

데이터베이스 기초 실습과제

ORACLE®

1. 실습과제

1. 실습과제 정보

과정명	오라클 데이터베이스 기초	실습 유형	코드 기술형
실습과제명	기본적인 모델링 및 ERD 작성과 기초 쿼리 실습과제	수행시간	120분
실습 목표	<ul style="list-style-type: none">- 데이터베이스 설계 및 모델링 이해- 엔테티의 선정, ERD 작성법에 대한 이해- 테이블 생성 스크립트에 대한 이해- 테이블의 Primary Key, Foreign Key 작성법 이해- INSERT 쿼리구문 이해- 기본적인 SELECT문 사용법 이해- Join 연산의 이해- 기본적인 조인, 서브쿼리 사용법에 대한 이해- PL/SQL 프로시저, 함수 사용법 이해		

2. 실습과제 지문

간단한 학사관리 시스템을 구축하려 합니다.

고객과의 상담을 통해서 관리할 필요가 있는 다음 정보들을 파악했습니다.

2-1. 관계

- 학과와 학생은 1 : 다 관계
학과는 많은 학생을 가질 수 있다.
학생은 한 학과의 소속된다.
- 학과와 교수는 1 : 다 관계
학과는 많은 교수를 가질 수 있다.
교수는 한 학과의 소속된다.

- 교수와 개설과목은 1 : 다 관계
교수는 많은 과목을 가르칠 수 있다.
과목은 강의하는 교수 한 명이 지정된다.
- 과목과 학생은 다 : 다 관계
과목은 수강하는 많은 학생을 가진다.
학생은 많은 과목을 수강할 수 있다.

관계형 데이터베이스는 다:다 관계를 허용하지 않는다. 과목과 학생의 다:다 관계를 해소하기 위해서 조인테이블 역할을 수행할 수강 엔티티를 추가한다.

- 과목과 수강은 1 : 다 관계
- 학생과 수강은 1 : 다 관계

2-2. 데이터

학생(Student)

학번(student_id), 성명(student_name), 키(height), **학과코드(department_id)**

학과(Department)

학과코드(department_id), 학과명(department_name)

교수(Professor)

교수코드(professor_id), 교수명(professor_name), **학과코드(department_id)**

개설과목(Course)

과목코드(course_id), 과목명(course_name), **교수코드(professor_id)**,
시작일(start_date), 종료일(end_date)

수강(Student_Course)

학번(student_id), **과목코드(course_id)**

외부참조 키(FK) : 해당 엔티티와 관련된 데이터를 갖고 있는 프로퍼티가 아닌 다른 엔티티와의 관계를 설정하기 위한 키

3. 실습하기

부록 1. Oracle Database 문서를 참조하여 오라클 데이터베이스를 설치합니다.

부록 2. Oracle Database Sql Developer 문서를 참조하여 GUI 쿼리 툴을 설치합니다.

과제 1 : ERD 작성

앞서서 분석한 내용을 바탕으로 ERD 를 작성하시오.

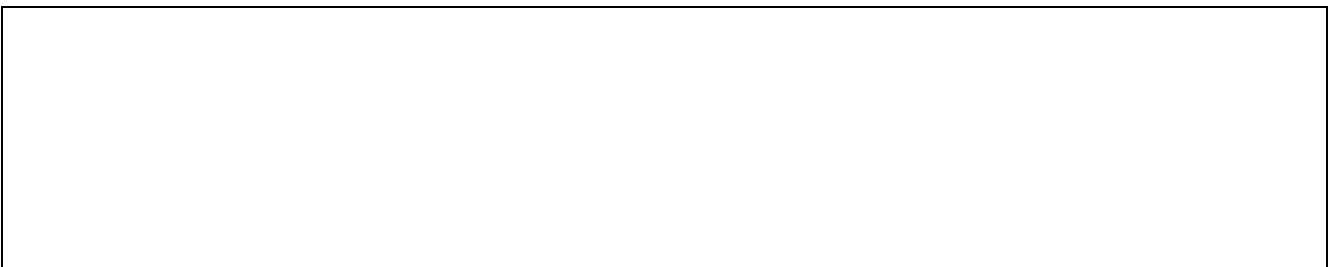
1. 엔티티들을 등록합니다.
2. 해당 엔티티의 고유 프로퍼티를 등록합니다.
3. 자료형의 도메인을 등록하여 엔티티 프로퍼티의 자료형을 설정할 때 적용합니다.
4. 엔티티의 PK 를 설정합니다.
5. 엔티티들의 관계를 맺어 외부참조 키를 추가합니다.

- 논리 모델



- 물리 모델

1. 앞에서 만든 논리 모델을 이용하여 릴레이션 모델을 생성합니다.
2. 사용하는 데이터베이스에 맞게 설정을 조정합니다.



실습 : 테이블 생성

- 과제 1 을 수행한 결과로 작성한 물리모델이 있다면 데이터 모델러를 사용하여 테이블 생성을 위한 DDL 문을 생성합니다.
- 과제 1 을 건너 뛰고 다음 과제를 진행하고자 하시는 분은 다음 DDL 문을 실행하여 테이블들을 데이터베이스에 생성합니다.

테이블 생성 쿼리

```
CREATE TABLE Course(  
    course_id NUMBER NOT NULL,  
    course_name VARCHAR2(100),  
    start_date DATE,  
    end_date DATE,  
    professor_id NUMBER NOT NULL);  
  
ALTER TABLE Course ADD CONSTRAINT pk_course PRIMARY KEY(course_id);  
  
CREATE TABLE Department(  
    department_id NUMBER NOT NULL,  
    department_name VARCHAR2(100));  
  
ALTER TABLE Department ADD CONSTRAINT pk_department PRIMARY KEY(department_id);  
  
CREATE TABLE Professor(  
    professor_id NUMBER NOT NULL,  
    professor_name VARCHAR2(100),  
    department_id NUMBER NOT NULL);  
  
ALTER TABLE Professor ADD CONSTRAINT pk_professor PRIMARY KEY(professor_id);  
  
CREATE TABLE Student(  
    student_id NUMBER NOT NULL,  
    student_name VARCHAR2(100),  
    height NUMBER,  
    department_id NUMBER NOT NULL);  
  
ALTER TABLE Student ADD CONSTRAINT pk_student PRIMARY KEY(student_id);
```

```
CREATE TABLE Student_Course(  
    student_id NUMBER NOT NULL,  
    course_id NUMBER NOT NULL);  
  
ALTER TABLE Course ADD CONSTRAINT fk_course_professor  
    FOREIGN KEY(professor_id)REFERENCES Professor(professor_id);  
  
ALTER TABLE Professor ADD CONSTRAINT fk_professor_department  
    FOREIGN KEY(department_id)REFERENCES Department(department_id);  
  
ALTER TABLE Student_Course ADD CONSTRAINT fk_student_course_course  
    FOREIGN KEY(course_id)REFERENCES Course(course_id);  
  
ALTER TABLE Student_Course ADD CONSTRAINT fk_student_course_student  
    FOREIGN KEY(student_id)REFERENCES Student(student_id);  
  
ALTER TABLE Student ADD CONSTRAINT fk_student_department  
    FOREIGN KEY(department_id)REFERENCES Department(department_id);
```

실습 : 테스트 데이터 생성

다음 작업은 테스트를 위한 가상데이터 등록입니다. 다음 쿼리를 실행하여 데이터를 입력합니다.

학과

```
insert into department values(1, '수학');  
insert into department values(2, '국문학');  
insert into department values(3, '정보통신공학');  
insert into department values(4, '모바일공학');
```

학생

```
insert into student values(1, '가길동', 177, 1);  
insert into student values(2, '나길동', 178, 1);  
insert into student values(3, '다길동', 179, 1);  
insert into student values(4, '라길동', 180, 2);  
insert into student values(5, '마길동', 170, 2);  
insert into student values(6, '바길동', 172, 3);  
insert into student values(7, '사길동', 166, 4);  
insert into student values(8, '아길동', 192, 4);
```

교수

```
insert into professor values(1, '가교수' ,1);  
insert into professor values(2, '나교수' ,2);  
insert into professor values(3, '다교수' ,3);  
insert into professor values(4, '빌게이츠' ,4);  
insert into professor values(5, '스티브잡스' ,3);
```

개설과목

```
insert into course values(1, '교양영어', '2016/9/2', '2016/11/30', 1);  
insert into course values(2, '데이터베이스 입문', '2016/8/20','2016/10/30', 3);  
insert into course values(3, '회로이론', '2016/10/20', '2016/12/30', 2);  
insert into course values(4, '공업수학', '2016/11/2', '2017/1/28', 4);  
insert into course values(5, '객체지향프로그래밍', '2016/11/1', '2017/1/30', 3);
```

수강

```
insert into student_course values(1, 1);  
insert into student_course values(2, 1);  
insert into student_course values(3, 2);  
insert into student_course values(4, 3);  
insert into student_course values(5, 4);  
insert into student_course values(6, 5);  
insert into student_course values(7, 5);
```


과제 2 : 테스트 쿼리 작성

과제 1 을 진행하지 않고 과제 2 를 진행하기 위해서는 앞 부분의 실습을 먼저 선행해야 합니다.

- 실습 : 테이블 생성
- 실습 : 테스트 데이터 생성

아래 질문내용에 맞는 SQL 문을 작성하세요.

과제 2-1.

학생번호, 학생명, 키높이, 학과번호, 학과명 정보를 출력합니다.

STUDENT_ID	STUDENT_NAME	HEIGHT	DEPARTMENT_ID	DEPARTMENT_NAME
1	1 가길동	177	1	수학
2	2 나길동	178	1	수학
3	3 다길동	179	1	수학
4	4 라길동	180	2	국문학
5	5 마길동	170	2	국문학
6	6 바길동	172	3	정보통신공학
7	7 사길동	166	4	모바일공학
8	8 아길동	192	4	모바일공학

과제 2-2.

'가교수' 교수의 교수아이디를 출력하세요.

PROFESSOR_ID
1

과제 2-3.

학과이름별 교수의 수를 출력하세요.

DEPARTMENT_NAME	COUNT(P,PROFESSOR_ID)
1 국문학	1
2 정보통신공학	2
3 수학	1
4 모바일공학	1

과제 2-4.

'정보통신공학'과의 학생정보를 출력하세요.

STUDENT_ID	STUDENT_NAME	HEIGHT	DEPARTMENT_ID	DEPARTMENT_NAME
1	6 바길동	172	3	정보통신공학

과제 2-5.

'정보통신공학'과의 교수명을 출력하세요.

PROFESSOR_ID	PROFESSOR_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	3 다교수	3	정보통신공학
2	5 스티브잡스	3	정보통신공학

과제 2-6.

가장 많은 학생이 있는 학과명을 출력하세요.

DEPARTMENT_NAME
1 수학

과제 2-7.

학생 중 성이 '아'인 학생이 속한 학과명과 학생명을 출력하세요.

STUDENT_NAME	DEPARTMENT_NAME
1 아길동	모바일공학

과제 2-8.

키가 180~190 사이에 속하는 학생 수를 출력하세요.

COUNT(STUDENT_ID)
1 1

과제 2-9.

학과이름별 키의 최고값, 평균값을 출력하세요.

DEPARTMENT_NAME	MAX(HEIGHT)	AVG(HEIGHT)
1 국문학	180	175
2 정보통신공학	172	172
3 수학	179	178
4 모바일공학	192	179

과제 2-10.

'다길동' 학생과 같은 학과에 속한 학생의 이름을 출력하세요.

STUDENT_NAME
1 가길동
2 나길동
3 다길동

과제 2-11.

2016 년 11 월 시작하는 과목을 수강하는 학생의 이름과 수강과목을 출력하세요.

STUDENT_NAME	COURSE_NAME
1 마길동	공업수학
2 바길동	객체지향프로그래밍
3 사길동	객체지향프로그래밍

과제 2-12.

'데이터베이스 입문' 과목을 수강신청한 학생의 이름은?

STUDENT_NAME
1 다길동

과제 2-13.

'빌게이츠' 교수의 과목을 수강신청한 학생수는?

COUNT(STUDENT_ID)
1 1

과제 2-14.

'사길동' 학생과 같은 과목을 수강신청한 학생이름을 출력하세요.

STUDENT_NAME
1 바길동
2 사길동

과제 2-15.

'사길동' 학생이 수강신청한 과목의 과목명과 시작일자를 출력하세요.

COURSE_NAME	START_DATE
1 객체지향프로그래밍	16/11/01

과제 2-16.

'다교수' 교수의 과목 중 2016 년 11 월 개강하는 과목을 수강 신청한 학생이름을 출력하세요.

STUDENT_NAME
1 바길동
2 사길동

과제 2-17.

개설과목이름별, 수강자수를 출력하세요.

힌트: 개설과목이름은 course 테이블을 참조하고 수강자수는 student_course 테이블을 참조한다.

COURSE_NAME	CNT
1 공업수학	1
2 교양영어	2
3 회로이론	1
4 객체지향프로그래밍	2
5 데이터베이스 입문	1