

3과목-데이터베이스 구축

(Part 3. SQL 응용)

데이터베이스 구축 총 파트

데이터베이스 구축 3과목은 총 5Part로 이루어져 있다.

1장 논리 데이터베이스 설계(49.72%)

2장 물리 데이터베이스 설계(22.91%)

3장 SQL 응용(26.26%)

4장 SQL 활용(1.12%)

5장 데이터 전환(0.00%)

물리 데이터베이스 설계

SQL 응용 Part는 7개의 섹션으로 구성되어 있다.

001 SQL의 개념

002 DDL

003 DCL

004 DML

005 DML-SELECT-1

006 DML-SELECT-2

007 DML-JOIN

3. SQL 응용-SEC_01(SQL의 개념)

이 장에서 꼭 알아야 할 키워드 Best 10

SQL, DDL, DROP, DCL, GRANT, DML, DELETE, SELECT, DISTINCT, JOIN

1) SQL(Structured Query Language)의 개요

- 1974년 IBM 연구소에서 개발한 SEQUEL에서 유래한다.
- 국제 표준 데이터베이스 언어이며, 많은 회사에서 관계형 데이터베이스(RDB)를 지원하는 언어로 채택하고 있다.
- 관계 대수와 관계 해석을 기초로 한 혼합 데이터 언어이다.
- 질의어지만 질의 기능만 있는 것이 아니라 데이터 구조의 정의, 데이터 조작, 데이터 제어 기능을 모두 갖추고 있다.

관계 대수(Relation Algebra) : 기존 릴레이션(테이블)들로부터 새로운 릴레이션을 생성하는 절차적 언어 문법 이라고 보면 된다. 릴레이션에 대해 기본적인 연산자들을 적용하여 보다 복잡한 관계 대수식을 점차적으로 만들 수 있다.

관계 대수의 9대 연산자

선택, 투영, 합집합, 교집합, 차집합, 카티션 곱, 조인, 디비전

관계 해석(Relation Calculus) : 관계 해석은 원하는 데이터가 무엇인지만 선언하는 이른바 비절차적 언어이다. 즉, 원하는 데이터만 명시하고 "어떻게 질의를 해석하는가"에 대해 언급이 없는 선언만 하는 언어인 것이다.

관계 해석은 튜플 관계 해석(Tuple Relational Calculus), 도메인 관계 해석(Domain Relational Calculus) 종류가 있다.

질의어(Query Language) : 질의어는 데이터베이스 파일과 범용 프로그래밍 언어를 정확히 알지 못하는 단말 사용자들이 단말기를 통해서 대화식으로 쉽게 DB를 이용할 수 있도록 되어 있는 비 절차어의 일종이다.

3. SQL 응용-SEC_01(SQL의 개념)

2) SQL의 분류

- DDL(Data Define Language, 데이터 정의어)

- DDL은 SCHEMA, DOMAIN, TABLE, VIEW, INDEX를 정의하거나 변경 또는 삭제할 때 사용하는 언어이다.
- 논리적 데이터 구조와 물리적 데이터 구조의 사상을 정의한다.
- 데이터베이스 관리자나 데이터베이스 설계자가 사용한다.
- DDL(데이터 정의어)의 세 가지 유형

명령어	기능
CREATE	SCHEMA, DOMAIN, TABLE, VIEW, INDEX를 정의한다.
ALTER	TABLE에 대한 정의를 변경하는 데 사용한다.
DROP	SCHEMA, DOMAIN, TABLE, VIEW, INDEX를 삭제한다.

스키마란?

1. 스키마는 데이터베이스의 구조와 제약 조건에 관한 전반적인 명세를 기술한 메타데이터의 집합이다.
2. 스키마는 데이터베이스를 구성하는 데이터 개체(Entity), 속성(Attribute), 관계(Relationship) 및 데이터 조작 시 데이터 값들이 갖는 제약 조건 등에 관해 전반적으로 정의한다.
3. 스키마는 사용자의 관점에 따라 외부 스키마, 개념 스키마, 내부 스키마로 나뉜다.

3. SQL 응용-SEC_01(SQL의 개념)

2) SQL의 분류

- DML(Data Manipulation Language, 데이터 조작어)

- DML은 데이터베이스 사용자가 응용 프로그램이나 질의어를 통하여 저장된 데이터를 실질적으로 처리하는 데 사용되는 언어이다.
- 데이터베이스 사용자와 데이터베이스 관리 시스템 간의 인터페이스를 제공한다.
- DML(데이터 조작어)의 네 가지 유형

명령어	기능
SELECT	테이블에서 조건에 맞는 튜플을 검색한다.
INSERT	테이블에 새로운 튜플을 삽입한다.
DELETE	테이블에서 조건에 맞는 튜플을 삭제한다.
UPDATE	테이블에서 조건에 맞는 튜플의 내용을 변경한다.

3. SQL 응용-SEC_01(SQL의 개념)

2) SQL의 분류

- DCL(Data Control Language, 데이터 제어어)

- DCL은 데이터의 보안, 무결성, 회복, 병행 수행 제어 등을 정의하는 데 사용되는 언어이다.
- 데이터베이스 관리자가 데이터 관리를 목적으로 사용한다.
- DCL(데이터 제어어)의 종류

명령어	기능
COMMIT	명령에 의해 수행된 결과를 실제 물리적 디스크로 저장하고, 데이터베이스 조작 작업이 정상적으로 완료되었음을 관리자에게 알려준다.
ROLLBACK	데이터베이스 조작 작업이 비정상적으로 종료되었을 때 원래의 상태로 복구한다.
GRANT	데이터베이스 사용자에게 사용 권한을 부여한다.
REVOKE	데이터베이스 사용자의 사용 권한을 취소한다.

SQL 응용 - SEC_01(SQL의 개념) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(SQL의 개념)

1. SQL의 분류 중 DDL에 해당하지 않는 것은?

- ① UPDATE ② ALTER
- ③ DROP ④ CREATE

UPDATE 명령어는 DML 명령어이다.

DDL(Data Define(Definition) Language, 데이터 정의어)

- DDL은 SCHEMA, DOMAIN, TABLE, VIEW, INDEX 등 을 정의하거나 혹은 변경 또는 삭제할 때 사용하는 언어이다.
- 논리적 데이터 구조와 물리적 데이터 구조의 사상을 정의한다.
- DDL(데이터 정의어)의 세 가지 명령어
 - > **CREATE** : SCHEMA, DOMAIN, TABLE, VIEW, INDEX 등을 정의한다.
 - > **ALTER** : 테이블에 대한 정의를 변경하는 데 사용한다.
 - > **DROP** : SCHEMA, DOMAIN, TABLE, VIEW, INDEX 등을 완전 삭제한다.

2. DML에 해당하는 SQL 명령으로만 나열된 것은?

3. 데이터 제어 언어(DCL)의 기능으로 옳지 않은 것은?

- ① 데이터 보안
- ② 논리적, 물리적 데이터 구조 정의
- ③ 무결성 유지
- ④ 병행수행 제어

논리적, 물리적 데이터 구조 정의하는 언어는 DDL이다.

DCL(Data Control Language, 데이터 제어어)

- DCL은 데이터의 보안, 무결성, 회복, 병행 수행 제어 등을 정의하는 데 사용되는 언어이다.
- 데이터베이스 관리자가 데이터 관리를 목적으로 사용한다.
- DCL(데이터 제어어)의 종류
 - > **COMMIT** : 명령에 의해 수행된 결과를 실제 물리 디스크로 저장하고, 데이터베이스 조작 작업이 정상적으로 완료되었음을 관리자에게 알려준다.
 - > **ROLLBACK** : 데이터베이스 조작 작업이 비정상적으로 종료되었을 때 원래의 상태로 복구한다. 단, COMMIT된 데이터는 다시 ROLLBACK이 되지 아니한다.
 - > **GRANT** : 데이터베이스 사용자에게 사용 권한을 부여한다.

SQL 응용 - SEC_01(SQL의 개념) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(SQL의 개념)

5. DDL(Data Define Language)의 명령어 중 스키마, 도메인, 인덱스 등을 정의할 때 사용하는 SQL문은?

- ① ALTER ② SELECT
- ③ CREATE ④ INSERT

CREATE : 정의, ALTER : 변경, DROP : 삭제

6. SQL에서 스키마(Schema), 도메인(Domain), 테이블(Table), 뷰(View), 인덱스(Index)를 정의하거나 변경 또는 삭제할 때 사용하는 언어는?

- ① DML(Data Manipulation Language)
- ② DDL(Data Definition Language)
- ③ DCL(Data Control Language)
- ④ IDL(Interactive Data Language)

IDL은 과학 연구를 위한 자료 처리 및 분석, 가시화 그리고 빠른 프로그램 개발을 위한 도구를 의미한다.

7. SQL의 TRUNCATE 명령어에 대한 설명으로 옳지 않은 것은?

- ① DELETE와 같이 테이블의 모든 데이터를 삭제한다.
- ② DROP과 달리 테이블 스키마는 제거되지 않고 유지된다.
- ③ DELETE에 비해 빠르게 데이터를 제거하는 것이 가능하다.
- ④ DELETE와 동일하게 ROLLBACK 명령어로 삭제된 데이터를 되살릴 수 있다.

데이터 삭제 방법 3가지

1. **DELETE** : 조건이 있다면 조건에 맞는 데이터를 삭제한다. 조건이 없다면 한 행, 한 행씩 삭제한다. COMMIT을 하지 않은 데이터는 얼마든지 ROLLBACK으로 복구가 가능하다.
2. **TRUNCATE** : 최초 테이블이 만들어졌던 상태, 즉 데이터가 1건도 없는 상태로 모든 데이터를 마치 삽으로 파듯이 삭제한다. 단, 컬럼 값은 존재한다. ROLLBACK으로 데이터 복구가 안된다. DDL언어에 속한다.
3. **DROP** : 데이터와 테이블 전체를 삭제하게 하고 사용하고 있는 저장 공간도 모두 반납하고 인덱스나 제약조건 등 모두 삭제된다.

8. SQL의 명령어를 DCL, DML, DDL로 구분할 경우, 다음 중 성격이 다른 하나는?

- ① CREATE ② SELECT

3. SQL 응용-SEC_02(DDL)

1) DDL(Data Define Language, 데이터 정의어)의 개념

; DDL(데이터 정의어)는 DB 구조, 데이터 형식, 접근 방식 등 DB를 구축하거나 수정할 목적으로 사용하는 언어이다.

- DDL은 번역한 결과가 데이터 사전(Data Dictionary)이라는 특별한 파일에 여러 개의 테이블로서 저장된다.
- DDL에는 CREATE SCHEMA, CREATE DOMAIN, CREATE TABLE, CREATE VIEW, CREATE INDEX, ALTER TABLE, DROP 등이 있다.

2) CREATE SCHEMA

; CREATE SCHEMA는 스키마를 정의하는 명령문이다.

- 스키마의 식별을 위해 스키마 이름과 소유권자나 허가권자를 정의한다.

표기 형식

CREATE SCHEMA 스키마명 AUTHORIZATION 사용자_id;

예제) 소유권자의 사용자 ID가 '홍길동'인 스키마 '대학교'를 정의하는 SQL문은 다음과 같다.

CREATE SCHEMA 대학교 AUTHORIZATION 홍길동;

3. SQL 응용-SEC_02(DDL)

3) CREATE DOMAIN

; CREATE DOMAIN은 도메인을 정의하는 명령문이다.

- 임의의 속성에서 취할 수 있는 값의 범위가 SQL에서 지원하는 전체 데이터 타입의 값이 아니고 일부분일 때, 사용자는 그 값의 범위를 도메인으로 정의할 수 있다.
- 정의된 도메인명은 일반적인 데이터 타입처럼 사용한다.

표기 형식

CREATE DOMAIN 도메인명 [AS] 데이터_타입

[DEFAULT 기본값]

[CONSTRAINT 제약 조건명 CHECK (범위 값)];

구문에서 대괄호([])의 의미 : SQL문에서 [AS] 처럼 대괄호로 묶은 명령어들은 생략이 가능하다는 의미한다.

3. SQL 응용-SEC_02(DDL)

3) CREATE DOMAIN

- 데이터 타입 : SQL에서 지원하는 데이터 타입
- 기본값 : 데이터를 입력하지 않았을 때 자동으로 입력되는 값

예제 '성별'을 '남' 또는 '여'와 같이 정해진 1개의 문자로 표현되는 도메인 GENDER를 정의하는 SQL문은 다음과 같다.

CREATE DOMAIN GENDER CHAR(1) 정의된 도메인은 이름이 'GENDER'이며, 문자형이고 크기는 1이다.

DEFAULT '남'

도메인 GENDER를 지정한 속성의 기본값은 '남'이다.

CONSTRAINT VALID-GENDER CHECK(VALUE IN ('남', '여')); GENDER를 지정한 속성에는 남, '여' 하나의 값만을 저장할 수 있다.

3. SQL 응용-SEC_02(DDL)

3) CREATE DOMAIN

- SQL에서 지원하는 기본 데이터 타입

- 정수(Integer) : INTEGER(4Byte 정수), SMALLINT(2Byte 정수)
- 실수(Float) : FLOAT, REAL, DOUBLE PRECISION
- 형식화된 숫자 : DEC(i, j), 단 i : 전체 자릿수, j : 소수부 자릿수
- 고정길이 문자: CHAR(n), CHARACTER(n), 단 n : 문자수
- 가변길이 문자 : VARCHAR(n), CHARACTERVARYING(n), 단 n : 최대 문자수
- 고정길이 비트(Bit String): BIT(n)
- 가변길이 비트열 : VARBIT(n)
- 날짜 : DATE
- 시간 : TIME

3. SQL 응용-SEC_02(DDL)

4) CREATE TABLE

; CREATE TABLE은 테이블을 정의하는 명령문이다.

표기 형식

CREATE TABLE 테이블명

(속성명 데이터_타입 [DEFAULT 기본값] [NOT NULL], ...

[, PRIMARY KEY(기본키_속성명, ...)]

[, UNIQUE(대체키_속성명, ...)]

[, FOREIGN KEY(외래키_속성명, ...)]

[REFERENCES 참조 테이블(기본키_속성명, ...)]

[ON DELETE 옵션]

[ON UPDATE 옵션]

[, CONSTRAINT 제약조건명] [CHECK (조건식)];

3. SQL 응용-SEC_02(DDL)

4) CREATE TABLE

- 기본 테이블에 포함될 모든 속성에 대하여 속성명과 그 속성의 데이터 타입, 기본 값, NOT NULL 여부를 지정한다.
- PRIMARY KEY : 기본키로 사용할 속성 또는 속성의 집합을 지정한다.
- UNIQUE : 대체키로 사용할 속성 또는 속성의 집합을 지정하는 것으로 UNIQUE로 지정한 속성은 중복된 값을 가질 수 없다.
- FOREIGN KEY~ REFERENCES ~
 - 참조할 다른 테이블과 그 테이블을 참조할 때 사용할 외래키 속성을 지정한다.
 - 외래키가 지정되면 참조 무결성의 CASCADE 법칙이 적용된다.
 - ON DELETE 옵션 : 참조 테이블의 튜플이 삭제되었을 때 기본 테이블에 취해야 할 사항을 지정한다.

옵션에는 NO ACTION, CASCADE, SET NULL, SET DEFAULT가 있다.

NOT NULL : NULL이란 모르는 값 또는 적용할 수 없는 값을 의미하는 것으로 NOT NULL은 특정 속성이 데이터 없이 비어 있어서는 안 된다는 것을 지정할 때 사용한다.

참조 무결성의 CASCADE 법칙 : 참조 무결성 제약이 설정된 기본 테이블의 어떤 데이터를 삭제할 경우, 그 데이터와 밀접하게 연관되어 있는 다른 테이블의 데이터들도 도미노처럼 자동으로 삭제된다. 이러한 법칙을 '계단식', '연속성'이라는 사전적 의미를 가진 CASCADE 법칙이라고 한다.

3. SQL 응용-SEC_02(DDL)

4) CREATE TABLE

- ON UPDATE 옵션 : 참조 테이블의 참조 속성 값이 변경되었을 때 기본 테이블에 취해야 할 사항을 지정한다. 옵션에는 NO ACTION, CASCADE, SET NULL, SET DEFAULT가 있다.
 - ▶ NO ACTION : 참조 테이블에 변화가 있어도 기본 테이블에는 아무런 조치를 취하지 않는다.
 - ▶ CASCADE : 참조 테이블의 튜플이 삭제되면 기본 테이블의 관련 튜플도 모두 삭제되고, 속성이 변경되면 관련 튜플의 속성 값도 모두 변경된다.
 - ▶ SET NULL : 참조 테이블에 변화가 있으면 기본 테이블의 관련 튜플의 속성 값을 NULL로 변경한다.
 - ▶ SET DEFAULT : 참조 테이블에 변화가 있으면 기본 테이블의 관련 튜플의 속성값을 기본값으로 변경한다.
- CONSTRAINT : 제약 조건의 이름을 지정한다. 이름을 지정할 필요가 없으면 CHECK절만 사용하여 속성 값에 대한 제약 조건을 명시한다.
- CHECK : 속성 값에 대한 제약 조건을 정의한다.

3. SQL 응용-SEC_02(DDL)

4) CREATE TABLE

예제) '이름', '학번', '전공', '성별', '생년월일'로 구성된 <학생> 테이블을 정의하는 SQL문을 작성하시오. 단, 제약조건은 다음과 같다.

- '이름' 은 NULL이 올 수 없고 '학번' 은 기본키이다.
- '전공'은 <학과> 테이블의 '학과코드'를 참조하는 외래키로 사용된다.
- <학과> 테이블에서 삭제가 일어나면 관련된 튜플들의 전공 값을 NULL로 만든다.
- <학과> 테이블에서 '학과코드'가 변경되면 전공 값도 같은 값으로 변경한다.
- '생년월일'은 1980-01-01 이후의 데이터만 저장할 수 있다.
- 제약 조건의 이름은 '생년월일제약'으로 한다.
- 각 속성의 데이터 타입은 적당하게 지정한다. 단 '성별'은 도메인 'GENDER'를 사용한다.

3. SQL 응용-SEC_02(DDL)

4) CREATE TABLE

CREATE TABLE 학생

(이름 VARCHAR(15) NOT NULL

학번 CHAR(8),

전공 CHAR(5),

성별 GENDER,

생년월일 DATE,

PRIMARY KEY(학번),

FOREIGN KEY(전공) REFERENCES 학과(학과코드)

ON DELETE SET NULL

ON UPDATE CASCADE,

CONSTRAINT 생년월일제약 CHECK(생년월일 >='1980-01-01')); '생년월일' 속성에는 1980-01-01 이후의 값만을 저장할 수 있으며, 이 제약 조건의 이름은 '생년월일제약'이다.

<학생> 테이블을 생성한다.

'이름' 속성은 최대 문자 15자로 NULL 값을 갖지 않는다.

'학번' 속성은 문자 8자이다.

'전공' 속성은 문자 5자이다.

'성별' 속성은 'GENDER' 도메인을 자료형으로 사용한다.

'생년월일' 속성은 DATE 자료형을 갖는다.

'학번'을 기본키로 정의한다.

'전공' 속성은 <학과> 테이블의 '학과코드' 속성을 참조하는 외래키

<학과> 테이블에서 튜플이 삭제되면 관련된 모든 튜플의 '전공'

속성의 값을 NULL로 변경한다.

<학과> 테이블에서 '학과코드'가 변경되면 관련된 모든 튜플의

'전공' 속성의 값도 같은 값으로 변경한다.

3. SQL 응용-SEC_02(DDL)

4) CREATE TABLE

; 기존 테이블의 정보를 이용해 새로운 테이블을 정의할 수 있다.

표기 형식

CREATE TABLE 신규 테이블명 AS SELECT 속성명[, 속성명, ...] FROM 기존 테이블명;

- 기존 테이블에서 추출되는 속성의 데이터 타입과 길이는 신규 테이블에 그대로 적용된다.
- 기존 테이블의 NOT NULL의 정의는 신규 테이블에 그대로 적용된다.
- 기존 테이블의 제약 조건은 신규 테이블에 적용되지 않으므로 신규 테이블을 정의한 후 ALTER TABLE 명령을 이용해 제약 조건을 추가해야 한다.
- 기존 테이블의 일부 속성만 신규 테이블로 생성할 수 있으며, 기존 테이블의 모든 속성을 신규 테이블로 생성할 때는 속성명 부분에 '*'를 입력한다.

예제) <학생> 테이블의 '학번', '이름', '학년' 속성을 이용하여 <재학생> 테이블을 정의하는 SQL문을 작성하시오.

CREATE TABLE 재학생 AS SELECT 학번, 이름, 학년 FROM 학생;

3. SQL 응용-SEC_02(DDL)

5) CREATE VIEW

; CREATE VIEW는 뷰(View)를 정의하는 명령문이다.

표기 형식

CREATE VIEW 뷰명[(속성명, 속성명, ...)] AS SELECT문;

- SELECT문을 서브 쿼리로 사용하여 SELECT문의 결과로서 뷰를 생성한다.
- 서브 쿼리인 SELECT문에는 UNION이나 ORDER BY절을 사용할 수 없다.
- 속성명을 기술하지 않으면 SELECT문의 속성명이 자동으로 사용된다.

예제) <고객> 테이블에서 '주소'가 '안산시'인 고객들의 '성명'과 '전화번호'를 '안산고객' 이라는 뷰로 정의하시오.

CREATE VIEW 안산고객(성명, 전화번호)

AS SELECT 성명, 전화번호

FROM 고객

WHERE 주소 = '안산시';

뷰(View) : 뷰는 하나 이상의 기본 테이블로부터 유도되는 이름을 갖는 가상 테이블(Virtual Table)이다. 테이블은 물리적으로 구현되어 실제로 데이터가 저장되지만, 뷰는 물리적으로 구현 되지 않는다. 즉 뷰를 생성하면 뷰 정의가 시스템 내에 저장되었다가 SQL내에서 뷰 이름을 사용 하면 실행 시간에 뷰 정의가 대체되어 수행된다.

서브 쿼리(Sub Query) : 서브 쿼리는 조건절에 주어진 질의로서, 상위 질의에 앞서 실행되며 그 검색 결과는 상위 질의의 조건절의 피연산자로 사용된다.

3. SQL 응용-SEC_02(DDL)

6) CREATE INDEX

; CREATE INDEX는 인덱스를 정의하는 명령문이다.

표기 형식

CREATE [UNIQUE] INDEX 인덱스명

ON 테이블명(속성명 [ASC | DESC] [, 속성명 [ASC DESC]])

[CLUSTER];

● UNIQUE

- 사용된 경우 : 중복 값이 없는 속성으로 인덱스를 생성한다.
- 생략된 경우 : 중복 값을 허용하는 속성으로 인덱스를 생성한다.

● 정렬 여부 지정

- ASC : 오름차순 정렬
- DESC : 내림차순 정렬
- 생략된 경우 : 오름차순으로 정렬됨

● CLUSTER : 사용하면 인덱스가 클러스터드 인덱스로 설정됨

인덱스(Index) : 인덱스는 검색 시간을 단축시키기 위해 만든 보조적인 데이터 구조다.
클러스터드 인덱스(Clustered Index) : 인덱스 키의 순서에 따라 데이터가 정렬되어 저장되는 방식이다. 실제 데이터가 순서대로 저장되어 있어 인덱스를 검색 하지 않아도 원하는 데이터를 빠르게 찾을 수 있다. 하지만 데이터 삽입, 삭제, 발생 시 순서를 유지하기 위해 데이터를 재정렬해야 한다.
논 클러스터드 인덱스(Non-Clustered Index) : 인덱스의 키 값만 정렬되어 있을 뿐 실제 데이터는 정렬되지 않는 방식이다. 데이터를 검색하기 위해서는 먼저 인덱스를 검색하여 실제 데이터의 위치를 확인해야 하므로 클러스터드 인덱스에 비해 검색 속도 가 떨어진다.

3. SQL 응용-SEC_02(DDL)

6) CREATE INDEX

예제) <고객> 테이블에서 UNIQUE한 특성을 갖는 '고객번호' 속성에 대해 내림차순으로 정렬하여 '고객번호_idx'라는 이름으로 인덱스를 정의하시오.

```
CREATE UNIQUE INDEX 고객번호_idx  
ON 고객(고객번호 DESC);
```

7) ALTER TABLE

; ALTER TABLE은 테이블에 대한 정의를 변경하는 명령문이다.

표기 형식

```
ALTER TABLE 테이블명 ADD 속성명 데이터_타입 [DEFAULT '기본값'];
```

```
ALTER TABLE 테이블명 ALTER 속성명 [SET DEFAULT '기본값'];
```

```
ALTER TABLE 테이블명 DROP COLUMN [CASCADE];
```

- ADD : 새로운 속성(열)을 추가할 때 사용한다.
- ALTER : 특정 속성의 Default 값을 변경할 때 사용한다.
- DROP COLUMN : 특정 속성을 삭제할 때 사용한다.

3. SQL 응용-SEC_02(DDL)

7) ALTER TABLE

예제1) <학생> 테이블에 최대 3문자로 구성되는 '학년' 속성 추가하시오.

```
ALTER TABLE 학생 ADD 학년 VARCHAR2(9);
```

예제2) <학생> 테이블의 '학번' 필드의 데이터 타입과 크기를 VARCHAR2(10)으로 하고 NULL 값이 입력되지 않도록 변경하시오.

```
ALTER TABLE 학생 ALTER 학번 VARCHAR2(10) NOT NULL;
```

3. SQL 응용-SEC_02(DDL)

8) DROP

; DROP은 스키마, 도메인, 기본 테이블, 뷰, 테이블, 인덱스, 제약 조건 등을 제거하는 명령문이다.

표기 형식

DROP SCHEMA 스키마명 [CASCADE | RESTRICT];

DROP DOMAIN 도메인명 [CASCADE | RESTRICT];

DROP TABLE 테이블명 [CASCADE | RESTRICT];

DROP VIEW 뷰명 [CASCADE | RESTRICT];

DROP INDEX 인덱스명 [CASCADE | RESTRICT];

DROP CONSTRAINT 제약조건명;

- DROP SCHEMA : 스키마를 제거한다.
- DROP DOMAIN : 도메인을 제거한다.
- DROP TABLE : 테이블을 제거한다.
- DROP VIEW : 뷰를 제거한다.
- DROP INDEX : 인덱스를 제거한다.
- DROP CONSTRAINT : 제약 조건을 제거한다.

3. SQL 응용-SEC_02(DDL)

8) DROP

- CASCADE : 제거할 요소를 참조하는 다른 모든 개체를 함께 제거한다. 즉 주 테이블의 데이터 제거 시 각 외래키와 관계를 맺고 있는 모든 데이터를 제거하는 참조 무결성 제약 조건을 설정하기 위해 사용된다.
- RESTRICT : 다른 개체가 제거할 요소를 참조 중일 때는 제거를 취소한다.

예제) <학생> 테이블을 제거하되, <학생> 테이블을 참조하는 모든 데이터를 함께 제거하시오.

```
DROP TABLE 학생 CASCADE;
```

SQL 응용 - SEC_02(DDL) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DDL)

1. CREATE TABLE문에 포함되지 않는 기능은?

- ① 속성 타입 변경
- ② 속성의 NOT NULL 여부 지정
- ③ 기본키를 구성하는 속성 지정
- ④ CHECK 제약조건의 정의

속성 타입 변경은 ALTER TABLE로 가능한 것이다.

1. CREATE TABLE

CREATE TABLE은 테이블을 정의하는 명령어이다.

표기형식

CREATE TABLE 테이블명

(속성명 데이터_타입 [DEFAULT 기본값] [NOT NULL], ...

[, PRIMARY KEY(기본키_속성명, ...)]

[, UNIQUE(대체키_속성명, ...)]

[, FOREIGN KEY(외래키_속성명, ...)]

[REFERENCES 참조 테이블명(기본키_속성명, ...)]

[ON DELETE 옵션]

[ON UPDATE 옵션]

3. 테이블 두 개를 조인하여 뷰 V_1을 정의하고, V_1을 이용하여 뷰 V_2를 정의하였다. 다음 명령 수행 후 결과로 옳은 것은?

DROP VIEW V_1 CASCADE;

- ① V_1만 삭제된다.
- ② V_2만 삭제된다.
- ③ V_1과 V_2 모두 삭제된다.
- ④ V_1과 V_2 모두 삭제되지 않는다.

CASCADE : 제거할 요소를 참조하는 다른 모든 개체를 함께 제거한다.

즉 주 테이블의 데이터 제거 시 각 외래키와 관계를 맺고 있는 모든 데이터를 제거하는 참조 무결성 제약 조건을 설정하기 위해 사용된다.

RESTRICT : 다른 개체가 제거할 요소를 참조 중일 때는 제거를 취소한다.

4. SQL의 DROP문에 관한 설명 중 잘못된 것은?

- ① 해당 TABLE에 삽입된 TUPLE들도 없어진다.
- ② 해당 TABLE에 대해 만들어진 INDEX가 없어진다.
- ③ 해당 TABLE에 대해 만들어진 VIEW가 없어진다.
- ④ 해당 TABLE에 참조관계가 있는 TABLE이 없어진다.

DROP 명령은 삭제하려는 테이블과 함께 그 테이블로부터 유도하여 만든 인덱스, 뷰, 트리거 등 모두 제거한다. 그러나 참조하던 테이블은 해당

SQL 응용 - SEC_02(DDL) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DDL)

5. 참조 무결성을 유지하기 위하여 DROP문에서 부모 테이블의 항목 값을 삭제할 경우 자동적으로 자식 테이블의 해당 레코드를 삭제하기 위한 옵션은?

- ① CLUSTER
- ② CASCADE
- ③ SET-NULL
- ④ RESTRICTED

제거할 요소를 참조하는 다른 모든 개체를 함께 제거하는 옵션은 CASCADE이다.

CLUSTER : 사용하면 인덱스가 클러스터드 인덱스로 설정이 된다.

SET NULL : 참조 테이블에 변화가 있으면 기본 테이블의 관련 튜플의 속성 값을 NULL값으로 변경한다.

6. 학생 테이블을 생성한 후, 성별 필드가 누락되어 이를 추가하려고 한다. 이에 적합한 SQL 명령어는?

- ① INSERT ② ALTER
- ③ DROP ④ MODIFY

테이블 생성은 CREATE, 테이블 삭제는 DROP, 테이블 수정은 ALTER

7. 스키마의 식별을 위해 스키마 이름과 소유권자나 허가권자를 정의하는 명령문은?

- ① DEFINE SCHEMA
- ② ALTER SCHEMA
- ③ CREATE SCHEMA
- ④ DROP SCHEMA

CREATE SCHEMA

CREATE SCHEMA는 스키마를 정의하는 명령문이다.

- 스키마의 식별을 위해서 스키마 이름과 소유권자나 허가권자를 정의한다.

표기 형식

CREATE SCHEMA 스키마명 AUTHORIZATION 사용자_ID;

8. SQL의 데이터 타입 중에서 4바이트 정수형을 나타내는 것은?

- ① DECIMAL ② INTEGER
- ③ SMALLINT ④ FLOAT

SQL에서 지원하는 기본적인 데이터 타입

1. 정수(INTEGER) : INTEGER(4바이트 정수), SMALLINT(2바이트)

2. 실수(FLOAT) : FLOAT, REAL, DOUBLE PRECISION

SQL 응용 - SEC_02(DDL) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DDL)

9. 학년 속성이 가질 수 있는 값의 범위를 1~4의 2바이트 정수로만 사용할 수 있는 도메인 SYEAR를 정의하는 명령문으로 올바른 것은? (단, 기본값은 1이며, NULL일 수 없음)

① CREATE DOMAIN SYEAR SMALLINT DEFAULT 1 CONSTRAINT VALID-SYEAR CHECK(VALUE IN (1, 2, 3, 4)) NOT NULL;

② DEFINE DOMAIN SYEAR SMALLINT DEFAULT 1 CONSTRAINT VALID-SYEAR CHECK(VALUE IN (1, 2, 3, 4)) NOT NULL;

③ CREATE DOMAIN SYEAR SMALLINT DEFAULT 1 CONSTRAINT VALID-SYEAR CHECK(VALUE IN (1~4)) NOT NULL;

④ DEFINE DOMAIN SYEAR SMALLINT DEFAULT 1 CONSTRAINT VALID-SYEAR CHECK(VALUE IN (1~4)) NOT NULL;

도메인(사용자 정의 데이터 타입) 정의문

CREATE DOMAIN 도메인명 데이터_타입

[DEFAULT 기본값]

[CONSTRAINT 제약조건명 CHECK(범위값)];

10. SQL에 존재하는 CREATE TABLE 명령어에 대한 설명으로 옳지 않은 것은?

11. CREATE TABLE에 대한 설명으로 틀린 것은?

① 테이블명 및 해당 테이블에 속하는 컬럼 이름, 데이터 타입 등을 명시한다.

② PRIMARY KEY 절에서는 기본키 속성을 지정한다.

③ CHECK 절은 인덱스에 대한 정보를 저장한다.

④ NOT NULL은 널 값을 허용하지 않을 때 지정한다.

CHECK 절은 특정한 속성값의 제약조건을 정의할 때 사용한다. 보통, 조건절이나 범위값이 들어간다.

12. 다음 명령문에 대한 설명으로 잘못된 것은?

CREATE TABLE 학생

(이름 VARCHAR(15) Not Null,

학번 VARCHAR(15) Not Null,

전공 VARCHAR(20) Not Null,

성별 GENDER,

생년월일 DATE,

PRIMARY KEY (학번),

FOREIGN KEY (전공) REFERENCES 학과(학과코드)

CHECK (성별 = '남'));

SQL 응용 - SEC_02(DDL) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DDL)

13. '고객' 테이블의 기본키인 '국가' 속성에 대해 오름차순 정렬하여 고객_INX 인덱스를 구성하기 위한 명령문은? (단, 동일 인덱스 값을 갖는 튜플들을 그룹으로 묶음)

- ① CREATE UNIQUE INDEX 고객_INX ON 고객(국가 ASC);
- ② CREATE UNIQUE INDEX 고객_INX ON 고객(국가 ASC) CLUSTER;
- ③ CREATE UNIQUE INDEX 고객_INX ON 고객(국가 DESC);
- ④ CREATE UNIQUE INDEX 고객_INX ON 고객(국가 DESC) CLUSTER;

인덱스 표기 형식

CREATE [UNIQUE] INDEX 인덱스명

ON 테이블명(속성명 [ASC | DESC])

[CLUSTER];

정렬 여부 지정

- ASC : 기본값으로 오름차순 정렬

- DESC : 명시적으로 표현해야 하면 내림차순 정렬

CLUSTER : 동일 인덱스 값을 갖는 튜플들을 그룹으로 묶을 때 사용하는 클러스터드 인덱스를 생성한다.

14. 다음 SQL 명령어에 대해 바르게 설명한 것은?

15. SQL의 DROP TABLE 명령어에 대한 설명으로 옳은 것은?

- ① 테이블을 수정하는 데 사용된다.
- ② CASCADE 옵션 하나만 존재한다.
- ③ 제거될 테이블을 참조하는 모든 제약과 뷰를 자동적으로 스키마로부터 삭제시키는 옵션이 존재한다.
- ④ 테이블이 제약이나 뷰로부터 참조되지 않는 경우에만 삭제되도록 하는 옵션은 존재하지 않는다.

DROP TABLE 명령어

SCHEMA, DOMAIN, TABLE, VIEW, TRIGGER 등을 제거하는데 사용된다.

- CASCADE와 RESTRICT의 두 개의 옵션이 존재한다.

- CASCADE 옵션을 사용하면 제거될 테이블을 참조하는 모든 제약과 뷰도 자동으로 스키마로부터 삭제된다.

- RESTRICT 옵션이 사용되면 테이블이 제약이나 뷰로부터 참조되지 않는 경우에만 삭제된다. 그리고 참조가 되어있으면 이 옵션은 명령을 취소한다.

3. SQL 응용-SEC_03(DCL)

1) DCL(Data Control Language, 데이터 제어어)의 개념

; DCL(데이터 제어어)는 데이터의 보안, 무결성, 회복, 병행 제어 등을 정의하는 데 사용하는 언어이다.

- DCL은 데이터베이스 관리자(DBA)가 데이터 관리를 목적으로 사용한다.
- DCL에는 GRANT, REVOKE, COMMIT, ROLLBACK, SAVEPOINT 등이 있다.

2) GRANT/REVOKE

; 데이터베이스 관리자가 데이터베이스 사용자에게 권한을 부여하거나 취소하기 위한 명령어이다.

- GRANT: 권한 부여를 위한 명령어
- REVOKE : 권한 취소를 위한 명령어
- 사용자 등급 지정 및 해제

표기 형식

- GRANT 사용자등급 TO 사용자_ID_리스트 [IDENTIFIED BY 암호];
- REVOKE 사용자등급 FROM 사용자_ID_리스트;

사용자 등급

- DBA : 데이터베이스 관리자
- RESOURCE: 데이터베이스 및 테이블 생성 가능자
- CONNECT: 단순 사용자

3. SQL 응용-SEC_03(DCL)

2) GRANT/REVOKE

예제1) 사용자 ID가 "NABI"인 사람에게 데이터베이스 및 테이블을 생성할 수 있는 권한을 부여하는 SQL문을 작성하시오.

GRANT RESOURCE TO NABI;

예제2) 사용자 ID가 "STAR"인 사람에게 단순히 데이터베이스에 있는 정보를 검색할 수 있는 권한을 부여하는 SQL문을 작성하시오.

GRANT CONNECT TO STAR;

● 테이블 및 속성에 대한 권한 부여 및 취소

- GRANT 권한_리스트 ON 개체 TO 사용자 [WITH GRANT OPTION];

- REVOKE [GRANT OPTION FOR] 권한_리스트 ON 개체 FROM 사용자[CASCADE];

권한 종류 : ALL, SELECT, INSERT, DELETE, UPDATE, ALTER 등

WITH GRANT OPTION : 부여 받은 권한을 다른 사용자에게 다시 부여할 수 있는 권한을 부여함

GRANT OPTION FOR : 다른 사용자에게 권한을 부여할 수 있는 권한을 취소함

CASCADE : 권한 취소 시 권한을 부여 받았던 사용자가 다른 사용자에게 부여한 권한도 연쇄적으로 취소함

3. SQL 응용-SEC_03(DCL)

2) GRANT/REVOKE

예제3) 사용자 ID가 "NABI"인 사람에게 <고객> 테이블에 대한 모든 권한과 다른 사람에게 권한을 부여할 수 있는 권한까지 부여하는 SQL문을 작성하시오.

GRANT ALL ON TO NABI WITH GRANT OPTION;

예제4) 사용자 ID가 "STAR"인 사람에게 부여한 <고객>테이블에 대한 권한 중 UPDATE 권한을 다른 사람에게 부여할 수 있는 권한만 취소하는 SQL문을 작성하시오.

REVOKE GRANT OPTION FOR UPDATE ON 고객 FROM STAR;

3) COMMIT

; 트랜잭션이 성공적으로 끝나면 데이터베이스가 새로운 일관성(Consistency) 상태를 가지기 위해 변경된 모든 내용을 데이터베이스에 반영하여야 하는데, 이때 사용하는 명령이 COMMIT이다.

- COMMIT 명령을 실행하지 않아도 DML문이 성공적으로 완료되면 자동으로 COMMIT되고, DML이 실패하면 자동으로 ROLLBACK이 되도록 **"Auto Commit"** 기능을 설정할 수 있다.

COMMIT, ROLLBACK, SAVEPOINT는 트랜잭션을 제어하는 용도로 사용되므로 TCL (Transaction Control Language)로 분류하기도 한다. 하지만 기능을 제어하는 명령이라는 공통점으로 DCL의 일부로 분류하기도 한다.

트랜잭션(Transaction) : 트랜잭션은 데이터베이스에서 하나의 논리적 기능을 수행하기 위한 일련의 연산 집합으로서 작업의 단위가 된다. 트랜잭션은 데이터베이스 관리 시스템에서 회복 및 병행 제어 시에 처리되는 작업의 논리적 단위이다. 하나의 트랜잭션은 COMMIT되거나 ROLLBACK 되어야 한다.

COMMIT 명령 사용 여부 : 트랜잭션이 시작되면 데이터베이스의 데이터를 주 기억장치에 올려 처리하다가 COMMIT 명령이 내려지면 그때서야 처리된 내용을 보조기억장치에 저장한다. 그러니까 COMMIT 명령을 사용하지 않고 DBMS를 종료하면

그때까지 작업했던 모든 내용이 보조기억장치의 데이터베이스에 하나도 반영되지 않고 종료되는 것이다. 이처럼 실수로 COMMIT 명령 없이 DBMS를 종료하는 것에 대비하여 대부분의 DBMS들은 Auto Commit 기능을 제공하고 있다.

3. SQL 응용-SEC_03(DCL)

4) ROLLBACK

; ROLLBACK은 아직 COMMIT되지 않은 변경된 모든 내용들을 취소하고 데이터베이스를 이전 상태로 되돌리는 명령어이다.

- 트랜잭션 전체가 성공적으로 끝나지 못하면 일부 변경된 내용만 데이터베이스에 반영되는 비일관성(Inconsistency)인 상태를 가질 수 있기 때문에 일부분만 완료된 트랜잭션은 롤백(Rollback)되어야 한다.

5) SAVEPOINT

; SAVEPOINT는 트랜잭션 내에 ROLLBACK할 위치인 저장점을 지정하는 명령어이다.

- 저장점을 지정할 때는 이름을 부여하며, ROLLBACK시 지정된 저장점까지의 트랜잭션 처리 내용이 취소된다.

3. SQL 응용-SEC_03(DCL)

〈사원〉		
사원번호	이름	부서
10	김기혁	기획부
20	박인사	인사부
30	최재무	재무부
40	오영업	영업부

예제1) <사원> 테이블에서 '사원번호'가 40인 사원의 정보를 삭제한 후 COMMIT을 수행하시오.

DELETE FROM 사원 WHERE 사원번호 = 40;

COMMIT;

해설 : DELETE 명령을 수행한 후 COMMIT 명령을 수행했으므로 DELETE 명령으로 삭제된 레코드는 이후 ROLLBACK 명령으로 되돌릴 수 없다.

〈사원〉 테이블 상태		
사원번호	이름	부서
10	김기혁	기획부
20	박인사	인사부
30	최재무	재무부

3. SQL 응용-SEC_03(DCL)

예제2) <사원> 테이블에서 '사원번호'가 30인 사원의 정보를 삭제하시오.

DELETE FROM 사원 WHERE 사원번호 = 30;

해설 : DELETE 명령을 수행한 후 COMMIT 명령을 수행하지 않았으므로 DELETE 명령으로 삭제된 레코드는 이후 ROLLBACK 명령으로 되돌릴 수 있다.

〈사원〉 테이블 상태		
사원번호	이름	부서
10	김기혁	기획부
20	박인사	인사부

예제3) SAVEPOINT 'S1'을 설정하고 '사원번호'가 20인 사원의 정보를 삭제하시오.

SAVEPOINT S1;

DELETE FROM 사원 WHERE 사원번호 = 20;

〈사원〉 테이블 상태		
사원번호	이름	부서
10	김기혁	기획부

예제4) SAVEPOINT 'S2'를 설정하고 '사원번호'가 10인 사원의 정보를 삭제하시오.

SAVEPOINT S2;

DELETE FROM 사원 WHERE 사원번호 = 10;

〈사원〉 테이블 상태		
사원번호	이름	부서

3. SQL 응용-SEC_03(DCL)

예제5) SAVEPOINT 'S2'까지 ROLLBACK을 수행하시오.

ROLLBACK TO S2;

해설 : ROLLBACK이 적용되는 시점을 'S2'로 지정했기 때문에 예제5)의 ROLLBACK에 의해 <사원> 테이블의 상태는 예제4)의 작업을 수행하기 전으로 되돌려진다.

〈사원〉 테이블 상태		
사원번호	이름	부서
10	김기혁	기획부

예제6) SAVEPOINT 'S1'까지 ROLLBACK을 수행하시오.

ROLLBACK TO S1;

〈사원〉 테이블 상태		
사원번호	이름	부서
10	김기혁	기획부
20	박인사	인사부

예제7) SAVEPOINT 없이 ROLLBACK을 수행하시오.

ROLLBACK;

〈사원〉 테이블 상태		
사원번호	이름	부서
10	김기혁	기획부
20	박인사	인사부
30	최재무	재무부

해설 : '사원번호'가 40인 사원의 정보를 삭제한 후 COMMIT을 수행했으므로 예제7)의 ROLLBACK이 적용되는 시점은 예제1)의 COMMIT 이후 새롭게 작업이 수행되는 예제2)의 작업부터이다.

SQL 응용 - SEC_03(DCL) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DCL)

1. SQL의 기능에 따른 분류 중에서 REVOKE문과 같이 데이터의 사용 권한을 관리하는데 사용하는 언어는?

- ① DDL(Data Definition Language)
- ② DML(Data Manipulation Language)
- ③ DCL(Data Control Language)
- ④ DUL(Data User Language)

DCL(Data Control Language, 데이터 제어어)

DCL(데이터 제어어)는 데이터의 보안, 무결성, 회복, 병행 제어 등을 정의하는데 사용하는 언어이다.

- DCL은 데이터베이스 관리자(DBA)가 데이터 관리를 목적으로 사용한다.
- DCL에는 GRANT, REVOKE, COMMIT, ROLLBACK, SAVEPOINT 가 있다.

2. 데이터 제어어(DCL)에 대한 설명으로 옳은 것은?

- ① ROLLBACK : 데이터의 보안과 무결성을 정의한다.
- ② COMMIT : 데이터베이스 사용자의 사용 권한을 취소한다.
- ③ GRANT : 데이터베이스 사용자의 사용 권한을 부여한다.

3. 트랜잭션의 실행이 실패하였음을 알리는 연산자로 트랜잭션이 수행한 결과를 원래의 상태로 원상 복구시키는 연산은?

- ① COMMIT 연산 ② BACKUP 연산
- ③ LOG 연산 ④ ROLLBACK 연산

COMMIT은 트랜잭션의 처리 성공, ROLLBACK은 트랜잭션의 실패로 취소하는 연산이다.

4. DBA가 사용자 PARK에게 테이블 [STUDENT]의 데이터를 갱신할 수 있는 시스템 권한을 부여하고자 하는 SQL문을 작성하고자 한다. 다음에 주어진 SQL문의 빈 칸을 알맞게 채운 것은?

SQL> GRANT __㉠__ __㉡__ STUDENT TO PARK;

- ① ㉠ INSERT ㉡ IN TO
- ② ㉠ ALTER ㉡ TO
- ③ ㉠ UPDATE ㉡ ON
- ④ ㉠ REPLACE ㉡ IN

데이터를 갱신하는 명령어는 UPDATE이다.

SQL 응용 - SEC_03(DCL) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DCL)

5. 사용자 X1에게 department 테이블에 대한 검색 권한을 회수 하는 명령은?

- ① delete select on department to X1;
- ② remove select on department from X1;
- ③ revoke select on department from X1;
- ④ grant select on department from X1;

테이블 및 속성에 대한 권한 부여 및 취소

- GRANT 권한_리스트 ON 테이블명 TO 사용자ID [WITH GRANT OPTION];

- 권한 종류 : ALL, SELECT, INSERT, DELETE, UPDATE, ALTER
WITH GRANT OPTION : 부여 받은 권한을 다른 사용자에게 다시 부여할 수 있는 권한을 부여하는 옵션

GRANT OPTION FOR : 다른 사용자에게 권한을 부여할 수 있는 권한을 취소함

CASCADE : 권한 취소 시 권한을 부여 받았던 사용자가 다른 사용자에게 부여한 권한도 연쇄적으로 취소가 됨

6. 사용자 'PARK'에게 테이블을 생성할 수 있는 권한을 부여하기 위한

7. SQL과 관련한 설명으로 틀린 것은?

- ① REVOKE 키워드를 사용하여 열 이름을 다시 부여할 수 있다.
- ② 데이터 정의어는 기본 테이블, 뷰 테이블 또는 인덱스 등을 생성, 변경, 제거하는데 사용되는 명령어이다.
- ③ DISTINCT를 활용하여 중복 값을 제거할 수 있다.
- ④ JOIN을 통해 여러 테이블의 레코드를 조합하여 표현할 수 있다.

데이터 조회 시 데이터 중복을 제거하기 위해서는 대표적으로 2가지 방법이 존재한다. **DISTINCT 키워드**를 사용하여 중복을 제거하는 방법과 **GROUP BY 절**을 사용하여 데이터 중복을 제거하는 방법이다.

또한, DISTINCT 키워드를 사용하여 데이터 중복을 제거할 때는 SELECT 절에 DISTINCT 키워드만 명시하면 되므로 쿼리문이 복잡하지 않고 간결하다. 그러나 DISTINCT 키워드를 사용하면 temp tablespace에 임시로 저장하고 작업하는 방식이라서 시스템에 조금 부하가 발생할 수도 있다. GROUP BY 절을 사용하여 데이터 중복을 제거할 때는 SELECT 절의 컬럼을 GROUP BY 절에도 동일하게 명시해야 하므로 쿼리문이 조금 더 길어질 수 있다. DISTINCT에 비해서 조금 성능이 좋다고 알려져 있지만, 크게 체감할 수 없다.

8. 다음 중, TCL(Transaction Control Language)에 속하지 않는 명령어는?

3. SQL 응용-SEC_04(DML)

1) DML(Data Manipulation Language, 데이터 조작어)의 개념

; DML(데이터 조작어)은 데이터베이스 사용자가 응용 프로그램이나 질의어를 통해 저장된 데이터를 실질적으로 관리하는데 사용되는 언어이다.

- DML은 데이터베이스 사용자와 데이터베이스 관리 시스템 간의 인터페이스를 제공한다.
- DML의 유형

명령문	기능
SELECT	테이블에서 튜플을 검색한다.
INSERT	테이블에 새로운 튜플을 삽입한다.
DELETE	테이블에서 튜플을 삭제한다.
UPDATE	테이블에서 튜플의 내용을 갱신한다.

3. SQL 응용-SEC_04(DML)

2) 삽입문(INSERT INTO~)

; 삽입문은 기본 테이블에 새로운 튜플을 삽입할 때 사용한다.

일반 형식

INSERT INTO 테이블명([속성명1, 속성명2, ...])

VALUES (데이터1, 데이터2, ...);

- 대응하는 속성과 데이터는 개수와 데이터 유형이 일치해야 한다.
- 기본 테이블의 모든 속성을 사용할 때는 속성명을 생략할 수 있다.
- SELECT문을 사용하여 다른 테이블의 검색 결과를 삽입할 수 있다.

3. SQL 응용-SEC_04(DML)

2) 삽입문(INSERT INTO~)

〈사원〉				
이름	부서	생일	주소	기본급
홍길동	기획	04/05/61	망원동	120
임꺽정	인터넷	01/09/69	성산동	80
황진이	편집	07/21/75	연희동	100
김선달	편집	10/22/73	망원동	90
성춘향	기획	02/20/64	망원동	100
장길산	편집	03/11/67	상암동	120
일지매	기획	04/29/78	합정동	110
강호동	인터넷	12/11/80		90

예제1) <사원> 테이블에 (이름 - 홍승현, 부서 - 인터넷)을 삽입하시오.

```
INSERT INTO 사원(이름, 부서) VALUES '홍승현', '인터넷');
```

예제2) <사원> 테이블에 (장보고, 기획, 05/03/73, 홍제동, 90)을 삽입하시오.

```
INSERT INTO 사원 VALUES ('장보고', '기획', #05/03/73#, '홍제동', 90);
```

날짜 데이터는 숫자로 취급하지만
' ' 또는 # #으로 묶어준다.

예제3) <사원> 테이블에 있는 편집부의 모든 튜플을 편집부원(이름, 생일, 주소, 기본급) 테이블에 삽입하시오.

```
INSERT INTO 편집부원(이름, 생일, 주소, 기본급) SELECT 이름, 생일, 주소, 기본급 FROM 사원  
WHERE 부서 = '편집';
```

3. SQL 응용-SEC_04(DML)

3) 삭제문(DELETE FROM~)

; 삭제문은 기본 테이블에 있는 튜플들 중에서 특정 튜플(행)을 삭제할 때 사용한다.

일반 형식

DELETE FROM 테이블명 [WHERE 조건];

- 모든 레코드를 삭제할 때는 WHERE절을 생략한다.
- 모든 레코드를 삭제하더라도 테이블 구조는 남아 있기 때문에 디스크에서 테이블을 완전히 제거하는 DROP과는 다르다.

예제1) <사원> 테이블에서 "임꺽정"에 대한 튜플을 삭제하시오.

DELETE FROM 사원 WHERE 이름 = '임꺽정';

예제2 <사원> 테이블에서 "인터넷" 부서에 대한 모든 튜플을 삭제하시오.

DELETE FROM 사원 WHERE 부서 = '인터넷';

예제3 <사원> 테이블의 모든 레코드를 삭제하시오.

DELETE FROM 사원;

3. SQL 응용-SEC_04(DML)

4) 갱신문(UPDATE~ SET~)

; 갱신문은 기본 테이블에 있는 튜플들 중에서 특정 튜플의 내용을 변경할 때 사용한다.

일반 형식

UPDATE 테이블명 SET 속성명 = 데이터[, 속성명 = 데이터, ...]

[WHERE 조건];

예제1) <사원> 테이블에서 "홍길동"의 주소'를 "수색동"으로 수정하시오.

UPDATE 사원 SET 주소 = '수색동' WHERE 이름 = '홍길동'

예제2) <사원> 테이블에서 "황진이" 의 '부서'를 "기획부" 로 변경하고 '기본급'을 5만원 인상
시키시오.

UPDATE 사원 SET 부서 = '기획' 기본급 = 기본급 + 5

WHERE 이름 = '황진이';

데이터 조작문의 네 가지 유형

1. SELECT(검색): SELECT~ FROM~ WHERE~

2. INSERT(삽입) : INSERT INTO VALUES~

3. DELETE(삭제) : DELETE~ FROM~ WHERE~

4. UPDATE(변경) : UPDATE~ SET~ WHERE~

SQL 응용 - SEC_04(DML) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DML)

1. SQL의 기술이 옳지 않은 것은?

- ① SELECT... FROM ... WHERE...
- ② INSERT... ON... VALUES...
- ③ UPDATE... SET... WHERE...
- ④ DELETE... FROM... WHERE...

데이터 조작문의 네 가지 유형

- 1. SELECT(조회) : SELECT~ FROM~ WHERE~
- 2. INSERT(삽입) : INSERT INTO VALUES(...)
- 3. DELETE(삭제) : DELETE FROM WHERE~
- 4. UPDATE(변경) : UPDATE~ SET~ WHERE~

2. 다음 SQL 문장이 뜻하는 것은 무엇인가?

```
INSERT INTO 컴퓨터과테이블(학번, 이름, 학년)
SELECT 학번, 이름, 학년 FROM 학생테이블
WHERE 학과 = '컴퓨터';
```

- ① 학생테이블에서 학과가 컴퓨터인 사람의 학번, 이름, 학년을 검색하라.
- ② 학생테이블에 학과가 컴퓨터인 사람의 학번, 이름, 학년을

3. SQL문에서 STUDENT(SNO, SNAME, YEAR, DEPT) 테이블에 학번 600, 성명 "홍길동", 학년 2학년인 학생 튜플을 삽입하는 명령으로 옳은 것은?(단, SNO는 학번, SNAME은 성명, YEAR는 학년, DEPT는 학생, 교수 구분 필드임)

- ① INSERT STUDENT INTO VALUES(600, '홍길동', 2);
 - ② INSERT FROM STUDENT VALUES(600, '홍길동', 2);
 - ③ INSERT INTO STUDENT(SNO, SNAME, YEAR) VALUES(600, '홍길동', 2);
 - ④ INSERT TO STUDENT(SNO, SNAME, YEAR) VALUES(600, '홍길동', 2);
- 삽입문(INSERT INTO~ VALUES(...));

삽입문은 기본 테이블에서 새로운 튜플을 삽입할 때 사용한다.
형식

```
INSERT INTO 테이블명([속성명1, 속성명2, ....])
VALUES(데이터1, 데이터2, ...);
```

- 대응하는 속성과 데이터는 개수와 데이터 유형이 일치해야 한다.
- 기본 테이블의 모든 속성을 사용할 때는 속성명을 생략할 수 있다.
- SELECT문을 사용하여 다른 테이블의 검색 결과를 삽입할 수 있다.

4. SQL에서 DELETE 명령에 대한 설명으로 옳지 않은 것은?

- ① 테이블의 행을 삭제할 때 사용한다.

SQL 응용 - SEC_04(DML) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DML)

5. 다음 SQL문에서 빈 칸에 들어갈 내용으로 옳은 것은?

UPDATE 회원 () 전화번호 = '010-14'

WHERE 회원번호 = 'N4';

① FROM ② SET

③ INTO ④ TO

갱신문(UPDATE~ SET~ WHERE~)

갱신문은 기본 테이블에 있는 튜플들 중에서 특정 튜플의 값을 변경하고자 할 때 사용한다.

형식

UPDATE 테이블명 SET 속성명 = 데이터[, 속성명 =
데이터, ..]
[WHERE 조건];

6. 다음 질의어를 SQL 문장으로 바르게 나타낸 것은? (단, 학생 테이블에 학번, 이름, 학과의 열이 있다고 가정한다.)

학번이 100, 이름이 홍길동, 학과가 컴퓨터인 학생을 학생 테이블에 삽입하라.

① UPDATE 학생 SET 학번 = 100, 이름 = '홍길동',

7. 다음 중 SQL의 INSERT 명령어에 대한 설명으로 옳지 않은 것은?

① 명령 하나로 동시에 두 개 이상의 테이블에 각각 삽입할 수 있다.

② 부속 질의어를 사용할 수 있다.

③ 하나의 INSERT 명령어를 사용하여 여러 개의 튜플을 삽입할 수도 있다.

④ NULL 값도 삽입할 수 있다.

명령 하나로 한 개의 테이블에만 삽입시킬 수 있다.

8. 다음 문장을 만족하는 SQL 문장은?

학번이 1000번인 학생을 학생 테이블에서 삭제하시오.

① DELETE FROM 학생 WHERE 학번 = 1000;

② DELETE FROM 학생 IF 학번 = 1000;

③ SELECT * FROM 학생 WHERE 학번 = 1000;

④ SELECT * FROM 학생 CONDITION 학번 = 1000;

문제의 지문을 구문절 별로 구분하면 SQL 문은 아래와 같다.

- '학생'테이블에서 삭제하시오 : DELETE FROM 학생

- '학번'이 1000번인 학생을 대상으로 하시오 : WHERE 학번 = 1000;

SQL 응용 - SEC_04(DML) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DML)

9. SQL의 UPDATE 명령어에 대한 설명으로 옳지 않은 것은?

- ① 하나 또는 그 이상의 튜플의 속성값을 변경하는 데 사용된다.
 - ② 테이블에서 수정할 튜플을 선택하기 위해 WHERE절이 사용된다.
 - ③ 변경할 속성과 그들의 새로운 값을 명시하기 위해 AS절이 사용된다.
 - ④ 참조 무결성 제약이 존재하는 경우에는 기본키 값을 변경하는 경우 그 변경이 외래키 값에 영향을 미칠 수 있다.
- 변경할 속성과 그들의 새로운 값을 명시하기 위해서는 AS가 아니라 SET절이 사용이 되어야 한다.

10. 다음 SQL문을 올바르게 설명한 것은?

```
UPDATE STUDENT  
SET SCORE = SCORE + 10  
WHERE SNAME = 'LEE';
```

- ① STUDENT 테이블에서 SNAME이 'LEE'인 모든 튜플의 SCORE 속성에 10을 더한다.
- ② STUDENT 테이블에서 SNAME이 'LEE'인 첫 번째 튜플의 SCORE 속성에 10을 더한다.

11. 다음 SQL문 중에서 구문적 오류가 있는 것은?

- ① DELETE FROM STUDENT, ENROL WHERE SNO = 100;
 - ② INSERT INTO STUDENT(SNO, SNAME, YEAR) VALUES(100, '홍길동', 4);
 - ③ INSERT INTO COMPUTER(SNO, SNAME, YEAR) SELECT SNO, SNAME, YEAR FROM STUDENT WHERE DEPT = 'CE';
 - ④ UPDATE STUDENT SET DEPT = (SELECT DEPT FROM COURSE WHERE CNO='C123')
WHERE YEAR = 4;
- 명령 하나로 한 개의 테이블에만 삭제를 하거나 삽입을 하거나 갱신을 할 수가 있다.

3. SQL 응용-SEC_05(DML-SELECT-1)

1) 일반 형식

```
SELECT [PREDICATE] [테이블명.]속성명 [AS 별칭][, [테이블명.]속성명, ...]  
    [, 그룹함수(속성명) [AS 별칭]]  
    [, Window 함수 OVER (PARTITION BY 속성명1, 속성명2, ...  
        ORDER BY 속성명3, 속성명4, ...)]  
  
FROM 테이블명[, 테이블명, ...]  
  
[WHERE 조건]  
[GROUP BY 속성명 속성명, ...]  
[HAVING 조건]  
[ORDER BY 속성명 [ASC DESC]];
```

● SELECT절

- PREDICATE : 불러올 튜플 수를 제한할 명령어를 기술한다.

- ▶ ALL : 모든 튜플을 검색할 때 지정하는 것으로, 주로 생략한다.
- ▶ DISTINCT : 중복된 튜플이 있으면 그 중 첫 번째 한 개만 검색한다.
- ▶ DISTINCTROW : 중복된 튜플을 제거하고 한 개만 검색하지만 선택된 속성의 값이 아닌, 튜플 전체를 대상으로 한다.

3. SQL 응용-SEC_05(DML-SELECT-1)

1) 일반 형식

- SELECT절

- 속성명 : 검색하여 불러올 속성(열) 또는 속성을 이용한 수식을 지정한다.
 - ▶ 기본 테이블을 구성하는 모든 속성을 지정할 때는 '*'를 기술한다.
 - ▶ 두 개 이상의 테이블을 대상으로 검색할 때는 '테이블명.속성명'으로 표현한다.
- AS : 속성 및 연산의 이름을 다른 제목(별칭)으로 표시하기 위해 사용된다.

- FROM절 : 질의에 의해 검색될 데이터들을 포함하는 테이블명을 기술한다.

- WHERE : 검색할 조건을 기술한다.

- ORDER BY절 : 특정 속성을 기준으로 정렬하여 검색할 때 사용한다.

- 속성명 : 정렬의 기준이 되는 속성명을 기술한다.
- [ASC DESC] : 정렬 방식으로서 'ASC'는 오름차순, 'DESC'는 내림차순이다. 생략하면 오름차순으로 지정된다.

3. SQL 응용-SEC_05(DML-SELECT-1)

1) 일반 형식

- 조건 연산자/연산자 우선순위

- 조건 연산자

- ▶ 비교 연산자

연산자	=	<>	>	<	>=	<=
의미	같다	같지 않다	크다	작다	크거나 같다	작거나 같다

- ▶ 논리 연산자 : NOT, AND, OR

- ▶ LIKE 연산자 : 대표 문자를 이용해 지정된 속성의 값이 문자 패턴과 일치하는 튜플을 검색하기 위해 사용된다.

대표 문자	%	_	#
의미	모든 문자를 대표함	문자 하나를 대표함	숫자 하나를 대표함

- 연산자 우선순위

종류	연산자	우선순위
산술 연산자	×, /, +, -	왼쪽에서 오른쪽으로 갈수록 낮아집니다.
관계 연산자	=, <, >, >=, <=	모두 같습니다.
^{21.8} 논리 연산자	NOT, AND, OR	왼쪽에서 오른쪽으로 갈수록 낮아집니다.

※ 산술, 관계, 논리 연산자가 함께 사용되었을 때는
산술 > 관계 > 논리 연산자 순서로 연산자 우선순위가
정해진다.

3. SQL 응용-SEC_05(DML-SELECT-1)

1) 일반 형식

● 다음과 같은 기본 테이블에 대해 다음 예제의 결과를 확인하시오.

〈사원〉				
이름	부서	생일	주소	기본급
홍길동	기획	04/05/61	망원동	120
임꺽정	인터넷	01/09/69	서교동	80
황진이	편집	07/21/75	합정동	100
김선달	편집	10/22/73	망원동	90
성춘향	기획	02/20/64	대흥동	100
장길산	편집	03/11/67	상암동	120
일지매	기획	04/29/78	연남동	110
강건달	인터넷	12/11/80		90

〈여가활동〉		
이름	취미	경력
김선달	당구	10
성춘향	나이트댄스	5
일지매	태권	15
임꺽정	씨름	8

2) 기본 검색

; SELECT 절에 원하는 속성을 지정하여 검색한다.

예제1) <사원> 테이블의 모든 튜플을 검색하시오.

- SELECT * FROM 사원;
- SELECT 사원.* FROM 사원;
- SELECT 이름, 부서, 생일, 주소, 기본급 FROM 사원;
- SELECT 사원.이름, 사원.부서, 사원.생일, 사원.주소, 사원.기본급 FROM 사원;

※ 위의 SQL은 모두 보기에 주어진 <사원> 테이블 전체를 그대로 출력한다.

〈결과〉				
이름	부서	생일	주소	기본급
홍길동	기획	04/05/61	망원동	120
임꺽정	인터넷	01/09/69	서교동	80
황진이	편집	07/21/75	합정동	100
김선달	편집	10/22/73	망원동	90
성춘향	기획	02/20/64	대흥동	100
장길산	편집	03/11/67	상암동	120
일지매	기획	04/29/78	연남동	110
강건달	인터넷	12/11/80		90

3. SQL 응용-SEC_05(DML-SELECT-1)

2) 기본 검색

예제2) <사원> 에서 '주소'만 검색하되 같은 '주소'는 한 번만 출력하시오.

- **SELECT DISTINCT 주소 FROM 사원;**

〈결과〉
주소
대흥동
망원동
상암동
서교동
연남동
합정동

예제3) <사원> 테이블에서 '기본급'에 특별수당 10을 더한 월급을 "XX부서의 XXX의 월급 XXX" 형태로 출력하시오.

- **SELECT 부서 + '부서의' AS 부서2, 이름 + '의 월급' AS 이름2, 기본급 + 10 AS 기본급2 FROM 사원;**

부서2	이름2	기본급2
기획부서의	홍길동의 월급	130
인터넷부서의	임꺽정의 월급	90
편집부서의	황진이의 월급	110
편집부서의	김선달의 월급	100
기획부서의	성춘향의 월급	110
편집부서의	장길산의 월급	130
기획부서의	일지매의 월급	120
인터넷부서의	강건달의 월급	100

부서 + "부서의" AS 부서2
'부서'에 '부서의'를 연결하여 표시하되, '부서'라는 속성 으로
표시한다.

3. SQL 응용-SEC_05(DML-SELECT-1)

3) 조건 지정 검색

; WHERE 절에 조건을 지정하여 조건에 만족하는 튜플만 검색한다.

예제1) <사원> 테이블에서 '기획'부의 모든 튜플을 검색하시오.

- **SELECT * FROM 사원 WHERE 부서 = '기획';**

〈결과〉				
이름	부서	생일	주소	기본급
홍길동	기획	04/05/61	망원동	120
성춘향	기획	02/20/64	대흥동	100
일지매	기획	04/29/78	연남동	110

예제2) <사원> 테이블에서 "기획" 부서에 근무하면서 "대흥동"에 사는 사람의 튜플을 검색하시오.

- **SELECT * FROM 사원 WHERE 부서 = '기획' AND 주소 = '대흥동';**

〈결과〉				
이름	부서	생일	주소	기본급
성춘향	기획	02/20/64	대흥동	100

예제3) <사원> 테이블에서 '부서'가 "기획"이거나 "인터넷"인 튜플을 검색하시오.

- **SELECT * FROM 사원 WHERE 부서 = '기획' OR 부서 = '인터넷';**
- **SELECT * FROM 사원 WHERE 부서 IN('기획', '인터넷');**

여기서 IN은 포함된 튜플을 의미한다.

〈결과〉				
이름	부서	생일	주소	기본급
홍길동	기획	04/05/61	망원동	120
임꺽정	인터넷	01/09/69	서교동	80
성춘향	기획	02/20/64	대흥동	100
일지매	기획	04/29/78	연남동	110
강건달	인터넷	12/11/80		90

3. SQL 응용-SEC_05(DML-SELECT-1)

3) 조건 지정 검색

예제4) <사원> 테이블에서 성이 '김'인 사람의 튜플을 검색하시오.

- **SELECT * FROM 사원 WHERE 이름 LIKE "김%";**

〈결과〉				
이름	부서	생일	주소	기본급
김선달	편집	10/22/73	망원동	90

예제4) <사원> 테이블에서 '생일'이 '01/01/69'에서 '12/31/73' 사이인 튜플을 검색하시오.

- **SELECT * FROM 사원 WHERE 생일 BETWEEN #01/01/69# AND #12/31/73#;**

〈결과〉				
이름	부서	생일	주소	기본급
임꺽정	인터넷	01/09/69	서교동	80
김선달	편집	10/22/73	망원동	90

예제6) <사원> 테이블에서 '주소'가 NULL인 튜플을 검색하시오.

- **SELECT * FROM 사원 WHERE 주소 IS NULL;**

〈결과〉				
이름	부서	생일	주소	기본급
강건달	인터넷	12/11/80		90

NULL이 아닌 값을 검색할 때는 IS NOT NULL을 사용한다.

<사원> 테이블에서 주소가 NULL이 아닌 튜플 검색

SELECT * FROM 사원 WHERE 주소 IS NOT NULL;

3. SQL 응용-SEC_05(DML-SELECT-1)

4) 정렬 검색

; ORDER BY 절에 특정 속성을 지정하여 지정된 속성으로 자료를 정렬하여 검색한다.

예제1) <사원> 테이블에서 '주소'를 기준으로 내림차순 정렬시켜 상위 2개 튜플만 검색하시오.

- SELECT TOP 2 * FROM 사원 ORDER BY 주소 DESC;

〈결과〉				
이름	부서	생일	주소	기본급
황진이	편집	07/21/75	합정동	100
일지매	기획	04/29/78	연남동	110

예제2) <사원> 테이블에서 '부서'를 기준으로 오름차순 정렬하고, 같은 '부서'에 대해서는 '이름'을 기준으로 내림차순 정렬시켜서 검색하시오.

- SELECT * FROM 사원 ORDER BY 부서 ASC, 이름 DESC;

〈결과〉				
이름	부서	생일	주소	기본급
홍길동	기획	04/05/61	망원동	120
일지매	기획	04/29/78	연남동	110
성춘향	기획	02/20/64	대흥동	100
임꺽정	인터넷	01/09/69	서교동	80
강건달	인터넷	12/11/80		90
황진이	편집	07/21/75	합정동	100
장길산	편집	03/11/67	상암동	120
김선달	편집	10/22/73	망원동	90

3. SQL 응용-SEC_05(DML-SELECT-1)

5) 하위 질의

; 하위 질의는 조건절에 주어진 질의를 먼저 수행하여 그 검색 결과를 조건절의 피연산자로 사용한다.

예제1) '취미'가 "나이트댄스"인 사원의 '이름'과 '주소'를 검색하시오.

- **SELECT 이름, 주소 FROM 사원 WHERE 이름 = (SELECT 이름 FROM 여가활동 WHERE 취미 = '나이트댄스');**

〈결과〉	
이름	주소
성춘향	대흥동

먼저 'SELECT 이름 FROM 여가활동 WHERE 취미 = '나이트댄스'를 수행하여 <여가활동> 테이블에서 '성춘향'을 찾는다. 그런 다음 하위 질의에 해당하는 피연산자의 자리에 '성춘향'을 대입하면 질의 문은 "SELECT 이름, 주소 FROM 사원 WHERE 이름 = '성춘향' 같다.

예제2) 취미활동을 하지 않는 직원들을 검색하시오.

- **SELECT * FROM 사원 WHERE 이름 NOT IN(SELECT 이름 FROM 여가활동);**

〈결과〉				
이름	부서	생일	주소	기본급
홍길동	기획	04/05/61	망원동	120
황진이	편집	07/21/75	합정동	100
장길산	편집	03/11/67	상암동	120
강건달	인터넷	12/11/80		90

Not In()은 포함되지 않는 데이터를 의미한다. 즉 <사원> 테이블에서 모든 자료를 검색하는데, <여가활동> 테이블에 '이름'이 있는 자료는 제외하고 검색한다.

3. SQL 응용-SEC_05(DML-SELECT-1)

5) 하위 질의

예제3) 취미활동을 하는 사원들의 부서를 검색하시오.

- **SELECT 부서 FROM 사원 WHERE EXISTS(SELECT 이름 FROM 여가활동 WHERE 여가활동.이름 = 사원.이름);**

〈결과〉	
부서	
인터넷	
편집	
기획	
기획	

EXISTS()는 하위 질의로 검색된 결과가 존재하는지 확인할 때 사용한다. 즉 <사원> 테이블의 '이름'이 <여가활동> 테이블의 '이름'에도 있는지 확인하는 것이다.

6) 복수 테이블 검색 질의

; 여러 테이블을 대상으로 검색을 수행한다.

예제) '경력'이 10년 이상인 사원의 '이름', '부서', '취미', '경력'을 검색하시오.

- **SELECT 사원.이름, 사원.부서, 여가활동.취미, 여가활동.경력 FROM 사원, 여가활동
WHERE 여가활동.경력 >= 10
AND 사원.이름 = 여가활동.이름;**

〈결과〉			
이름	부서	취미	경력
김선달	편집	당구	10
일지매	기획	태권	15

SQL 응용 - SEC_05(DML-SELECT-1) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DML-SELECT-1)

1. player 테이블에는 player_name, team_id, height 컬럼이 존재한다. 아래 SQL문에서 문법적 오류가 있는 부분은?

- (1) SELECT player_name, height
- (2) FROM player
- (3) WHERE team_id = "Korea"
- (4) AND height BETWEEN 170 OR 180;

- ① (1) ② (2)
- ③ (3) ④ (4)

BETWEEN 연산자는 AND와 짝을 이루어야 하므로 (4)번을 AND height BETWEEN 170 AND 180이라고 기술해야 된다. 즉 (3)부터는 조건문이므로 team_id가 Korea이고 height가 170이상~180 사이인 데이터를 검색하는 SQL문이 되는 것이다.

2. SQL 문에서 SELECT에 대한 설명으로 옳지 않은 것은?

- ① FROM 절에는 질의에 의해 검색될 데이터들을 포함하는 테이블명을 기술한다.
- ② 검색 결과에 중복되는 레코드를 없애기 위해서는 WHERE 절에 'DISTINCT' 키워드를 사용한다.

3. STUDENT 테이블에 독일어과 학생 50명, 중국어과 학생 30명, 영어 영문학과 학생 50명의 정보가 저장되어 있을 때, 다음 두 SQL문의 실행 결과 튜플 수는? (단, DEPT 컬럼은 학과명)

- ㉠ SELECT DEPT FROM STUDENT;
- ㉡ SELECT DISTINCT DEPT FROM STUDENT;

- ① ㉠ 3, ㉡ 3 ② ㉠ 50, ㉡ 3
- ③ ㉠ 130, ㉡ 3 ④ ㉠ 130, ㉡ 1300

㉠ <STUDENT> 테이블에서 DEPT 속성을 검색한다. 총 130개의 튜플이 저장되어 있고 검색 조건이 없으므로 출력하는 튜플의 수는 130이다.

㉡ <STUDENT> 테이블에서 DEPT 속성을 검색하는데 DISTINCT 키워드에 의해서 중복된 결과는 처음의 한 개만 검색에 포함한다. 독일어과 50개 튜플을 DEPT 속성의 값이 같기 때문에 1개, 중국어과 30개 튜플의 DEPT 속성의 값이 같으므로 1개, 영어 영문학과 50개 튜플의 DEPT 속성의 값이 같으므로 1개를 검색에 포함시키므로 총 3개의 튜플이 검색 된다.

4. 관계 데이터베이스인 테이블 R1에 대한 아래 SQL문의 실행 결과로 옳은 것은?

[R1]				
학번	이름	성별	학과	점수

SQL 응용 - SEC_05(DML-SELECT-1) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DML-SELECT-1)

5. 다음 SQL문에서 사용된 BETWEEN 연산의 의미와 동일한 것은?

```
SELECT * FROM 성적
WHERE (점수 BETWEEN 90 AND 95)
AND 학과 = '컴퓨터공학과';
```

- ① 점수 >= 90 AND 점수 <= 95
- ② 점수 > 90 AND 점수 < 95
- ③ 점수 > 90 AND 점수 <=95
- ④ 점수 >= 90 AND 점수 < 95

점수 BETWEEN 90 AND 95는 점수가 90점 이상 95이하라는 의미이다.

6. 다음 SQL문의 실행 결과는?

```
SELECT 가격 FROM 도서가격
WHERE 책번호 = (SELECT 책번호 FROM 도서
                책명 = '자료구조');
```

[도서]	
책번호	책명
111	운영체제
222	자료구조
333	컴퓨터구조

[도서가격]	
책번호	가격
111	20,000
222	25,000
333	10,000
444	15,000

7. 학적 테이블에서 전화번호가 Null 값이 아닌 학생명을 모두 검색할 때, SQL 구문으로 옳은 것은?

- ① SELECT 학생명 FROM 학적 WHERE 전화번호 DON'T NULL;
 - ② SELECT 학생명 FROM 학적 WHERE 전화번호 != NOT NULL;
 - ③ SELECT 학생명 FROM 학적 WHERE 전화번호 IS NOT NULL;
 - ④ SELECT 학생명 FROM 학적 WHERE 전화번호 IS NULL;
- NULL 값일 경우는 IS NULL, NULL값이 아닐 경우는 IS NOT NULL이다.

8. 다음 테이블을 보고 강남지점의 판매량이 많은 제품부터 출력되도록 할 때 다음 중 가장 적절한 SQL 구문은? (단, 출력은 제품명과 판매량이 출력되도록 한다.)

[푸드] 테이블		
지점명	제품명	판매량
강남지점	비빔밥	500
강북지점	도시락	300
강남지점	도시락	200
강남지점	미역국	550
수원지점	비빔밥	600
인천지점	비빔밥	800
강남지점	잡채밥	250

- ① SELECT 제품명, 판매량 FROM 푸드 ORDER BY 판매량 ASC
- ② SELECT 제품명, 판매량 FROM 푸드 ORDER BY 판매량 DESC

SQL 응용 - SEC_05(DML-SELECT-1) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DML-SELECT-1)

9. 다음 [조건]에 부합하는 SQL문을 작성하고자 할 때, [SQL문]의 빈칸에 들어갈 내용으로 옳은 것은? (단, '팀코드' 및 '이름'은 속성이며, '직원'은 테이블이다.)

[조건]

이름이 '정도일'인 팀원이 소속된 팀코드를 이용하여 해당팀에 소속된 팀원들의 이름을 출력하는 SQL문 작성

[SQL문]

SELECT 이름 FROM 직원
WHERE 팀코드 = ();

- ① WHERE 이름 = '정도일'
 - ② SELECT 팀코드 FROM 이름 WHERE 직원 = '정도일'
 - ③ WHERE 직원 = '정도일'
 - ④ SELECT 팀코드 FROM 직원 WHERE 이름 = '정도일'
1. 서브쿼리 SELECT 팀코드 FROM 직원 WHERE 이름 = '정도일'; : '직원' 테이블에서 '이름'속성의 값이 '정도일'과 같은 레코드의 '팀코드'속성의 값을 검색하여 반환한다.
2. 'SELECT 이름 FROM 직원 WHERE 팀코드 = 반환값(팀코드); :

11. SQL의 논리 연산자가 아닌 것은?

- ① AND ② OTHER
- ③ OR ④ NOT

논리 연산자 세 가지는 AND, OR, NOT 이다. 기억하도록 하자.

12. 결과 값이 아래와 같을 때 SQL 질의로 옳은 것은?

[공급자] Table		
공급자번호	공급자명	위치
16	대신공업사	수원
27	삼진사	서울
39	삼양사	인천
62	진아공업사	대전
70	신촌상사	서울

[결과]		
공급자번호	공급자명	위치
16	대신공업사	수원
70	신촌상사	서울

- ① SELECT * FROM 공급자 WHERE 공급자명 LIKE '%신%';
 - ② SELECT * FROM 공급자 WHERE 공급자명 LIKE '대%';
 - ③ SELECT * FROM 공급자 WHERE 공급자명 LIKE '%사';
 - ④ SELECT * FROM 공급자 WHERE 공급자명 IS NOT NULL;
- LIKE '%신%' : 공급자명에 "신"이 포함된 레코드를 출력
LIKE '대%' : 공급자명이 "대"로 시작하는 레코드를 출력
LIKE '%사' : 공급자명이 "사"로 끝나는 레코드를 출력

SQL 응용 - SEC_05(DML-SELECT-1) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DML-SELECT-1)

13. SQL에서 조건문으로 기술할 수 있는 구문은?

- ① SELECT ② FROM
③ DISTINCT ④ LIKE

LIKE : (WHERE 속성 LIKE "부분 문자%" 형식을 사용하여 지정된 속성에서 부분 문자가 들어있는 튜플을 대상으로 검색시킬 조건을 표현할 때 사용한다.

% : 부분 문자 포함 레코드 출력, 모든 문자를 대표함

_ : 문자 하나를 대표함

: 숫자 하나를 대표함

14. 다음 SQL문을 올바르게 설명한 것은?

```
SELECT * FROM STUDENT  
WHERE SNAME LIKE '홍%';
```

- ① SNAME이 '홍'씨로 시작하면 삭제한다.
② SNAME이 '홍'씨로 시작되는 튜플을 찾는다.
③ SNAME이 '홍'씨로 시작하면 0으로 치환한다.
④ SNAME이 '홍'씨로 시작되는 튜플을 삭제한다.

LIKE는 문자열의 패턴을 비교할 때 사용하는 일종의 연산자이고,

15. 다음 SQL문의 실행 결과로 생성되는 튜플 수는?

SELECT 급여 FROM 사원;

[사원] 테이블			
사원ID	사원명	급여	부서ID
101	박철수	30000	1
102	한나라	35000	2
103	김감동	40000	3
104	이구수	35000	2
105	최초록	40000	3

- ① 1 ② 3
③ 4 ④ 5

WHERE문이 없으므로 <사원>테이블에서 '급여'필드의 전체 레코드를 조회한다. 하여 5개의 튜플이 출력이 된다.

16. 입교 지원 현황을 조회하고자 할 때 다음 예시된 SQL 구문으로 알 수 없는 것은?

```
SELECT 지원, 지원학과, 전화번호  
FROM 지원자  
WHERE 점수 > 59  
ORDER BY 지원학과, 점수 DESC;
```

- ① 지원자 테이블을 검색한다.
② 점수가 60점 이상인 지원자만을 검색한다.

SQL 응용 - SEC_05(DML-SELECT-1) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DML-SELECT-1)

17. 다음은 질의 수행 결과를 정렬하는 방법에 대한 설명이다.

옳지 않은 것은?

- ① ORDER BY절을 사용한다.
- ② 기본 정렬 방식은 내림차순이다.
- ③ ASC 키워드를 사용하여 사용자가 오름차순 정렬 방식을 지정할 수 있다.
- ④ DESC 키워드를 사용하여 사용자가 내림차순 정렬 방식을 지정 할 수 있다.

정렬

질의를 수행하여 얻은 튜플을 사용자가 정렬할 수 있도록 한다.

ORDER BY 절을 사용한다. ASC, DESC 키워드를 사용하여 사용자가 각각 오름차순 정렬, 내림차순 정렬 방식을 지정할 수 있다.

기본 정렬 방식은 ASC를 생략해도 오름차순 정렬이며, DESC는 반드시 명시적으로 적어줘야 하며 내림차순 정렬이다.

18. 다음의 관계형 데이터베이스에서 "질의 : 제일은행(Company) 직원들의 이름(Name)과 그들이 사는 도시(City)를 찾아라."로 작성한 것의 ㉠, ㉡, ㉢의 위치에 적당한 내용은?

3. SQL 응용-SEC_06(DML-SELECT-2)

1) 일반 형식

```
SELECT [PREDICATE] [테이블명.]속성명 [AS 별칭][, [테이블명.]속성명, ...]  
      [, 그룹함수(속성명) [AS 별칭]]  
      [, Window함수 OVER (PARTITION BY 속성명1, 속성명2, ...  
                          ORDER BY 속성명3, 속성명4, ...)]  
FROM 테이블명[, 테이블명, ...]  
[WHERE 조건]  
[GROUP BY 속성명 속성명, ...]  
[HAVING 조건]  
[ORDER BY 속성명 [ASC | DESC]];
```

- 그룹함수 : GROUP BY절에 지정된 그룹별로 속성의 값을 집계할 함수를 기술한다.
- WINDOW 함수 : GROUP BY절을 이용하지 않고 속성의 값을 집계할 함수를 기술한다.
 - PARTITION BY : WINDOW 함수가 적용될 범위로 사용할 속성을 지정한다.
 - ORDER BY : PARTITION 안에서 정렬 기준으로 사용할 속성을 지정한다.
- GROUP BY절 : 특정 속성을 기준으로 그룹화하여 검색할 때 사용한다. 일반적으로 GROUP BY절은 그룹 함수와 함께 사용된다.

3. SQL 응용-SEC_06(DML-SELECT-2)

1) 일반 형식

- HAVING절 : GROUP BY와 함께 사용되며, 그룹에 대한 조건을 지정한다.
- 그룹 함수 / WINDOW 함수
 - 그룹 함수
 - ; GROUP BY절에 지정된 그룹별로 속성의 값을 집계할 때 사용된다.
 - > COUNT(속성명) : 그룹별 튜플 수를 구하는 함수
 - > SUM(속성명) : 그룹별 합계를 구하는 함수
 - > AVG(속성명) : 그룹별 평균을 구하는 함수
 - > MAX(속성명) : 그룹별 최대값을 구하는 함수
 - > MIN(속성명) : 그룹별 최소값을 구하는 함수
 - > STDDEV(속성명) : 그룹별 표준편차를 구하는 함수
 - > VARIANCE(속성명) : 그룹별 분산을 구하는 함수
 - > ROLLUP(속성명, 속성명,)
 - 인수로 주어진 속성을 대상으로 그룹별 소계, 총 합계를 구하는 함수이다.
 - 속성의 개수가 n개이면, n + 1 레벨까지, 하위 레벨에서 상위 레벨 순으로 데이터가 집계된다.

3. SQL 응용-SEC_06(DML-SELECT-2)

1) 일반 형식

- 그룹 함수 / WINDOW 함수

- 그룹 함수

- > CUBE(속성명, 속성명, ...)

- ROLLUP과 유사한 형태이나 CUBE는 인수로 주어진 속성을 대상으로 모든 조합의 그룹별 소계, 총 합계를 구한다.

- 속성의 개수가 n 개이면, 2^n 레벨까지, 상위 레벨에서 하위 레벨 순으로 데이터가 집계된다.

- WINDOW 함수

- GROUP BY절을 이용하지 않고 함수의 인수로 지정한 속성을 범위로 하여 속성의 값을 집계한다.

- 함수의 인수로 지정한 속성이 대상 레코드의 범위가 되는데, 이를 윈도우(WINDOW)라고 부른다.

- WINDOW 함수 종류

- > ROW_NUMBER() : 윈도우 별로 각 레코드에 대한 일련 번호를 반환한다.

- > RANK() : 윈도우 별로 순위를 반환하며, 공동 순위를 반영한다.

- > DENSE_RANK() : 윈도우 별로 순위를 반환하며, 공동 순위를 무시하고 순위를 부여한다.

3. SQL 응용-SEC_06(DML-SELECT-2)

2) WINDOW 함수 이용 검색

; GROUP BY절을 이용하지 않고 함수의 인수로 지정한 속성을 범위로 하여 속성의 값을 집계한다.

〈상여금〉			
부서	이름	상여내역	상여금
기획	홍길동	연장근무	100
기획	일지매	연장근무	100
기획	최준호	야간근무	120
기획	장길산	특별근무	90
인터넷	강건달	특별근무	90
인터넷	서국현	특별근무	90
인터넷	박인식	연장근무	30
편집	김선달	특별근무	80
편집	황종근	연장근무	40
편집	성춘향	야간근무	80
편집	임꺽정	야간근무	80
편집	황진이	야간근무	50

〈결과〉		
상여내역	상여금	NO
야간근무	120	1
야간근무	80	2
야간근무	80	3
야간근무	50	4
연장근무	100	1
연장근무	100	2
연장근무	40	3
연장근무	30	4
특별근무	90	1
특별근무	90	2
특별근무	90	3
특별근무	80	4

예제1) <상여금> 테이블에서 '상여내역'별로 '상여금'에 대한 일련 번호를 구하시오.(단 순서는 내림차순이며 속성명은 'NO'로 할 것)

SELECT 상여내역, 상여금,

ROW_NUMBER() OVER(PARTITION BY 상여내역 ORDER BY 상여금 DESC) AS NO

FROM 상여금;

3. SQL 응용-SEC_06(DML-SELECT-2)

2) WINDOW 함수 이용 검색

예제2) <상여금> 테이블에서 '상여내역'별로 '상여금'에 대한 순위를 구하시오.(단, 순서는 내림차순이며, 속성명은 '상여금 순위'로 하고, RANK() 함수를 이용할 것)

SELECT 상여내역, 상여금,

RANK() OVER(PARTITION BY 상여내역 ORDER BY 상여금 DESC) AS 상여금순위

FROM 상여금;

〈상여금〉			
부서	이름	상여내역	상여금
기획	홍길동	연장근무	100
기획	일지매	연장근무	100
기획	최준호	야간근무	120
기획	장길산	특별근무	90
인터넷	강건달	특별근무	90
인터넷	서국현	특별근무	90
인터넷	박인식	연장근무	30
편집	김선달	특별근무	80
편집	황중근	연장근무	40
편집	성춘향	야간근무	80
편집	임꺽정	야간근무	80
편집	황진이	야간근무	50

〈결과〉		
상여내역	상여금	상여금순위
야간근무	120	1
야간근무	80	2
야간근무	80	2
야간근무	50	4
연장근무	100	1
연장근무	100	1
연장근무	40	3
연장근무	30	4
특별근무	90	1
특별근무	90	1
특별근무	90	1
특별근무	80	4

3. SQL 응용-SEC_06(DML-SELECT-2)

3) 그룹 지정 검색

; GROUP BY절에 지정한 속성을 기준으로 자료를 그룹화하여 검색한다.

예제1) <상여금> 테이블에서 '부서별 상여금'의 평균을 구하시오.

SELECT 부서, AVG(상여금) AS 평균

FROM 상여금

GROUP BY 부서;

〈상여금〉			
부서	이름	상여내역	상여금
기획	홍길동	연장근무	100
기획	일지매	연장근무	100
기획	최준호	야간근무	120
기획	장길산	특별근무	90
인터넷	강건달	특별근무	90
인터넷	서국현	특별근무	90
인터넷	박인식	연장근무	30
편집	김선달	특별근무	80
편집	황중근	연장근무	40
편집	성춘향	야간근무	80
편집	임꺽정	야간근무	80
편집	황진이	야간근무	50

〈결과〉	
부서	평균
기획	102.5
인터넷	70
편집	66

3. SQL 응용-SEC_06(DML-SELECT-2)

3) 그룹 지정 검색

예제2) <상여금> 테이블에서 부서별 튜플 수를 검색하시오.

```
SELECT 부서, COUNT(*) AS 사원수  
FROM 상여금  
GROUP BY 부서;
```

〈상여금〉			
부서	이름	상여내역	상여금
기획	홍길동	연장근무	100
기획	일지매	연장근무	100
기획	최준호	야간근무	120
기획	장길산	특별근무	90
인터넷	강건달	특별근무	90
인터넷	서국현	특별근무	90
인터넷	박인식	연장근무	30
편집	김선달	특별근무	80
편집	황중근	연장근무	40
편집	성춘향	야간근무	80
편집	임꺽정	야간근무	80
편집	황진이	야간근무	50

〈결과〉	
부서	사원수
기획	4
인터넷	3
편집	5

3. SQL 응용-SEC_06(DML-SELECT-2)

3) 그룹 지정 검색

예제3) <상여금> 테이블에서 '상여금'이 100 이상인 사원이 2명 이상인 '부서'의 튜플 수를 구하시오.

SELECT 부서, COUNT(*) AS 사원수

FROM 상여금

WHERE 상여금 >= 100

GROUP BY 부서

HAVING COUNT(*) >= 2;

1. 'WHERE 상여금 >=100' 절에 의해서 '상여금' 이 100 이상인 자료만 검색 대상이 된다.
2. GROUP BY 부서 절에 의해서 '상여금' 이 100 이상인 자료에 대해서만 부서별로 그룹을 지정한다.
3. HAVING COUNT(*) >= 2 절에 의해서 부서의 인원이 2 이상인 부서의 인원만 검색한다.

〈상여금〉			
부서	이름	상여내역	상여금
기획	홍길동	연장근무	100
기획	일지매	연장근무	100
기획	최준호	야간근무	120
기획	장길산	특별근무	90
인터넷	강건달	특별근무	90
인터넷	서국현	특별근무	90
인터넷	박인식	연장근무	30
편집	김선달	특별근무	80
편집	황중근	연장근무	40
편집	성춘향	야간근무	80
편집	임꺽정	야간근무	80
편집	황진이	야간근무	50

〈결과〉	
부서	사원수
기획	3

3. SQL 응용-SEC_06(DML-SELECT-2)

3) 그룹 지정 검색

예제4) <상여금> 테이블에서 '부서', '상여내역', 그리고 '상여금'에 대해 부서별 상여 내역별 소계와 전체 합계를 검색하시오.(단, 속성명은 '상여금합계'로 하고, ROLLUP 함수를 사용할 것)

SELECT 부서, 상여내역, SUM(상여금) AS 상여금합계

FROM 상여금

GROUP BY ROLLUP(부서, 상여내역);

ROLLUP 함수가 적용되는 속성이 2개이므로 집계되는 레벨 수는 2+1로 총 3레벨이다. 가장 하위 레벨인 3레벨부터 표시된다. 3레벨은 부서별 상여내역별 '상여금'의 합계, 2레벨은 부서별 '상여금'의 합계, 1레벨은 전체 '상여금'의 합계가 표시된다. ROLLUP 함수는 표기된 속성의 순서에 따라 표시되는 집계 항목이 달라지므로 속성의 순서에 주의해야 한다. ROLLUP(상여내역, 부서로 지정하면 3레벨은 상여내역별 부서별 상여금'의 합계, 2레벨은 상여내역별 상여금의 합계, 1레벨은 전체 '상여금'의 합계가 표시되므로 부서별 상여금'의 집계는 확인할 수 없다.

〈상여금〉			
부서	이름	상여내역	상여금
기획	홍길동	연장근무	100
기획	일지매	연장근무	100
기획	최준호	야간근무	120
기획	장길산	특별근무	90
인터넷	강건달	특별근무	90
인터넷	서국현	특별근무	90
인터넷	박인식	연장근무	30
편집	김선달	특별근무	80
편집	황종근	연장근무	40
편집	성춘향	야간근무	80
편집	임꺽정	야간근무	80
편집	황진이	야간근무	50

〈결과〉			
부서	상여내역	상여금합계	
기획	야간근무	120	} 3레벨(부서별, 상여내역별 '상여금'의 합계)
기획	연장근무	200	
기획	특별근무	90	
기획		410	→ 2레벨(부서별 '상여금'의 합계)
편집	야간근무	210	} 3레벨
편집	연장근무	40	
편집	특별근무	80	
편집		330	→ 2레벨
인터넷	연장근무	30	} 3레벨
인터넷	특별근무	180	
인터넷		210	→ 2레벨
		950	→ 1레벨(전체 '상여금'의 합계)

3. SQL 응용-SEC_06(DML-SELECT-2)

3) 그룹 지정 검색

예제5) <상여금> 테이블에서 '부서', '상여내역', 그리고 '상여금'에 대해 부서별 상여내역별 소계와 전체 합계를 검색하시오.(단, 속성명은 '상여금합계'로 하고, CUBE 함수를 사용할 것)

SELECT 부서, 상여내역, SUM(상여금) AS 상여금합계

FROM 상여금

GROUP BY CUBE(부서, 상여내역);

CUBE 함수가 적용되는 속성이 2개이므로 집계되는 레벨 수는 2^n 로 총 4레벨이다. CUBE 함수는 가장 상위 레벨인 1레벨부터 표시된다. 1레벨은 전체 '상여금'의 합계, 2레벨은 상여내역별 '상여금'의 합계, 3레벨은 부서별 '상여금'의 합계, 4레벨은 부서별 상여내역별 '상여금'의 합계가 표시된다.

CUBE 함수는 ROLLUP 함수와 달리 인수로 주어진 속성을 대상으로 결합 가능한 모든 집계를 표시하므로 인수로 주어진 속성의 순서가 바뀌어도 표시 순서만 달라질 뿐 표시되는 집계 항목은 동일하다.

〈상여금〉			
부서	이름	상여내역	상여금
기획	홍길동	연장근무	100
기획	일지매	연장근무	100
기획	최준호	야간근무	120
기획	장길산	특별근무	90
인터넷	강건달	특별근무	90
인터넷	서국현	특별근무	90
인터넷	박인식	연장근무	30
편집	김선달	특별근무	80
편집	황종근	연장근무	40
편집	성춘향	야간근무	80
편집	임꺽정	야간근무	80
편집	황진이	야간근무	50

〈결과〉

부서	상여내역	상여금합계	
		950	→ 1레벨(전체 '상여금'의 합계)
	야간근무	330	} 2레벨(상여내역별 '상여금'의 합계)
	연장근무	270	
	특별근무	350	
기획		410	→ 3레벨(부서별 '상여금'의 합계)
기획	야간근무	120	} 4레벨(부서별, 상여내역별 '상여금'의 합계)
기획	연장근무	200	
기획	특별근무	90	
편집		330	→ 3레벨
편집	야간근무	210	} 4레벨
편집	연장근무	40	
편집	특별근무	80	
인터넷		210	→ 3레벨
인터넷	연장근무	30	} 4레벨
인터넷	특별근무	180	

3. SQL 응용-SEC_06(DML-SELECT-2)

4) 집합 연산자를 이용한 통합 질의

; 집합 연산자를 사용하여 2개 이상의 테이블의 데이터를 하나로 통합한다.

표기 형식

SELECT 속성명1, 속성명2, ...

FROM 테이블명

UNION | UNION ALL | INTERSECT | EXCEPT

SELECT 속성명1, 속성명2, ...

FROM 테이블명

[ORDER BY 속성명 [ASC | DESC]];

- 두 개의 SELECT문에 기술한 속성들은 개수와 데이터 유형이 서로 동일해야 한다.
- 집합 연산자의 종류(통합 질의의 종류)

집합 연산자	설명	집합 종류
UNION	두 SELECT문의 조회 결과를 통합하여 모두 출력한다. 중복된 행은 한 번만 출력한다.	합집합
UNION ALL	두 SELECT문의 조회 결과를 통합하여 모두 출력한다. 중복된 행도 그대로 출력한다.	합집합
INTERSECT	두 SELECT문의 조회 결과 중 공통된 행만 출력한다.	교집합
EXCEPT	첫 번째 SELECT문의 조회 결과에서 두 번째 SELECT문의 조회 결과를 제외한 행을 출력한다.	차집합

3. SQL 응용-SEC_06(DML-SELECT-2)

4) 집합 연산자를 이용한 통합 질의

〈사원〉	
사원	직급
김형석	대리
홍영선	과장
류기선	부장
김현천	이사

〈직원〉	
사원	직급
신원섭	이사
이성호	대리
홍영선	과장
류기선	부장

예제1) <사원> 테이블과 <직원> 테이블을 통합하는 질의문을 작성하시오.(단, 같은 레코드가 중복되어 나오지 않게 하시오.)

```
SELECT * FROM 사원  
UNION  
SELECT * FROM 직원;
```

〈결과〉	
사원	직급
김현천	이사
김형석	대리
류기선	부장
신원섭	이사
이성호	대리
홍영선	과장

3. SQL 응용-SEC_06(DML-SELECT-2)

4) 집합 연산자를 이용한 통합 질의

〈사원〉		〈직원〉	
사원	직급	사원	직급
김형석	대리	신원섭	이사
홍영선	과장	이성호	대리
류기선	부장	홍영선	과장
김현천	이사	류기선	부장

예제2) <사원> 테이블과 <직원> 테이블에 공통으로 존재하는 레코드만 통합하는 질의문을 작성하시오.

SELECT * FROM 사원

INTERSECT

SELECT * FROM 직원;

〈결과〉	
사원	직급
류기선	부장
홍영선	과장

SQL 응용 - SEC_06(DML-SELECT-2) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DML-SELECT-2)

1. 다음 중 SQL의 집계 함수(Aggregation Function)가 아닌 것은?

- ① AVG ② COUNT
- ③ SUM ④ CREATE

집계 함수(Aggregation Function)

집계 함수는 여러 행으로부터 하나의 결과값을 반환하는 함수이다. SELECT 구문에서만 사용되며, 집계 함수는 열(컬럼)끼리 연산을 수행한다. 주로, 평균, 총 합계, 최대값, 최소값 등을 구하는데 사용된다. AVG 함수는 선택한 열(컬럼)의 평균을 계산하고, SUM은 선택한 열(컬럼)의 총 합계를 계산하는데, MIN/MAX 함수와 다르게 숫자인 값에 대해서만 연산이 가능하며, **NULL값은 무시하고 계산**한다. 따라서, NULL 0으로 취급하여 평균을 구하고 싶다면 NULL을 0을 지정하는 작업(DEFAULT값 설정)을 추가로 해야 한다.

COUNT 함수는 특정 열(컬럼)의 행의 개수를 세는 함수이다.

COUNT(*)로 작성하면 테이블에 존재하는 행의 모든 개수가 반환되고 특정 열에 대해서 COUNT 함수를 수행하면 **해당 열이 NULL이 아닌 행의 행의 개수를 반환**한다. 아울러, **DISTINCT 명령어를 이용하면 중복을 제외한 값의 개수를 구할 수도 있다.**

3. [상반기진급] 테이블과 [하반기진급] 테이블은 모두 '사번', '이름', '부서' 필드로 구성되어 있다. 다음 두 테이블의 레코드를 통합하려고 할 때 쿼리문으로 올바른 것은?

- ① Select 사번, 이름, 부서 From 상반기진급, 하반기진급
Where 상반기진급.사번 = 하반기진급.사번;
- ② Select 사번, 이름, 부서 From 상반기진급
JOIN Select 사번, 이름, 부서 From 하반기진급;
- ③ Select 사번, 이름, 부서 From 상반기진급
OR Select 사번, 이름, 부서 From 하반기진급;
- ④ Select 사번, 이름, 부서 From 상반기진급
UNION
Select 사번, 이름, 부서 From 하반기진급;

4. SQL문에서 HAVING을 사용할 수 있는 절은?

- ① LIKE 절 ② WHERE 절
- ③ GROUP BY 절 ④ ORDER BY 절

HAVING 절 : GROUP BY와 함께 사용되며, 그룹에 대한 조건을 지정한다. 하나 하나의 레코드에는 WHERE절을 사용하여 조건을 지정한다. LIKE절은 와일드 카드 값 %(모든 문자를 대변), _(한 문자를 대변),

SQL 응용 - SEC_06(DML-SELECT-2) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DML-SELECT-2)

5. 테이블 R1, R2에 대하여 다음 SQL문의 결과는?

(SELECT 학번 FROM R1)

INTERSECT

(SELECT 학번 FROM R2)

[R1] 테이블		[R2] 테이블	
학번	학점 수	학번	과목번호
20201111	15	20202222	CS200
20202222	20	20203333	CS300

①

학번	학점 수	과목번호
20202222	20	CS200

②

학번
20202222

③

학번
20201111
20202222
20203333

④

학번	학점 수	과목번호
20201111	15	NULL
20202222	20	CS200

7. 다음 SQL에서 사용되는 내장 함수 중 잘못 표현된 것은?

① SUM - 열에 있는 값들의 합계

② COUNT - 튜플의 개수

③ MAX - 열에서 최대값

④ AVR - 열에 있는 값들의 평균

평균을 구하는 함수는 AVG이다.

8. 다음은 질의 수행 결과를 정렬하는 방법에 대한 설명이다.

옳지 않은 것은?

① ORDER BY절을 사용한다.

② 기본 정렬 방식은 내림차순이다.

③ ASC 키워드를 사용하여 사용자가 오름차순 정렬 방식을 지정할 수 있다.

④ DESC 키워드를 사용하여 사용자가 내림차순 정렬 방식을 지정할 수 있다.

ORDER BY의 기본 정렬 방식은 오름차순이다. 그리고, ASC는 생략 가능하며 하지만 DESC는 반드시 표식 해야 한다.

3. SQL 응용-SEC_07(DML-JOIN)

1) JOIN의 개념

; JOIN(조인)은 2개의 테이블에 대해 연관된 튜플들을 결합하여, 하나의 새로운 릴레이션을 반환한다.

- JOIN은 크게 INNER JOIN과 OUTER JOIN으로 구분된다.
- JOIN은 일반적으로 FROM절에 기술하지만, 릴레이션이 사용되는 어느 곳에서나 사용할 수 있다.

2) INNER JOIN

; INNER JOIN은 일반적으로 EQUI JOIN과 NON-EQUI JOIN으로 구분된다.

- 조건이 없는 INNER JOIN을 수행하면 CROSS JOIN과 동일한 결과를 얻을 수 있다.
- EQUI JOIN
 - EQUI JOIN은 JOIN 대상 테이블에서 공통 속성을 기준으로 '='(equal) 비교에 의해 같은 값을 가지는 행을 연결하여 결과를 생성하는 JOIN 방법이다.
 - EQUI JOIN에서 JOIN 조건이 '='일 때 동일한 속성이 두 번 나타나게 되는데, 이중 중복된 속성을 제거하여 같은 속성을 한 번만 표기하는 방법을 NATURAL JOIN이라고 한다.
 - EQUI JOIN에서 연결 고리가 되는 공통 속성을 JOIN 속성이라고 한다.

CROSS JOIN(교차 조인)

- 교차 조인은 조인하는 두 테이블에 있는 튜플들의 순서쌍을 결과로 반환한다.
- 교차 조인의 결과로 반환되는 테이블의 행의 수는 두 테이블의 행 수를 곱한 것과 같다.(카티션 곱)

3. SQL 응용-SEC_07(DML-JOIN)

2) INNER JOIN

- EQUI JOIN

- WHERE절을 이용한 EQUI JOIN의 표기 형식

```
SELECT [테이블명1.]속성명, [테이블명2.]속성명, ...  
FROM 테이블명1, 테이블명2,  
WHERE 테이블명1.속성명 = 테이블명2.속성명;
```

- NATURAL JOIN절을 이용한 EQUI JOIN의 표기 형식

```
SELECT [테이블명1.]속성명, [테이블명2.]속성명, ...  
FROM 테이블명1 NATURAL JOIN 테이블명2;
```

- JOIN ~ USING절을 이용한 EQUI JOIN의 표기 형식

```
SELECT [테이블명1.]속성명, [테이블명2.]속성명, ...  
FROM 테이블명1 JOIN 테이블명2 USING(속성명);
```

실무에서 가장 많이 사용되는 조인 형식은 WHERE절을 이용한 조인이다.

3. SQL 응용-SEC_07(DML-JOIN)

2) INNER JOIN

● EQUI JOIN

〈학생〉				
학번	이름	학과코드	선배	성적
15	고길동	com		83
16	이순신	han		96
17	김선달	com	15	95
19	아무개	han	16	75
37	박치민		17	55

〈학과〉	
학과코드	학과명
com	컴퓨터
han	국어
eng	영어

〈성적등급〉		
등급	최저	최고
A	90	100
B	80	89
C	60	79
D	0	59

예제 1) <학생> 테이블과 <학과> 테이블에서 '학과코드' 값이 같은 튜플을 JOIN하여 '학번', '이름', '학과코드', '학과명'을 출력하는 SQL문을 작성하시오.

- **SELECT** 학생.학번, 학생.이름, 학생.학과코드, 학과.학과명 **FROM** 학생, 학과
WHERE 학생.학과코드 = 학과.학과코드;
- **SELECT** 학생.학번, 학생.이름, 학생.학과코드, 학과.학과명
FROM 학생 **NATURAL JOIN** 학과;
- **SELECT** 학생.학번, 학생.이름, 학생.학과코드, 학과.학과명
FROM 학생 **JOIN** 학과 **USING**(학과.학과코드);

〈결과〉			
학번	이름	학과코드	학과명
15	고길동	com	컴퓨터
16	이순신	han	국어
17	김선달	com	컴퓨터
19	아무개	han	국어

NATURAL JOIN은 조인할 속성을 지정하지 않기 때문에 조인하려는 두 테이블에는 이름과 도메인이 같은 속성이 반드시 존재해야 한다. <학생> 테이블과 <학과> 테이블에는 같은 이름의 속성과 범위가 같은 도메인을 갖는 '학과코드'가 있기 때문에 **NATURAL JOIN** 가능한 것이다.

3. SQL 응용-SEC_07(DML-JOIN)

2) INNER JOIN

● NON-EQUI JOIN

- NON-EQUI JOIN은 JOIN 조건에 '=' 조건이 아닌 나머지 비교 연산자, 즉 >, <, <>, >=, <= 연산자를 사용하는 JOIN 방법이다.

- 표기 형식

SELECT [테이블명1.]속성명, [테이블명2.]속성명, ...

FROM 테이블명1, 테이블명2, ...

WHERE (NON-EQUI JOIN 조건);

예제2) <학생> 테이블과 <성적등급> 테이블을 JOIN하여 각 학생의 '학번', '이름', '성적', '등급'을 출력하는 SQL문을 작성하시오.

SELECT 학생.학번, 학생.이름, 학생.성적, 성적등급.등급

FROM 학생, 성적등급

WHERE 학생.성적 BETWEEN 성적등급.최저

AND 성적등급.최고;

〈학생〉				
학번	이름	학과코드	선배	성적
15	고길동	com		83
16	이순신	han		96
17	김선달	com	15	95
19	아무개	han	16	75
37	박치민		17	55

〈성적등급〉		
등급	최저	최고
A	90	100
B	80	89
C	60	79
D	0	59

〈결과〉			
학번	이름	성적	등급
15	고길동	83	B
16	이순신	96	A
17	김선달	95	A
19	아무개	75	C
37	박치민	55	D

3. SQL 응용-SEC_07(DML-JOIN)

3) OUTER JOIN

; OUTER JOIN은 릴레이션에서 JOIN 조건에 만족하지 않는 튜플도 결과로 출력하기 위한 JOIN 방법으로, LEFT OUTER JOIN, RIGHT OUTER JOIN, FULL OUTER JOIN이 있다.

- LEFT OUTER JOIN : INNER JOIN의 결과를 구한 후, 우측 항 릴레이션의 어떤 튜플과도 맞지 않는 좌측 항의 릴레이션에 있는 튜플들을 전부 출력하고 우측 항에 해당하는 튜플들에 NULL 값을 붙여서 INNER JOIN의 결과에 추가한다.

- 표기 형식

- SELECT [테이블명1.]속성명, [테이블명2.]속성명, ...
FROM 테이블명1 LEFT OUTER JOIN 테이블명2
ON 테이블명1.속성명 = 테이블명2.속성명;
- SELECT [테이블명1.]속성명, [테이블명2.]속성명, ...
FROM 테이블명1, 테이블명2
WHERE 테이블명1.속성명 = 테이블명2.속성명(+);

3. SQL 응용-SEC_07(DML-JOIN)

3) OUTER JOIN

● RIGHT OUTER JOIN : INNER JOIN의 결과를 구한 후, 좌측 항 릴레이션의 어떤 튜플과도 맞지 않는 우측 항의 릴레이션에 있는 튜플들을 전부 출력하고 좌측 항에 해당하는 튜플들은 NULL 값을 붙여서 INNER JOIN의 결과에 추가한다.

- 표기 형식

- SELECT [테이블명1.]속성명, [테이블명2.]속성명, ...
FROM 테이블명1 RIGHT OUTER JOIN 테이블명2
ON 테이블명1.속성명 = 테이블명2.속성명;
- SELECT [테이블명1.]속성명, [테이블명2.]속성명, ...
FROM 테이블명1, 테이블명2
WHERE 테이블명1.속성명(+) = 테이블명2.속성명;

3. SQL 응용-SEC_07(DML-JOIN)

3) OUTER JOIN

- FULL OUTER JOIN

- LEFT OUTER JOIN과 RIGHT OUTER JOIN을 합쳐 놓은 것이다.
- INNER JOIN의 결과를 구한 후, 좌측 항의 릴레이션의 튜플들에 대해 우측 항의 릴레이션의 어떤 튜플과도 맞지 않는 튜플들에 NULL 값을 붙여서 INNER JOIN의 결과에 추가한다. 그리고 유사하게 우측 항의 릴레이션의 튜플들에 대해 좌측 항의 릴레이션의 어떤 튜플과도 맞지 않는 튜플들에 NULL 값을 붙여서 INNER JOIN의 결과에 추가한다.
- 표기 형식
 - SELECT [테이블명1.]속성명, [테이블명2.]속성명, ...
FROM 테이블명1 FULL OUTER JOIN 테이블명2
ON 테이블명1.속성명 = 테이블명2.속성명;

3. SQL 응용-SEC_07(DML-JOIN)

3) OUTER JOIN

예제1) <학생> 테이블과 <학과> 테이블에서 '학번', '이름', '학과코드', '학과명'을 출력하는 SQL문을 작성하시오. 이때, '학과코드'가 입력되지 않은 학생도 출력하시오.

- SELECT 학생.학번, 학생.이름, 학생.학과코드, 학과.학과명
FROM 학생 LEFT OUTER JOIN 학과
ON 학생.학과코드 = 학과.학과코드;
- SELECT 학생.학번, 학생.이름, 학생.학과코드, 학과.학과명
FROM 학생, 학과
WHERE 학생.학과코드 = 학과.학과코드(+);

해설 : INNER JOIN을 하면 '학과코드'가 입력되지 않은 '박치민'은 출력되지 않는다. 그러므로 JOIN 구문을 기준으로 왼쪽 테이블, 즉 <학생>의 자료는 모두 출력되는 LEFT OUTER JOIN을 사용한 것이다.

〈학생〉				
학번	이름	학과코드	선배	성적
15	고길동	com		83
16	이순신	han		96
17	김선달	com	15	95
19	아무개	han	16	75
37	박치민		17	55

〈학과〉	
학과코드	학과명
com	컴퓨터
han	국어
eng	영어

〈결과〉			
학번	이름	학과코드	학과명
15	고길동	com	컴퓨터
16	이순신	han	국어
17	김선달	com	컴퓨터
19	아무개	han	국어
37	박치민		

3. SQL 응용-SEC_07(DML-JOIN)

3) OUTER JOIN

예제2) <학생> 테이블과 <학과> 테이블에서 '학과코드' 값이 같은 튜플을 JOIN하여 '학번', '이름', '학과코드', '학과명'을 출력하는 SQL문을 작성하시오. 이때, '학과코드'가 입력 안 된 학생이나 학생이 없는 '학과코드'도 모두 출력하시오.

•SELECT 학생.학번, 학생.이름, 학과.학과코드, 학과.학과명
FROM 학생 FULL OUTER JOIN 학과
ON 학생.학과코드 = 학과.학과코드;

해설 : FULL OUTER JOIN을 하면 JOIN 구문으로 연결되지 않는 자료도 모두 출력된다. "박치민"은 '학과코드'가 없고, "eng"는 <학생> 테이블에 등록되지 않아서 연결고리가 없지만 FULL OUTER JOIN을 했으므로 모두 출력된다.

〈결과〉			
학번	이름	학과코드	학과명
15	고길동	com	컴퓨터
16	이순신	han	국어
17	김선달	com	컴퓨터
19	아무개	han	국어
37	박치민		
		eng	영어

〈학생〉				
학번	이름	학과코드	선배	성적
15	고길동	com		83
16	이순신	han		96
17	김선달	com	15	95
19	아무개	han	16	75
37	박치민		17	55

〈학과〉	
학과코드	학과명
com	컴퓨터
han	국어
eng	영어

3. SQL 응용-SEC_07(DML-JOIN)

4) SELF JOIN

- SELF JOIN은 같은 테이블에서 2개의 속성을 연결하여 EQUI JOIN을 하는 JOIN 방법이다.
- 표기 형식
 - SELECT [별칭1.]속성명, [별칭1.]속성명, ...
FROM 테이블명1 [AS] 별칭1 JOIN 테이블명1 [AS] 별칭2
ON 별칭1.속성명 = 별칭2.속성명;
 - SELECT [별칭1.]속성명, [별칭1.]속성명, ...
FROM 테이블명1 [AS] 별칭1, 테이블명1 [AS] 별칭2
WHERE 별칭1.속성명 = 별칭2.속성명;

3. SQL 응용-SEC_07(DML-JOIN)

4) SELF JOIN

예제) <학생> 테이블을 SELF JOIN하여 선배가 있는 학생과 선배의 '이름'을 표시하는 SQL문을 작성하시오.

- SELECT A.학번, A.이름, B.이름 AS 선배
FROM 학생 AS A JOIN 학생 AS B
ON A.선배 = B.학번;
- SELECT A.학번, A.이름, B.이름 AS 선배
FROM 학생 AS A, 학생 AS B
ON A.선배 = B.학번;

〈학생〉				
학번	이름	학과코드	선배	성적
15	고길동	com		83
16	이순신	han		96
17	김선달	com	15	95
19	아무개	han	16	75
37	박치민		17	55

- B.이름 AS 선배는 테이블의 '이름'을 출력하되 필드 명을 '선배'로 표시하라는 의미이다.

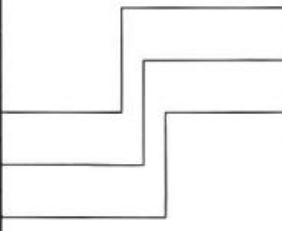
〈결과〉		
학번	이름	선배
17	김선달	고길동
19	아무개	이순신
37	박치민	김선달

3. SQL 응용-SEC_07(DML-JOIN)

4) SELF JOIN

; SELF 조인은 1개의 테이블을 2개의 이름으로 사용하므로 종종 결과가 혼동된다. 이럴 때는 같은 테이블을 2개 그려서 생각하면 쉽게 결과를 알아낼 수 있다. '학번', '이름', '선배' 필드만 사용하므로 3개의 필드만 가지고 생각해 보자.

〈A〉			〈B〉		
학번	이름	선배	학번	이름	선배
15	고길동		15	고길동	
16	이순신		16	이순신	
17	김선달	15	17	김선달	15
19	아무개	16	19	아무개	16
37	박치민	17	37	박치민	17



해설 : <A>테이블의 '선배'와 테이블의 '학번'이 같은 튜플을 조인하면 위 그림과 같이 연결된다. 여기서 두 테이블 간 조인된 튜플들만을 대상으로 <A>테이블에서 '학번', '이름'을 표시하고, 테이블에서 이름을 출력하되 필드 명을 '선배'로 하여 출력하면 앞의 결과와 같이 된다.

SQL 응용 - SEC_07(DML-JOIN) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(DML-JOIN)

1. 다음 R1과 R2의 테이블에서 아래의 실행 결과를 얻기 위한 SQL 문은?

[R1] 테이블				
학번	이름	학년	학과	주소
1000	홍길동	1	컴퓨터공학	서울
2000	김철수	1	전기공학	경기
3000	강남길	2	전자공학	경기
4000	오말자	2	컴퓨터공학	경기
5000	장미화	3	전자공학	서울

[실행결과]	
과목번호	과목이름
C100	컴퓨터구조
C200	데이터베이스

[R2] 테이블				
학번	과목번호	과목이름	성적	주소
1000	C100	컴퓨터구조	A	91
2000	C200	데이터베이스	A+	99
3000	C100	컴퓨터구조	B+	89
3000	C200	데이터베이스	B	85
4000	C200	데이터베이스	A	93
4000	C300	운영체제	B+	88
5000	C300	운영체제	B	82

- ① `SELECT R2.과목번호, R2.과목이름 FROM R1, R2
WHERE R1.학번 = R2.학번
AND R1.학과 = '전자공학'
AND R1.이름 = '강남길';`
- ② `SELECT R2.과목번호, R2.과목이름 FROM R1, R2
WHERE R1.학번 = R2.학번
OR R1.학과 = '전자공학'
OR R1.이름 = '홍길동';`

3. 다음 중 조인(Join)에 대한 설명으로 옳지 못한 것은?

- ① 두 개 이상의 테이블로부터 원하는 데이터를 검색하는 방법이다.
② 조인에 사용되는 기준 필드는 동일하거나 호환되는 데이터 형식을 가져야 한다.
③ 조인되는 두 테이블의 필드 수가 동일할 필요는 없다.

④ 같은 테이블에서 2개의 속성을 연결하여 EQUI JOIN을 하는 방법을 CROSS JOIN이라고 한다.

같은 테이블에서 2개의 속성을 통해서 조인을 수행한다는 것은 SELF JOIN 방법이다.

4. 다음 쿼리에서 두 테이블의 필드 값이 일치하는 레코드만 조인하기 위해 괄호 안에 넣어야 할 것으로 옳은 것은?

`SELECT 필드목록
FROM 테이블1, 테이블2
WHERE 테이블1.필드 () 테이블2.필드;`

① = ② JOIN

③ + ④ -

조인된 필드의 값이 일치하는 행을 연결하여 결과를 생성하는 JOIN 방법을 EQUI JOIN이라고 하며 조건절에 연산자로 '='을 사용한다.

A close-up, low-angle shot of a white car's side mirror and door handle against a light blue sky. The car is on the left side of the frame, and the sky is on the right. The text "감사합니다." is overlaid on the right side of the image.

감사합니다.