04

CHAPTER

데이터 삽입, 수정, 삭제



1-1 SQL 문의 종류

- DML(Data Manipulation Language, 데이터 조작어)
 - 데이터를 검색 및 삽입, 수정, 삭제하는데 사용하는 언어
 - SELECT 문, INSERT, UPDATE, DELETE 문 등
- DDL(Data Definition Language, 데이터 정의어)
 - 데이터베이스, 테이블, 뷰, 인덱스 등의 데이터베이스 개체를 생성, 삭제, 변경하는 데 사용하는 언어
 - CREATE, DROP, ALTER, TRUNCATE 문 등
- DCL(Data Control Language, 데이터 제어어)
 - 사용자에게 어떤 권한을 부여하거나 빼앗을 때 사용하는 언어
 - GRANT, REVOKE, DENY 문 등

- INSERT 문
 - 테이블에 데이터를 삽입하는 명령어

INSERT [INTO] 테이블이름[(열1, 열2, ...)] VALUES (값1, 값2, ...)

■ INSERT 문에서 테이블 이름 다음에 나오는 열 생략 가능(단 열의 순서 및 개수는 동일해야 함)

```
USE cookDB;
CREATE TABLE testTBL1 (id int, userName char(3), age int);
INSERT INTO testTBL1 VALUES (1, '뽀로로', 16);
```

■ id와 이름만 입력하고 나이는 입력하고 싶지 않다면

INSERT INTO testTBL1 (id, userName) VALUES (2, '크롱');

■ 열의 순서를 바꾸어 입력하고 싶을 때

INSERT INTO testTBL1 (userName, age, id) VALUES ('루피', 14, 3);

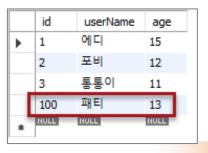
- AUTO_INCREMENT 키워드
 - 자동으로 1부터 증가하는 값을 입력하는 키워드
 - 특정 열을 AUTO_INCREMENT로 지정할 때는 반드시 PRIMARY KEY(기본키) 또는 UNIQUE(유일한 값)로 설정해야 함
 - 데이터 형식이 숫자인 열에만 사용 가능
 - AUTO_INCREMENT로 지정된 열은 INSERT 문에서 NULL 값으로 지정하면 자동으로 값이 입력됨

```
USE cookDB;
CREATE TABLE testTBL2
( id int AUTO_INCREMENT PRIMARY KEY,
    userName char(3),
    age int
);
INSERT INTO testTBL2 VALUES (NULL, '에디', 15);
INSERT INTO testTBL2 VALUES (NULL, '포비', 12);
INSERT INTO testTBL2 VALUES (NULL, '통통이', 11);
SELECT * FROM testTBL2;
```

	id	userName	age
Þ	1	에디	15
	2	포비	12
	3	통통이	11
	NULL	NULL	NULL

■ AUTO_INCREMENT 입력 값을 100부터 시작하도록 변경하고 싶다면

ALTER TABLE testTBL2 AUTO_INCREMENT=100; INSERT INTO testTBL2 VALUES (NULL, '패티', 13); SELECT * FROM testTBL2;



■ 대량 데이터 삽입 형식

```
INSERT INTO 테이블이름 (열1, 열2, ...)
SELECT 문;
```

■ employees 테이블의 데이터를 가져와 testTBL4 테이블에 입력

```
USE cookDB;
CREATE TABLE testTBL4 (id int, Fname varchar(50), Lname varchar(50));
INSERT I NTO testTBL4
SELECT emp_no, first_name, last_name FROM employees.employees;
```



■ 아예 테이블 정의까지 생략하고 싶다면 CREATE TABLE ... SELECT 문 사용

CREATE TABLE testTBL5 (SELECT emp_no, first_name, last_name FROM employees.employees); SELECT * FROM testTBL5 LIMIT 3;

	emp_no	first_name	last_name
•	10001	Georgi	Facello
	10002	Bezalel	Simmel
	10003	Parto	Bamford

■ CREATE TABLE ... SELECT 문에서 열 이름을 바꾸어 테이블을 생성하려면

CREATE TABLE testTBL6

(SELECT emp_no AS id, first_name AS Fname, last_name AS Lname FROM employees.employees);

SELECT * FROM testTBL6 LIMIT 3;

	id	Fname	Lname
▶*	10001	Georgi	Facello
	10002	Bezalel	Simmel
	10003	Parto	Bamford

1-3 UPDATE 문

- UPDATE 문
 - 테이블에 입력되어 있는 값을 수정하는 명령어

```
UPDATE 테이블이름
SET 열1=값1, 열2=값2, ...
WHERE 조건;
```

■ 'Kyoichi'의 Lname을 '없음'으로 수정

```
USE cookDB;
UPDATE testTBL4
SET Lname = '없음'
WHERE Fname = 'Kyoichi';
```

■ 전체 테이블의 내용을 수정하고 싶을 때는 WHERE 절 생략

```
UPDATE buyTBL
SET price = price * 1.5;
```

1-4 DELETE 문

- DELETE 문
 - 테이블에 데이터를 행 단위로 삭제하는 명령어

DELETE FROM 테이블이름 WHERE 조건;

■ DELETE 문에서 WHERE 절을 생략하면 테이블에 저장된 전체 데이터가 삭제

USE cookDB; DELETE FROM testTBL4 WHERE Fname = 'Aamer';

■ Aamer 중에서 상위 몇 건만 삭제하고자 할 때는 추가로 LIMIT 절 사용

DELETE FROM testTBL4 WHERE Fname = 'Aamer' LIMIT 5;

[실습 4-1] 대용량 테이블 삭제하기

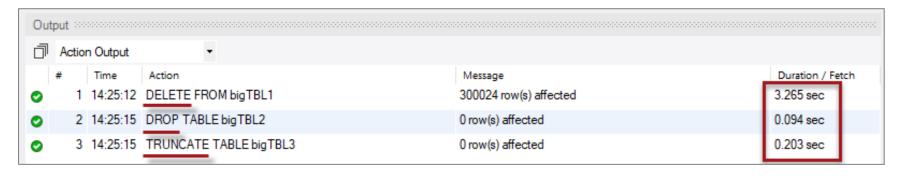
- 1 대용량 테이블 생성
 - 1-1 대용량 테이블 3개 생성

```
USE cookDB;
CREATE TABLE bigTBL1 (SELECT * FROM employees.employees);
CREATE TABLE bigTBL2 (SELECT * FROM employees.employees);
CREATE TABLE bigTBL3 (SELECT * FROM employees.employees);
```

- 2 데이터 삭제하기
 - 2-1 DELETE, DROP, TRUNCATE 문으로 3개의 테이블 삭제

```
DELETE FROM bigTBL1;
DROP TABLE bigTBL2;
TRUNCATE TABLE bigTBL3;
```

- 3 결과 확인하기
 - 3-1 [Output] 창의 결과에서 실행 시간 확인



1-4 DELETE 문

- 실행 시간을 고려한 테이블 삭제 방법
 - 대용량 테이블 전체 내용을 삭제할 때 테이블 자체가 필요 없는 경우에는 DROP 문 사용
 - 테이블의 구조를 남겨놓고 싶은 경우에는 TRUNCATE 문으로 삭제