

CUBRID 소개

—

작성년월일: 2009년 3월

목차

1. CUBRID 소개

- 1.1 CUBRID 특징
- 1.2 CUBRID 구조

2. CUBRID 설치

- 2.1 지원 플랫폼
- 2.2 설치 전 준비사항
- 2.3 Window에서 CUBRID 설치하기
- 2.4 Linux에서 CUBRID 설치하기

3. CUBRID 매니저 사용

- 3.1 CUBRID 매니저 소개
- 3.2 CUBRID 매니저 시작과 종료
- 3.3 CUBRID 매니저로 쿼리 실행
- 3.4 CUBRID 매니저로 브로커 실행

4. CSQL 실행

- 4.1 CSQL 시작
- 4.2 원격에서 CSQL 실행

1. CUBRID 소개

1.1 CUBRID 일반적 특징

1.2 CUBRID 구조

1.0 CUBRID 소개

CUBRID

<http://www.cubrid.com>

NAVER
DeveloperCenter

<http://dev.naver.com/projects/cubrid>

1.1 CUBRID 특징 1

기능 구분	CUBRID 7.3
SQL	SQL-92, SQL-99(ODB)
Data type	Alphanumeric, multimedia(CLOB, BLOB)
Modeling	RDB (Table, column, RI), ODB (Class, method, nested column, set)
Object	Table, view, trigger
API	ODBC, JDBC, OLEDB, Native API, PHP, etc.
Procedures	Method, Java SP
Transaction	Record locking, Online incremental backup/recovery, XA
Security	Authorization, Data encryp., PKI login
Availability	HA, Replication
Miscellaneous	Partition, MDBC, multimedia framework

- RDBMS 계승
 - 보편성, 확장성, 안정성
- 대용량
 - 멀티볼륨 DB
 - DB/테이블: 개수 및 크기 무제한
- 확장성
 - 멀티 쓰레드 서버
 - 복제
 - 분할
- 고성능 서버/클라이언트 구조
 - 미들웨어(Broker)가 쓰레드 풀 관리 및 부하 최적화 기능 수행
 - 멀티 쓰레드 서버
 - 복제 : 질의 분산
 - 분할: 질의 분할
 - 쿼리 플랜 캐쉬

1.1 CUBRID 특징 2

기능 구분	CUBRID 7.3
SQL	SQL-92, SQL-99(ODB)
Data type	Alphanumeric, multimedia(CLOB, BLOB)
Modeling	RDB (Table, column, RI), ODB (Class, method, nested column, set)
Object	Table, view, trigger
API	ODBC, JDBC, OLEDB, Native API, PHP, etc.
Procedures	Method, Java SP
Transaction	Record locking, Online incremental backup/recovery, XA
Security	Authorization, Data encryp., PKI login
Availability	HA, Replication
Miscellaneous	Partition, MDBC, multimedia framework

● 트랜잭션

- ACID 보장: commit, rollback, savepoint
- 장애시 트랜잭션 일치성 보장
- 복제간 트랜잭션 일치성 보장
- 백업/복구 시 트랜잭션 일치성 보장
- 다중 단위 잠금: DB, 테이블, 레코드 단위
- 데드락(dead lock) 자동해결

● 백업 및 복구

- 온라인/오프라인 백업 지원
- 증분 백업 지원 : 전체백업, 1단계, 2단계 백업
- 실시간 압축 백업 지원
- 병렬 백업 및 복구 지원
- 특정 시점에서의 복구 지원

● 보안 및 권한 관리

- 사용자 권한 관리: 그룹, 멤버, 개체
- 트랜잭션 별 사용 권한 설정

1.1 CUBRID 특징 3

기능 구분	CUBRID 7.3
SQL	SQL-92, SQL-99(ODB)
Data type	Alphanumeric, multimedia(CLOB, BLOB)
Modeling	RDB (Table, column, RI), ODB (Class, method, nested column, set)
Object	Table, view, trigger
API	ODBC, JDBC, OLEDB, Native API, PHP, etc.
Procedures	Method, Java SP
Transaction	Record locking, Online incremental backup/recovery, XA
Security	Authorization, Data encryp., PKI login
Availability	HA, Replication
Miscellaneous	Partition, MDBC, multimedia framework

- 풍부한 개발 환경
 - 표준 API: JDBC, ODBC, OLEDB, PHP, E-SQL
 - Native API: CCI-API
 - Java Stored Procedure
- 큐브리드 매니저
 - 플랫폼에 독립적인 GUI 도구
 - 통합 도구: 관리, 질의, 진단, 튜닝 등

1.1 CUBRID 특징 4

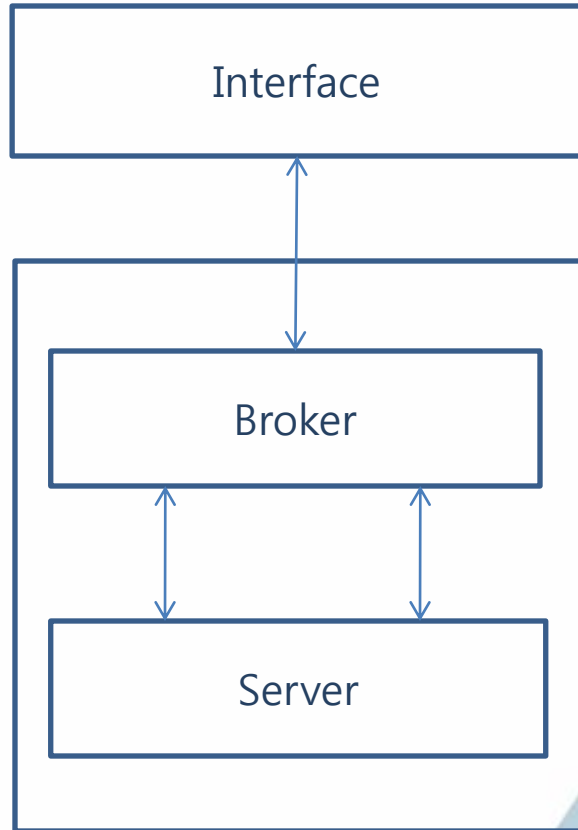
- Object 확장
 - 사용자 정의 데이터 타입(User Defined Data Type)
 - 집합형 데이터 타입(Collection Data Type)
 - 메소드(Method)
 - 상속(Inheritance)
- OID (Object Identifier)
 - Database 내 class 또는 record 에 대한 Identifier
 - 물리적 위치 정보(volume id, page id, slot id)
 - 검색을 하지 않으므로 데이터에 대한 빠른 접근 보장
 - 재구성되지 않는 한 변경되지 않음
 - 사용자 정의 데이터 타입 사용시 해당 컬럼에 저장되는 값
- 데이터 타입 확장
 - 모든 클래스가 데이터 타입으로 사용 가능
 - 중첩 객체 (nested object)
 - 조인 연산 없이 연결 객체 직접 접근
- 집합 객체
 - set, sequence, multi-set
 - 별도의 테이블/조인 필요 없음

캐릭터	
ID	char(10)
스킬수준	int
이름	char(10)
인벤토리	SET(아이템)

아이템	
아이템정보	아이템상세
취득시각	char(30)
취득방법	char(10)

아이템상세	
ID	int
이름	char(30)
종류	char(10)

1.2 CUBRID 구조 1



Server:

- 고성능의 클라이언트/서버 구조
- 대용량, 확장성, 가용성을 제공
- 완벽한 트랜잭션 기능을 제공

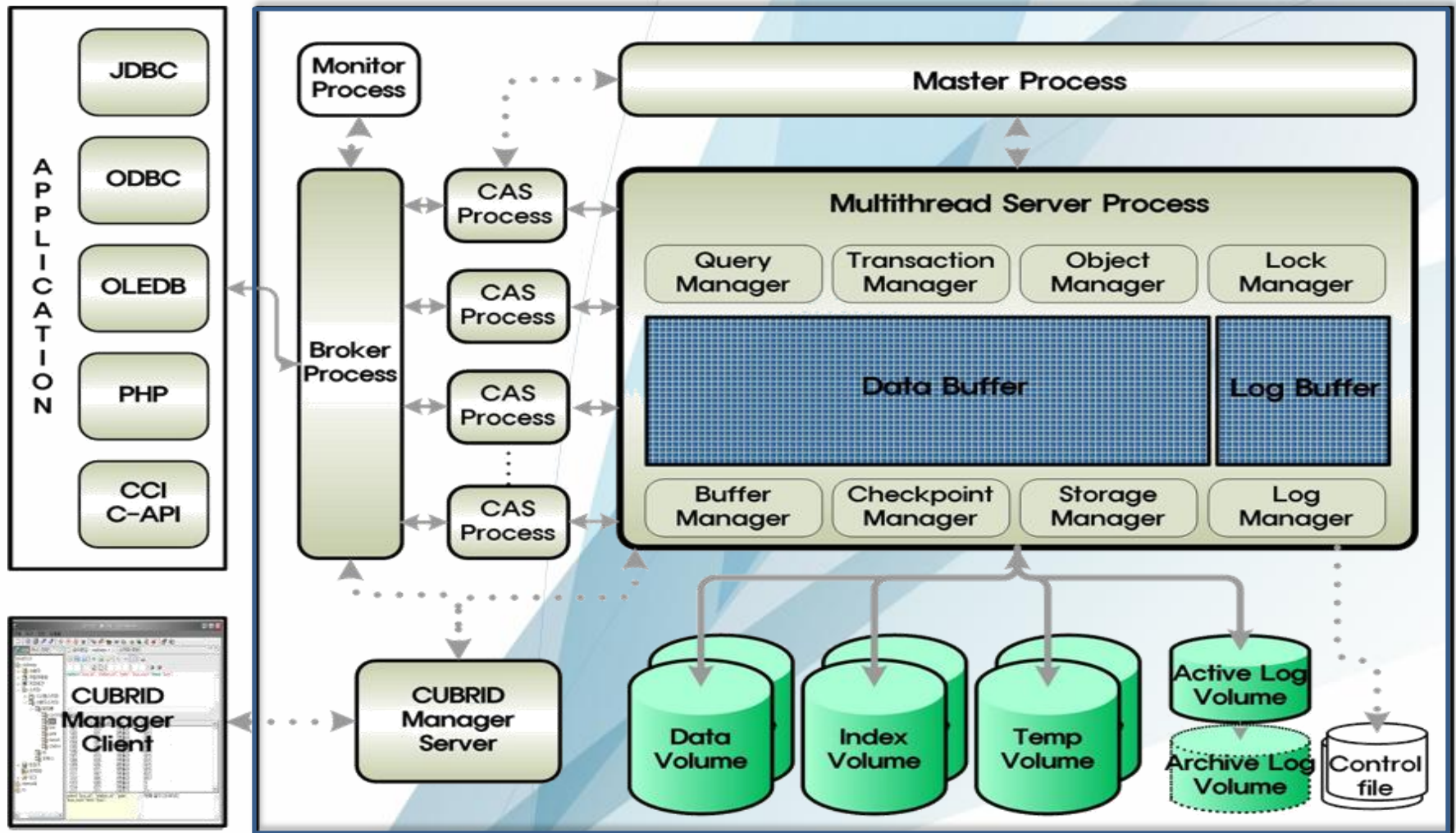
Broker:

- 응용 프로그램의 통신을 중계하는 미들웨어
- 커넥션 풀링, 모니터링, 로그 분석, 로드 밸런싱 제공

CUBRID Manager:

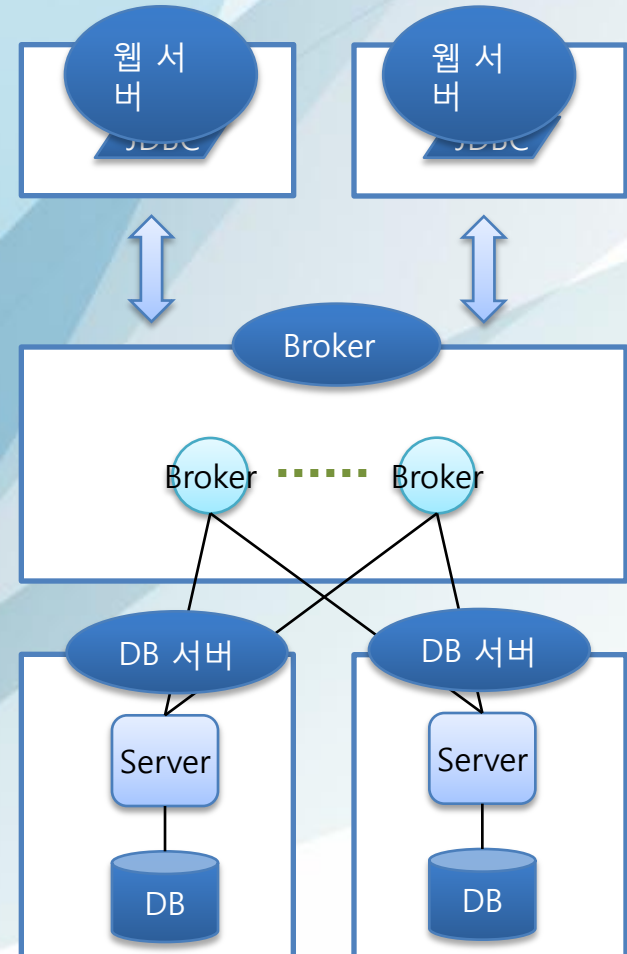
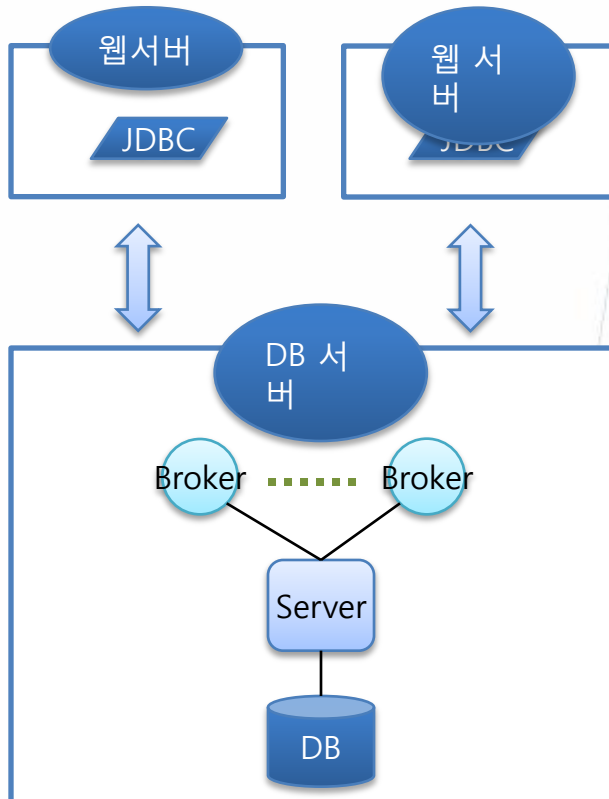
- Server 및 Broker 기능을 수행하는 GUI 도구
- Server 및 Broker 를 모니터링 및 분석하는 도구
- 질의 편집기를 통한 데이터베이스 개발 및 질의 튜닝 기능 제공

1.2 CUBRID 구조 2



1.2 CUBRID 구조 3

- Server와 Broker를 분리할 수 있는 구조



2. CUBRID 설치

2.1 지원 플랫폼

2.2 설치 전 준비사항

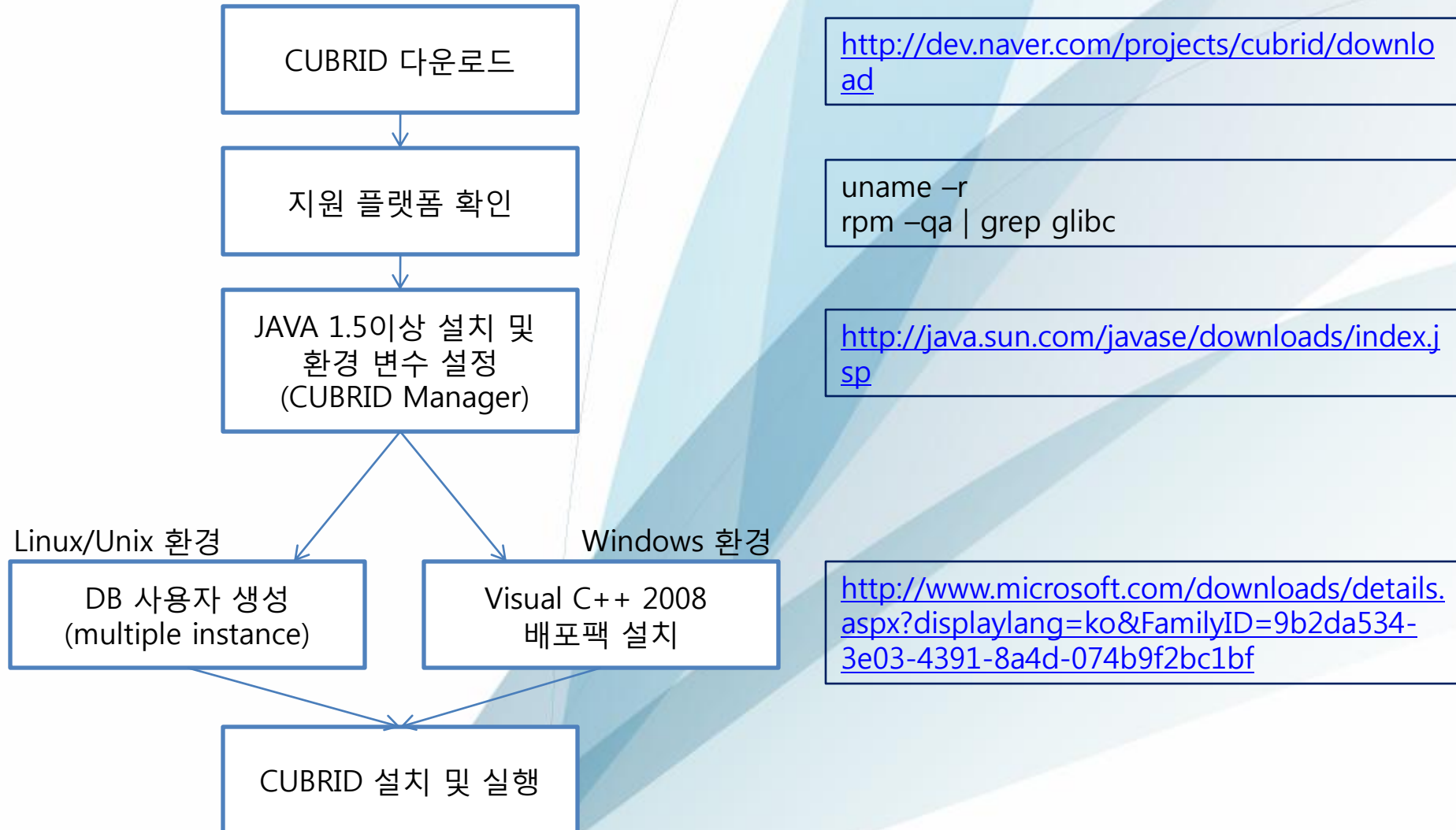
2.3 Window에서 CUBRID 설치하기

2.4 Linux에서 CUBRID 설치하기

2.1 지원 플랫폼

OS	OS version	CPU
Microsoft Windows	Windows NT / 2000 / XP / 2003 / vista 32	x86
LINUX	glibc-2.3.2 기반 - Asianux 1.0 - Redhat 9 - Redhat Enterprise ES/AS 3.0 - Fedora core 1 - CentOS 3.x	x86 / Intel EM64T / AMD64
	glibc-2.3.4 기반 - Asianux 2.0 - Fedora core 4 이상 - Redhat Enterprise ES/AS 4.0 - CentOS 4.x, 5.x - SUSE 9.0 이상 - Ubuntu 6.10 이상	x86 / Intel EM64T / AMD64
SUN	solaris 7 ~ 10 x86 solaris 10	sparc x86
HP	HP-UX 11i	PA-RISC
IBM	AIX 5.3	PowerPC

2.2 설치 전 준비사항 1



2.2 설치 전 준비사항 2

✓ JAVA 환경 변수 설정

– Windows

```
set JAVA_HOME=C:\Program Files\JAVA\jdk1.6.4
set PATH=%PATH%;%JAVA_HOME%\jre\bin\client
set CLASSPATH=%CUBRID%\jdbc\cubrid_jdbc.jar
```

– Linux

▪ bash shell

```
set JAVA_HOME=/usr/local/jdk1.6.4; export JAVA_HOME
set LANG=korean; export LANG
set PATH=$PATH:$JAVA_HOME/bin; export PATH
set
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$JAVA_HOME/jre/lib/i386:$JAVA_HOME/jre
/lib/i386/client; export LD_LIBRARY_PATH
set CLASSPATH=$CUBRID/jdbc/cubrid_jdbc.jar; export CLASSPATH
```

▪ c shell

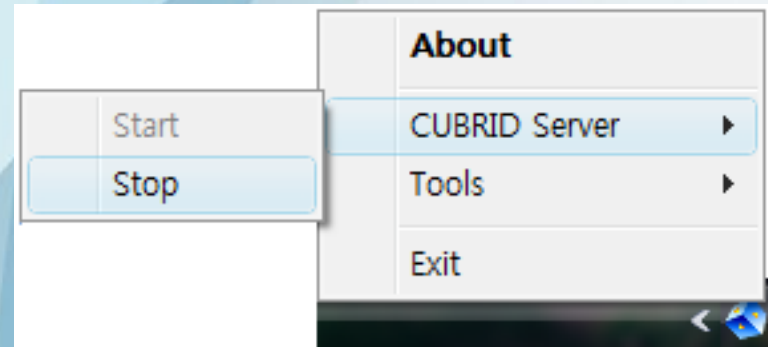
```
setenv JAVA_HOME=/usr/local/jdk1.6.4
setenv LANG=korean
set path=($path $JAVA_HOME/bin)
setenv LD_LIBRARY_PATH
$LD_LIBRARY_PATH:$JAVA_HOME/jre/lib/i386:$JAVA_HOME/jre/lib/i386/client
setenv CLASSPATH $CUBRID/jdbc/cubrid_jdbc.jar
```

– Unix

- 위 LINUX 환경변수 중 LD_LIBRARY_PATH 를 libjvm.so 가 있는 곳으로 수정
- 예) SUN : \$JAVA_HOME/jre/lib/**sparc**:\$JAVA_HOME/jre/lib/**sparc**/client

2.3 Windows에서 CUBRID 설치하기

- ✓ 실습을 위해 demodb 데이터베이스 생성
- ✓ CUBRID Service 트레이 확인
 - tray 상에 CUBRID icon 이 파란색으로 표시됨을 확인



- 서비스 등록 확인 (제어판→관리도구→서비스)

COM+ System Applicat...	CO...		수동	로컬 시스템
Computer Browser	네트...	시작됨	자동	로컬 시스템
Cryptographic Services	Win...	시작됨	자동	로컬 시스템
CUBRIDService		시작됨	자동	로컬 시스템
DCOM Server Process ...	DC...	시작됨	자동	로컬 시스템
DHCP Client	IP ...	시작됨	자동	로컬 시스템
Distributed Link Trackin...	네트...	시작됨	자동	로컬 시스템
Distributed Transaction...	데이...		수동	네트워크 서버

2.4 Linux에서 CUBRID 설치하기

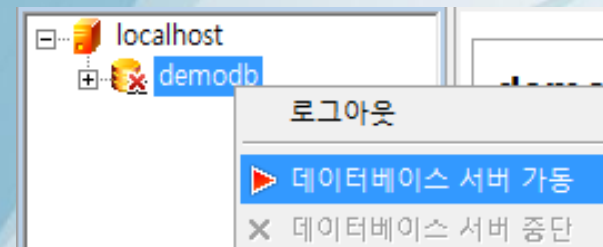
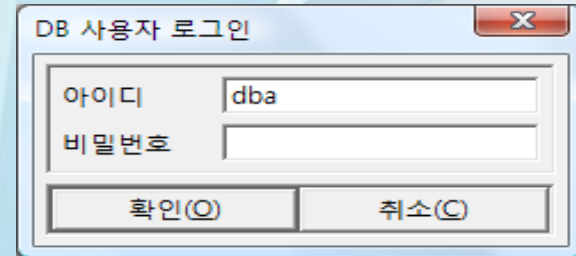
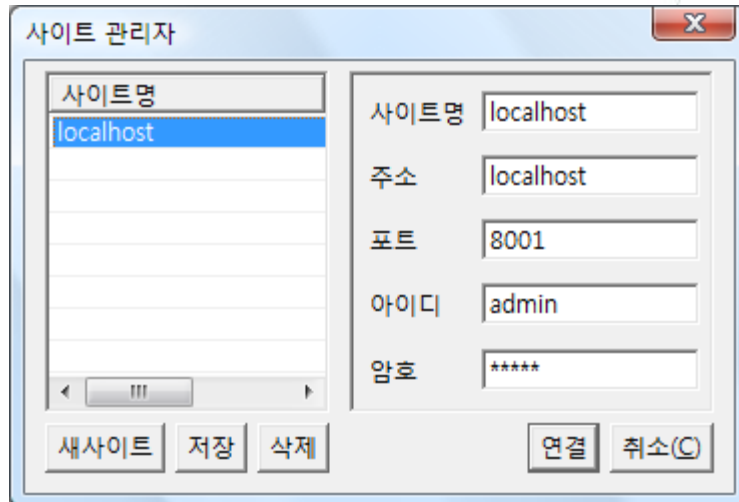
✓ CUBRID 설치 파일 실행

```
% sh CUBRID-8.1.3.1020-linux.sh
% . /home/cubrid/.cubrid.sh
% 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
```

2.5 CUBRID DB구동하기

- ✓ CUBRID Manager로 demo DB 구동



- ✓ 명령어로 demo DB 구동

```
% cubrid server start demodb
```

```
CUBRID 2008 R1.3
```

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

3. CUBRID Manager 사용

3.1 CUBRID 매니저 소개

3.2 CUBRID 매니저 시작

3.3 CUBRID 매니저로 쿼리 실행

3.4 CUBRID 매니저로 종료

3.1 CUBRID Manager 소개

- ✓ Java 기반의 GUI도구 → JRE/JDK 버전 1.5 이상 설치 필요
- ✓ Server 및 Broker의 기능 제어, 모니터링 및 로그 분석을 수행하는 도구

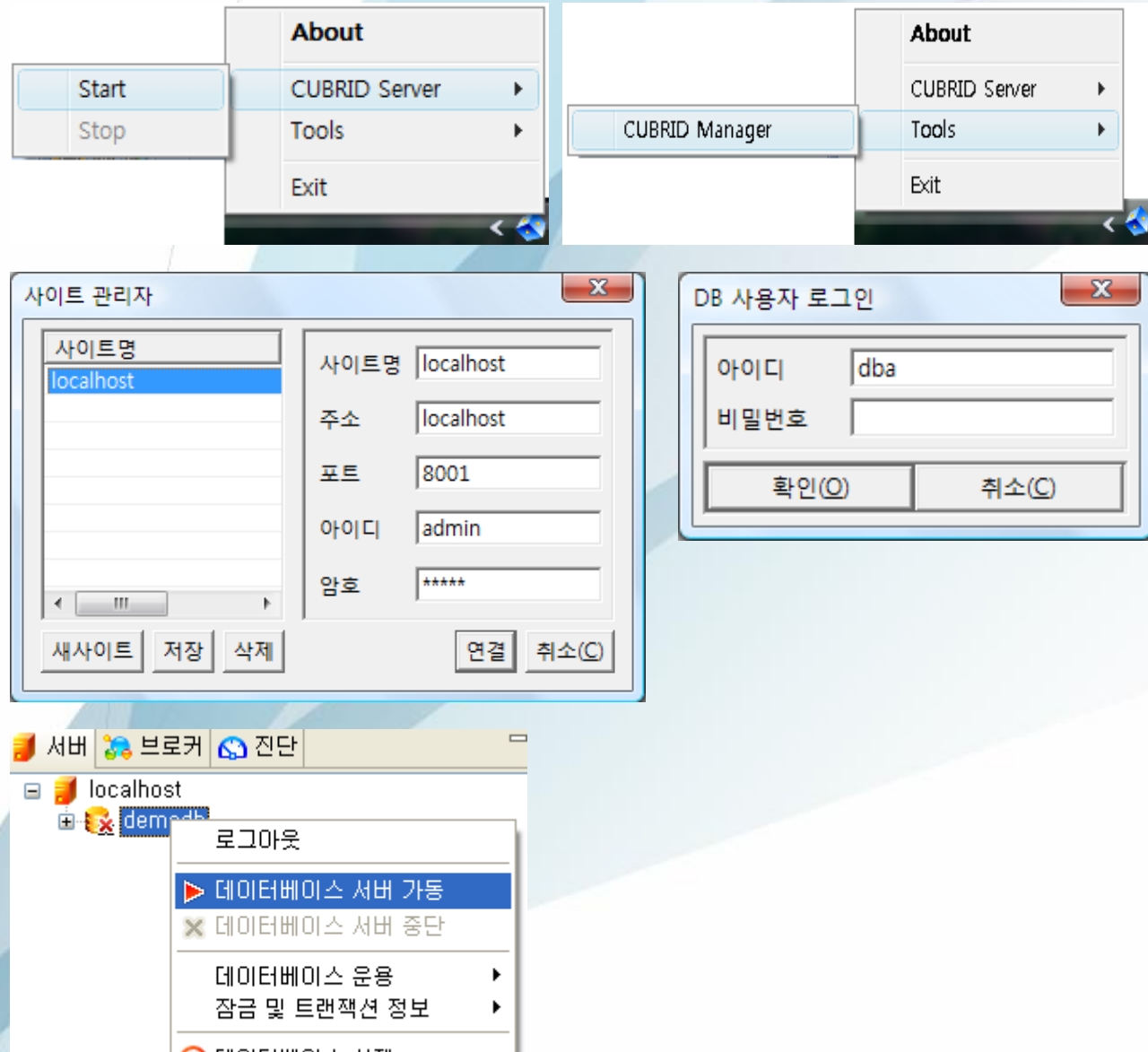
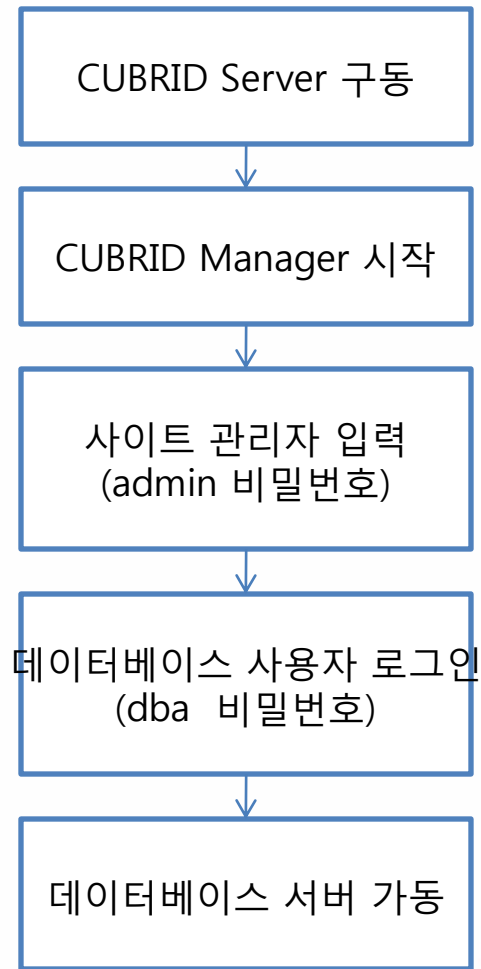
The screenshot shows the CUBRID Manager application window. The interface is divided into several sections:

- 트리창 (Tree View):** Located on the left, it displays a hierarchical tree of the database structure, including 'localhost', 'demodb', '사용자' (users), 'dba', 'public', '작업 자동화' (job automation), '백업 자동화' (backup automation), '질의 자동화' (query automation), '저장 공간' (storage space), '볼륨' (volumes), '활성 로그' (active logs), '보관 로그' (archive logs), '스키마' (schemas), '시스템 스키마' (system schemas), '사용자 스키마' (user schemas), '클래스' (classes), 'athlete', 'code', 'event', 'game', 'history', 'nation', 'olympic', 'participant', 'record', 'stadium', '가상 클래스' (virtual classes), '트리거' (triggers), and '로그' (logs).
- 질의 편집기 툴 (Query Editor Tools):** A toolbar at the top right of the main window, used for editing queries.
- 질의 편집창 (Query Editor):** The central area where SQL queries are written. It contains the text: `select * from event`.
- 질의 결과창 (Query Results):** A table displaying the results of the query. It has columns: NO, code, sports, name, gender, and players. The data shows 14 rows of wrestling events.
- 질의 결과 표시창 (Query Results Display):** A yellow box at the bottom left containing the SQL query: `select * from 'event' where rownum between 1 and 5000;`.
- 수행 시간 표시창 (Execution Time Display):** A box at the bottom right showing the execution time: `[0.047 초 , 총 행수 : 422]`.

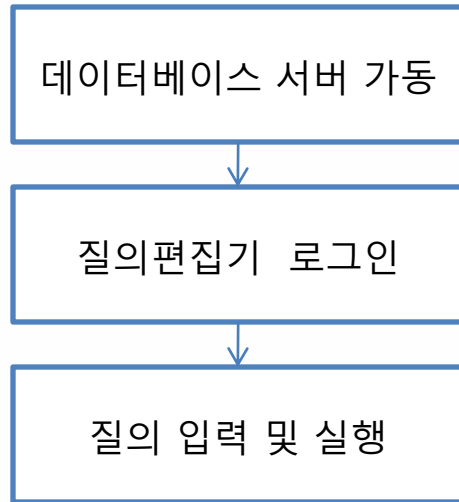
Annotations in red text and arrows point to these specific components: '트리창' points to the tree view, '질의 편집기 툴' points to the toolbar, '질의 편집창' points to the query editor, '질의 결과창' points to the results table, '질의 결과 표시창' points to the query display box, and '수행 시간 표시창' points to the execution time box.

NO	code	sports	name	gender	players
1	20421	Wrestling	Greco-Roman 97kg	M	1
2	20420	Wrestling	Greco-Roman 96kg	M	1
3	20419	Wrestling	Greco-Roman 90kg	M	1
4	20418	Wrestling	Greco-Roman -90 kg	M	1
5	20417	Wrestling	Greco-Roman 85kg	M	1
6	20416	Wrestling	Greco-Roman 84kg	M	1
7	20415	Wrestling	Greco-Roman 82kg	M	1
8	20414	Wrestling	Greco-Roman -82 kg	M	1
9	20413	Wrestling	Greco-Roman 76kg	M	1
10	20412	Wrestling	Greco-Roman 74kg	M	1
11	20411	Wrestling	Greco-Roman -74 kg	M	1
12	20410	Wrestling	Greco-Roman 69kg	M	1
13	20409	Wrestling	Greco-Roman 68kg	M	1
14	20408	Wrestling	Greco-Roman -68 kg	M	1

3.2 CUBRID Manager 시작



3.3 CUBRID Manager에서 쿼리 실행



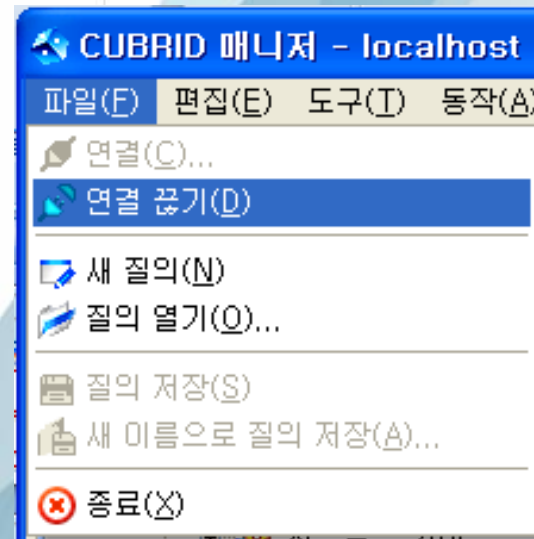
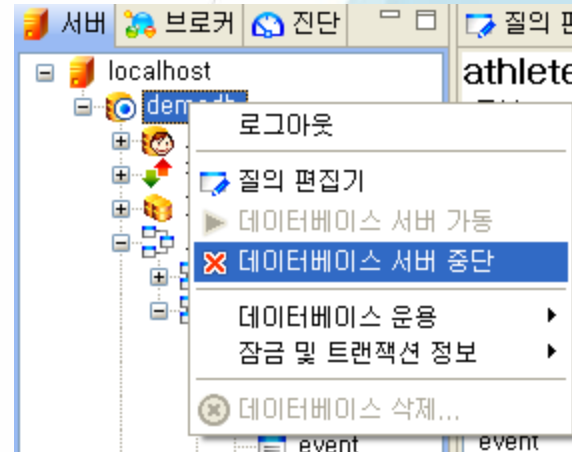
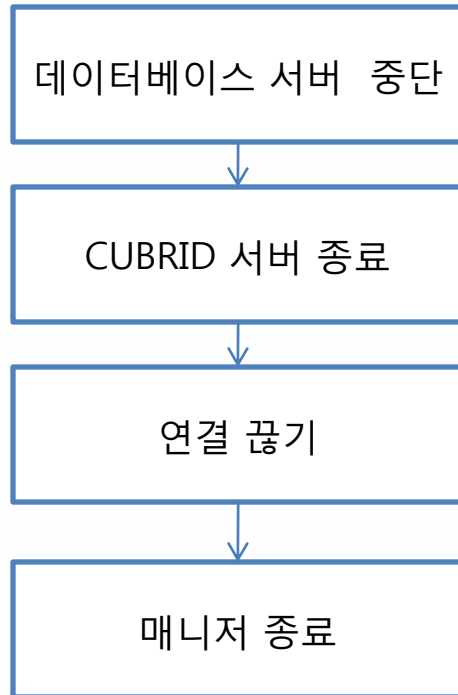
The screenshot shows the CUBRID Manager interface. At the top, there's a menu bar with '서버' (Server), '브로커' (Broker), and '진단' (Diagnosis). Below it, a tree view shows 'localhost' with a sub-item 'demodb'. A context menu is open over 'demodb', listing options: '로그아웃' (Logout), '데이터베이스 서버 가동' (Start Database Server), '데이터베이스 서버 중단' (Stop Database Server), '데이터베이스 운용' (Database Operation), '잠금 및 트랜잭션 정보' (Locks and Transaction Information), and '데이터베이스 삭제...' (Delete Database...). The '데이터베이스 서버 가동' option is highlighted.

To the right, a '데이터베이스 선택' (Select Database) dialog box is open. It lists 'demodb' as the available database. Below the list, there are input fields for '아이디' (ID) with the value 'dba' and '비밀번호' (Password). At the bottom are '확인(O)' (OK) and '취소(C)' (Cancel) buttons.

At the bottom of the screenshot, the 'Query Editor' window is visible. It contains a SQL query: `1 select * from "athlete";`. Below the query, the results are displayed in a table titled '결과1' (Result1).

NO	code	name	gender	nation_code	event
1	10999	Fernandez Jesus	M	ESP	Handba
2	10998	Fernandez Jaime	M	AUS	Rowing
3	10997	Fernandez Isabel	W	ESP	Judo
4	10996	Fernandez Gigi	W	USA	Tennis
5	10995	Fernandez Ana lhis	W	CUB	Volleyb.

3.4 CUBRID Manager 종료



4. CSQL 실행

4.1 CSQL 시작

4.2 원격에서 CSQL 실행

4.1 CSQL 시작

- ✓ 소개
 - 콘솔 기반에서 CUBRID SQL문을 실행 및 결과를 조회할 수 있는 유틸리티.
- ✓ CSQL 실행 모드
 - 배치 모드: SQL문이 저장된 파일을 지정하여 SQL문 일괄 수행 가능.
 - 독립 모드: 데이터베이스 서버 프로세스가 구동되어 있지 않아도 SQL문 실행 가능.
 - 클라이언트/서버 모드: CSQL이 클라이언트로서 데이터베이스 서버에 접속함.

4.2 CSQL 실행

✓ 로컬호스트에서 실행

```
csql -S demodb  
csql -C demodb  
csql -c "select * from olympic;select * from stadium" demodb
```

✓ 원격호스트에서 실행

```
csql -C demodb@192.168.1.3
```