor(req, 1, CCI_CURSOR_CURRENT, &error)) == 0);
error = cci_close_req_handle (req); cci_end_transaction(con, CCI_TRAN_COMMIT, &error); cci_disconnect(con);
```



6. 활용예제



6.2 데이터서버 구축 예제(1/7)

데이터베이스 생성

- 데이터베이스 생성명령어
 - `cubrid created [options] databases_name locale_name.charset`
- 데이터베이스 생성하기

- CUBRID 설치 계정 생성

```
root#> useradd cubrid
```

- CUBRID 생성 계정 비밀번호 입력

```
root#> passwd cubrid
```

- CUBRID 생성 계정 로그인

```
root#> su - cubrid
```

- 데이터베이스 설치 디렉토리 생성

```
cubrid$> mkdir /home/cubrid/db_data
```

- 데이터베이스 설치 디렉토리 이동

```
cubrid$> cd /home/cubrid/db_data
```

- 데이터베이스 생성(DB명: **nipa_db**) cubrid

```
$> cubrid createdb nipa_db ko_KR.utf8
```

- 데이터베이스 볼륨 생성

```
cubrid$> cubrid addvoldb -S -p data --volume-name=nipa_db_data_x001--db-volume-size=10G nipa_db
```

```
cubrid$> cubrid addvoldb -S -p index --volume-name=nipa_db_index_x002--db-volume-size=2G nipa_db
```

```
cubrid$> cubrid addvoldb -S -p temp --volume-name=nipa_db_temp_x003--db-volume-size=2G nipa_db
```



6. 활용예제



6.2 데이터서버 구축 예제(2/7)

데이터베이스 환경설정

- 데이터베이스 파라미터
 - \$CUBRID/conf/cubrid.conf
- 데이터베이스 파라미터 변경(WAS connection pool max 300 기준)

- 데이터베이스 파라미터 파일 오픈
cubrid\$> vi \$CUBRID/conf/cubrid.conf

- 데이터베이스 메모리 및 기본 파라미터 변경
The list of database servers in all by 'cubrid service start' command.
This property is effective only when the above 'service' property contains 'server' keyword.
server=**nipa_db**

- # Size of data buffer are using K, M, G, T unit
data_buffer_size=**10G**

- # The maximum number of concurrent client connections the server will accept.
This value also means the total # of concurrent transactions.
max_clients=**330**



6. 활용예제



6.2 데이터서버 구축 예제(3/7)

브로커 환경설정

- 데이터베이스 파라미터
 - \$CUBRID/conf/cubrid_broker.conf
- 데이터베이스 파라미터 변경 (WAS connection pool max 300 기준)

• 데이터베이스 파라미터 파일 오픈
cubrid\$> vi \$CUBRID/conf/cubrid_broker.conf

• 브로커 포트 및 connection pool max size 변경

```
[broker]
MASTER_SHM_ID    =50001
ADMIN_LOG_FILE    =log/broker/cubrid_broker.log
```

```
[%query_editor] SERV
ICE               =ON
BROKER_PORT       =50000
MIN_NUM_APPL_SERVER =3
MAX_NUM_APPL_SERVER =40
APPL_SERVER_SHM_ID =50000
LOG_DIR           =log/broker/sql_log ERRO
R_LOG_DIR         =log/broker/error_log
SQL_LOG           =ON
TIME_TO_KILL      =120
SESSION_TIMEOUT   =300 KEEP
_CONNECTION       =AUTO C
CI_DEFAULT_AUTOCOMMIT =ON
```

```
[%BROKER1]
SERVICE         =ON BROK
ER_PORT          =51000
MIN_NUM_APPL_SERVER =150
MAX_NUM_APPL_SERVER =300
APPL_SERVER_SHM_ID =51000
LOG_DIR          =log/broker/sql_log ERRO
R_LOG_DIR        =log/broker/error_log
SQL_LOG          =ON
TIME_TO_KILL     =120
SESSION_TIMEOUT   =300 KEEP
_CONNECTION       =AUTO C
CI_DEFAULT_AUTOCOMMIT =ON
```



6. 활용예제



6.2 데이터서버 구축 예제(4/7)

매니저 환경설정

- CUBRID 매니저 파라미터
 - \$CUBRID/conf/cm.conf
- CUBRID 매니저 파라미터 변경

```
• 데이터베이스 파라미터 파일 오픈
cubrid$> vi $CUBRID/conf/cm.conf

• 데이터베이스 메모리 및 기본 파라미터 변경
# cm.conf
# -- CUBRID database management tool server configuration file
#
#
# When server starts, it looks for the environment variable
# 'CUBRID_MANAGER' and use it to locate this file. It is assumed that
# 'CUBRID_MANAGER' is the root directory of all CUBRID Manager related files.
#
# Manager server section - a section for 'cubrid service' command
# Common section - properties for CUBRID Manager Server
# This section will be applied before starting manager server.
[cm]
#
# Port number designation
# A port for the connection between CUBRID Manager server and Client.
# CUBRID Manager server uses the value cm_port.
# The default value is 8001.
#
cm_port=9001

#
# CMS Process Monitoring interval setting
#
cm_process_monitor_interval=3
```



6. 활용예제



6.2 데이터서버 구축 예제(5/7)

서버 환경설정

- 리눅스 방화벽과 커널 및 환경설정
 - iptables, /etc/security/limits.conf, /etc/hosts
- 방화벽, limits.conf, hosts 파일 변경(CentOS release 6.7 기준)

- 방화벽 정지

```
root#> service iptables stop
iptables: 체인을 ACCEPT 규칙으로 설정 중: filter [ OK ]
iptables: 방화벽 규칙을 지웁니다: [ OK ]
iptables: 모듈을 언로드하는 중: [ OK ]
```

- /etc/security/limits.conf 파일 변경

```
root#> vi /etc/security/limits.conf cu
brid soft nofile 300000
cubrid hard nofile 300000
cubrid soft core unlimited
cubrid hard core unlimited
cubrid soft stack 10240 cu
brid hard stack 10240
```

- /etc/hosts 파일 변경

```
root#> hostname cubri
d_db01
root#> ifconfig
inet addr:192.168.0.178 r
oot#> vi /etc/hosts 192.1
68.0.178 cub_db01
```



6. 활용예제



6.2 데이터서버 구축 예제(6/7)

CUBRID service 구동 및 프로세스 확인

- CUBRID service 구동
 - cubrid service start
- CUBRID service 구동과 프로세스 확인

```
•CUBRID service 구동
cubrid$> cubrid service start
@ cubrid master start
++ cubrid master start: success
@ cubrid server start: niap_db
This may take a long time depending on the amount of recovery works to do.
CUBRID 9.3
++ cubrid server start: success
@ cubrid broker start
++ cubrid broker start: success
@ cubrid manager server start
++ cubrid manager server start: success
```

- CUBRID service 프로세스 확인

```
cubrid$> cubrid service status
@ cubrid master status
++ cubrid master is running.
@ cubrid server status
Server testdb (rel 9.3, pid 11981)
Server demo_bak (rel 9.3, pid 11547)
@ cubrid broker status
NAME      PID PORT AS JQ TPS QPS SELECT INSERT UPDATE DELETE OTHERS LONG-T LONG-Q ERR-Q
=====
* query_editor 12200 50000 5 0 0 0 0 0 0 0 0 0/60.0 0/60.0 0
* broker1      12211 51000 5 0 0 0 0 0 0 0 0 0/60.0 0/60.0 0
@ cubrid manager server status
++ cubrid manager server is running.
```



6. 활용예제



6.2 데이터서버 구축 예제(7/7)

데이터베이스 계정과 테이블 생성

- 데이터베이스 접속
 - csql [options]
- csql 인터프리터 명령어로 데이터베이스 접속

- csql 인터프리터 접속

```
cubrid$> csql -u dba testdb
CUBRID SQL Interpreter
Type `;help` for help messages.
csql>
```

- 신규 계정과 비밀번호 생성

```
csql> create user user_01 password 'cubrid123!!!';
```

- 테이블과 인덱스 생성 및 스키마 확인

```
csql> create table tbl_01(id int primary key, [date] datetime, name varchar(20), jumin char(14));
csql> create index idx01_tbl_01_jumin_name on tbl_01(jumin, name);
csql> ;sc tbl_01
<ClassName>
tbl_01
<Attributes>
id      INTEGER NOT NULL
date    DATETIME
name    CHARACTER VARYING(20)
jumin   CHARACTER(14)
<Constraints>
PRIMARY KEY pk_tbl_01_id ON tbl_01 (id)
INDEX idx01_tbl_01_jumin_name ON tbl_01 (jumin, name)
```

- 데이터 입력과 조회

```
csql> insert into tbl_01 values(101, sysdatetime, 'kim', '123456-1234567');
```

```
csql> select * from tbl_01
```

id	date	name	jumin
101	03:47:28.376 PM 09/25/2017	'kim'	'123456-1234567'





Q CUBRID DBMS를 활용하여 솔루션 개발 후 해당 솔루션의 소스코드를 open해야 하나요?

&

A CUBRID DBMS는 Interface(JDBC, ODBC, PHP등)를 BSD License로 채택하고 있어, 솔루션의 연동 시 해당 솔루션의 소스코드를 Open하여야 할 의무가 없습니다. 다만, 서버는 GPL 라이선스를 채택하고 있어 서버의 소스코드 또는 library를 이용하시거나 수정하신 경우에는 소스코드를 open해야 할 의무가 있습니다.

Q CUBRID DBMS를 사용 시 비용은 어떻게 되나요?

&

A CUBRID DBMS의 라이선스는 무료입니다. 다만, 기술지원서비스(개발지원서비스, 유지관리서비스)가 필요하신 사용자는 큐브리드 사에서 제공하는 서비스를 선택하시어 계약을 체결하여 비용을 지불하면 됩니다.



7. FAQ



Q CUBRID DBMS 사용 시 기술지원서비스의 종류는 어떻게 되나요?

&

A 큐브리드 사 홈페이지 참조 http://www.cubrid.com/zbxe/service_enduser



8. 용어정리



용어	설명
Broker	Broker는 CUBRID DB와 interface(JDBC, ODBC, PHP 등)을 연결하는 CUBRID 전용 middleware
CM	Cubrid Manager로 query browser와 db manager 기능을 제공하는 tool
CMT	Cubrid Migration Tool로 타 DBMS(Oracle, MS-SQL, MySQL) 또는 CUBRID로부터 schema와 data를 이관하는 tool



Open Source Software Installation & Application Guide



이 저작물은 크리에이티브 커먼즈 저작자표시 2.0 대한민국 라이선스에 따라 이용하실 수 있습니다.