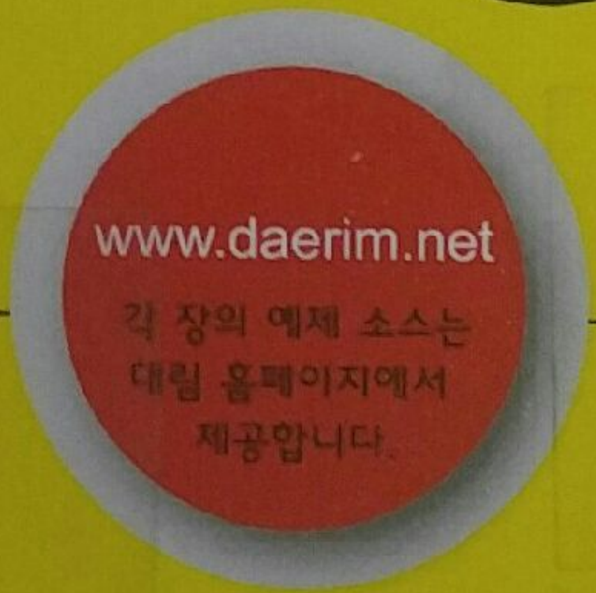
Tomcat + MYSQL + JSP

금성진 · 김인섭 · 최광돈 공저



- JSP를 처음 접하는 초보자를 위하여 **쉬운 예제**로 설명했습니다.
- Tomcat, MySQL과 JSP의 연동을 통해 **실무 웹 프로그래밍 환경을 구성**하였습니다.
- 실무에 사용 가능한 예제를 **단계별로 기능을 추가하는 방식**으로 설명하였습니다.
- **따라하기식 설명**으로 이해가 쉽도록 설명합니다.

도서출판 대림
www.daerim.net

클릭하세요 Tomcat+MySQL+JSP

초판 1쇄 인쇄 / 2007년 5월 15일
초판 1쇄 발행 / 2007년 5월 25일

지은이 / 금성진, 김인섭, 최광돈 공저
발행인 / 이수길

발행처 / 도서출판 대림
등록일자 / 1977년 3월 5일
등록번호 / 제 10-60호

주소 / 서울 마포구 서교동 404-26 팜파스 빌딩 2F
전화 / 02-335-0652
팩스 / 02-335-2917
홈페이지 / www.daerim.net

기획, 진행 / 박기덕
편집 디자인 / 조서봉

정가 22,000원

ISBN / 978-89-7280-769-8

잘못된 책은 구입처에서 교환해 드립니다.

Copyright 2007 by daerim publishing Co. All reserved.

이 책의 일부 혹은 전체 내용을 무단 복사, 복제, 전재하는 것은 저작권법에 저촉됩니다.
본문에 포함된 모든 프로그램은 각 개발자(개발사)에 의해 그 권리를 보호받고 있습니다.

책을 쓰면서...

이 책의 집필이 왜 필요할까?라는 생각을 해 본다. 예전에 초급자였던 필자는 프로그램에 대한 목마름을 해소할 길이 없어 부단히도 웹 사이트를 찾아다니며 기초를 배웠고 실무에 투입되어 열심히 일한 덕분에 그래도 주어진 일에 책임감을 가지고 개발에 참여했다는 소리는 들었다. 허나 필자는 아직도 배우고 있는 중이다. 아직 개발자 아니 소프트웨어 엔지니어로서의 인생이 좀 남아있다고 생각하고 있기 때문일 것이다.

이 책의 주요 독자는 프로그램을 처음 접하는 일반인 또는 학교에서 컴퓨터 학과의 학생들일 것이다. 처음 필자가 겪었던 배고픔과 목마름에 허덕이지 않게 하기 위해서, 같이 일하는 신입사원들에게 어떻게 하면 조금 더 쉽게 웹 개발에 접근할 수 있을까? 어떻게 하면 개발에 대한 친근감을 높일 수 있을까? 등등의 고민이 이 책의 집필 계기가 된 것 같다.

TMJ라는 용어는 신용어가 아니다. 이미 알고 있는 사실을 TMJ(Tomcat, MySql, Jsp)라는 말로 표현한 것뿐이다. TMJ가 가지는 좋은 이점은 이로 말할 수 없다. 이러한 이점들이 예전에는 특화된 기술인 줄 알고 중소기업 또는 기타 IT 쪽에서는 AMP(아파치+mysql+PHP) 등에 치중을 했었다.

하지만 처음에는 그랬을지는 모르지만 이제는 다르다는 것을 알고 있어야 한다. 하루가 다르게 IT 기술이 발전하고 그에 따른 소프트웨어 엔지니어 또한 발전해 나감으로서 개인, 회사가 모두 발전할 수 있는 계기가 될 것이라고 생각하기 때문이다.

이 책의 구성은 프로그래머로서 처음 시작하는 독자를 대상으로 하는 만큼 철저하게 예제를 중심으로 이루진다. 또 앞 예제를 한번만 하고 끝내는 것이 아니라 그것을 응용하고 공통적으로 사용할 수 있는 부분들을 찾아내어 처음과 끝이 연결되는 구조이다. 따라서 이 책을 통해 공부하는 독자는 프로그램 개발에 대한 두려움을 없애고 이것을 발판으로 중급, 고급 소프트웨어 엔지니어로서 거듭 발전함에 작은 도움이 되었으면 하는 바이다.

머릿속에 있는 무엇인가를 끄집어내어 남에게 전달한다는 것은 무척이나 어려운 일이다. 책의 내용 중에 엉성한 부분이 없도록 몇 번이고 보고 또 보았지만 그래도 남아있을지 모르는 부분들을 찾아 본인에게 맞게 고쳐 나가는 방법도 좋은 공부가 되리가 생각한다.

이 책이 만들어지면서 고생하신 도서출판 대림 박기덕 실장님과 편집해주신 조서봉님에게 고마움을 느끼며 디자인을 잠시(?) 도와준 10년지기 친구 강상익 실장, 슈퍼맨 리턴즈에 나오는 주인공과 흡사한 외모와 키를 가진 신정훈 대리에게 고마움을 전한다.

일을 마치고 와서 졸린 눈으로 컴퓨터 앞에서 무엇인가를 할 때 과일과 따스한 차 한잔을 내어준 쌍둥이 엄마이자 세 아이의 엄마인 미란씨에게 머리 숙여 감사와 고마운 마음을 전하고 싶다.

금성진

kumdong@dreamwiz.com

1장 MySql의 설치 그리고 MySQL-Front

왜 Mysql를 사용하는가? 14

Mysql의 설치 14

Msyql 기본설치 15

● 따라하기 -mysql의 설치 15

윈도우 서비스 등록 18

● 따라하기 윈도우 서비스 등록 18

MySQL-Front 20

관리도구의 필요성 20

MySql-Front의 설치 21

● 따라하기 MySQL-Front2.5설치 21

MySQL-Front의 주요기능 24

File 25

Edit 26

Tools 27

Im/Export 28

Tab 메뉴 30

2장 데이터베이스의 이해와 기본 SQL

데이터베이스의 기본 정의 33

데이터베이스 구축의 목적 37

데이터의 일관성 37

데이터의 무결성 37

데이터 중복의 최소화 38

데이터 보안성 38

데이터 독립성 38

데이터베이스 시스템의 구성 .....

기본 SQL 41

SELECT 41

DELETE 43

INSERT 45

UPDATE 46

3장 MySql의 기초 세우기

데이터베이스의 생성 48

- 따라하기 데이터베이스 생성 48

데이터베이스의 삭제 49

- 따라하기 데이터베이스 삭제 49

테이블의 생성 49

- 따라하기 테이블 생성 50

테이블의 삭제 53

- 따라하기 테이블 삭제 53

MySql 데이터 타입 54

문자형 54
숫자형 55
날짜형 56

MySql 테이블 타입 57

MyISAM 57
InnoDB 58

- 따라하기 InnoDB 생성 58
- 따라하기 InnoDB의 트랜잭션 처리 60

4장 MySql의 기본 쿼리문

SELECT 64

INSERT 68

필드를 사용하지 않는 insert 방법 68
Front에서의 직접 입력 69
select를 통한 insert 69

DELETE 70

Front에서의 삭제 71
쿼리를 통한 삭제 71

UPDATE ····· 72
Front에서의 Update ····· 72
쿼리를 통한 update ····· 72

필드의 수정, 삭제 및 속성 수정 ····· 73
테이블과 테이블과의 JOIN과 UNION ····· 74
● 따라하기 사용자 정보 테이블 생성 ····· 75
● 따라하기 물품 정보 테이블 생성 ····· 76
● 따라하기 구매 내역 테이블 생성 ····· 79
JOIN ····· 82
UNION ····· 83

5장 MySql 함수와 백업

MySql 함수 ····· 85
MySql의 백업 ····· 93
Front에서의 백업 ····· 94
● 따라하기 테이블의 파일 백업 ····· 94
Front에서 테이블의 같은 서버 내 데이터베이스로 백업 ····· 97
● 따라하기 테이블을 같은 서버로 백업 ····· 97
Front에서 테이블의 다른 서버 데이터베이스로 백업 ····· 98
● 따라하기 테이블을 다른 서버로 백업 ····· 98
● 따라하기 Refresh하기 ····· 100
Dump 백업 ····· 100
● 따라하기 모든 데이터베이스 백업 ····· 100
● 따라하기 일부 테이블만 백업 ····· 102
● 따라하기 특정 DB의 특정 테이블 백업 ····· 103

백업 데이터 복구 ····· 103
● 따라하기 데이터의 복구 ····· 104

자동 백업 ····· 105

6장 Front의 기타 기능과 ODBC 사용하기

Front의 기타 기능 ····· 107

HTML_VIEW ····· 107
Copy as CSV-data ····· 108
Copy as HTML-table ····· 108
Copy as XML-data ····· 109
Export ····· 109
단축아이콘 ····· 110
이미지 데이터 삽입하기 ····· 111
● 따라하기 이미지 데이터 삽입 ····· 111

ODBC를 이용한 데이터 작업 ····· 113

ODBC란? ····· 113
테스트 테이블의 생성 ····· 113
● 따라하기 ODBC 테스트 테이블 ····· 113
MySql ODBC의 설치 ····· 115
● 따라하기 MySql ODBC 설치 ····· 115
DSN 등록 ····· 116
● 따라하기 DSN 등록 ····· 116
CSV 데이터 생성과 Access에서 ODBC를 이용한 데이터베이스 연결 ····· 118
● 따라하기 액세스에서 ODBC 연결 ····· 119
● 따라하기 테이블 내에 붙여넣기 ····· 122

7장 JDK 1.5 설치와 Tomcat5의 설치

JDK1.5의 설치 ····· 126

● 따라하기 JDK 설치 ····· 126

환경설정 ····· 129

● 따라하기 JDK 환경설정 ····· 130
● 따라하기 JDK 설치 테스트 ····· 132

톰켓(Tomcat) ····· 133

톰켓 설치하기 ····· 134
● 따라하기 톰캣 설치 ····· 134
● 따라하기 JDK와 TOMCAT 연동 확인 ····· 139

8장 TMJ의 시작

TMJ를 사용하자 141

기본 예제 142

입력 값 받기 142

● 따라하기 입력값 받기 142

입력 값 체크 146

● 따라하기 입력값 확인 146

스크립트를 이용한 체크 148

● 따라하기 스크립트를 이용한 체크 148

JSP 내에서의 에러처리 149

● 따라하기 에러처리 149

기본적인 방법 151

메시지 처리방법 151

스크립트 이용하기 153

기본적인 스크립트 함수 153

onClick() 154

focus() 155

onLoad() 155

9장 MySql과 연동하기

MySql과의 연동 157

● 따라하기 mysql과의 연동 예제 158

기본예제-데이터 저장하기 159

● 따라하기 데이터 저장 159

기본예제- 데이터 리스트 163

● 따라하기 데이터 리스트 163

기본예제- 데이터 저장과 리스트를 한 화면에서의 처리 165

● 따라하기 데이터 저장과 리스트를 한 화면에서의 처리하기 165

데이터의 수정 ····· 168
따라하기 데이터의 수정 ····· 168

기본예제- 데이터의 삭제 ····· 173
따라하기 데이터의 삭제 ····· 173

10장 DBCP를 이용한 TMJ

DBCP가 필요한 이유 ····· 176

톰켓에서의 DBCP의 설정 ····· 176
따라하기 DBCP의 설정 ····· 177

기본예제 - DBCP를 이용한 입력 값의 저장 ····· 181
따라하기 DBCP를 이용한 입력 값의 저장 ····· 181

기본예제 -사용자체크 ····· 184
따라하기 사용자체크를 위한 로그인 작성 ····· 184

11장 페이지와 파일관련

페이지 나누기 ····· 192
따라하기 페이지 나누기 ····· 192

파일 업로드와 다운로드 ····· 202
따라하기 파일 업로드 ····· 202
따라하기 파일다운로드 ····· 205

이미지 업로드와 활용예제 ····· 210
따라하기 이미지 업로드 ····· 210
따라하기 이미지 수정 및 삭제 ····· 224

12장 Excel 데이터 데이터베이스 저장과 체크박스를 이용한 리스트 컨트롤

Excel 데이터 저장방법 233

JExcelAPI를 이용한 엑셀 파일의 저장 234

- 따라하기 JExcelAPI를 이용한 엑셀파일의 저장 234

POI를 이용한 엑셀 파일의 저장 239

- 따라하기 POI를 이용한 엑셀파일의 저장 239

JExcelAPI를 이용한 엑셀파일 다운로드 243

- 따라하기 JExcelAPI를 이용한 엑셀파일 다운로드 243

POI를 이용한 엑셀파일 다운로드 247

- 따라하기 POI를 이용한 엑셀파일 다운로드 248

체크박스를 이용한 리스트 컨트롤 251

- 따라하기 체크박스를 이용한 리스트 컨트롤 251

13장 프로젝트 관리프로그램 [관리자]

테이블의 생성 –관리자 모드 259

- 따라하기 테이블의 생성 259

Context의 생성 및 폴더생성 261

로그인 작성 263

- 따라하기 로그인 만들기 263

세션 저장하기 268

- 따라하기 세션저장하기 268

프로젝트 리스트 만들기 271

- 따라하기 프로젝트 리스트 271

프로젝트 생성 274

- 따라하기 프로젝트 입력화면 274

프로젝트 상세 ····· 278
- 따라하기 프로젝트 상세보기 ····· 278

게시판 생성하기 ····· 292
- 따라하기 게시판 테이블 생성 ····· 292

로그아웃 ····· 294
- 따라하기 로그아웃 ····· 295

14장 프로젝트 관리프로그램 [사용자]

신규 등록 ····· 296
- 따라하기 사용자 정보 입력 ····· 296

프로젝트 권한 확인 ····· 302
- 따라하기 체크수정 ····· 302

게시판 리스트 ····· 303
- 따라하기 게시판화면 출력 ····· 303

게시판 글쓰기 ····· 307
- 따라하기 게시판 글쓰기 ····· 307

상세보기 ····· 312
- 따라하기 상세보기 ····· 312

수정과 삭제 ····· 321
- 따라하기 게시판 수정,삭제 ····· 321

15장 Nonage 쇼핑몰-관리자 및 담당자

테이블의 설계 ····· 329
- worker – 쇼핑몰 담당자 등록 ····· 330
- compe –권한관련 테이블 ····· 330
- products –상품관련정보 ····· 330

orders –주문관련정보 ····· 331
order_result –주문입금, 배송, 반품정보 ····· 331
qna –Q&A테이블 ····· 331
programList – 권한 관리를 위한 프로그램 리스트 ····· 332

데이터베이스와 테이블 생성 ····· 332
- 따라하기 데이터베이스와 테이블의 생성 ····· 332

관리자 ····· 334
- 따라하기 관리자 로그인 만들기 ····· 334
- 따라하기 권한을 주기 위한 competence_frm.jsp 작성 ····· 341

메뉴 등록 ····· 346
- 따라하기 메뉴등록 ····· 346

담당자 등록 ····· 352
- 따라하기 담당자 등록 ····· 352

상품등록 ····· 358
- 따라하기 상품등록 ····· 359

상품상세보기 ····· 368
- 따라하기 상품 상세 보기 ····· 368

상품수정 ····· 372
- 따라하기 상품수정 ····· 372

주문리스트 ····· 383
배송 리스트 ····· 391
- 따라하기 배송 리스트 ····· 391

매출 리스트 ····· 399
회원 리스트 ····· 405
- 따라하기 회원 리스트 ····· 405

게시판(qna) ····· 408
- 따라하기 q&a게시판 ····· 408

공지사항 ····· 415
- 따라하기 공지사항 작성 ····· 416

16장 Nonage 쇼핑몰

content의 추가 ····· 428

index.jsp 작성 ····· 429

- 따라하기 index.jsp 작성 ····· 429

제품상세 ····· 439

- 따라하기 제품상세 작성 ····· 439

메뉴별 페이지 ····· 449

- 따라하기 메뉴별 페이지 ····· 449

로그인 및 회원가입 ····· 456

- 따라하기 로그인 및 회원가입 ····· 456

장바구니 담기 및 주문 ····· 472

- 따라하기 테이블의 생성 ····· 472

장바구니 ····· 477

- 따라하기 장바구니 ····· 477

마이페이지 ····· 491

- 따라하기 마이페이지 ····· 491

배송 조회 ····· 499

- 따라하기 배송 조회 ····· 499

q&a ····· 507

- 따라하기 q&a ····· 507

기타작업 ····· 529

최종 테스트 ····· 532

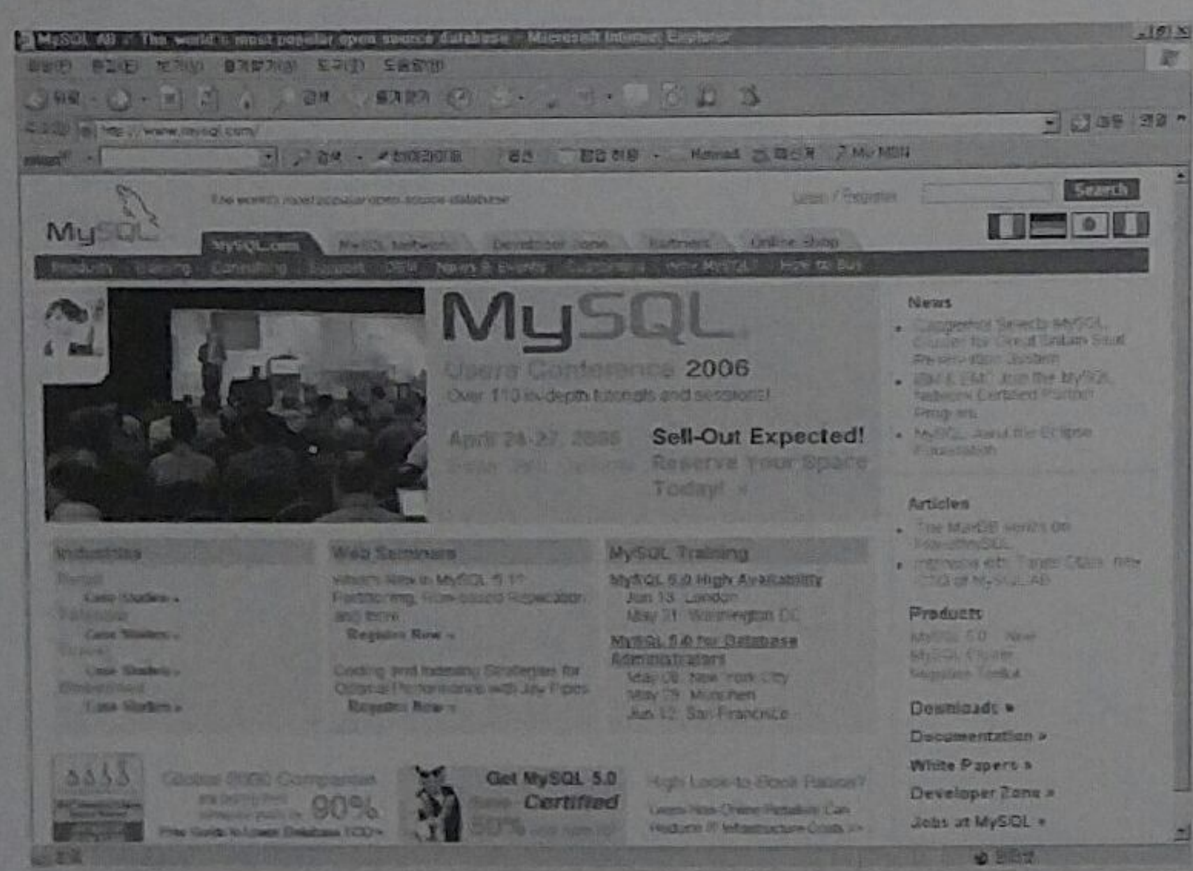
01 MySql의 설치 그리고 MySQL-Front

안녕하세요

이번 장에서는 이 책의 가장 기본이 되는 사항 중 mysql에 대한 기본과 설치 방법을 설명한다. 물론 웹 서핑을 통해 검색하면 더 많은 설치 방법에 대해서 많은 지식을 얻을 수 있다. 이번 장에서는 중복되는 설치 방법에 대한 설명대신 윈도우에서의 설치만을 다룬다. 리눅스에서의 설치 방법 등 기타 운영체계에서의 설치 방법은 부록을 참고하기 바란다. 또한 MySql를 효과적으로 관리하고 편리하게 사용하기 위해 MySQL-Front를 설치하고 MySQL-Front에 대한 전반적인 기능에 대해서 공부한다.

왜 Mysql를 사용하는가?

오라클/사이베이스 등등 많은 상용 데이터베이스가 있다. 이 상용화된 데이터베이스는 많은 기업들이 사용하여 그 사용에 따른 검증된 제품으로 많은 기업들이 데이터베이스에 대한 안정성과 신뢰성을 가지기 위하여 선호한다. 그러나 일부 기업 등에 해당되는 사항이고 중소기업 또는 개인 회사에서는 고가의 제품을 구매할 수 있는 능력이 모두 있다고 할 수 없다.



▶ www.mysql.com

이러한 상황에서의 Mysql는 가난한 IT 기업에게 있어서는 없어서는 안 될 중요한 위치에 있다. 그럼 왜 Mysql를 사용할까? 그 이유는 대략 다음의 몇 가지로 요약할 수 있다.

1. 상용 데이터베이스에 비해 기능은 작지만 상당히 빠른 쿼리 속도를 가진다.
2. 초보자도 쉽게 따라할 수 있다.
3. 다양한 언어와 연동한다.
4. 무료이다.

이외 몇 가지가 더 있지만 다양한 언어(JSP/ PHP/ ASP/ NET/ C#)와 연동할 수 있다는 장점과 제일 중요한 건 무료라는 것이다. 물론 기술적인 부분에 대해서는 비용을 부담해야 되겠지만 구입비용에 대한 부분은 걱정이 없다는 것이다. 그렇다고 공짜라고 해서 그 성능이 떨어지거나 많은 제약을 받지는 않는다. 꾸준한 버전업과 개발을 통해 더욱 더 향상되고 있다.

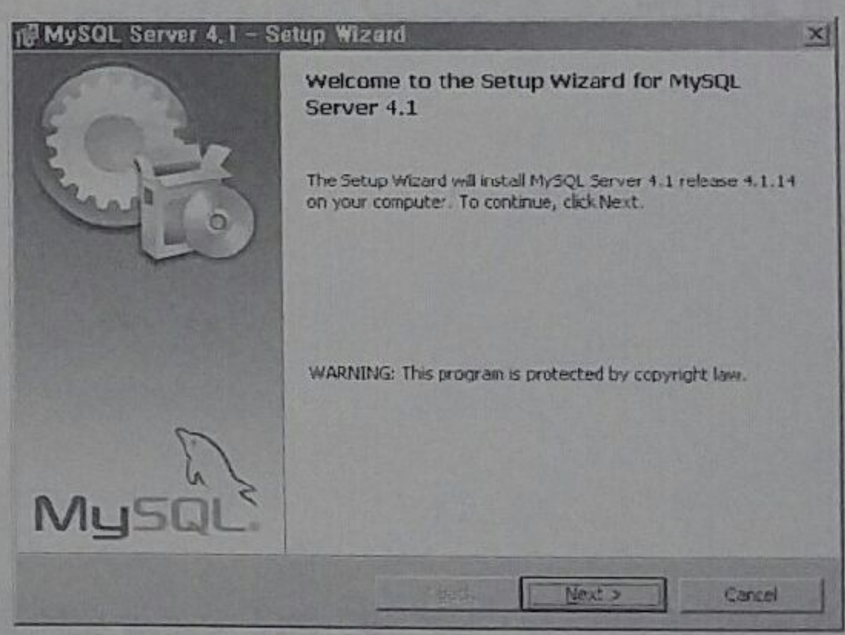
Mysql의 설치

www.mysql.com에 접속하여 다운을 받아도 되며 도서출판 대림 홈페이지 자료실에 있는 Mysql를 설치해도 무방하다. 이 책에서 mysql 4.1 버전에 대해 설치하고 공부하게 될 것이다.

Msyql 기본설치

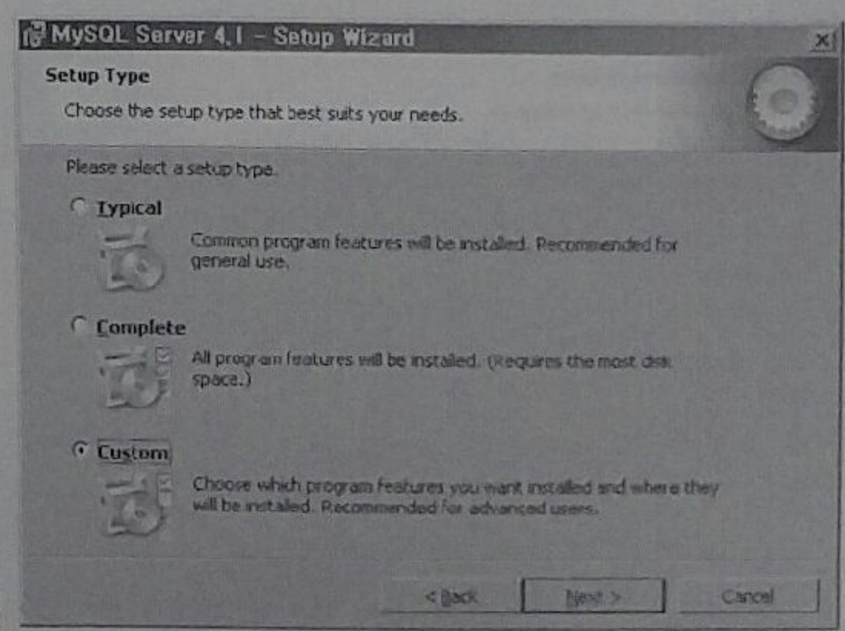
따라하기 -mysql의 설치

❶ 도서출판 대림 홈페이지 자료실에서 이 책의 유틸리티 파일을 다운받아서 그 중에 mysql-4.1.14-win32의 setup.exe 파일을 실행하면 그림과 같이 설치 화면이 보인다.



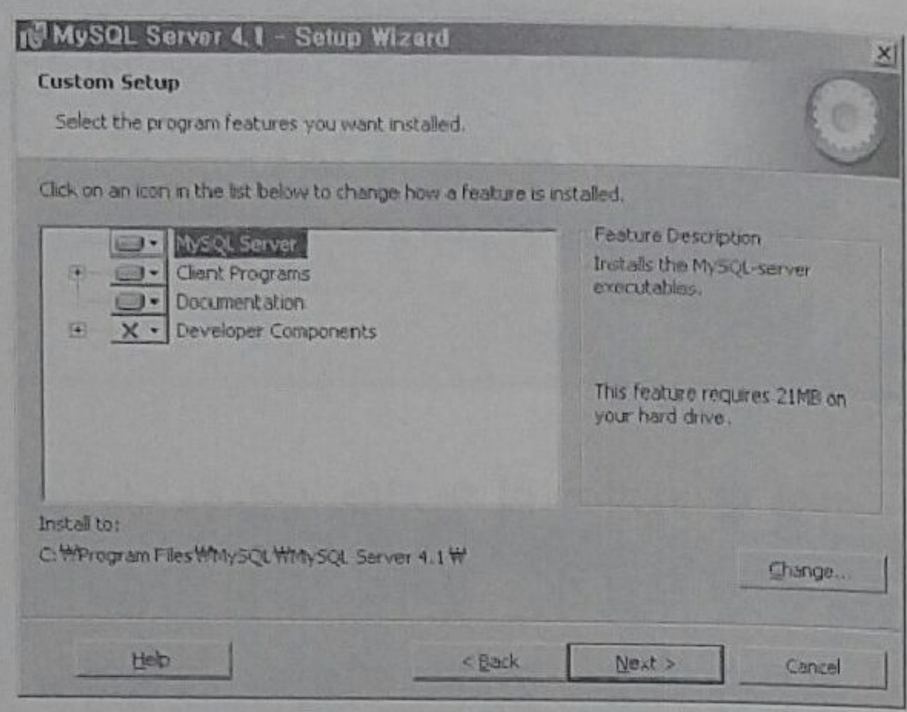
▶ mysql의 설치 실행

❷ next를 클릭 후에 기본설치를 하지 말고 Custom을 선택한다.



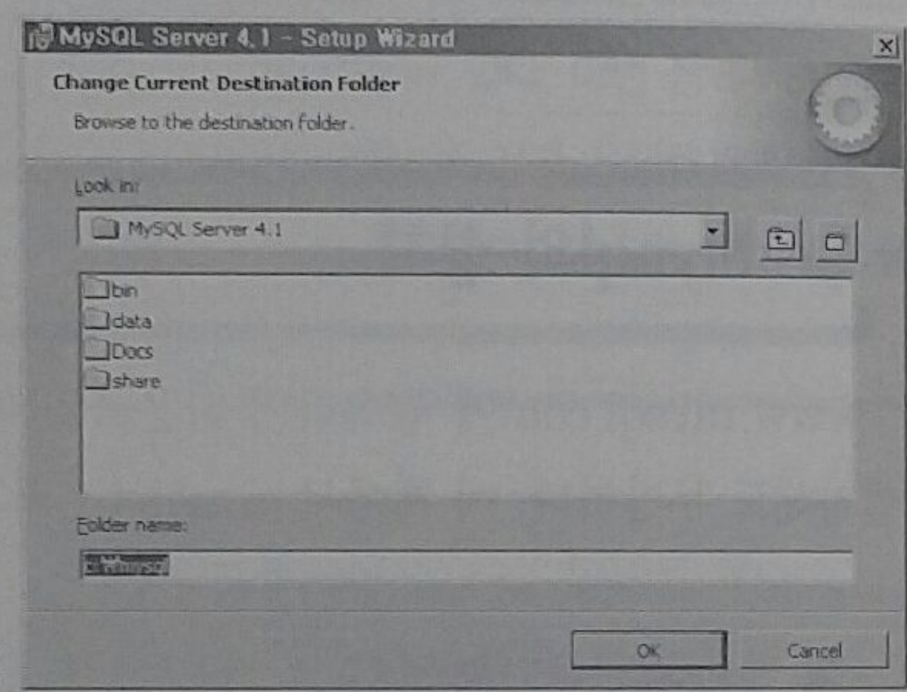
▶ 설치옵션 선택

❸ 설치 시 기본으로 설치를 하면 C:\Program Files 아래에 설치한다. 하지만 우리는 mysql이라는 폴더를 쉽게 알아보고 관리하기 위해 C:\mysql이라는 폴더를 만들어 관리하자.



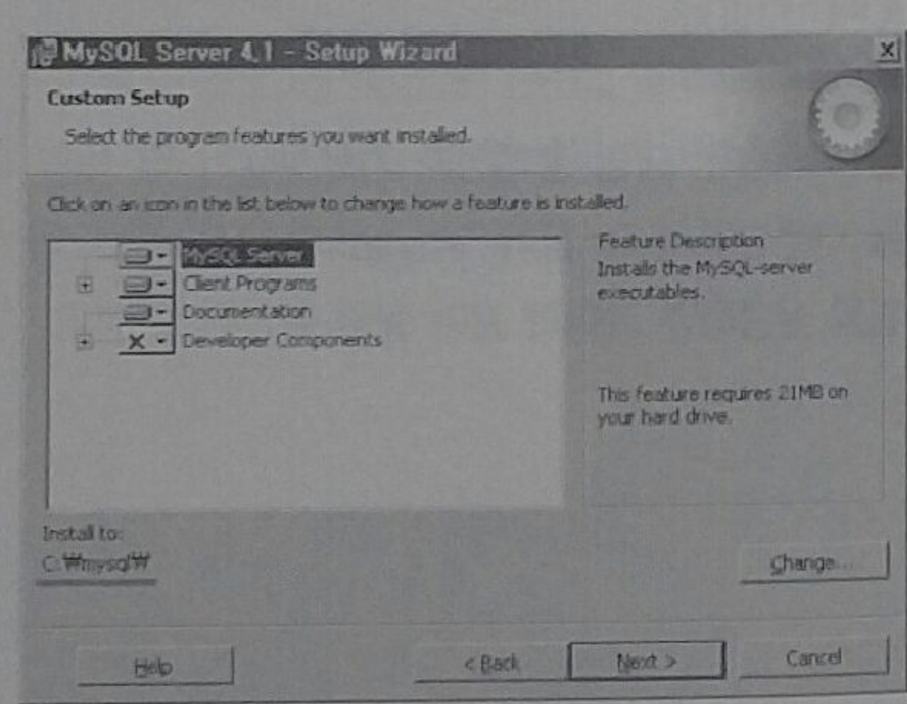
▶ 설치 디렉터리 변경

❹ Change 버튼을 클릭하여 c:\mysql이라는 폴더로 지정하고 ok 버튼을 클릭한다.



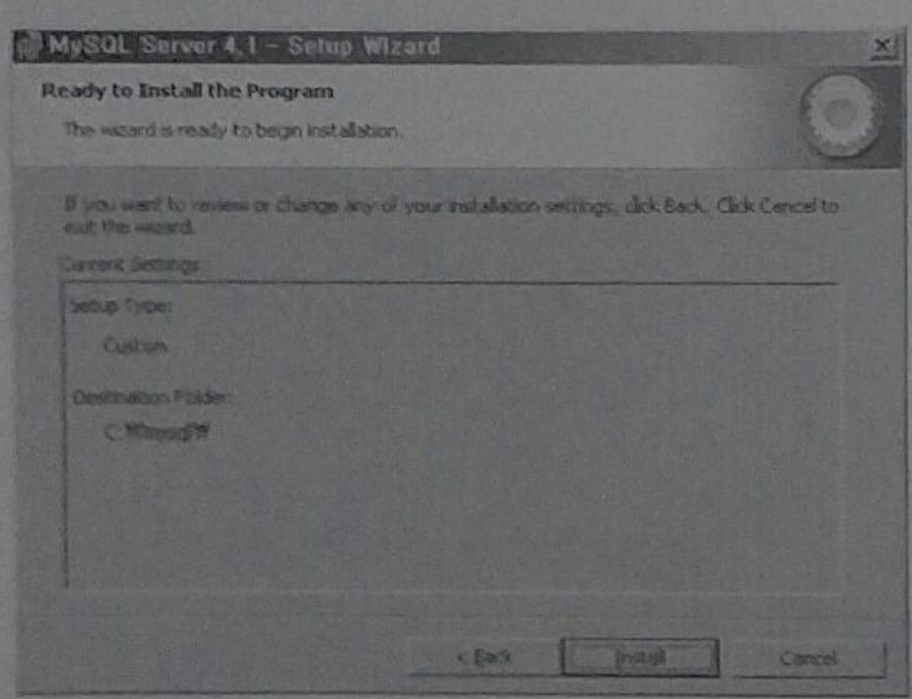
▶ 설치 폴더 입력

❺ 설치 화면에서 설치 폴더가 바뀐 것을 알 수 있다. next 버튼을 클릭한다.

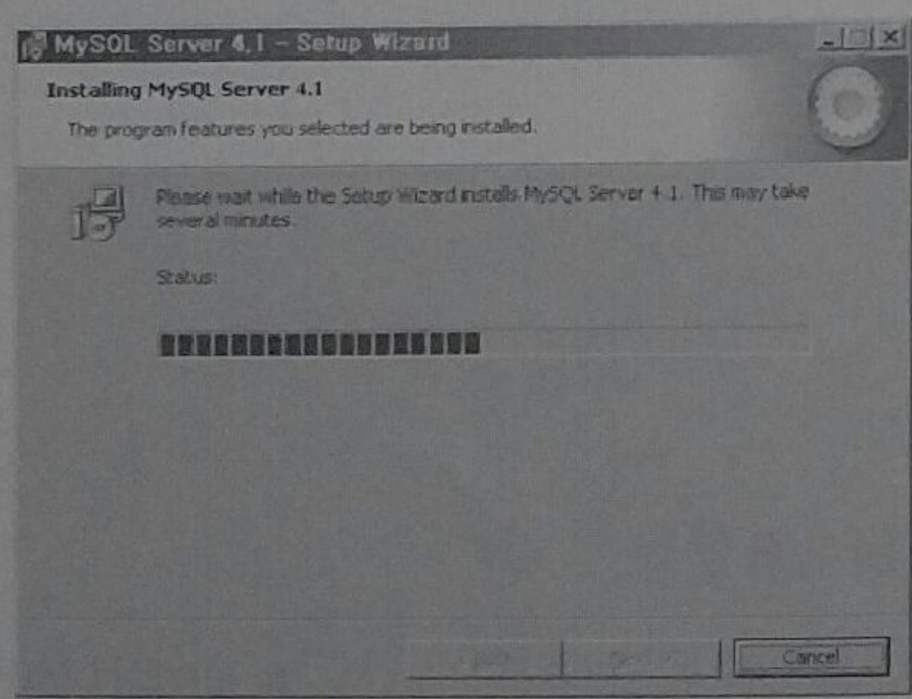


▶ 폴더 변경 확인

❻ 설치 폴더를 다시 한번 확인한다. install 버튼을 클릭한다.

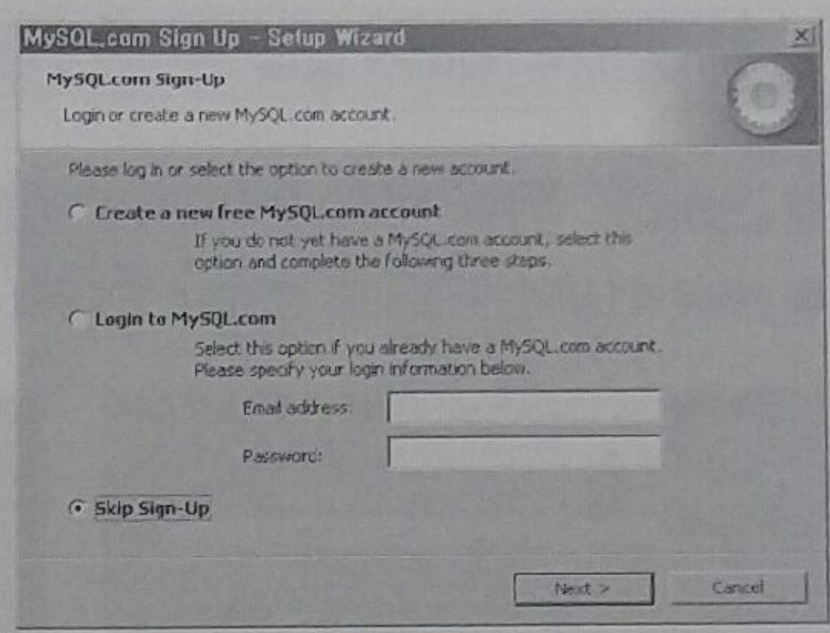


▶ mysql의 install



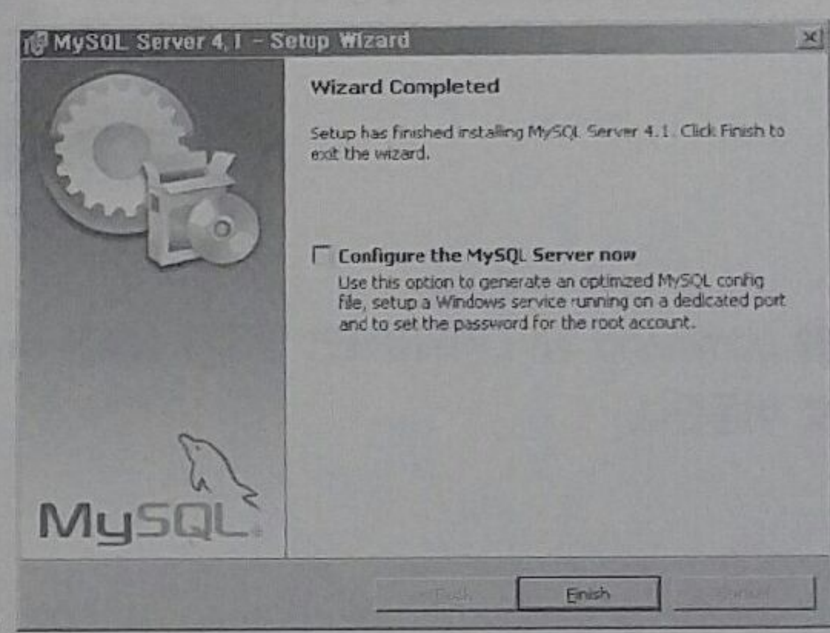
▶ install 과정

❼ 설치가 되면 그림과 같이 Mysql login 정보를 물어본다. 우리는 그냥 skip Sign-up을 선택하고 next 버튼을 클릭한다.



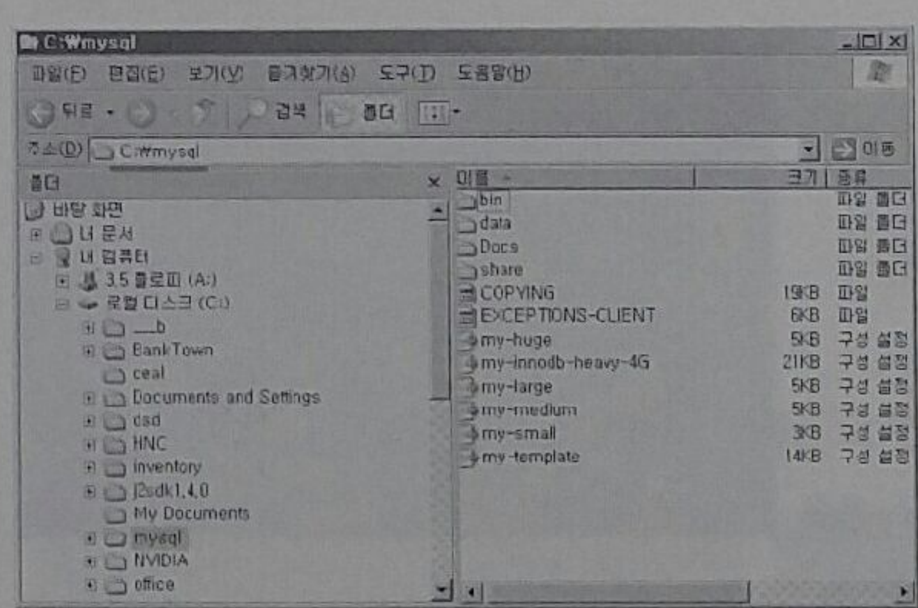
▶ skip Sing-up의 선택

❽ mysql의 Configure를 지금 할 것인지 물어본다. 지금 설정할 필요성이 없기 때문에 체크된 부분을 풀고 Finish 버튼을 클릭하여 설치를 마무리한다.



▶ 설치 종료

이로서 mysql의 install 부분이 끝났다. 탐색기를 열어 보면 c:\mysql이 설치된 것을 볼 수 있다.



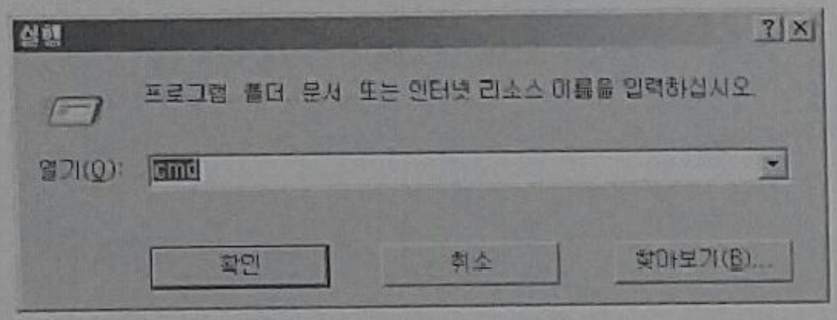
▶ mysql 설치 확인

그러나 설치가 완료되었다고 해서 사용할 수 있는 것은 아니다. mysql의 설치 후에는 반드시 mysql을 구동시켜 주어야 한다. 즉 mysql를 윈도우에서 서비스 할 수 있게 등록시켜주어야 한다.

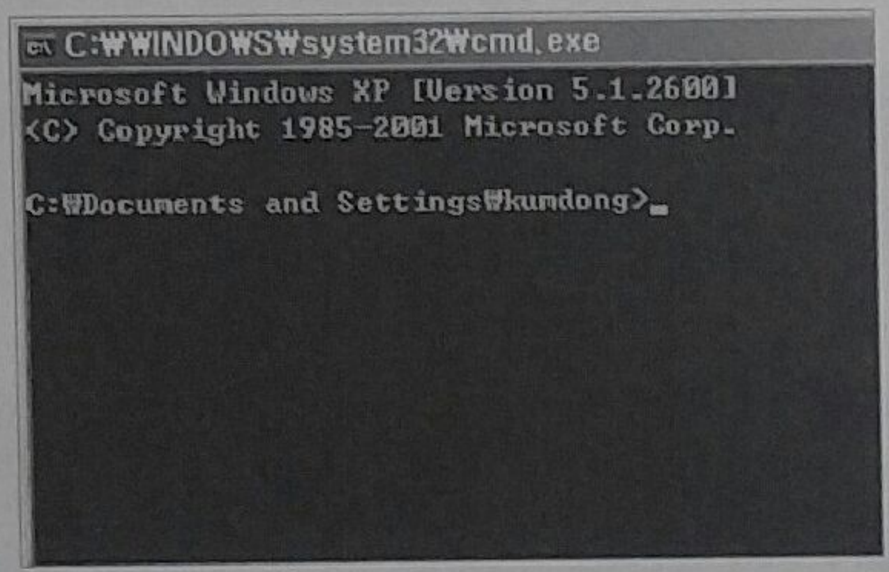
윈도우 서비스 등록

따라하기 윈도우 서비스 등록

❶ 윈도우의 [시작] → [설치]를 선택하여 윈도우의 command 창을 띄우기 위해 cmd를 입력한다.

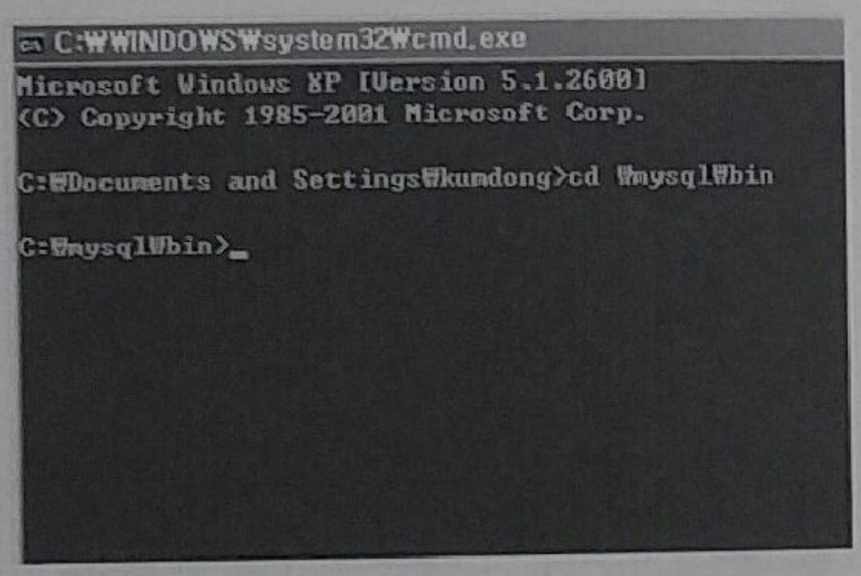


▶ 윈도우에서 command 실행하기



▶ command의 실행

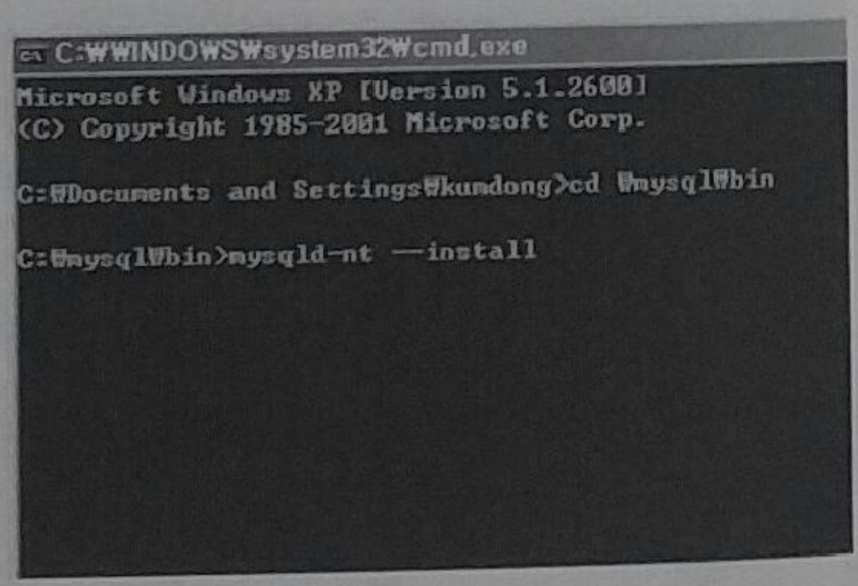
❷ command 창이 띄워졌으면 앞에서 설치된 mysql의 bin 폴더로 이동한다.



▶ mysql로 폴더 이동

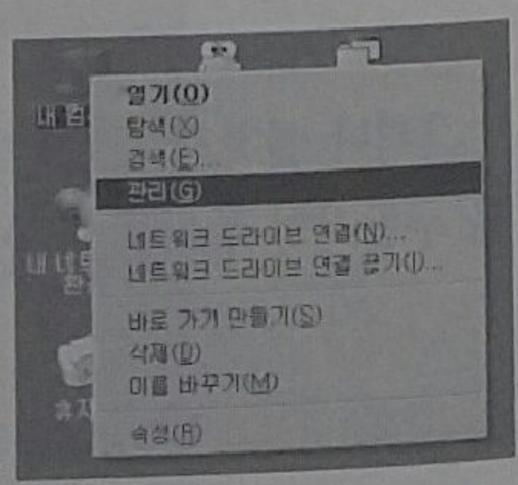
❸ 이동된 폴더에서 윈도우 서비스를 등록하기 위해 다음과 같이 입력을 한다.

```
c:\mysql\bin\mysqld-nt --install
```



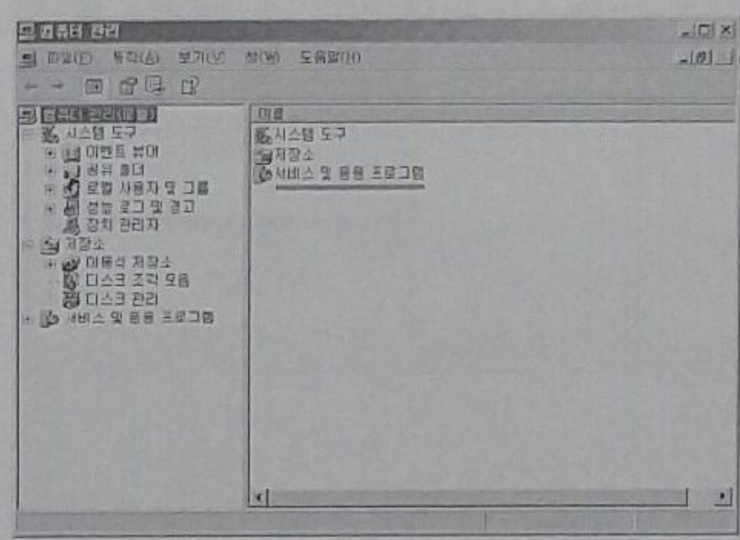
▶ 윈도우 서비스 등록

❹ 입력 후 엔터를 치면 새 창이 잠시 보였다가 사라질 것이다. 바탕화면에서 내 컴퓨터의 관리를 선택한다.

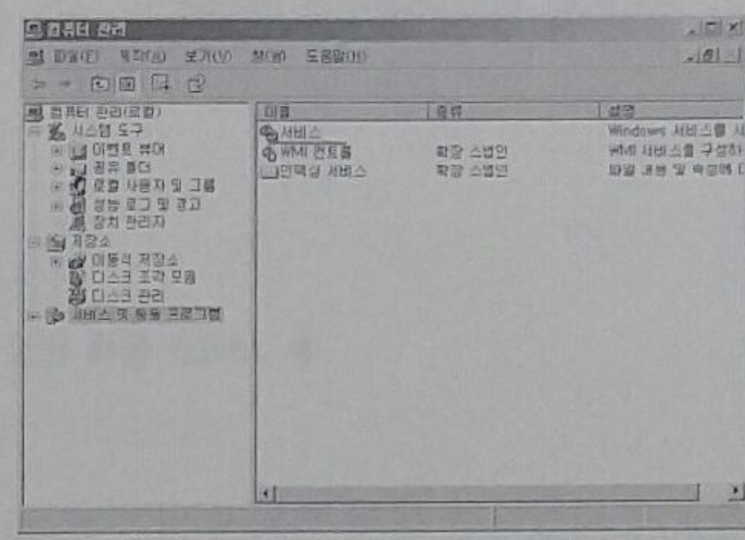


▶ 내 컴퓨터 팝업 메뉴

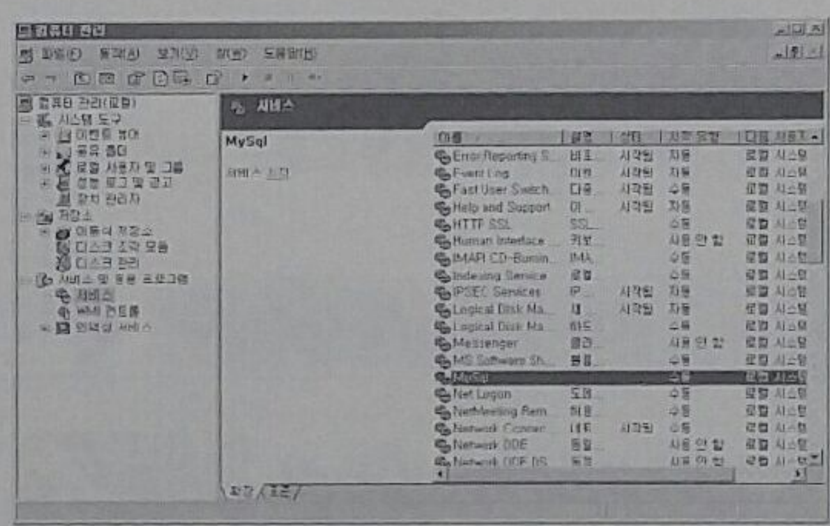
❺ 컴퓨터관리 창 안에 서비스 및 응용프로그램 항목이 있다. 이 항목을 클릭하면 서비스라는 항목이 있으며 이 항목을 클릭하여 보면 mysql이 서비스에 등록되어 있는 것을 볼 수 있다.



▶ 컴퓨터 관리

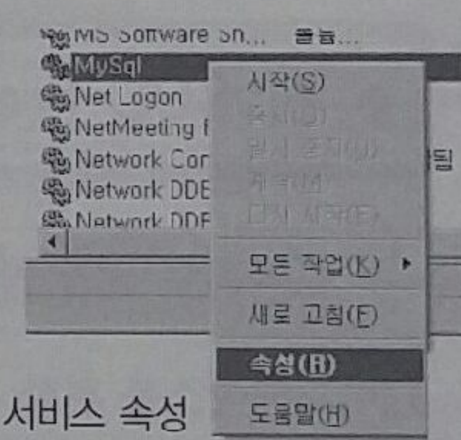


▶ 서비스 및 응용프로그램



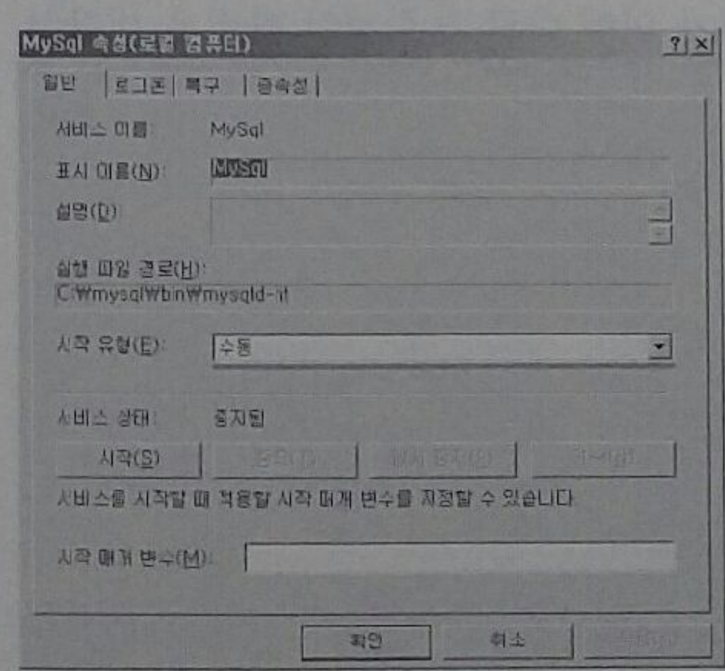
▶ 서비스 등록 확인

❻ msyql의 서비스를 선택한 후에 마우스의 오른쪽 버튼을 클릭하고 팝업 메뉴에서 속성을 선택한다.

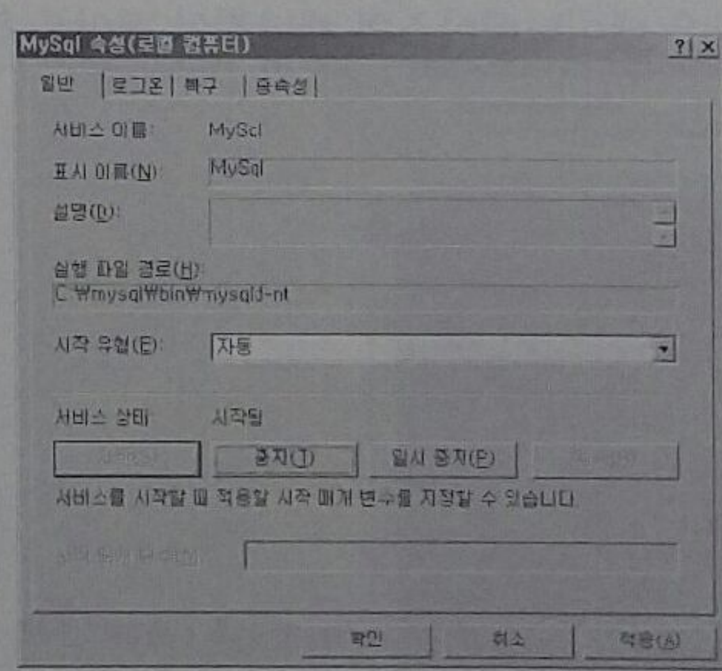


▶ 서비스 속성

❼ msyql 속성에서 시작 유형을 수동에서 자동으로 변경한 후 서비스 상태의 시작 버튼을 클릭한다.

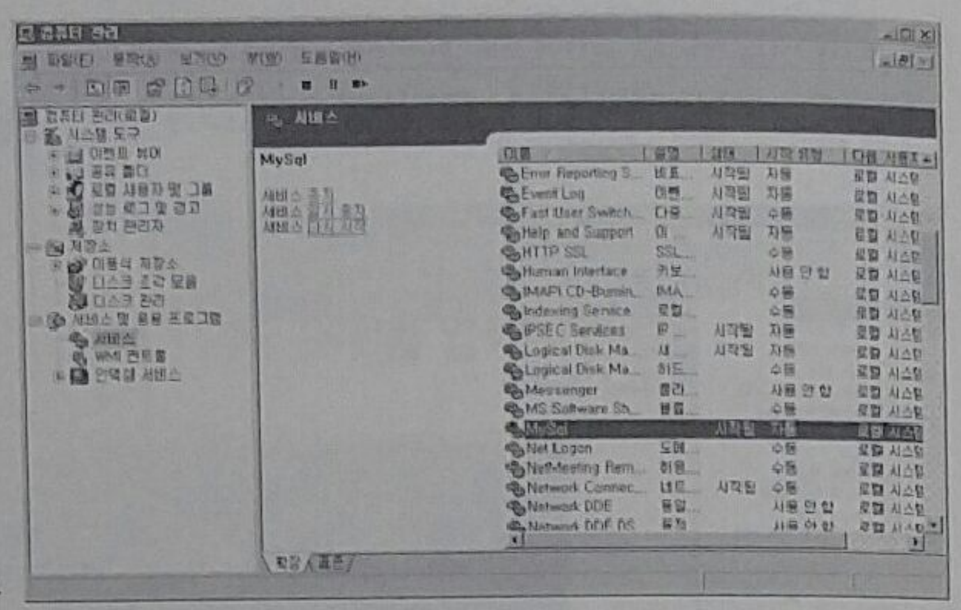


▶ mysql의 서비스 속성



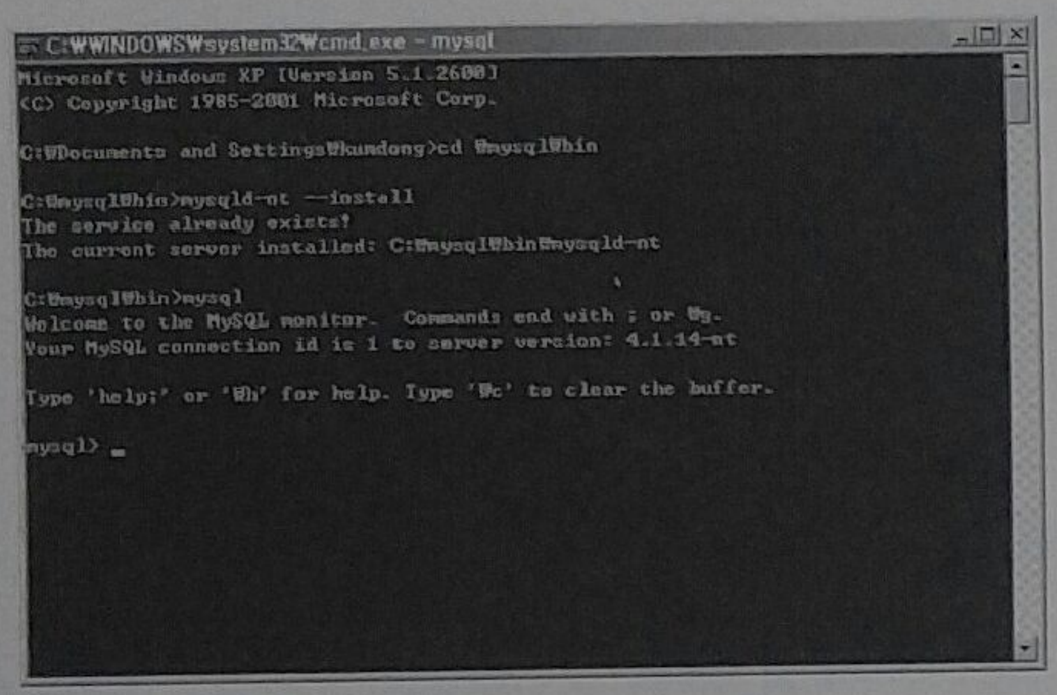
▶ 시작유형의 변경

❽ 적용 버튼을 클릭하면 mysql의 서비스가 수동에서 자동으로 변경이 되며 서비스는 항상 시작되어 있음을 볼 수 있다. 컴퓨터를 재부팅하더라도 msyql은 항상 자동으로 시작된다.



▶ 서비스 등록 완료

서비스 등록이 완료가 되었다면 현재 띄워져 있는 command 창에서 mysql의 접속을 확인한다.

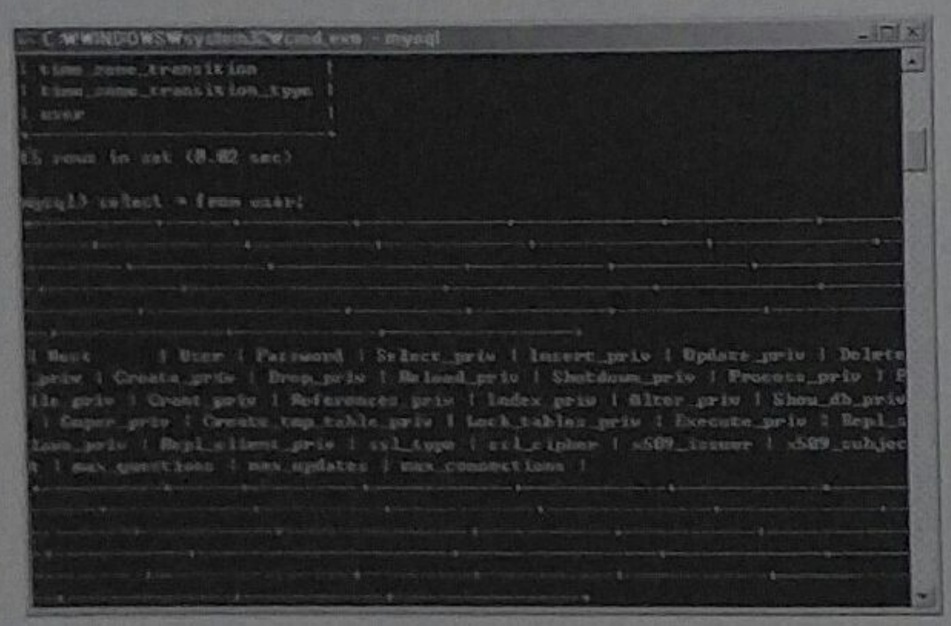


▶ mysql 접속

MySQL-Front

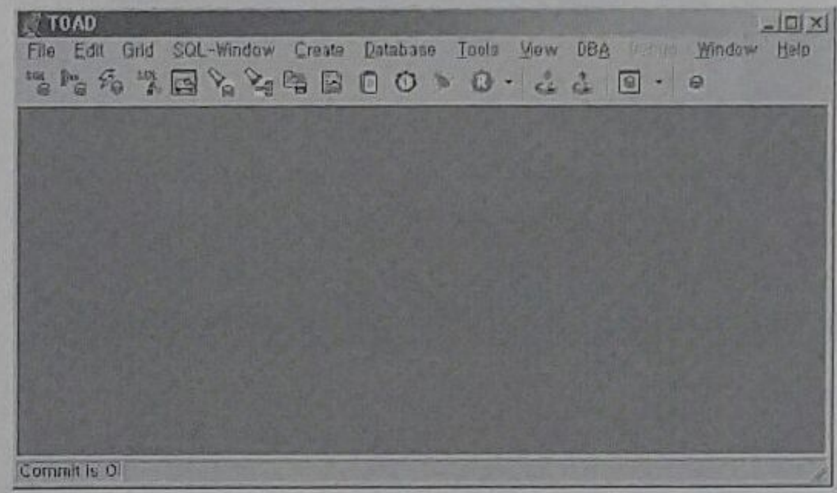
관리도구의 필요성

데이터베이스를 위한 관리도구가 없다면 데이터베이스 책임 관리자(DataBase Administrator)나 프로그램을 개발하는 개발자는 많은 짜증을 낼 것이다. 그 이유는 기본적으로 제공되는 데이터베이스에는 사용자위주의 편리함을 제공하지 않기 때문이다. 그러기에 관리도구가 제공되지 않는다면 사용자는 접속/생성/삭제/쿼리 등과 같은 데이터베이스의 제반적인 제어를 하기 위해 다른 부수적인 방법을 생각하게 된다.



▶ 데이터베이스에서 실행한 쿼리와 결과

한눈에 봐도 답답함을 느끼게 된다. 따라서 이를 사용자가 편리하고 관리할 수 있는 관리도구가 만들어지게 되었다. 관리 도구는 단순한 쿼리의 수행뿐만 아니라 데이터베이스의 생성과 삭제 테이블의 생성과 삭제 등등 데이터베이스가 제공하는 모든 기능을 편리하게 관리할 수 있다.



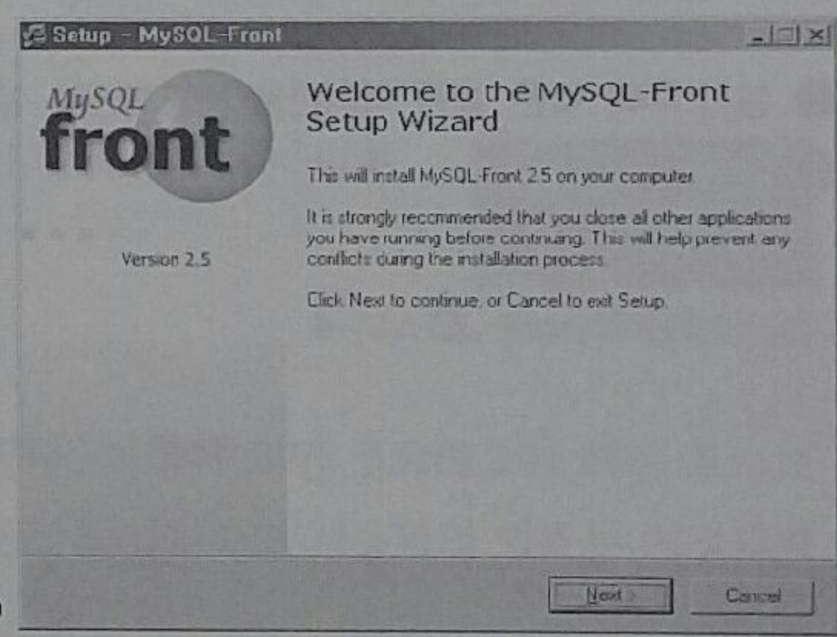
▶ 오라클의 관리도구 TOAD

오라클의 관리도구와 마찬가지로 MySql을 사용자가 편리하게 관리하고 작성하기 위해 MySQL-Front라는 프로그램을 설치하여 사용한다.

MySql-Front의 설치

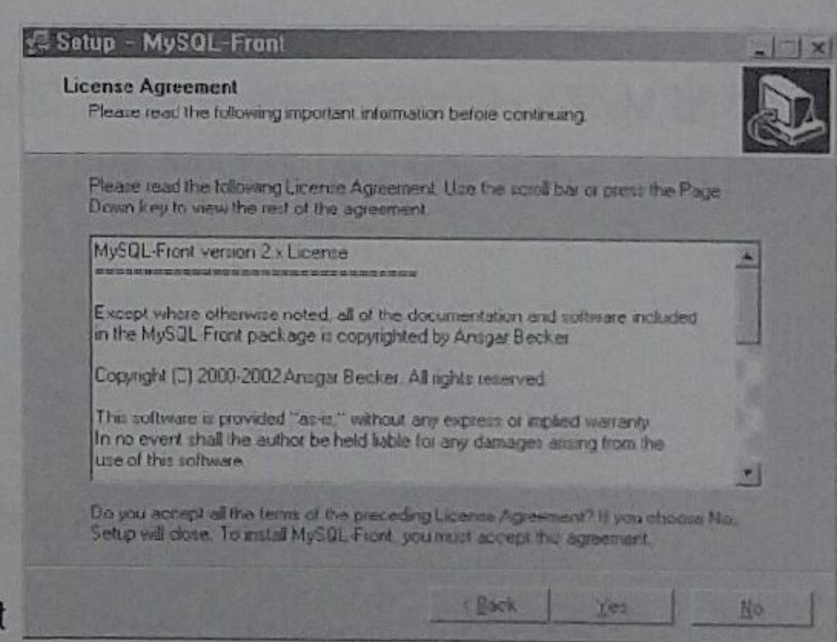
따라하기 MySQL-Front2.5설치

❶ 도서출판 대림 홈페이지 자료실에 있는 MySQL-Front2.5 압축을 다운받아 압축을 해지 한 후 MySQL-Front_2.5_Setup을 클릭하여 설치를 시작한다.



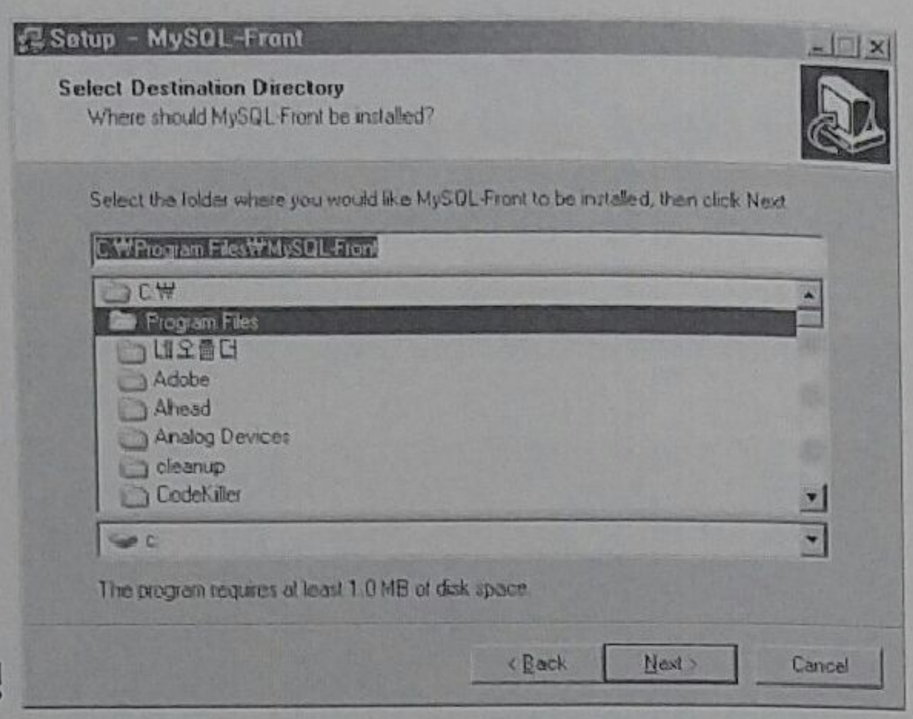
▶ MySQL-Front의 setup

❷ MySQL-Front 2.5에 대한 정보와 소개이므로 Yes를 클릭한다.



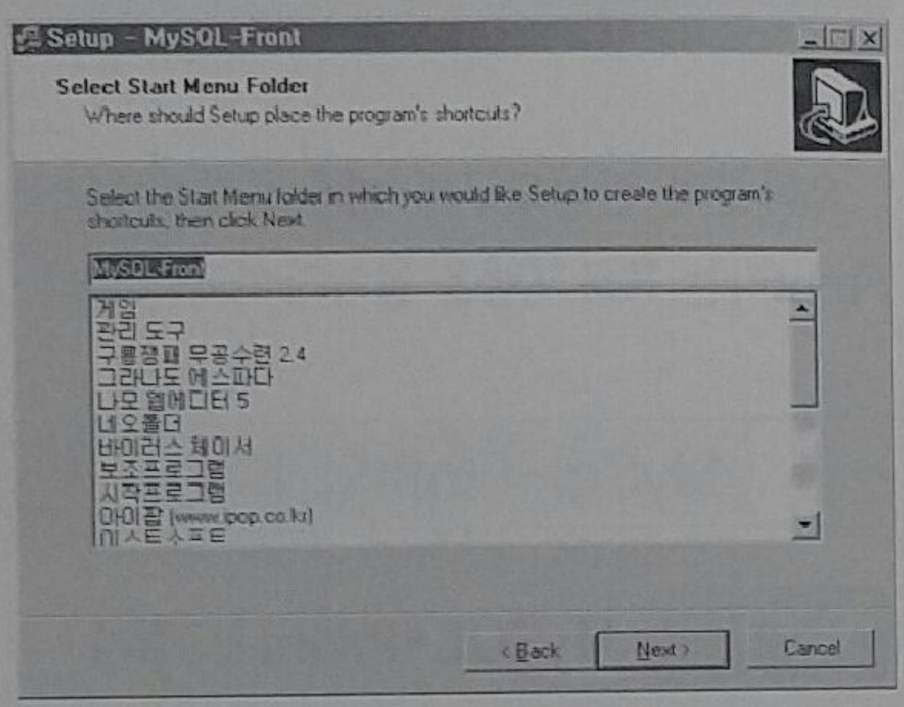
▶ License Agreement

❸ 설치될 mysql의 폴더를 지정한다. 만약 사용자가 다른 폴더로 지정을 하고 싶다면 설치폴더를 임의로 바꾸면 된다.



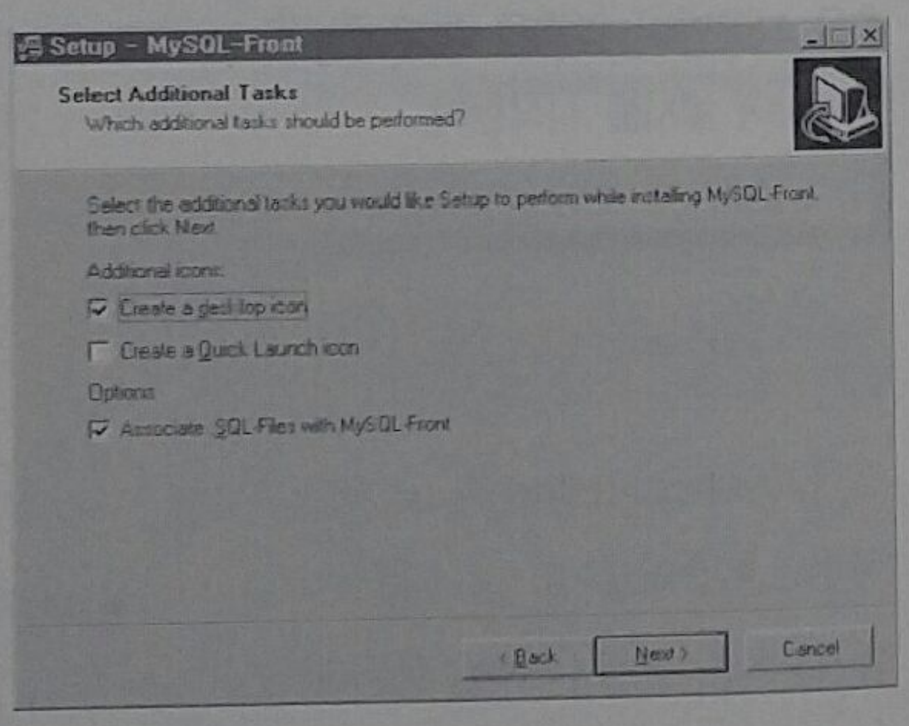
▶ 설치 폴더 지정

❹ 바탕화명과 그리고 윈도우 프로그램으로 등록된 명칭으로 사용자가 임의적으로 다른 명칭을 줄 수 있다.



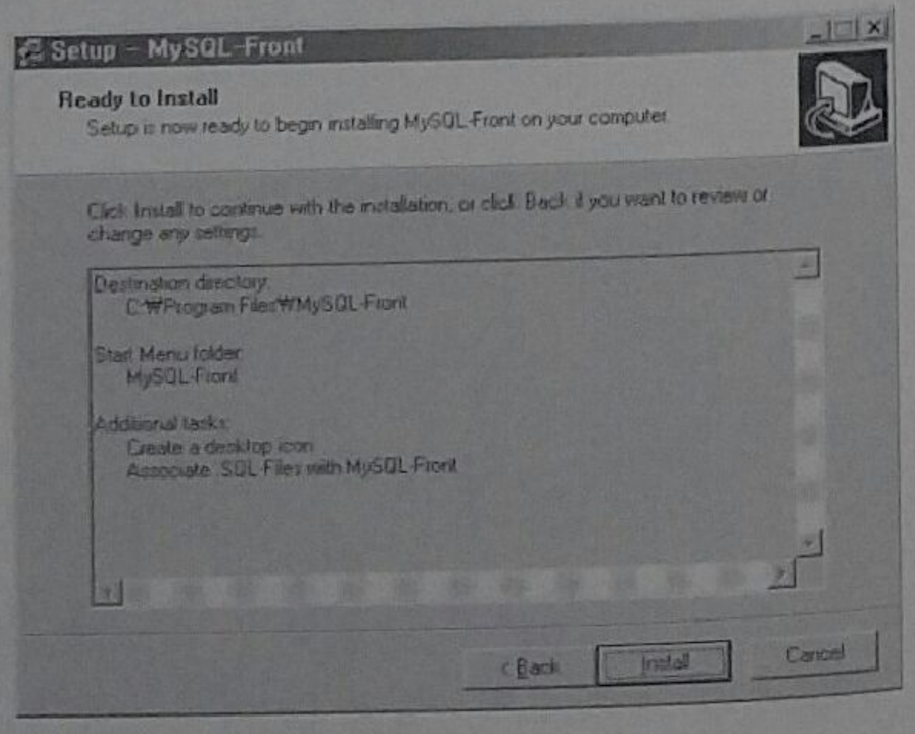
▶ 시작 메뉴의 명칭

❺ 설치 후에 빠른 실행 아이콘을 바탕화면에 설치할 것인가를 묻는다.



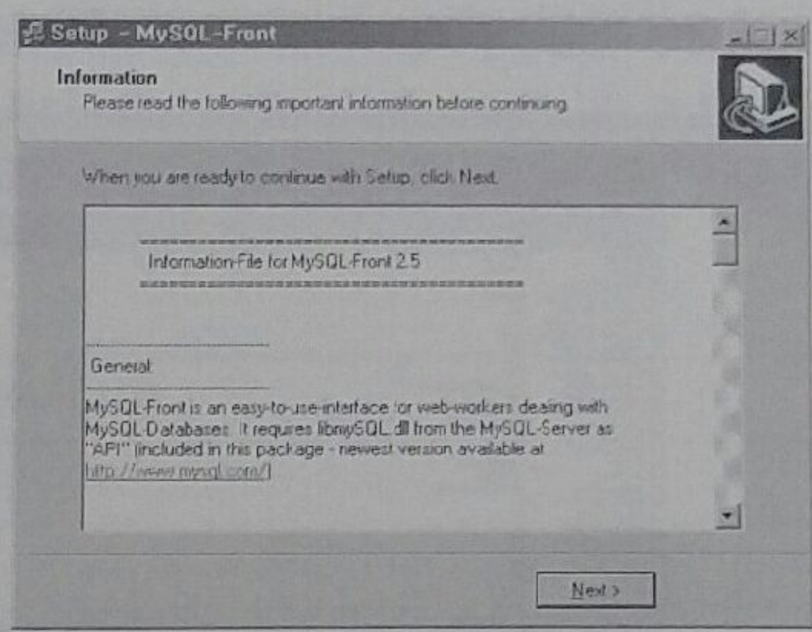
▶ 아이콘의 바탕화면 설치여부

❻ 설치될 MySQL-Front의 정보를 보여준다. 만약 이 정보가 아니라면 back 버튼을 통해 다시 지정하면 된다.



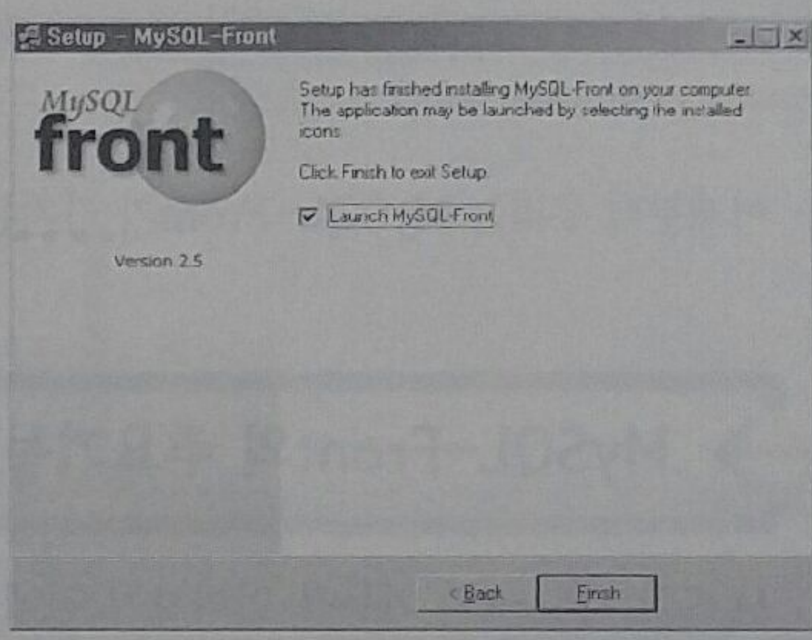
▶ MySql-Front가 설치될 정보

❼ MySql-Front를 사용을 위한 데이터베이스 정보를 보여준다.



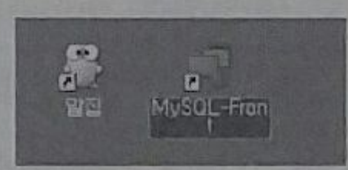
▶ MySql-Front를 위한 정보

❽ 설치의 종료와 함께 시작할 것인가를 체크한다. 만약 체크가 되어 있다면 설치 종료와 함께 MySQL-Front는 시작이 될 것이다. 체크된 부분을 풀고 Finish 버튼을 클릭한다.



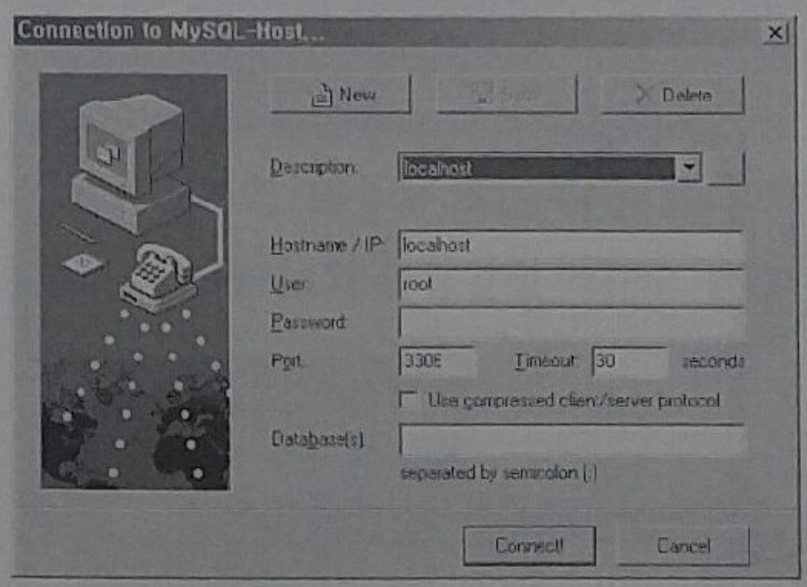
▶ 설치종료

설치가 종료 되고 본인 컴퓨터의 바탕 화면을 보면 MySQL-Front 2.5 아이콘이 만들어져 있는 것을 알 수 있다.



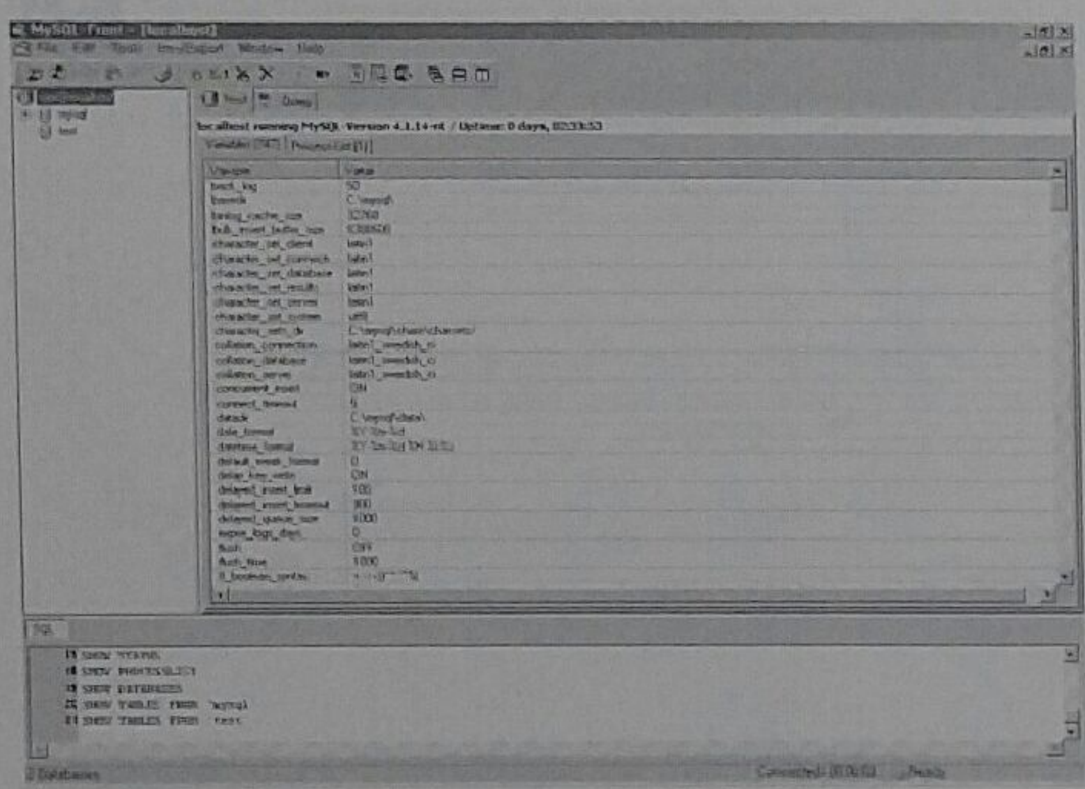
▶ MySQL-Front의 단축아이콘

이제 바탕화면에 있는 단축아이콘을 클릭하여 실행시켜 본다. 실행이 되면 접속하고자 하는 호스트명 또는 IP의 입력/ 아이디/ 패스워드를 입력한다. 처음 MySql을 설치하면 사용자의 암호는 지정되지 않았기에 입력하지 않는다.



▶ MySql-Front를 통한 MySql의 접속

접속이 되면 MySQL-Front 초기 화면이 나타난다.

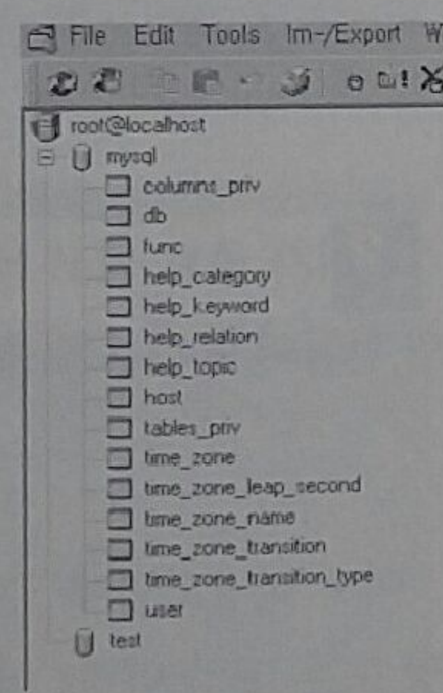


▶ MySQL-Front의 접속 초기화면

이제부터 우리는 MySQL-Front가 가지는 기능에 대해서 전반적으로 알아보자.

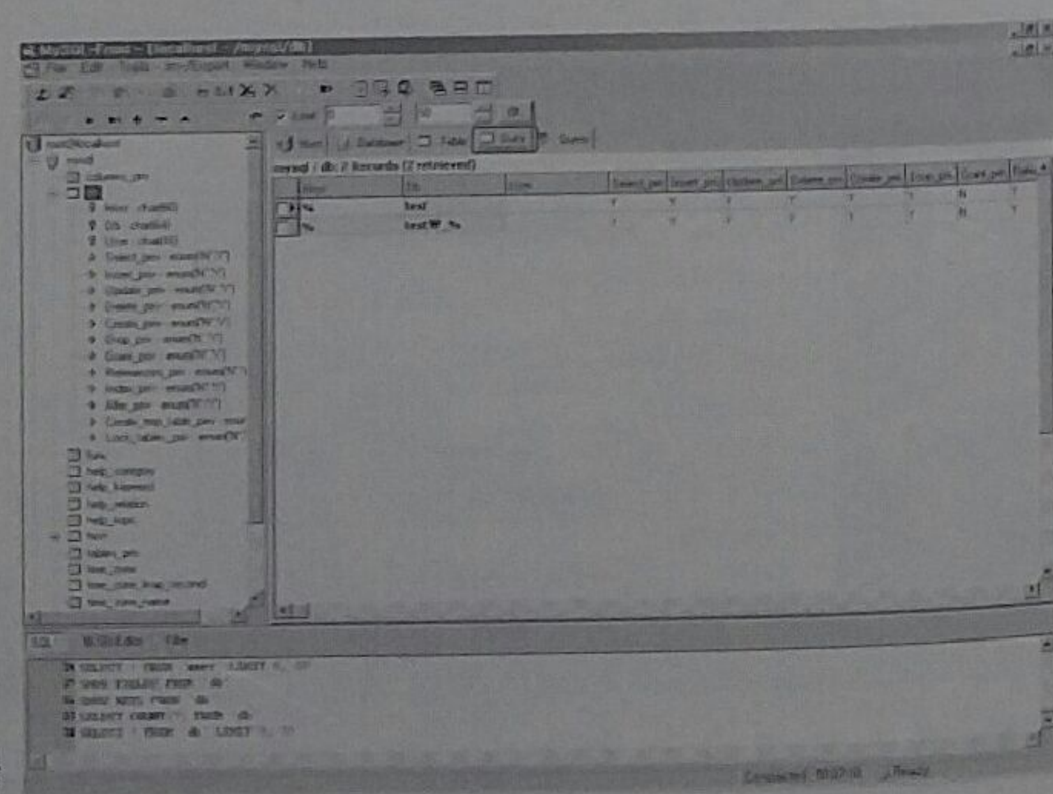
MySQL-Front의 주요기능

MySQL-Front의 기능을 알아보기 전에 먼저 mysql에서 기본적으로 제공되는 데이터베이스에 대해서 알아보자. 초기화면을 보면 왼쪽 편에 기본적으로 생성되는 mysql/ test 데이터베이스를 보게 될 것이다. +mysql 데이터베이스를 클릭하면 여러 개의 테이블이 나오는데 그중에서 우리가 사용할 테이블은 Db / User 테이블이다.



▶ MySql의 기본 테이블 mysql

Db 테이블을 선택 후에 화면 중앙에 있는 탭 메뉴 중 data를 클릭한다. 그럼 Db 테이블에 저장되어 있는 데이터가 화면에 보인다.

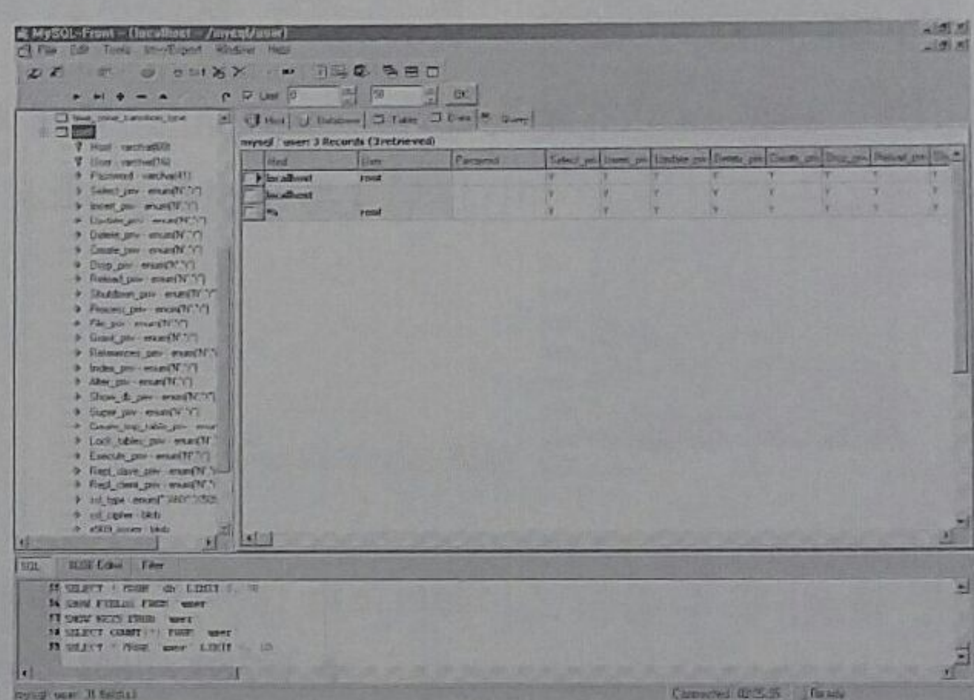


▶ Db 테이블

Db 테이블에 중요부분을 보면 Host/Db/User이다. Host는 생성된 데이터베이스의 접속제어로 현재 있는 %는 모든 곳으로부터 접속이 가능하다는 의미를 가진다. Db는 생성된 데이터베이스 이름이며 User는 생성된 데이터베이스를 사용하는 사용자를 의미한다. 간략하게 표현하면 다음과 같은 의미를 가진다.

Host	생성된 데이터베이스의 접근여부 %는 제한 없이 모든 곳에서 사용
Db	생성된 데이터베이스의 이름
User	사용자로 User 테이블에 등록되어 있는 사용자

User 테이블은 생성되는 데이터베이스의 사용자를 등록하고 그에 따른 권한을 부여하는 곳이다. 즉 이곳에 등록되지 않는 사용자는 mysql에 접속할 수 없다는 것이다.



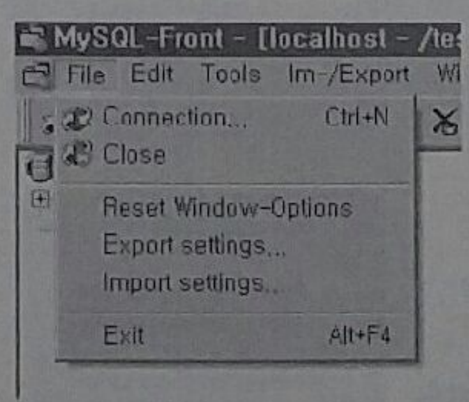
▶ User 테이블

Host	생성된 데이터베이스의 접근여부 %는 제한 없이 모든 곳에서 사용
User	사용자
password	사용자 암호

두개의 테이블을 보면 알 수 있듯이 공통적인 Host가 들어간다. Host에 특정 아이피를 입력하게 되면 그 아이피로 접속은 허용하고 그 외의 아이피로는 접속이 불가능하다는 것을 의미한다. 이 기능은 웹상에서 서비스를 할 때 적용하게 되면 사용자에 대한 제약이 따른다.

File

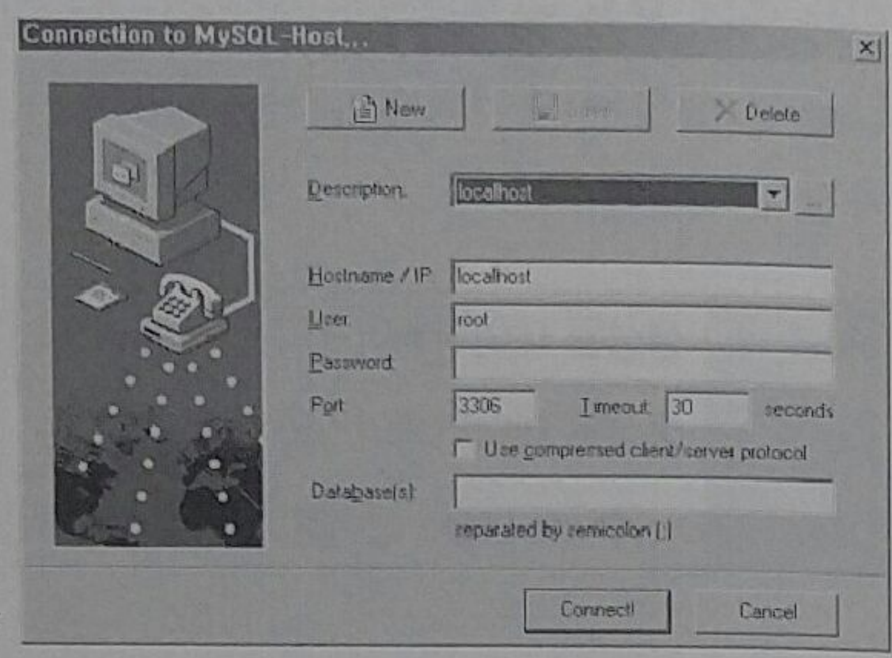
MySql과의 재접속 또는 다른 호스트에 있는 MySql과의 접속 및 연결 끊기 등이 사용되고 있는 MySQL-Front의 환경설정 메뉴로 구성되어 있다.



▶ File 메뉴

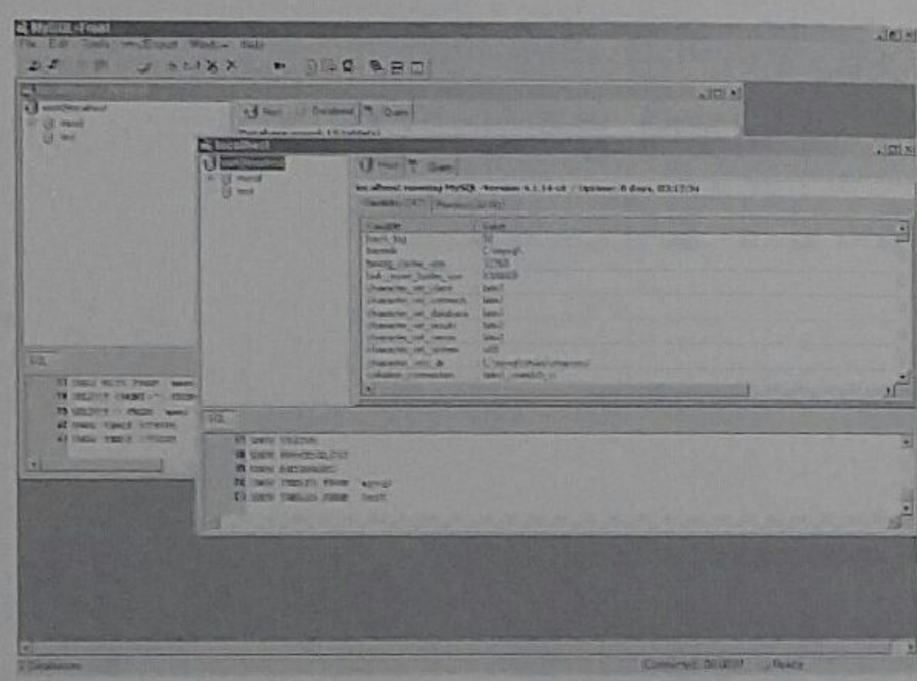
Connection

다른 MySql에 접속하거나 동일한 서버내의 MySql에 접속할 때 또는 재접속을 하기 위해 사용한다.



▶ File 메뉴의 Connection 창

만약 다른 서버에 접속하려면 Hostname/IP에 해당 입력 값을 입력하고 아이디/ 암호를 입력하면 된다.



▶ 동일한 서버에 Connection

Close

접속된 해당 Connection을 끊어버린다.

Reset Window-options

MySQL-Front의 초기화된 모습으로 Reset한다.

Export setting

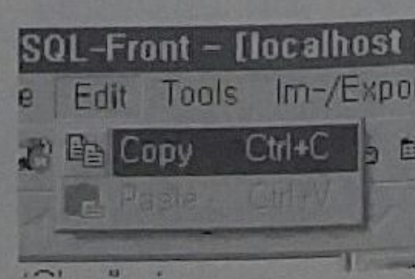
현재 사용하고 있는 MySQL-Front의 환경을 외부 파일로 저장한다.

Import setting

환경이 저장된 파일을 불러들여 적용한다.

Edit

Edit 메뉴는 간단하다. 메모장의 Copy와 Paste를 생각하면 된다. 다른 점은 선택한 데이터베이스 내의 테이블 안에 있는 데이터의 값을 복사하고 붙여넣기를 한다.



▶ File의 Edit 메뉴

Copy

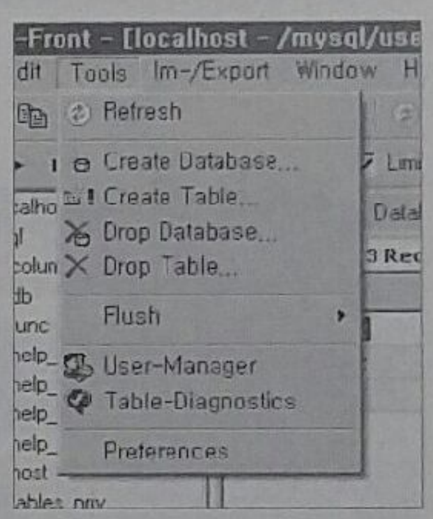
선택된 데이터베이스 내의 테이블에 대한 값을 복사한다.

Paste

해당 데이터베이스 내의 테이블에 복사된 값을 붙여 넣는다.

Tools

Tools 메뉴는 데이터베이스를 생성/ 삭제, 테이블의 생성/삭제, 권한 값 변경 시 적용 등을 관리하는 기능을 가진다.



▶ File의 Tools 메뉴

Create Database

사용자가 데이터베이스를 생성할 수 있도록 한다. 클릭하면 생성될 데이터베이스 명을 준 후 Ok를 클릭하면 된다.

Create Table

사용자가 선택한 해당 데이터베이스 안에 테이블을 생성하게 한다.

Drop Database

사용자가 생성한 데이터베이스를 삭제한다.

Drop Table

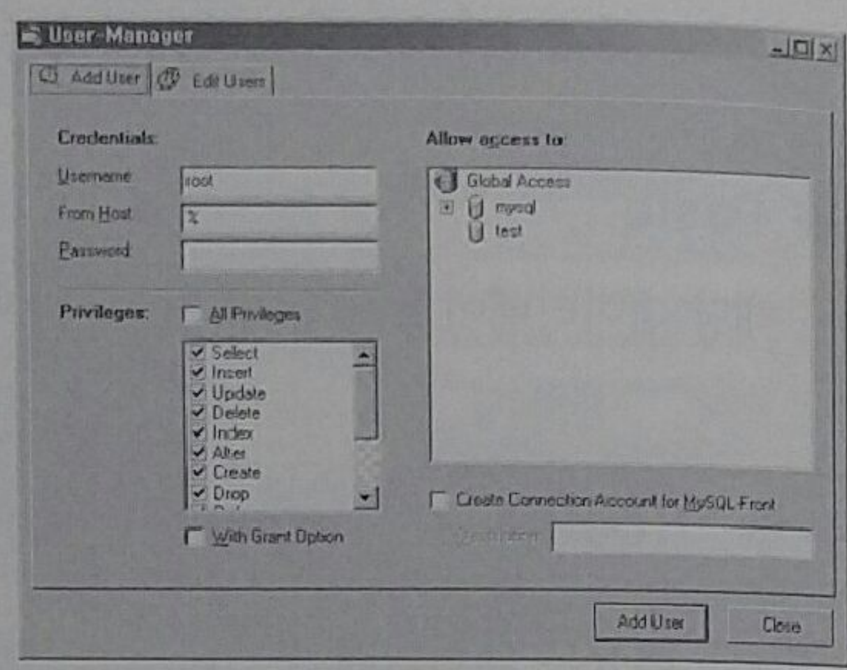
사용자가 생성한 데이터베이스내의 테이블을 삭제한다.

Flush

권한 값을 변경했을 경우 mysql의 재시작 없이 적용되게 한다. Flush 메뉴 중에서 privileges는 현재 사용중인 mysql의 캐시를 지우고 새로운 설정을 적용한다. 하지만 이 명령은 누구나 다 하는 것이 아니라 관리자로부터 reload의 권한을 받아야 사용할 수 있다.

User Manager

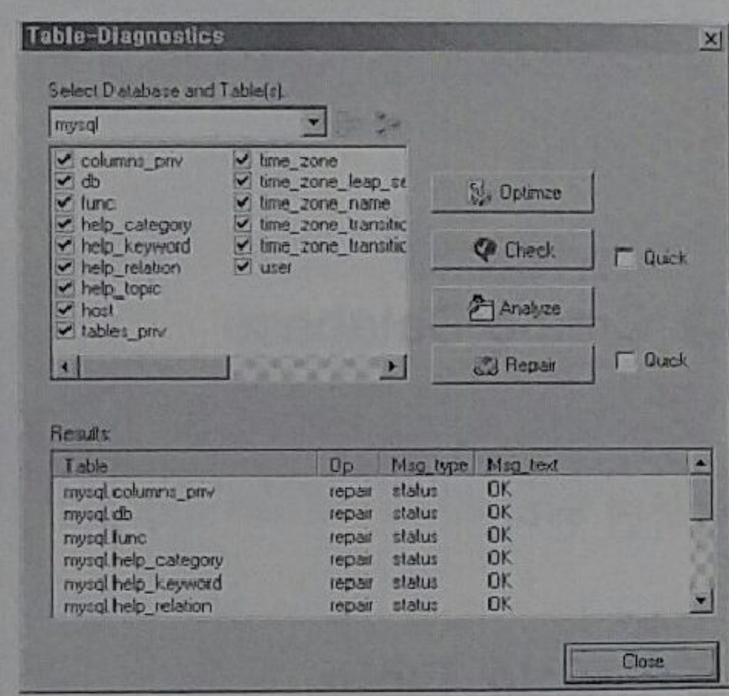
사용자를 생성하고 암호부여하며 생성된 사용자에 대한 권한을 부여한다.



▶ User Manager 창

Table-Diagnostics

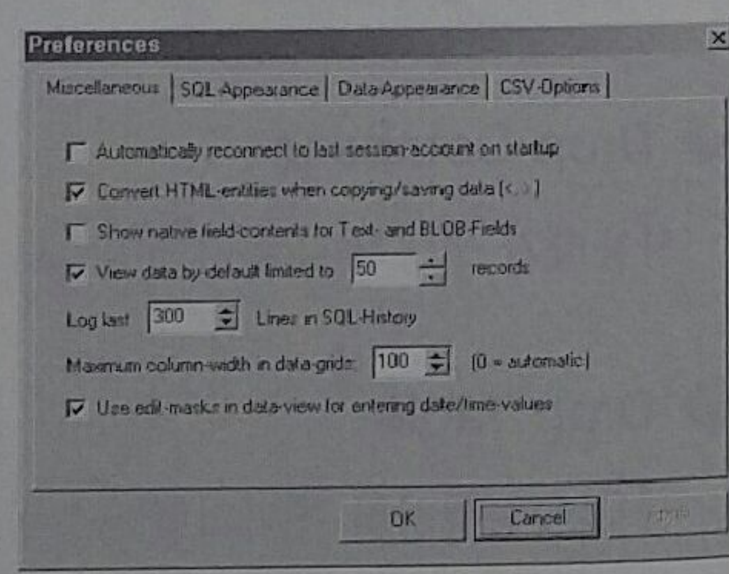
생성된 테이블에 대한 오류여부 등 진단사항을 관리하는 기능을 한다.



▶ Table-Diagnostics 창

Preferences

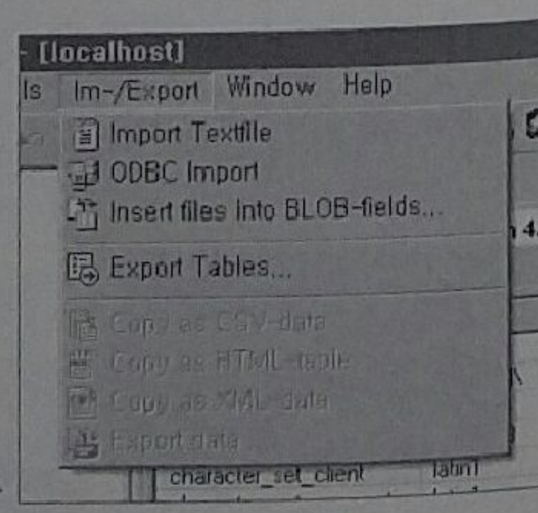
MySQL-Front에서 사용되는 환경을 설정한다.



▶ Preferences 창

Im/Export

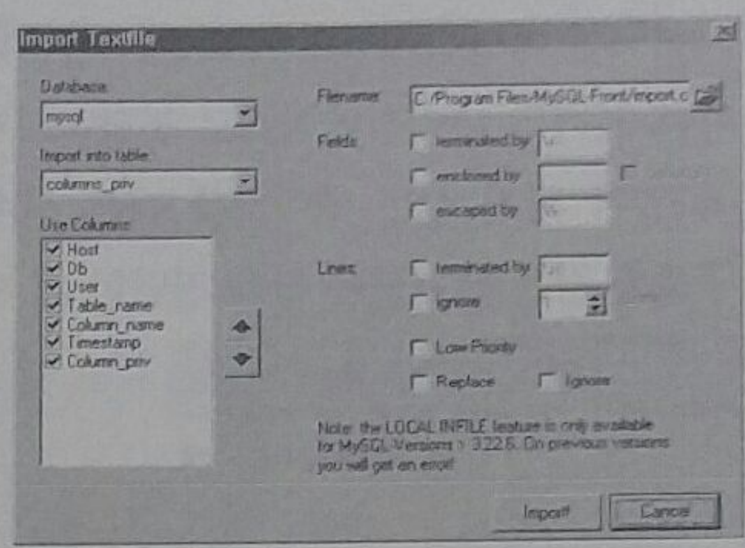
MySql 안의 데이터의 구조 또는 데이터의 값을 외부로부터 전달받거나 외부로 보내기 위한 파일을 생성 등의 기능을 가진다.



▶ Im/Export의 메뉴

Import Textfile

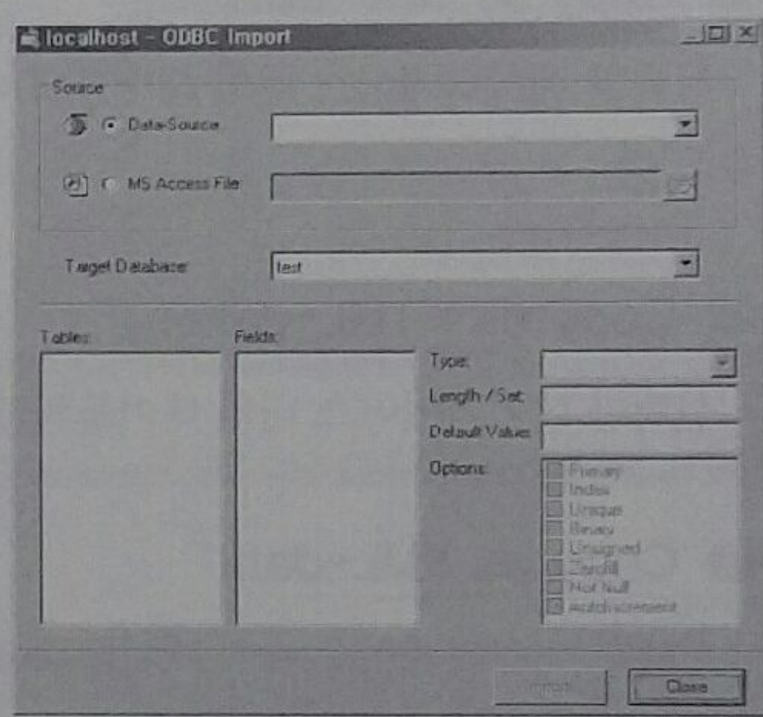
외부의 파일을 데이터베이스 내에 입력하고자 할 때 사용한다.



▶ Import Textfile 창

ODBC Import

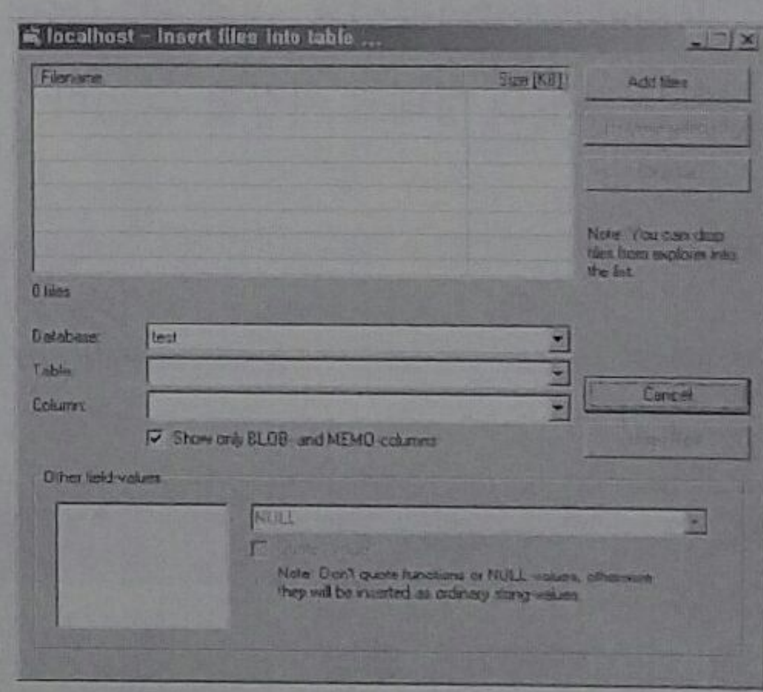
ODBC를 이용하여 데이터베이스내의 테이블에 값을 입력할 때 사용한다.



▶ ODBC Import 창

Insert files into BLOB-fields

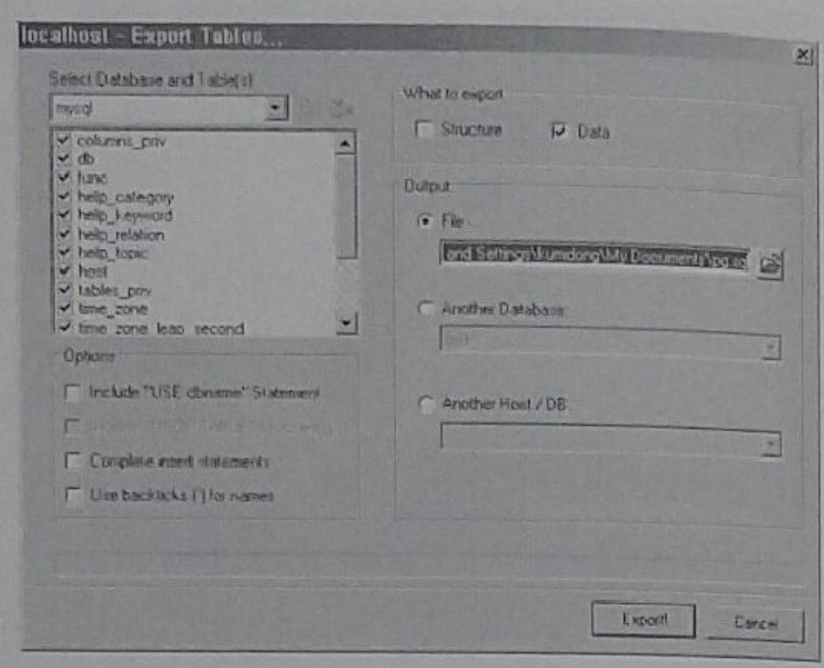
선택한 데이터베이스 내의 테이블에 BLOB 즉 바이너리 데이터를 입력할 때 사용한다.



▶ Insert files into BLOB-fields 창

Export Table

선택한 데이터베이스 내의 테이블을 외부파일로 만들거나 다른 서버의 데이터베이스로 전달하고자 할 때 사용된다. 간혹 얼마 되지 않는 데이터를 다른 서버에 존재하는 똑같은 구조를 같은 테이블에 값을 전달할 때 사용된다.



▶ Export Table 창

Copy as CSV-data

선택한 데이터베이스 내의 테이블의 데이터를 Copy하여 메모장이나 기타 에디터 프로그램에 복사할 수 있다.

Copy as HTML-data

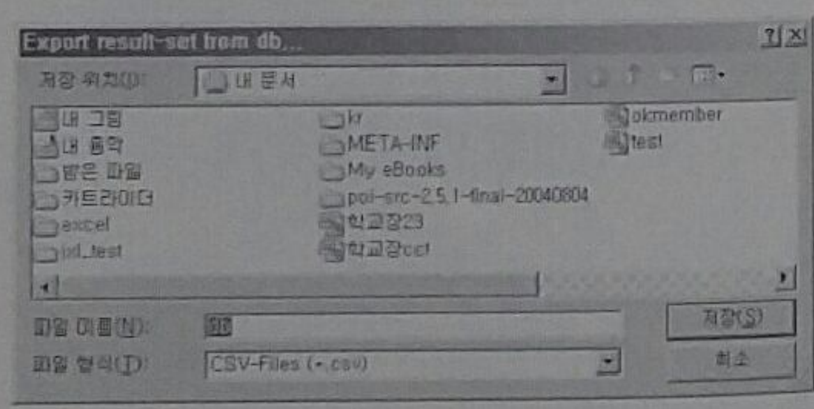
선택한 데이터베이스 내의 테이블의 데이터를 Copy하여 HTML 형식으로 복사할 수 있다.

Copy as XML-data

선택한 데이터베이스 내의 테이블의 데이터를 Copy하여 XML 형식으로 복사할 수 있다.

Export data

선택한 데이터베이스 내의 테이블의 데이터를 사용자가 원하는 파일의 형태로 저장할 수 있다.

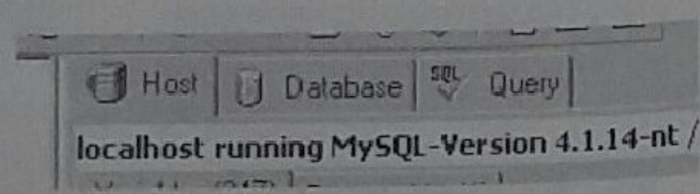


▶ Export data 파일 창

Tab 메뉴

데이터베이스 메뉴

탭 메뉴는 현재 선택되어진 데이터베이스의 정보를 보여주며 데이터베이스 내에 생성된 테이블의 정보 그리고 쿼리 작성 메뉴로 구성된다.



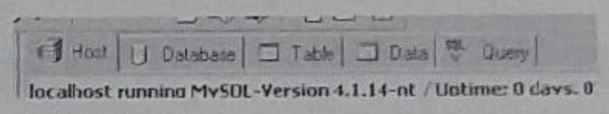
▶ 데이터베이스를 선택한 경우의 탭 메뉴

- host

선택한 데이터베이스에 대한 mysql에 대한 정보와 설치 시 환경을 보여준다. host 메뉴에는 2가지의 탭 메뉴가 있다. Variables / Process-List가 있다.

- Variables

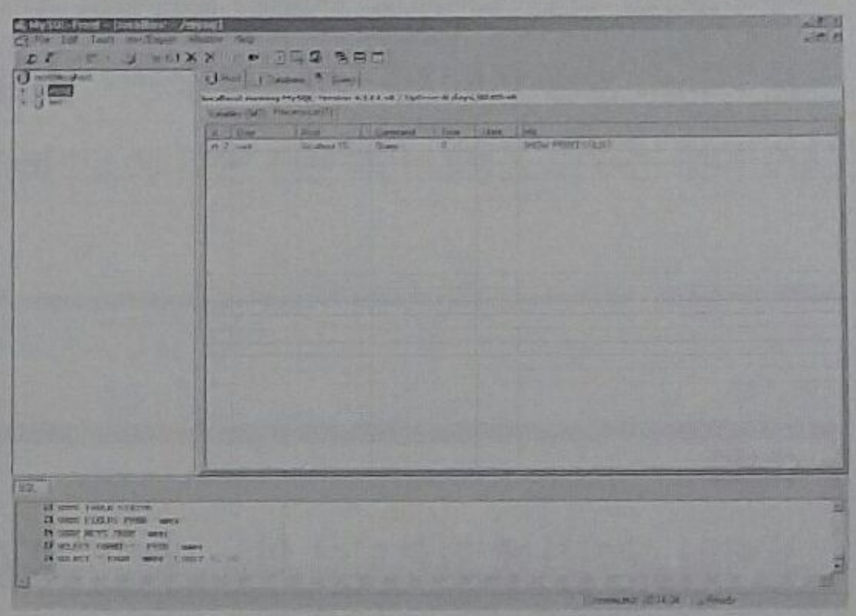
설치된 mysql의 정보과 환경에 대한 정보를 보여준다.



▶ Variables

- Process-List

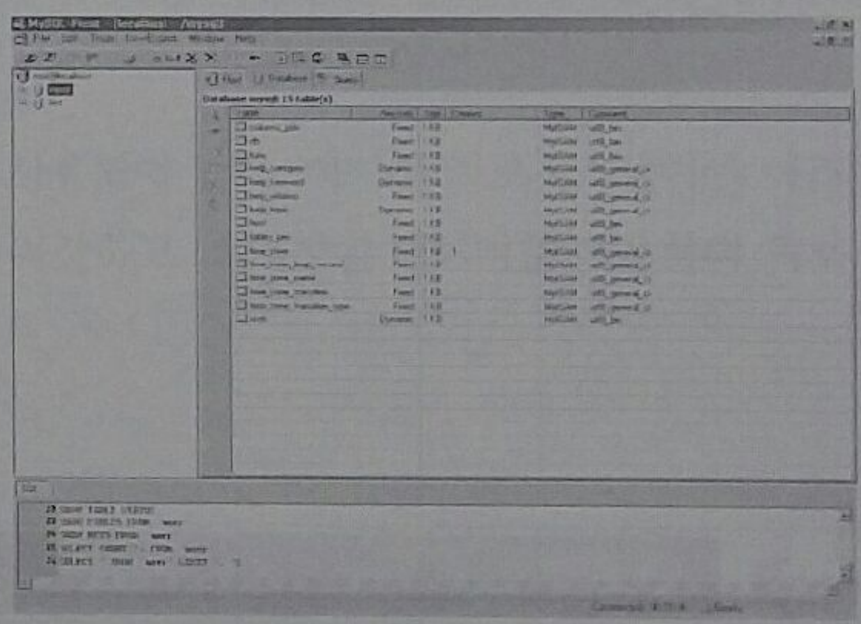
현재 msyql에서 작동되어 있는 프로세서를 보여준다.



▶ Process-List

- Database

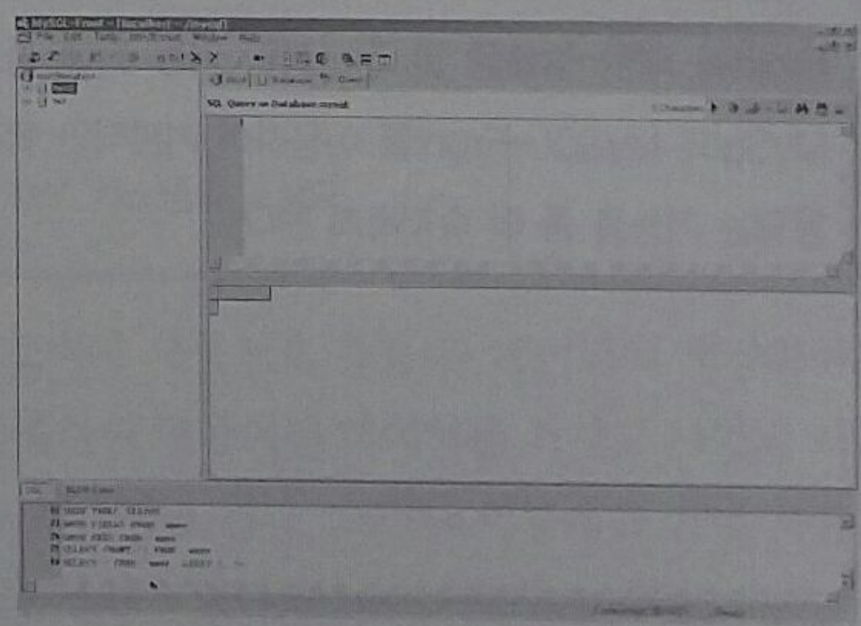
선택되어진 데이터베이스 내의 테이블에 대한 정보를 보여준다.



▶ Database 메뉴

- Query

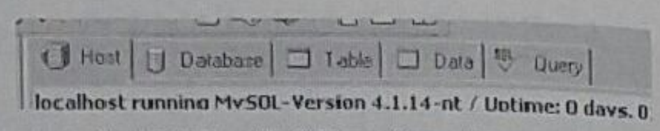
쿼리문을 작성할 수 있는 영역/ 결과 값 / 상태영역을 보여준다.



▶ Query 메뉴

테이블 메뉴

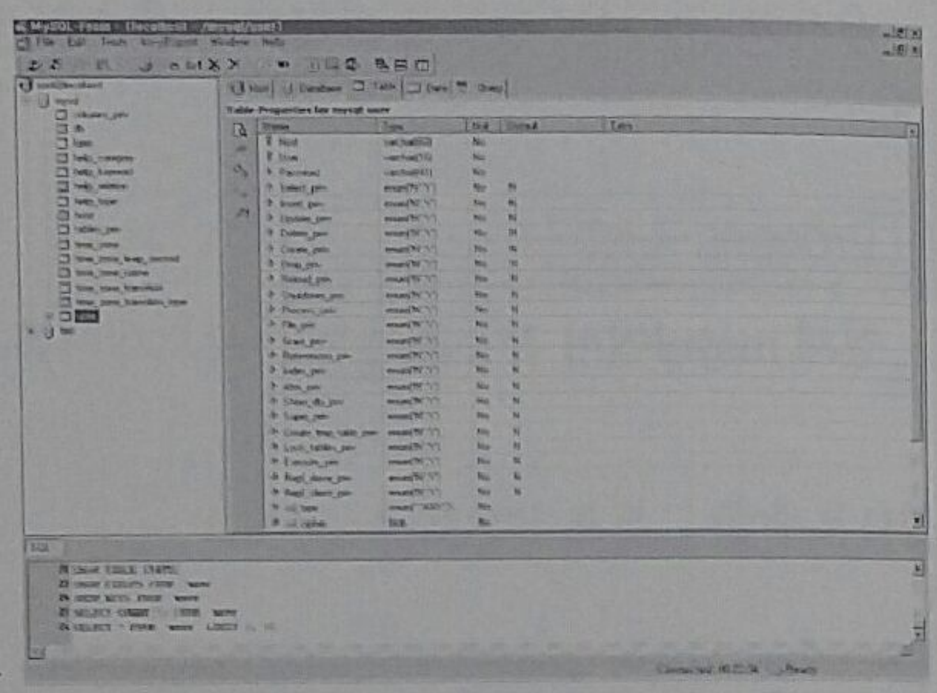
데이터베이스 내에 테이블을 선택하면 탭 메뉴가 두개가 추가된다. 추가된 메뉴 이외의 것은 동일하다.



▶ 테이블을 선택한 경우의 탭 메뉴

– table

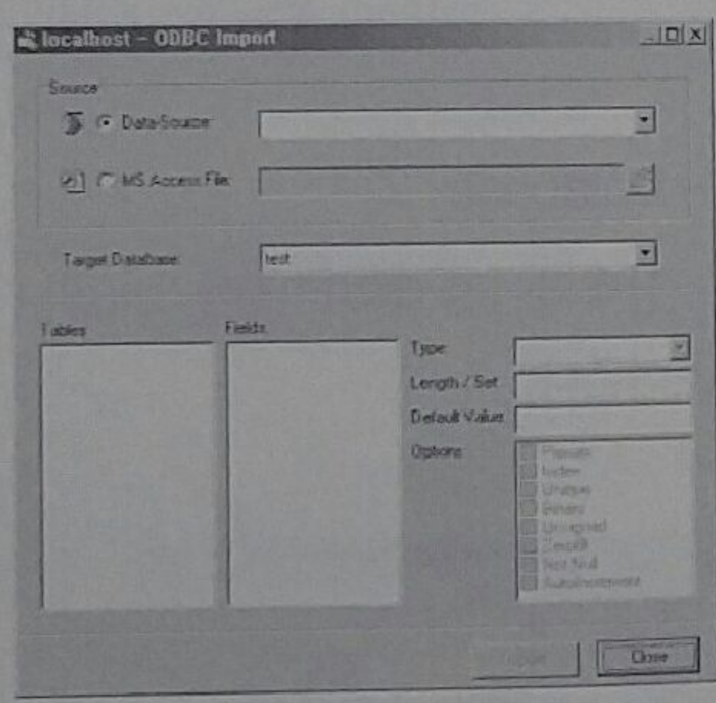
선택한 테이블에 대한 칼럼에 대한 정보를 보여준다.



▶ 테이블 정보

– data

선택한 테이블에 입력되어 있는 데이터의 리스트를 보여준다.



▶ 입력데이터 리스트

맺음말

이번 장에서는 이 책의 기본이 되는 MySql의 필요성과 설치 그리고 윈도우 서비스에 등록하는 방법, MySQL-Front에 대한 필요성 및 전반적인 기능 중에 중요한 기능에 대해서 알아보았다. 다음 장에서는 1장에서 설치된 MySql과 MySQL-Front를 이용하여 기본적인 쿼리문에 대해서 배우고 MySQL-Front에서의 기능 중에서 자주 사용되는 기능을 좀 더 숙지하게 된다.

02 데이터베이스의 이해와 기본 SQL

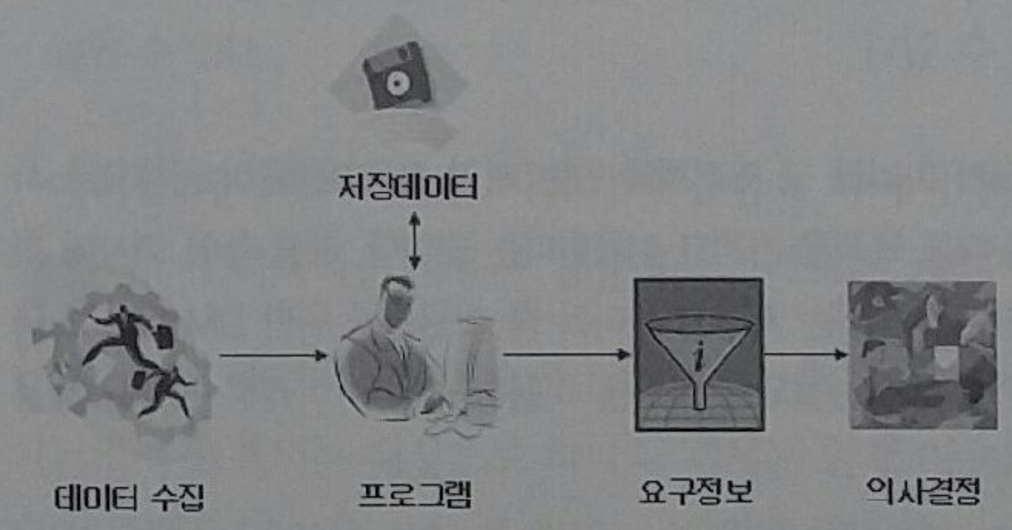
안녕하세요

이번 장에서는 데이터베이스의 기본 문법과 그에 따른 사용법에 대해서 알아본다. 또한 1장에서 전반적인 기능을 설명한 MySQL-Front에서 자주 사용되는 기능에 대해서도 숙지하게 된다. 그러기 위해서 먼저 데이터베이스가 무엇인지를 이해해야 한다. 그렇다고 데이터베이스에 대한 전문적인 내용을 다루는 것이 아니라 기본적인 개념, 정의 등의 중요한 사항을 알아본다.

데이터베이스의 기본 정의

데이터베이스의 정의를 알기 전에 먼저 데이터베이스의 등장부터 살펴보기로 하자. 데이터베이스는 컴퓨터의 파일시스템의 문제점들을 해결하기 위해 등장하였다. 파일 시스템은 정보처리기술이 발달하지 못한 시점에서 주로 사용되었던 방법이었다.

이 파일 시스템은 초기에 정보시스템의 기초적인 도구로 사용되어 왔다. 여기서 정보시스템이란 여러분들이 어떠한 의사결정에 도움을 받고자 수집한 데이터를 저장하고 그 저장된 데이터를 이용하여 의사결정에 도움을 주는 도구라고 생각을 하면 된다.



▶ 정보 시스템의 구성도

즉 사업을 하려는데 그 사업에 대한 기본자료(시장조사/소비자 제품선호 등등)를 데이터에 저장하여 사업에 필요한 자료(시장 분포도, 소비자 선호도 등등)를 축출하여 만들어진 데이터를 가지고 사업의 시작유무에 대한 의사결정의 자료에 도움이 되는 방법이라고 생각하면 된다. 그러나 이렇게 초기에 사용되어진 파일 시스템의 문제는 다음과 같은 6가지의 문제점을 가지고 있다.

1. 복잡한 프로그래밍 언어를 사용해야 데이터 접근이 가능하고 사용자들이 직접 저장된 데이터에서 알고자 하는 정보를 얻기에는 불가능하였다.
2. 해당 파일의 물리적인 구조를 알고 있어야 그 파일에 대한 접근방법을 응용하여 프로그래머가 프로그램을 조작할 수 있으므로 프로그래머는 파일의 물리적인 구조에 대해서도 전문가적 지식을 가지고 있어야 했다.
3. 파일 시스템은 파일의 구조가 갱신이 되면 관련된 모든 프로그램을 수정해야 한다. 왜냐하면 파일시스템은 데이터의 독립성을 지원하지 않기 때문이다.
4. 파일시스템은 부분적으로 동일한 데이터를 여러 파일에 중복 저장해야 하므로 데이터 중복에 따르는 여러 문제점을 발생시킬 수 있다.
5. 여러 사용자가 동일한 데이터를 접근하는 경우에 큰 오류를 발생 시킬 수 있다.
6. 파일 시스템은 파일 전체에 대한 사용자 접근을 단지 패스워드 시스템만을 지원하기 때문에 파일에 저장된 데이터에 대한 보안점이 취약하다.

이와 같은 6가지의 문제점 중 알고 있어야할 사항은 데이터의 종속성과 데이터의 중복성이다. 좀 더 쉽게 이야기하자면 데이터의 종속성은 파일의 논리적인 구조가 변경됨에 그에 따르는 데이터의 처리 방법이나 성능에 관련된 모든 프로그램을 수정해야 한다는 것이다.

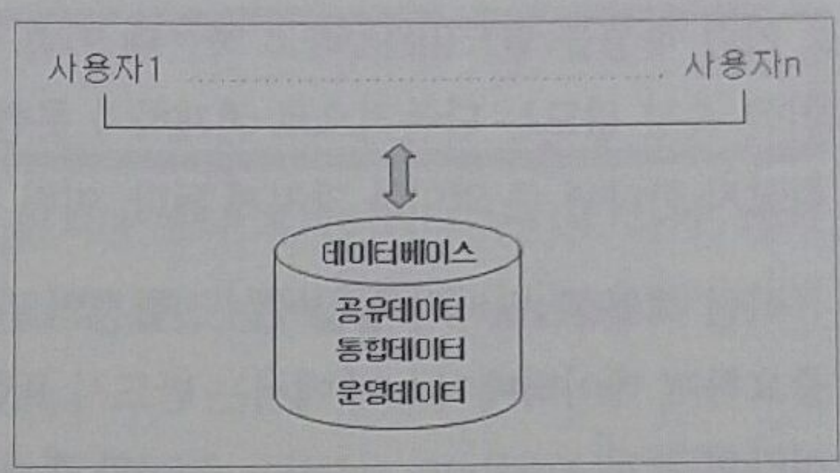
예로 전화번호의 국번이 3자리에서 4자리로 변경이 되면 그에 따르는 모든 프로그램이 변경이 되는 것도 이에 속하는 것이라고 할 수 있다.

또한 데이터의 중복성의 문제는 어쩌면 가장 중요한 문제일 수 있다. 만약에 동일한 데이터가 여러 파일에 중복이 되어 저장이 된다고 생각해 보자. 그럼 그 데이터를 저장하는 비용이 증가하고 데이터를 수정할 때마다 비용은 추가되며 데이터가 정확하게 일치한다는 보장을 받을 수 없다. 쉽게 주민번호를 생각하면 중복성의 중요성과 데이터의 일치성을 생각해 보면 된다.

이러한 문제점을 극복하기 위해 태어난 것이 데이터베이스이다. 데이터베이스의 기본정의는 한 조직에서 여러개의 프로그램들이 사용할 수 있도록 통합, 저장된 데이터의 집합이라고 정의 할 수 있다. 이런 의미의 데이터베이스를 크게 세 가지로 살펴볼 수 있다.

1. 데이터베이스는 어느 하나의 프로그램만을 위한 데이터가 아니라 그 조직체에 있는 여러 프로그램들이 공동으로 사용하는 공유 데이터이다. 다시 말하면 여러 사용자들이 다른 목적을 가지고 이용한다는 것이다. 예를 들어 회사에 회원들이 가입이 되었고 그 가입된 회원을 관리하는 회원관리 부서는 저장된 데이터 중 회원들을 위한 DM 발송에 대한 주소 데이터를 사용할 것이고 마케팅 부서에서는 가입된 회원들의 선호도를 이용하여 향후 마케팅 전략을 세울 것이다. 즉 데이터베이스는 공유데이터라고 할 수 있다.
2. 데이터베이스는 데이터를 하나의 논리적인 저장매체에 저장한다. 그럼으로써 데이터베이스 내에서는 중복된 데이터를 저장하지 않는다. 다만 인위적으로 필요에 따라 데이터베이스의 성능 향상을 위해서 부분적으로 허용(이러한 것을 최소의 중복성이라 한다)은 하지만 원칙적으로는 중복성을 허용하지 않는다. 이는 데이터베이스가 통합데이터이기 때문이다.
3. 데이터가 필요한 이유는 앞에서 언급했다시피 의사결정에 도움을 주는 자료로 활용하기 위한 것이다. 이러한 의사결정 및 어떤 조직체가 그 고유의 기능(회원관리부서/마케팅부서/물류부서 등등)을 수행하기 위해 반드시 지속적으로

유지하는 데이터를 운영 데이터라고 한다. 따라서 데이터베이스는 불필요한 데이터가 아닌 사용목적이나 기능에 따라 반드시 필요한 데이터의 집합이다.



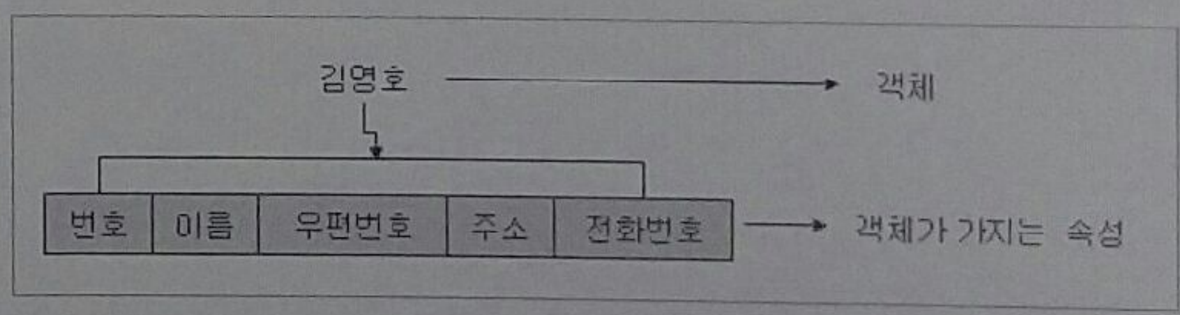
▶ 데이터베이스의 정의

이런 데이터베이스는 시스템의 복잡도 및 비용증가라는 단점도 존재하지만 데이터를 처리하고 저장하는데 더 많은 장점을 가지고 있다.

1. 앞서 말한 데이터베이스의 정의라고 할 수 있는 공유가 가능하다는 것이다.
2. 통합데이터이기 때문에 중앙에서 관리 각각의 개별적인 파일 시스템을 유지할 필요가 없으며 최소한의 중복성을 유지할 수 있다.
3. 통합된 데이터이며 그 통합된 데이터를 가지고 운영되는 데이터이기 때문에 데이터의 중복을 제거 데이터의 불일치는 발생하지 않는다. 행여 중복데이터가 있다고 하더라도 데이터베이스가 중복된 데이터를 인식하기 때문에 중복의 경우가 발생할 가능성이 최소화 된다.
4. 데이터에 대한 무결성을 유지할 수 있다는 것이다. 2장에서 배우겠지만 무결성이라는 것은 데이터베이스에 저장된 데이터가 정확하다는 것을 의미한다. 데이터베이스는 정확한 데이터를 위해 무결성 제약조건을 통해 구현이 가능하다는 것이다.
5. 데이터베이스는 공유데이터이다. 그럼으로 무엇보다 보안에 대한 문제는 중요하다. 데이터베이스는 모든 데이터의 중앙 집중관리 통해 해당 데이터에 접근해야 할 정당한 이유와 접근 권한이 있는 사용자에게만 데이터베이스의 접근을 허용한다. 즉 데이터베이스 관리자(DBA)가 모든 데이터에 대해 강력한 권한으로 중요 데이터에 대한 접근에 대해 보안 기능을 이용하여 제어할 수 있다.
6. 모든 데이터를 중앙 집중적인 관리를 통해 조직체에서 사용되는 데이터를 데이터베이스에 저장하고 관리하여 각 조직에 따라 필요한 적합한 데이터를 표준화하여 정립하여 데이터의 공유에 따른 사용자들이 데이터에 대한 이해를 높일 수 있다.

이러한 데이터베이스는 사용자의 관점에 따라서 논리적인 구성요소와 물리적인 구성요소로 분류할 수 있다. 이 책에서는 사용자 입장에서 바라보는 논리적인 구성요소에 대해서 언급할 것이다.

논리적인 데이터베이스의 구성요소에는 객체와 관계로 나눠진다. 데이터베이스에서 객체라는 것은 데이터로서 표현되는 대상으로 하나의 객체는 한개 이상의 속성으로 구성되어 있다. 쉽게 이야기하면 레코드의 개념을 가지는 것이 객체(또는 엔티티)라고 하고 이 레코드 안에 정의된 것들을 속성이라 한다. 예를 들어서 친구들의 주소록이 있다고 한다면 친구들 중 김영호가 '객체' 이고 이 김영호가 가지는 속성으로 나열번호, 이름, 우편번호, 주소, 전화번호가 '속성' 이 되는 것이다.

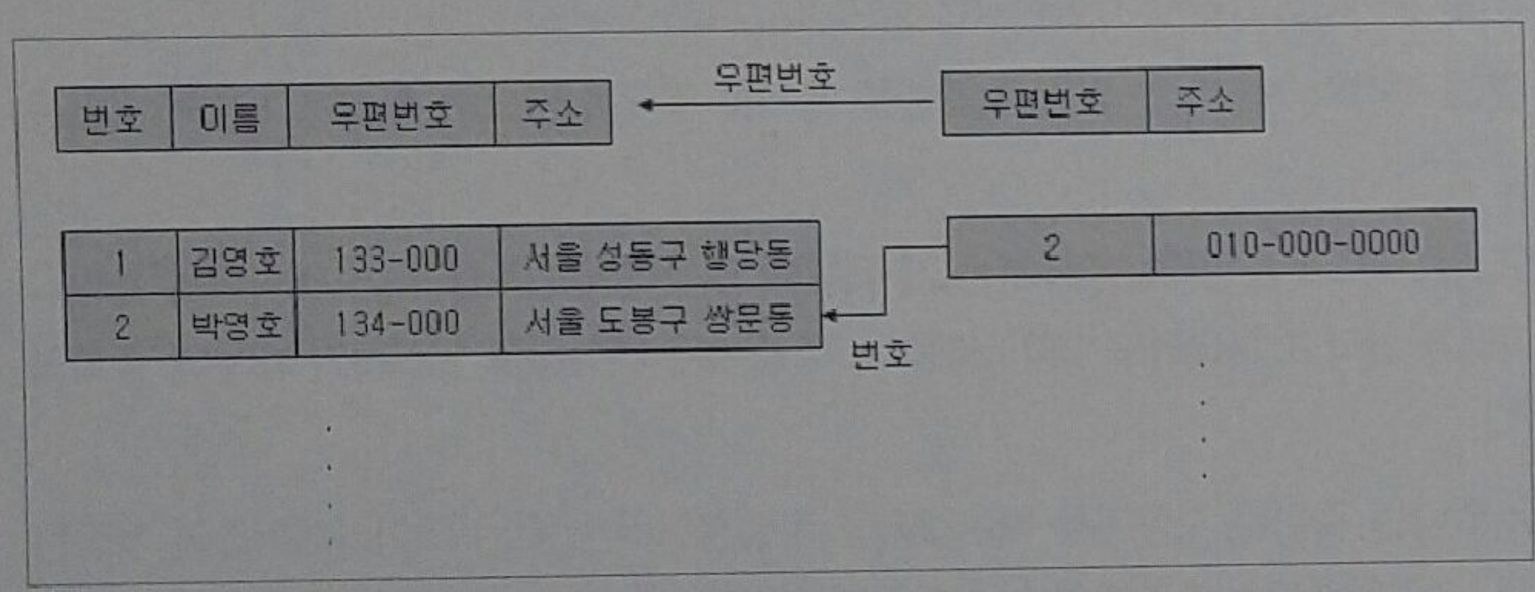


▶ 객체와 속성

이런 속성을 필드(field)라고 하는데 이런 필드 하나 하나는 아무런 의미를 가지지 못한다. 따라서 이러한 속성 필드는 단독적으로 존재하지 못한다. 그러나 이러한 속성들이 모여 김영호라는 객체를 표현하고자 한다면 큰 의미를 가지게 된다. 이런 객체들이 모이면 객체집합이라고 한다.

이런 객체(Entity)의 집합에는 집합간의 관계가 존재한다. 이러한 관계는 정보의 관점에서 볼 때 매우 중요하며 데이터베이스 내에서는 반드시 표현이 된다. 이 관계는 두 가지로 분류할 수 있는데 속성과 속성간의 관계(Attribute relationship)와 객체와 객체와의 관계(Entity relationship)로 나눌 수 있다.

이러한 관계를 쉽게 이해하기 위해 다음 그림을 살펴보자. 우편번호가 133-000인 친구를 검색한다면 속성을 통해서 처리할 것이고 번호가 2인 친구를 찾아 번호/ 이름/ 우편번호/ 주소/ 전화번호를 검색한다면 객체간의 관계를 통해 이루어진다.

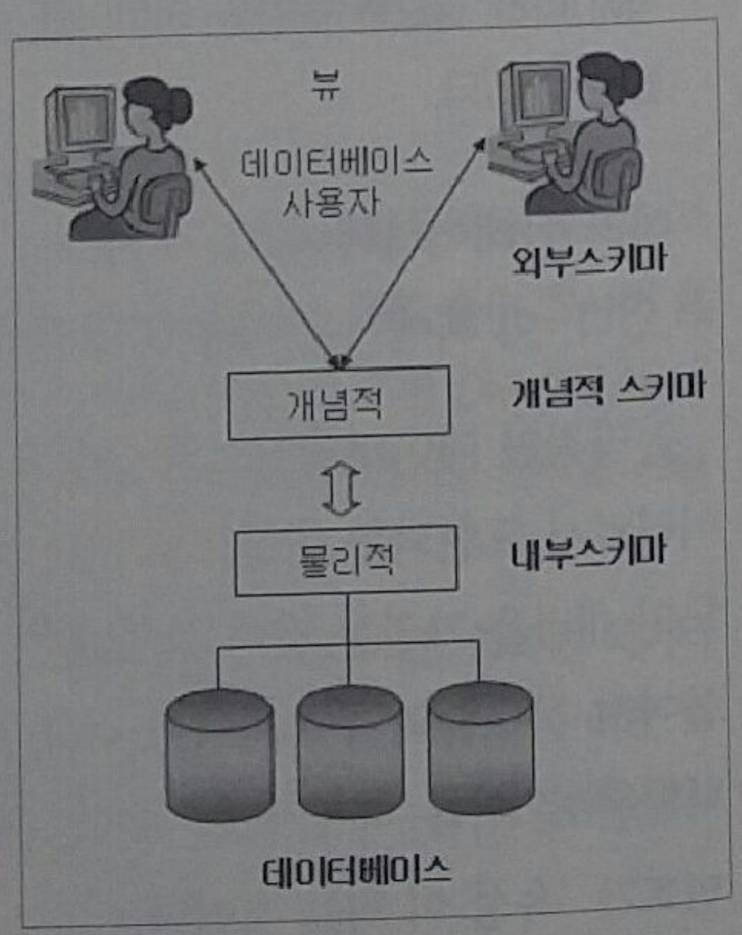


▶ 속성관계와 객체관계

이런 데이터베이스의 구성요소를 논리적 정의를 기술한 것을 스키마라고 하는데 스키마는 데이터의 주조를 표현하는 객체와 속성 그리고 객체 간에 존재하는 관계에 대한 정의와 이들이 서로 유지해야 할 제약조건들이 포함이 된다.

스키마에 대해서는 뒤에서 더 자세히 설명하기로 하며 일단 데이터베이스는 데이터를 공동으로 사용하기 때문에 개별적인 사용자가 바라보는 관점에 따라 세 가지 구조로 분리할 수 있다.

1. 물리적 – 추상화의 마지막 단계로 데이터가 실제로 어떻게 저장이 되는지를 기술하며 이렇게 기술된 내용을 물리적 스키마 또는 내부 스키마라고 한다.
2. 개념적 – 물리적 단계의 상위단계로서 실제로 저장된 데이터가 저장된 데이터들 간의 관계와 제약 조건을 기술한 것으로 개념적 스키마라고 한다.
3. 뷰 – 전체 데이터베이스 중 일부분만을 기술한 것으로 외부스키마라고 하며 각 사용자들이 관심을 갖는 데이터에 대한 부분만을 기술한다. 즉 자신에게 필요한 데이터만을 이용할 수 있는 단계라고 생각을 하면 된다.



▶ 3단계 데이터베이스 구조

데이터베이스 구축의 목적

데이터베이스 구축의 목적은 앞에서도 잠시 언급한 바가 있지만 보다 자세히 알아보기로 하자. 데이터베이스의 시스템의 종류와 구성방법에 따라 차이가 다소 있겠지만 기본적으로 데이터베이스가 가져야할 목적을 보자면 아래에 나열된 다섯 가지라 볼 수 있다.

1. 중복된 데이터의 일관성 유지
2. 정확한 데이터를 위한 무결성 유지
3. 데이터 중복의 최소화
4. 데이터의 보안성
5. 데이터의 논리적, 물리적 독립성이라고 볼 수 있다. 이중 우리가 앞으로 공부해 가면서 지켜야 하고 알아 두어야 할 몇 가지에 대해서 살펴보기로 하자.

데이터의 일관성

잘 설계된 데이터베이스는 중복을 최소화 하는 것이다(최소 중복성). 데이터의 중복은 공간의 낭비, 관리비용의 증가와 중요한 데이터의 일치성을 파괴하는 것이기 때문이다. 데이터의 일관성은 중복 데이터로 인해 파괴되는 것이 거의 대부분이다.

한 프로젝트가 장기간에 걸쳐 여러명의 프로그래머의 손을 통해 개발됨에 따라 동일한 정보가 여러 곳에서 중복될 수 있기 때문이다. 이런 일관성을 지키기 위해 트랜잭션을 이용하여 처리한다. 트랜잭션에 대해서는 16장에서 다루기 때문에 그때 다시 논의하기로 한다.

데이터의 무결성

데이터베이스에 들어있는 통합된 데이터의 정확한 데이터를 유지하는 것이다. 앞서 이야기한 데이터의 최소한의 중복에 대해 기억이 날 것이다. 데이터베이스에서도 중복의 문제는 발생하게 되어있다. 하나의 사실을 표시하는 2개 이상의 항목이 있고 그 항목의 내용이 서로 불일치 할 수 있기 때문이다. 또한 중복이 발생되지 않더라도 부정확한 데이터가 데이터베이스 안에 저장될 수 있다. 이러한 것을 방지하는 것을 '무결성을 유지한다' 라고 한다.

무결성을 유지하는 방법에는 일반적으로 DBA의 통제에 의한 데이터의 갱신방법이다. DBA는 모든 갱신처리과정에서 반드시 검증 단계를 거치도록 조치한다. 무결성의 유지는 데이터베이스의 중요한 목적 중에 하나이다.

데이터 중복의 최소화

데이터베이스 이전의 파일시스템에서는 중복이 매우 흔한 일이었다. 데이터베이스 경우에도 자동적으로 중복성을 배제하는 것은 아니다. 그러나 데이터베이스 기능 중에서 중복없이 동일한 데이터를 공동으로 사용할 수 있는 장점이 있기 때문에 중복을 최소화한다. 중복을 최소화함으로써 데이터 처리에 드는 저장 공간의 절약, 데이터 수정사항에 대해 신속하게 대응하여 변화된 데이터를 가지고 업무를 처리할 수 있다.

데이터 보안성

컴퓨터에 저장된 데이터는 언제나 안전하게 그리고 그 비밀을 유지 관리해야 한다. 만약 저장된 데이터에 보안성이 없다면 아마 개인 정보가 마구 흘러나가 불이익을 당할지도 모른다. 보안이라는 것은 하드웨어, 소프트웨어의 잘못으로 인한 데이터의 손실을 방지하는 것뿐만 아니라 천재지변이나 절도 또는 고의적인 자료유출로부터 데이터를 보호하는 것이다.

한 가지 예로 911 테러 당시 다른 곳에 데이터를 백업하지 않았다면 아마 막대한 금전적인 피해가 있었을 것이다. 기억장치 내에 있는 데이터 차원에서의 보안은 크게 두 가지로 볼 수 있다.

첫째는 데이터의 안전으로 우연이던 아니면 고의적이던 간에 허가된 사용자 이외의 사람에게 인지되는 것을 막고 파괴되는 것을 막는 것을 말한다. 둘째는 비밀유지로 소유하고 있는 또는 공유하고 있는 데이터를 취급할 수 있는 권리를 부여하고 그 한계를 지키는 것이다.

하지만 방패가 있으면 창이 있듯이 다변하는 기술적 진보로 인해 이를 막고자 하는 보안기술도 시급히 연구되어야 한다.

데이터 독립성

데이터의 독립성은 기존 파일 시스템이 가지는 문제점 중 기억장치의 변경 또는 운영업무의 변경 혹은 새로운 기능적 요소의 추가로 인해 파일의 구조 및 프로그램 모두를 수정해야 되는 문제점을 해결하기 위한 것으로 물리적 데이터 독립성과 논리적 데이터 독립성으로 분류된다.

물리적 데이터 독립성

물리적 데이터의 독립성이란 물리적으로 데이터의 구조가 변경되더라도 논리적 데이터 구조와 프로그래머가 보는 논리적 데이터 구조가 영향을 받지 않는 것을 말한다. 즉 프로그램이 사용자의 다양한 요구와 변화에 의해서 변경이 되어야 하기 때문에 물리적인 데이터 독립성은 필요하다. 쉽게 이야기하면 구형 컴퓨터에서 신형 컴퓨터로 바뀌었다고 해서 운영체제를 다시 프로그래밍을 할 필요가 없다는 것이다.

논리적 데이터 독립성

논리적 데이터의 독립성이란 프로그램의 재작성 없이 사용하는 것을 의미하는데 앞서서 말한바와 같이 개념적 단계 즉 실질적으로 데이터가 저장되고 저장된 데이터들과의 관계 그리고 제약조건 등에 관계없이 사용자가 사용하는 뷰(외부스키마)에는 전혀 영향을 주지 않아서 프로그램을 재작성할 이유가 없다는 것이다.

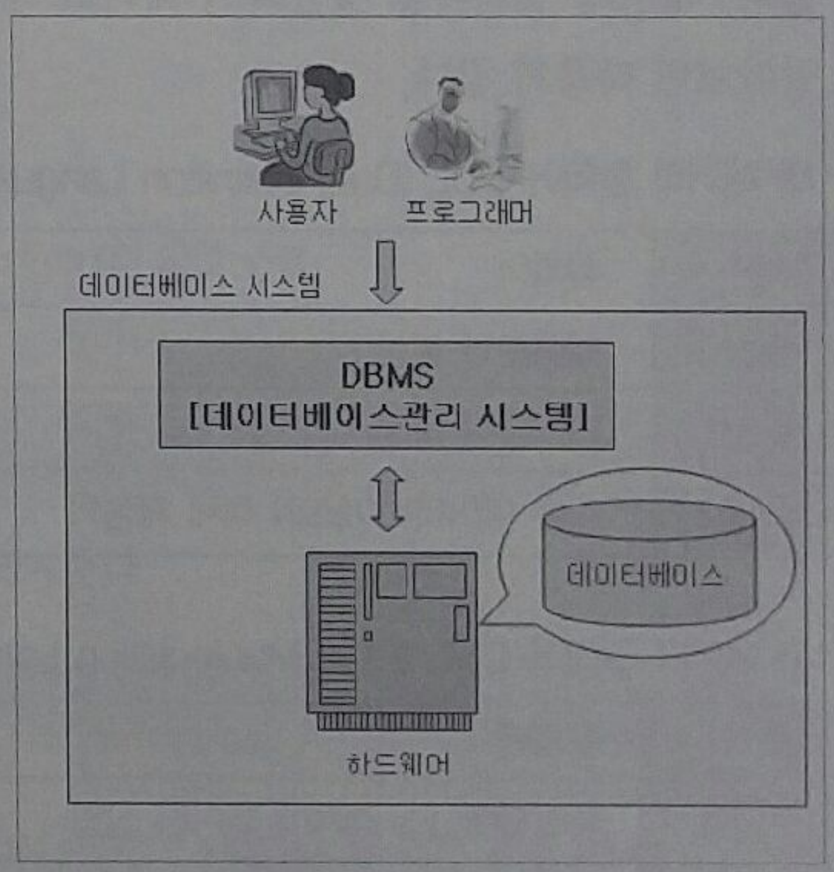
이러한 데이터베이스의 기본적은 구축의 목적을 뚜렷이 인식함으로서 종래의 파일시스템을 이용한 방법에서 문제점으로 지정된 변동이 심하고 다양한 업무에 효율성을 가져오게 될 것이다. 즉 본인 또는 회사의 업무를 행함에 있어 현실에 빠르게 대응하고 그 빠른 대응을 통해 안정적이고 편리하고 보안성이 인정되는 정보시스템을 구축함으로서 효과적인 업무를 통해 개인의 발전 또는 회사의 이익을 도모한다는 것이다.

데이터베이스 시스템의 구성

데이터베이스의 시스템이란 쉽게 말하자면 데이터베이스와 그 운영에 필요한 모든 주위 환경이라고 생각하면 된다. 데이터베이스 시스템에는 네 가지의 구성 요소를 가진다.

1. 데이터베이스
2. 데이터베이스를 관리하기 위한 운영도구
3. 하드웨어
4. 사용자

그런데 주변에서 우리는 데이터베이스라고 하면 DBMS(Data Base Management System)을 칭하는 경우도 적지 않다 그러나 그것은 정확히 다른 것이다. 데이터베이스는 연관성이 있는 데이터의 집합 즉 여러분들이 수집한 하여 입력된 자료들의 집합이며 그 하나의 자료에는 뚜렷한 의미가 있다는 것이다. 그럼 DBMS는 결론부터 이야기하자면 DBMS는 사용자들로 하여금 데이터베이스를 생성하고 유지하도록 제작된 프로그램이라는 것이다. 여러분은 이 부분을 잊지 말고 정확히 알고 있기를 바란다.



▶ 데이터베이스 시스템

이런 데이터베이스 시스템은 독립적인 기능을 갖는 다섯 가지 기능을 지닌 요소로 구성된다.

1. 파일관리기

디스크의 기억공간의 할당과 디스크에 정보를 저장하기 위한 데이터의 구조를 관리

2. DBMS

데이터베이스 내에 저장된 데이터와 시스템에 사용되는 프로그램과 쿼리(sql)간의 인터페이스를 제공

3. sql 처리기

쿼리(sql)를 DBMS 프로그램이 인식하는 명령어로 변환 또한 sql 처리기는 사용자의 요구에 맞게 변환

4. 프리 컴파일러

프로그램 내에 저장된 데이터 조작어를 호스트 언어로 표현된 프로시저로 호출 변환하는 것으로 프리 컴파일러는 적당한 코드를 생성하기 위해 쿼리 처리기와 상호 통신한다.

5. 컴파일러

데이터 정의어를 메타 데이터를 갖는 테이블로 변환하는 기능을 갖는다. 여기서 데이터 정의어란 DDL(Data Definition Language)로 데이터와 그 구조를 의미한다. 즉 데이터베이스 객체를 생성하는 CREATE 데이터베이스를 삭제하는 DROP 기존의 정의된 데이터베이스 객체를 다시 정의하는 ALTER를 말하며 메타 데이터란 쉽게 데이터를 꾸미기 위한 데이터, 즉 데이터를 빨리 찾기 위해 만들어진 데이터라 할 수 있다.

우리가 책의 내용을 찾을 때 책 앞의 목차나 책 뒤의 색인을 보고 찾으면 빨리 찾을 수 있듯이 우리가 많이 사용하는 데이터베이스도 이러한 메타 데이터가 잘 구성되어 있기 때문에 빨리 데이터를 찾을 수 있다. 데이터 정의어를 알았다면 데이터 제어어, 데이터 질의어에 대해서도 알아야 한다. 이를 간략히 알아보면 다음과 같다.

가) 데이터 정의어-DDL (Data Definition Language) : 데이터와 그 구조를 정의

sql	질의어
create	테이블의 생성
drop	테이블의 삭제
alter	기존 데이터베이스의 객체 재정의

나) 데이터 질의어-DML (Data Manipulation Language) : 데이터의 검색과 수정

sql	질의어
insert	데이터베이스 객체에 데이터 삽입
delete	데이터베이스 객체에 데이터 삭제
update	데이터베이스 객체에 데이터 수정
select	데이터베이스 객체에 데이터 검색

다) 데이터 제어어-DCL (Data Control Language) : 데이터베이스 사용자의 권한

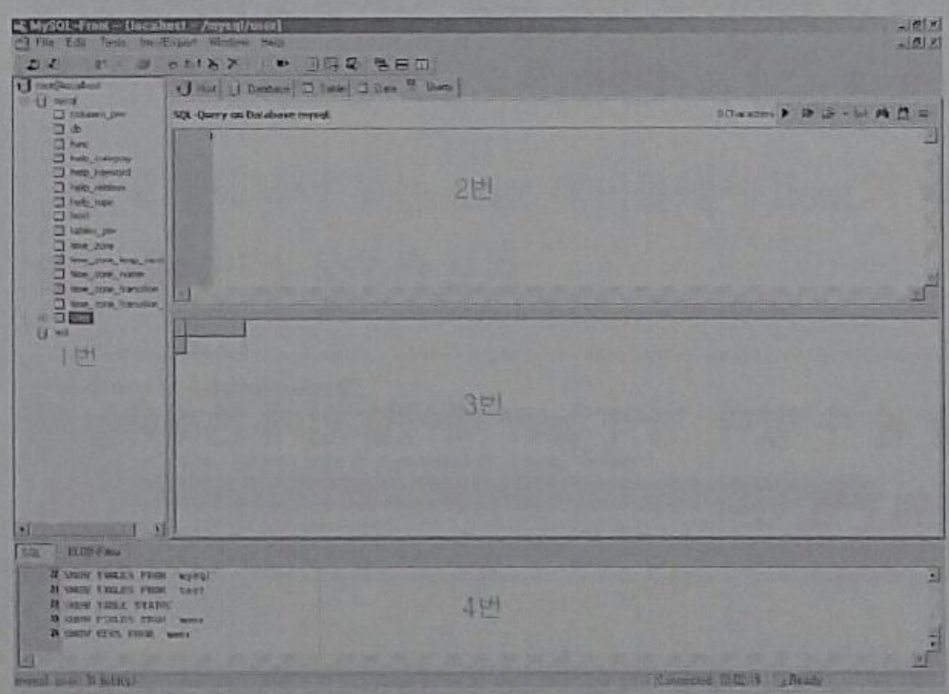
sql	질의어
grant	데이터베이스 객체에 권한
revoke	주어진 권한 취소

기본 SQL

데이터베이스 내에 수많은 데이터를 사용자가 원하는 또는 고객이 원하는 자료를 만들고 그 자료를 활용하기 위해서는 데이터베이스를 질의할 수 있는 질의어가 필요하다. 이를 쿼리(SQL)라고 한다. 쿼리는 기본적으로 SELECT, DELETE, INSERT, UPDATE가 있으며 이에 따른 조건절인 WHERE가 있다.

MySQL-Front를 실행하여 자신의 컴퓨터에 설치된 MySql에 접속한다. 그런 후 mysql이라는 데이터베이스 내에 있는 user라는 테이블을 선택하고 화면중앙에 있는 탭 메뉴중 Query메뉴를 선택한다.

1번은 현재 접속되어있는 데이터베이스 내에 만들어져 있는 데이터베이스와 테이블을 보여주고 2번은 쿼리를 작성하는 곳이며 3번은 쿼리의 결과 값을 뿌려주는 곳이다. 4번은 실행한 쿼리의 성공 및 작성된 쿼리 수행 시 에러 메시지를 보여주는 상태영역이다. 작성되는 쿼리는 앞으로 2번에 작성하면 된다.



▶ MySQL-Front의 화면 구성

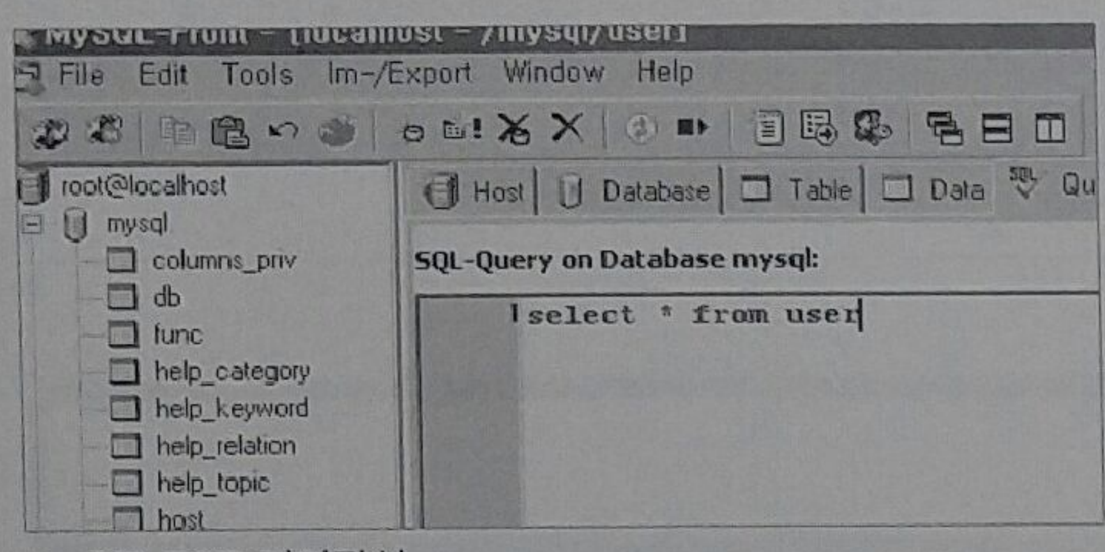
SELECT

데이터베이스 내의 테이블에서 컬럼의 값을 뽑아 세로로 보여준다.

```
SELECT 컬럼 FROM 테이블명 (WHERE 조건)
```

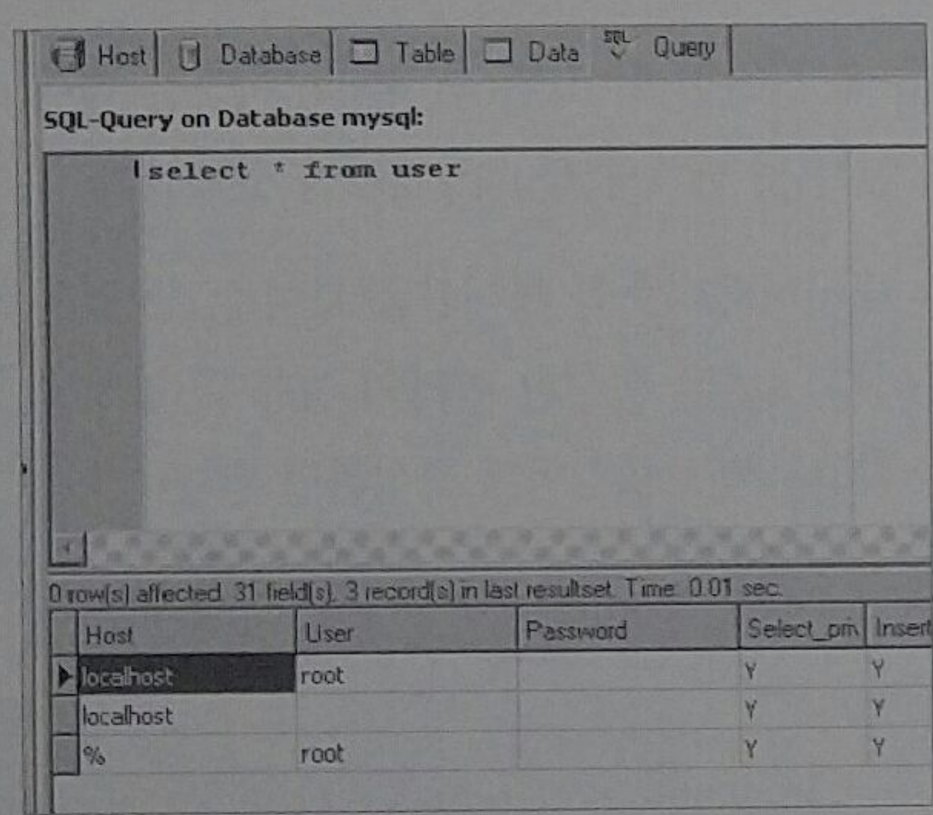
테이블 내의 모든 컬럼을 보기 위해서는 *으로 표시하며 SELECT * FROM 테이블 명을 주면 선택한 테이블의 모든 데이터를 보여준다.

쿼리 작성문에 SELECT 문을 입력하여보자



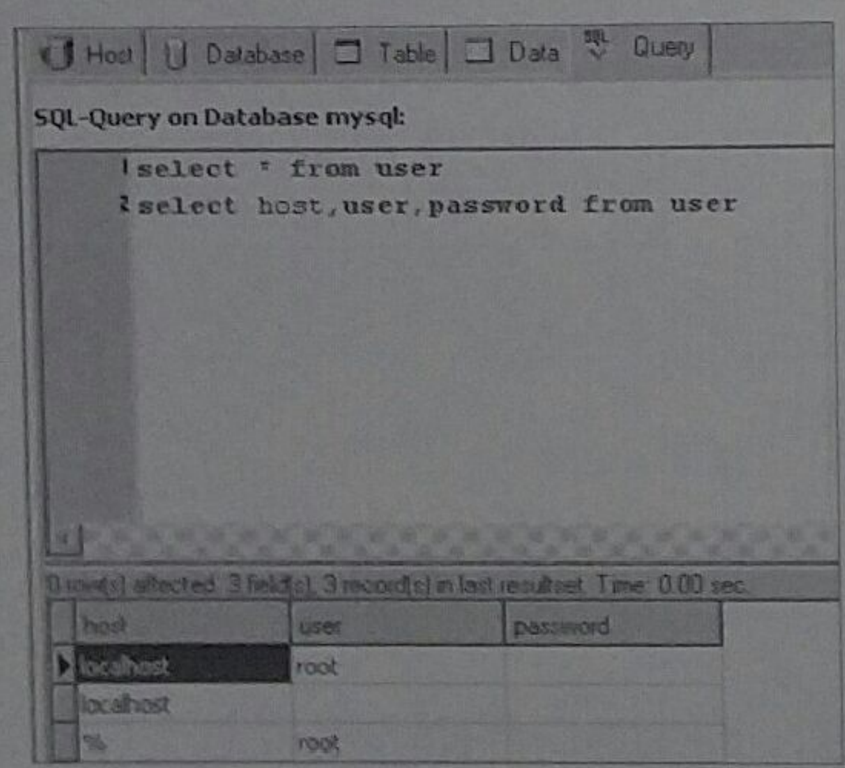
▶ SELECT 쿼리작성

작성된 쿼리문의 실행은 F9번을 눌러 실행하거나 또는 ▶ 으로 실행한다.



▶ 쿼리 수행 결과

*를 함으로서 user 테이블에 있는 모든 컬럼의 값을 볼 수 있다. 만약 user 테이블 안에 존재하는 특정한 필드만을 보려고 한다면 * 대신 컬럼의 이름을 적어두면 된다.

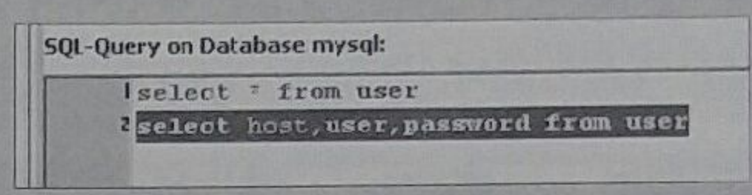


▶ 특정 컬럼의 select

TIP

쿼리문의 실행방법

MySQL-Front에서 여러개의 쿼리문을 작성하고 해당 쿼리문만을 실행하려면 해당 쿼리문을 선택하고 '▶'을 실행하던지 Ctrl+F9를 눌러 실행한다. 해당 쿼리문 만을 선택하려면 마우스로 쿼리문을 드래그하여 선택 또는 Shift 키를 누른 상태에서 방향키 또는 home, end 키를 이용하면 된다. 선택된 쿼리문은 MySQL-Front에서 파란색으로 바탕으로 보인다.



▶ 쿼리문 선택

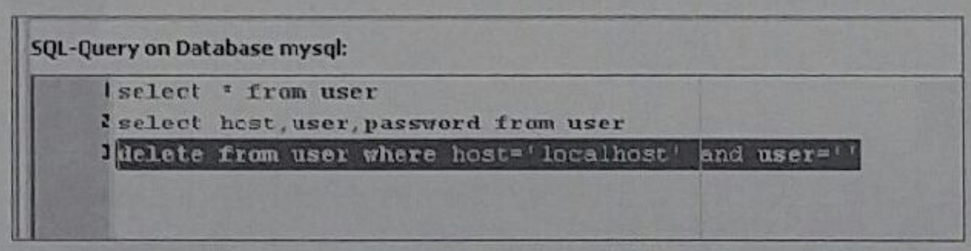
DELETE

특정 레코드를 삭제한다. DELETE는 테이블 안에 존재하는 데이터 중 사용자가 삭제하고자 하는 데이터의 조건에 맞게 삭제 하거나 또는 모든 데이터를 삭제할 수 있다. 삭제함에 있어서는 보다 신중을 기하는 것이 좋다.

```
DELETE FROM 테이블명 (WHERE 조건)
```

SELECT와 다르게 *은 사용함에 있어 신중을 기해야 한다. 만약에 *을 준다면 모든 데이터들이 한 순간에 사라질 것이다. 물론 모든 데이터를 삭제하고자 한다면 문제는 없겠다. 하지만 지워진 데이터는 MySql에서 특별한 타입으로 선언하지 않는 이상 reload 되지는 않기 때문에 주의해야 한다.

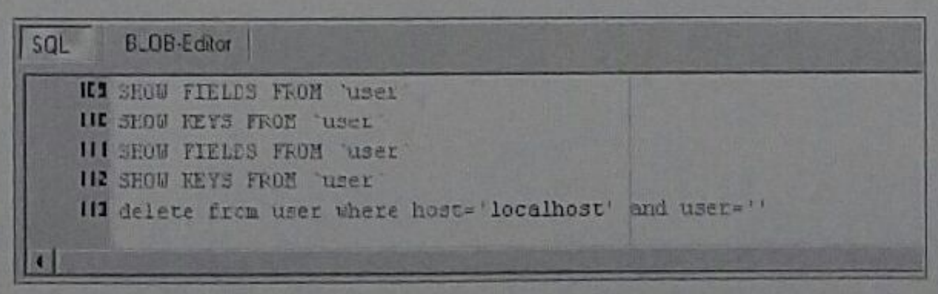
현재 선택되어 있는 user 테이블에서 host의 데이터의 값이 localhost를 가지고 user의 데이터 값이 없는 데이터를 삭제해 보자.



▶ DELETE 쿼리작성

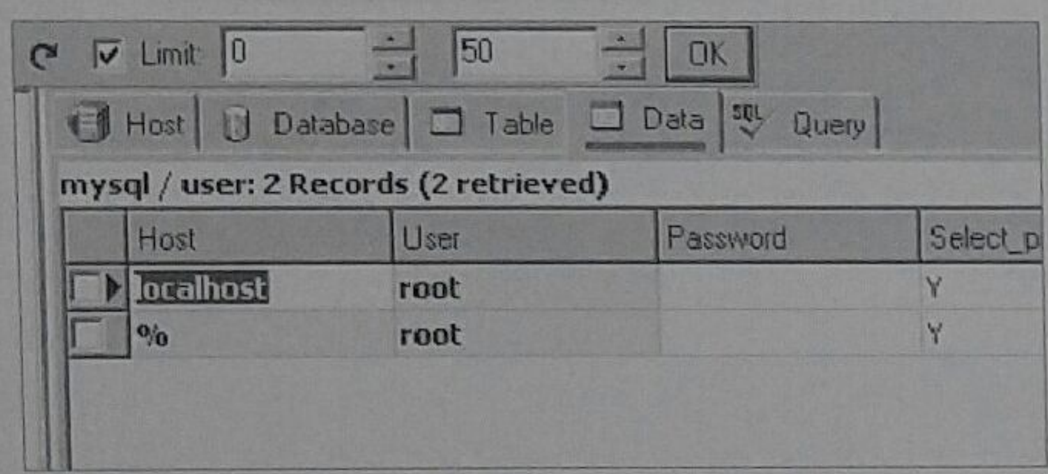
다음 장에서 자세히 배우겠지만 Where는 조건절로 기본 쿼리문에 필수적으로 따라 다니는 조작어이다. where 뒤에는 여러개의 조건이 올 수 있으며 여러개의 조건이 올 때는 and로 조건을 이어준다. 즉 'where 칼럼1=조건 and 컬럼2=조건' 이 되는 것이다.

여기서 Delete 쿼리를 보면 host 칼럼의 데이터 값들 중 localhost라는 이름을 가진 데이터와 user 칼럼의 값이 없는 것을 찾아 삭제하는 것이다. '' 으로 값을 지정한 이유는 나중에 이야기 하겠지만 테이블을 생성할 때 각 필드에 대한 속성 값 중 문자타입으로 선언하였기 때문이다. 쿼리를 실행하면 수행 상태 창에 쿼리에 대한 에러 등 메시지를 출력하게 된다.



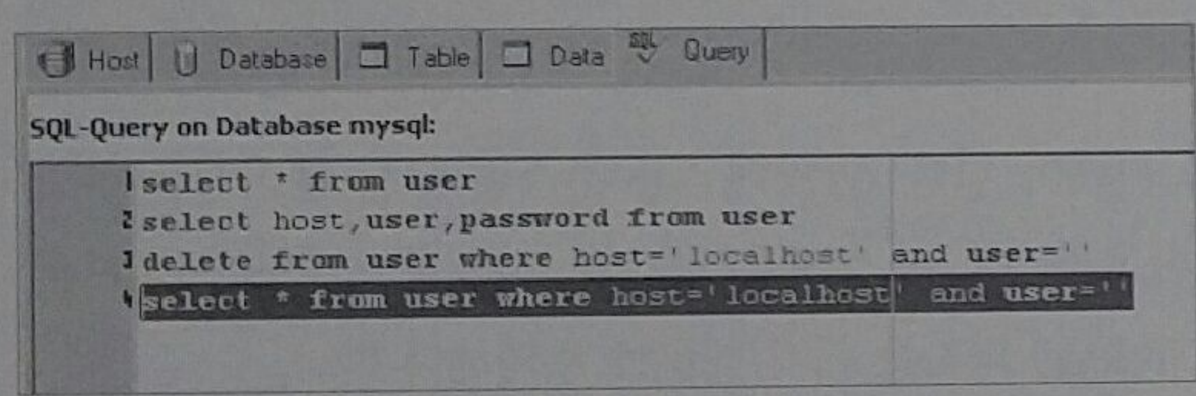
▶ 쿼리 상태영역

삭제되었는지를 확인하기 위한 방법은 2가지가 있다. 하나는 다시 SELECT 쿼리를 작성하여 실행하는 방법과 MySQL-Front의 탭 메뉴 중에서 data 메뉴를 선택하여 확인하는 방법이 있다. 그러나 탭 메뉴를 선택하여 확인하는 방법은 데이터의 양이 적을 때 확인하는 것이 좋으며 일반적으로 SELECT문을 작성하여 확인한다.



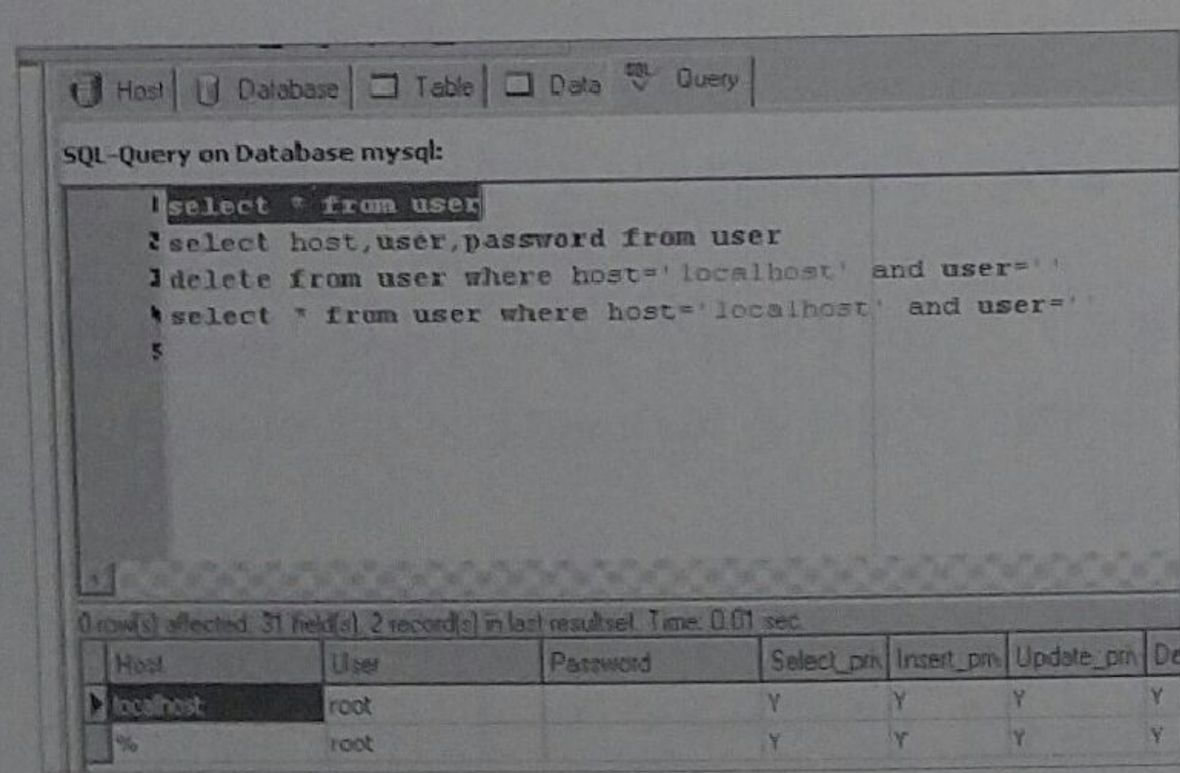
▶ 탭 메뉴 data에서의 확인

SELECT문을 통한 확인 시에는 삭제시의 조건을 같이 주게 되면 더 확실한 정보를 얻을 수 있다. MySQL-Front에서 SELECT시 데이터의 값이 없으면 하늘색으로 결과가 보인다.



▶ SELECT문을 통한 확인

다시 * 옵션으로 통해 user 테이블의 데이터를 확인해 보면 삭제된 것을 확인할 수 있다.



▶ *를 통한 결과확인

INSERT

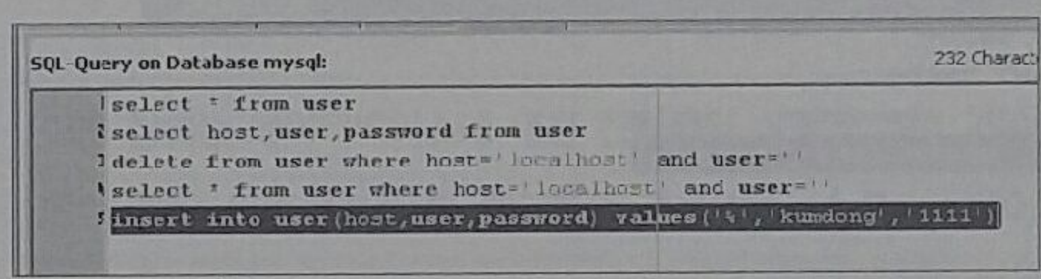
테이블에 데이터를 입력할 때 사용한다.

```
INSERT INTO 테이블명 (컬럼1, 컬럼2, .....) VALUES(컬럼값1, 컬럼값2..)
```

또는

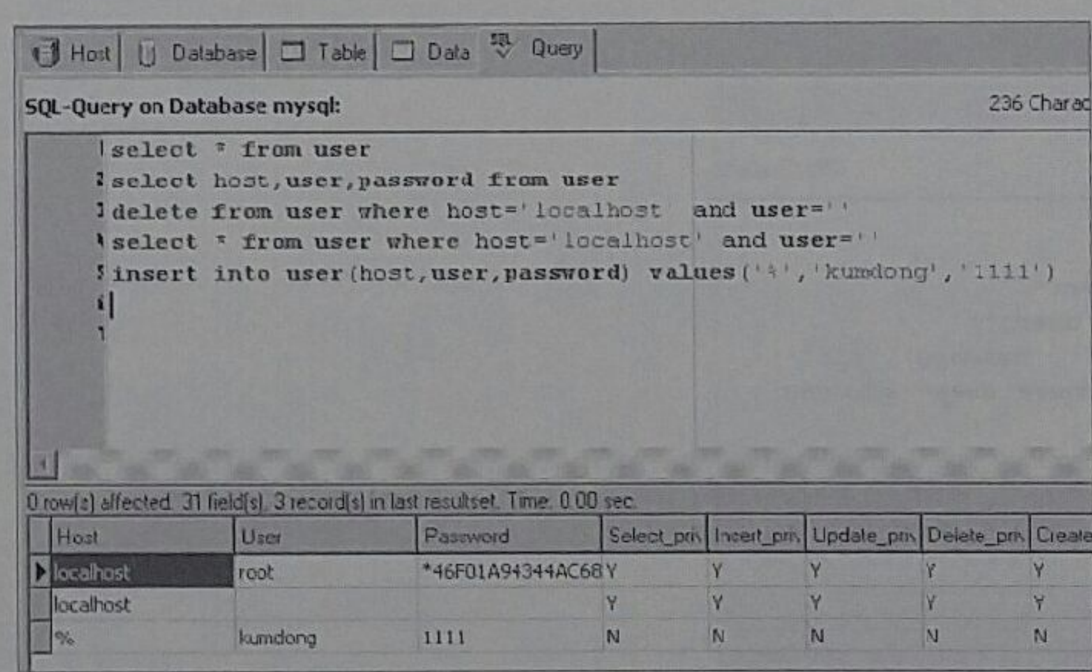
```
INSERT INTO 테이블명 VALUES(컬럼값1, 컬럼값2..)
```

이 방법의 단점은 원래 INSERT하고자 하는 테이블의 모든 컬럼의 값을 다 주어야 한다는 점이다. 현재 선택되어 있는 user 테이블에 값을 입력해 보자.



▶ user 테이블에 INSERT

그런 후 SELECT 쿼리를 이용하여 결과를 확인해 보자.



▶ INSERT 결과 확인

값이 없을 때는 기본값으로 설정

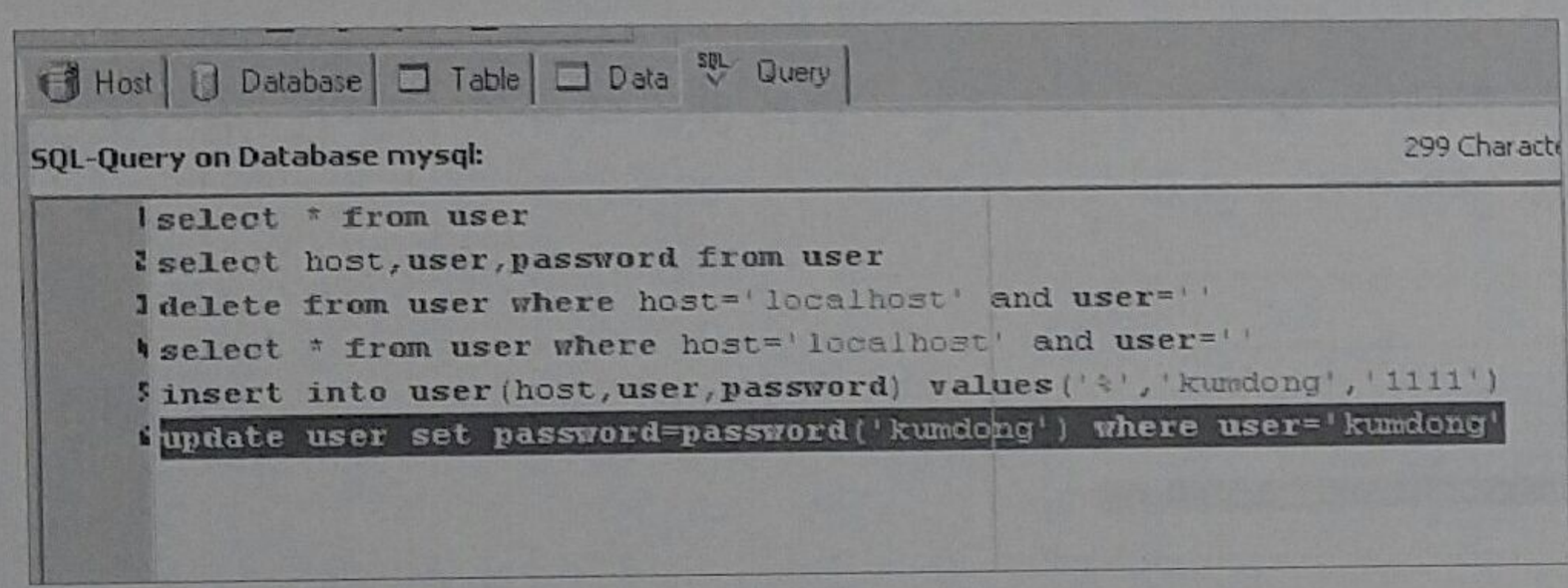
한 가지 의아해 하는 것은 host, user, password 칼럼을 제외하고 나머지 칼럼의 vlaues값을 주지 않았음에도 값이 입력이 되었다는 것이다. 이는 user 테이블의 칼럼이 세 개의 칼럼을 제외한 나머지 부분에 기본 값을 지정했기 때문이다. 1장에서 배운 테이블의 정보를 보여주는 탭 메뉴를 통해보면 알 수 있다.

UPDATE

테이블에 입력된 데이터를 수정하는 질의어이다. 즉 어떤 데이터의 값이 변동이 이루어졌다면 반드시 UPDATE 문을 사용하여 수정한다.

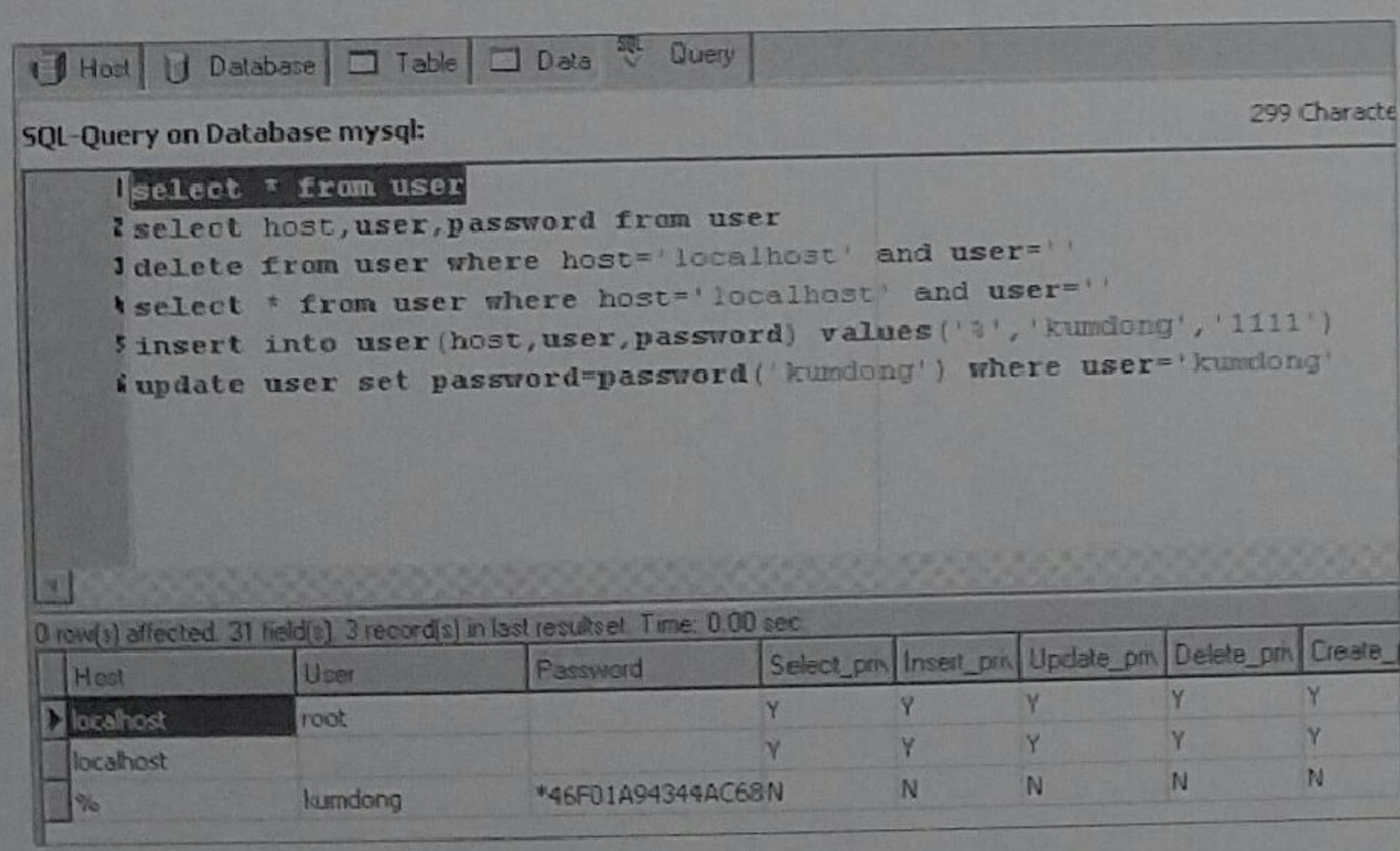
```
UPDATE  테이블명 SET 컬럼1='변경값' , 컬럼2='변경값' ...(WHERE 조건)
```

user 테이블의 데이터 중에서 user 컬럼의 값이 kumdong인 데이터의 암호를 변경한다.



▶ kumdong의 암호변경

변경 후 SELECT를 통해 변경된 값을 확인한다.



▶ 변경확인

TIP

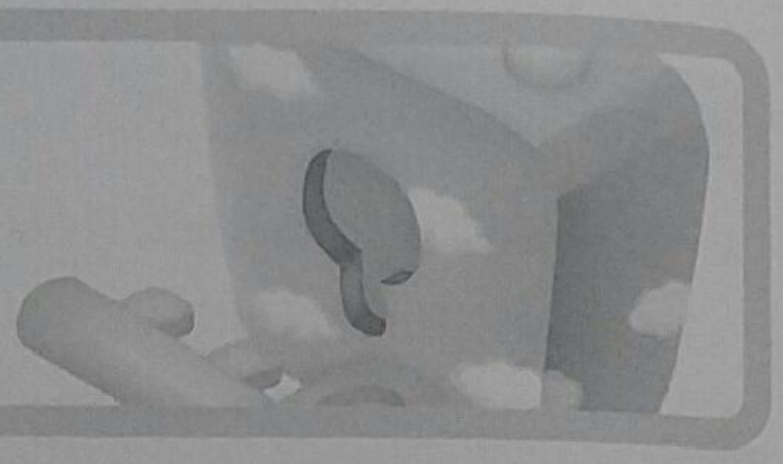
암호를 변경하고 싶다면

1. 처음 MySql이 설치되고 root의 암호를 변경해야 해야 한다. root가 관리자이기 때문에 보안성을 위해 그 암호를 지정하고 변경하는 것이 좋다.
2. root의 암호변경은 MySQL-Front에서 작업하는 것이 아니라 cmd 모드에서 작업을 해야 한다. 부록에 나와 있는 부분을 꼭 참고하여 실행하기 바란다.
3. 암호를 변경할 때 update문 사용 시 password=password(")를 사용하는데 이는 입력되는 값이 암호화하여 입력을 시킴으로서 보안성을 강조하기 위한 것이다.

맺음말

이번 장에는 데이터베이스에 대한 전반적인 설명과 그에 기본이 되는 기본 SQL에 대해서 공부했다. 다음 장에서는 데이터베이스의 생성/삭제, 테이블의 생성/삭제 MySql의 데이터형, 칼럼의 생성 등에 대해서 공부한다.

03 MySql의 기초 세우기



안녕하세요

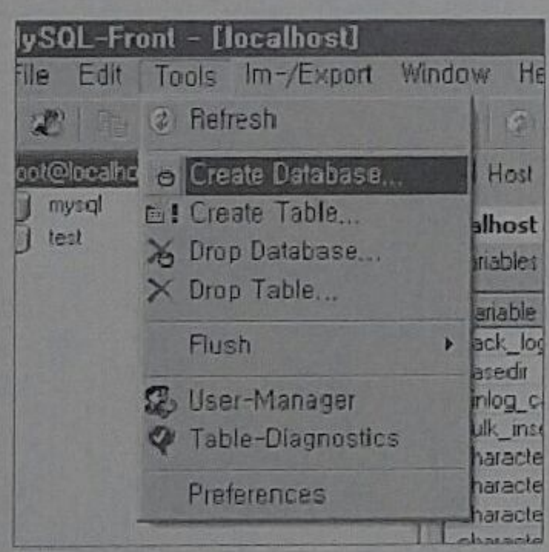
이번 장에서는 2장에서 배웠던 기본 SQL를 좀 더 숙지하고 MySQL-Front에서의 데이터베이스 생성/삭제, 테이블의 생성/삭제와 MySql에서의 테이블 생성 시에 해당되는 데이터 타입과 MySql에서 제공되는 테이블의 타입에 대해서 공부한다. 이 부분은 MySql에서의 가장 기초적인 부분의 시작이므로 관심 있게 보기 바란다.

데이터베이스의 생성

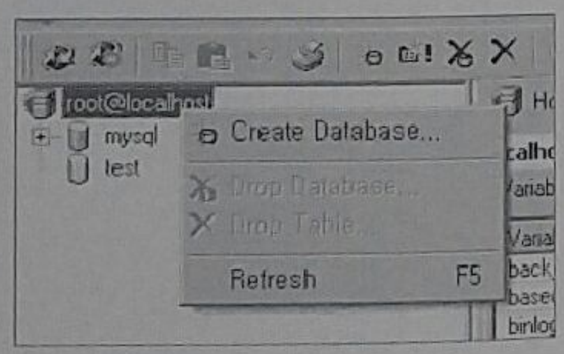
Front를 통해 MySql에 접속하여 Tools 메뉴의 create database를 실행한다. 또는 root@localhost를 선택하여 마우스의 오른쪽 버튼을 클릭하여 팝업 메뉴에 있는 create database를 선택해도 무방하다.

따라하기 데이터베이스 생성

❶ 사용자만의 데이터베이스를 만들기 위해서는 Tools에서 Create Database를 실행한다.

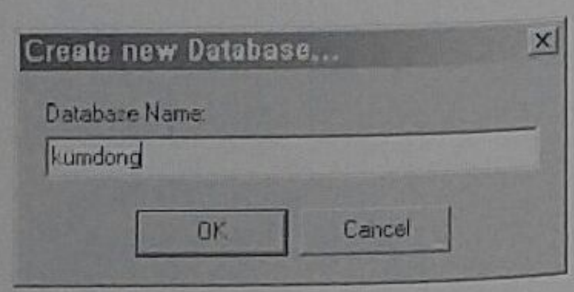


▶ Create Database의 실행



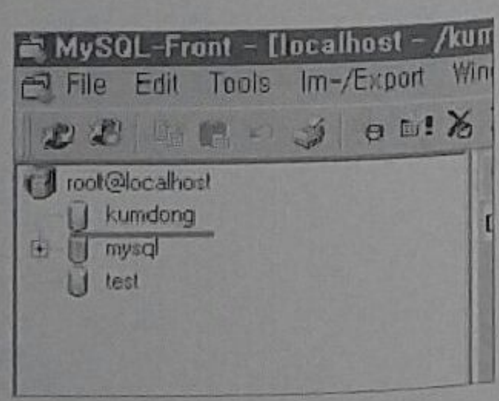
▶ 팝업메뉴 Create Database

❷ 생성할 데이터베이스의 이름을 입력하고 ok 버튼을 클릭한다.



▶ 데이터베이스 이름 입력

❸ kumdong이라는 데이터베이스가 생성된 것을 확인한다.



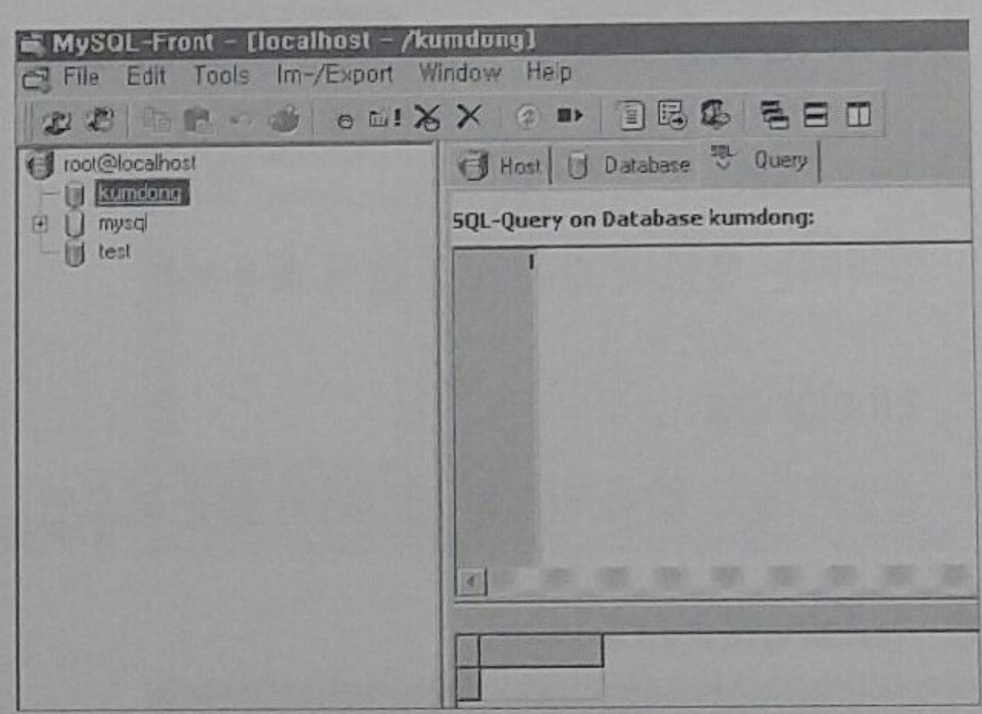
▶ 생성된 데이터베이스의 확인

데이터베이스의 삭제

삭제하고자 하는 데이터베이스를 선택하고 Tools 메뉴의 Drop Database를 선택하면 생성된 또는 기존의 데이터베이스를 삭제할 수 있다. 조심해야 할 것은 MySql에서 기본적으로 제공되는 mysql 데이터베이스는 삭제하면 안 된다는 점이다.

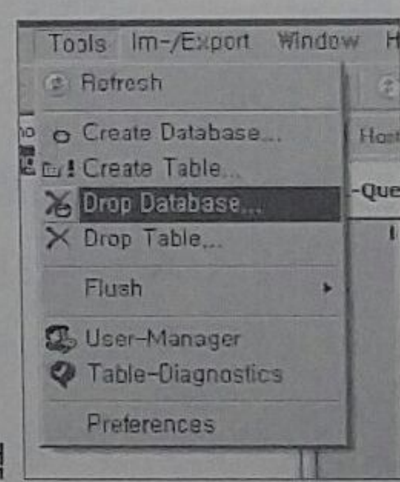
따라하기 데이터베이스 삭제

❶ 삭제할 데이터베이스를 선택한다.



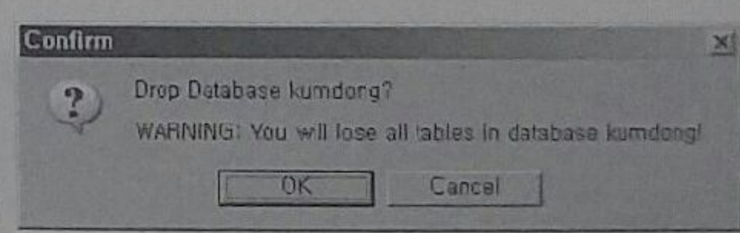
▶ 삭제할 데이터베이스의 선택

❷ Tools 메뉴의 Drop database를 선택한다.



▶ Drop database의 선택

❸ 경고 메시지와 함께 삭제할 것인가를 묻는다. ok 버튼을 클릭하면 선택한 데이터베이스는 삭제된다.



▶ 삭제 경고 메시지

테이블의 생성

테이블은 선택된 데이터베이스 내에 생성해야 한다. 따라서 새로운 데이터베이스를 생성한 후 생성한 데이터베이스를 선택한다. 테이블에는 여러 가지의 속성이 있다. 그 속성을 살펴보면 다음과 같다.

1. Primary

테이블에 입력되는 값이 중복여부를 결정한다. 만약 생성되는 필드를 Primary로 선언한 후 같은 값을 입력하면 에러가 발생한다.

2. Index

쉽게 이야기하면 색인이라고 생각하면 된다. 보다 쉽게 원하는 데이터를 찾기 위해 설정한다.

3. Unique

입력되는 데이터의 유일성을 보장하는 것으로 절대로 중복되는 데이터가 생길 수 없다.

4. Not Null

입력되는 필드의 값에 null 값의 허용여부

5. Unsigned

생성된 필드 값이 지정된 데이터의 값을 넘으면 그 넘은 값만을 저장한다.

6. AutoIncrement

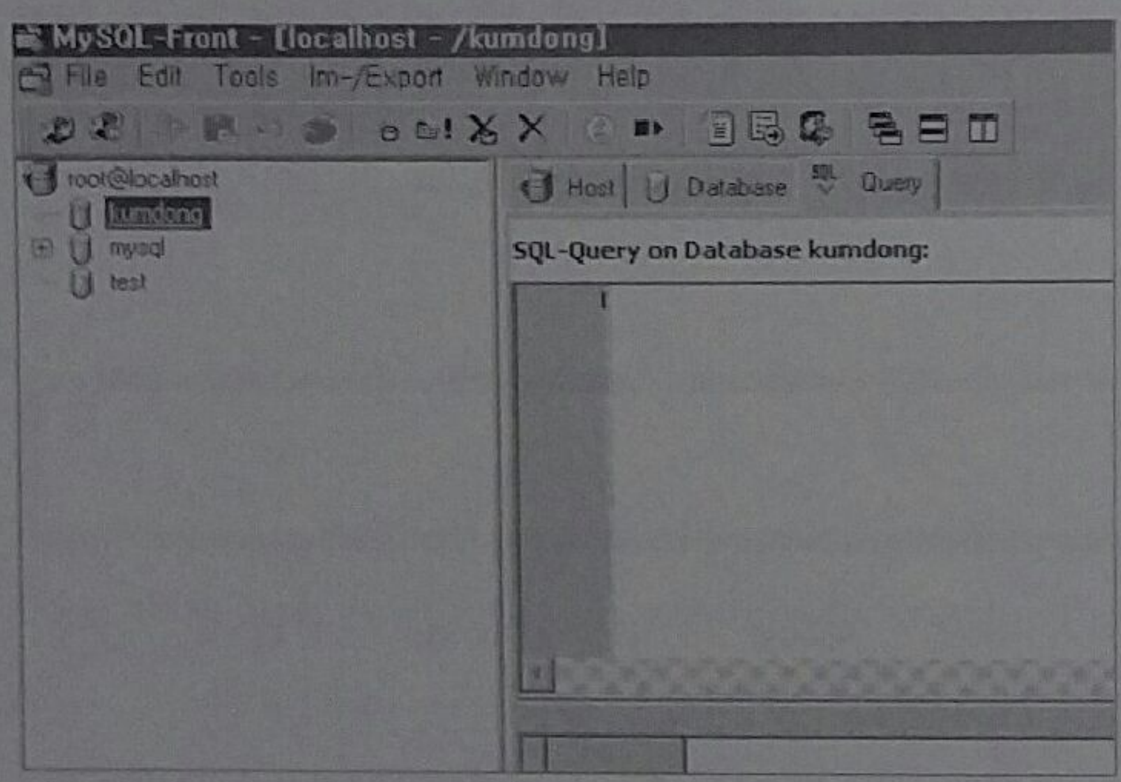
생성되는 필드의 값을 입력하지 않아도 자동 증가 값으로 지정한다.

7. Zerofill

생성된 필드에 입력되는 데이터의 값이 지정한 자리 수만큼 0으로 채워지게 한다. 즉 int 5 자리에 입력 값이 20이라면 00020 이런식이 된다.

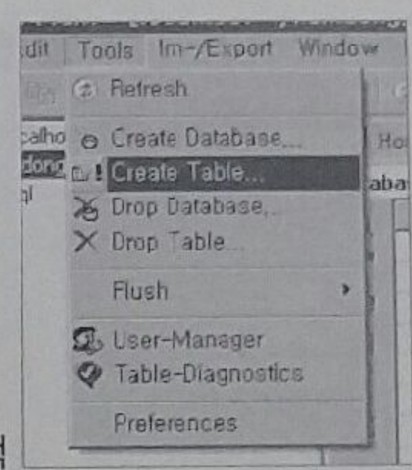
테이블 생성

❶ 생성된 데이터베이스를 선택한다.



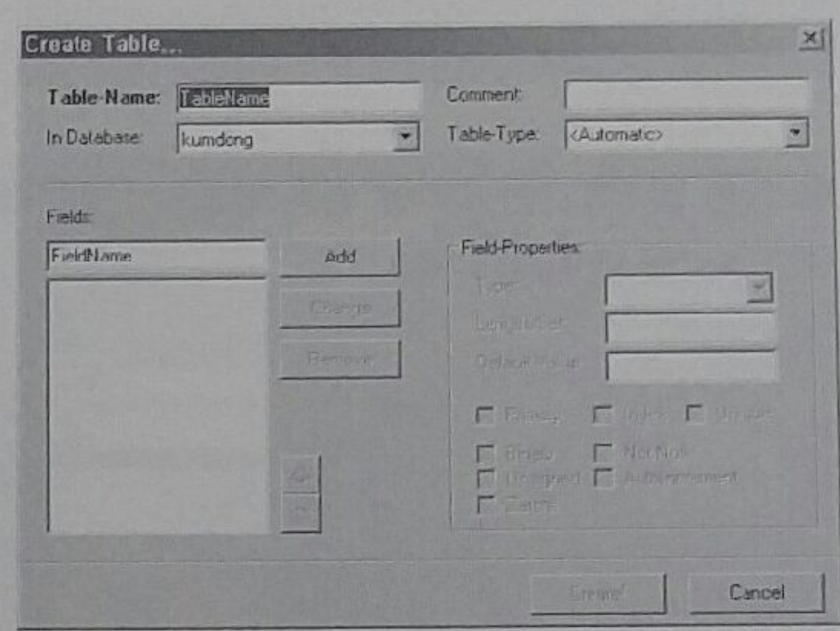
▶ 생성된 데이터베이스의 선택

❷ Tools 메뉴 중 Create table을 선택한다.



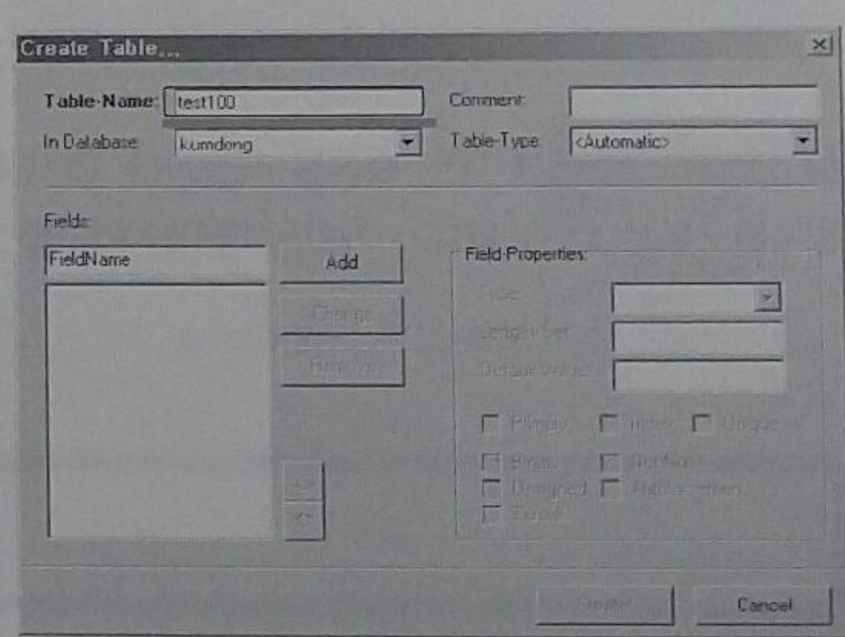
▶ Create table의 선택

❸ 생성할 테이블에 대한 테이블 이름, 필드 등 속성을 지정하는 창이 출력된다.



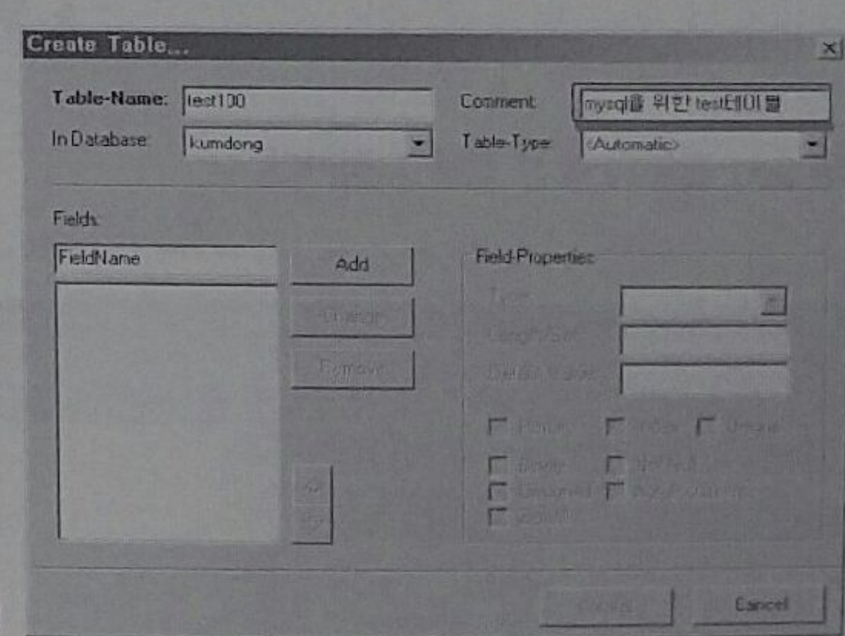
▶ 테이블 내 칼럼 생성 및 속성입력창

❹ 생성할 테이블의 이름을 입력한다.



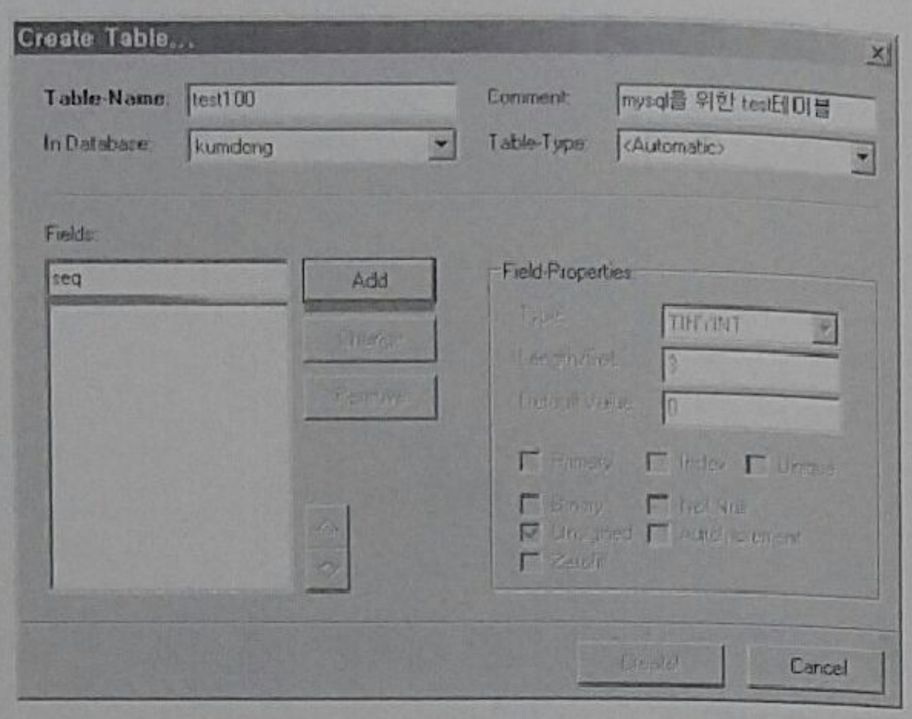
▶ 테이블 이름 입력

❺ Comment를 입력한다. Comment는 생성되는 테이블에 대한 일종의 정보를 입력하는 것으로 입력을 안 해도 상관은 없지만 가급적이면 생성되는 테이블이 어떤 테이블인지를 알기 위해선 입력하는 것이 좋다.



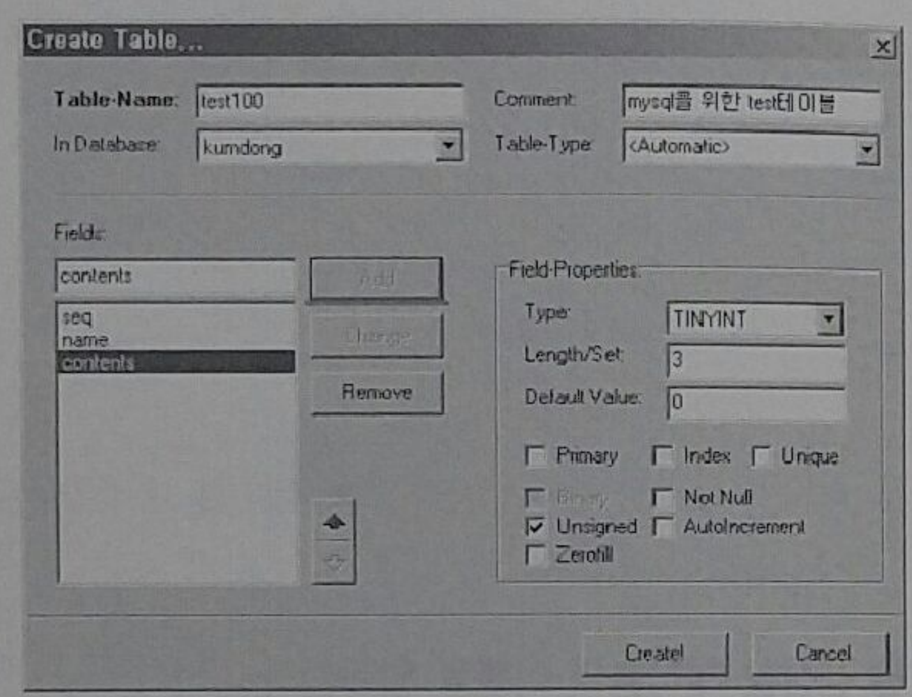
▶ Comment의 입력

❻ 테이블 내에 생성할 필드 즉 칼럼을 생성하는 곳으로 생성할 필드의 이름을 입력하고 Add 버튼을 클릭하면 필드가 생성이 된다.



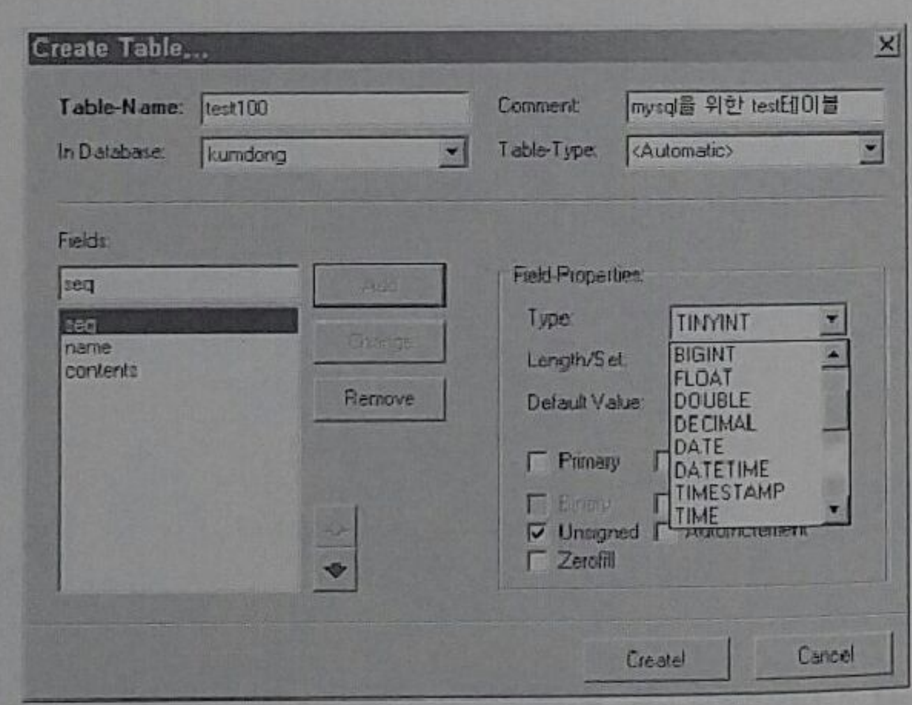
▶ 필드의 생성

❼ 다른 필드를 생성하려면 같은 방법으로 필드의 이름을 입력하고 Add를 클릭하면 된다.



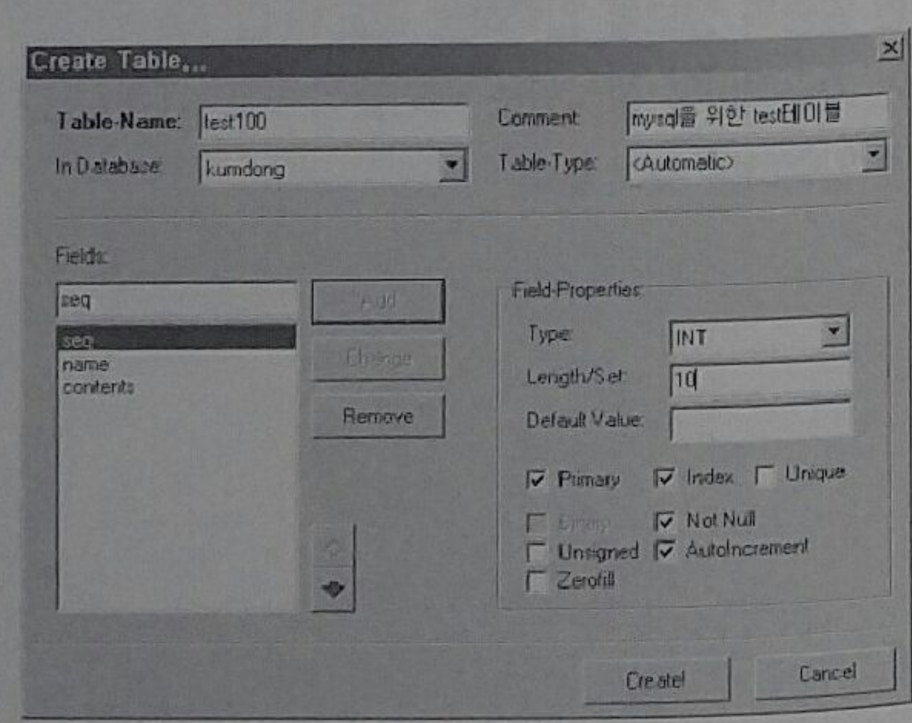
▶ 여러 필드의 생성

❽ 생성된 필드들에 대한 속성을 지정한다. 먼저 속성을 지정할 필드를 선택하고 그 필드가 가지는 데이터 타입을 지정한다.



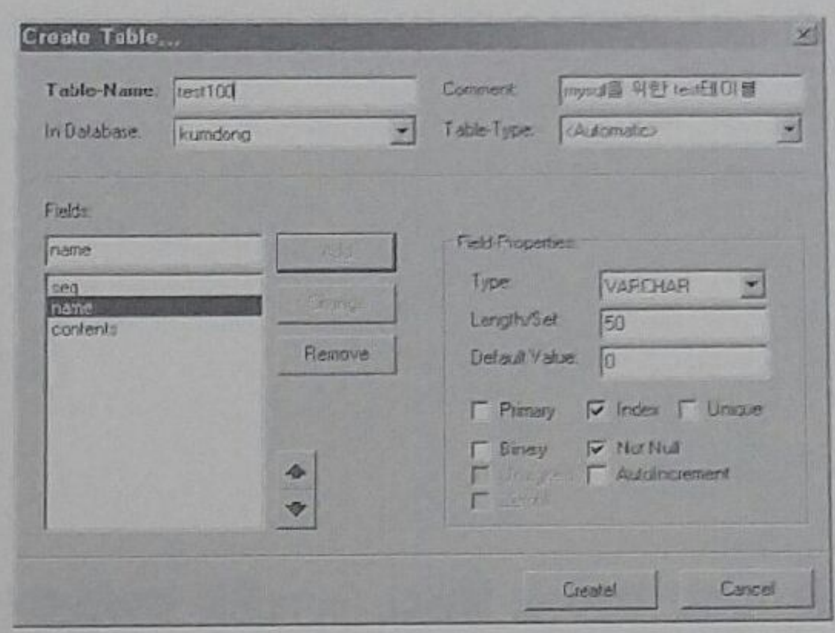
▶ 필드의 데이터 타입

❾ seq 필드의 속성 값을 지정한다.



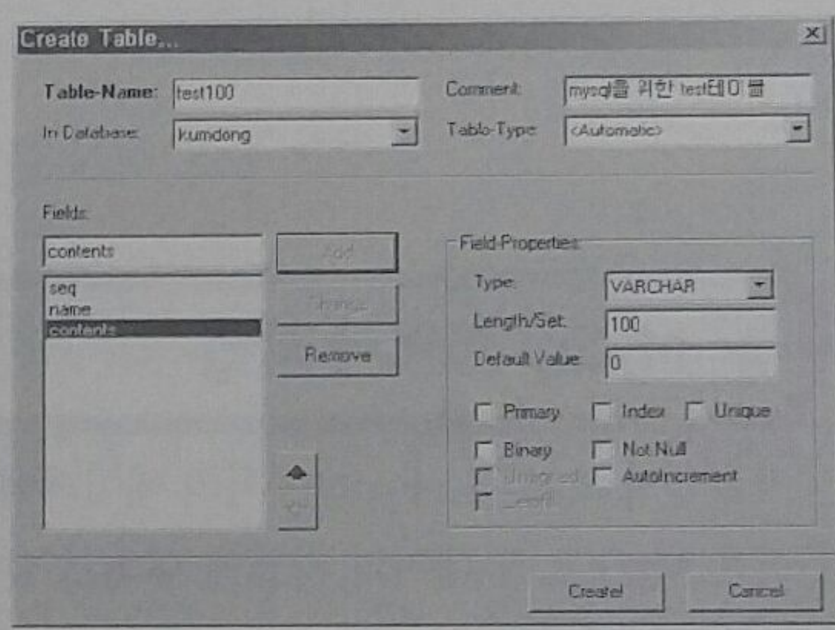
▶ seq 필드의 속성 값 지정

⑩ name 필드의 속성 값을 지정한다.



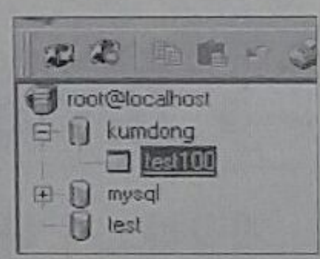
▶ name 필드의 속성 값 지정

⑪ contents의 속성 값을 지정한다.



▶ contents 필드의 속성 값 지정

⑫ create 버튼을 클릭하여 테이블을 생성한다.



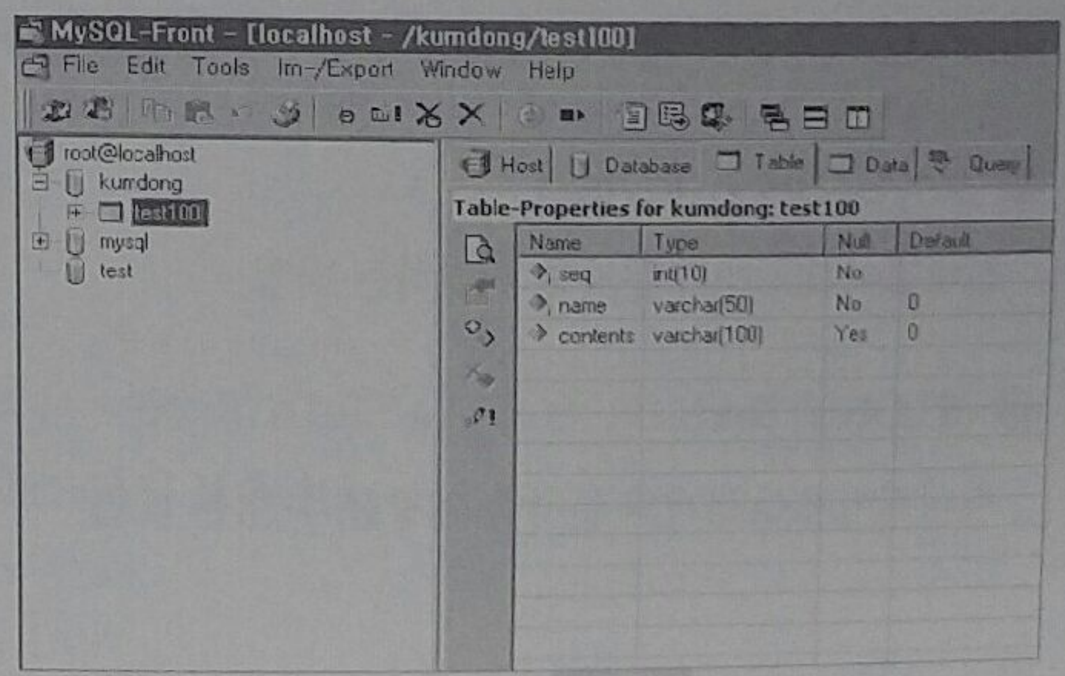
▶ 테이블의 생성 확인

테이블의 삭제

생성된 테이블이 불필요할 때 사용한다. 대부분 삭제 시에는 혹시 있을지 모르는 상황에 대비하여 데이터를 백업 받는 것이 좋다. 백업에 대한 부분은 MySql의 백업 부분에서 배우게 된다.

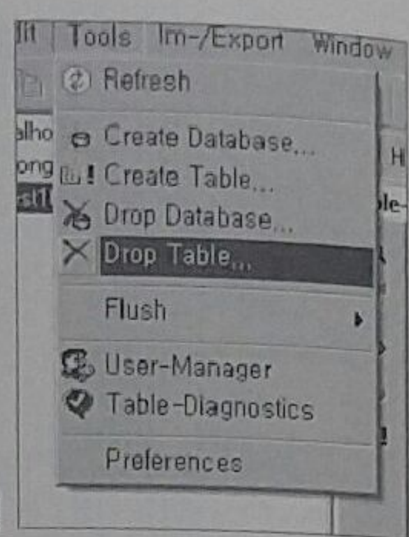
따라하기 테이블 삭제

❶ 삭제하려는 테이블을 선택한다.



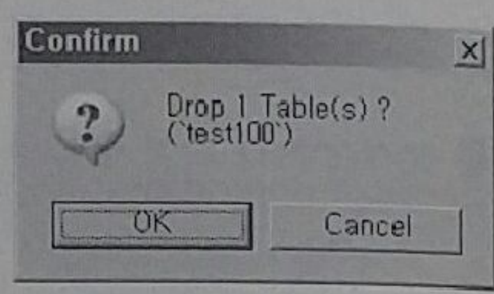
▶ 삭제 테이블의 선택

❷ Tools 메뉴의 Drop table을 선택한다.



▶ Drop table의 선택

❸ 삭제할 것인가를 묻는다. ok 버튼을 클릭하면 선택한 테이블이 삭제된다.



▶ 테이블의 삭제

MySql 데이터 타입

지금까지 MySQL-Front에서의 데이터베이스 생성, 삭제 테이블의 생성, 삭제에 대해서 살펴보았다. 다음은 MySQL의 데이터 타입에 대해서 살펴보기로 하자. 먼저 MySql의 타입을 종합해 보면 세 가지로 나눌 수 있다.

문자형

저장하고자 하는 데이터에 문자를 저장시키기 위해서는 테이블 내 필드의 속성이 반드시 문자형으로 지정이 되어야 한다.

char(m)

고정길이 문자열을 표현하는 타입으로 m=1~255으로 타입을 지정 시 Binary로 지정하지 않으면 대소문자를 구분하지 않는다.

varchar(m)

가변길이 문자열을 표현하는 타입으로 m=1~255이며 char보다 검색속도가 느리다. 또한 타입을 Binary로 지정하지 않으면 대소문자를 구분하지 않는다.

tinyBLOB / tinytext

최대길이 255개의 문자를 저장하는 타입

BLOB / Text

최대길이 63535개의 문자를 저장할 수 있는 타입

mediumBLOB / mediumText

최대길이 16777215의 문자를 저장할 수 있는 타입

LongBLOB / LongText

최대길이 4294967295개의 문자(4G)를 저장할 수 있는 타입

ENUM('문자열1', '문자열2'....)

목록 형으로 저장되는 문자열의 값이 목록 중에서 오직 하나 만을 저장

set('문자열1', '문자열2'....)

ENUM과 다른 게 저장되는 문자열이 0 또는 1개 이상의 문자열의 값을 저장

숫자형

저장하고자 하는 데이터에 숫자를 저장시키기 위해서는 테이블 내 필드의 속성이 반드시 숫자형으로 지정이 되어야 한다.

tinyint

저장 공간이 1byte 정수형으로 부호 있는 정수는 -128에서 127까지, 부호 없는 정수는 0-255까지 표현한다.

smallint

저장 공간이 2byte 정수형으로 부호 있는 정수는 -32768~32768까지, 부호 없는 정수는 0~65535까지 표현된다.

mediumint

저장 공간이 3byte 정수형으로 부호 있는 정수는 -8388608~8388607까지, 부호 없는 정수는 0~16777215까지 표현된다.

int / integer

저장 공간이 4byte 정수형으로 부호 있는 정수는 -2147483648~2147483647까지, 부호 없는 정수는 0~4294967285까지 표현된다.

bigint

저장 공간이 8byte 정수형으로 부호 있는 정수는 -9223372036854775808~9223372036854775807까지 부호 없는 정수는 0~18446744073709551615까지 표현된다.

float

저장 공간이 4byte 실수형으로 단일 정밀도를 가진 부동소수점 표현으로 -3.402823466E+38~3.402823466E+38까지 표현된다.

double

저장 공간이 8byte로 2배의 정밀도를 가진 부동소수점 표현으로-1.79769313486231517E+308~1.79769313486231517E+308까지 표현된다.

decimal

char 형태로 동작하는 부동소수점 실수형으로 숫자가 아닌 문자 형태로 저장이 된다.

날짜형

저장되는 데이터가 일자인 경우 사용되며 주로 입력의 형식은 date를 사용하고 시분초까지 저장하려면 datatime보다는 timestamp을 주로 사용한다.

date

날짜를 표현하는 유형으로 1000-01-01~9999-12-31까지 표현이 되며 기본적으로 'YYYY-MM-DD' 형식으로 표현된다.

datatime

1000-01-01:00:00:00 ~9999-12-31:23:59:59 까지 표현이 되며 Mysql에서 기본타입으로 설정이 된다.

timestamp

1970-0101 00:00:00부터 2037년 임의의 시간까지 이미 YYYYMMDDHHMMSS, YYMMDDHHMMSS, YYYYMMDD, YYMMDD 형식으로 표현한다.

time

시간형으로 -838:59:59~838:59:59까지이며 HH:MM:SS으로 표현된다.

year

년도형이며 1901~2155까지 그리고 0000까지이다. 00~69까지는 2000~2069년 70~99까지는 1970~1999년으로 인식한다.

세 종류의 데이터 타입의 형식을 알아보았다. 각각의 타입은 필드에 적용시키는 것은 앞서서 Front에서 우리는 배웠다. 필드의 특징에 맞게 속성을 타입을 부여했다면 그에 맞게 프로그램 작업을 통해 데이터를 입력시키면 된다.

MySql 테이블 타입

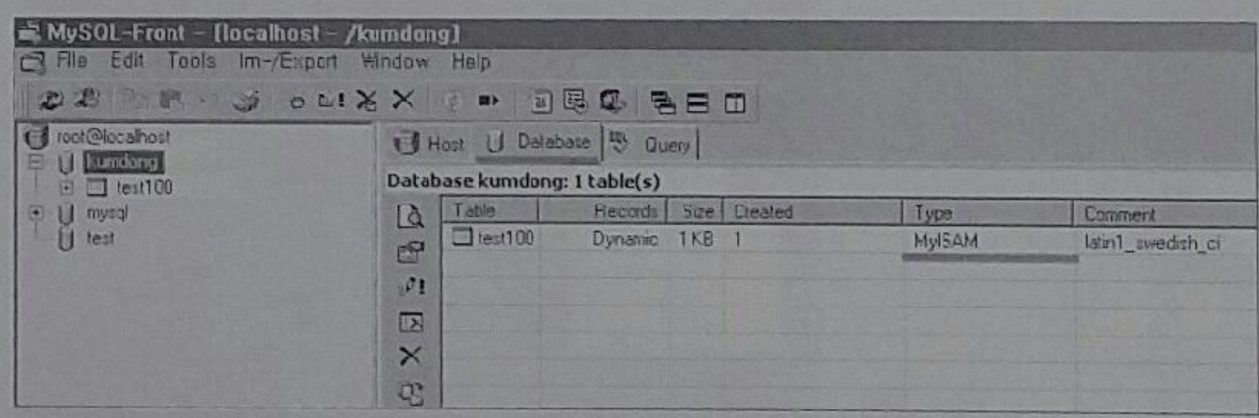
MySql에서 생성되는 테이블에는 타입을 지정할 수 있다. MySql에서 지원하는 테이블의 타입에는 6가지(MyISAM, ISAM, MERGE, HEAP, BDB, InnoDB)가 있다. 주로 사용되는 타입은 MyISAM, MERGE, InnoDB를 가장 많이 사용한다. 이 책에는 MyISAM과 InnoDB에 대해서만 설명할 것이다. 왜냐하면 이 두 가지 테이블 타입이 가장 보편화 되고 많이 사용되기 때문이다.

MyISAM

MySql에서 테이블을 만들면 기본적인 타입으로 선언된다. MyISAM 타입의 특징 중 몇 가지를 살펴보면 다음과 같다.

1. 인덱스 데이터를 위한 키 캐시가 존재한다.
2. Varchar, Char 데이터 형은 64K까지 사용할 수 있다.
3. BLOB, TEXT 데이터형도 인덱스 지정이 가능하다.
4. 트랜잭션이 지원되지 않는다.

MyISAM 타입의 테이블은 myisamchk 유틸리티를 이용하여 테이블을 체크하거나 복구할 수 있다. 또한 myisampacl 유틸리티를 통해 압축이 가능한 장점을 지니고 있다.



▶ 테이블 생성 시 기본타입으로 설정되는 MyISAM

일반적인 테이블의 생성은 보통 MyISAM 타입으로 설정하여 사용한다.

트랜잭션이란?

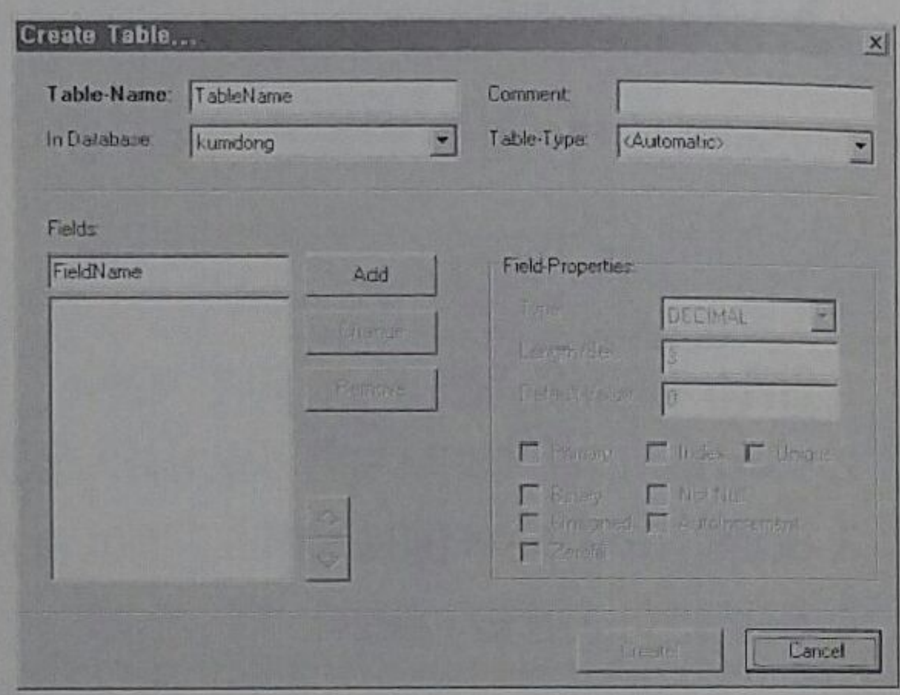
트랜잭션이란 데이터 입력 시 어떠한 오류가 발생했다면 그 데이터의 처리가 이루어지지 않고 원래의 상태로 돌아가는 것을 말한다. 즉 데이터를 처리함에 있어 올바른 처리 과정 중 어떠한 에러도 없다면 그 데이터는 정상적인 데이터임으로 우리가 만들어 놓은 테이블에 저장이 되지만 데이터의 처리도중에 프로그램상의 오류 또는 기타 전산상의 오류가 발생했을 시 데이터가 입력이 된다면 데이터의 무결성이 없어진다.
그러한 것을 방지하는 것을 트랜잭션 처리라 한다. 상용화된 데이터베이스는 가격대비 성능으로 모든 테이블에 트랜잭션의 처리기능을 가지고 있지만 MySql은 그러하지 못하고 타입을 지정하여 사용해야 한다.

InnoDB

만약 트랜잭션의 처리를 요구하는 테이블을 생성하고자 한다면 InnoDB 타입을 지정하면 된다. InnoDB는 커밋, 롤백 등의 기능도 제공한다. 트랜잭션을 사용하기 위해서는 MySql에서 기본적으로 자동 커밋이 되는 부분을 처리하고 프로그램 내에 처리하는 것이 일반적이지만 프로그램 내에서 처리하는 방법에 대해서는 좀 더 뒤에서 배우게 될 것이므로 Front에서 따라하기를 통해 배워보자.

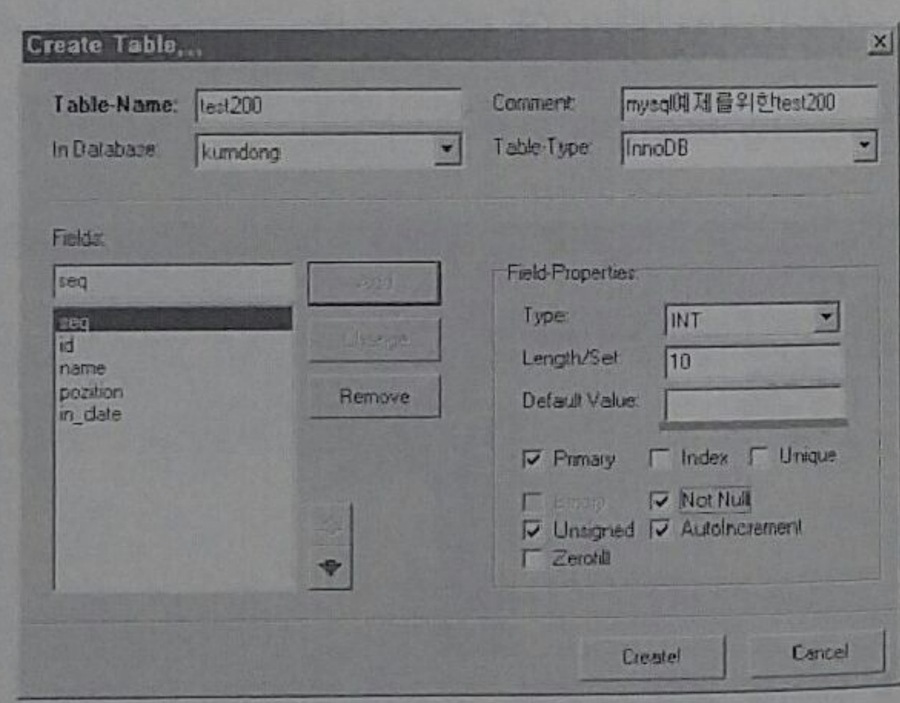
따라하기 InnoDB 생성

❶ kumdong이라는 데이터베이스 내에 Tools 메뉴를 사용하여 테이블 생성 창을 출력한다.



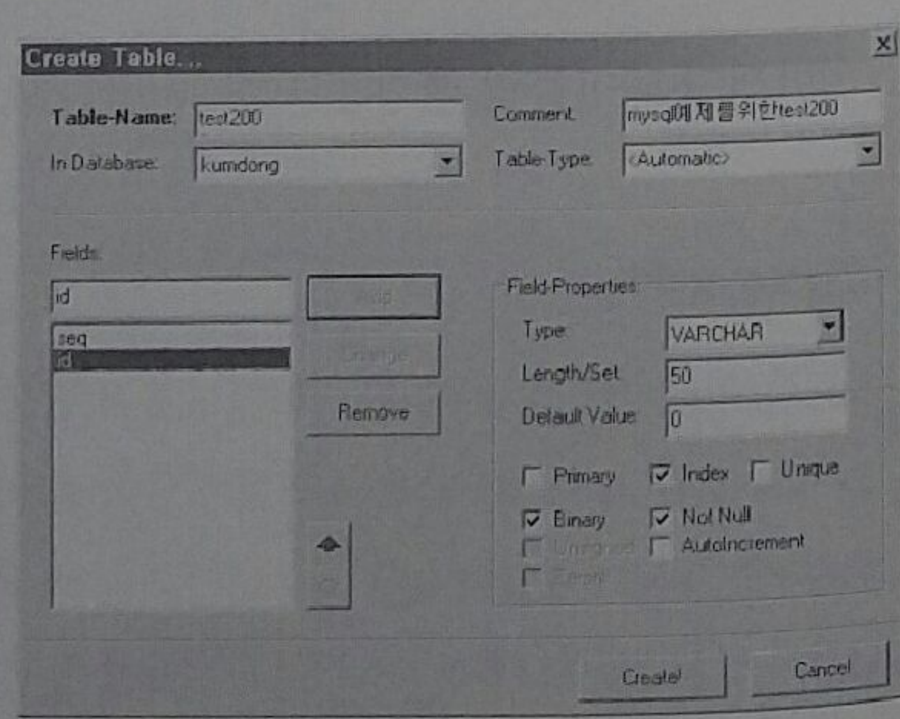
▶ 테이블 생성창

❷ seq 필드를 생성하고 필드의 타입과 속성을 부여한다. 자동 증가 값이므로 디폴트 값 0을 삭제해야 한다.



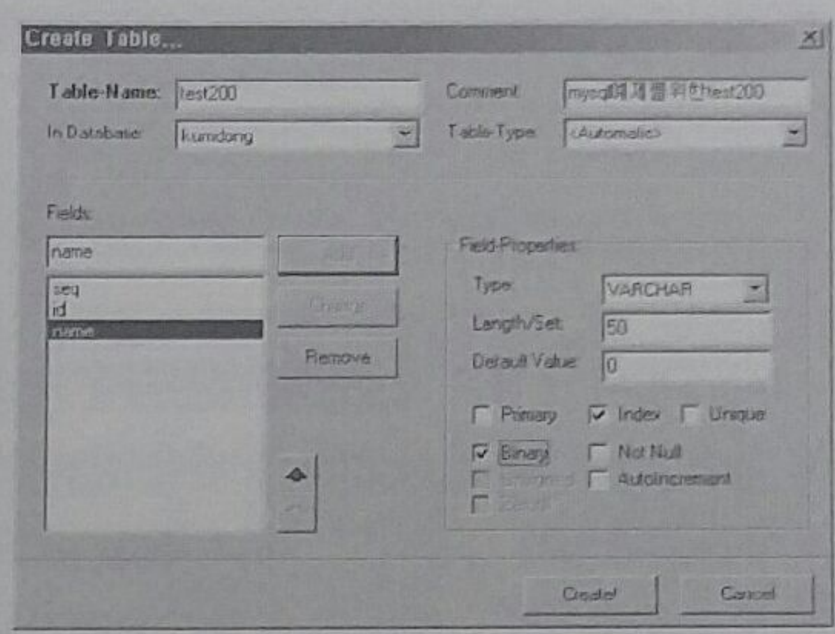
▶ seq 필드생성 및 속성부여

❸ id 필드를 생성하고 필드의 타입과 속성을 부여한다.



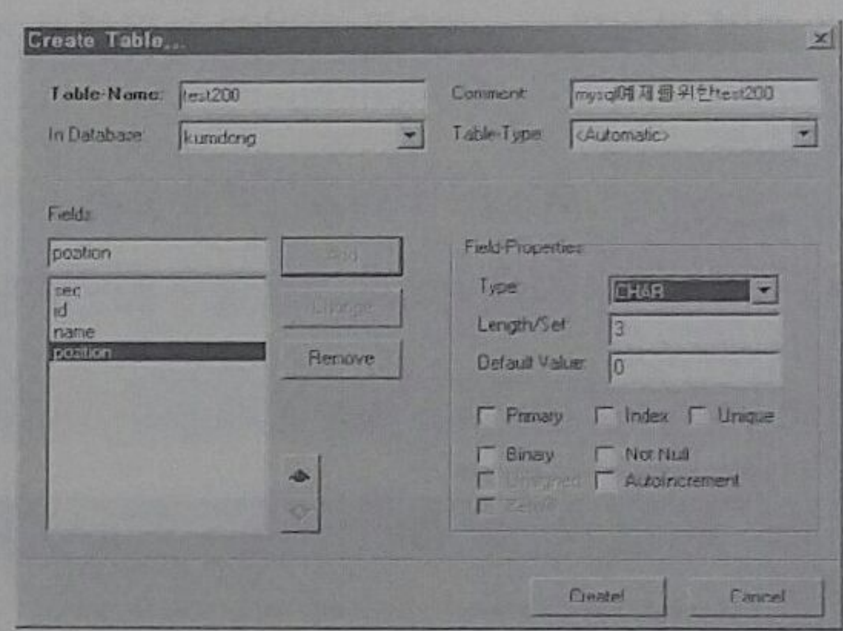
▶ id 필드생성 및 속성부여

❹ name 필드를 생성하고 필드의 타입과 속성을 부여한다.



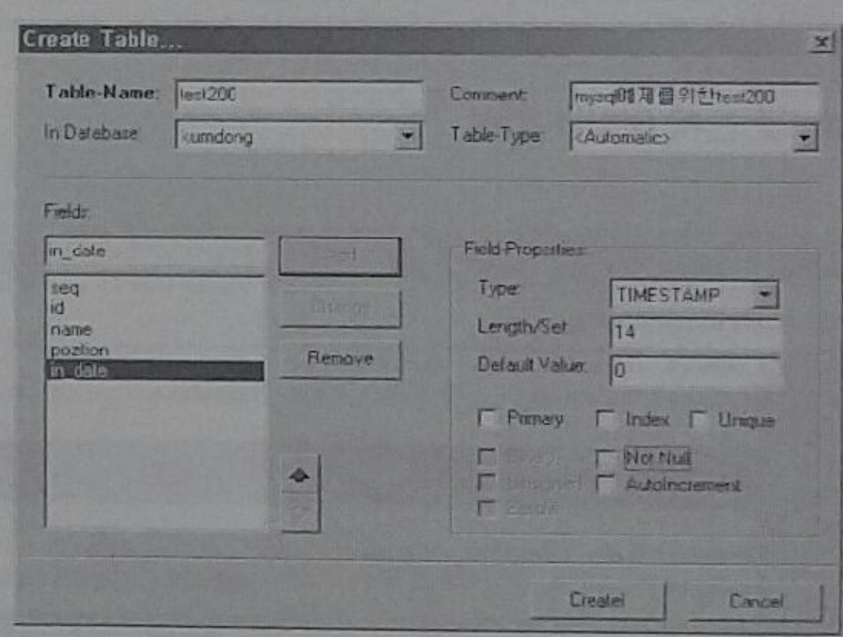
▶ name 필드생성 및 속성부여

❺ pozition 필드를 생성하고 필드의 타입과 속성을 부여한다.



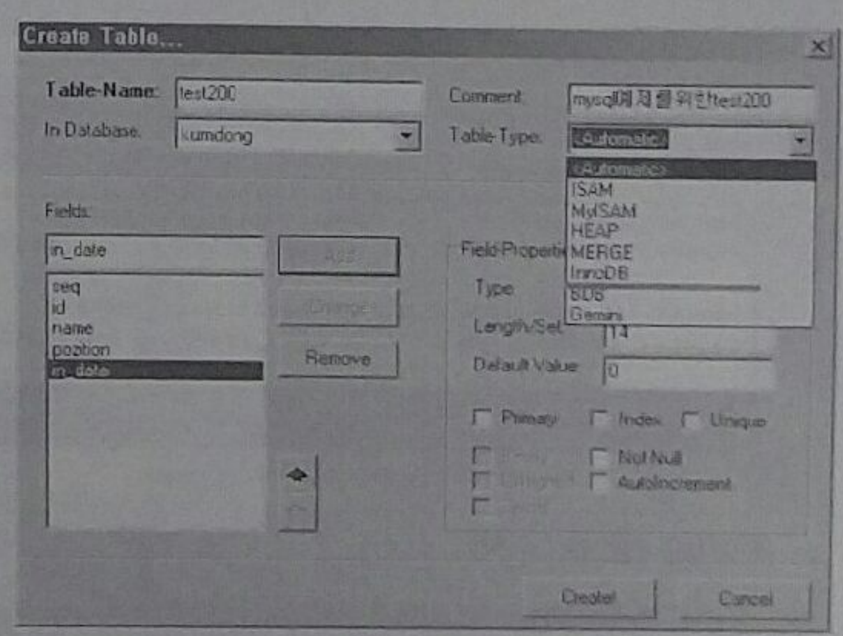
▶ pozition 필드 생성 및 속성 부여

❻ in_date 필드를 생성하고 필드의 타입과 속성을 부여한다.



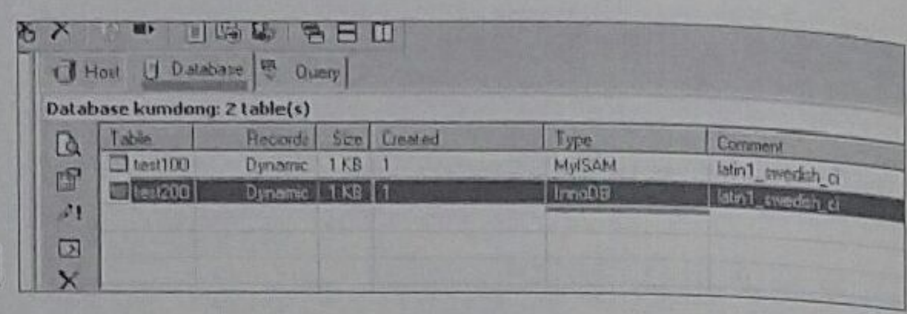
▶ in_date 필드 생성 및 속성 부여

❼ 생성되는 테이블의 타입을 InnoDB로 지정하고 Create 버튼을 클릭한다.



▶ InnoDB 지정

❽ 생성이 되었는지 확인한다.



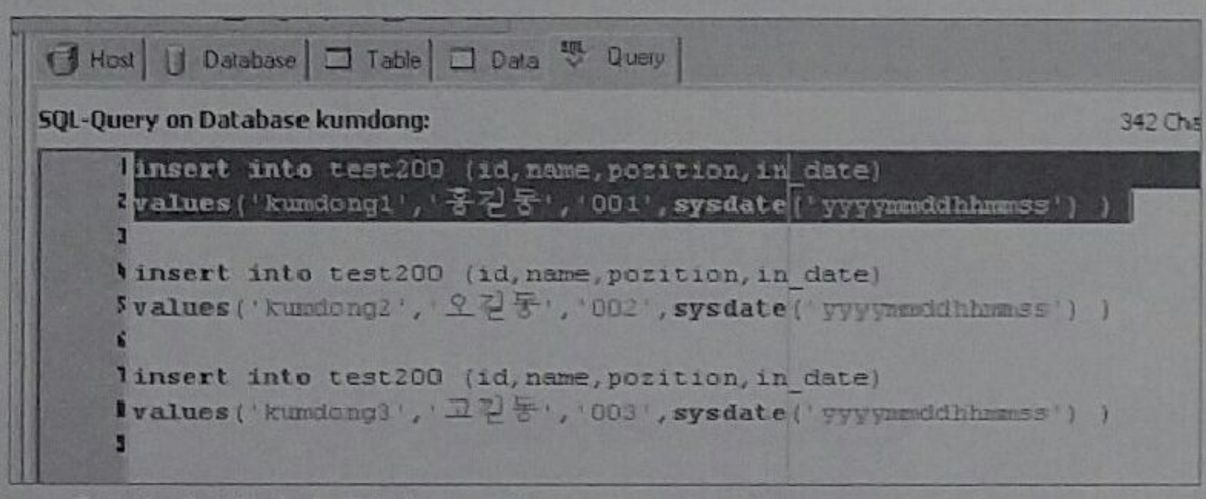
▶ InnoDB 생성확인

InnoDB의 생성은 자동으로 생성되는 MyISAM 타입의 테이블의 필드값 등등 제반 사항과 다르지 않고 다만 type 옵션에서 InnoDB만을 선택하여 생성하면 그뿐이다. 다만 Front나 다른 관리도구가 있을 시에만 간편하게 사용되고 그러지 못한 경우에는 command 창에서 생성해야 할 것이다(그러나 거의 관리 도구를 사용하지 않는 곳은 없기에 걱정할 필요가 없다).

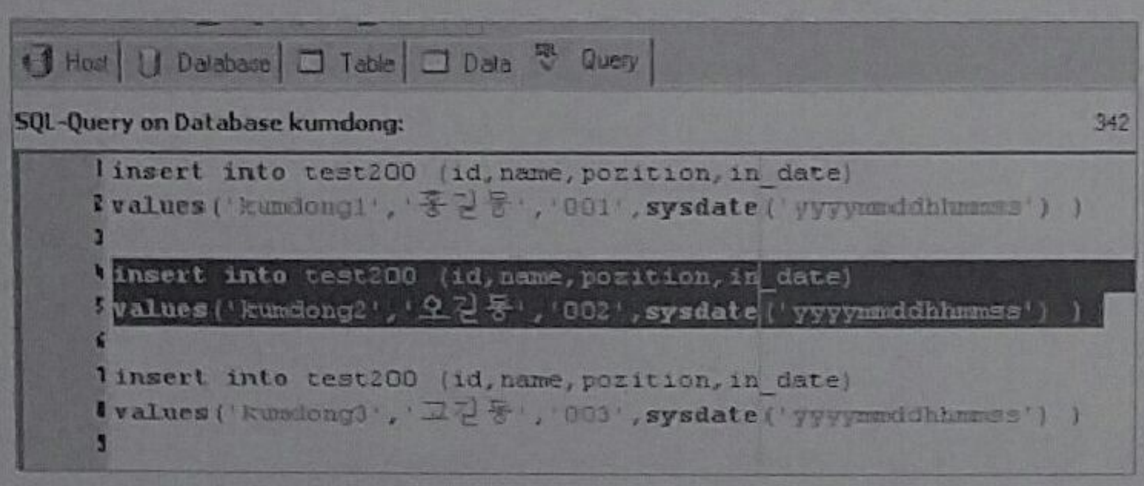
그럼 생성된 InnoDB를 가지고 트랜잭션 처리를 어떻게 하는지를 살펴보자.

InnoDB의 트랜잭션 처리

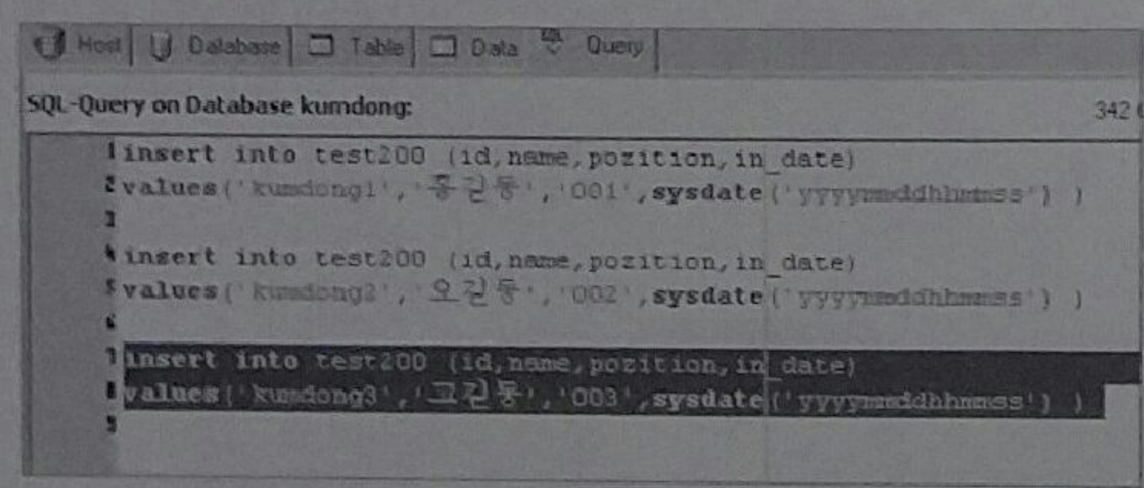
❶ 먼저 생성된 test200이라는 테이블에 INSERT문을 통해 데이터를 적어도 3개는 입력한다.



▶ 첫 번째 데이터 입력

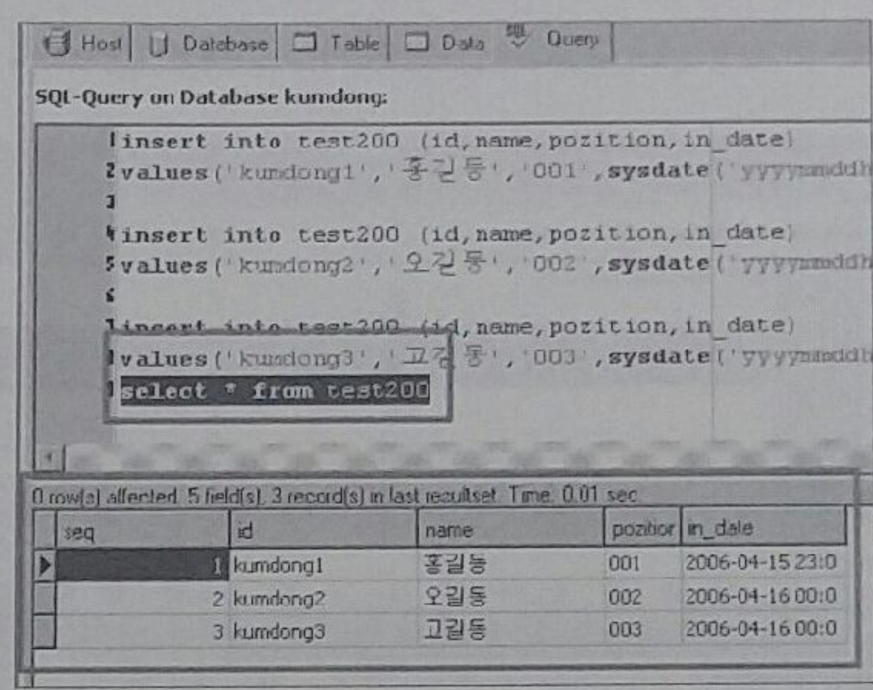


▶ 두 번째 데이터 입력



▶ 세 번째 데이터 입력

❷ 결과 값을 select 해 본다.



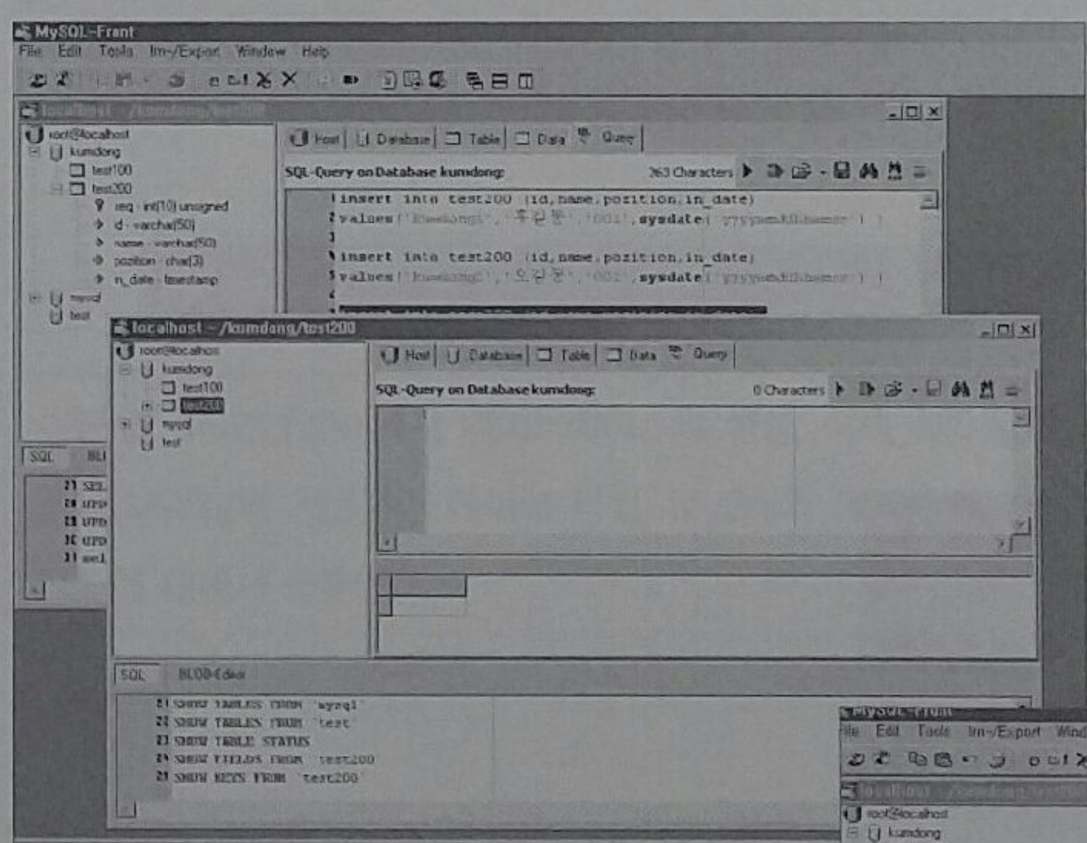
▶ INSERT 결과 확인

TIP

쿼리 실행

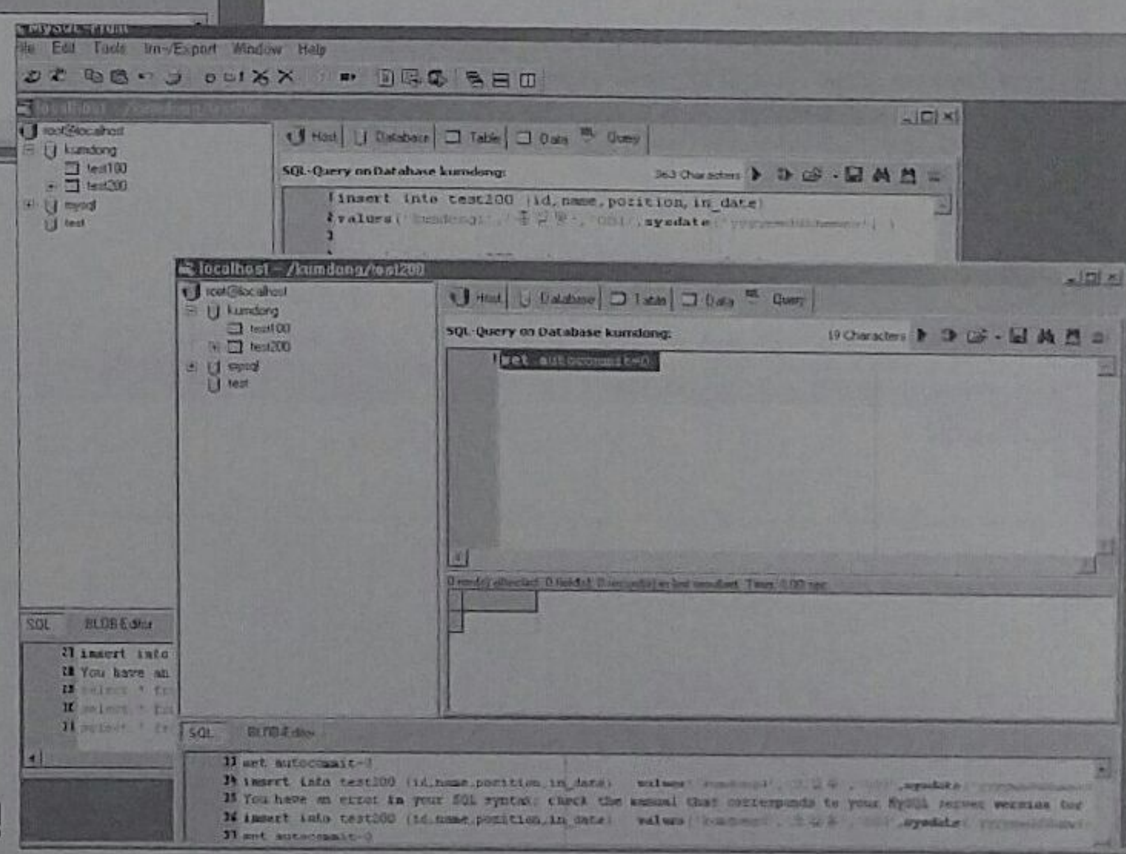
설마 잊은 건 아닌가 싶어 다시 설명을 한다. Front에서 여러 개의 쿼리가 있고 그것을 실행하려면 실행하고자 하는 쿼리를 선택(마우스 드래그 또는 키보드의 Shift+home, end, 방향키로 선택)하고 Ctrl+F9를 눌러 실행하던지 아니면 쿼리를 선택하고 execute SQL 단축키를 클릭하여 실행한다.

❸ InnoDB 작업을 확인하기 위해 또 다시 File의 Connection을 통해 접속한다.



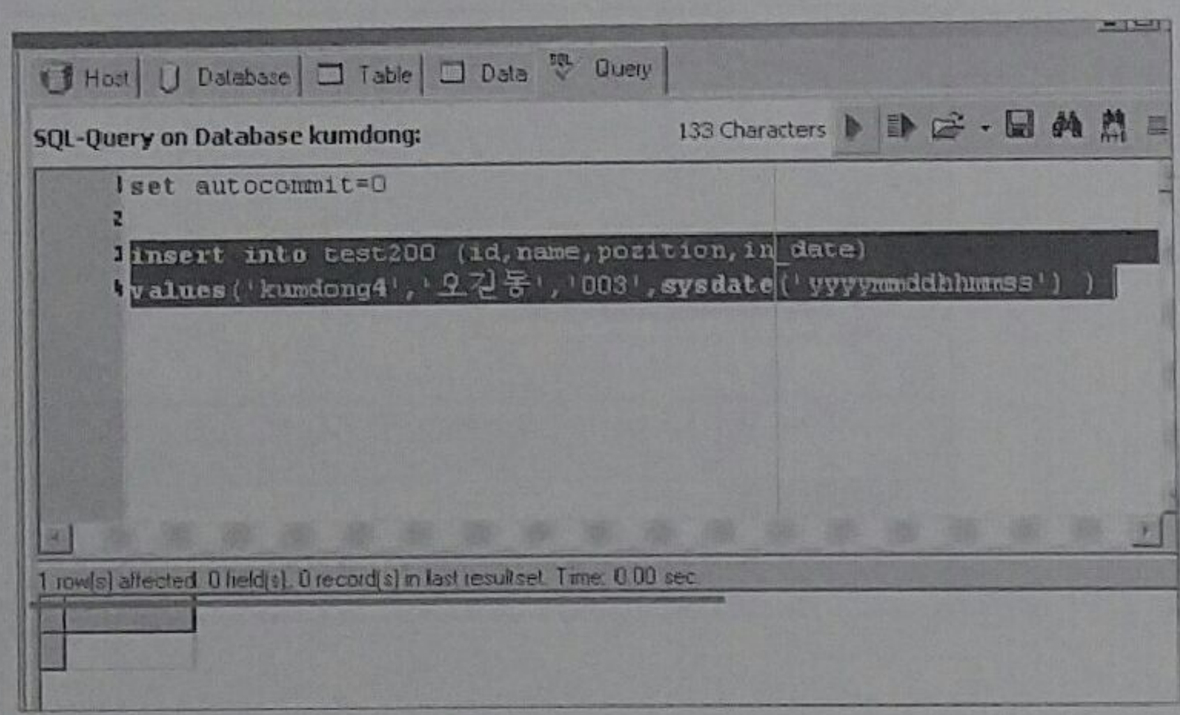
▶ 새로운 Connection을 통한 접속

❹ 트랜잭션 처리를 위해 autocommit의 값을 0으로 세팅한다.



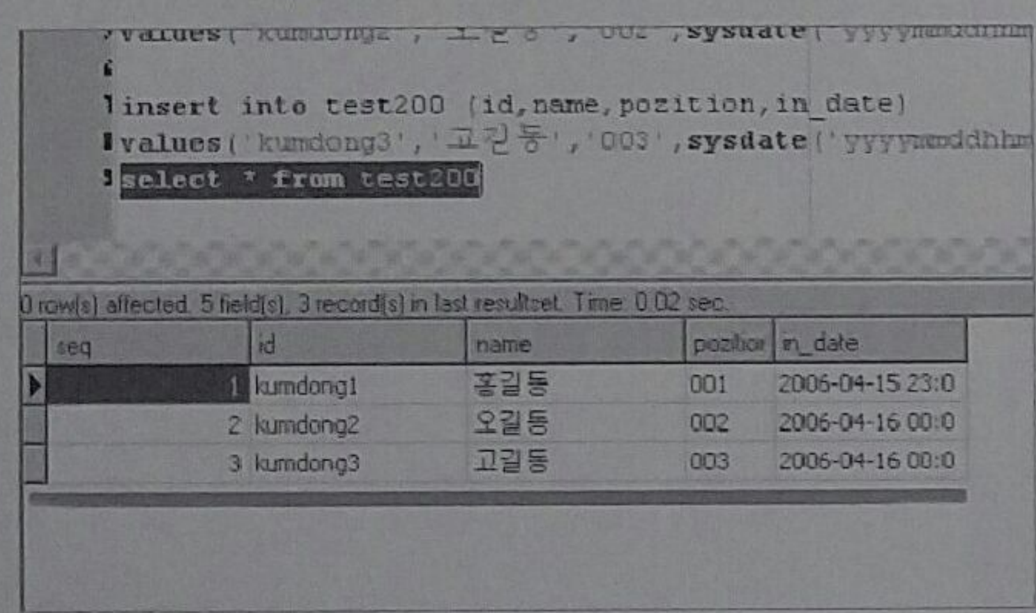
▶ autocommit의 값 세팅

❺ Insert 문을 실행한다. insert가 되었다는 메시지가 보인다.



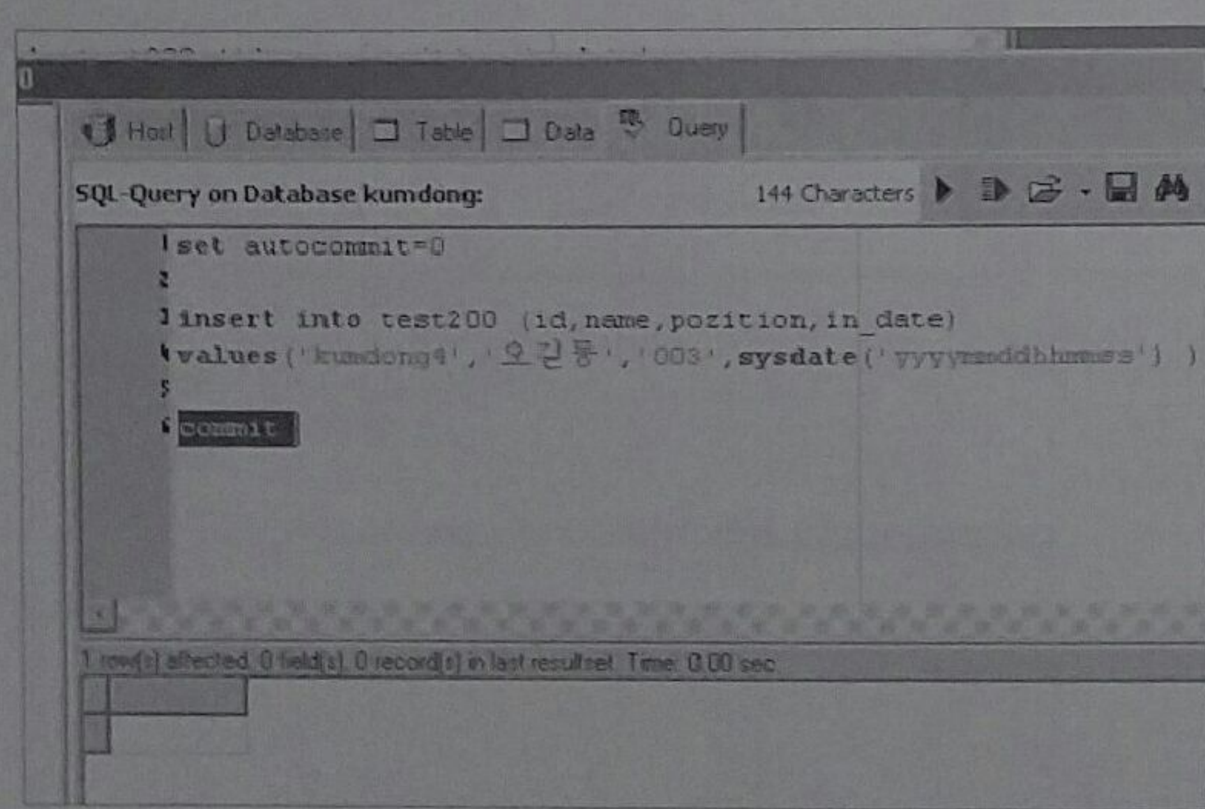
▶ INSERT문 실행

❻ 다른 창에서 select 하면 테이블에 추가된 데이터의 값은 저장되지 않음을 확인한다.



▶ Select

❼ 데이터를 추가했던 창에서 commit를 해보자.



▶ commit

❽ 다른 창에서 select를 다시 실행해 보자 그럼 kumdong4라는 아이디를 가진 데이터가 비로소 저장이 된다.

```
insert into test200 (id,name,pozition,in_date)
values('kumdong3','고길동','003',sysdate('yyyymmddhhm
select * from test200
```

0 row(s) affected. 5 field(s). 4 record(s) in last resultset. Time: 0.00 sec.

seq	id	name	pozition	in_date
1	kumdong1	홍길동	001	2006-04-15 23:0
2	kumdong2	오길동	002	2006-04-16 00:0
3	kumdong3	고길동	003	2006-04-16 00:0
5	kumdong4	오길동	004	2006-04-16 11:0

▶ 트랜잭션 처리 결과

rollback의 처리는 어떤 예외를 만났을 경우에 처리하는 방식으로 뒤에서 배우는 JSP 프로그램에서 예제를 통해 배우게 될 것이다.

TIP

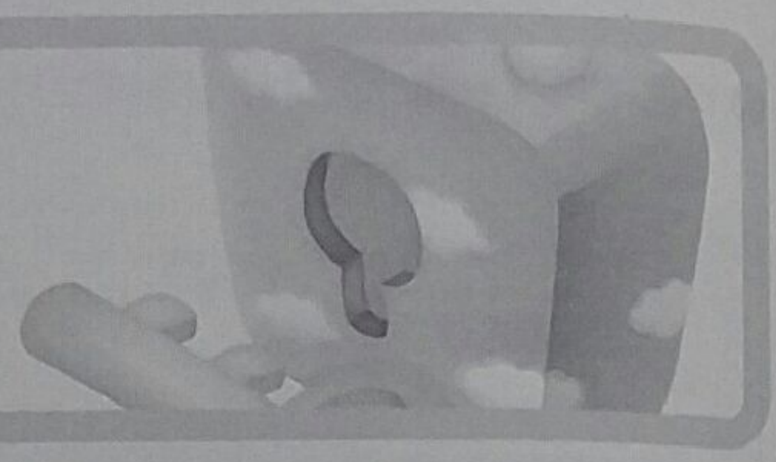
autoincrement의 초기화

InnoDB를 생성할 때 우리는 필드 seq에 autoincrement의 속성을 부여했다. autoincrement는 자동 증가값 임으로 insert문을 수행하는데 있어서 특별한 값을 넣어주지않는다. 만약 autoincrement의 값을 초기화하려면 alert table 테이블명 auto_increment=0으로 하게 되면 증가 값은 0부터 시작한다.

맺음말

이번 장에서는 Front에서 데이터베이스의 생성, 삭제, 테이블의 생성, 삭제 그리고 테이블 생성 시 필드에 대한 속성과 MySql의 기본적인 데이터 타입, MySql의 테이블 타입에 대해서 알아보았다. 다음 장에서는 MySql의 기본 SQL에 대해서 공부한다.

04 MySql의 기본 쿼리문



안녕하세요

이번 장에서는 MySql을 사용함에 있어서 질의어(SQL)에 대해서 공부한다. 앞서 배운 insert, select, update, delete도 MySql의 기본 SQL 중의 일부이다. 이 장에서는 Front를 이용하기 때문에 Create 부분에 대해서는 특별한 SQL를 사용하지 않는다. 다만 테이블을 생성하고 그 생성된 테이블 내의 데이터를 활용하는 SQL에 대해서 공부한다.

SELECT

앞서 우리는 select에 대한 부분을 배웠다. 그리고 잠시 맛을 본 where에 대한 부분도 기억하고 있을 것이다. 다시 말하면 Select는 테이블에 저장되어 있는 데이터를 사용자가 원하는 형태의 자료로 그 결과 값을 얻고자 할 때 사용되는 SQL이다. 원하는 형태의 자료를 얻기 위해 여러 가지의 조건 등이 있으며 이 장에서 그것을 배우게 된다.

별칭(Alias)

어린 시절 친구들에게 본인이 이름이 아닌 다른 별명으로 불러지는 경우가 많다. 그런 것처럼 MySql의 select에도 그런 별명이 있는데 우리는 그것을 별칭(Alias)이라고 한다.

select 시 별칭의 사용은 필드에 as를 붙여서 사용한다.

```
values('kumdong2','오길동','002',sysdate(
insert into test200 (id,name,pozition,in_c
values('kumdong3','고길동','003',sysdate(
select * from test200
select id as kumdong from test200
```

0 row(s) affected, 1 field(s), 4 record(s) in last resultset. Time: 0.00 sec.

kumdong
kumdong1
kumdong2
kumdong3
kumdong4

▶ 별칭의 사용

order by

데이터의 결과를 정렬하기 위해 사용되며 오름(asc) /내림(desc) 차순으로 정렬할 수 있다. order by의 사용 시 정렬의 기준이 되는 필드를 지정한다.

```
insert into test200 (id,name,pozition,in_date)
values('kumdong3','고길동','003',sysdate('yyyymmddhhmmss') )
select * from test200

select id as kumdong from test200

select * from test200 order by seq asc
```

0 row(s) affected. 5 field(s), 4 record(s) in last resultset. Time: 0.00 sec.

seq	id	name	pozitior	in_date
1	kumdong1	홍길동	C01	2006-04-15 23:0
2	kumdong2	오길동	C02	2006-04-16 00:0
3	kumdong3	고길동	C03	2006-04-16 00:0
4	kumdong4	오길동	C04	2006-04-16 11:0

▶ seq 기준의 order by asc

```
select id as kumdong from test200

select * from test200 order by seq asc

select * from test200 order by seq desc
```

0 row(s) affected. 5 field(s), 4 record(s) in last resultset. Time: 0.00 sec.

seq	id	name	pozitior	in_date
4	kumdong4	오길동	004	2006-04-16 11:0
3	kumdong3	고길동	003	2006-04-16 00:0
2	kumdong2	오길동	002	2006-04-16 00:0
1	kumdong1	홍길동	001	2006-04-15 23:0

▶ seq 기준의 order by desc

distinct

데이터 내에 중복된 값을 제외하고 보여주기 위해 사용된다.

```
select * from test200 order by seq desc

select distinct name from test200
```

0 row(s) affected. 1 field(s), 3 record(s) in last resultset. Time: 0.00 sec.

name
홍길동
오길동
고길동

▶ name을 기준으로 한 distinct

group by

그룹별로 검색할 때 사용하는 것으로 어떤 특정한 필드에 해당하는 값만을 그룹의 단위로 묶어 자료를 구축할 때 사용한다. group by는 그룹함수인 count, sum, max, min과 같이 사용하기도 한다. 이런 함수는 다음 장에 배우게 된다.

```
select * from test200 order by seq desc

select distinct name from test200

select count(0) from test200 group by name
```

0 row(s) affected. 1 field(s), 3 record(s) in last resultset. Time: 0.02 sec.

count(0)
1
2
1

▶ group by name

count(0) 또는 count(*)는 전체의 카운트를 구하는 것인데 group by 다음에 name이므로 name이라는 필드를 그룹으로 지정한 것이다. 따라서 이름이 같은 오길동은 2개이며 나머지는 각각 1개씩이다.

limit

쿼리의 결과 데이터에서 일정한 부분만을 가져오는 사용되며 limit 숫자1, 숫자2 형식을 가진다. 즉 limit는 범위를 나타내는 것이 아니라 쿼리한 결과의 순서 1, 2, 3 ... n까지 있다면 limit 1, 3이면 결과의 순서 중 1, 2, 3의 레코드만을 가지고 온다는 의미이다. limit는 게시판의 페이지 인덱스 처리할 때도 사용이 된다.

```
select count(0) from test200 group by name

select * from test200 limit 0,2
```

0 row(s) affected. 5 field(s), 2 record(s) in last resultset. Time: 0.00 sec.

seq	id	name	pozition	in_date
1	kumdong1	홍길동	001	2006-04-15 23:0
2	kumdong2	오길동	002	2006-04-16 00:0

▶ limit에 따른 결과

having

group by를 통해 나온 데이터에 조건을 부여하여 그 조건에 맞는 데이터를 얻는 방법이다.

```
select * from test200 limit 0,2

select count(*),id,name,pozition from test200 group by id,name having pozition > '002'
```

0 row(s) affected. 4 field(s), 2 record(s) in last resultset. Time: 0.02 sec.

count(*)	id	name	pozition
1	kumdong3	고길동	003
1	kumdong4	오길동	004

▶ having에 따른 결과

where

기본 SQL에서 가장 많이 사용되는 것으로 사용자가 원하는 데이터에 조건을 주어 결과를 얻어내는 방법이다.

```

select count(*),id,name,pozition from test200 group

select * from test200 where name='오길동'
```

0 row(s) affected. 5 field(s), 2 record(s) in last resultset. Time: 0.02 sec.

seq	id	name	pozition	in_date
2	kumdong2	오길동	002	2006-04-16 00:0
4	kumdong4	오길동	004	2006-04-16 11:0

▶ test200 테이블에서 name 필드의 값이 오길동인 데이터

```

select * from test200 where name='오길동'
select * from test200 where name='오길동' and pozition='004'
```

0 row(s) affected. 5 field(s), 1 record(s) in last resultset. Time: 0.03 sec.

seq	id	name	pozition	in_date
4	kumdong4	오길동	004	2006-04-16 11:0

▶ test200 테이블에서 name 필드의 값이 오길동 이고 포지션이 004인 데이터

between

limit가 범위와는 상관없이 결과에 대한 순서라면 between은 범위에 따른 결과 값을 가지고 오는 방법이다. 즉 필드명 between 어디서부터 and 어디까지가 되는 것이다.

```
select * from test200 where name='오길동'
select * from test200 where name='오길동' and pozition='004'
select * from test200 where pozition between 002 and 004
```

0 row(s) affected. 5 field(s), 3 record(s) in last resultset. Time: 0.00 sec.

seq	id	name	pozition	in_date
2	kumdong2	오길동	002	2006-04-16 00:0
3	kumdong3	고길동	003	2006-04-16 00:0
4	kumdong4	오길동	004	2006-04-16 11:0

▶ between의 결과

in

특정한 값을 주어 그 값에 해당하는 결과를 얻어오는 방법으로 in(값1, 값2...)으로 나타낸다.

```
select * from test200 where name='오길동'
select * from test200 where name='오길동' and pozition='004'
select * from test200 where pozition between 002 and 004
select * from test200 where pozition in(002,004)
```

0 row(s) affected. 5 field(s), 2 record(s) in last resultset. Time: 0.01 sec.

seq	id	name	pozition	in_date
2	kumdong2	오길동	002	2006-04-16 00:0
4	kumdong4	오길동	004	2006-04-16 11:0

▶ in의 결과

like

특정문자열을 주어 그 문자열과 같거나 그 문자열이 들어가 있는 데이터의 결과를 얻어올 때 사용된다. 옵션으로 %를 사용하는데 % 의미는 모든 것을 의미한다.

```
select * from test200 where name='오길동'
select * from test200 where name='오길동' and pozition='004'
select * from test200 where pozition between 002 and 004
select * from test200 where pozition in(002,004)
select * from test200 where name like '오길%'
```

0 row(s) affected. 5 field(s), 2 record(s) in last resultset. Time: 0.00 sec.

seq	id	name	pozition	in_date
2	kumdong2	오길동	002	2006-04-16 00:0
4	kumdong4	오길동	004	2006-04-16 11:0

▶ like의 결과

like에서 %을 앞, 뒤 또는 앞뒤로 사용함에 있어 약간의 차이가 있다. 만약 %을 앞에 사용하면 다음과 같은 결과가 나온다.

```
select * from test200 where pozition between 002 and
select * from test200 where pozition in(002,004)
select * from test200 where name like '%오길'
```

0 row(s) affected. 5 field(s). 0 record(s) in last resultset. Time: 0.00 sec.

seq	id	name	pozition	in_date

▶ %을 앞에 사용한 결과

%은 모든 것을 의미한다고 했다. 즉 %오길 이라면 오길이라는 데이터를 찾을 것이다. 따라서 그런 데이터가 없다고 나오게 된다. 반면에 앞뒤로 %오길%이라고 한다면 오길%과 같다. 따라서 like을 사용할 경우에는 뒤에 %을 주는 것이 바람직하다.

```
select * from test200 where pozition in(002,004)
select * from test200 where name like '오길%'
select * from test200 where name like '%오길'
select * from test200 where name like '%오길%'
```

0 row(s) affected. 5 field(s). 2 record(s) in last resultset. Time: 0.00 sec.

seq	id	name	pozition	in_date
2	kumdong2	오길동	002	2006-04-16 00:0
4	kumdong4	오길동	004	2006-04-16 11:0

▶ %을 앞뒤로 사용한 결과

INSERT

생성된 테이블에 데이터를 저장하는 SQL이다. 앞서 배운바 있지만 insert의 방법에는 몇가지가 더 있다 이 장에서는 그러한 방법을 설명하겠다. 기본적으로 insert의 형식은 다음과 같다.

```
insert into 테이블명 (필드1,필드2....n) values (값1,값2....n)
```

하지만 굳이 이러한 방식이 아닌 다른 방식도 있다.

필드를 사용하지 않는 insert 방법

테이블에 생성된 필드의 개수만큼 테이블에 저장할 데이터의 값의 개수를 맞춰서 입력하면 된다.

```
select * from test200 where name like '오길%'
select * from test200 where name like '%오길'
select * from test200 where name like '%오길%'
insert into test200 values(seq,'kumdong5','이승엽','005',sysdate('yyyymmddhhmmss') )
```

▶ values 값만으로 insert

```
select * from test200 where name like '%오길'
select * from test200 where name like '%오길%'
insert into test200 values(seq,'kumdong5','이승엽','0
select * from test200
```

0 row(s) affected. 5 field(s), 5 record(s) in last resultset. Time: 0.02 sec.

seq	id	name	position	in_date
1	kumdong1	홍길동	001	2006-04-15 23:0
2	kumdong2	오길동	002	2006-04-16 00:0
3	kumdong3	고길동	003	2006-04-16 00:0
4	kumdong4	오길동	004	2006-04-16 11:0
7	kumdong5	이승엽	005	2006-04-16 21:3

▶ insert 결과

한 가지 주목해야 할 부분은 seq가 자동증가 값이기 때문에 어떠한 값을 입력하는 것이 아니라 필드의 이름을 주었다는 것이다. 만약 임의의 값을 주게 되면 저장은 되지만 나중에 자동증가 값에 대한 오류를 발생시키게 되기 때문이다.

Front에서의 직접 입력

프로그램에 처리 방법에 의해 데이터를 입력하는 것이 아닐 때 사용한다. 선택한 테이블의 탭 메뉴 중에서 data를 선택하고 마치 엑셀 작업하듯이 값을 입력하면 된다. 입력 시에 자동 증가값 등 not null 값에 주의해서 입력해야 한다.

Host | Database | Table | Data | Query

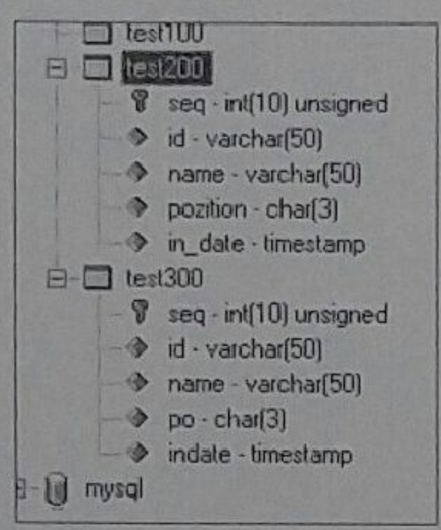
kumdong / test200: 6 Records (6 retrieved)

seq	id	name	position	in_date
1	kumdong1	홍길동	001	2006-04-15 23:0
2	kumdong2	오길동	002	2006-04-16 00:0
3	kumdong3	고길동	003	2006-04-16 00:0
4	kumdong4	오길동	004	2006-04-16 11:0
7	kumdong5	이승엽	005	2006-04-16 21:3
8	kumdong6	김영호	006	2006-04-16 21:

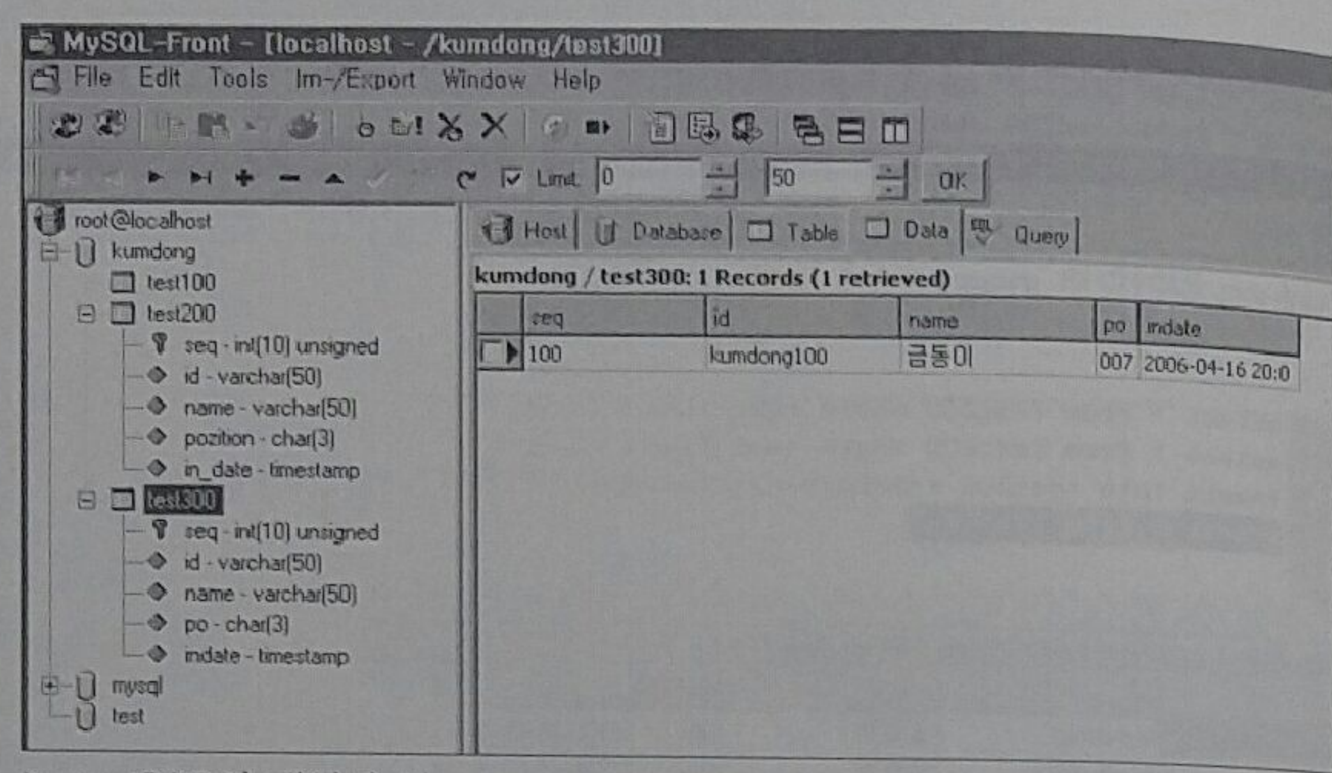
▶ Front에서 데이터 insert

select를 통한 insert

insert시 다른 테이블에 있는 데이터를 가지고와서 insert하는 방법이다. 즉 현재 test200의 테이블에 id, name, position의 필드에 다른 테이블의 데이터를 select하여 저장하는 방법이다. 단 조건은 같은 형태의 구조이어야 하며 또한 자동증가 값 또는 키 값에 대한 중복성이 없어야 한다.

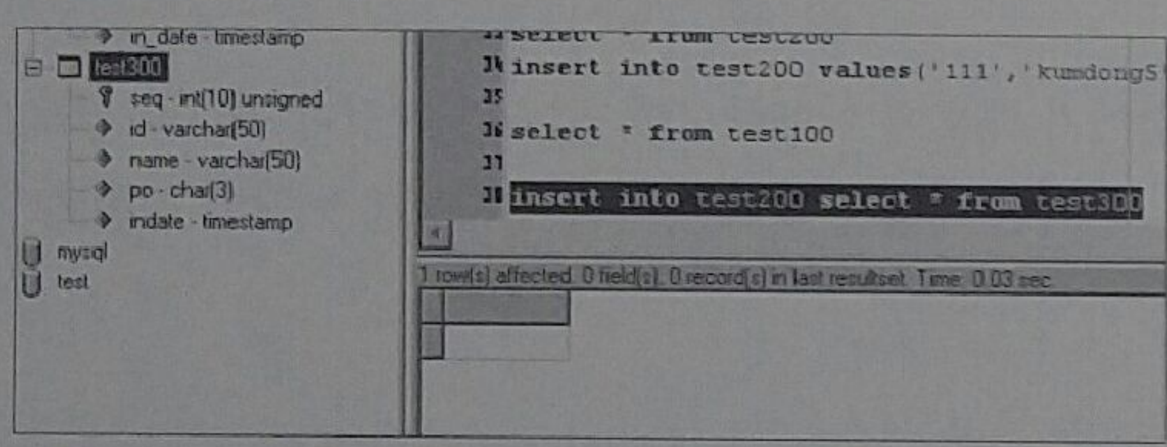


▶ test200과 같은 구조를 가진 test300

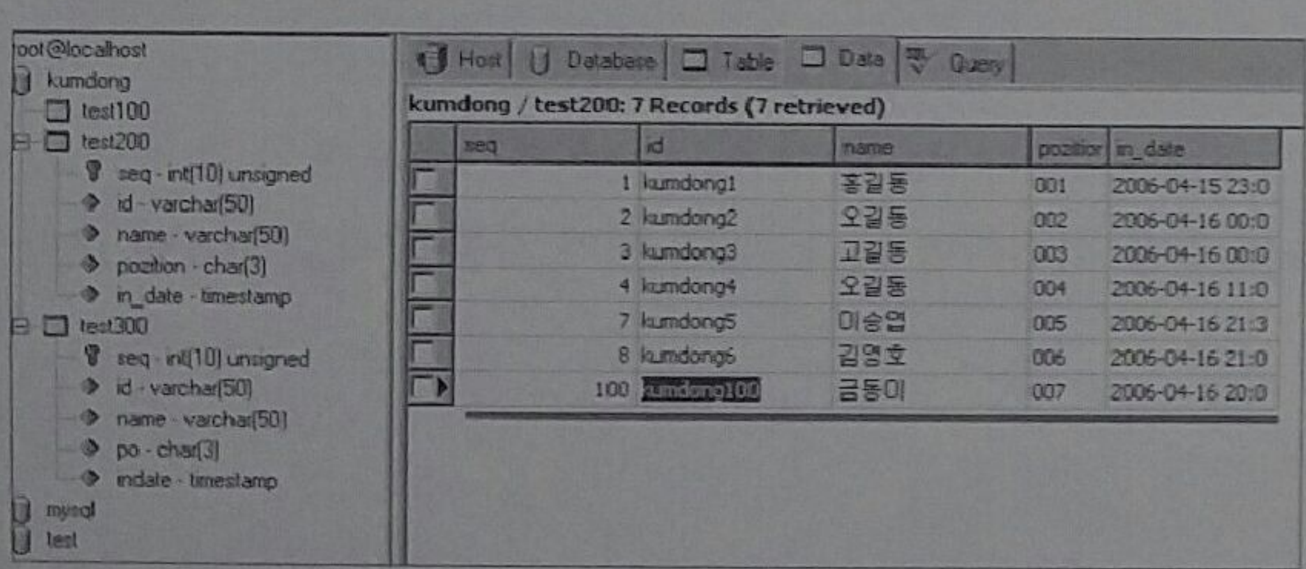


▶ test300의 데이터 값

조건이 맞는다면 select문을 통해 insert를 할 수 있다.



▶ select를 통한 insert 쿼리



▶ select를 통한 insert 쿼리 결과

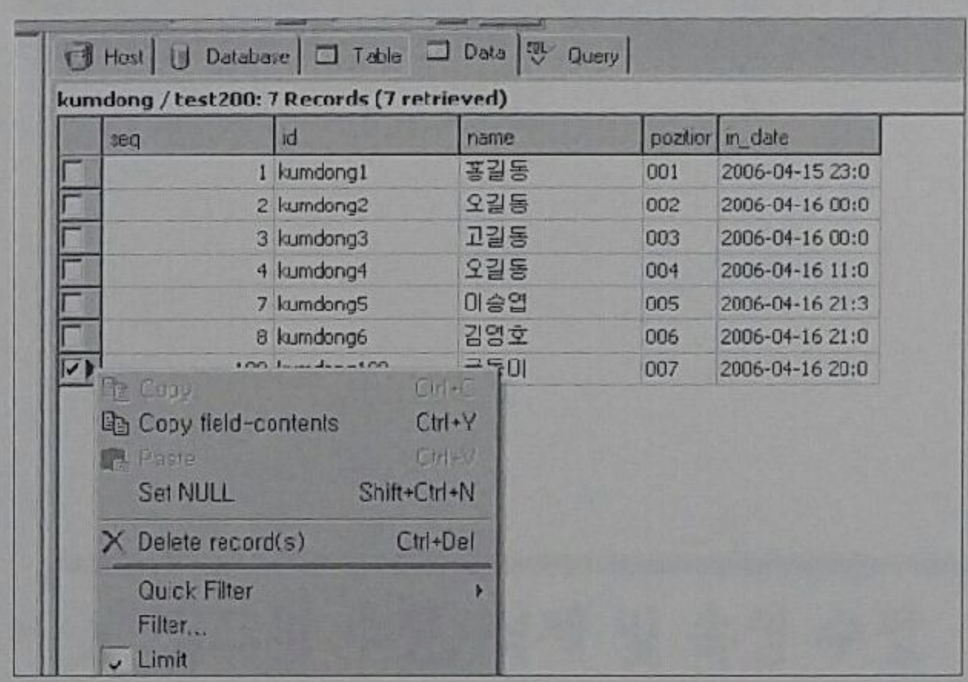
이런 select를 통한 insert의 작업은 히스토리를 남기기 위한 작업이나 데이터의 복사에 사용되기도 한다.

→ DELETE

데이터의 양이 작은 경우에는 Front에서 제공하는 메뉴를 사용하면 되지만 많은 데이터가 존재할 때는 삭제하려는 데이터를 찾는 것이 피곤한 일이므로 쿼리문을 통해 삭제한다.

Front에서의 삭제

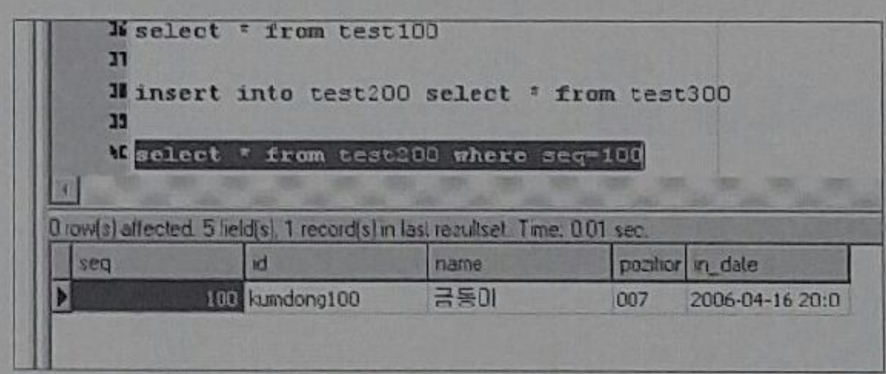
탭 메뉴의 data를 선택하면 저장된 데이터들이 나열이 된다. 나열된 데이터 중에서 삭제하고자 하는 레코드의 앞에 있는 체크박스를 클릭하고 마우스의 오른쪽 버튼을 클릭하면 팝업메뉴가 나온다. 메뉴 중에 Delete Record를 선택하면 삭제가 된다. 참고로 여러개의 데이터를 삭제하려면 체크박스에 체크한 후 Delete Record를 선택하면 체크된 데이터는 모두 삭제가 된다.



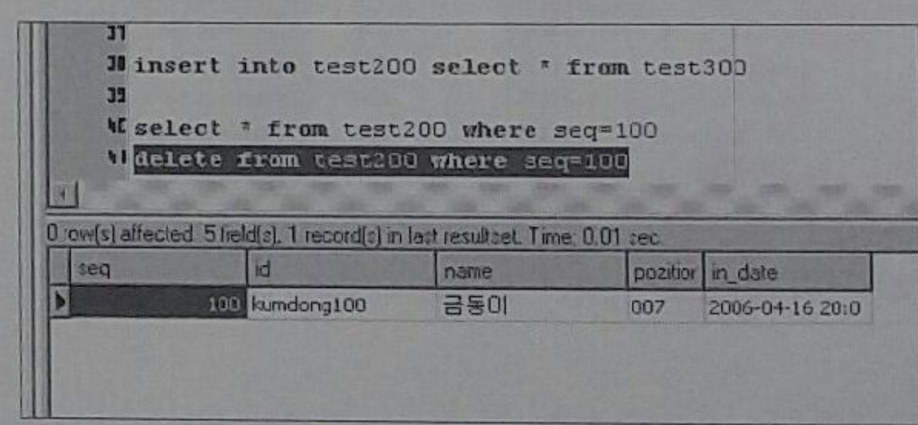
▶ Front에서의 데이터 삭제

쿼리를 통한 삭제

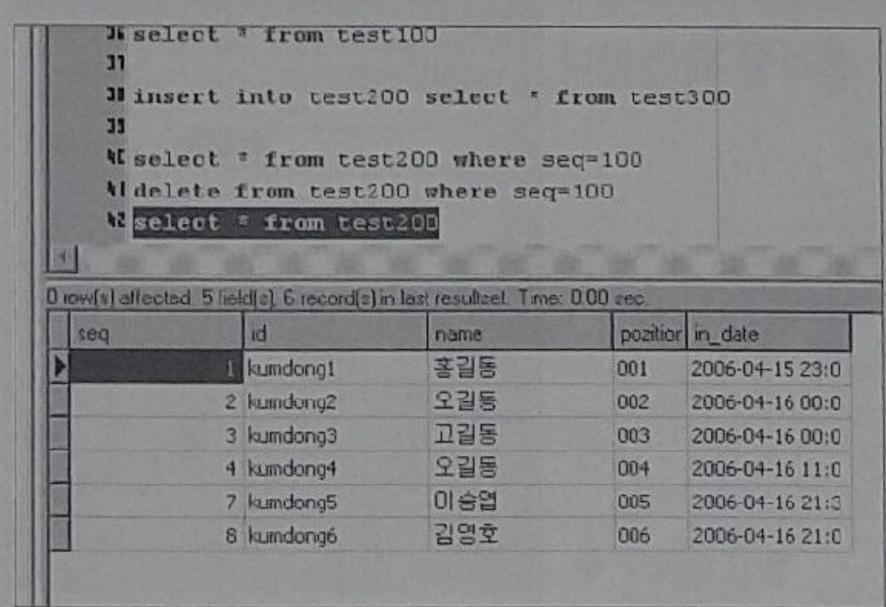
앞 장에서도 이야기했지만 데이터의 삭제 시에는 조심해야 한다. 특히 *을 사용할 때는 더욱더 조심을 한다. 삭제 시에는 먼저 select를 통해 삭제할 데이터를 확인하고 삭제해야한다.



▶ select를 통한 삭제 데이터의 확인



▶ delete의 실행



▶ delete 결과 확인

UPDATE

입력된 데이터의 갱신으로 Front에서는 insert에서와 마찬가지 메뉴에서 선택한 필드의 값을 수정만 해 주면 된다.

Front에서의 Update

테이블의 수정할 필드값을 선택한다.

Host | Database | Table | Data | Query

kumdong / test200: 7 Records (6 retrieved)

seq	id	name	pozition	in_date
1	kumdong1	홍길동	001	2006-04-15 23:0
2	kumdong2	오길동	002	2006-04-16 00:0
3	kumdong3	고길동	003	2006-04-16 00:0
4	kumdong4	오길동	004	2006-04-16 11:0
7	kumdong5	이승엽	005	2006-04-16 21:3
8	kumdong6	김영호	006	2006-04-16 21:0

▶ Front에서 필드값의 선택

필드값을 수정하고 키보드의 방향기 ↓로 레코드를 이동한다. ↓은 commit과 같은 역할을 한다.

Host | Database | Table | Data | Query

kumdong / test200: 7 Records (6 retrieved)

seq	id	name	pozition	in_date
1	kumdong1	홍길동	001	2006-04-15 23:0
2	kumdong2	오길동	002	2006-04-16 00:0
3	kumdong3	고길동	003	2006-04-16 00:0
4	kumdong4	오길동	004	2006-04-16 11:0
7	kumdong5	이승엽	005	2006-04-16 21:3
8	kumdong7	김영호	006	2006-04-16 21:0

▶ Front에서의 update

쿼리를 통한 update

삭제와 마찬가지로 수정할 데이터를 먼저 select를 통해 결과를 확인한다.

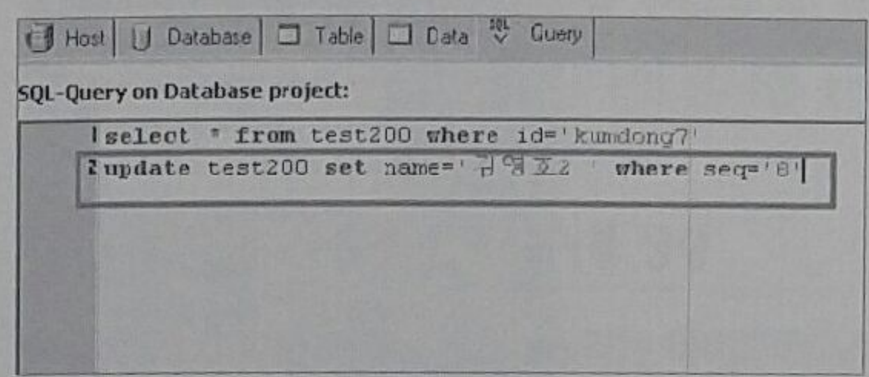
```
insert into test200 select * from test300

select * from test200 where seq=100
delete from test200 where seq=100
select * from test200
select * from test200 where id='kumdong7'
```

0 row(s) affected, 5 link(s), 1 record(s) in last resultset. Time: 0.00 sec.

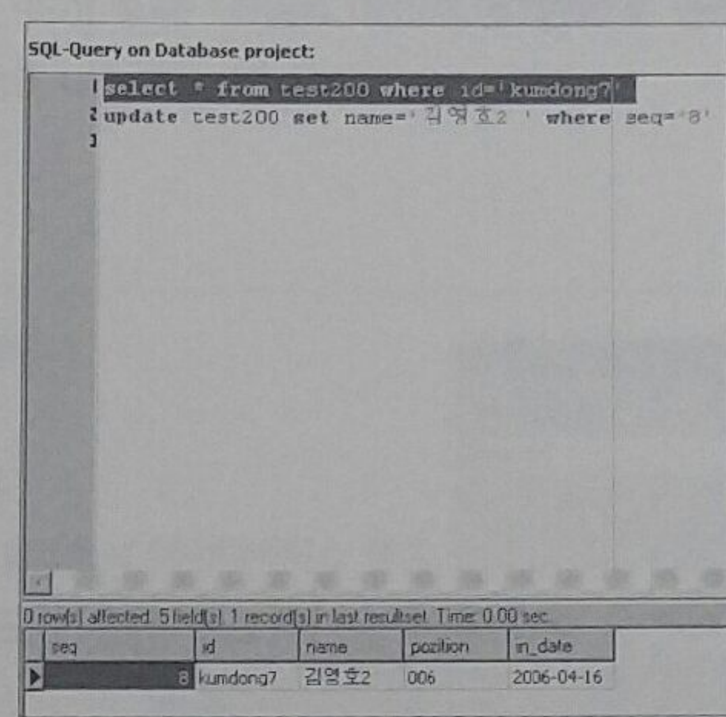
seq	id	name	pozition	in_date
8	kumdong7	김영호	006	2006-04-16 21:0

▶ update할 데이터의 select

확인이 되었다면 update 쿼리를 통해 데이터를 갱신한다.



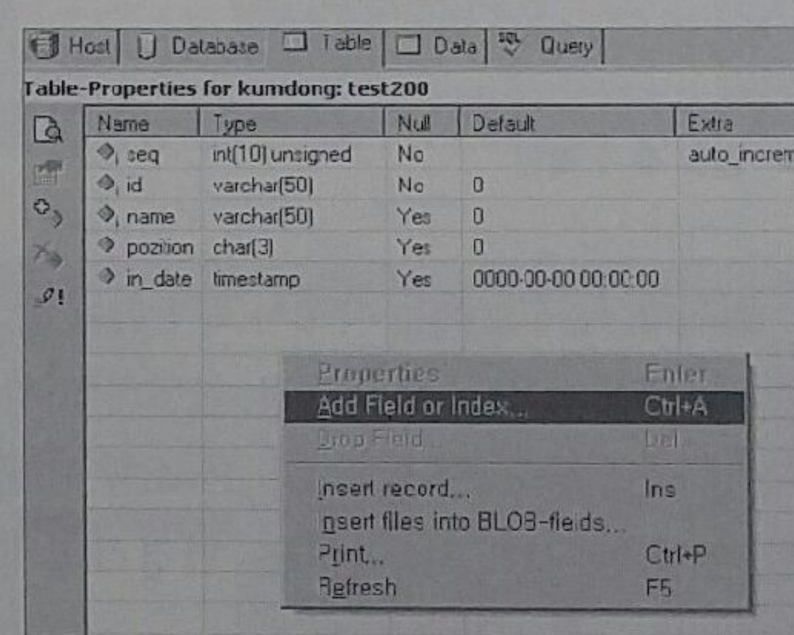
▶ update 쿼리



▶ update 결과 확인

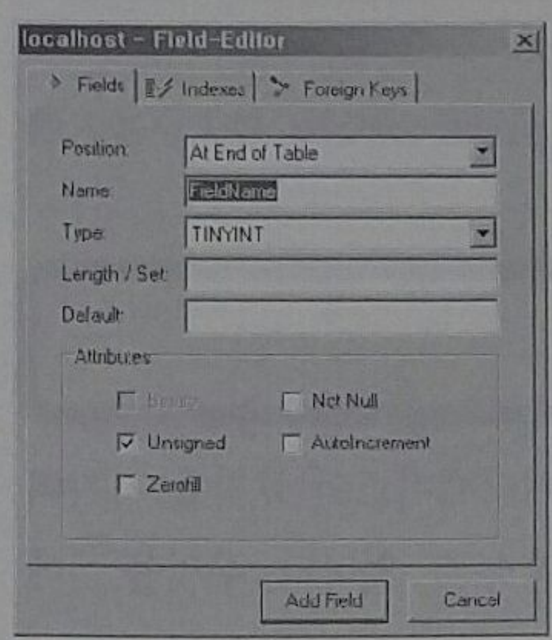
필드의 수정, 삭제 및 속성 수정

테이블을 생성하고 필드를 생성한 후에 필드의 추가가 필요한 경우에 사용된다. 탭 메뉴의 table을 선택하고 마우스 오른쪽 버튼을 클릭하면 팝업메뉴가 나온다. 팝업메뉴에서 Add Field or index를 선택한다.



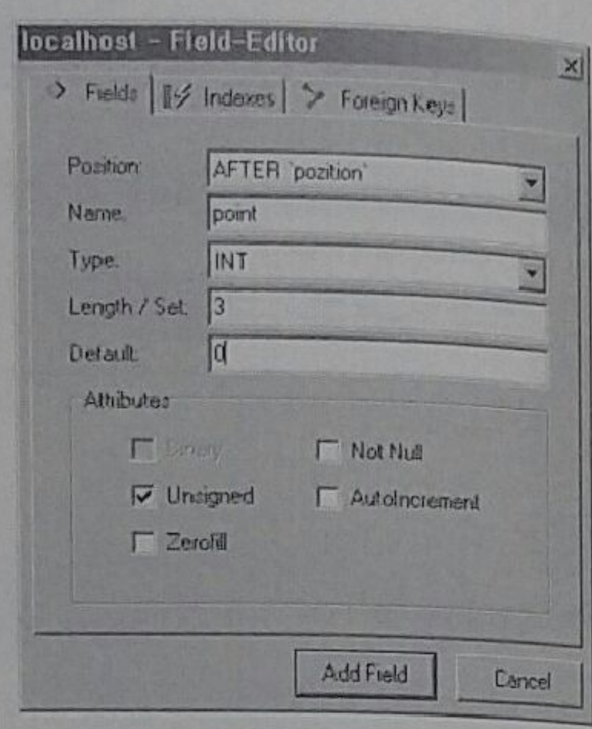
▶ Add Field or index선택

Field Edit창이 출력된다.



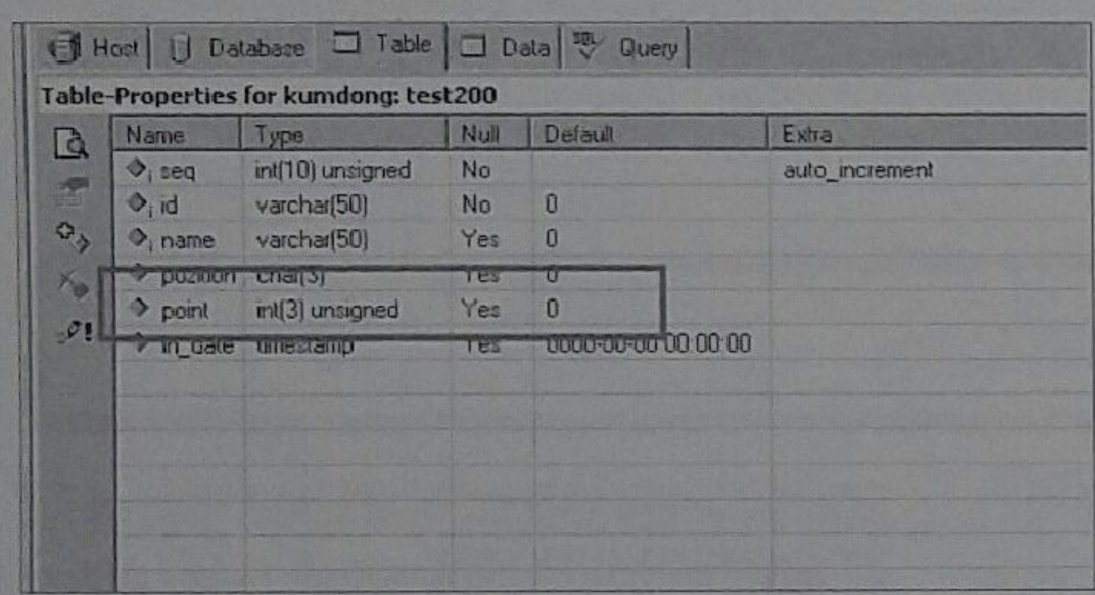
▶ Field Edit

추가할 필드가 어느 뒤에 위치할 것인지 선택하고 추가할 필드의 이름과 속성 등을 부여한다.



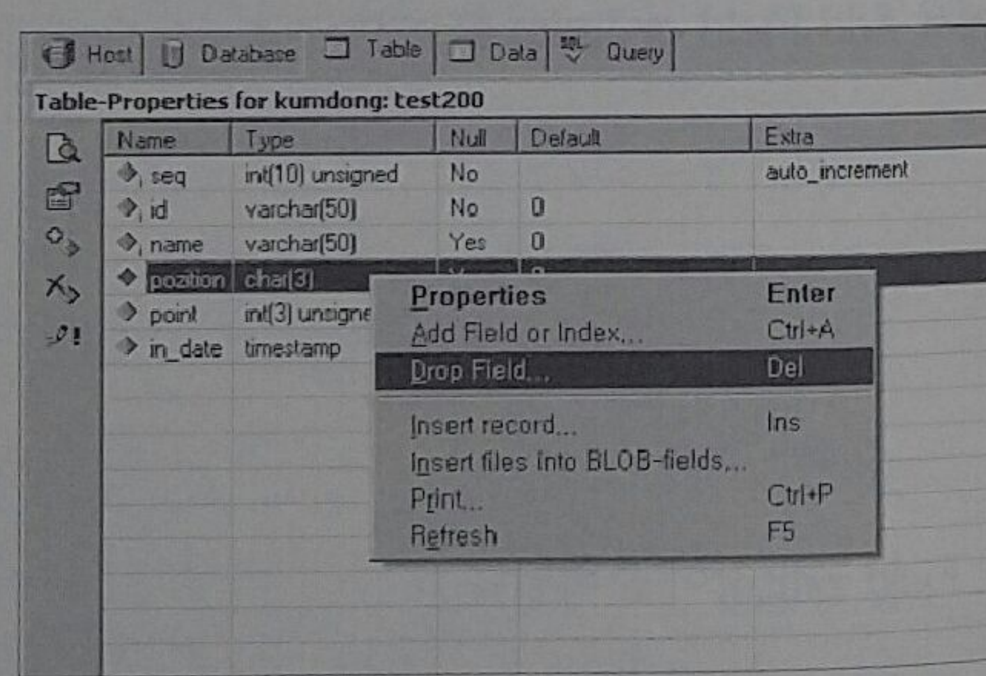
▶ 필드 속성 및 타입 지정

add field 버튼을 클릭 후에 추가하고자 하는 필드가 추가되었는지를 확인한다.



▶ 필드 추가 확인

필드를 삭제할 때는 삭제하고자 하는 필드를 선택한 후 마우스 오른쪽 버튼을 클릭하면 팝업메뉴가 나온다. 팝업메뉴에서 Drop Field를 선택한다. 또한 선택한 필드의 속성을 변경하고자 한다면 Properties를 선택하면 된다.



▶ 필드 삭제와 속성 수정

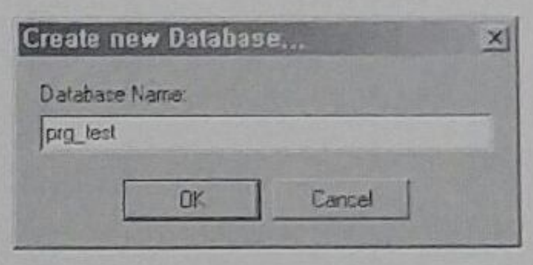
테이블과 테이블과의 JOIN과 UNION

테이블의 효율적인 관리 및 운영을 위해 테이블을 설계하는 DAB는 데이터베이스의 규칙을 지키며 설계 작업을 한다. 만약 물품구매를 관리하는 프로그램을 개발한다고 한다면 누가, 얼마나, 무엇을....에 대해서 데이터를 담고 있는 테이블을 설계할 것이다.

이렇게 각각의 정보를 담고 있는 테이블을 하나의 결과로 보이기 위해서는 각 테이블과의 결합 또는 합치기를 통해 데이터의 결과를 산출할 수 있다. 즉 두개 또는 다수의 테이블이 있다고 한다면 각 테이블에 대한 정보를 가공하여 하나의 결과를 산출하는데 사용하는 것이다.

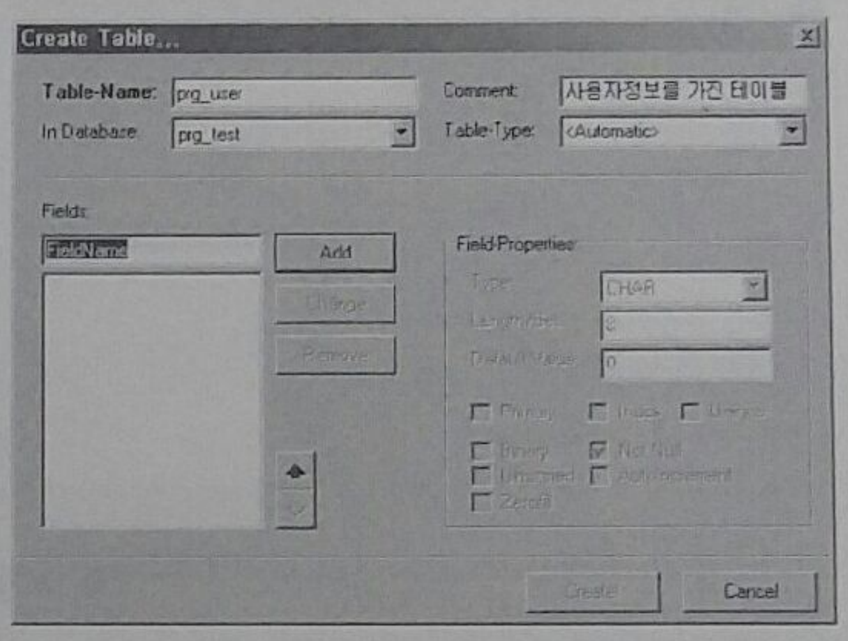
따라하기 사용자 정보 테이블 생성

❶ 사용자 정보를 가지는 prg_test라는 데이터베이스를 생성한다.



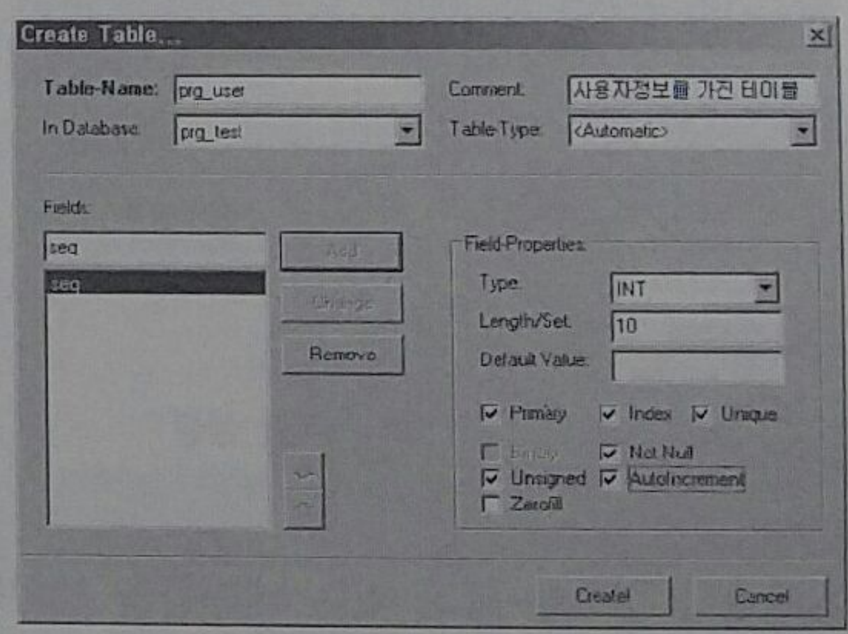
▶ prg_test 데이터베이스 생성

❷ 사용자 정보를 저장하는 테이블을 생성한다.



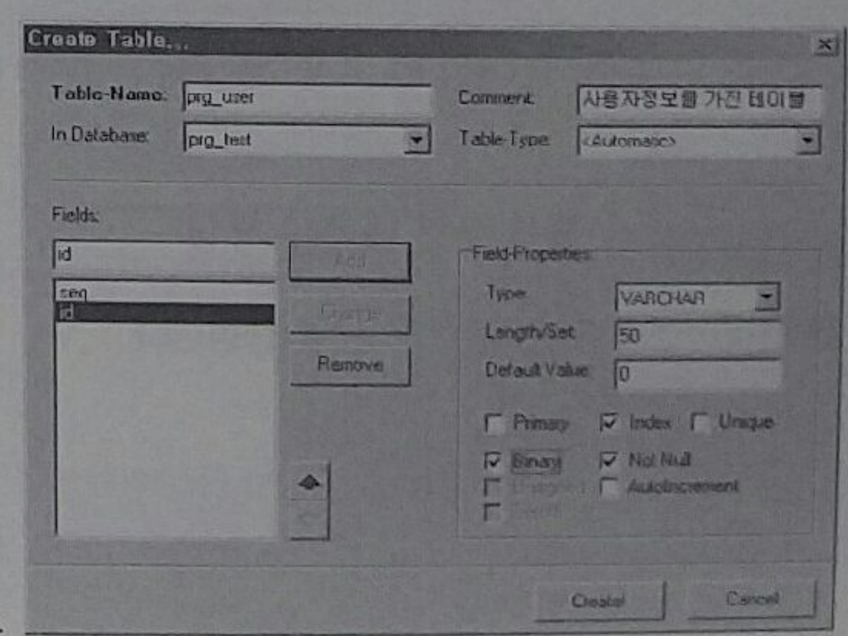
▶ 테이블 생성 창

❸ 일련번호를 저장하는 필드를 생성한다.



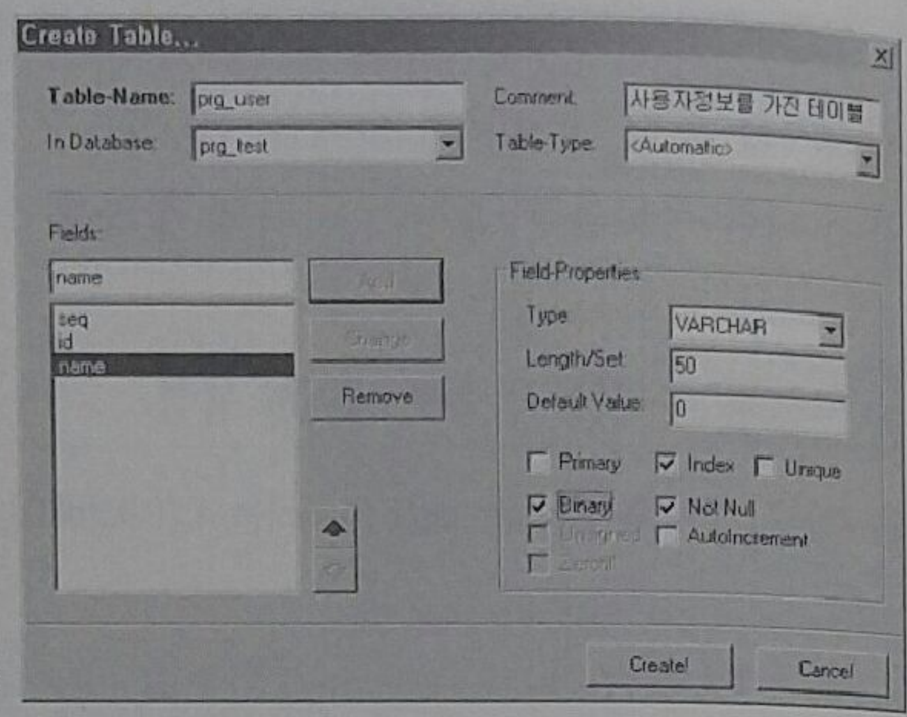
▶ 일련번호 seq 필드

❹ 사용자 ID를 저장하는 필드를 생성한다.



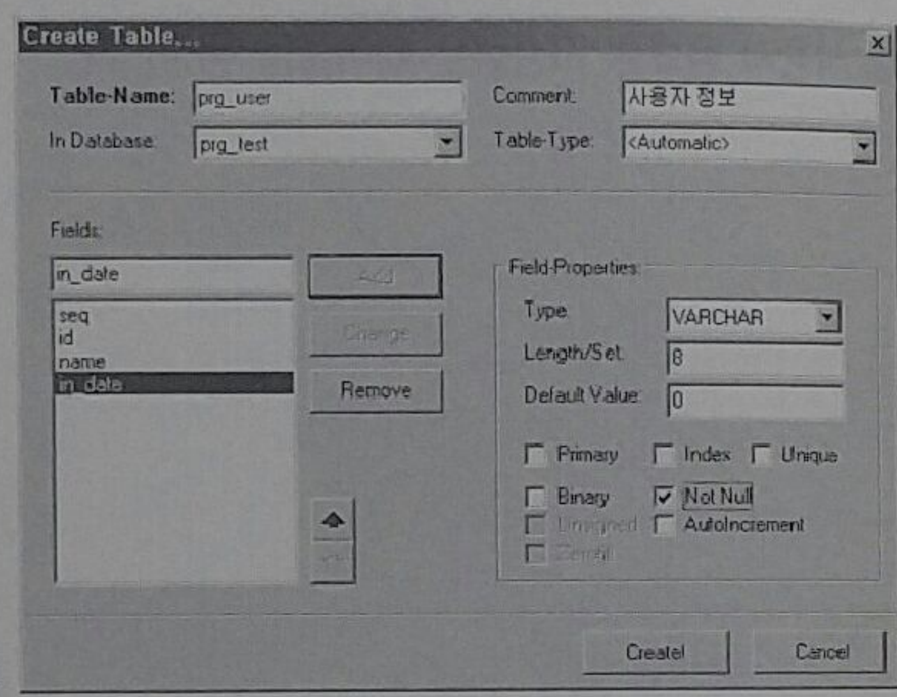
▶ 사용자 id 필드

❺ 사용자 이름을 저장하는 필드를 생성한다.



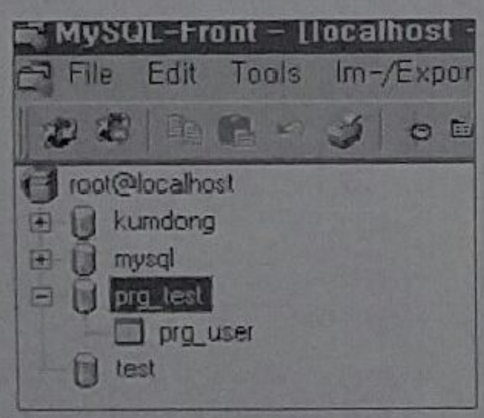
▶ 사용자 이름 필드

❻ 사용자 등록일자 필드를 생성한다.



▶ 사용자 등록일자 필드

❼ 테이블 생성을 확인한다.



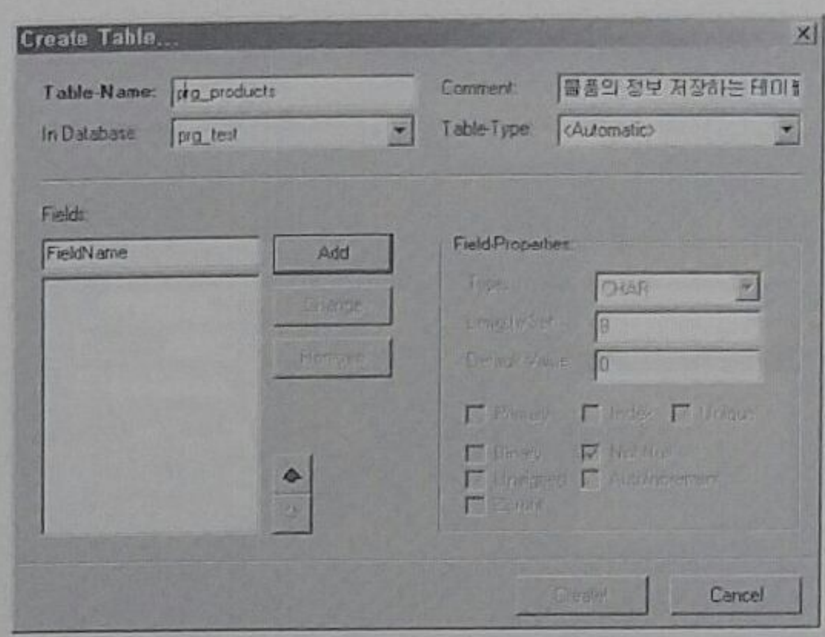
▶ 테이블 생성

사용자 정보를 담고 있는 테이블을 생성했다. 사용자 등록일을 저장하는 필드를 생성할 때 date, datetime 등 날짜 관련 데이터 타입을 선언하지 않고 varchar(m)으로 선언했음을 눈여겨보자.

다음은 물품에 대한 정보를 저장하는 use_products라는 테이블을 생성한다.

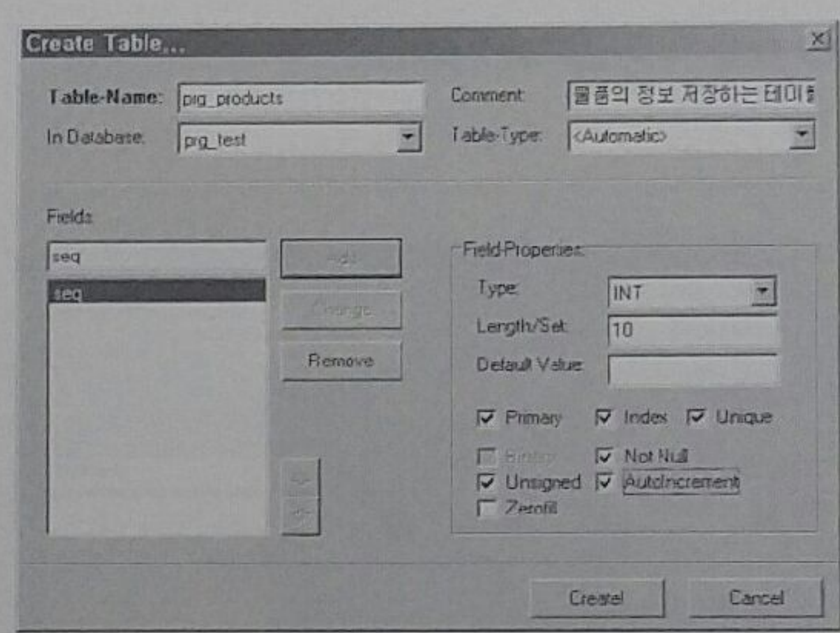
따라하기 물품 정보 테이블 생성

❶ 물품정보를 저장하는 테이블을 생성한다.



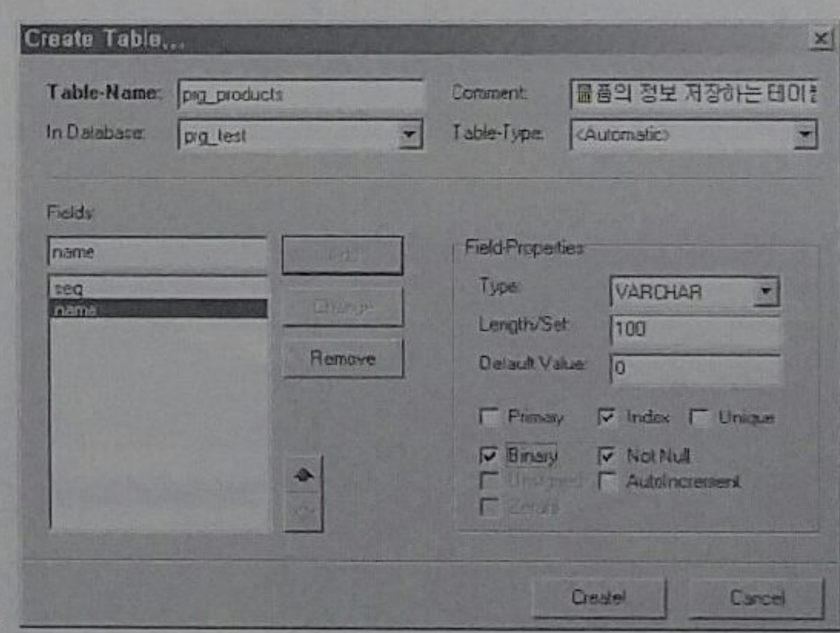
▶ prg_products 생성

❷ 등록 일련번호 필드를 생성한다.



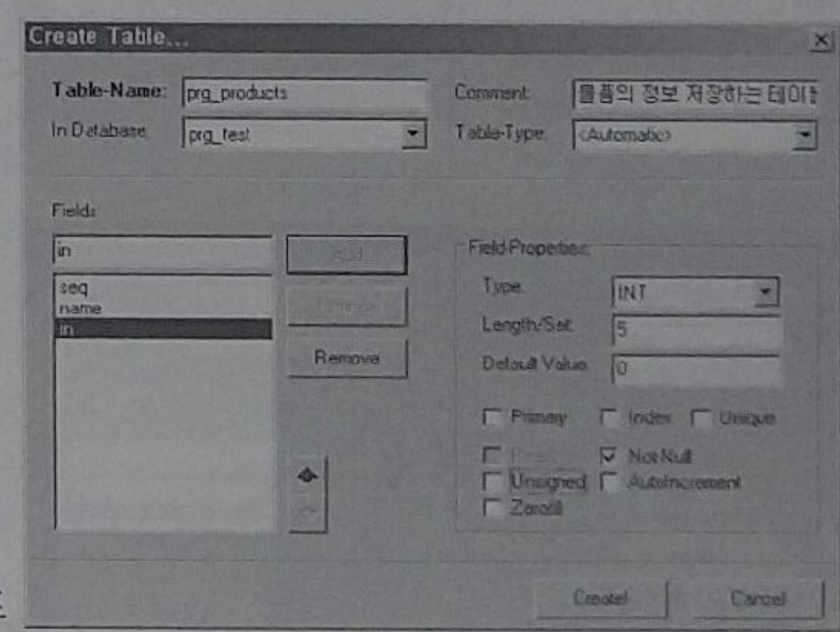
▶ 등록일련번호 생성

❸ 물품 이름이 들어가는 필드를 생성한다.



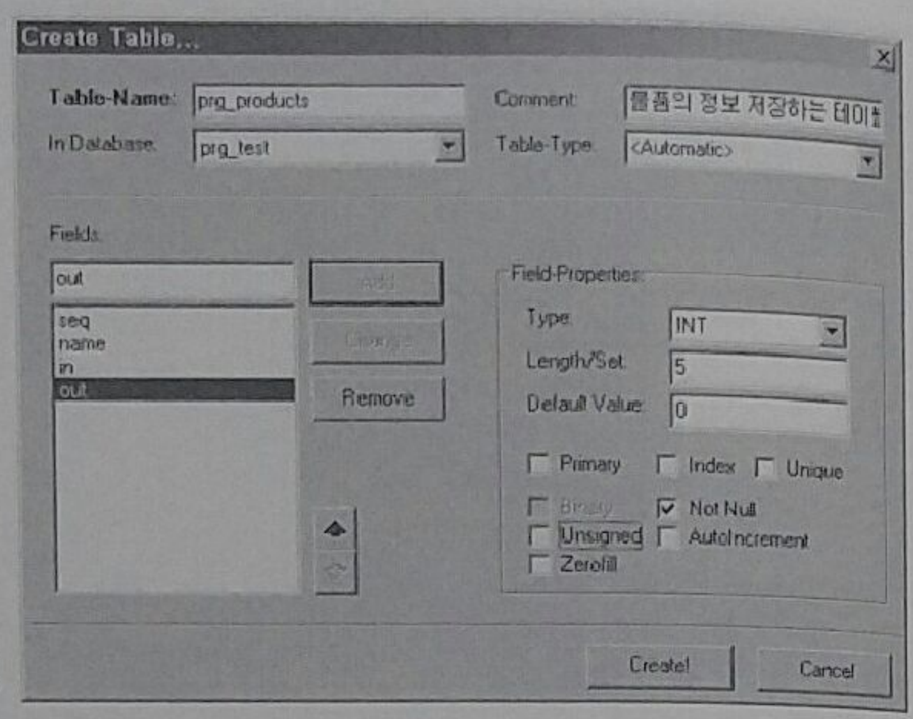
▶ 물품이름 필드

❹ 물품 입고수량 필드를 생성한다.



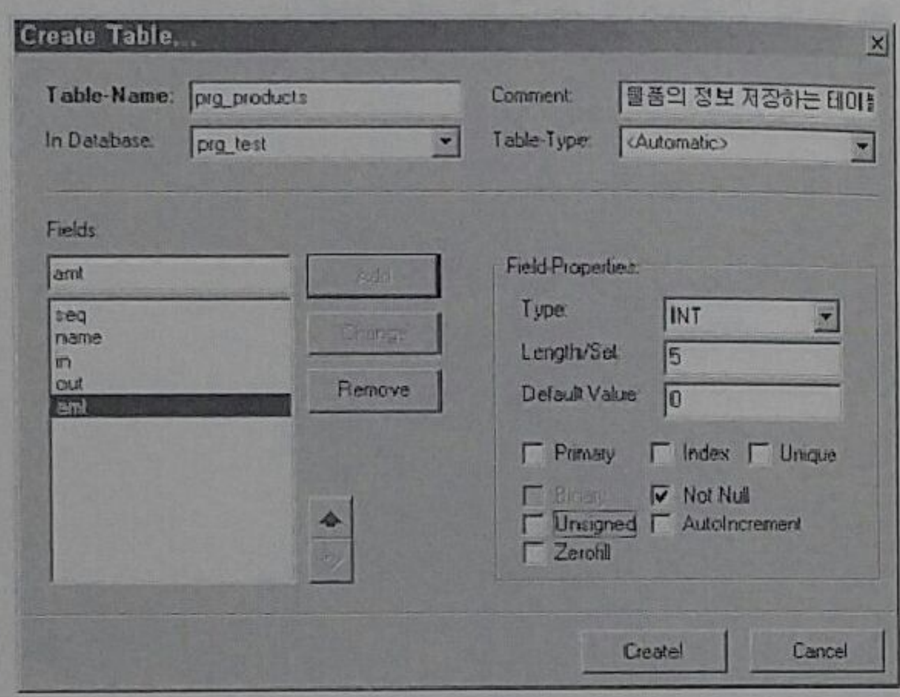
▶ 물품입고 수량 필드

❺ 물품 출고 수량 필드를 생성한다.



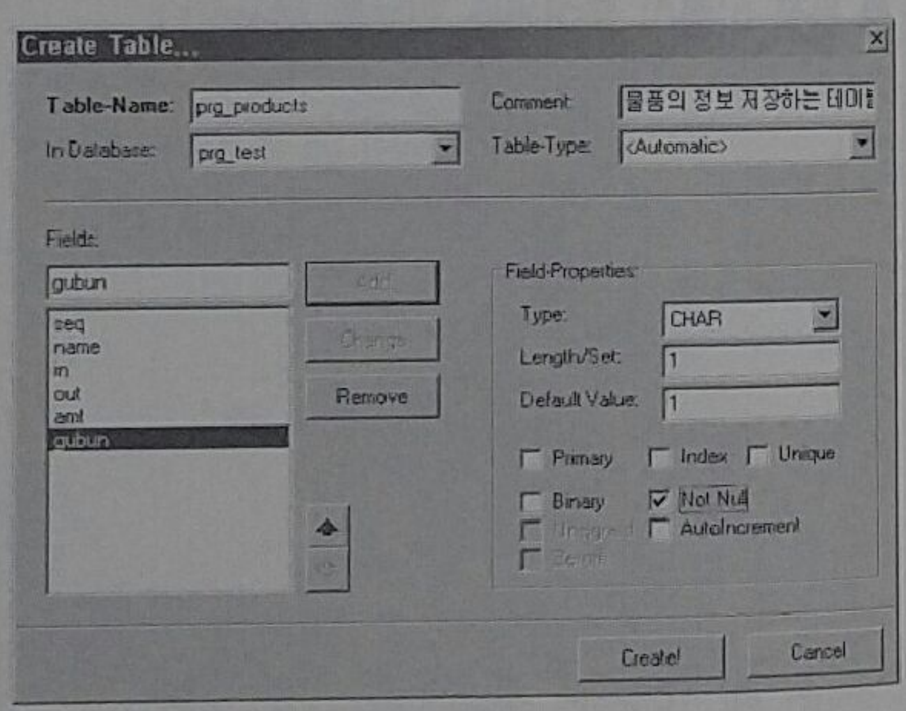
▶ 물품출고 수량 필드

❻ 물품 가격 필드를 생성한다.



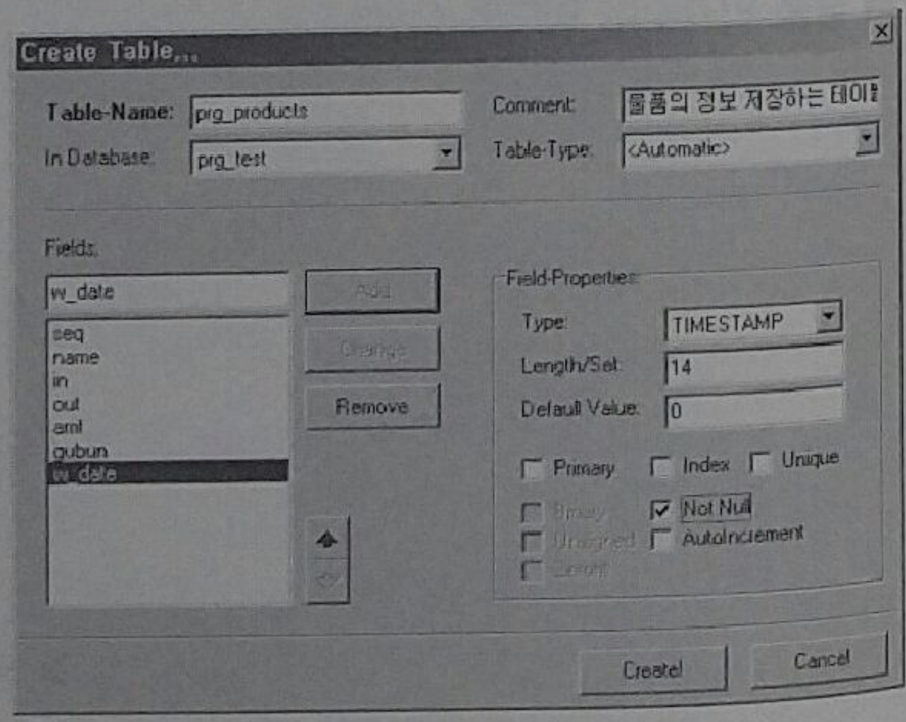
▶ 물품가격 필드

❼ 물품 출고에 따른 구분자 필드를 생성한다.



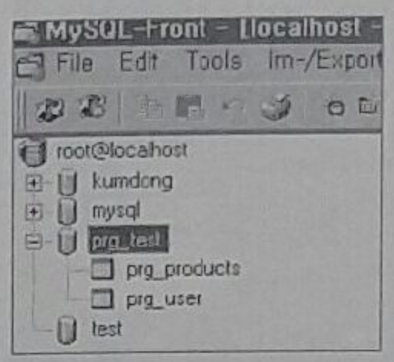
▶ 구분자 필드

❽ 물품 입출고일 필드를 생성한다.



▶ 입출고일자 생성

❾ prg_products 테이블 생성을 확인한다.

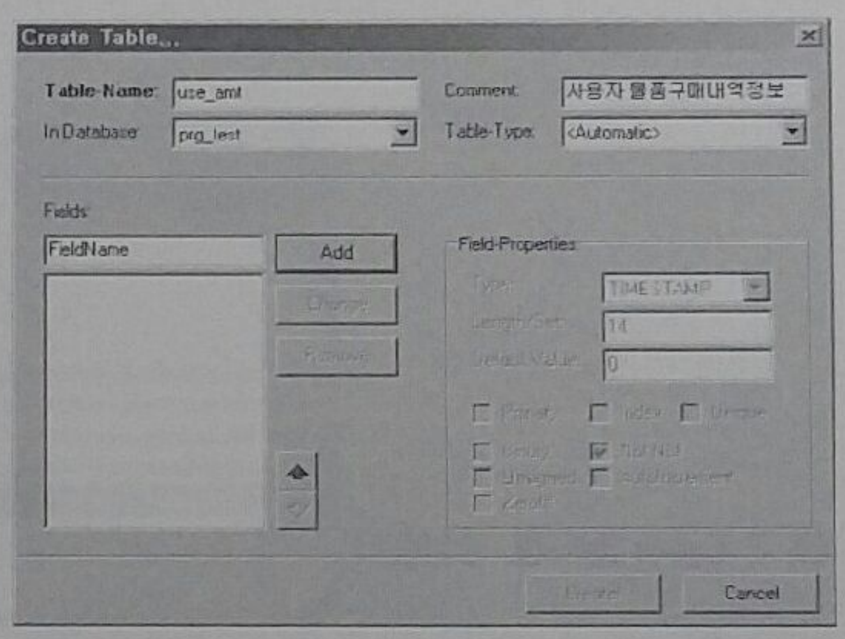


▶ 테이블 생성확인

물품을 저장하는 테이블이 만들어졌다면 이제 사용자가 물품을 구매한 내역을 저장하는 use_amt 테이블을 생성한다.

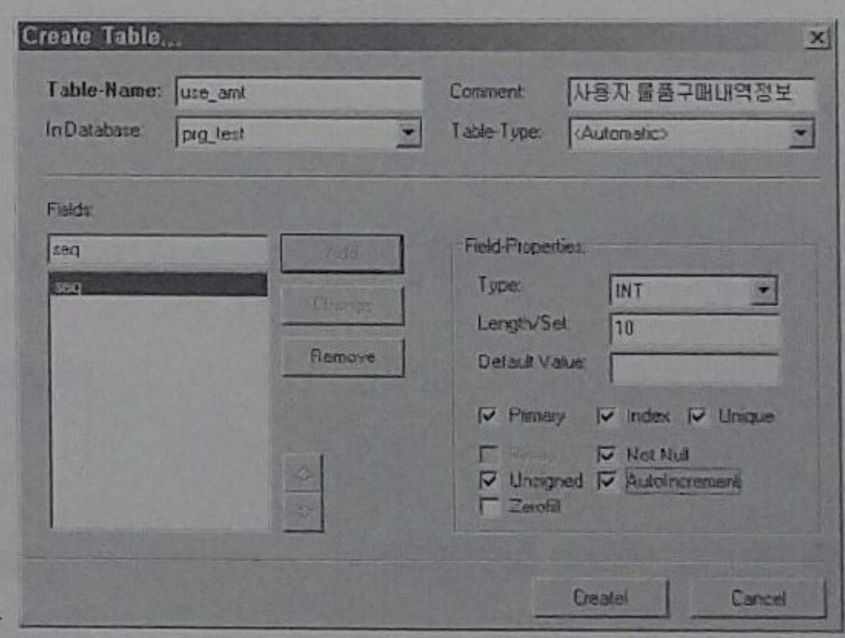
따라하기 구매 내역 테이블 생성

❶ 물품정보를 저장하는 테이블을 생성한다.



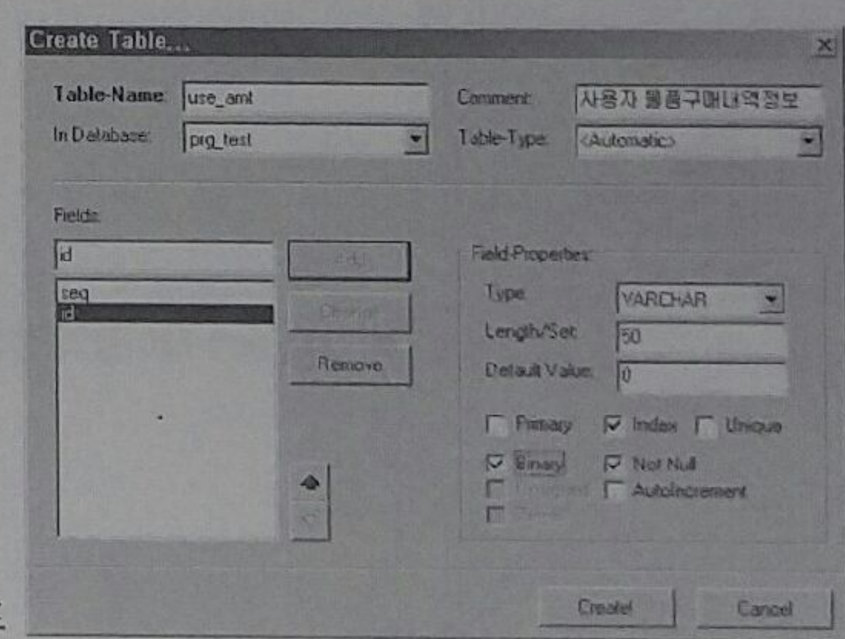
▶ use_amt 생성

❷ 구매일련번호 필드를 생성한다.



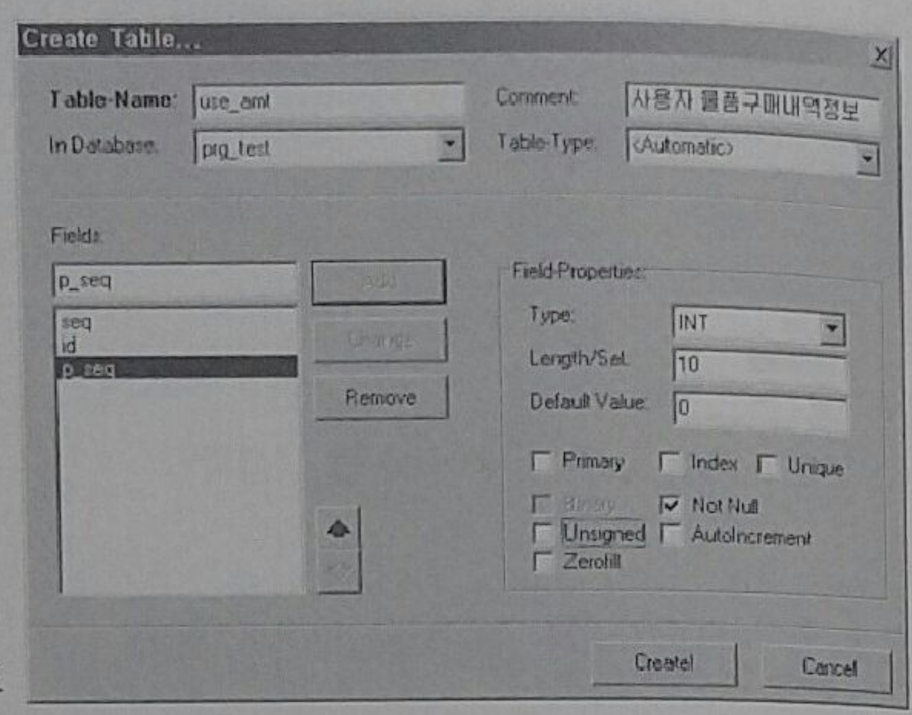
▶ 구매 일련번호 필드

❸ 구매자 id 필드를 생성한다.



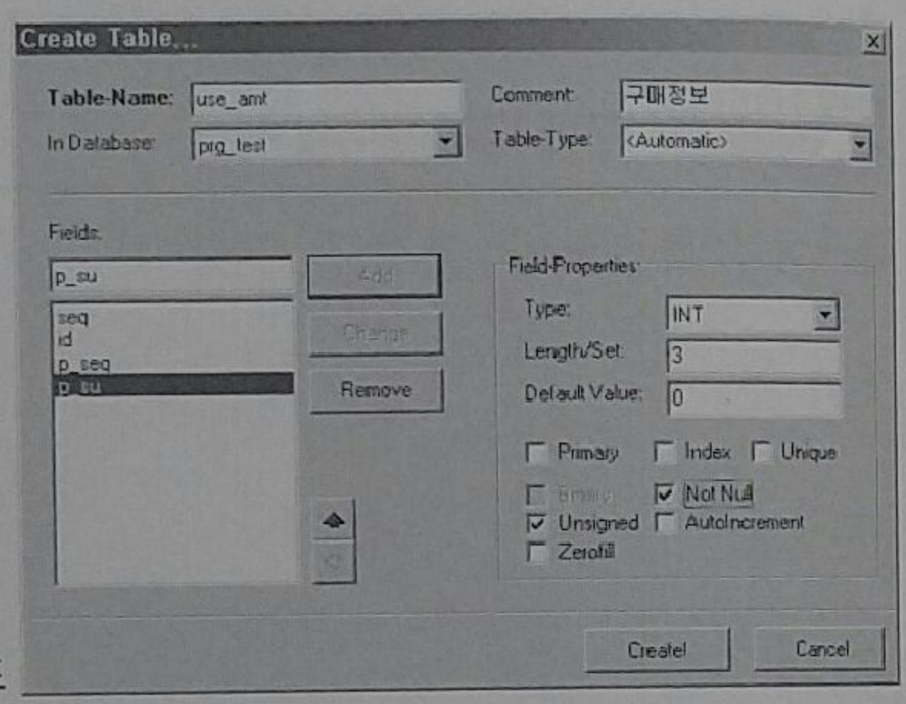
▶ 구매자 id 필드

❹ 구매물품번호를 생성한다.



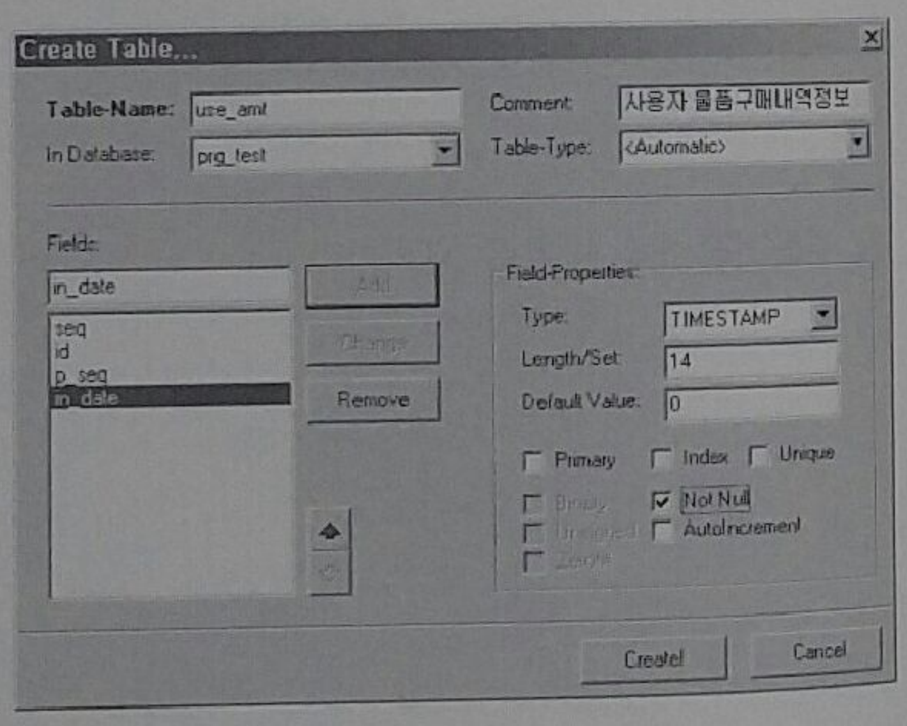
▶ 구매물품번호

❺ 구매수량 필드를 생성한다.



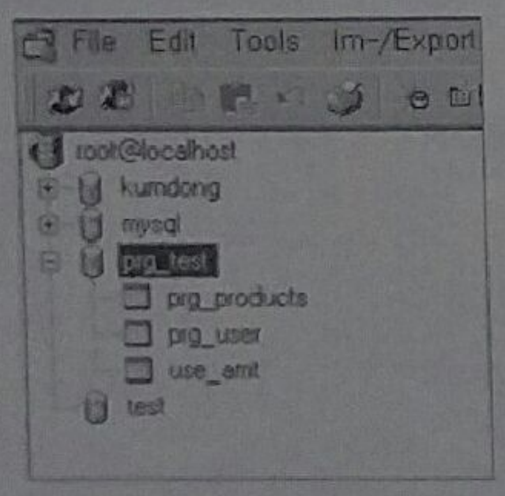
▶ 구매수량 필드

❻ 구매일자 필드를 생성한다.



▶ 구매일자 필드

❼ 테이블 생성을 확인한다.



▶ use_amt 테이블의 생성

테이블이 만들어졌다면 우리는 테이블에 데이터를 insert해야 한다. 그러나 프로그램에서의 insert가 아니라면 우리는 10개의 데이터를 insert 해야 한다. 좀 더 후에 ODBC를 이용한 데이터의 삽입 방법에 대한 부분에 대해 공부하게 된다.

사용자 정보 테이블에 3개의 데이터를 insert한다.

```
Host | Database | Table | Data | Query
SQL-Query on Database prg_test:                                  228 Characters
insert into prg_user values (seq,'kumdong1','김영호',sysdate('yyyymmdd') )
insert into prg_user values (seq,'kumdong2','금동이',sysdate('yyyymmdd') )
insert into prg_user values (seq,'kumdong3','용남이',sysdate('yyyymmdd') )
```

▶ 사용자 데이터의 insert

물품 정보 테이블에 5개의 데이터를 insert한다.

```
Host | Database | Table | Data | Query
SQL-Query on Database prg_test:                                  690 Characters
insert into prg_user values (seq,'kumdong1','김영호',sysdate('yyyymmdd') )
insert into prg_user values (seq,'kumdong2','금동이',sysdate('yyyymmdd') )
insert into prg_user values (seq,'kumdong3','용남이',sysdate('yyyymmdd') )

insert into prg_products values (seq,'연필12세트',3,0,1000,'1',sysdate('yyyymmddhhmmdd') )
insert into prg_products values (seq,'네임펜세트',2,0,2000,'1',sysdate('yyyymmddhhmmdd') )
insert into prg_products values (seq,'스프링노트',5,0,3000,'1',sysdate('yyyymmddhhmmdd') )
insert into prg_products values (seq,'CD케이스(24)',10,0,4000,'1',sysdate('yyyymmddhhmmdd
insert into prg_products values (seq,'A4용지',20,0,5000,'1',sysdate('yyyymmddhhmmdd') )
```

▶ 물품 데이터의 insert

구매내역정보에 5개의 데이터를 insert한다.

만들어진 테이블 3개는 나중에 JSP 프로그램에서도 사용될 것이므로 삭제하지 않는다.

```
insert into prg_products values (seq,'A4용지',20,0,5000,'1',sysdate('yyyymmdd

insert into use_amt values (seq,'kumdong1',1,1,sysdate('yyyymmddhhmmdd') )
insert into use_amt values (seq,'kumdong1',4,2,sysdate('yyyymmddhhmmdd') )
insert into use_amt values (seq,'kumdong2',3,4,sysdate('yyyymmddhhmmdd') )
insert into use_amt values (seq,'kumdong3',2,2,sysdate('yyyymmddhhmmdd') )
insert into use_amt values (seq,'kumdong3',1,2,sysdate('yyyymmddhhmmdd') )

ow(s) affected. 0 field(s), 0 record(s) in last resultset. Time: 0.00 sec.
```

▶ 구매 데이터의 insert

JOIN

각각의 테이블에 있는 정보를 가지고 와서 사용자가 원하는 데이터로 조합하여 결과를 얻을 때 사용한다.

Equi-join

두개의 테이블 또는 그 이상의 테이블에서 공통적인 필드를 연결하여 결과를 가지고 오는 것을 말한다. 즉 A 테이블에 a라는 필드가 있고 B 테이블에 b라는 필드가 있다면 a=b와 같은 데이터를 추출한다.

prg_products 테이블과 use_amt 테이블을 보면 공통적인 필드가 있는데 그건 바로 물품일련번호이다. 구매 내역에서 내가 어떤 물품을 구매했는지의 정보를 가지고 있지만 실제적으로 그 물품에 대한 정보는 prg_products가 가지고 있고 use_amt는 다만 prg_products 테이블 내에 존재하는 물품등록에 대한 일련번호 밖에 가지고 있지 않다. 사용자는 구매 내역 데이터에서 필요한 물품의 이름과 가격을 Equi-join을 통해서 그 결과를 추출할 수 있다.

```
select a.id,a.p_su,a.in_date,b.name,b.amt
from
use_amt a ,
prg_products  b
where
a.p_seq=b.seq
```

0 row(s) affected. 5 field(s), 5 record(s) in last resultset. Time: 0.00 sec.

id	p_su	in_date	name	amt
kumdong1	1	2006-04-17 12:1	연필12세트	1000
kumdong1	2	2006-04-17 12:1	CD케이스(24)	4000
kumdong2	4	2006-04-17 12:1	스프링노트	3000
kumdong3	2	2006-04-17 12:1	네임펜세트	2000
kumdong3	2	2006-04-17 12:1	연필12세트	1000

▶ Equi-join 결과

Equi-join은 다음의 형식을 취한다.

```
select 테이블1.필드1, 테이블1.필드, 테이블2,필드1
       from 테이블1, 테이블2
       where 테이블1.필드=테이블2.필드
```

다음을 보면 use_amt a에서 a는 별칭이고 a.id는 별칭으로 정해진 use_amt 테이블의 id를 뜻한다.

```
select
a.id, a.p_su, a.in_date, b.name, b.amt
from use_amt a, prg_products b
where a.p_seq=b.seq
```

Equi-join을 통해 사용자의 정보까지 추출해 본다.

```
select a.id,c.name,a.p_su,a.in_date,b.name,b.amt
from
use_amt a ,
prg_products  b ,
prg_user c
where
a.p_seq=b.seq
and a.id=c.id
```

0 row(s) affected. 6 field(s). 5 record(s) in last resultset. Time: 0.02 sec.

id	name	p_su	in_date	name_1	amt
kumdong1	김영호	1	2006-04-17 12:1	연필12세트	1000
kumdong1	김영호	2	2006-04-17 12:1	CD케이스(24)	4000
kumdong2	금동이	4	2006-04-17 12:1	스프링노트	3000
kumdong3	용남이	2	2006-04-17 12:1	네임펜세트	2000
kumdong3	용남이	2	2006-04-17 12:1	연필12세트	1000

▶ 세개의 테이블 Equi-join

left-join

left-join은 Equi-join과는 다르게 기준 테이블을 중심으로 join하는 방법이다. 즉 기준 테이블 left join 다른 테이블의 형식으로 사용된다.

쿼리를 보면 prg_products이 기준이 되는 테이블이고 use_amt 테이블이 join되는 테이블이 된다. 따라서 기준에 합당하지 않는 데이터는 하늘색 표시인 데이터가 없음을 나타내게 된다.

```
select
a.name, a.in,
a.amt,b.id,b.p_su
from prg_products a left join use_amt b on a.seq=b.p_seq
```

0 row(s) affected. 5 field(s). 6 record(s) in last resultset. Time: 0.02 sec

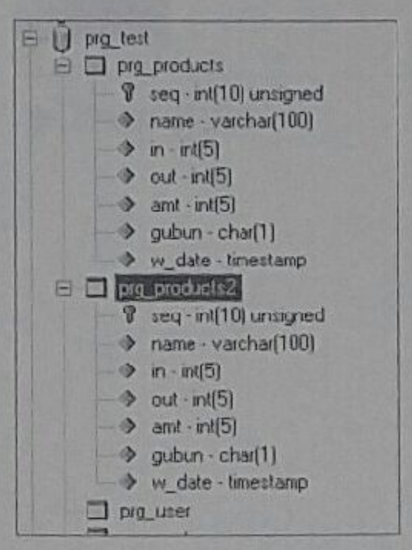
name	in	amt	id	p_su
연필12세트	3	1000	kumdong1	1
연필12세트	3	1000	kumdong3	2
네임펜세트	2	2000	kumdong3	2
스프링노트	5	3000	kumdong2	4
CD케이스(24)	10	4000	kumdong1	2
A4용지	20	5000		

▶ left join의 결과

UNION

두개 또는 두개 이상의 테이블을 결합하여 그 결과를 추출하는 방법이다. 단 조건은 결합하려는 테이블의 필드 데이터 형이 일치하여야만 union이 될 수 있다.

prg_products 테이블과 동일한 필드, 동일한 데이터 형태의 prg_products2 테이블을 만든다.



▶ 생성된 prg_products2 테이블

약간의 변화를 주어 prg_products2에 데이터를 insert한다.

```
insert into prg_products2 values (seq,'연필12세트2',3,0,1000,'1',sysdate('yyyymmddhhmmdd')
insert into prg_products2 values (seq,'네임펜세트2',2,0,1000,'1',sysdate('yyyymmddhhmmdd')
insert into prg_products2 values (seq,'스프링노트2',5,0,1000,'1',sysdate('yyyymmddhhmmdd')
insert into prg_products2 values (seq,'CD케이스(24)2',10,0,1000,'1',sysdate('yyyymmddhhmmd
insert into prg_products2 values (seq,'A4용지2',20,0,1000,'1',sysdate('yyyymmddhhmmdd') )
```

row(s) affected. 0 field(s), 0 record(s) in last resultset. Time: 0.00 sec.

▶ prg_products2 insert

union을 사용하여 두개의 테이블을 결합한다.

```
select * from prg_products
union
select * from prg_products2
```

row(s) affected. 7 field(s), 10 record(s) in last resultset. Time: 0.00 sec.

seq	name	in	out	amt	gubun	w_date
1	연필12세트	3	0	1000	1	2006-04-17 12:0
2	네임펜세트	2	0	2000	1	2006-04-17 12:0
3	스프링노트	5	0	3000	1	2006-04-17 12:0
4	CD케이스(24)	10	0	4000	1	2006-04-17 12:0
5	A4용지	20	0	5000	1	2006-04-17 12:0
1	연필12세트2	3	0	1000	1	2006-04-17 13:1
2	네임펜세트2	2	0	1000	1	2006-04-17 13:1
3	스프링노트2	5	0	1000	1	2006-04-17 13:1

▶ union의 결과

맺음말

이번 장에서는 앞 장에서 배운바 있는 Select, insert, update, delete에 대해 좀 더 알아보았다. 또한 select에 따른 사용옵션에 대해서도 살펴보았다. 아울러 where와 Front에서의 필드 추가와 수정 삭제 그리고 Join과 Union에 대해서도 공부하였다. 몇 가지 빠진 부분이 있다면 그것은 그리 중요하게 다뤄지지 않는 부분이라 생각해도 좋다. 예제에서 만들어진 테이블은 추후에 JSP와 연동되는 프로그램 예제에 사용된다. 다음 장에서 MySql이 가지는 함수에 대해서 공부하게 된다.

05 MySql 함수와 백업

안녕하세요

이번 장에서는 MySql에서 사용할 수 있는 함수에 대해서 공부한다. 쿼리문을 사용할 때 유용하게 사용되는 것이 이 함수 부분이다. 함수에는 여러 가지의 종류가 있는데 많이 사용되는 함수는 기억하고 있는 것이 좋다. 또한 함수 부분을 공부하고 Front에서의 데이터 백업방법과 그리고 dump를 통한 백업방법에 대해서 알아보자.

MySql 함수

함수를 사용한다는 것은 수학공식에 수치를 대입하여 결과를 얻는 것과 같다고 생각하면 된다. 예를 들어 ABS(-50)이라고 하면 MySql에서는 -50의 절대 값인 50을 리턴할 것이다. 따라서 함수는 어떤 결과 값을 얻기 위해 편리하게 사용되는 것이다.

char(숫자1, 숫자2..)

아스키코드에 대응하는 문자를 추출한다.

```
select * from prg_products
union
select * from prg_products2

select char(66)
```

0 row(s) affected 1 field(s), 1 record(s) in last resultset Time

char(66
B

▶ char 결과

ABS(숫자)

주어진 값의 절대 값을 추출한다.

```
select * from prg_proc

select char(66)
select abs(-99)
```

0 row(s) affected. 1 field(s), 1 record(s) in last re

abs(-99)
99

▶ ABS 결과

curdate() or current_date()

현재 시스템의 날짜를 YYYY-MM-DD 형식으로 추출한다.

```
select char(66)
select abs(-99)

select curdate()
select current_date()
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset.

current_date(
2006-04-17

▶ curdate /current_date 결과

now() or sysdate()

현재 시스템의 날짜와 시간을 추출한다.

```
select curdate()
select current_date()

select now()
select sysdate()
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset. T

sysdate()
2006-04-17 오후 8:08:36

▶ now/ sysdate 결과

curtime() or current_time()

현재 시스템의 시간을 오전/오후 hh:mm:ss 형식으로 추출한다.

```
select now()
select sysdate()

select curtime()
select current_time()
```

0 row(s) affected, 1 field(s), 1 record(s) in last resultse

current_time()
오후 8:24:13

▶ curtime / current_time 결과

dayofmonth(now() or sysdate())

현재 시스템에서 며칠인지 일자를 추출한다.

```
select curtime()
select current_time()

select dayofmonth(now())
select dayofmonth(sysdate())
```

row(s) affected, 1 field(s), 1 record(s) in last resultset. Time: 0.0

dayofmonth(sysdate
17

▶ dayofmonth() 결과

dayofweek(now() or sysdate()) or weekday(now() or sysdate())

현재 시스템에서 요일을 추출하며 1: 일요일 2:월요일... 이다.

```
select dayofmonth(now())
select dayofmonth(sysdate())

select dayofweek(now())
select dayofweek(sysdate())
```

0 row(s) affected, 1 field(s), 1 record(s) in last resultset. Time: 0.02 s

dayofweek(sysdate(
2

▶ dayofweek / weekday 결과

dayofyear('yyyy-mm-dd')

현재 시스템에서 일년 중에 며칠이 지났는가를 추출한다. 주어진 yyyy 기준.

```
select dayofweek(now())
select dayofweek(sysdate())

select dayofyear('2006-02-28')
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset. Time: 0.00 s

dayofyear('2006-02-
59

▶ dayofyear 결과

encrypt('문자')

주어진 문자 또는 문자열을 Unix crypt()로 암호화하며 옵션 중에 salt가 주어진 경우에는 암호를 만드는데 사용된다. OS가 윈도우인 경우에는 null값을 보여준다. 왜냐하면 Unix crypt()가 없기 때문이다.

```
select dayofyear('2006-02-28')

select encrypt("kumdong")
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset. Time: 0.0

encrypt("kumdong"

▶ encrypt 결과

if(true or false, 값1, 값2)

주어진 조건이 참이면 값1을 거짓이면 값2를 추출한다. if는 select문에서 유용하게 사용됨을 기억하자.

```
select dayofyear('2006-02-28')

select encrypt("kumdong")

select if( 4 > 6 ,'T','F')
```

row(s) affected. 1 field(s), 1 record(s) in last resultset. Time: 0.0

if(4 > 6 ,'T','F')
F

▶ if 결과

upper(문자열)

주어진 문자열을 대문자로 변경한다.

```
select upper('kumdong')
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset. Time:

upper('kumdong'
KUMDONG

▶ upper 결과

lower(문자열)

주어진 문자열을 소문자로 변경한다.

```
select upper('kumdong')

select lower('KUMSDONG')
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset. T

lower('KUMSDONG
kumsdong

▶ lower 결과

mid(문자열, 시작, 길이)

주어진 문자열에서 시작을 기준으로 원하는 길이 만큼의 문자를 추출한다.

```
select lower('KUMSDONG')

select mid('kumdong',1,3)
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset. Time: 0.0

mid('kumdong',1,3
kum

▶ mid 결과

quarter(now() or sysdate())

1-3월 1분기, 4-6월 2분기, 7-9월 3분기, 10-12월 4분기인 것처럼 현재 시스템의 일자를 기준으로 몇 분기에 해당되는지를 추출한다.

```
select lower('KUMSDONG')

select mid('kumdong',1,3)

select quarter(now())
select quarter(sysdate())
```

row(s) affected. 1 field(s), 1 record(s) in last resultset. Time: 0.02 se

quarter(sysdate())
2

▶ quarter 결과

mod(숫자1, 숫자2)

숫자1로 숫자2를 나누었을 때 그 나머지를 추출한다.

```
select quarter(now())
select quarter(sysdate())

select mod(5,3)
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset. Time: 0

mod(5,3)
2

▶ mod 결과

replace(문자열, 기존문자, 새 문자)

주어진 문자열에서 기존문자를 새로운 문자로 변경한다.

```
select mod(5,3)

select replace ('kumdong babo', 'babo','manse')
```

row(s) affected. 1 field(s), 1 record(s) in last resultset. Time: 0.00 sec.

replace ('kumdong babo', 'babo','manse')
kumdong manse

▶ replace 결과

reverse(문자열)

주어진 문자열을 반대로 추출한다.

```
select mod(5,3)

select replace ('kumdong babo', 'babo','manse')

select reverse('kumdong')
```

0 row(s) affected. 1 field(s). 1 record(s) in last resultset. Time: 0.00 sec.

reverse('kumdong'
gnodmuk

▶ reverse 결과

round(소수점 숫자)

주어진 실수의 소수점 아래의 것을 반올림(.5 이상만 반올림)한다.

```
select round(5.6)
```

0 row(s) affected. 1 field(s). 1 record(s) in last resu

round(5.6)
6

▶ round 결과

trim(문자열)

주어진 문자열의 좌우 공백을 제거하고 문자열을 추출한다.

```
select round(5,4)

select trim('  kumdong  ')
```

0 row(s) affected. 1 field(s). 1 record(s) in last resultset. Time: 0.00

trim(' kumdong ')
kumdong

▶ trim 결과

rtrim(문자열)

주어진 문자열의 오른쪽 공백을 제거하고 문자열을 추출한다.

```
select trim('   kumdong   ')

select rtrim('kk2   ')
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset. Time:

rtrim('kk2 ')
kk2

▶ rtrim의 결과

concat(문자열1, 문자열2)

주어진 문자 또는 문자열을 결합해서 추출한다.

```
select rtrim('kk2   ')

select concat('금동이','만세')
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset. Time: 0.00 sec.

concat('금동이','만세')
금동이만세

▶ concat 결과

substring(문자열, 숫자1, 숫자2)

주어진 문자열의 숫자1에서부터 숫자2까지의 데이터를 추출한다.

```
select concat('금동이','만세')

select substring('abcdefghijklmnopqrstuvwxyz',1,5)
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset. Time: 0.00 sec.

substring('abcdefghi
abcde

▶ substring 결과

lpad(문자 또는 숫자, 길이, 삽입문자)

lpad는 주어진 문자 또는 숫자를 포함한 길이만큼 삽입문자를 자리 수만큼 앞쪽에서 채워준다.

```
select substring('abcdefghijklmnopqrstuvw
select lpad('hi',10,'0')
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset. Time: 0.00 sec.

lpad('hi',10,'0'
00000000hi

▶ lpad 결과

rpad(문자 또는 숫자, 길이, 삽입문자)

rpad는 주어진 문자 또는 숫자를 포함한 길이만큼 삽입문자를 자리 수만큼 뒤쪽에서 채워준다.

```
select lpad('hi',10,'0')
select rpad('hi',10,'0')
```

0 row(s) affected. 1 field(s), 1 record(s) in last resultset. Time:

rpad('hi',10,'0'
hi00000000

▶ rpad 결과

이 외에 다른 함수도 있지만 24가지 정도의 함수만 알고 있어도 개발하는데 어려움이 없을 것이다.

MySql의 백업

데이터를 삭제한 관리자는 용서할 수 있어도 데이터를 백업하지 않는 관리자는 용서할 수 없다는 말처럼 데이터의 백업은 아주 중요한 부분이다. 만약 서비스중인 데이터를 일정한 일자 또는 일정한 시간에 백업하지 않는다면 어떠한 예기치 못한 상황이 발생했을 경우 커다란 문제를 발생시킨다. MySql이 아무리 무료버전이라고 하지만 기능 중에 당연히 백업부분이 포함이 되어 있다. MySql에서는 백업에는 여러 가지 방법이 있다. 그 중에서 가장 보편화된 방법과 front에서의 백업방법에 대해서 설명한다.

Front에서의 백업

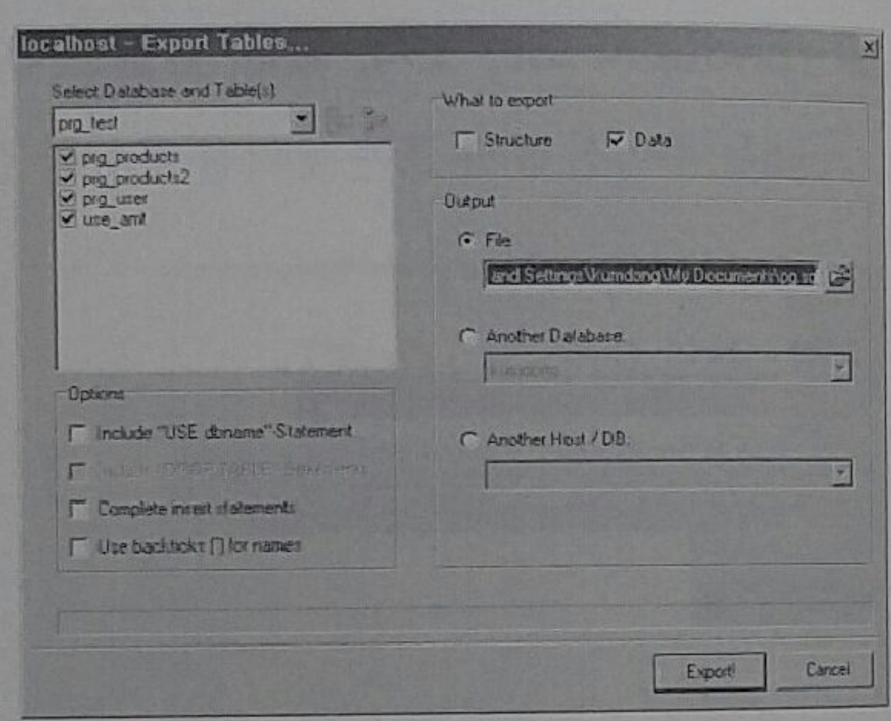
데이터의 양이 적다면 상관이 없겠지만 데이터의 양이 많다면 Front에서 백업의 방법은 시간이 다소 오래 걸린다는 것이다. 그에 반해서 서버에 접속하여 mysql에서 제공되는 백업은 시간이 그리 오래 걸리지는 않는다는 것이고 Linux에 cron에 자동백업을 설치해두면 백업이 다소 편안하다.

Front에서 테이블의 File 백업

앞서 배운 Front의 메뉴 중에 Im/Export 메뉴 중에 Export table을 선택하고 백업을 하고자 하는 데이터베이스 선택 → 테이블 선택 → out에 대한 옵션결정 후 백업을 하면 된다.

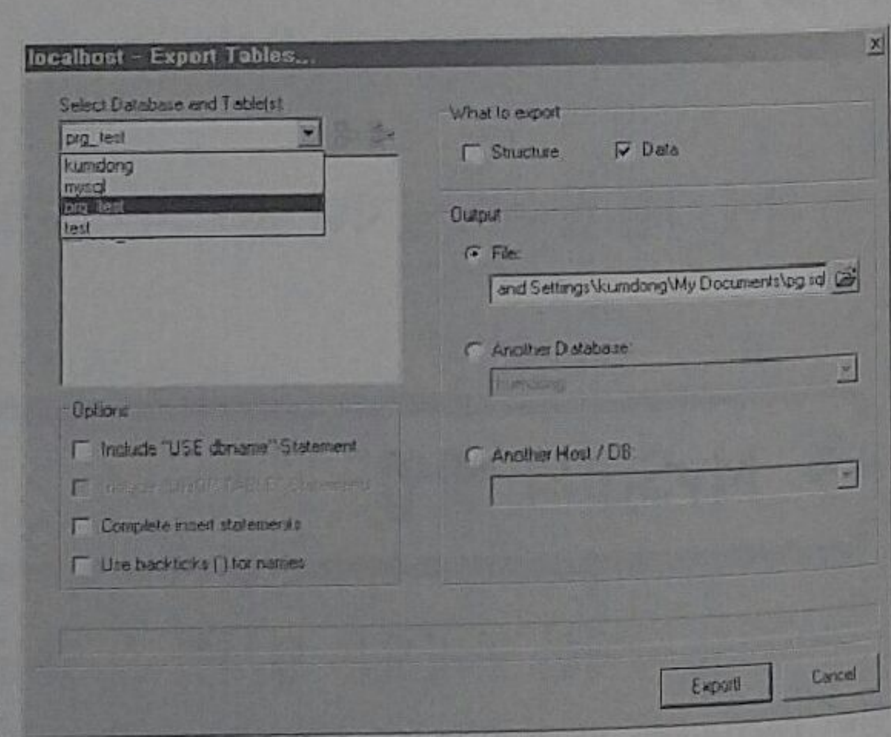
테이블의 파일 백업

❶ Im/Export 메뉴 중 Export table을 선택한다.



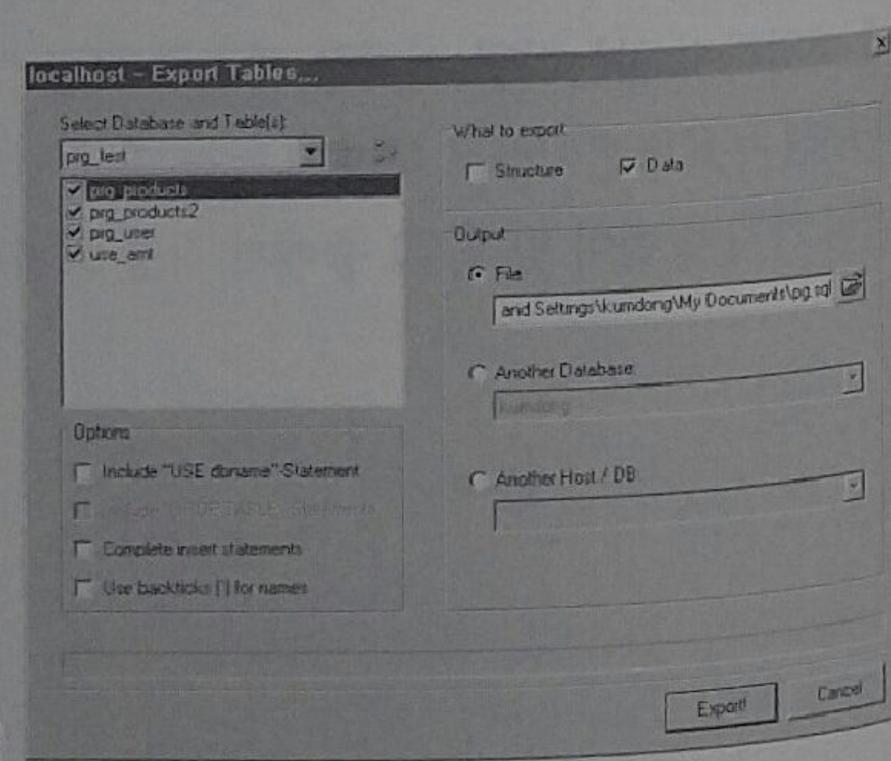
▶ Export table 창

❷ 백업을 원하는 데이터베이스를 선택한다.



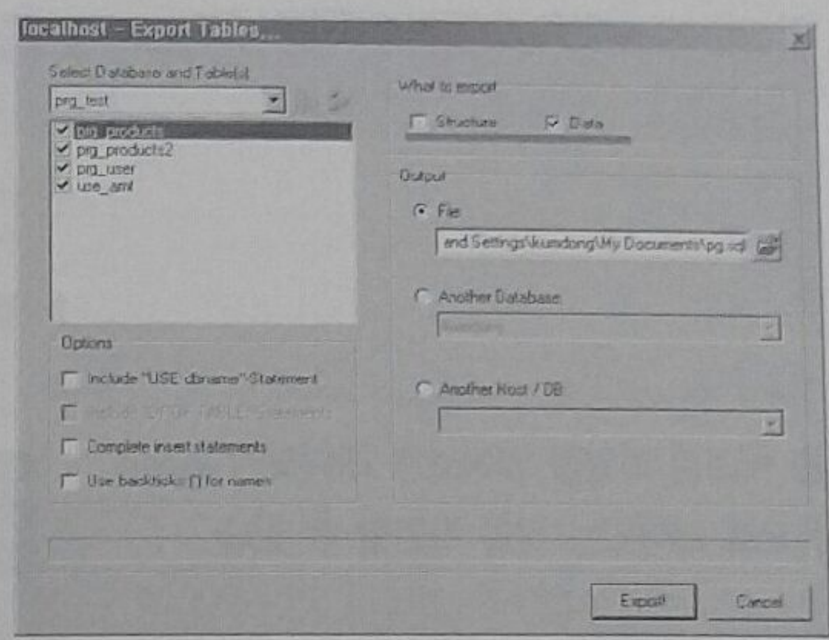
▶ 데이터베이스의 선택

❸ 체크박스 옵션을 통해 백업을 받고자 하는 테이블을 선택한다. 기본적으로 모두 선택이 되어 있다.



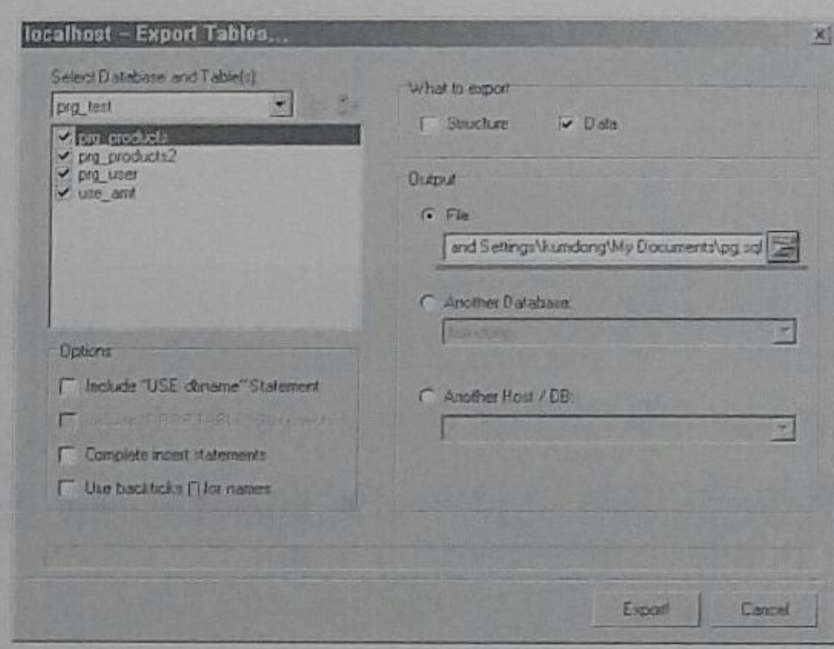
▶ 테이블의 선택

❹ 백업할 테이블의 데이터 중에서 구조 또는 데이터를 포함할 것인지 선택을 한다. 기본적으로 데이터만 선택이 되어 있다.



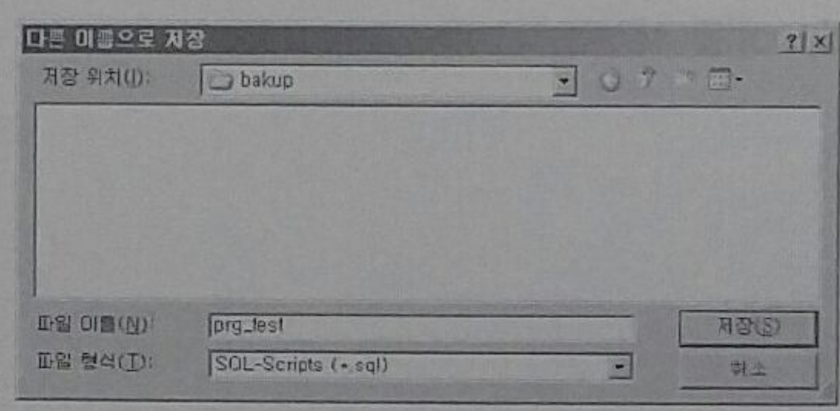
▶ 옵션 선택

❺ 보관할 데이터의 OutPut을 설정하는 것으로 기본적으로 File 옵션으로 선택 되어 있다. 백업의 데이터는 사용자가 원하는 디렉토리에 위치시킬 수 있다.



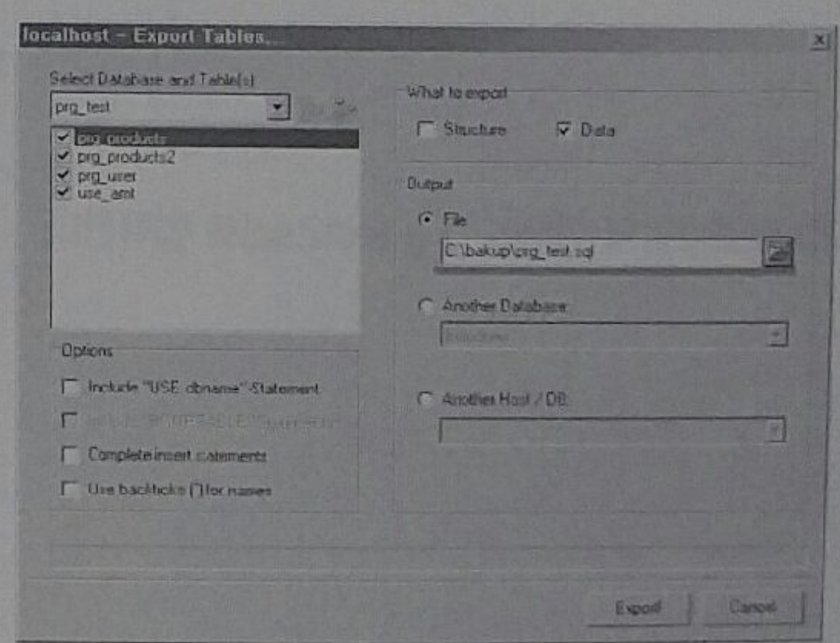
▶ output File

❻ 저장하기 아이콘을 선택하면 사용자가 원하는 디렉토리에 저장할 수 있다.



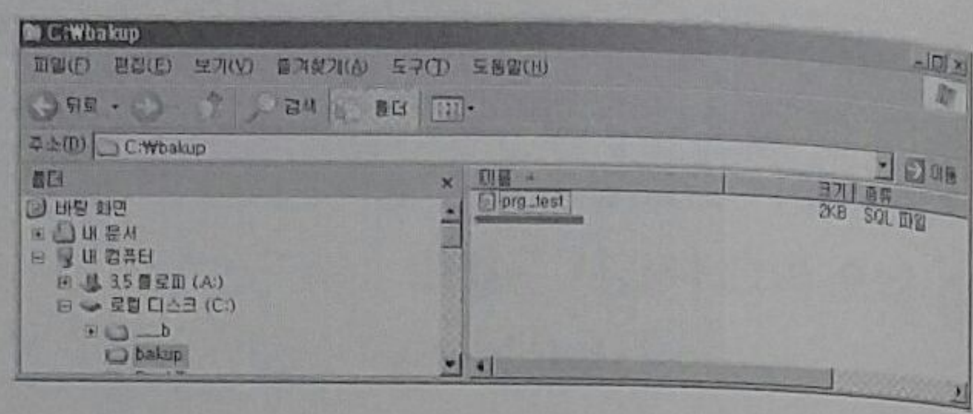
▶ 사용자 백업 파일 저장 지정

❼ 백업될 파일의 이름과 디렉토리를 저장하면 경로가 설정되어 있는 것을 확인할 수 있다.



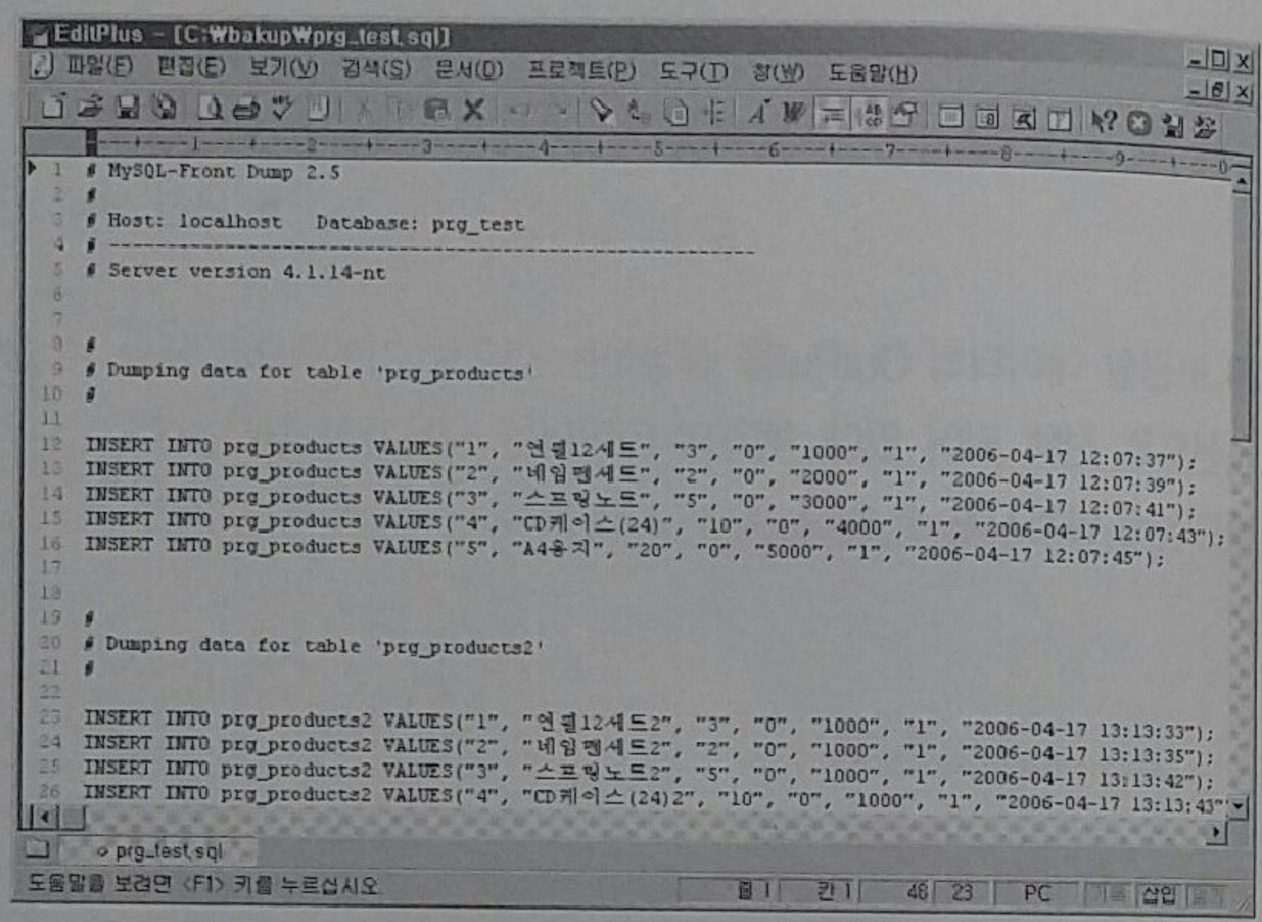
▶ 백업 경로 설정 및 백업 파일의 이름

❽ Export 버튼을 클릭하면 백업이 된다.



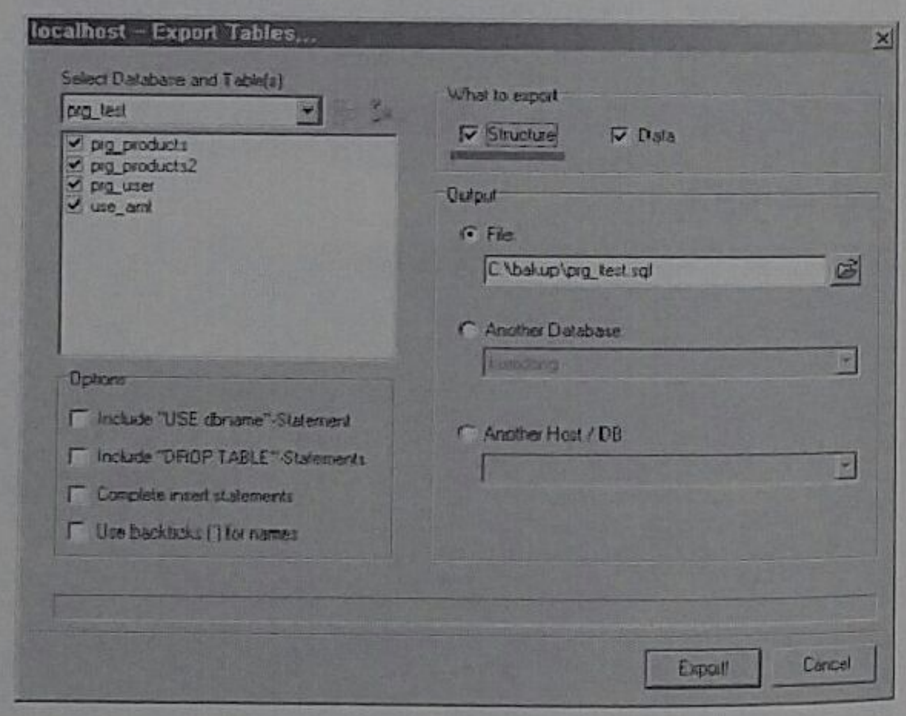
▶ 백업파일 확인

❾ 백업된 파일을 열어보면 데이터만 insert 문으로 되어있는 것을 확인할 수 있다.



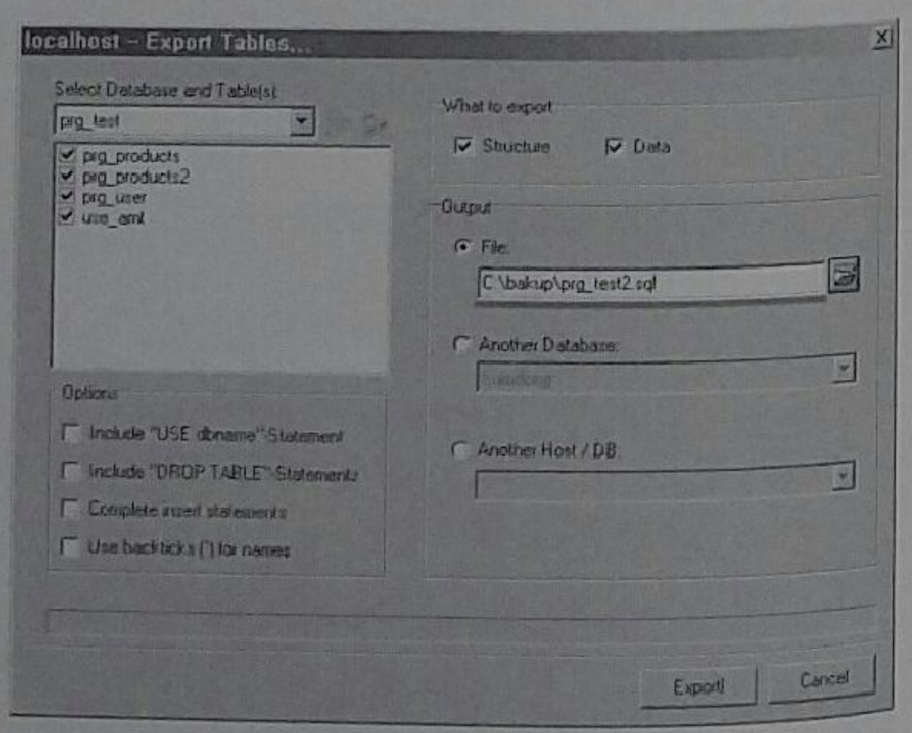
▶ 백업파일의 내용

❿ 백업의 옵션 중에 Structure 옵션을 체크한 후 같은 방법으로 Export를 시킨다.



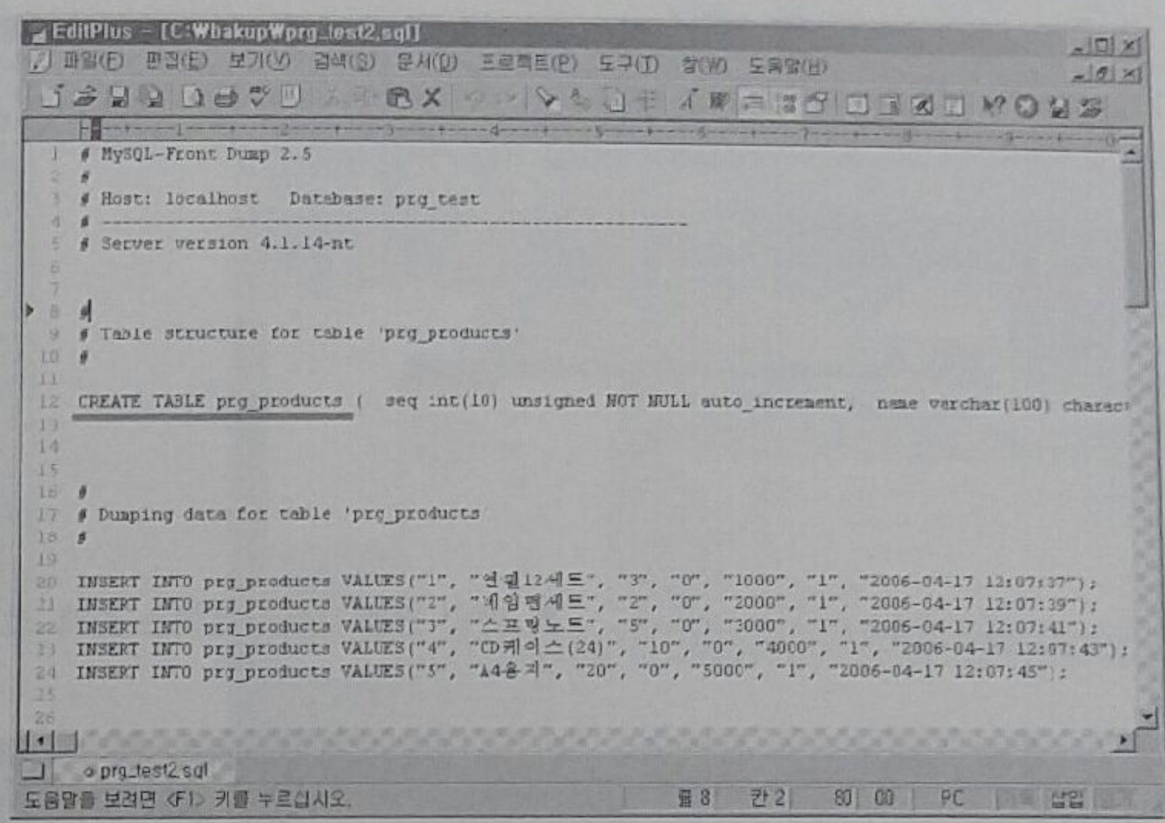
▶ Structure 선택

⓫ 백업파일의 이름을 prg_test2.sql로 입력한다.



▶ 이름변경

⓬ Export를 하여 백업파일의 내용을 보면 create .. 등 테이블의 구조 생성 또는 백업이 되어 있는 것을 확인할 수 있다.



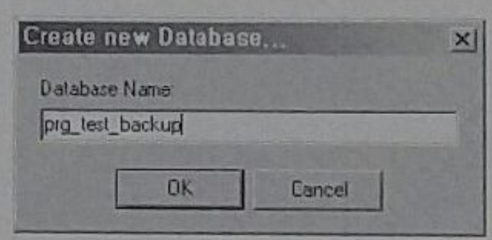
▶ Structure 선택 후 백업파일 내용

Front에서 테이블의 같은 서버 내 데이터베이스로 백업

File 백업으로 파일을 남기는 것이 아니라 또 다른 데이터베이스로 테이블 자체를 통째로 백업하는 방법이다. 데이터의 양이 적을 때는 상관이 없지만 데이터의 양이 많을 경우에 시간이 다소 걸린다는 단점이 있다.

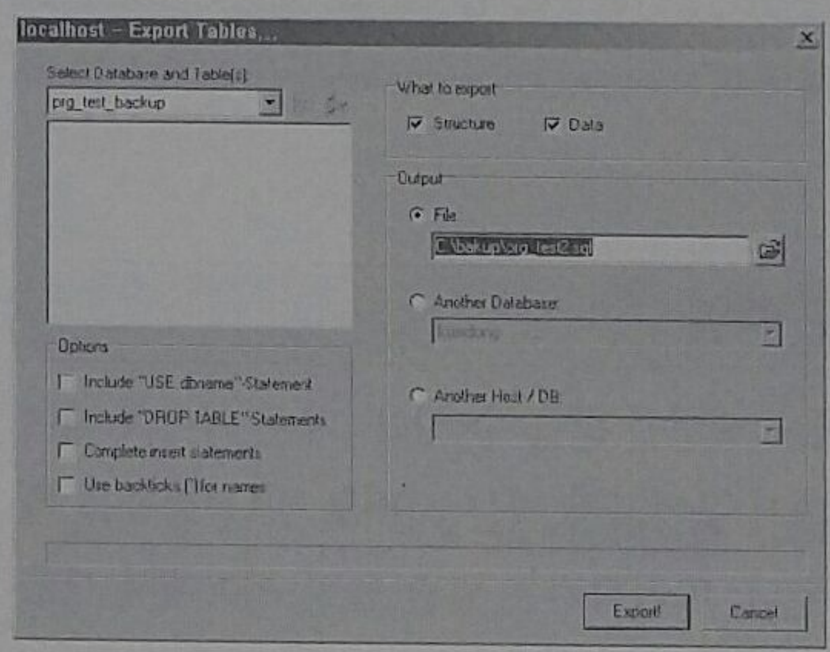
따라하기 테이블을 같은 서버로 백업

❶ Tools의 Create Database를 통해 새로운 데이터베이스를 만든다.



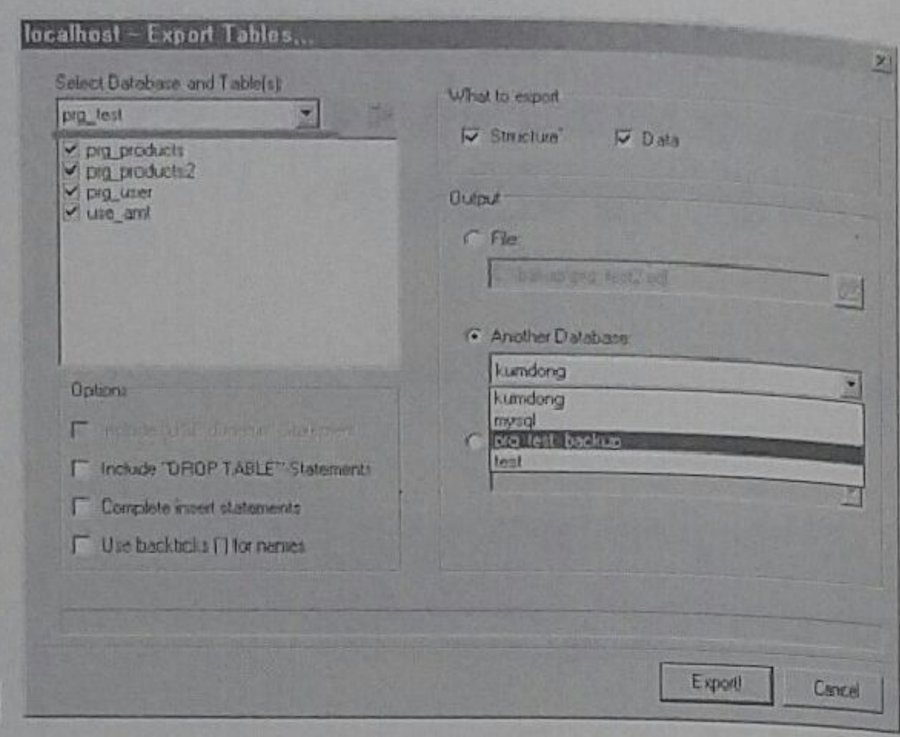
▶ 백업할 데이터베이스 생성

❷ Im/Export 메뉴의 Export tables 메뉴를 선택한다.



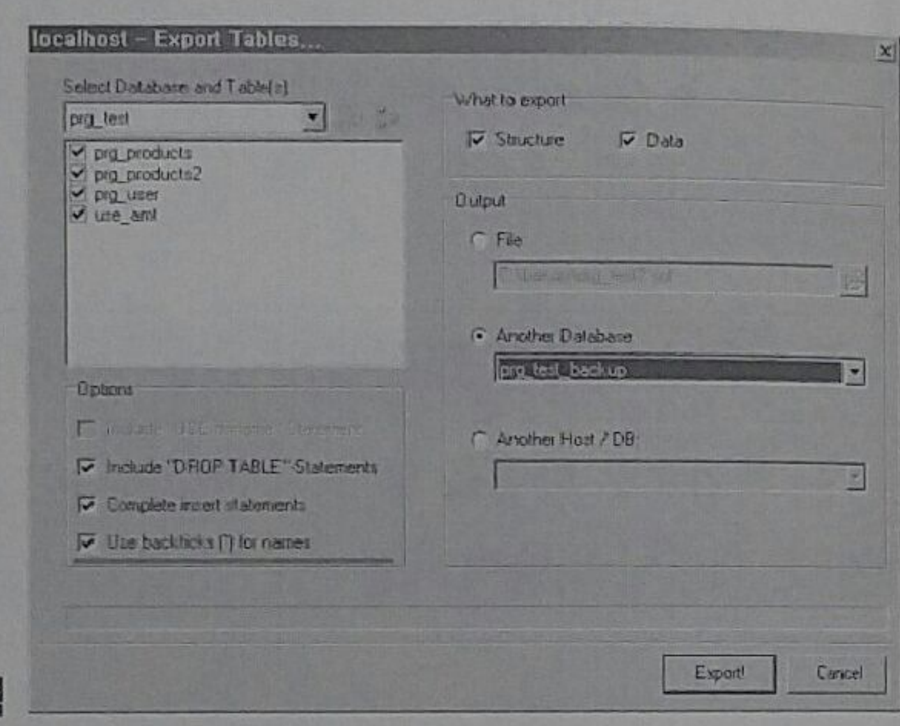
▶ Export tables 창

❸ 원본 데이터베이스를 선택한다.



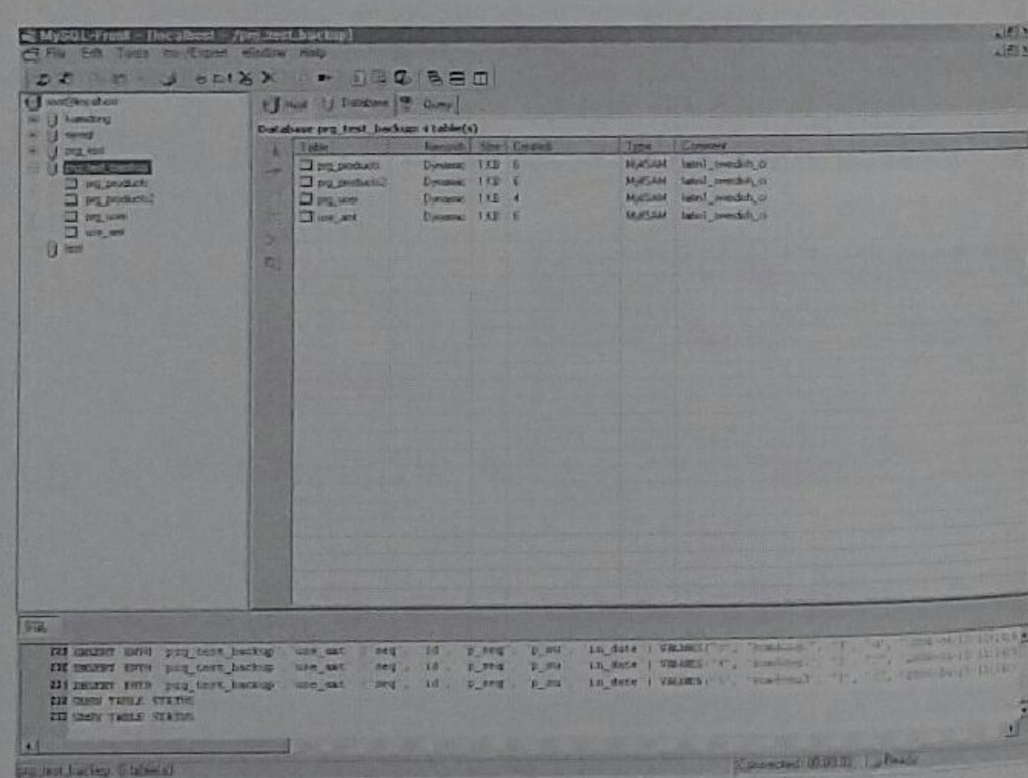
▶ 백업 원본 데이터베이스 선택

❹ Another Database 옵션에서 백업을 받을 데이터베이스를 선택한다.



▶ 백업받을 데이터베이스 선택

❺ 모든 옵션을 선택하고 Export 버튼을 클릭하면 백업이 시작된다.



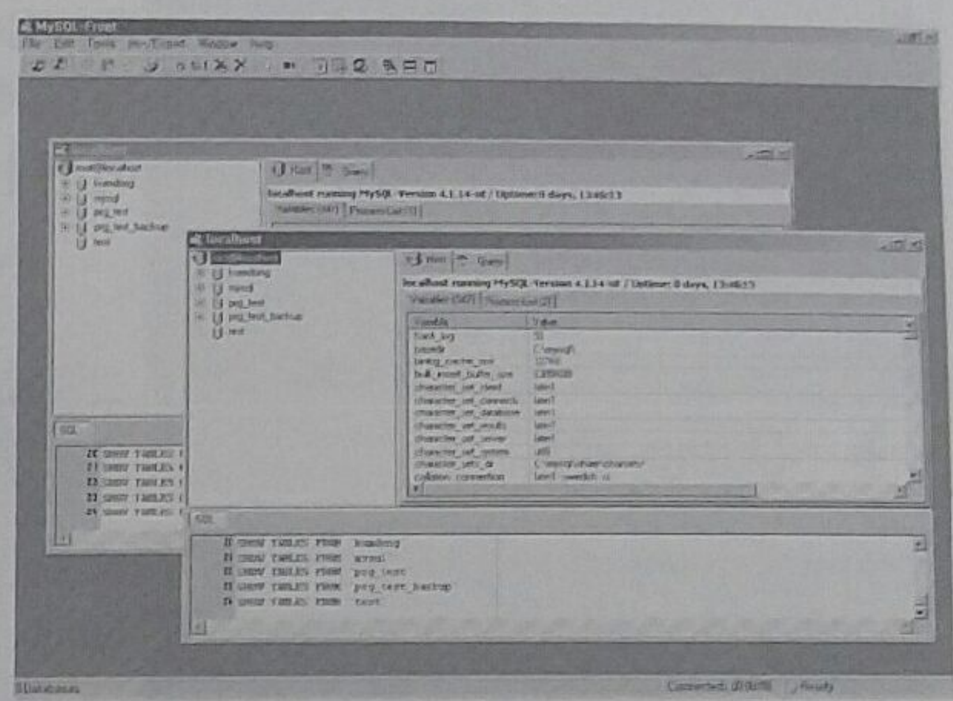
▶ 백업완료

→ Front에서 테이블의 다른 서버 데이터베이스로 백업

같은 서버 내에 백업을 받을 수 있지만 host가 다른 서버에도 백업을 받을 수 있다. 다른 서버의 데이터베이스에 백업을 받기 위해서는 File의 Connection을 통해서 옮기고자 하는 서버로 일단 접속을 해야 한다. 그런 다음에 위와 같은 동일한 맥락으로 실행하면 된다. 따라하기는 일단 동일 서버에서 새로운 접속을 하여 마치 다른 서버에 접속하는 것처럼 보이는 상태에서 이루어진다.

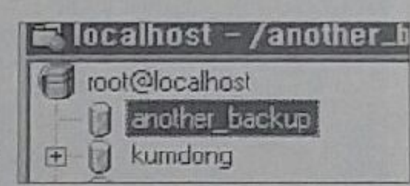
따라하기 테이블을 다른 서버로 백업

❶ 백업할 서버에 새로운 접속을 한다.



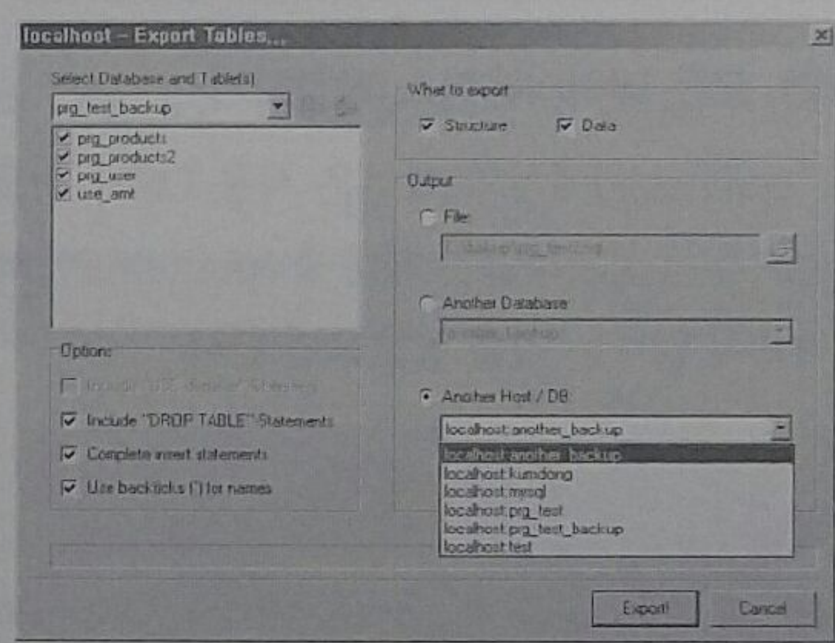
▶ 백업할 서버에 접속

❷ 백업을 받을 데이터베이스에 백업될 데이터베이스를 생성한다.



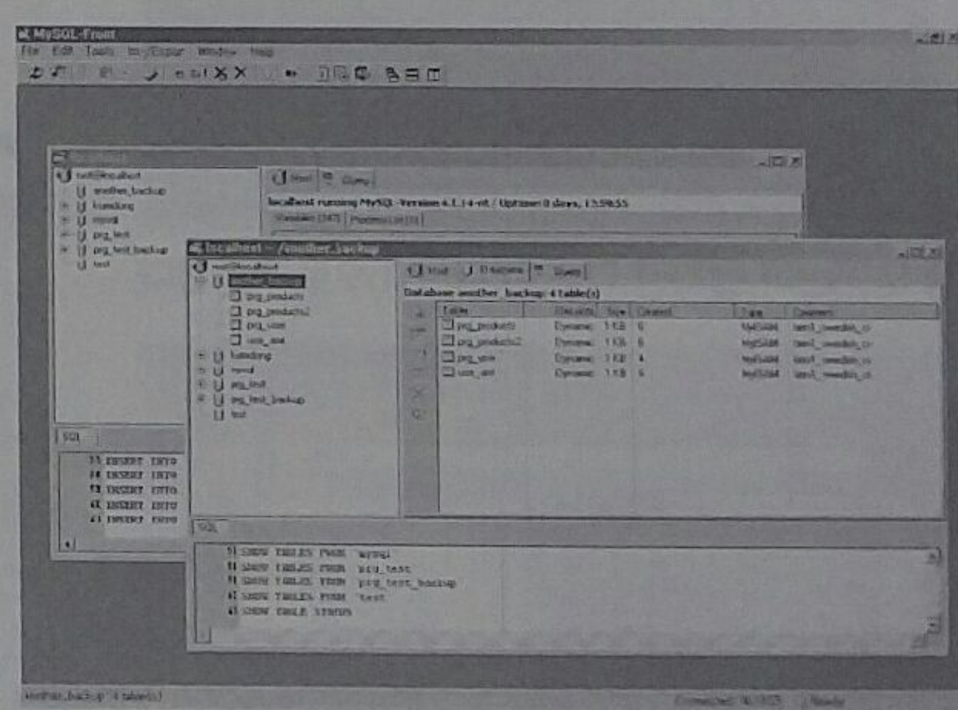
▶ 백업용 데이터베이스 생성

❸ Im/Export의 Export tables 메뉴를 선택하여 백업 데이터베이스 내 테이블을 모두 선택하고 옵션 및 구조, 데이터 모두를 선택한다.



▶ another host 선택 및 옵션선택

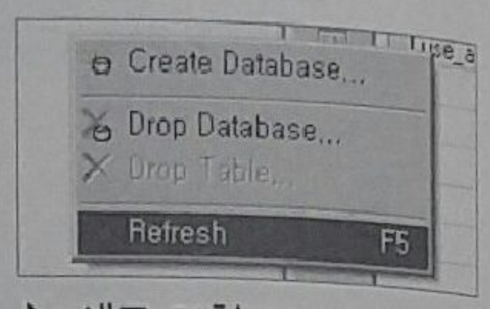
❹ Export 버튼을 클릭하면 백업이 시작된다.



▶ another host를 이용한 백업완료

따라하기 Refresh하기

Front에서 데이터베이스를 생성하거나 테이블을 생성했음에도 화면에 나타나지 않는다면 새로 고침을 해 주면 된다. 새로 고침은 마우스의 오른쪽 버튼을 클릭하여 팝업 메뉴의 Refresh를 클릭 하거나 단축키 F5 버튼을 누르면 된다.



▶ 새로 고침

Dump 백업

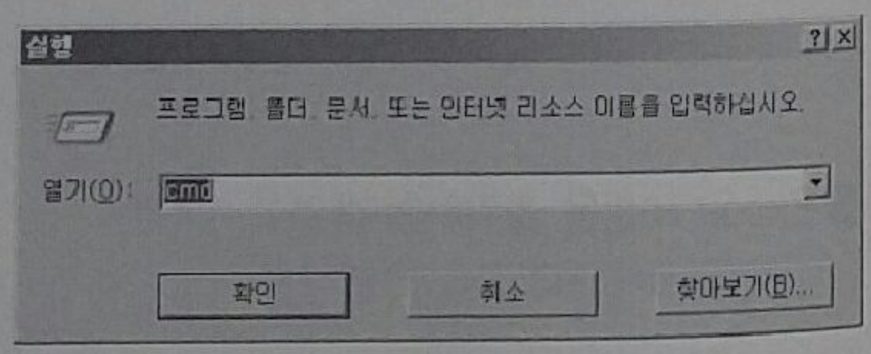
Front에서 백업에 대한 방법에 대해서 살펴보았다. 물론 Front에서 백업을 받아도 아무런 하자도 없다. 다만 관리자가 일정한 시간에 자동으로 백업을 받지 못해 불편한 점이 있다. command 창에서 또는 shell 창에서의 mysql 데이터 백업은 MySql에서 제공하는 유틸리티를 가지고 백업을 하며 자동백업은 Linux 또는 Unix에서 제공하는 cron에 shell을 만들어 정해진 시간에 백업하는 방법이 있고 이 둘 중에 cron을 더 선호한다. 허나 Front 백업에서 언급했지만 데이터의 용량이 많다면 이 방법 또한 다소의 시간이 소요됨을 상기하자.

모든 데이터베이스 백업하기

command에서의 백업은 사용 OS가 윈도우인 경우에 해당되는 부분이다. 우리가 설치한 MySql은 윈도우에 설치하였기 때문에 command에서의 백업 부분이 필요하다. 그렇다고 Linux에서 사용방법이 다르지 않다는 것을 알았으면 한다.

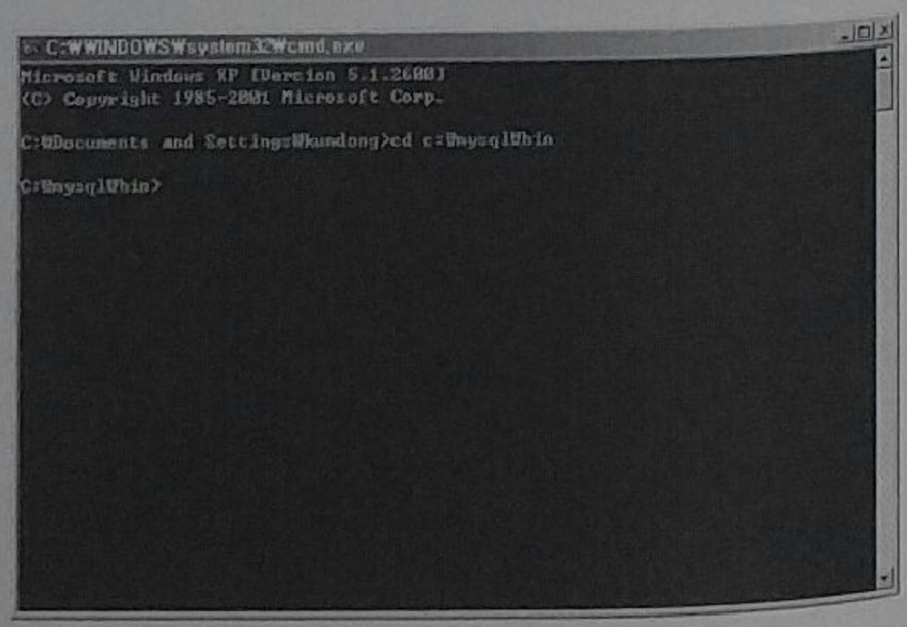
따라하기 모든 데이터베이스 백업

❶ 윈도우의 시작에서 실행을 선택하여 command 창을 띄우기 위해 cmd를 입력한다.



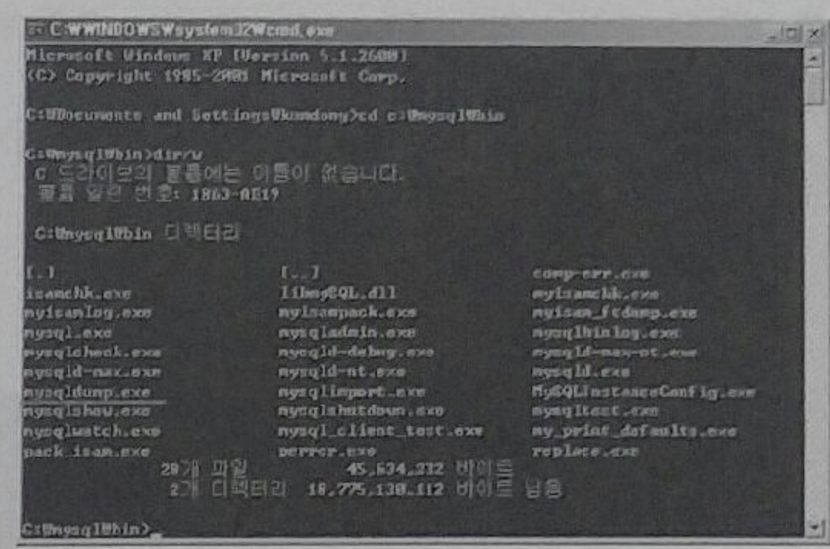
▶ cmd 입력

❷ MySql이 설치된 디렉토리의 bin으로 이동한다.



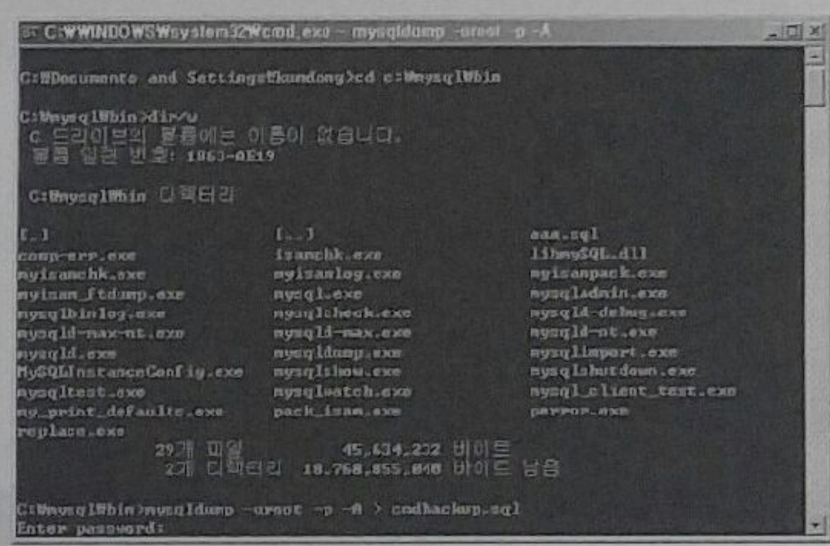
▶ 경로이동

❸ 목록확인을 통해 백업 유틸리티인 mysqldump.exe 파일이 있는지 확인해 본다.



▶ mysqldump.exe 파일 확인

❹ MySql에 존재하는 모든 테이블을 백업한다.

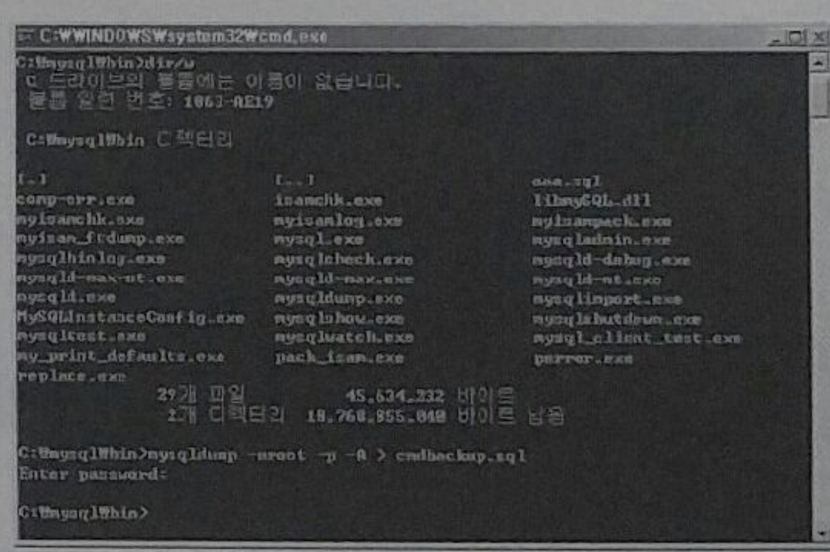


▶ 모든 테이블의 백업

다음에서 -uroot는 사용자가 root이며 -p는 암호를 입력할 것이며 -A는 모든 테이블을 의미한다. 그리고 백업이름.sql이 되는 것이다.

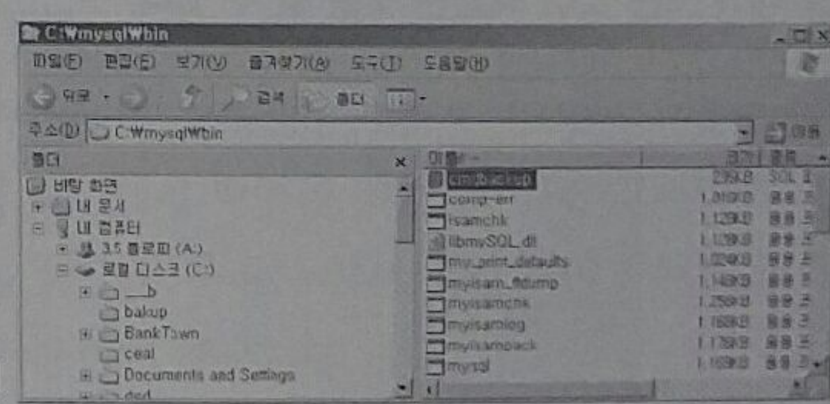
```
mysqldump -uroot -p -A > cmdbackup.sql
```

❺ 초기설정에서 암호를 지정하지 않았음으로 엔터를 치면 백업이 완료가 된 것이다.



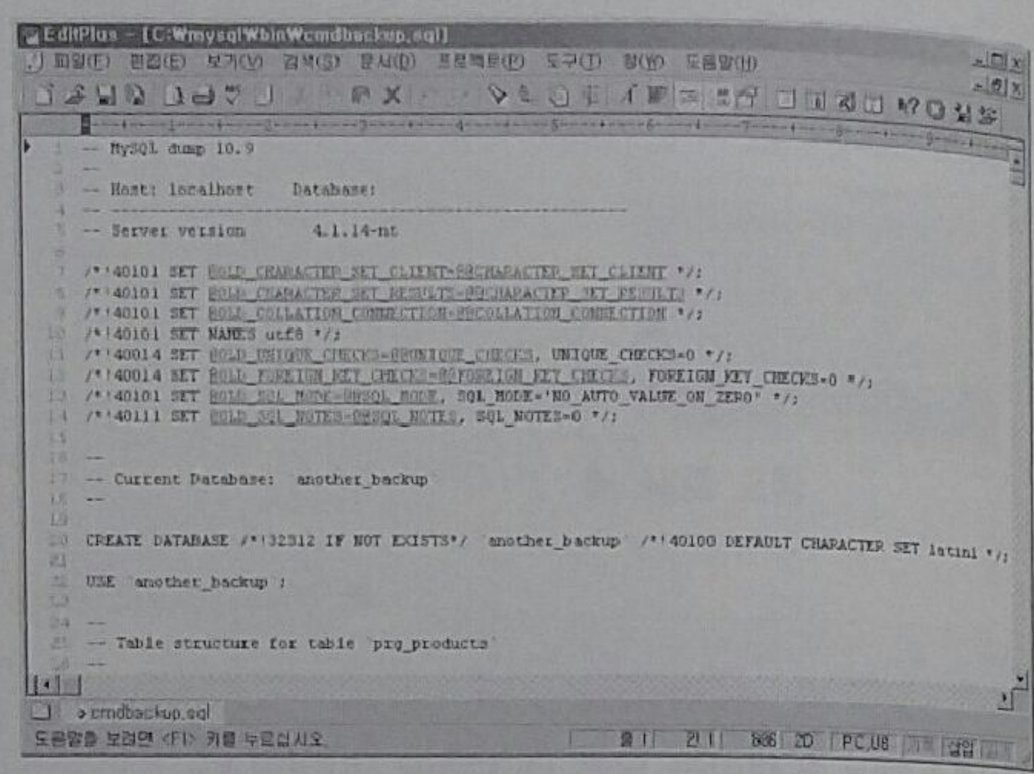
▶ 백업완료

❻ 백업 sql 생성을 확인한다.



▶ 백업파일 생성확인

❼ 백업된 파일을 열어보면 MySql에 있는 모든 테이블이 sql로 저장된 것을 볼 수 있다.



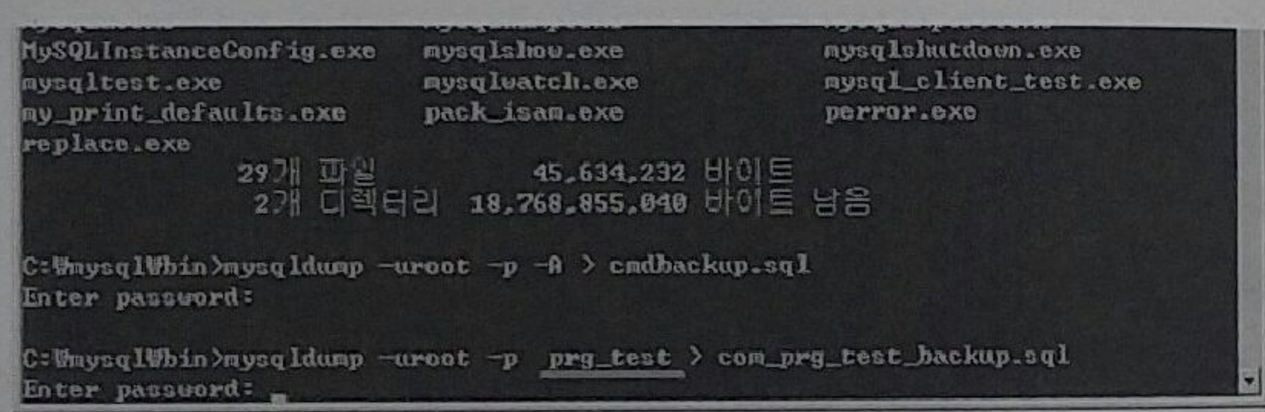
▶ sql 파일 확인

● 특정 데이터베이스 백업하기

모든 테이블을 백업하기에는 다소 부담이 된다면 아주 중요한 특정 테이블 만을 백업받으면 된다.

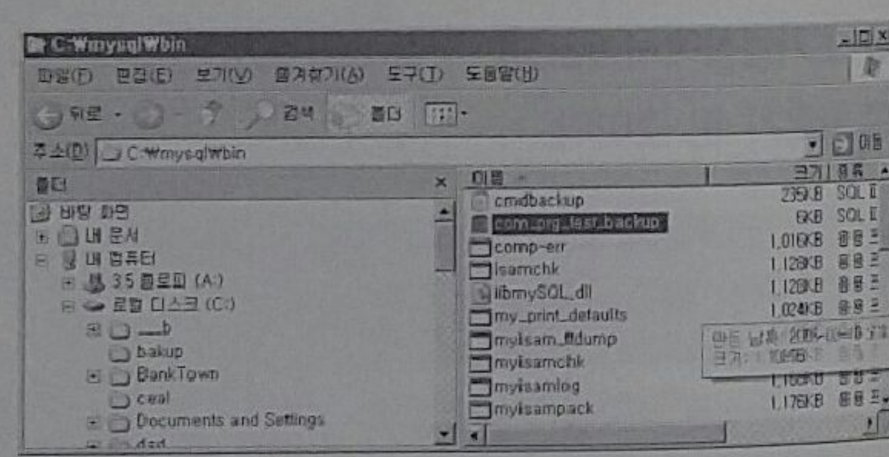
일부 테이블만 백업

❶ 열려진 command 창에서 -A 대신 백업하고자 하는 데이터베이스의 이름을 준다.



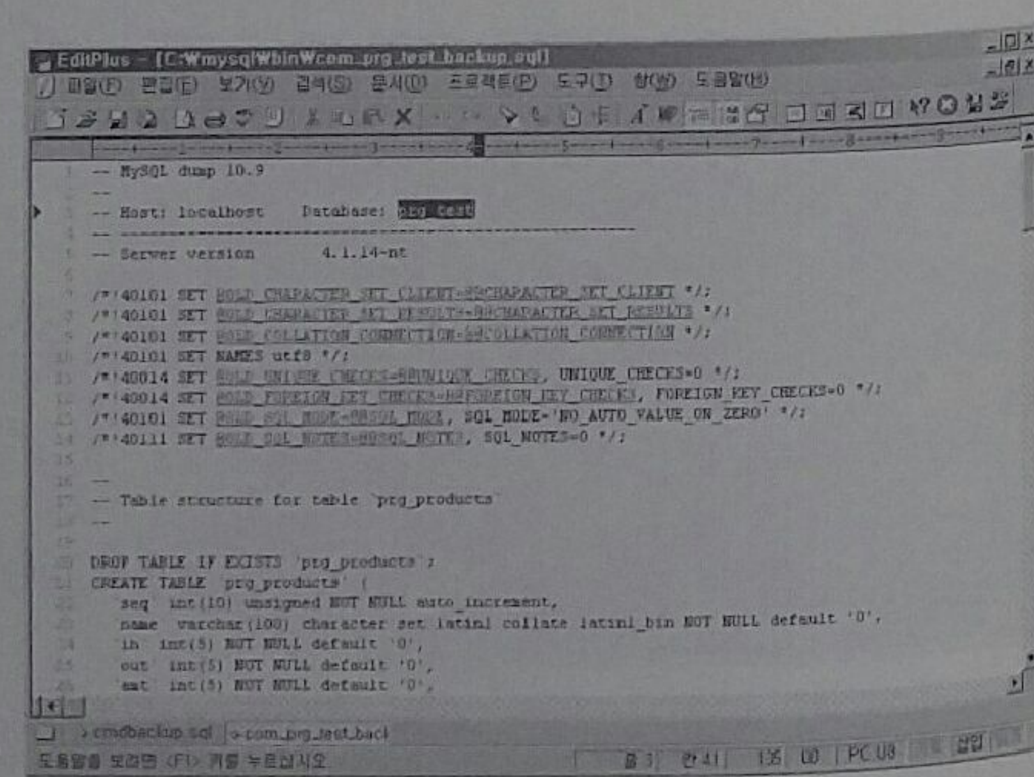
▶ 특정 데이터베이스 백업

❷ 엔터를 치면 백업이 완료되고 백업파일이 생성된 것을 알 수 있다.



▶ 백업 파일 생성

❸ 백업된 파일을 열어보면 백업하고자 하는 데이터베이스 내의 모든 것이 백업된 것을 알 수 있다.



▶ 백업 파일 확인

특정 데이터베이스의 특정 테이블 백업하기

데이터베이스 내에 백업하고자 하는 테이블 만을 백업하기 위한 방법이다.

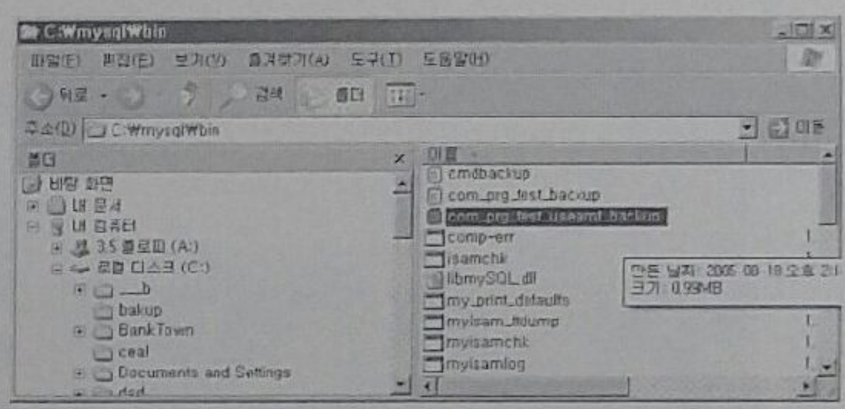
따라하기 특정 DB의 특정 테이블 백업

❶ 열려진 command 창에서 -A 대신 데이터베이스명 테이블 이름을 준다.

```
C:₩mysql₩bin>mysqldump -uroot -p -A > cmdbackup.sql
Enter password:

C:₩mysql₩bin>mysqldump -uroot -p  prg_test > com_prg_test_backup.sql
Enter password:

C:₩mysql₩bin>mysqldump -uroot -p  prg_test use_amt > com_prg_test_useamt_backup.
sql
Enter password:

C:₩mysql₩bin>
```

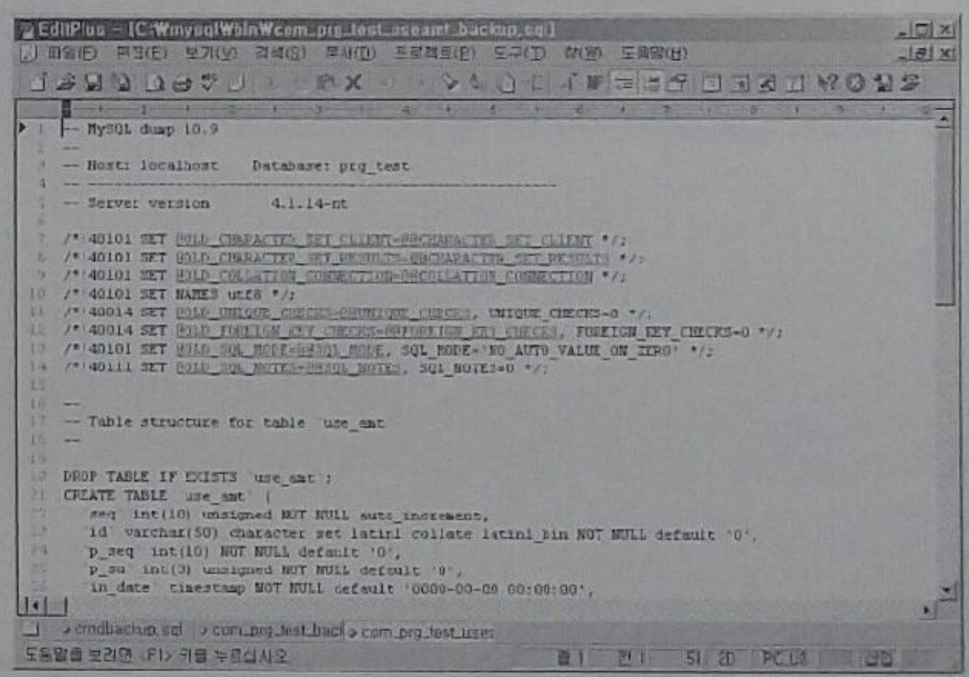
▶ 특정 테이블의 백업

❷ 엔터를 치면 백업이 완료되고 백업파일이 생성된 것을 알 수 있다.



▶ 백업 파일 생성

❸ 백업된 파일을 열어보면 백업하고자 하는 데이터베이스 내의 특정 테이블의 모든 것이 백업된 것을 알 수 있다.



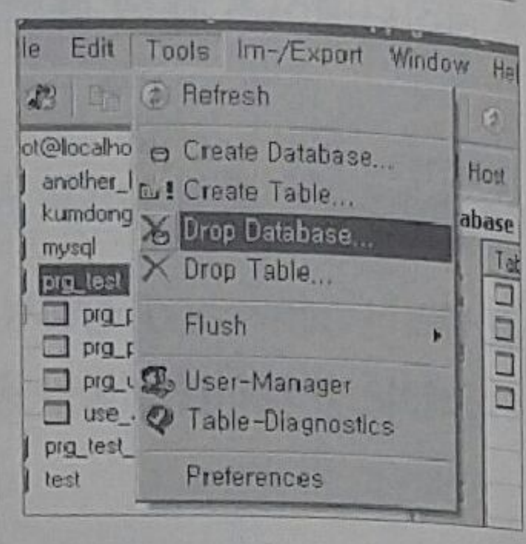
▶ 백업 파일 확인

백업 데이터 복구

데이터를 백업한다는 것이 중대한 오류 또는 기타 상황에 대비해서 백업을 하는 것이다. 그런다면 그런 상황이 되었을 때 어떻게 복구를 하는가도 알아두어야 할 것이다. 우리는 앞서 백업 시에 〉 부호를 사용한 것을 기억할 것이다. 복구는 이와 반대로 〈 의 부호를 사용하여 복구한다.

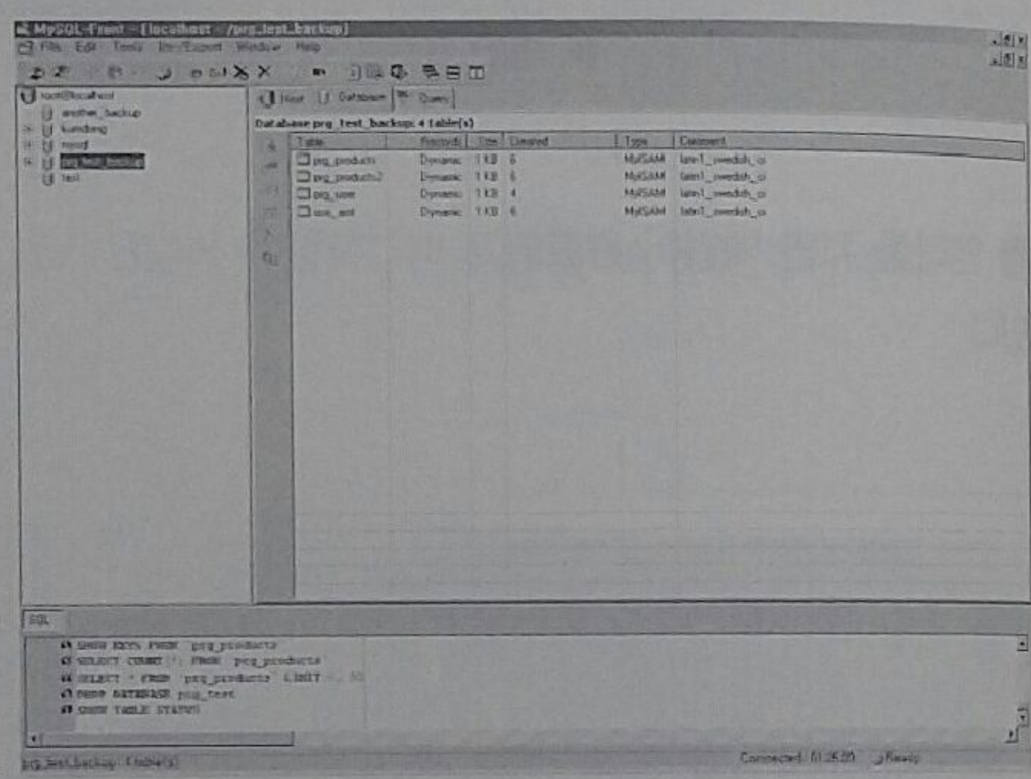
따라하기 데이터의 복구

❶ 복구확인 실습을 위해 Front에서 생성된 prg_test 데이터베이스를 삭제한다.



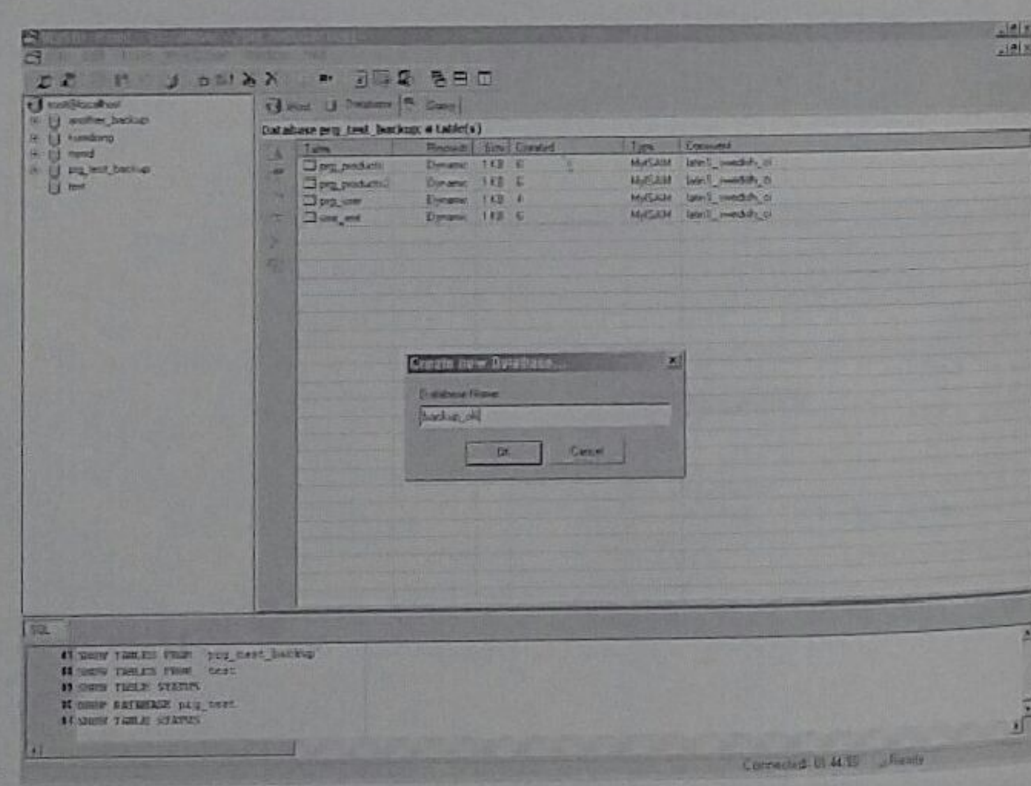
▶ 데이터베이스 삭제

❷ 데이터베이스 삭제를 확인한다.



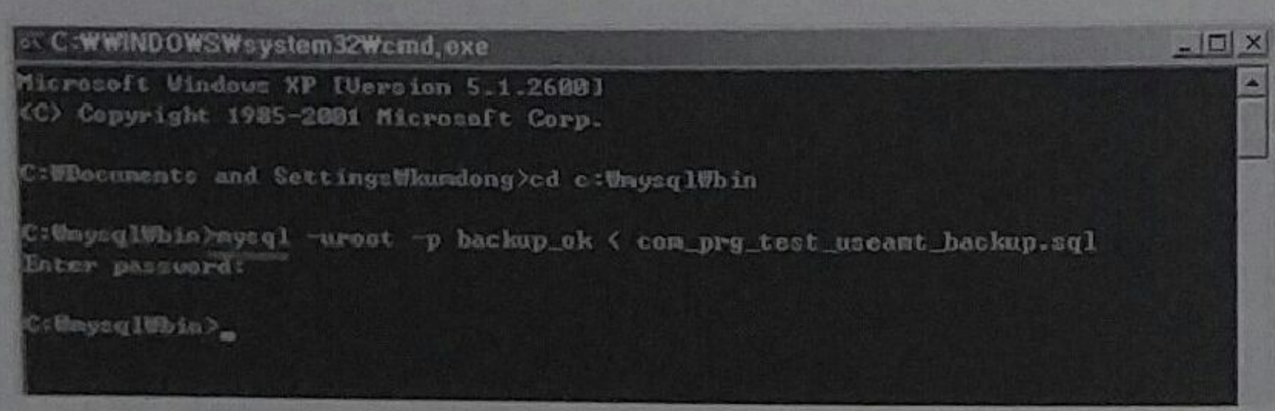
▶ 삭제 확인

❸ 복구 확인을 위해 새로운 데이터베이스를 만든다.



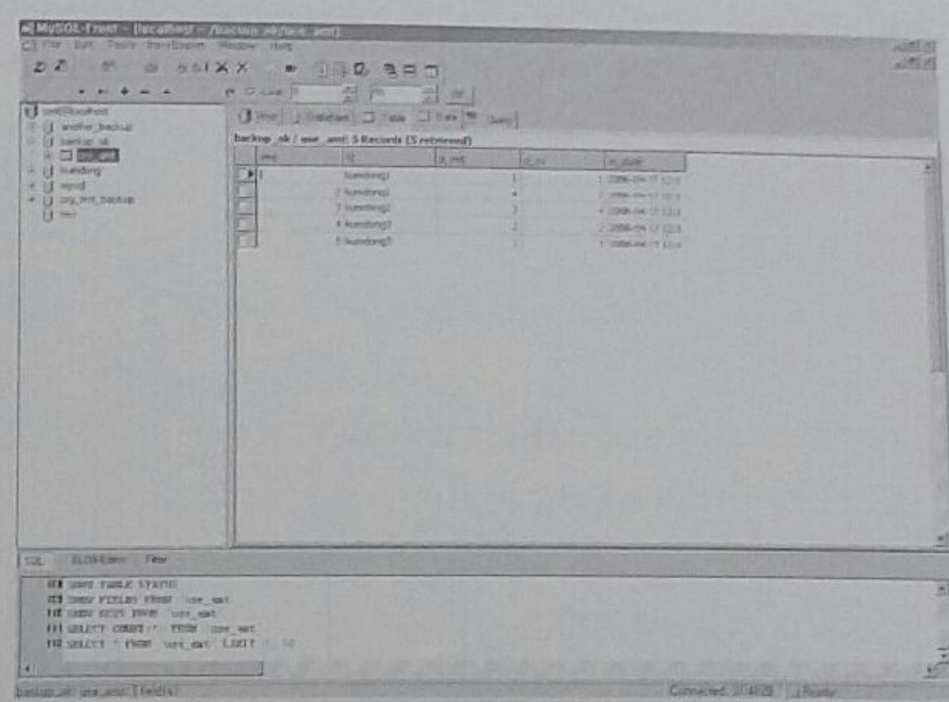
▶ 복구확인 데이터베이스

❹ command 창에서 백업된 com_prg_test_backup.sql 파일로 복구한다.



▶ 데이터 복구

❺ Front에서 테이블이 복구가 되었는지를 확인한다.



▶ 테이블의 복구확인

백업 및 복구

1. 윈도우의 Command에서나 Linux에서 백업 및 복구는 같은 유틸리티를 사용함으로 동일한 명령어로 백업 및 복구가 이루어진다.
2. 복구 시 복구하려는 테이블이 위치할 데이터베이스가 항상 존재하여야 한다. 즉 데이터베이스 전체가 복구가 되는 것이 아니라 데이터베이스 내에 존재하는 테이블이 복구되기 때문에 복구할 테이블이 존재하게 될 데이터베이스가 필요한 것이다.

```
C:₩WINDOWS₩system32₩cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:₩Documents and Settings₩kumdong>cd c:₩mysql₩bin

C:₩mysql₩bin>mysql -uroot -p backup_ok < com_prg_test_useamt_backup.sql
Enter password:

C:₩mysql₩bin>mysql -uroot -p < com_prg_test_backup.sql
Enter password:
ERROR 1046 (3D000) at line 20: No database selected

C:₩mysql₩bin>
```

▶ 에러

3. Front에서의 복구는 백업된 부분을 host Export 방법을 사용하여 Export 시키면 된다.
4. 백업은 mysqldump를 사용하고 복구는 mysql로 사용하는 것을 잊지 말아야 한다.

자동 백업

mysql의 자동백업은 sh을 작성할 수 있는 OS에서만 가능하다. 거의 대부분 MySql은 Linux에 설치하는데 Linux에서 shell을 작성하여 cron에 등록하여 일정시간에 만들어진 shell 프로그램을 구동하게 한다.

shell의 작성

Linux에서 vi 편집기를 이용하여 다음과 같이 shell 프로그램을 작성한다.

```
#vi mysql_dump_backup.sh

#mysql 백업
#날짜지정
DATE =' /bin/date +%d-%b-%Y-%H-%M-%S
#파일명
FILEPREFIX="Data_backup=$DATE"
TEMPDIR="/backupdata";

#root 그리고 root의 암호입력
USERNAME="root"
PASSWORD="kumdong"
DATABASE="prg_test"

#백업 성공시에 메일전송
RECIPIENT="kumdong@dreamwiz.com"
NICEDATE=' /bin/date +%b-%d-%Y'
SUBJECT="백업이 성공적으로 이루어 졌습니다. -$NICEDARE"
/usr/local/mysql/bin/mysqldump -u$USERNAME -p$PASSWORD
>$TEMPDIR/$FILEPREFIX.sql | tar -cf $TEMPDIR/$FILEPREFIX.tar.gz-z
$TEMPDIR/$FILEPREFIX.sql | rm -f $TEMPDIR/$FILEPREFIX.sql
:wq!
```

shell 파일이 작성이 되었다면 이 파일에 대한 사용 권한을 주면 된다.

```
#chmod 711  mysql_dump_backup.sh
```

그런 후에 cron에 작성된 sh 파일을 새벽 03시30분에 매일 실행하라고 등록한다.

```
#crontab -e
00 03 * * * /usr/local/mysql/bin/ mysql_dump_backup.sh
wq!
```

이렇게 되면 일정한 시간에 prg_test라는 데이터베이스 내의 모든 테이블은 자동백업이 된다.

맺음말

이번 장에서는 MySql에서 제공되는 함수와 그리고 데이터 백업과 복구에 대해서 살펴보았다. 여기까지 무난히 따라온 독자는 이제 MySql의 초급에서 벗어날 수 있다고 할 수 있다. 다음 장에서는 마지막으로 Front에서의 기타 여러 가지 기능에 대해서 공부한다.