02 SQL 언어

- SQL 명령문의 종류와 사용 방법
- 오라클 오류 메시지

SQL 언어의 정의

❑ SQL 언어의 정의

- ✓ 관계형 데이터베이스를 조작하기 위한 표준 언어
- ✓ 관계 대수와 관계 해석의 수학적인 이론을 기초로 개발
- ✓ 데이터베이스의 구조를 정의하거나 데이터베이스에 저장된 데이터를 검색하기 위한 목적

❑ SQL 언어의 특징

- ✓ 비절차적 언어
  - 기존 프로그래밍 언어 : 레코드 단위로 처리 조건에 따라 데이터에 대한 접근 경로가 달라짐
  - SQL 언어 : 조건을 만족하는 데이터를 집합 단위로 한꺼번에 처리
- ✓ 대화식으로 사용되거나 응용 프로그램에 삽입하여 사용

SQL 명령문의 종류와 사용방법

❑ SQL 명령문의 종류

- ✓ 질의어 (DQL : Data Query Language)
  - 데이터베이스에 저장된 데이터를 조회하는 명령어
- ✓ 데이터 조작어 (DML : Data Manipulation Language)
  - 데이터베이스 응용 프로그램 개발을 위해 주로 사용되며 데이터를 입력, 수정 삭제하는 명령문 포함
- ✓ 데이터 정의어 (DDL : Data Definition Language)
  - 데이터베이스 설계자가 데이터베이스의 구조를 정의하거나 수정하기 위한 목적으로 사용하며 데이터베이스 객체를 생성, 수정, 삭제하는 명령문 포함
- ✓ 트랜잭션 처리어 (TCL : Transaction Control Language)
  - 트랜잭션 관리를 위한 목적으로 사용
- ✓ 데이터 제어어 (DCL : Data Control Language)
  - 데이터베이스 관리자가 데이터나 데이터베이스 객체에 대한 접근 권한 부여와 같은 데이터베이스 시스템을 관리하기 위한 목적으로 사용

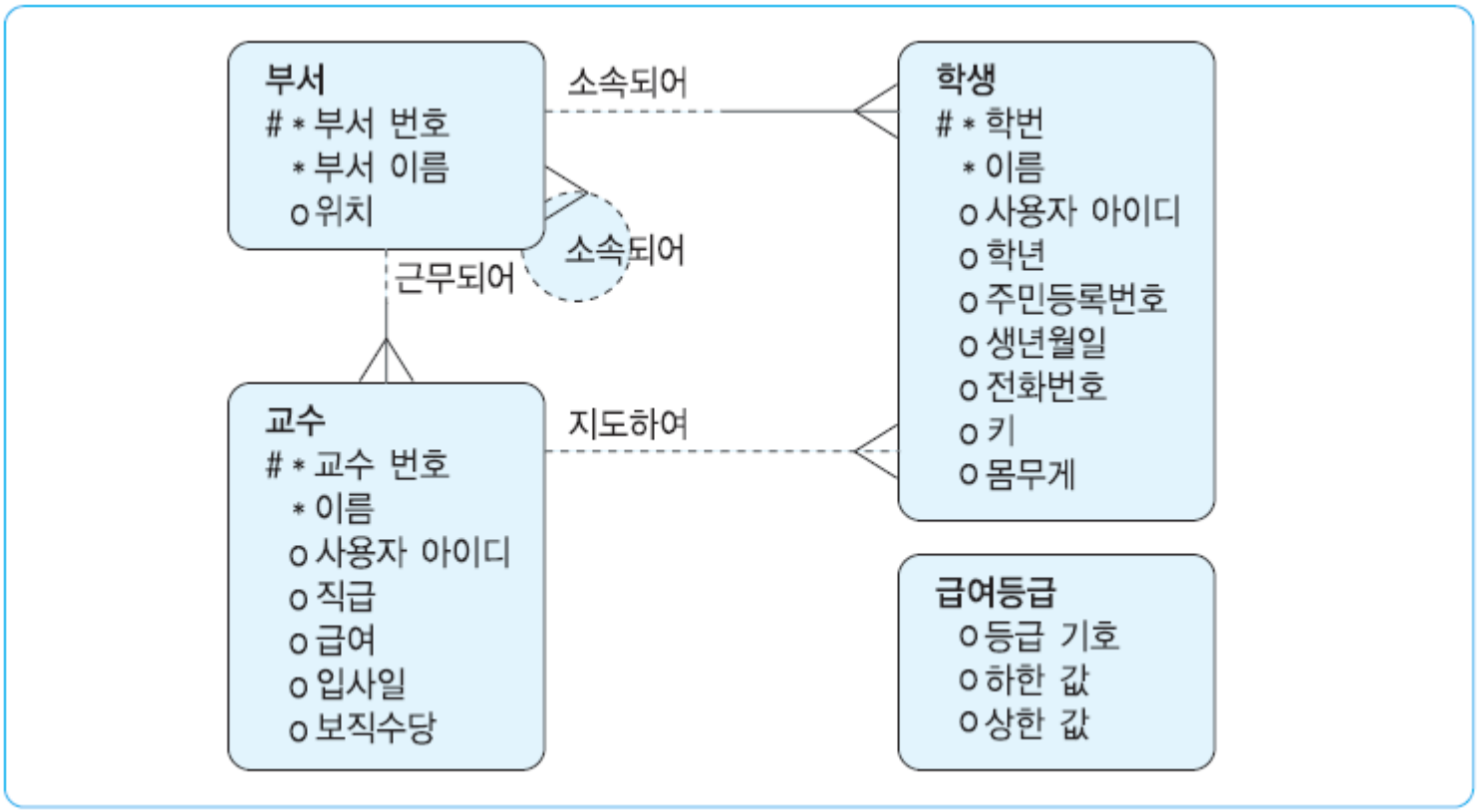
SQL 명령문의 종류

〈표 1-1〉 SQL 명령문의 종류

유형	명령문	기능
질의어 (DQL : Data Query Language)	SELECT	데이터 검색
데이터 조작어 (DML : Data Manipulation Language)	INSERT UPDATE DELETE	데이터 입력 데이터 수정 데이터 삭제
데이터 정의어 (DDL : Data Definition Language)	CREATE ALTER DROP RENAME TRUNCATE	데이터베이스 객체 생성 데이터베이스 객체 변경 데이터베이스 객체 삭제 데이터베이스 객체 이름 변경 데이터 및 저장 공간 삭제
트랜잭션 처리 (TCL : Transaction Control Language)	COMMIT ROLLBACK SAVEPOINT	트랜잭션의 정상적인 종료 처리 트랜잭션 취소 트랜잭션내에 임시 저장점 설정
데이터 제어어 (DCL : Data Control Language)	GRANT REVOKE	데이터베이스 객체에 대한 접근 권한 부여 데이터베이스 객체에 대한 접근 권한 취소

실습 예제 E/R 다이어그램

❑ 실습 예제 데이터베이스의 E/R 다이어그램



테이블 인스턴스

❑ 테이블 인스턴스(table instance)

- ✓ 데이터베이스 설계시에 테이블의 구조와 칼럼의 특성을 알기 쉽게 요약한 내용
- ✓ 테이블 인스턴스는 테이블의 칼럼 이름, 데이터 타입, 키 종류, NULL이나 중복 값의 허용 여부, 외래 키 그리고 칼럼에 대한 설명으로 구성

학생 테이블 인스턴스 & 예제 데이타

칼럼 이름	데이터 타입	Key Type	NN/ Unique	FK table	FK column	설명
STUDNO	NUMBER(5)	PK	NN, U			학번
NAME	VARCHAR2(10)		NN			이름
USERID	VARCHAR2(10)					사용자 아이디
GRADE	VARCHAR2(1)					학년
IDNUM	VARCHAR2(13)					주민등록번호
BIRTHDATE	DATE					생년월일
TEL	VARCHAR2(13)					전화번호
HEIGHT	NUMBER(5, 2)					키
WEIGHT	NUMBER(5, 2)					몸무게
DEPTNO	NUMBER(4)	FK	NN	DEPARTMENT	DEPTNO	학과 번호
PROFNO	NUMBER(4)					지도 교수 번호

```
STUD NAME   USERID G  IDNUM          BIRTH    TEL             HE  WE DNO PNO
---- ------ ------ -- ------------ ------- ----------- --- --- --- ---
10101 전인하  jun123  4 7907021369824 79/07/02 051)781-2158 176 72  101 993
20101 이동훈  Dals    1 8312101128467 83/12/10 051)426-1752 172 64  201
```

교수 테이블 인스턴스 & 예제 데이타

칼럼 이름	데이터 타입	Key Type	NN/ Unique	FK table	FK column	설명
PROFNO	NUMBER(4)	PK	NN, U			교수 번호
NAME	VARCHAR2(10)		NN			이름
USERID	VARCHAR2(10)					사용자 아이디
POSITION	VARCHAR2(20)					직급
SAL	NUMBER(10)					급여
HIREDATE	DATE					입사일
COMM	NUMBER(2)					보직수당
DEPTNO	NUMBER(4)	FK	NN	DEPARTMENT	DEPTNO	학과 번호

```
PROFNO  NAME   USERID   POSITION SAL HIREDATE  COMM DEPTNO
------- ------ -------- -------- --- --------- ----- -------
9901    김도훈  capool    교수     500 24-JUN-82     20    101
9902    이재우  sweat413 조교수    320 12-APR-95           201
```

부서 등급 테이블 인스턴스 & 예제 데이타

칼럼 이름	데이터 타입	Key Type	NN/ Unique	FK table	FK column	설명
DEPTNO	NUMBER(4)	PK	NN, U			부서 번호
DNAME	VARCHAR2(16)		NN			부서 이름
COLLEGE	NUMBER(4)					상위 부서 번호
LOC	VARCHAR2(10)					위치

```
DEPTNO DNAME          COLLEGE  LOC
------- ------------ -------- ---------
     101  컴퓨터공학과          100  1호관
     102  멀티미디어학과        100  2호관
```

급여 등급 테이블 인스턴스 & 예제 데이타

칼럼 이름	데이터 타입	Key Type	NN/ Unique	FK table	FK column	설명
GRADE	NUMBER(2)					등급 기호
LOSAL	NUMBER(5)					하한 값
HISAL	NUMBER(5)					상한 값

```
GRADE  LOSAL  HISAL
-----  -------  -----
    1      100    300
    2      301    400
```

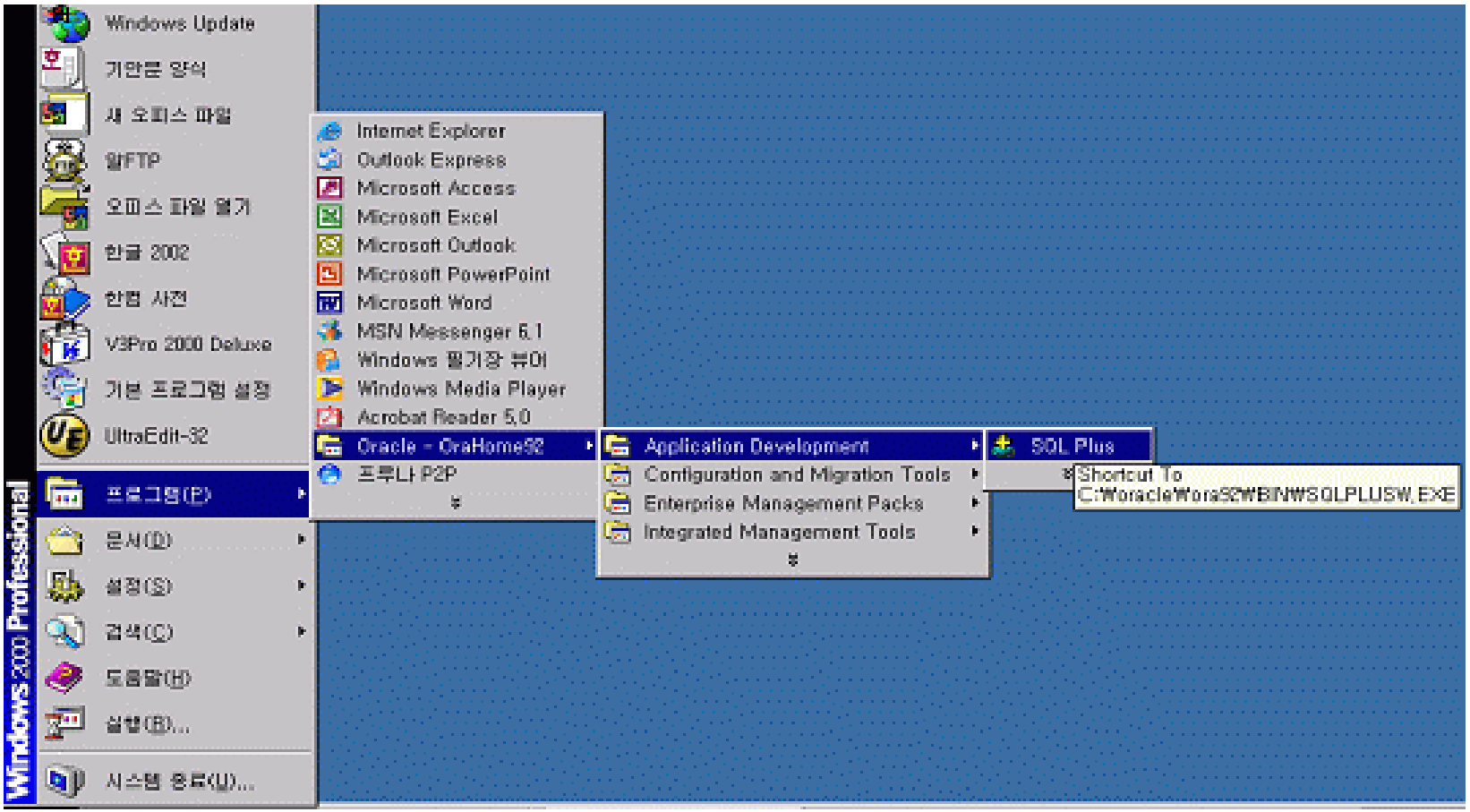
SQL*Plus의 기본 사용자 계정

❑ 기본 사용자 계정

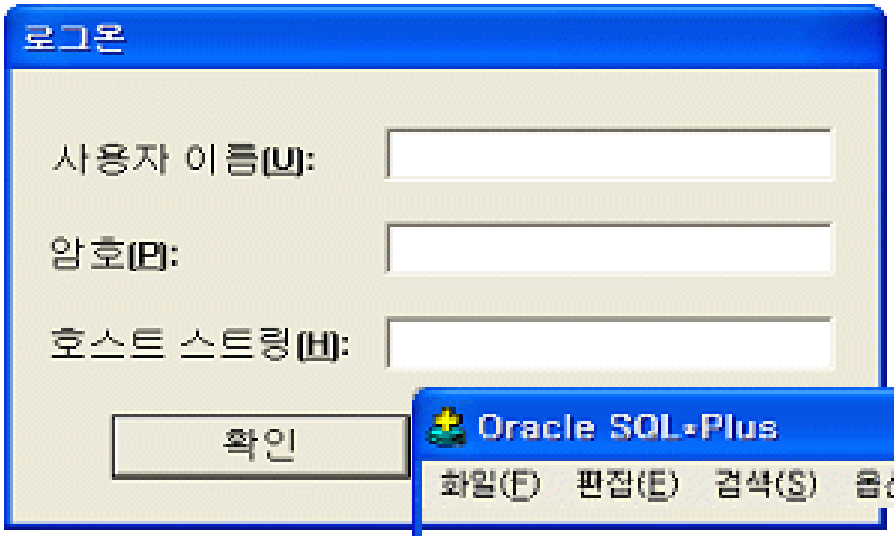
- ✓ 오라클을 인스톨하면 기본적으로 SYS, SYSTEM, SCOTT라는 3 종류의 사용자 계정이 생성
- ✓ sys/change_on_install : 데이터베이스 관리자 권한, 모든 객체의 소유주
- ✓ system/manager : 데이터베이스 관리자 권한
- ✓ scott/tiger : 일반 사용자 권한

SQL*Plus의 실행

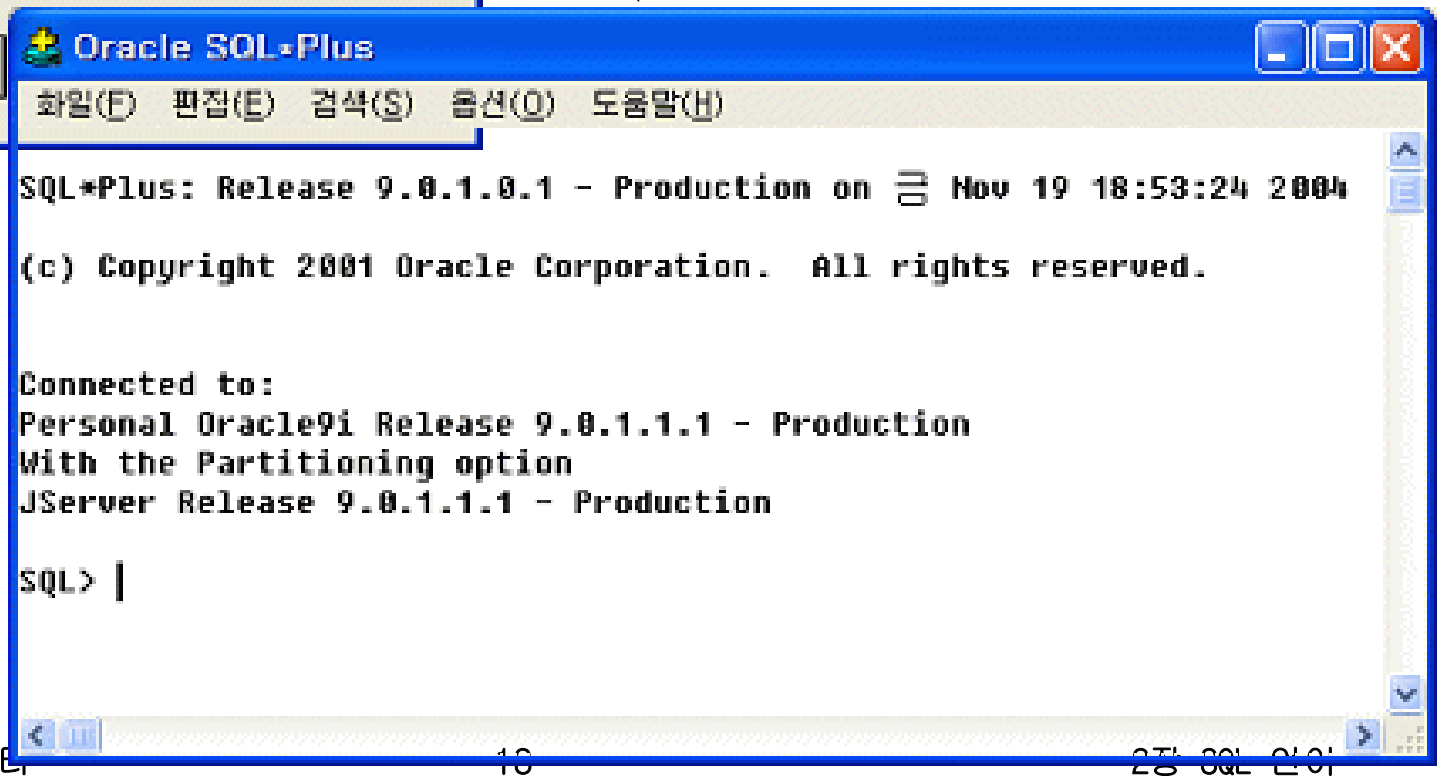
❑ 윈도우 환경에서 SQL*Plus 실행



PC에서의 SQL*Plus 로그인



- sys/change_on_install
- system/manager
- scott/tiger



UNIX/LINUX에서의 SQL*Plus 로그인

❑ SQL*Plus 실행 방법

✓ sqlplus username/password 형식으로 접속 또는 sqlplus 를 먼저 실행한 후, 사용자 이름과 암호 입력 가능

사용법

```
$ sqlplus [ username[ /password[ @database]]]
```

사용예

```
$ sqlplus
SQL*Plus: Release 9.2.0.1.0 - Production on Fri Jan 3 10:50:36 2003
(c) Copyright 2000 Oracle Corporation.  All rights reserved.

Enter user-name: scott
Enter password:

Connected to:
Oracle9i Enterprise Edition Release 9.2.0.1.0 - 64bit Production
With the Partitioning option
JServer Release 9.2.0.1.0 - 64bit Production
SQL>
```

사용자 이름은 화면에 표시되지만
암호는 화면에 표시되지 않는다.

UNIX/LINUX에서의 SQL*Plus 로그인

사용예

```
$ sqlplus scott/tiger
SQL*Plus: Release 9.2.0.1.0 - Production on Fri Jan 3 10:51:12 2003
(c) Copyright 2000 Oracle Corporation.  All rights reserved.
Connected to:
Oracle9i Enterprise Edition Release 9.2.0.1.0 - 64bit Production
With the Partitioning option

JServer Release 9.2.0.1.0 - 64bit Production
SQL>
```

❑ SQL*Plus 실행 종료 방법

✓ SQL> 프롬프트상에서 “exit” 명령을 입력

초기 데이타베이스 생성 방법

❑ 본 교재의 실습용 테이블과 데이터 생성 방법

1. 먼저, http://www.dbcore.net의 SQL 강좌 메뉴에서 실습용 테이블과 데이터를 저장한 table_exam.sql 스크립트를 다운받는다.
2. 앞에서 설명한 방법에 의해 SQL*Plus를 실행한다.
3. SCOTT 사용자로 접속한다. 암호는 TIGER를 입력한다.
4. 실습용 스크립트를 다운받은 경로가 d:₩down일 경우 다음 명령어에 의해 다운받은 table_exam1.sql을 실행한다.
   SQL>@d:₩down₩table_exam.sql
5. 생성된 테이블을 확인한다.
   SQL> SELECT * FROM tab;
6. 테이블에 저장된 데이터를 확인한다.
   SQL> SELECT * FROM student;
   SQL> SELECT * FROM professor;
   SQL> SELECT * FROM department;
   SQL> SELECT * FROM salgrade;

데이터 딕셔너리를 이용한 테이블 조회

❑ 데이터 딕셔너리 (Data dictionary)

✓ 데이타베이스 객체에 대한 다양한 시스템 정보를 저장하고 있는 테이블의 집합

❑ 테이블 이름 확인

✓ 사용자 데이타베이스 계정에 생성된 테이블 이름을 확인하기 위하여 tab 이라는 데이터 딕셔너리 테이블을 조회

사용예

현재 접속한 데이터베이스 계정에 생성된 모든 테이블 이름을 확인하여라.

```
SQL> SELECT * FROM tab;

☞ 출력 결과
TNAME                TABTYPE            CLUSTERID
-------------------- ------------------ ----------
DEPARTMENT           TABLE
PROFESSOR            TABLE
SALGRADE             TABLE
STUDENT              TABLE
```

실습용 스크립트(table_exam.sql)을 실행하면 생성된다.

데이터 딕셔너리를 이용한 테이블 조회

❑ 테이블의 구조 확인

- ✓ 데이터베이스의 모든 데이터는 테이블이라는 논리적인 구조에 저장
- ✓ 사용자가 테이블의 데이터 조회시 칼럼 이름을 잘못 적을 경우, 'ORA-000904 : invalid identifier' 오류 발생
- ✓ 테이블의 구조 정보를 조회하여 정확한 칼럼 이름 확인 필요
- ✓ DESC[RIBE] 명령어
  - 테이블의 칼럼 이름, 데이터 타입과 길이, NULL 허용 유무 등과 같은 테이블 구조 정보 확인 가능

사용법

```
SQL> DESC[ ribe]  table_name
```

학생(Student) 테이블 구조 확인

사용예

학생(student) 테이블의 구조를 확인하여라.

```
SQL> DESC student

이름                                      널?       유형
----------------------------------------- -------- ------------
STUDNO                                    NOT NULL NUMBER(5)
NAME                                      NOT NULL VARCHAR2(10)
USERID                                             VARCHAR2(10)
GRADE                                              VARCHAR2(1)
IDNUM                                              VARCHAR2(13)
BIRTHDATE                                          DATE
TEL                                                VARCHAR2(13)
HEIGHT                                             NUMBER(5,2)
WEIGHT                                             NUMBER(5,2)
DEPTNO                                             NUMBER(4)
PROFNO                                             NUMBER(4)
```

SQL 명령문의 기초

❑ SELECT 명령문

- ✓ 테이블에 저장된 데이터를 검색하기 위한 명령문
- ✓ SQL 명령문 중에서 가장 자주 사용하는 명령문
- ✓ SELECT 명령문은 일반 프로그래밍 언어에서 수 백줄로 표현되는 알고리즘을 몇 줄로 표현할 수 있는 강력한 기능 제공
- ✓ SELECT 명령문에서 SELECT절과 FROM절은 필수절
- ✓ SELECT …… FROM 테이블명
  - FROM 절에서 지정한 테이블에서 SELECT 절에서 나열한 칼럼에 저장된 데이터를 검색하라는 의미

사용법

```
SELECT      *|columnlist
FROM        table;
```

테이블의 모든 데이터를 검색

❑ 모든 칼럼에 저장된 모든 데이터 검색

✓ 검색 방법 1

- SELECT 명령문의 FROM 절에 검색을 원하는 테이블 이름 작성
- SELECT 절에 FROM 절에 기술한 테이블의 모든 칼럼 이름 나열

✓ 검색 방법 2

- SELECT 절에 모든 칼럼 이름을 일일이 나열하는 과정이 번거로우므로 와일드 문자인 '*' 를 사용하여 모든 칼럼의 데이터 검색 가능
- 출력순서는 테이블 생성시 정의한 칼럼의 순서와 동일

사용예

부서 테이블의 모든 칼럼에 저장된 데이터를 출력하여라.

```
SQL> SELECT deptno, dname, college, loc
  2  FROM   department;

또는

SQL> SELECT * FROM department;
```

두 SELECT 명령문의 출력 결과는 동일하다.

테이블의 특정 칼럼을 선택

❑ 특정 칼럼의 선택적 출력방법

✓ SELECT 절 다음에 출력을 원하는 칼럼 이름을 순서대로 나열

✓ 칼럼을 구분하기 위해 칼럼 이름과 칼럼 이름 사이에 콤마(,) 기술

사용예

부서 테이블에서 부서 이름(dname)과 부서 번호(deptno)를 출력하여라.

```
SQL> SELECT dname, deptno
  2 FROM   department;
```

☞ 출력 결과

DNAME	DEPTNO
컴퓨터공학과	101
멀티미디어학과	102
전자공학과	201
기계공학과	202
정보미디어학부	100
메카트로닉스학부	200
공과대학	10

SELECT 절에서 나열한 칼럼 이름 순서대로 출력된다.

SQL 명령문의 표준 형식

❑ 표준 형식

- ✓ 데이터 조회를 위해서는 SELECT 절과 FROM 절이 반드시 존재해야 함
- ✓ WHERE, GROUP BY, HAVING, ORDER BY 절은 필요에 따라 선택적으로 사용 가능

➪SQL 명령문의 표준 형식

```
SELECT      [ DISTINCT] {*|column[ alias] ....}
FROM        table
[ WHERE     condition]
[ GROUP BY  group_by_expression]
[ HAVING    group_condition]
[ ORDER BY  column]
```

- ✓ 대괄호로 표시된 형식은 생략 가능

SQL 명령문의 표준 형식

❑ SELECT 명령문 작성시 규칙

- ✓ SQL 명령문에서 대소문자를 구분하지 않음
  - 키워드는 주로 대문자로 사용하고 테이블명, 칼럼 이름은 소문자로 작성하는 것을 권장
- ✓ 가능하면 절은 줄을 구분하여 작성하고, 들여쓰기를 사용하여 읽기 쉽게 작성할 것을 권장
- ✓ 테이블명, 칼럼 이름, 키워드(SELECT, FROM, WHERE 등)는 축약할 수 없음

중복행 출력 방지

❑ 중복행 제거

✓ SELECT 절에 DISTINCT 키워드를 사용하여 중복행 제거 가능

단일 칼럼에서 DISTINCT 키워드 사용예

학생 테이블에서 중복되는 학과 번호(deptno)를 제외하고 출력하여라.

```
SQL> SELECT deptno
  2 FROM   student;

☞ 출력 결과
    DEPTNO
----------
       101
       201
       101
       101
       201
       102
       101
       102
       102
       101
       101
       102
       101
       201
       201
       101

16 개의 행이 선택되었습니다.
```

DISTINCT 기능을 사용하지 않으면 테이블에 저장된 모든 행 수만큼 출력된다.

```
SQL> SELECT DISTINCT deptno
  2 FROM   student;

☞ 출력 결과
    DEPTNO
----------
       101
       102
       201

3 개의 행이 선택되었습니다.
```

DISTINCT 기능을 사용하면 중복되는 deptno를 제외하고 3건만 출력된다.

중복행 출력 방지

복수 칼럼에서 DISTINCT 키워드 사용예

학생 테이블에서 중복되는 행을 제외한 학과 번호와 학년을 출력하여라.

```
SQL> SELECT deptno, grade
  2  FROM   student;

☞ 출력 결과
    DEPTNO GRADE
---------- --------
       101 4
       201 1
       101 1
       101 3
       201 1
       102 2
       101 2
       102 4
       102 1
       101 2
       101 1
       102 3
       101 4
       201 2
       201 1
       101 2

16 개의 행이 선택되었습니다.
```

```
SQL> SELECT DISTINCT deptno, grade
  2  FROM   student;

☞ 출력 결과
    DEPTNO GRADE
---------- --------
       101 1
       101 2
       101 3
       101 4
       102 1
       102 2
       102 3
       102 4
       201 1
       201 2

10 개의 행이 선택되었습니다.
```

(deptno, grade)를 쌍으로 하여 중복되는 행을 제외하고 출력된다.

칼럼에 대한 별명 부여

❑ 별명을 부여하는 일반적인 경우

- ✓ 칼럼 이름이 너무 길어서 한 화면에 칼럼 이름 전체를 출력하기 어렵운 경우
- ✓ SQL 함수나 산술 연산에 의해 일시적으로 만들어지는 가상 칼럼이 생기는 경우

❑ 별명 부여 방법

- ✓ 칼럼 이름과 별명 사이에 공백을 추가하는 방법
- ✓ 칼럼 이름과 별명 사이에 AS 키워드를 추가하는 방법
- ✓ 큰따옴표(“”)를 사용하는 방법
  - 칼럼 이름과 별명 사이에 공백을 추가하고 큰따옴표를 사용
  - 별명에 공백, 특수문자, 대소문자가 포함된 경우 주로 사용

칼럼 별명 부여 예

사용예

부서 테이블에서 부서 이름 칼럼의 별명은 dept_name, 부서 번호 칼럼의 별명은 DN으로 부여하여 출력하여라.

```
SQL> SELECT dname dept_name, deptno AS DN
  2  FROM   department;

☞ 출력 결과
DEPT_NAME          DN
---------------- --------
컴퓨터공학과          101
멀티미디어학과        102
전자공학과            201
기계공학과            202
정보미디어학부        100
메카트로닉스학부      200
공과대학               10
```

칼럼 별명 부여 예

사용예

부서 테이블에서 부서 이름 칼럼의 별명은 "Department Name", 부서 번호 칼럼의 별명은 "부서 번호#"로 부여하여 출력하여라.

```
SQL> SELECT dname "Department Name", deptno "부서 번호#"
  2 FROM   department;

☞ 출력 결과

Department Name    부서 번호#
---------------- ----------
컴퓨터공학과            101
멀티미디어학과          102
전자공학과              201
기계공학과              202
정보미디어학부          100
메카트로닉스학부        200
공과대학                 10
```

별명에 대소문자를 구분하거나 공백 및 특수문자를 사용하는 경우에는 " "를 사용한다.

합성 연산자 사용

❑ 합성(concatenation)연산자 (||)

- ✓ 하나의 칼럼과 다른 칼럼, 산술 표현식 또는 상수 값과 연결하여 하나의 칼럼처럼 출력할 경우에 사용
- ✓ 출력 형식은 합성 연산자를 기준으로 좌, 우로 연결된 칼럼의 데이터가 하나의 칼럼 데이터처럼 출력
- ✓ 합성연산자를 사용하여 리터럴(literal) 문자 출력 가능
  - 리터럴 문자를 칼럼 값처럼 출력할 경우, 리터럴 문자를 단일 인용부호('') 사이에 포함하여 사용

합성 연산자 사용 예1

사용예

학생 테이블에서 학번과 이름 칼럼을 연결하여 "Student"라는 별명으로 하나의 칼럼처럼 연결하여 출력하여라.

```
SQL> SELECT studno || ' ' || name "Student"
  2  FROM   student;

☞ 출력 결과
Student
--------------------------------------------------------------------------------
10101 전인하
20101 이동훈
10102 박미경
10103 김영균
20102 박동진
10201 김진영
10104 지은경
10202 오유석
10203 하나리
10105 임유진
10106 서재진
10204 윤진욱
10107 이광훈
20103 김진경
20104 조명훈
10108 류민정

16 개의 행이 선택되었습니다.
```

학번과 이름사이에 공백 문자(' ')를 합성 연산자로 연결하여 출력하면 학번과 이름을 공백으로 구분하여 읽기 편하다.

합성 연산자 사용 예2

사용예

교수 테이블에서 교수 이름과 직급 칼럼사이에 '의 직급은' 라는 문자열을 추가하고 칼럼 이름을 "Title of Professor" 라는 별명을 사용하여 출력하여라.

```
SQL> SELECT name || '의 직급은' || position "Title of Professor"
  2 FROM   professor;
```

대소문자를 구별하여 칼럼 이름을 출력할 경우에 " "를 사용하여 칼럼 별명을 지정한다.

☞ 출력 결과

```
Title of Professor
----------------------------------------------------
김도훈의 직급은교수
이재우의 직급은조교수
성연희의 직급은조교수
염일웅의 직급은전임강사
권혁일의 직급은교수
이만식의 직급은부교수
전은지의 직급은전임강사
남은혁의 직급은부교수

8 개의 행이 선택되었습니다.
```

합성 연산자 사용 예3

실습예

학생 테이블에서 '홍길동의 키는 180, 몸무게는 55' 와 같은 형식으로 출력되도록 리터럴 문자를 추가하고, 칼럼 이름은 "학생의 키와 몸무게 정보" 라는 별명으로 출력하여라.

```
SQL> SELECT name || '의 키는 ' || height || ', 몸무게는 ' || weight
  2            "학생의 키와 몸무게 정보"
  3 FROM  student;
```

☞ 출력 결과

```
학생의 키와 몸무게 정보
------------------------------------------------------------
전인하의 키는 176, 몸무게는 72
이동훈의 키는 172, 몸무게는 64
박미경의 키는 168, 몸무게는 52
김영균의 키는 170, 몸무게는 88
박동진의 키는 182, 몸무게는 70
김진영의 키는 164, 몸무게는 48
지은경의 키는 161, 몸무게는 42
오유석의 키는 177, 몸무게는 92
하나리의 키는 160, 몸무게는 68
임유진의 키는 171, 몸무게는 54
서재진의 키는 186, 몸무게는 72
윤진욱의 키는 171, 몸무게는 70
이광훈의 키는 175, 몸무게는 92
김진경의 키는 166, 몸무게는 51
조명훈의 키는 184, 몸무게는 62
류민정의 키는 162, 몸무게는 72

16 개의 행이 선택되었습니다.
```

산술 연산자

❑ 산술 연산자(+,-, *, /)

- ✓ 칼럼 값에 산술 연산자를 적용하여 계산된 결과를 출력할 수 있는 기능을 제공
- ✓ 산술연산은 숫자 또는 날짜 타입으로 지정된 칼럼에만 사용 가능
- ✓ 산술연산자의 우선순위는 수학에서의 산술 연산자의 우선순위와 동일
  - 괄호를 사용하여 우선순위 변경 가능

산술 연산자 사용 예1

사용예

학생의 몸무게를 pound로 환산하고 칼럼 이름은 weight_pound 라는 별명으로 출력하여라. 1kg은 2.2pound이다.

```
SQL> SELECT name, weight, weight*2.2 AS weight_pound
  2 FROM   student;
```

☞ 출력 결과

NAME	WEIGHT	WEIGHT_POUND
전인하	72	158.4
이동훈	64	140.8
박미경	52	114.4
김영균	88	193.6
박동진	70	154
김진영	48	105.6
지은경	42	92.4
오유석	92	202.4
하나리	68	149.6
임유진	54	118.8
서재진	72	158.4
윤진욱	70	154
이광훈	92	202.4
김진경	51	112.2
조명훈	62	136.4
류민정	72	158.4

16 개의 행이 선택되었습니다.

산술 연산자 사용 예2

사용예

교수 테이블에서 교수 이름, 급여 그리고 보너스를 포함한 연봉을 출력하여라. 단, 보너스를 포함한 연봉은 급여*12를 한 결과에 보너스 100을 더한 값으로 계산한다.

```
SQL> SELECT name, sal, sal*12+100
  2 FROM  professor;
```

☞ 출력 결과

NAME	SAL	SAL*12+100
김도훈	500	6100
이재우	320	3940
성연희	360	4420
염일웅	240	2980
권혁일	450	5500
이만식	420	5140
전은지	210	2620
남은혁	400	4900

8 개의 행이 선택되었습니다.

연산자의 우선순위를 지정하지 않았으므로 기본적인 우선순위를 적용하여 sal*12를 먼저 계산한 다음 100을 더한다.

자주 발생하는 오라클 오류 메시지

❑ 오라클 오류 메시지

✓ 오류 메시지는 'ORA-nnnnn' 식으로 표현되며 오류의 원인과 해결 방법을 출력

❑ ORA-00942 : 테이블 또는 뷰가 존재하지 않습니다(table or view does not exist)

✓ 원인

- FROM 절에서 참조한 테이블이나 뷰가 존재하지 않거나 사용자가 해당 테이블에 대한 접근 권한이 없는 경우 발생
- 대부분은 테이블 이름을 잘못 입력한 경우에 발생

✓ 해결 방법

- FROM 절에서 테이블 이름을 정확하게 기술
  - SELECT * FROM tab; 명령문으로 데이터 딕셔너리를 조회하여 확인 필요
- 데이타베이스 관리자나 테이블 소유자로부터 해당 테이블에 접근할 수 있는 권한 부여받아야 함

자주 발생하는 오라클 오류 메시지

❑ ORA-00942 : 테이블 또는 뷰가 존재하지 않습니다

```
SQL> SELECT * FROM s_student;

☞ 출력 결과
SELECT * FROM s_student
              *
1행에 오류:
ORA-00942: 테이블 또는 뷰가 존재하지 않습니다
```

```
SQL> SELECT * FROM tab;

TNAME          TABTYPE
-------------- -------------
DEPARTMENT     TABLE
PROFESSOR      TABLE
SALGRADE       TABLE
STUDENT        TABLE
```

자주 발생하는 오라클 오류 메시지

❑ ORA-00904 : 열명이 부적합합니다(invalid identifier)

✓ 원인

- SELECT 절에서 지정한 칼럼이 FROM 절에서 지정한 테이블에 없는 경우에 발생

✓ 해결 방법

- 칼럼 이름을 정확하게 기술
- 칼럼 이름을 모를 경우, DESC 테이블명 명령어로 정확한 칼럼 이름 조회 필요

```
SQL> SELECT studno, name, salary
  2 FROM   student;

☞ 출력 결과
SELECT studno, name, salary
                     *
1행에 오류:
ORA-00904: 열명이 부적합합니다
```

자주 발생하는 오라클 오류 메시지

❑ ORA-00936: 누락된 표현식(missing expression)

✓ 원인

- SELECT 절에서 지정한 여러 칼럼 이름 사이에 콤마(,)를 누락한 경우
- SELECT 절의 맨 마지막 칼럼에 콤마를 추가한 경우

✓ 해결 방법

- 칼럼 이름 사이에 콤마(,)를 기술
- SELECT 절의 마지막에 있는 콤마를 삭제

```
SQL> SELECT studno, name, grade,
  2 FROM   student;

☞ 출력 결과
FROM student
*
2행에서 오류:
ORA-00936: 누락된 표현식
```

자주 발생하는 오라클 오류 메시지

❑ ORA-00923 : FROM 키워드가 있어야 할 곳에 없습니다 (FROM keyword not found WHERE expected)

✓ 원인

- SELECT 절 다음에 FROM 키워드를 입력하지 않은 경우
- FROM 키워드를 잘못 입력한 경우

✓ 해결 방법

- SELECT 절 다음에 FROM 키워드를 입력
- FROM 키워드를 올바르게 수정

```
SQL> SELECT studno, name, grade,
 2   student;

☞ 출력 결과
student
*
2행에서 오류:
ORA-00923: FROM 키워드가 있어야할
           곳에 없습니다
```

```
SQL> SELECT studno, name, grade,
   2 FRM    student;

☞ 출력 결과
FRM student
*
2행에서 오류:
ORA-00923: FROM 키워드가 있어야할
           곳에 없습니다
```