기본 SQL 작성하기

기본 SQL 작성하기

- DDL 활용(학습모듈 p.15~)
- 학습 목표 • 테이블의 구조와 제약 조건을 생성, 삭제하고 수정하는 DDL(Data Definition Language) 명령문을 작성할 수 있다.

▣ DDL 개요

1. DDL 대상

✓ DDL(Data Definition Language)은 `데이터를 정의하는 언어'로서, 보다 엄밀하게 말하면 `데이터를 담는 그릇을 정의하는 언어'이며, 이러한 그릇을 DBMS에서는 오브젝트라고 한다. DDL을 통해 정의할 수 있는 대상, 오브젝트 유형은 다음과 같다.

<표 1-1> DDL 대상

DDL 대상	설명	비고
스키마(Schema)	- DBMS 특성과 구현 환경을 감안한 데이터 구조 - 직관적으로 하나의 데이터베이스로 이해 가능	DBMS마다 차이
도메인(Domain)	- 속성의 데이터 타입과 크기, 제약 조건 등을 지정한 정보 - 속성이 가질 수 있는 값의 범위로 이해 가능	예를 들어, 주소를 VARCHAR(120)로 정의
테이블(Table)	- 데이터 저장 공간	본 학습 대상
뷰(View)	- 하나 이상의 물리 테이블에서 유도되는 가상의 논리 테이블	학습 2-2 참조
인덱스(Index)	- 검색을 빠르게 하기 위한 데이터 구조	학습 2-1 참조

기본 SQL 작성하기

- DDL 활용(학습모듈 p.16~)

2. DDL 조작 방법

- 오브젝트를 생성, 변경 그리고 제거하기 위해 다음과 같은 명령어를 사용한다.

<표 1-2> DDL 명령어

구분	DDL 명령어	내용
생성	CREATE	데이터베이스 오브젝트 생성
변경	ALTER	데이터베이스 오브젝트 변경
삭제	DROP	데이터베이스 오브젝트 삭제
	TRUNCATE	데이터베이스 오브젝트 내용 삭제

- DDL 명령어로 분류되지는 않지만 DDL과 같이 사용되는 명령어가 있다.
- 비상용 제품인 M*SQL의 경우, 생성된 오브젝트의 목록을 조회하기 위해서는 SHOW 명령문을 사용하며, 내용을 조회하기 위해서는 SELECT 문을 사용한다.
- 상용 제품인 O*의 경우 SELECT로 목록과 내용을 조회한다.

기본 SQL 작성하기

- DDL 활용(학습모듈 p.16~)

▣ DDL 활용

✓ 데이터베이스를 구축하기 위해 스키마, 테이블, 도메인, 인덱스, 뷰와 같은 오브젝트에 대한 DDL 적용이 필요하나, 본 학습에서는 테이블만을 대상으로 한다.

1. 테이블 생성

✓ 테이블 생성을 위한 DDL 사용 방법은 다음과 같이 두 종류로 분류할 수 있다.

<표 1-3> 테이블 생성 SQL문

구분	문법
신규 생성	CREATE TABLE 테이블이름 (열이름 데이터 타입 [DEFAULT 값] [NOT NULL] {,열이름 데이터 타입 [DEFAULT 값] [NOT NULL] }* [PRIMARY KEY (열 리스트),] {[FOREIGN KEY (열 리스트) REFERENCES 테이블이름 [(열이름)] [ON DELETE 옵션] [ON UPDATE 옵션]], }* [CHECK (조건식) ¦ UNIQUE(열이름)]) ;
다른 테이블 정보를 이용한 테이블 생성[1)]	CREATE TABLE 테이블이름 AS SELECT 문;

1) 다른 테이블을 이용해서 신규 테이블을 생성하는 방법은 DBMS 제품마다 차이가 있다.

기본 SQL 작성하기

- DDL 활용(학습모듈 p.17~)

▣ DDL 활용

2. 테이블 변경

✓ ALTER를 이용하여 테이블 구조를 변경하는 문법은 다음과 같다.

<표 1-4> ALTER 이용한 테이블 변경 SQL문

구분	문법
열 추가	ALTER TABLE 테이블이름 ADD 열이름 데이터타입 [DEFAULT 값]
열 데이터 타입 변경	ALTER TABLE 테이블이름 MODIFY 열이름 데이터타입 [DEFAULT 값]
열 삭제	ALTER TABLE 테이블이름 DROP 열이름

기본 SQL 작성하기

- DDL 활용(학습모듈 p.17~)

▣ DDL 활용

3. 테이블 삭제, 절단, 이름 변경

✓ DROP TABLE, TRUNCATE TABLE, RENAME TABLE 명령문을 사용하여 테이블을 삭제, 절단, 이름 변경을 할 수 있다.

✓ 테이블 및 테이블 내용을 삭제하기 위한 명령어의 사용 문법은 다음과 같다.

<표 1-5> 테이블 삭제 및 이름 변경 방법

구분	문법
테이블 삭제	DROP TABLE 테이블이름
테이블 내용 삭제	TRUNCATE TABLE 테이블이름
테이블이름 변경	RENAME TABLE 이전테이블이름 TO 새로운테이블이름
	ALTER TABLE 이전테이블이름 RENAME 새로운테이블이름

기본 SQL 작성하기

- DDL 활용(학습모듈 p.17~)

▣ 제약 조건 적용

1. 제약 조건 유형

✓ 다음과 같은 제약 조건을 테이블 생성 과정에 적용할 수 있다.

<표 1-6> 테이블 생성에 사용되는 제약 조건

제약 조건	설명
PRIMARY KEY	테이블의 기본키를 정의함. 기본으로 NOT NULL, UNIQUE 제약이 포함됨.
FOREIGN KEY	외래키를 정의함. 참조 대상을 테이블이름(열이름)으로 명시해야 함. 참조 무결성 위배 상황 발생 시 처리 방법으로 옵션 지정 가능 - NO ACTION, SET DEFAULT, SET NULL, CASCADE
UNIQUE	테이블 내에서 열은 유일한 값을 가져야 함. 테이블 내에서 동일한 값을 가져서는 안 되는 항목에 지정함.
NOT NULL	테이블 내에서 관련 열의 값은 NULL일 수 없음. 필수 입력 항목에 대해 제약 조건으로 설정함.
CHECK	개발자가 정의하는 제약 조건 상황에 따라 다양한 조건 설정 가능

2. 제약 조건 활용

✓ 테이블 생성을 위한 CREATE 문에 제약 조건을 명시하는 형태로 사용하며, ALTER를 통해 테이블의 제약 조건을 변경할 수 있다.

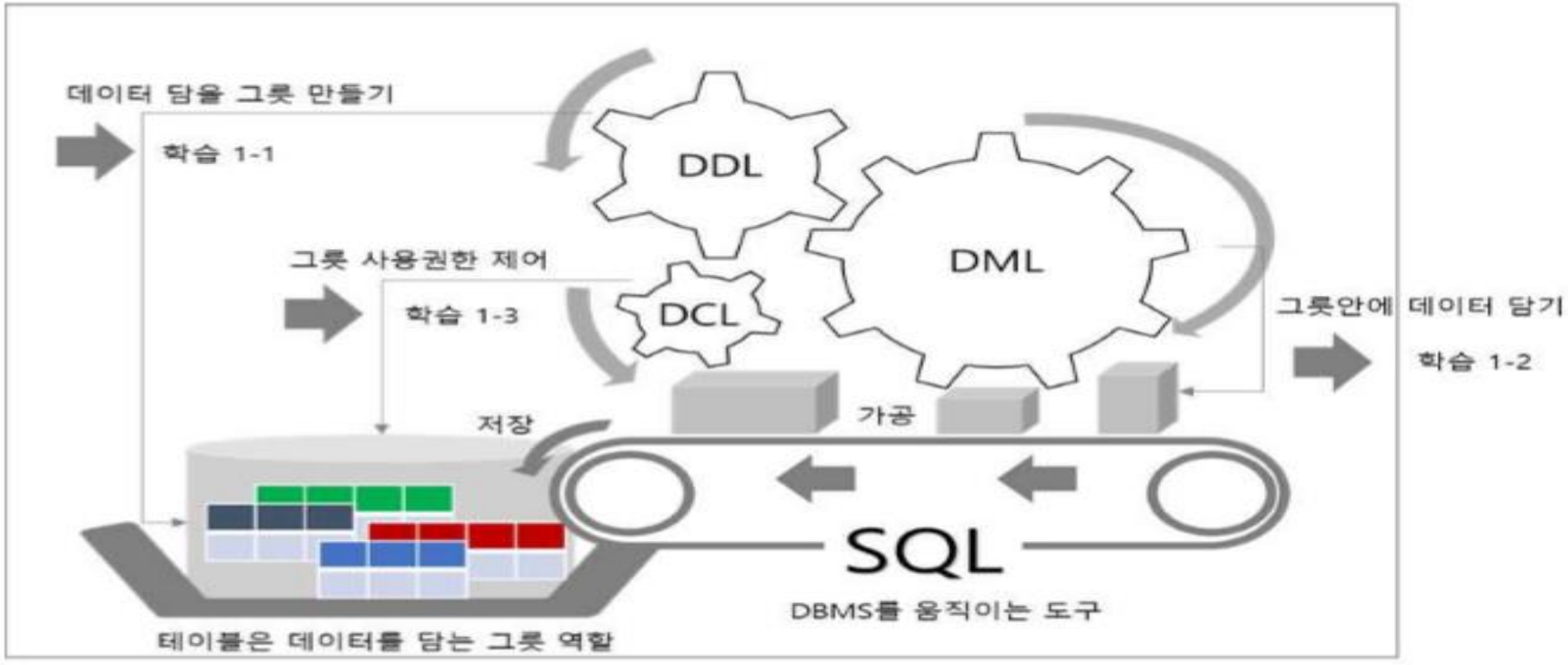
기본 SQL 작성하기

- DDL 활용(학습모듈 p.18~)

▣ SQL 활용 주요 내용

✓ SQL은 DDL, DML 및 DCL과 같은 유형의 작업을 통해 데이터베이스 안에 그릇을 만들고 그 안에 데이터를 담거나 꺼내어 사용하는 도구다.

✓ 본 학습의 주요 내용과 이들 간의 관계는 다음 그림과 같다.



[그림 1-1] SQL 활용을 위한 주요 학습 내용

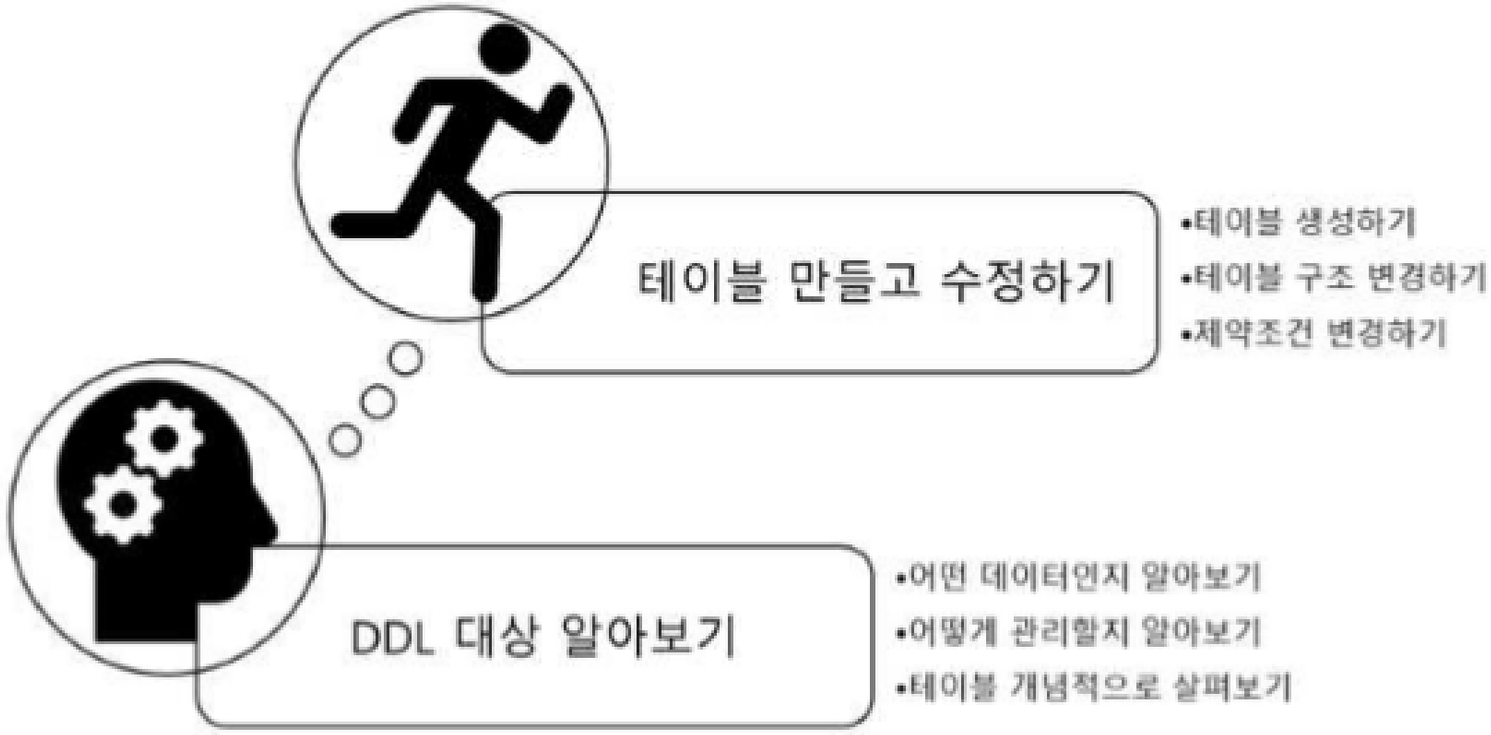
기본 SQL 작성하기

- DDL 활용(학습모듈 p.19~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서



[그림 1-2] DDL 활용을 위한 수행내용

기본 SQL 작성하기

- DDL 활용(학습모듈 p.20~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서

① 테이블을 통해 관리할 데이터를 확인한다.

1. 데이터를 확인한다.

✓ 본 학습 주제인 데이터를 정의 (Data Definition) 한다는 것은 다음과 같은 데이터를 담는 그릇을 만드는 것과 같다.

주문테이블

주문번호	고객번호	주문일	주문가격	배송 도시	배송완료일	결제금액	할인금액
A0100	24680	20170704	55,000	서울	20170707	45,000	10,000
X0300	13579	20170801	70,000	부산	Null	50,000	20,000

[그림 1-3] 주문 데이터 예시

기본 SQL 작성하기

- DDL 활용(학습모듈 p.20~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서

2. 데이터 타입을 결정한다.

✓ 데이터 타입은 자료를 표현할 때 필요한 공간과 형식을 규정하는 기준이다.

✓ SQL에서 컬럼을 정의할 때 규정되는 데이터 타입은 이후 해당 컬럼에 입력되는 데이터의 종류와 크기를 제한하며, 다른 종류의 데이터나 큰 데이터가 입력되면 오류가 나타난다.

✓ SQL 데이터의 유형은 크게 숫자형, 문자형, 날짜 등의 유형으로 나눌 수 있으며, 구체적인 형식은 NUMERIC, CHARACTER, VARCHAR, DATETIME 등이 있으나, 실제 사용하는 SQL 제품에 따라 사용되는 표현은 조금씩 달라질 수 있다.

✓ [그림 1-3]과 같은 데이터를 저장할 테이블을 만들고자 할 때, 각 컬럼의 값은 크게 문자열과 숫자로 구분할 수 있으므로 각각 다음과 같은 데이터 타입을 사용한다.

- 문자열 – VARCHAR
- 숫자 – DECIMAL

기본 SQL 작성하기

- DDL 활용(학습모듈 p.20~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서

3. 제약 조건이 무엇인지 확인한다.

✓ 제약 조건으로는 주문 번호를 기본키(Primary Key)로 한다. 하지만 테이블의 변경으로 추가적인 제약 조건이 발생할 수 있다.

⑵ DDL 관리 대상인 테이블에 대해 조사한다.

1. 테이블의 구성요소에 대해 조사한다.

✓ 테이블은 행(Row)과 열(Column)로 구성되는 가장 기본적인 데이터베이스 객체로, 데이터베이스 내에서 모든 데이터는 테이블 안에 저장된다.

✓ 다음 그림에서 볼 수 있듯이, 하나의 행은 다른 행과 구별되는 정보를 가지도록 구성되어 있으며, 이러한 개별 행을 유일하게 식별하는 속성을 기본키 (Primary Key) 라고 한다.

✓ 또 데이터베이스는 정보의 중복을 최소화하기 위해 정보 종류를 테이블에 해당하는 엔티티(Entity) 단위로 분리하여 저장하며, 이렇게 분리된 정보 사이의 관계를 이어주는 수단이 외래키(Foreign Key)이다.

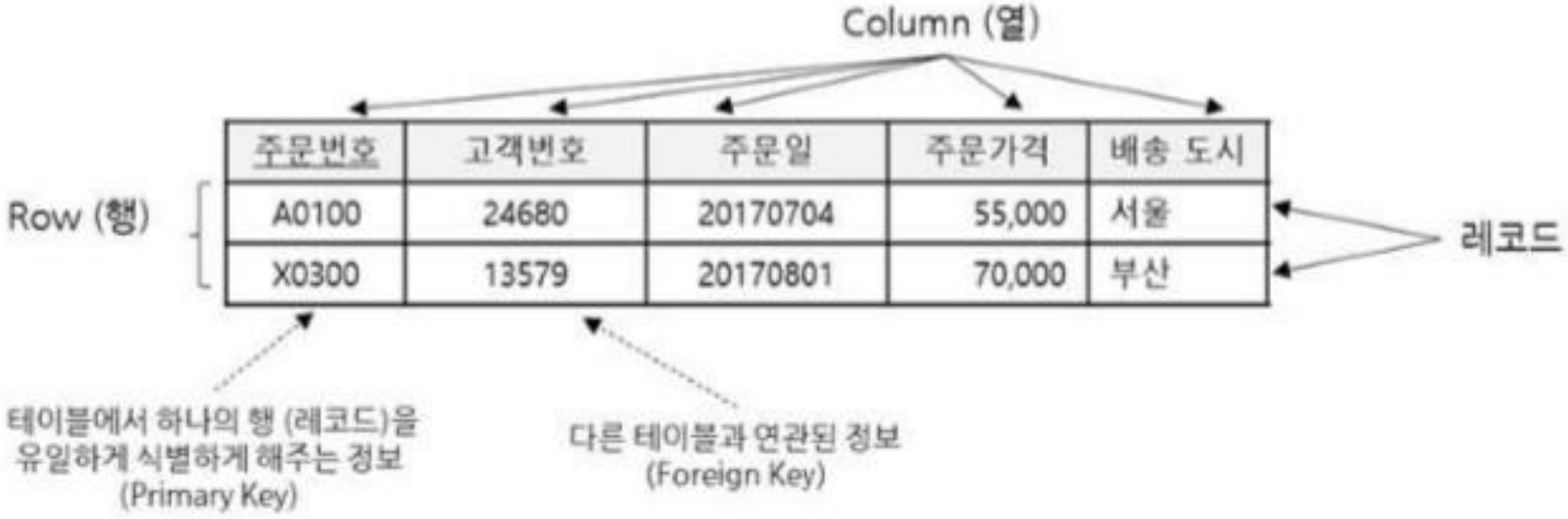
기본 SQL 작성하기

- DDL 활용(학습모듈 p.21~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서



[그림 1-4] 테이블의 내부 구성 요소

기본 SQL 작성하기

- DDL 활용(학습모듈 p.21~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서

2. 테이블에서 사용할 제약조건을 선택한다.

✓ 테이블 구조를 정의하면서 데이터 무결성을 유지하기 위해 제약 조건을 이용한다. 제약 조건이 테이블 생성 과정의 필수 요소는 아니지만 데이터베이스의 일관성(Consistency)을 유지하고 잘못된 데이터의 입력과 수정으로부터 데이터베이스를 보호하는 데 필수적인 요소이므로 반드시 정의하는 것이 좋다.

✓ 대표적인 제약 조건은 기본키(Primary Key)와 외래키(Foreign Key)에 의한 제약 조건이다.

✓ 그 밖에도 UNIQUE KEY, PRIMARY KEY, FOREIGN KEY, NOT NULL, CHECK 등이 있다.

기본 SQL 작성하기

- DDL 활용(학습모듈 p.22~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서.

3 테이블을 생성하고 변경한다.

1. CREATE 문으로 테이블을 생성한다.

✓ [그림 1-3]과 같은 데이터를 저장하기 위한 테이블을 생성하기 위해 다음과 같은 SQL문을 사용한다 (각 데이터 타입의 괄호 안 숫자는 최대 길이를 의미한다.).

<표 1-7> 테이블 생성 SQL문

위치	테이블 생성 SQL문

```
SQL> create table ordertbl(
  2  orderNo varchar2(16) not null,
  3  customNo varchar2(16) not null,
  4  orderDate varchar2(8) not null,
  5  orderPrice number(15,2) not null,
  6  city varchar(256),
  7  endDate varchar2(8),
  8  payPrice number,
  9  salePrice number,
 10  point number,
 11  primary key(orderNo));

테이블이 생성되었습니다.
```

기본 SQL 작성하기

- DDL 활용(학습모듈 p.22~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서.

2. SELECT를 이용하여 테이블을 생성한다.

✓ 만일 생성하고자 하는 '기존테이블'과 동일한 컬럼을 가진 '신규테이블'이 있다면 다음과 같이 기존에 존재하는 테이블 정보를 이용하여 새로운 테이블을 만들 수 있다.

```
CREATE TABLE 신규테이블 AS SELECT * FROM 기존테이블;
```

```
SQL> create table ordertbl1 as select * from ordertbl;

테이블이 생성되었습니다.

SQL>
```

기본 SQL 작성하기

- DDL 활용(학습모듈 p.22~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서.

✓ SELECT 문을 이용해 테이블을 생성하는 방식은 <표 1-7>과 같은 테이블 생성 방법 대비 다음과 같은 특징이 있음에 유의한다.

- 생성된 테이블은 기존 테이블의 컬럼 및 데이터 유형과 길이 등을 그대로 적용함. •
- NOT NULL의 정의는 그대로 적용함.
- 제약 조건은 적용되지 않음.
- ALTER TABLE을 사용하여 제약 조건을 추가해야 함.
- 동일한 컬럼들로 생성된 경우 `*'를 사용함
- 필요한 컬럼만을 지정하여 테이블을 생성할 수 있음

기본 SQL 작성하기

- DDL 활용(학습모듈 p.23~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서.

3. 생성한 테이블을 확인한다.

✓ 테이블의 생성 유무를 확인하기 위해 다음과 같은 명령어를 사용한다.

<표 1-8> 테이블 확인 SQL문

DBMS 제품	테이블 목록 확인 SQL문
대표적인 상용 O 제품	SELECT * FROM user_tables;
대표적인 비상용 M 제품	SHOW TABLES;

기본 SQL 작성하기

- DDL 활용(학습모듈 p.23~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서.

3. 생성한 테이블을 확인한다.

```
SQL> select * from tab;

TNAME                          TABTYPE  CLUSTERID
------------------------------ -------  ----------
BIN$ni3s6PMTQJyAeh8et6Lwjw==$0 TABLE
BONUS                          TABLE
CAR_MEMBER                     TABLE
COURSE_TBL                     TABLE
DEPT                           TABLE
DEPTT                          TABLE
DEPT_TABLE                     TABLE
EMP                            TABLE
LECTURER_TBL                   TABLE
MEMBER                         TABLE
ORDERTBL                       TABLE

TNAME                          TABTYPE  CLUSTERID
------------------------------ -------  ----------
ORDERTBL1                      TABLE
SALGRADE                       TABLE
TEST1                          TABLE

14 개의 행이 선택되었습니다.
```

기본 SQL 작성하기

- DDL 활용(학습모듈 p.23~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서.

➢ 결과에 앞서 생성한 `주문테이블'이 있으면 테이블 생성이 성공한 것이다. 또는 다음 명령어를 통해 테이블 구조를 확인할 수 있다.

<표 1-9> 테이블 구조 보기

구분	내용
테이블 구조 보기 명령	DESC '주문테이블';

```
SQL> desc ordertbl;
 이름                                      널?       유형
 ----------------------------------------- --------  ----------------------
 ORDERNO                                   NOT NULL  VARCHAR2(16)
 CUSTOMNO                                  NOT NULL  VARCHAR2(16)
 ORDERDATE                                 NOT NULL  VARCHAR2(8)
 ORDERPRICE                                NOT NULL  NUMBER(15,2)
 CITY                                                VARCHAR2(256)
 ENDDATE                                             VARCHAR2(8)
 PAYPRICE                                            NUMBER
 SALEPRICE                                           NUMBER
 POINT                                               NUMBER
```

기본 SQL 작성하기

- DDL 활용(학습모듈 p.23~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서

4. CREATE 문에 사용된 제약 조건을 조사한다.

기본 SQL 작성하기

- DDL 활용(학습모듈 p.24~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서

[4] 테이블 구조를 변경한다.

1. 테이블 구조 변경 대상을 정의한다.
2. 컬럼 이름과 타입을 변경한다.

✓ 앞에서 생성한 `city'를 `cityCode'로 속성 이름을 바꾸어 보자

```
SQL> ALTER TABLE ordertbl RENAME COLUMN city TO cityCode;

테이블이 변경되었습니다.
```

✓ 'cityCode' 새로운 데이터 타입을 정수형으로 정의한다.

```
SQL> ALTER TABLE ordertbl modify  citycode  number;

테이블이 변경되었습니다.

SQL>
```

기본 SQL 작성하기

- DDL 활용(학습모듈 p.24~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서

➢ 열의 크기 확대는 별 제한이 없지만 축소나 데이터 타입 변환은 열의 모든 데이터가 NULL 값을 가지는 경우에만 허용된다.

➢ DEFAULT 값도 수정할 수 있다.

➢ DEFAULT 값을 수정하면 수정 이후에 입력되는 값에만 수정된 DEFAULT 값이 적용된다.

기본 SQL 작성하기

- DDL 활용(학습모듈 p.24~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서

3. 컬럼을 추가하고 삭제한다.

➢ 다음과 같은 명령어를 통해 컬럼의 추가 및 삭제가 가능하다.

<표 1-11> 테이블 변경을 위한 SQL문

구분		SQL문 형식
추가	맨 뒤에	ALTER TABLE '테이블명' ADD '컬럼명' 자료형;
삭제		ALTER TABLE '테이블명' DROP '컬럼명';

➢ `ALTER TABLE 테이블명 ADD 컬럼명 자료형' 문을 사용하여 테이블에 열을 추가하면 열은 테이블의 마지막에 추가된다.

➢ 테이블이 가지는 열의 개수에는 제한이 있어서 추가가 안 될 수도 있다.

기본 SQL 작성하기

- DDL 활용(학습모듈 p.24~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서

3. 컬럼을 추가하고 삭제한다.

➢ 다음과 같은 명령어를 통해 컬럼의 추가 및 삭제가 가능하다.

```
SQL> ALTER TABLE ordertbl add  cityaddress varchar(80);
테이블이 변경되었습니다.
```

```
SQL> ALTER TABLE ordertbl drop column cityaddress;
테이블이 변경되었습니다.
SQL>
```

기본 SQL 작성하기

- DDL 활용(학습모듈 p.25~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서

5 제약 조건을 변경한다

✓ [제약조건] : Constraint

1. 데이터의 무결성을 위해서 사용
2. 제양조건은 테이블정의시 필드에 설정
3. 종류
   1) not null : 필수입력
   2) unique : 고유해야된다. (도메인내에 동일한 값을 저장할 수 없다)
   3) check : 유효성(범위, 값..) <=테이블내에서 검색조건을 부여
   4) foreign key : 다른테이블에서 검색조건 검사
   5) primary key : not null + unique + index 결합형태
   6) default : 기본값이 입력

기본 SQL 작성하기

- DDL 활용(학습모듈 p.25~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서

5 제약 조건을 변경한다

✓ [제약조건] : Constraint

1. 데이터의 무결성을 위해서 사용
2. 제양조건은 테이블정의시 필드에 설정
3. 종류
   1) not null : 필수입력
   2) unique : 고유해야된다. (도메인내에 동일한 값을 저장할 수 없다)
   3) check : 유효성(범위, 값..) <=테이블내에서 검색조건을 부여
   4) foreign key : 다른테이블에서 검색조건 검사
   5) primary key : not null + unique + index 결합형태
   6) default : 기본값이 입력

기본 SQL 작성하기

- DDL 활용(학습모듈 p.25~)

▣ DDL 활용하기

✓ 주문 정보, 회원 정보 관련 데이터 요구사항

✓ 수행 순서

```
SQL> alter table ordertbl add constraint citycode  unique(citycode);

테이블이 변경되었습니다.
```

```
SQL> alter table ordertbl add constraint citycode_check
  2  check(citycode between 100 and 900);

테이블이 변경되었습니다.
```

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.27~)

▣ DML 활용하기

✓ 학습 목표 • 한 개의 테이블에 대해 데이터를 삽입, 수정, 삭제하고 행을 조회하는 DML(Data Manipulation Language) 명령문을 작성할 수 있다.

Ⅰ DML 개요

1. DML 의미

✓ 데이터를 조작하는 명령어를 DML(Data Manipulation Language)이라고 한다. 여기서 조작은 데이터 관점에서 생명 주기를 제어하는 것을 의미한다.

2. DML 유형

✓ 데이터의 생명 주기 관리 및 활용을 위해 사용하는 DML 명령어는 다음과 같다.

<표 1-13> DML 명령어

구분	DML 명령어	내용
데이터 생성	INSERT	삽입 형태로 신규 데이터를 테이블에 저장
데이터 조회	SELECT	테이블의 내용을 조회
데이터 변경	UPDATE	테이블의 내용을 변경
데이터 삭제	DELETE	테이블의 내용을 삭제

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.27~)

▣ DML 활용하기

2 DML 명령문

1. 데이터 삽입(INSERT)

✓ 데이터를 삽입하기 위한 명령어로 다음과 같이 두 가지 형태의 명령문 형식을 제공한다.

✓ 이때 데이터 삽입 결과로 하나의 레코드가 추가된다. 따라서 삽입에 사용되는 정보는 하나의 레코드를 충분히 묘사해야 한다.

<표 1-14> INSERT 명령문 유형

형태	INSERT 명령문
A	INSERT INTO table_name (column1, column2, ..) VALUES (value1, value2, ...);
B	INSERT INTO table_name VALUES (value1, value2, ...);

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.28~)

▣ DML 활용하기

2 DML 명령문

2. 데이터 조회(SELECT)

✓ 데이터의 내용을 조회할 때 사용하는 명령어이다. 가장 많이 사용되는 SQL 명령어로서, 다른 DML 명령어와 같이 사용되어 SQL의 활용을 풍부하게 한다.

✓ SELECT 명령어의 기본 형식은 다음과 같다.

```
SELECT [OPTION] columns FROM table [WHERE 절] ;
```

✓ SELECT 문에 사용되는 각 정보는 다음과 같다.

<표 1-15> SELECT 문에서 사용되는 요소

SELECT 문 요소	요소값	내용
OPTION	• ALL • DISTINCT	• 중복 포함한 조회 결과 출력 • 중복 제거한 조회 결과 출력
columns	• 컬럼명 목록 • 와이드카드	• SELECT 통해 조회할 컬럼명 지정 • 모두 또는 전체를 의미하는 *

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.28~)

▣ DML 활용하기

2 DML 명령문

3. 데이터 수정(UPDATE)

데이터를 수정할 때 다음과 같은 형태의 UPDATE 명령문을 사용한다.

```
UPDATE table SET column1 = value1, column2 = value2, ... [WHERE 절] ;
```

UPDATE 명령문은 보통 WHERE 절을 통해 어떤 조건이 만족할 경우에만 특정 컬럼의 값을 수정하는 용도로 많이 사용된다.

4. 데이터 삭제(DELETE)

레코드를 삭제할 때 다음과 같은 형태의 DELETE 명령문을 사용한다.

```
DELETE  FROM  table [WHERE 절] ;
```

조건절 없이 DELETE를 사용하는 경우, 테이블 전체가 한 번에 삭제되는 위험이 있다.

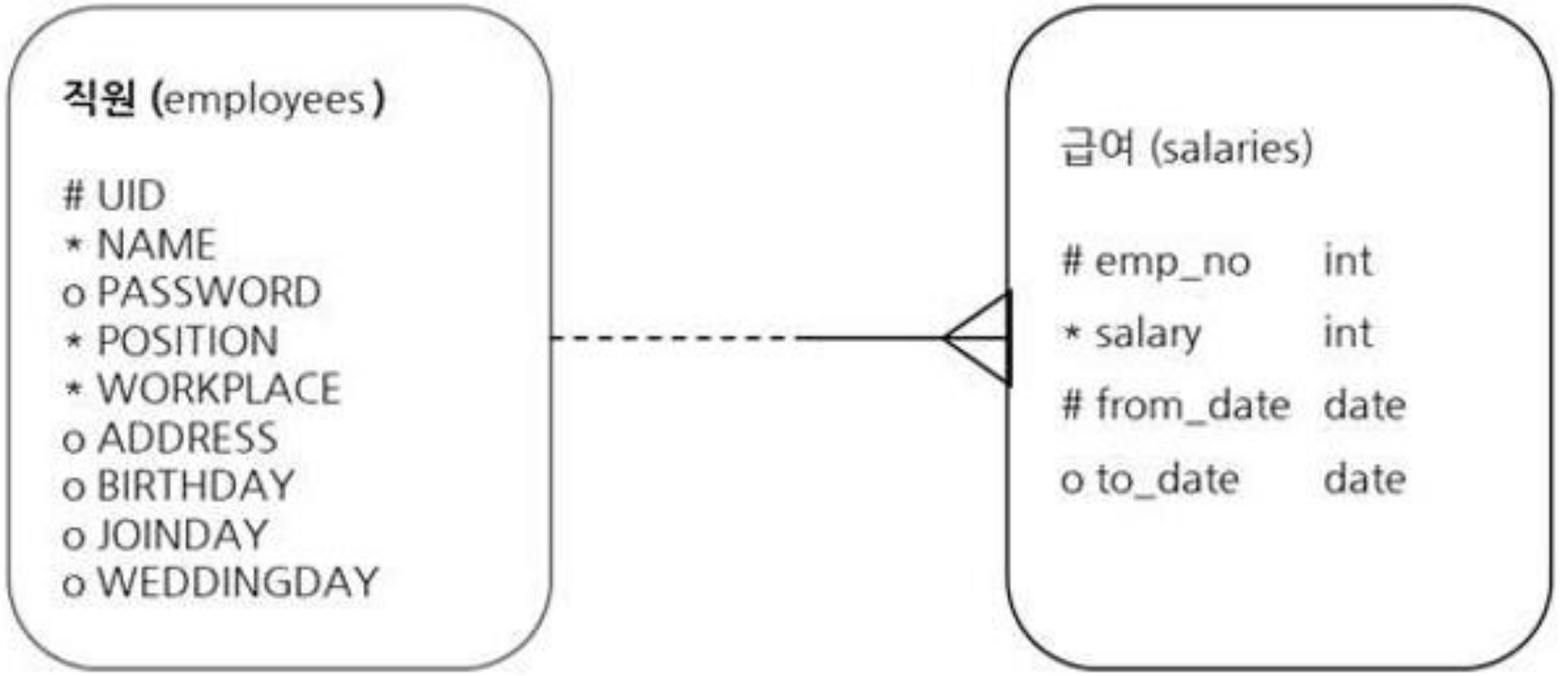
기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.30~)

▣ 수행 내용 / DML 활용하기

□ 테이블 설계서를 준비한다.

1. 수행에 사용할 테이블을 생성한다



[그림 1-7] DML 학습을 위한 대상 테이블의 개념 모델링 모습

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.30~)

▣ 수행 내용 / DML 활용하기

□ 테이블 설계서를 준비한다.

1. 수행에 사용할 테이블을 생성한다.

✓ 급여 테이블을 생성하기 위해 다음과 같은 SQL문을 사용한다.

```
SQL> create table employees(
  2  userID number not null,
  3  name varchar2(20) not null,
  4  password varchar2(20),
  5  position varchar2(40),
  6  workplace varchar2(40),
  7  address varchar2(80),
  8  birthday date,
  9  joinday date,
 10  weddingday date,
 11  primary key(userID));

테이블이 생성되었습니다.
```

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.30~)

▣ 수행 내용 / DML 활용하기

Ⅰ 테이블 설계서를 준비한다.

1. 수행에 사용할 테이블을 생성한다.

✓ 급여 테이블을 생성하기 위해 다음과 같은 SQL문을 사용한다.

```
SQL> create table salaries(
  2  emp_no number not null,
  3  salary number not null,
  4  from_date date not null,
  5  to_date date,
  6  foreign key(emp_no) references employees(userID) on delete cascade,
  7  primary key(emp_no,from_date));

테이블이 생성되었습니다.

SQL>
```

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.31~)

▣ 수행 내용 / DML 활용하기

2 데이터를 삽입한다.

1. 단일 레코드를 삽입한다.

✓ <표 1-14>의 삽입 명령문 유형 가운데 형태 A로, 하나의 레코드를 추가하는 방법은 다음과 같다.

```
SQL> insert into employees values(1001,'홍길동','1234','부산','양정',
  2  '부산시 부산진구 양정동 900','1999-09-09','2018-01-01','');

1 개의 행이 만들어졌습니다.

SQL>
```

```
SQL> insert into salaries (emp_no,salary,from_date)
  2  values(1001,4000000,'2019-08-01');

1 개의 행이 만들어졌습니다.

SQL>
```

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.31~)

▣ 수행 내용 / DML 활용하기

2 데이터를 삽입한다.

1. 단일 레코드를 삽입한다

```
SQL> insert into salaries values(1001,4000000,'2019-07-01');
insert into salaries values(1001,4000000,'2019-07-01')
            *
1행에 오류:
ORA-00947: 값의 수가 충분하지 않습니다


SQL>
```

✓ 테이블에 정의된 컬럼은 4개인데, 입력값은 3개뿐이라 수행 중 오류가 발생한다.

✓ 하지만 테이블 생성 과정에서 `to_date' 컬럼에 Default 값을 지정하면 오류가 발생하지 않는다.

✓ 현재 상황에서 오류 없이 입력하기 위해서는 다음과 같이 4개의 값을 지정해야 한다.

```
SQL> insert into salaries values(1001,4000000,'2019-07-01',null);

1 개의 행이 만들어졌습니다.

SQL>
```

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.31~)

▣ 수행 내용 / DML 활용하기

2 데이터를 삽입한다.

```
SQL> insert into salaries (emp_no,salary,from_date,to_date) values
  2  (1001,200000,'2012-07-01','2013-06-30');

1 개의 행이 만들어졌습니다.

SQL> insert into salaries (emp_no,salary,from_date,to_date) values
  2  (1001,300000,'2013-07-01','2014-06-30');

1 개의 행이 만들어졌습니다.

SQL> insert into salaries (emp_no,salary,from_date,to_date) values
  2  (1001,400000,'2014-07-01','2015-06-30');

1 개의 행이 만들어졌습니다.

SQL> insert into salaries (emp_no,salary,from_date,to_date) values
  2  (1001,450000,'2015-07-01','2016-06-30');

1 개의 행이 만들어졌습니다.

SQL>
```

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.32~)

▣ 수행 내용 / DML 활용하기

3 데이터를 조회하고 수정한다.

1. 데이터를 조회한다.

삽입한 데이터를 조회하기 위해 다음과 같은 명령문을 사용한다.

```
SELECT * FROM salaries ;
```

급여 (salaries) 테이블의 모든 컬럼에 대해 아무런 조건 없이 검색하는 기본적인 조회 명령문이다. 데이터가 많아서 일부 데이터만 조회하기 위해서는 조건절이 필요하다. 또 컬럼 가운데 일부만을 보고 싶은 경우에는 와일드카드(*) 대신에 특정 컬럼을 명시한다.
직원 번호가 1001인 사원의 급여 테이블 정보를 보기 위해서는 다음과 같이 조건을 명기한다.

```
SELECT * FROM salaries WHERE emp_no = 1001;
```

직원 번호 1001번 직원의 급여 테이블 가운데 직원 번호와 급여만을 보고 싶은 경우에는 다음과 같이 특정 컬럼만을 명기한다.

```
SELECT emp_no, salary FROM salaries WHERE emp_no = 1001;
```

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.33~)

▣ 수행 내용 / DML 활용하기

2. 데이터를 수정한다.

데이터를 수정하려면 다음과 같은 정보를 결정해야 한다.

- Condition: 특정 조건 지정
- What: 특정 대상 - 컬럼 지정
- How: 값 - 값 지정

이와 같은 정보를 표현하는 SQL문의 실제 사례는 다음과 같다.

```
UPDATE salaries SET salary = 1100000
    WHERE emp_no=1001 and from_date = '2014-07-01';
```

이 명령문에서 조건은 WHERE 절에 해당하며, 특정 대상과 값은 SET 절에 해당한다.

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.34~)

▣ 수행 내용 / DML 활용하기

④ 데이터를 삭제한다.

1. 데이터를 삭제한다.

[그림 1-7]과 같은 개념 모델링 정보로 만들어진 테이블 구조에서 종속적 개념을 가지는 급여 테이블(salaries)의 특정 항목을 지우는 것은 용이하다. 다음과 같이 그 대상을 조건절에 명시하면 된다.

```
DELETE  FROM  salaries WHERE emp_no=1001;
```

2. 데이터 삭제 오류가 발생한 이유를 조사한다.

다음과 같은 삭제 명령문을 수행해 보자. 급여 테이블이 아닌 직원 테이블에 있는 특정 직원의 레코드이다.

기본 SQL 작성하기

1-2. DML 활용 (학습모듈 p.31~)

▣ 수행 내용 / DML 활용하기

④ 데이터를 삭제한다.

1. 데이터를 삭제한다.

[그림 1-7]과 같은 개념 모델링 정보로 만들어진 테이블 구조에서 종속적 개념을 가지는 급여 테이블(salaries)의 특정 항목을 지우는 것은 용이하다. 다음과 같이 그 대상을 조건절에 명시하면 된다.

```
DELETE  FROM  salaries WHERE emp_no=1001;
```

2. 데이터 삭제 오류가 발생한 이유를 조사한다.

다음과 같은 삭제 명령문을 수행해 보자. 급여 테이블이 아닌 직원 테이블에 있는 특정 직원의 레코드이다.