HAPPY HACKING

Database

Database

HAPPY HACKING

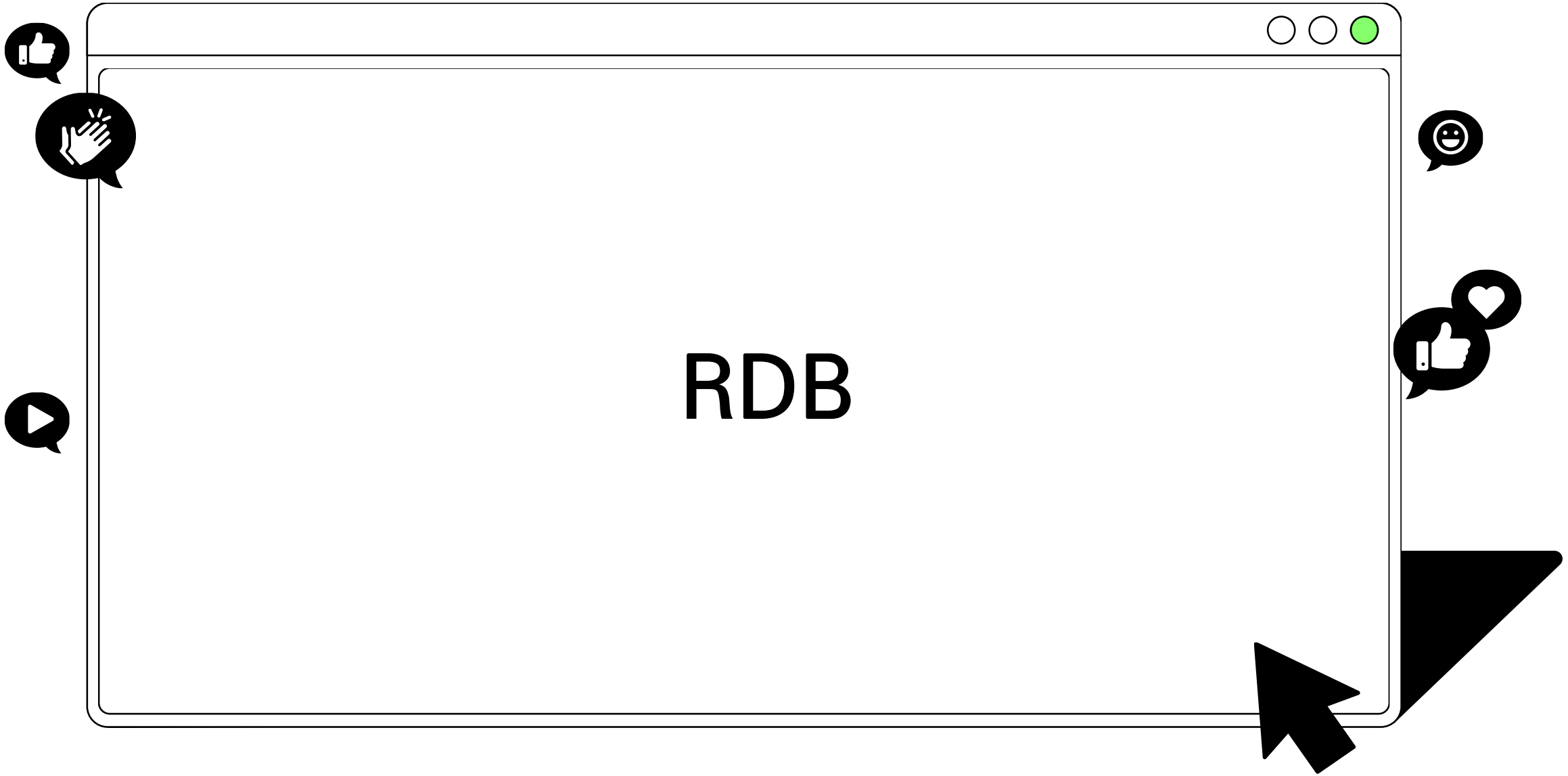
- 데이터베이스는 체계화된 데이터의 모임
- 여러 사람이 공유하고 사용할 목적으로 통합 관리되는 정보의 집합
- 논리적으로 연관된 (하나 이상의) 자료의 모음으로
  그 내용을 고도로 구조화 함으로써 검색과 갱신의 효율화를 꾀한 것
- 즉, 몇 개의 자료 파일을 조직적으로 통합하여
  자료 항목의 중복을 없애고 자료를 구조화하여 기억시켜 놓은 자료의 집합체

https://en.wikipedia.org/wiki/Database

HAPPY HACKING

- 데이터베이스로 얻는 장점들
  - 데이터 중복 최소화
  - 데이터 무결성 (정확한 정보를 보장)
  - 데이터 일관성
  - 데이터 독립성 (물리적 / 논리적)
  - 데이터 표준화
  - 데이터 보안 유지

HAPPY HACKING



RDB

RDB

HAPPY HACKING

- 관계형 데이터베이스 (RDB, Relational Database)
  - 서로 관련된 데이터를 저장하고 접근할 수 있는 데이터베이스 유형
  - 키(key)와 값(value)들의 간단한 관계(relation)를 표(table) 형태로 정리한 데이터베이스

고유 번호	이름	주소	나이
1	홍길동	제주	20
2	김길동	서울	30
3	박길동	독도	40

- 스키마 (schema)
  - 데이터베이스에서 자료의 구조, 표현방법, 관계등 전반적인 명세를 기술한 것

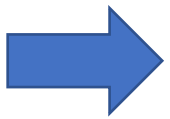
column	datatype
id	INT
name	TEXT
address	TEXT
age	INT

HAPPY HACKING

- 테이블 (table)
  - 열(컬럼/필드)과 행(레코드/값)의 모델을 사용해 조직된 데이터 요소들의 집합

column	datatype
id	INT
name	TEXT
address	TEXT
age	INT

Schema



id	name	address	age
1	홍길동	제주	20
2	김길동	서울	30
3	박길동	독도	40

Table

- 열 (column) : 각 열에 고유한 데이터 형식 지정
  - 아래의 예시에서는 name이란 필드에 고객의 이름(TEXT) 정보가 저장

id	name	address	age
1	홍길동	제주	20
2	김길동	서울	30
3	박길동	독도	40

HAPPY HACKING

- 행 (row) : 실제 데이터가 저장되는 형태
  - 아래의 예시에서는 총 3명의 고객정보가 저장되어 있음(레코드가 3개)

id	name	address	age
1	홍길동	제주	20
2	김길동	서울	30
3	박길동	독도	40

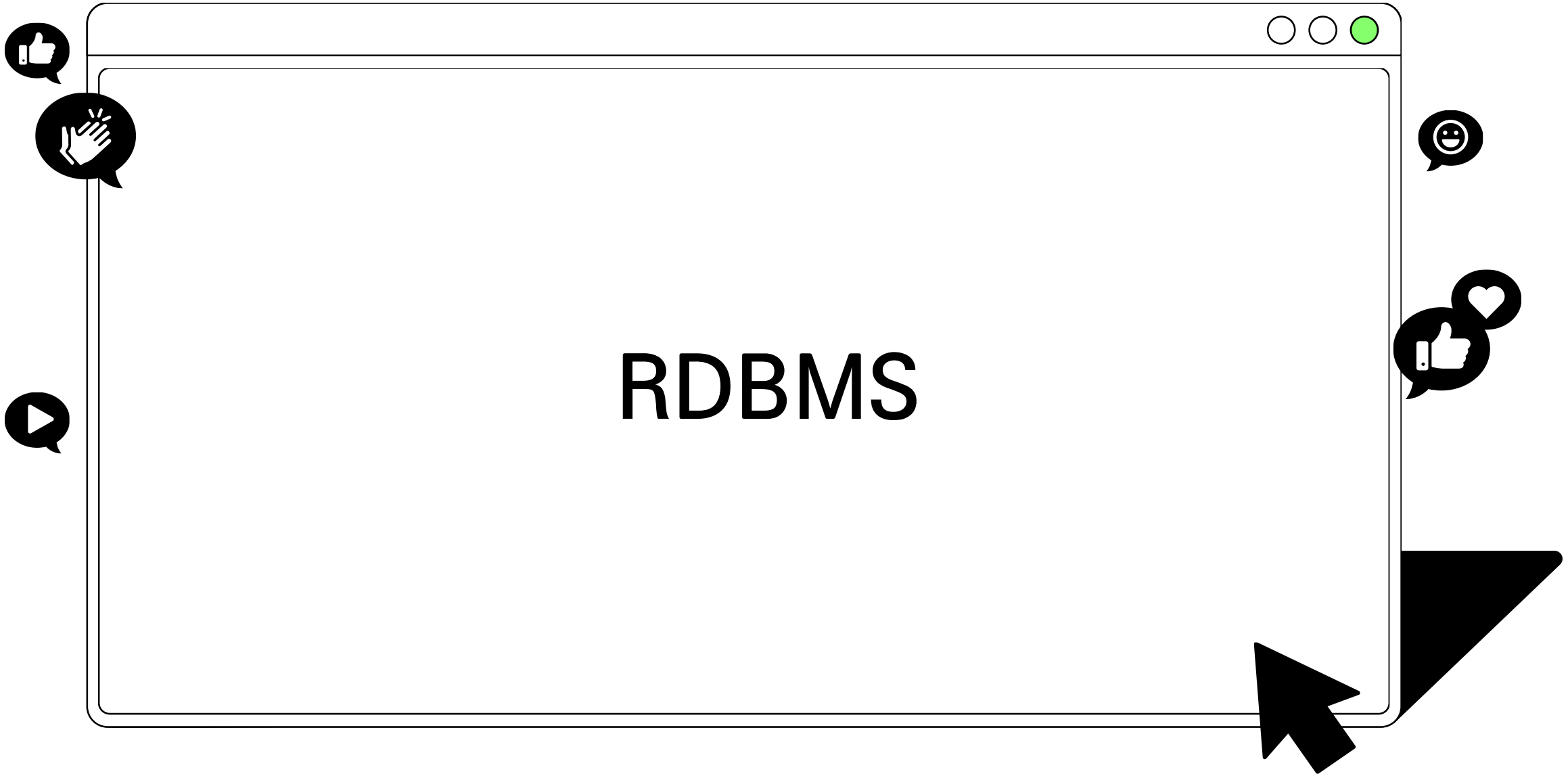
행, 로우, 레코드 등의 이름으로 불림

- 기본키 (Primary Key) : 각 행(레코드)의 고유 값
  - 반드시 설정해야 하며, 데이터베이스 관리 및 관계 설정 시 주요하게 활용 됨

PK, 기본키

id	name	address	age
1	홍길동	제주	20
2	김길동	서울	30
3	박길동	독도	40

HAPPY HACKING



RDBMS

RDBMS

HAPPY HACKING

- 관계형 데이터베이스 관리 시스템 (RDBMS)
  - 관계형 모델을 기반으로 하는 데이터베이스 관리시스템을 의미

- SQLite
  - 서버 형태가 아닌 파일 형식으로 응용 프로그램에 넣어서 사용하는 비교적 가벼운 데이터베이스
  - 구글 안드로이드 운영체제에 기본적으로 탑재된 데이터베이스이며, 임베디드 소프트웨어에도 많이 활용됨
  - 로컬에서 간단한 DB 구성을 할 수 있으며, 오픈소스 프로젝트이기 때문에 자유롭게 사용가능

HAPPY HACKING

- SQLite Data Type
  1. NULL
  2. INTEGER
     - 크기에 따라 0, 1, 2, 3, 4, 6 또는 8바이트에 저장된 부호 있는 정수
  3. REAL
     - 8바이트 부동 소수점 숫자로 저장된 부동 소수점 값
  4. TEXT
  5. BLOB
     - 입력된 그대로 정확히 저장된 데이터 (별다른 타입 없이 그대로 저장)

https://www.sqlite.org/datatype3.html

- Sqlite Type Affinity (1/2)
  - 특정 컬럼에 저장하도록 권장하는 데이터 타입

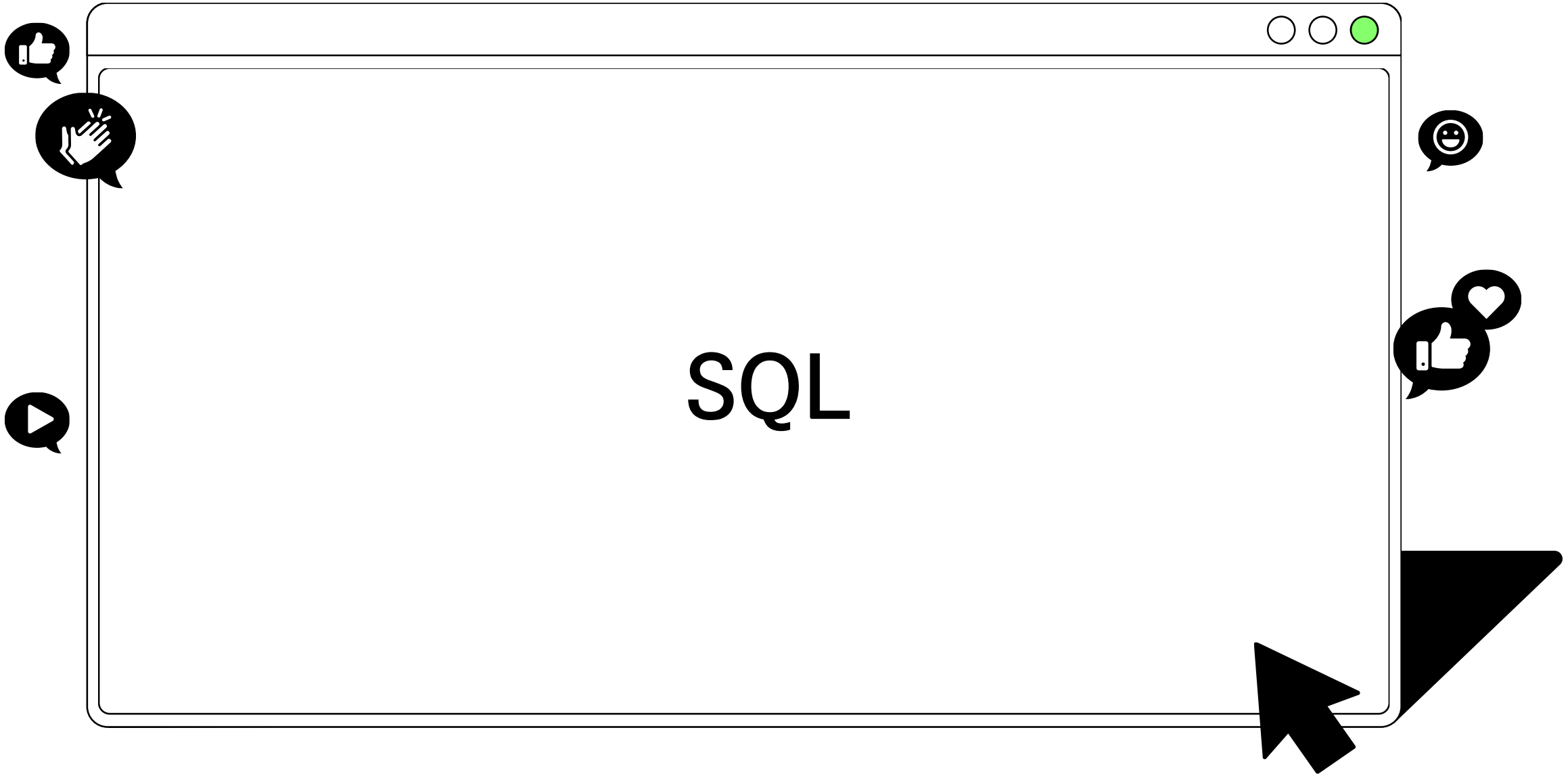
1. INTEGER
2. TEXT
3. BLOB
4. REAL
5. NUMERIC

RDBMS

HAPPY HACKING

Example Typenames From The CREATE TABLE Statement	Resulting Affinity
INT INTEGER TINYINT SMALLINT MEDIUMINT BIGINT UNSIGNED BIG INT INT2 INT8	INTEGER
CHARACTER(20) VARCHAR(255) VARYING CHARACTER(255) NCHAR(55) NATIVE CHARACTER(70) NVARCHAR(100) TEXT CLOB	TEXT
BLOB (no datatype specified)	BLOB
REAL DOUBLE DOUBLE PRECISION FLOAT	REAL
NUMERIC DECIMAL(10,5) BOOLEAN DATE DATETIME	NUMERIC

HAPPY HACKING



SQL

SQL

HAPPY HACKING

- SQL (Structured Query Language)
  - 관계형 데이터베이스 관리시스템의 데이터 관리를 위해 설계된 특수 목적으로 프로그래밍 언어
  - 데이터베이스 스키마 생성 및 수정
  - 자료의 검색 및 관리
  - 데이터베이스 객체 접근 조정 관리

SQL

HAPPY HACKING

분류	개념	예시
DDL – 데이터 정의 언어 (Data Definition Language)	관계형 데이터베이스 구조(테이블, 스키마)를 정의하기 위한 명령어	CREATE DROP ALTER
DML – 데이터 조작 언어 (Data Manipulation Language)	데이터를 저장, 조회, 수정, 삭제 등을 하기 위한 명령어	INSERT SELECT UPDATE DELETE
DCL – 데이터 제어 언어 (Data Control Language)	데이터베이스 사용자의 권한 제어를 위해 사용하는 명령어	GRANT REVOKE COMMIT ROLLBACK

- SQL Keywords – Data Manipulation Language
  - INSERT : 새로운 데이터 삽입(추가)
  - SELECT : 저장되어있는 데이터 조회
  - UPDATE : 저장되어있는 데이터 갱신
  - DELETE : 저장되어있는 데이터 삭제

HAPPY HACKING

Hello World!

HAPPY HACKING

데이터베이스 생성하기

```
$ sqlite3 tutorial.sqlite3
sqlite> .database
```

‘.’ 은 sqlite에서 활용되는 명령어

HAPPY HACKING

csv 파일을 table로 만들기

```
sqlite> .mode csv
sqlite> .import hellodb.csv examples
sqlite> .tables
examples
```

HAPPY HACKING

SELECT

```
SELECT * FROM examples;
```

HAPPY HACKING

SELECT 확인하기

```
sqlite> SELECT * FROM examples;

1,"길동","홍",600,"충청도",010-0000-0000
```

SELECT 문은 특정 테이블의 레코드(행) 정보를 반환!

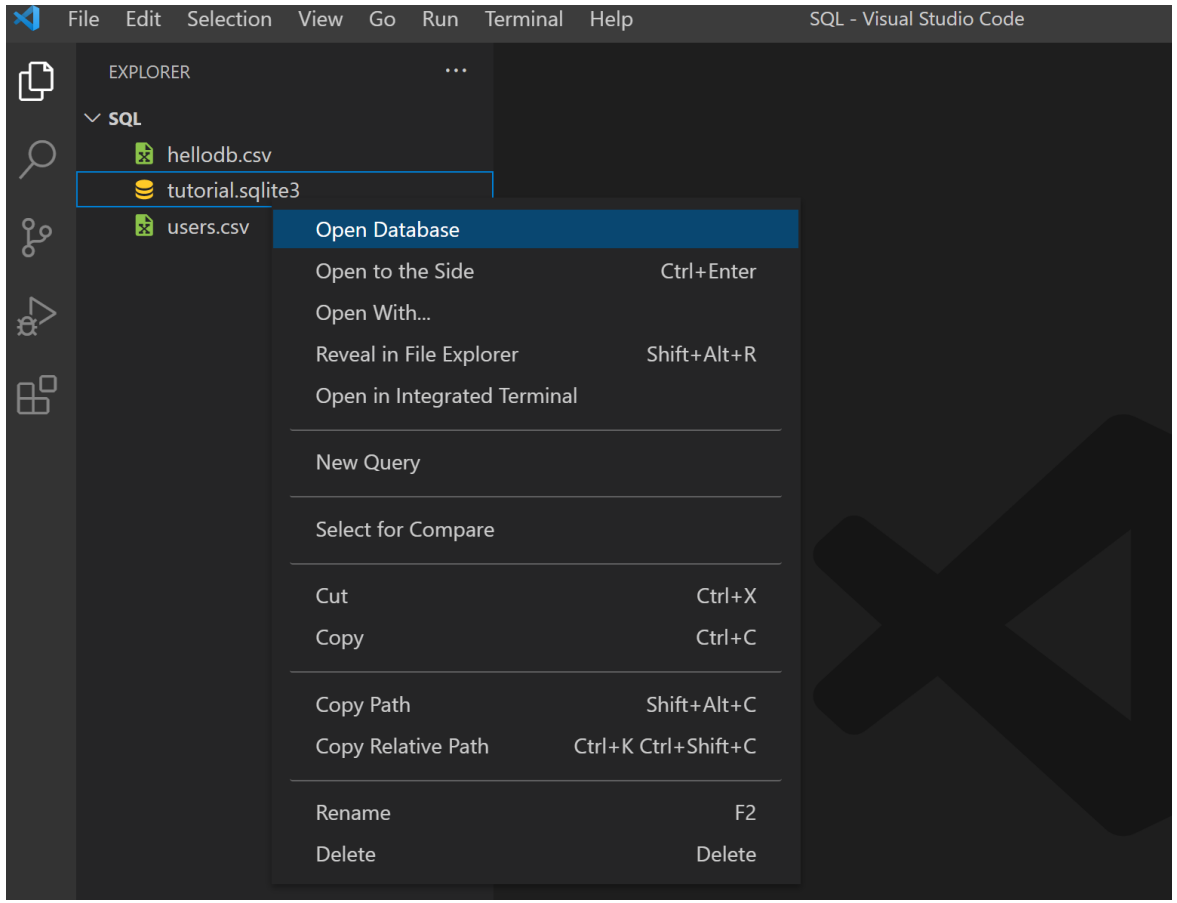
HAPPY HACKING

(Optional) 터미널 view 변경하기

```
sqlite> SELECT * FROM examples;
1,"길동","홍",600,"충청도",010-0000-0000
sqlite> .headers on
sqlite> SELECT * FROM examples;
id,first_name,last_name,age,country,phone
1,"길동","홍",600,"충청도",010-0000-0000
sqlite> .mode column
sqlite> SELECT * FROM examples;
id  first_name  last_name  age  country  phone
--  ----------  ---------  ---  -------  -------------
1   길동          홍          600  충청도      010-0000-0000
```

HAPPY HACKING

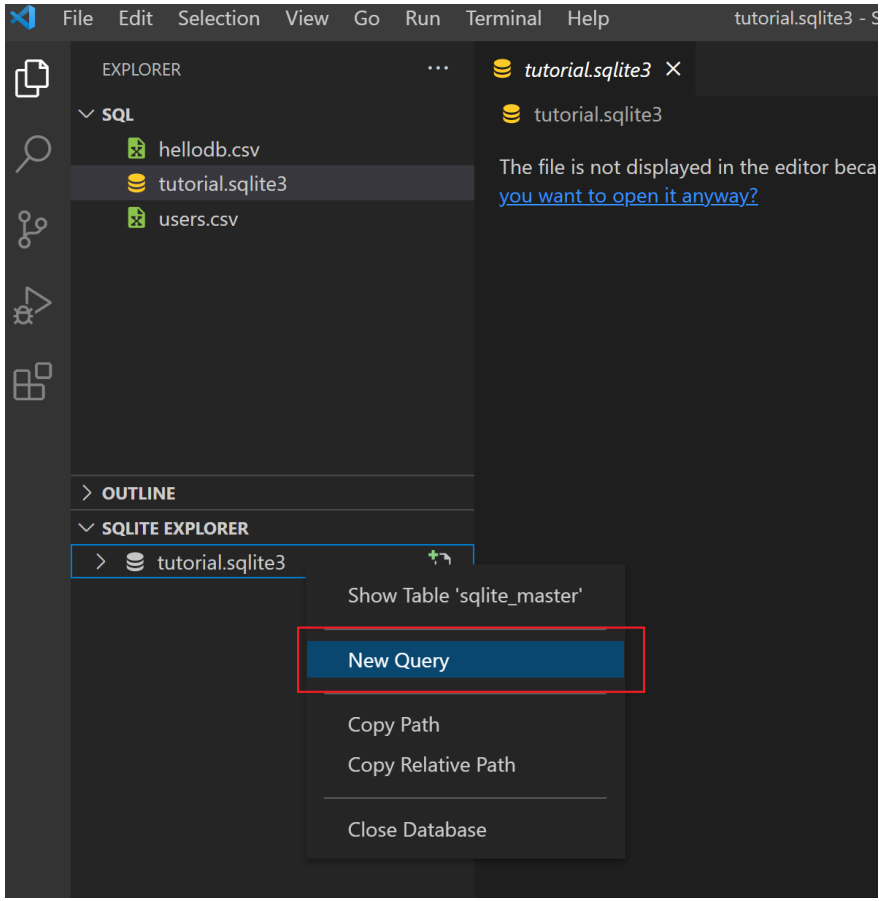
진행 TIP – sqlite 확장프로그램 사용하기 (1/5)



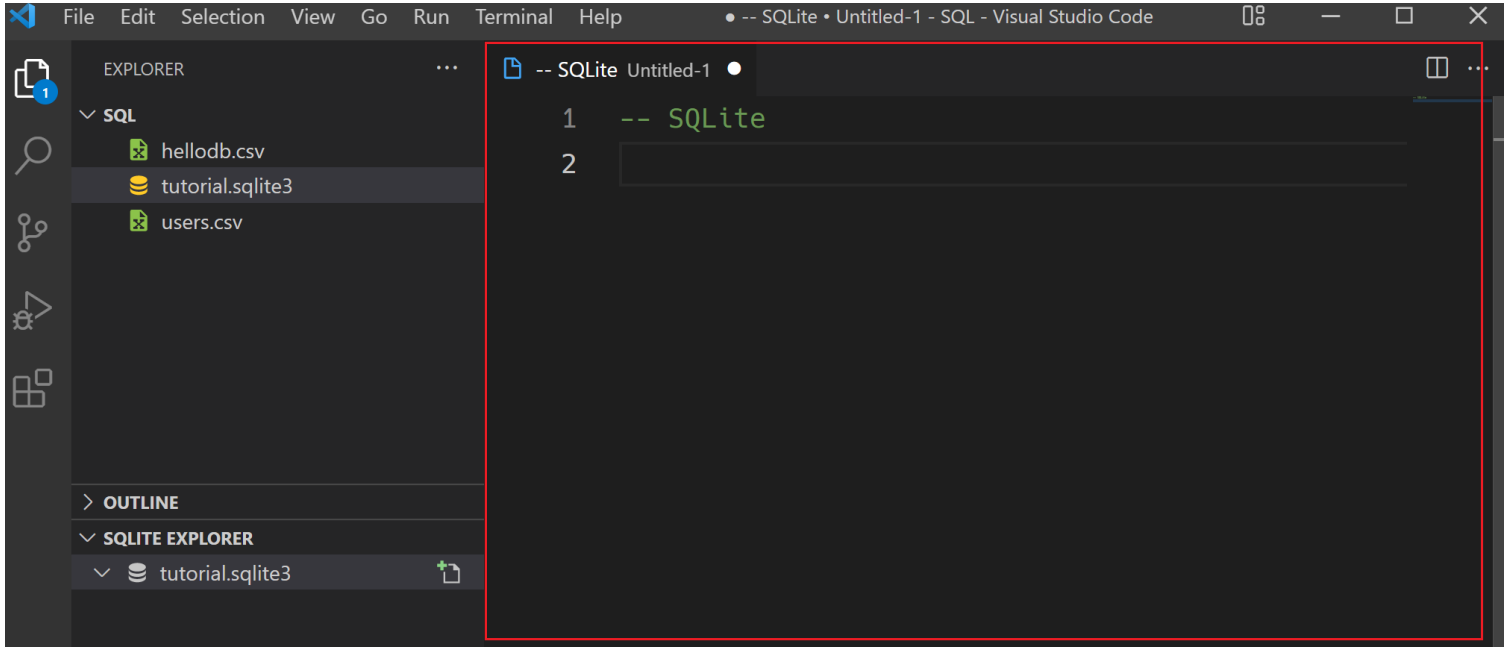
sqlite 파일 우측 클릭 – Open Database

HAPPY HACKING

진행 TIP – sqlite 확장프로그램 사용하기 (2/5)



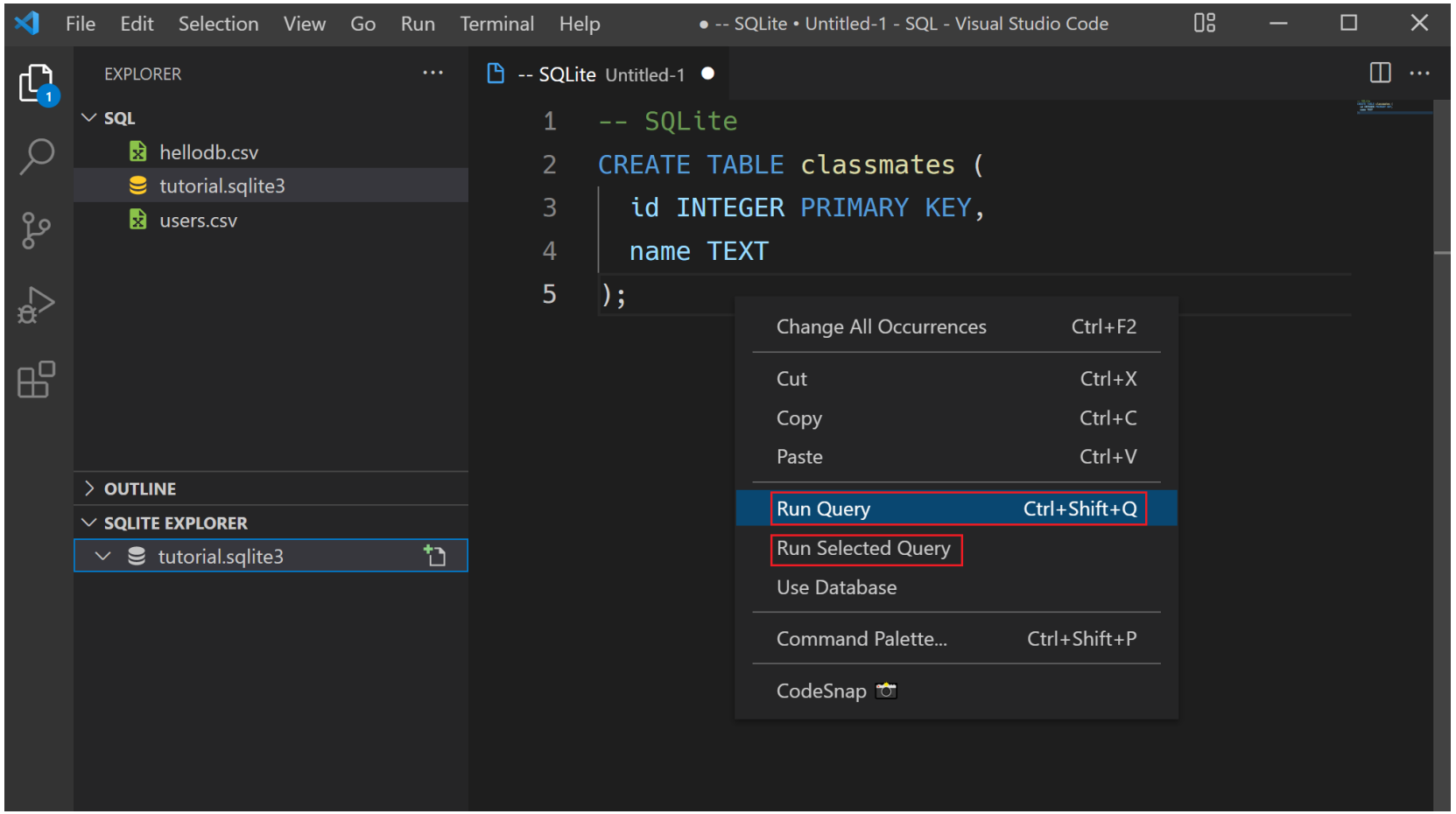
New Query 클릭



우측 화면에 sql 명령어를 작성하는 페이지가 출력 됨

HAPPY HACKING

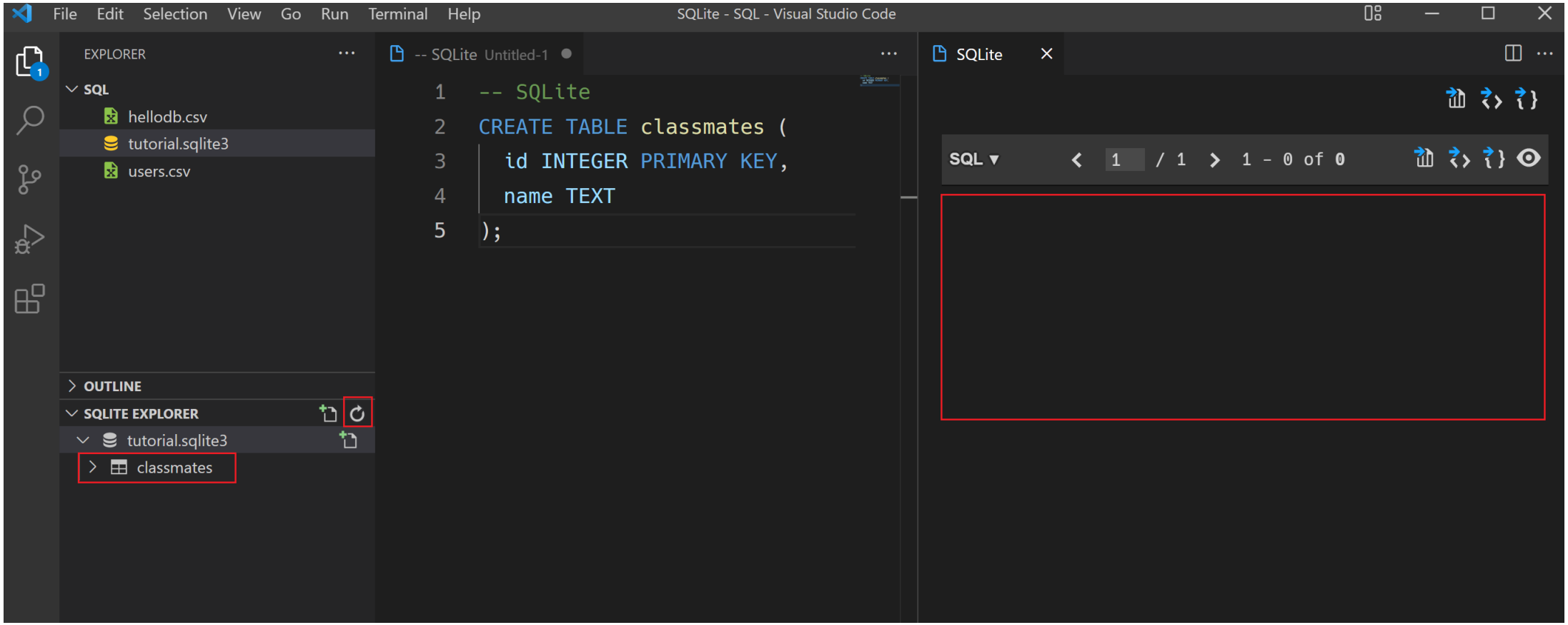
진행 TIP – sqlite 확장프로그램 사용하기 (3/5)



코드 작성 후 우측 클릭 – Run Query(전체 코드 실행) or Run Selected Query(선택 코드만 실행)

HAPPY HACKING

진행 TIP – sqlite 확장프로그램 사용하기 (4/5)

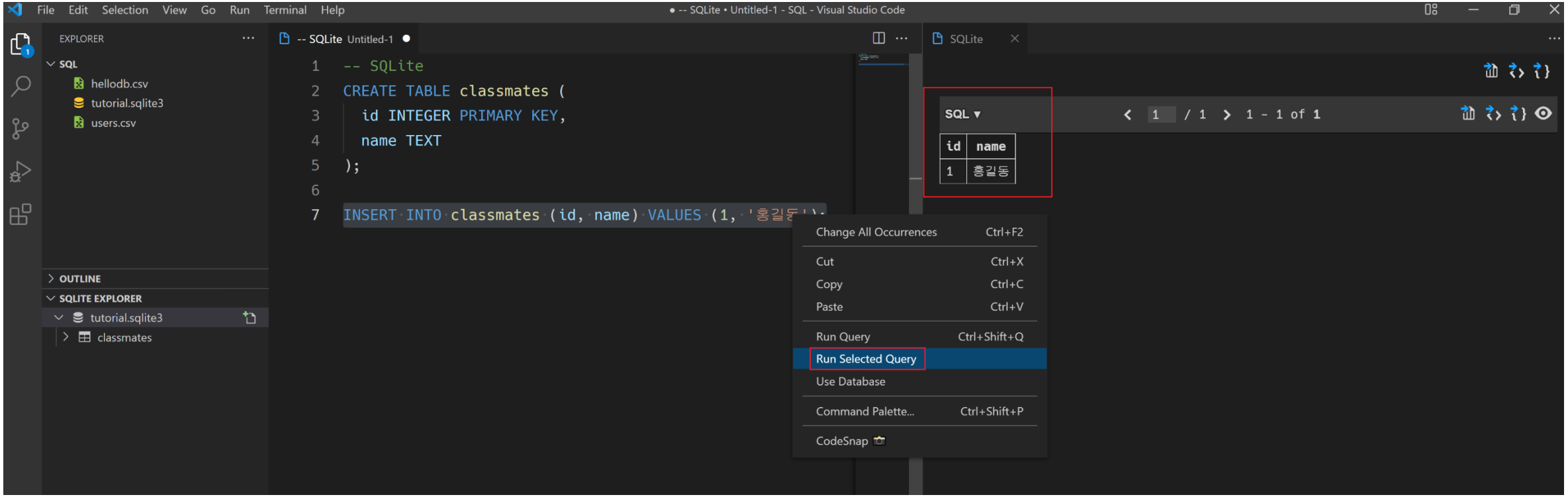


새로고침 클릭 후 데이터베이스 변화 확인

테이블 생성 및 삭제

HAPPY HACKING

진행 TIP – sqlite 확장프로그램 사용하기 (5/5)



특정 코드만 실행 후 가장 우측 화면에서 결과 확인해보기

테이블 생성 및 삭제

HAPPY HACKING

- 테이블 생성 및 삭제 statement
  - **CREATE TABLE**
    - 데이터베이스에서 테이블 생성
  - **DROP TABLE**
    - 데이터베이스에서 테이블 제거

https://www.sqlitetutorial.net/sqlite-create-table/
https://www.sqlitetutorial.net/sqlite-drop-table/

HAPPY HACKING

CREATE

```sql
CREATE TABLE classmates (
id INTEGER PRIMARY KEY,
name TEXT
);
```

HAPPY HACKING

CREATE – 테이블 생성 및 확인하기

```
sqlite> CREATE TABLE classmates (
   ...> id INTEGER PRIMARY KEY,
   ...> name TEXT
   ...> );
sqlite> .tables
classmates  examples
```

CREATE는 테이블을 생성!

HAPPY HACKING

특정 테이블의 schema 조회

```
sqlite> .schema classmates
CREATE TABLE classmates (
id INTEGER PRIMARY KEY,
name TEXT
);
```

HAPPY HACKING

DROP

```
DROP TABLE classmates;
```

HAPPY HACKING

DROP

```
sqlite> DROP TABLE classmates;
sqlite> .tables
examples
```

테이블 생성 및 삭제

HAPPY HACKING

- 다음과 같은 스키마(schema)를 가지고 있는 classmates 테이블을 만들고 스키마를 확인해보세요.

column	datatype
name	TEXT
age	INT
address	TEXT

HAPPY HACKING

SQL

- 테이블 생성 실습 (2/2)

```
CREATE TABLE classmates (
name TEXT,
age INT,
address TEXT
);
```

터미널 창 확인

```
sqlite> .schema classmates
CREATE TABLE classmates (
name TEXT,
age INT,
address TEXT
);
```

테이블 생성 및 삭제

HAPPY HACKING

- 필드 제약 조건
  - NOT NULL : NULL 값 입력 금지
  - UNIQUE : 중복 값 입력 금지 (NULL 값은 중복 입력 가능)
  - PRIMARY KEY : 테이블에서 반드시 하나. NOT NULL + UNIQUE
  - FOREIGN KEY : 외래키. 다른 테이블의 Key
  - CHECK : 조건으로 설정된 값만 입력 허용
  - DEFAULT : 기본 설정 값

https://www.sqlite.org/lang_createtable.html

테이블 생성 및 삭제

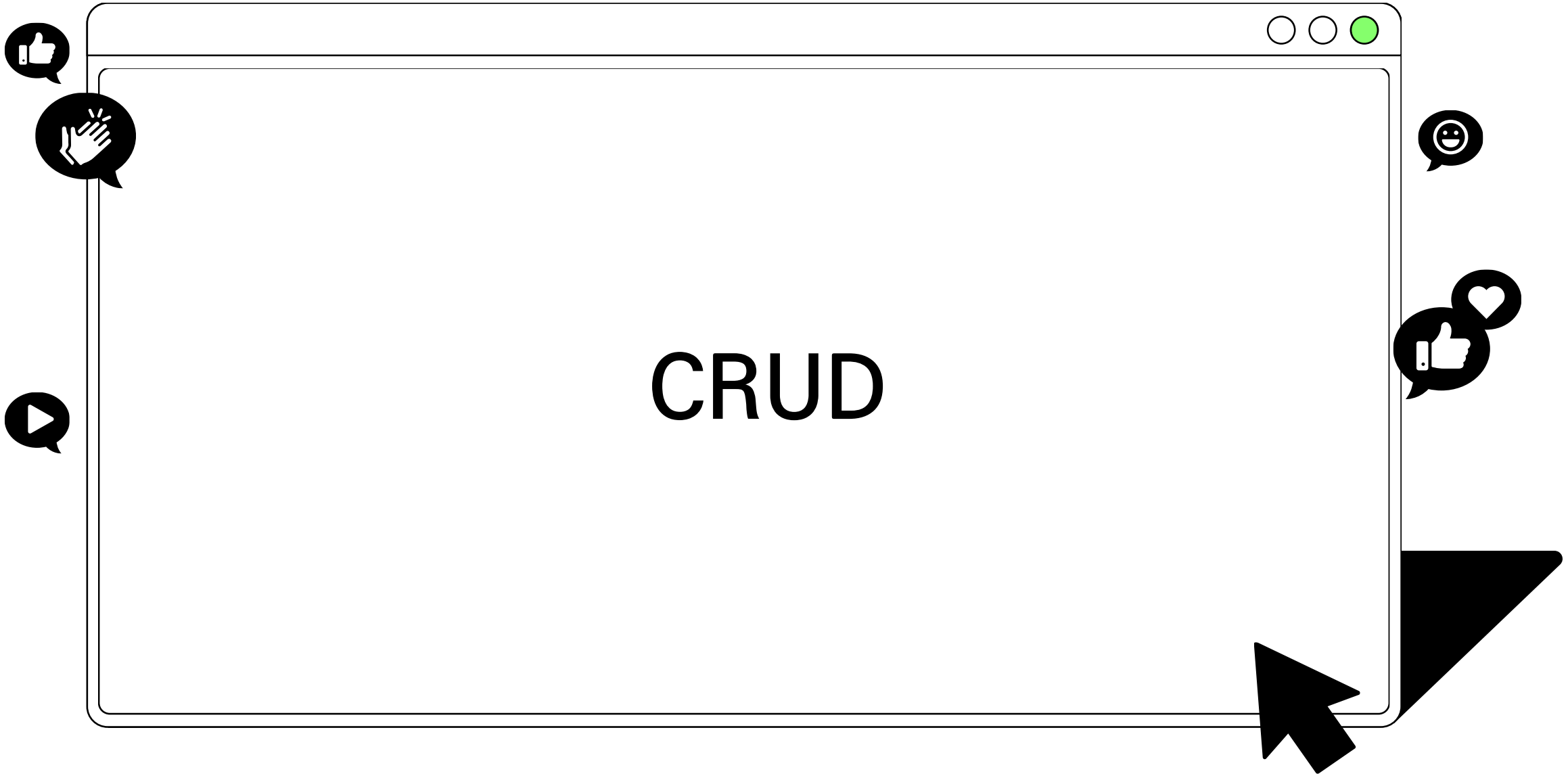
HAPPY HACKING

- 아래 테이블의 의미를 확인해보세요.

```sql
CREATE TABLE students(
    id INTEGER PRIMARY KEY,
    name TEXT NOT NULL,
    age INTEGER DEFAULT 1 CHECK (0 < age)
);
```

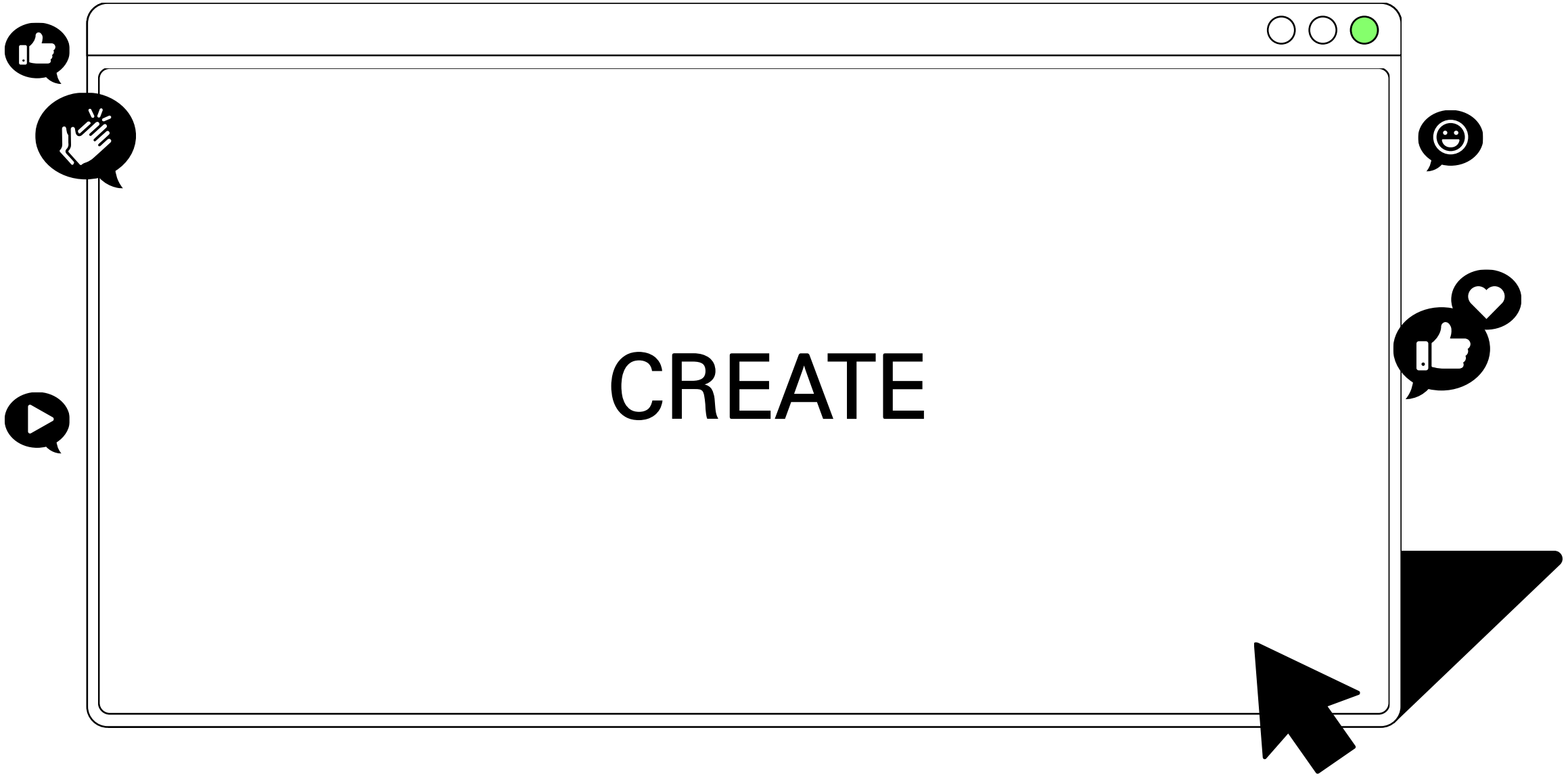
https://www.sqlite.org/lang_createtable.html

HAPPY HACKING



CRUD

HAPPY HACKING



HAPPY HACKING

- **INSERT**
  - “insert a single row into a table”
  - 테이블에 단일 행 삽입

```
INSERT INTO 테이블_이름 (컬럼1, 컬럼2) VALUES (값1, 값2);
```

  - 테이블에 정의된 모든 컬럼에 맞춰 순서대로 입력

```
INSERT INTO 테이블_이름 VALUES (값1, 값2, 값3);
```

CREATE

HAPPY HACKING

- classmates 테이블에 이름이 홍길동이고 나이가 23인 데이터를 넣어봅시다.
  SELECT문을 통해 확인해보세요.

CREATE

HAPPY HACKING

- classmates 테이블에 이름이 홍길동이고, 나이가 23인 데이터를 넣어봅시다.
  SELECT문을 통해 확인해보세요.

```sql
INSERT INTO classmates (name, age) VALUES ('홍길동', 23);
```

```
sqlite> SELECT * FROM classmates;
name        age         address
----------  ----------  ----------
홍길동         23
```

CREATE

HAPPY HACKING

- classmates 테이블에 이름이 홍길동이고, 나이가 30이고, 주소가 서울인 데이터를 넣어봅시다. SELECT문을 통해 확인해보세요.

CREATE

HAPPY HACKING

- classmates 테이블에 이름이 홍길동이고, 나이가 30이고, 주소가 서울인 데이터를 넣어봅시다. SELECT문을 통해 확인해보세요.

```
INSERT INTO classmates VALUES ('홍길동', 30, '서울');
```

CREATE

HAPPY HACKING

- classmates 테이블에 이름이 홍길동이고, 나이가 30이고, 주소가 서울인 데이터를 넣어봅시다. SELECT문을 통해 확인해보세요.

```
INSERT INTO classmates VALUES ('홍길동', 30, '서울');
```

```
sqlite> SELECT * FROM classmates;
name        age         address
----------  ----------  ----------
홍길동         23
홍길동         30          서울
```

CREATE

HAPPY HACKING

- 아래의 형태로 데이터가 기록되도 괜찮을까?

```
sqlite> SELECT * FROM classmates;
name         age          address
-----------  -----------  -----------
홍길동          23
홍길동          30           서울
```

HAPPY HACKING

- rowid : SQLite에서 PRIMARY KEY가 없는 경우 자동으로 증가하는 PK 컬럼

```
sqlite> SELECT rowid, * FROM classmates;
rowid       name        age         address
----------  ----------  ----------  ----------
1           홍길동          23
2           홍길동          30          서울
```

- 비어 있는 주소

```
sqlite> SELECT * FROM classmates;
name         age          address
----------   ----------   ----------
홍길동          23
홍길동          30           서울
```

CREATE

HAPPY HACKING

- 지우고 새로 만들기

```
sqlite> DROP TABLE classmates;
sqlite> CREATE TABLE classmates (
   ...> id INTEGER PRIMARY KEY,
   ...> name TEXT NOT NULL,
   ...> age INT NOT NULL,
   ...> address TEXT NOT NULL
   ...> );
```

- Q. classmates 테이블에 이름이 홍길동이고, 나이가 30이고, 주소가 서울인 데이터를 넣어봅시다. 그리고 SELECT문을 통해 확인해보세요.

```
INSERT INTO classmates VALUES ('홍길동', 30, '서울');
```

- 실패

```
sqlite> INSERT INTO classmates VALUES ('홍길동', 30, '서울');
Error: table classmates has 4 columns but 3 values were supplied
```

스키마에 id를 직접 작성했기 때문에

입력할 column들을 명시하지 않으면 자동으로 입력되지 않음

CREATE

HAPPY HACKING

- 첫번째 방법

```
INSERT INTO classmates VALUES (1, '홍길동', 30, '서울');
```

1. id를 포함한 모든 value를 작성

HAPPY HACKING

- 두번째 방법

```
INSERT INTO classmates (name, age, address) VALUES ('홍길동', 30, '서울');
```

2. 각 value에 맞는 column들을 명시적으로 작성

HAPPY HACKING

- 성공

```
sqlite> INSERT INTO classmates VALUES (1, '홍길동', 30, '서울');
sqlite> INSERT INTO classmates (name, age, address) VALUES ('홍길동', 30, '서울');
sqlite> SELECT * FROM classmates;
id          name        age         address
----------  ----------  ----------  ----------
1           홍길동         30          서울
2           홍길동         30          서울
```

이번 실습에서는 rowid를 사용해서 편하게 공부하자

CREATE

HAPPY HACKING

```
sqlite> DROP TABLE classmates;
sqlite> CREATE TABLE classmates (
   ...> name TEXT NOT NULL,
   ...> age INT NOT NULL,
   ...> address TEXT NOT NULL
   ...> );
```

이번 실습에서는 rowid를 사용해서 편하게 진행하자

- INSERT 직접 해보기
  - 방금 새롭게 생성한 테이블에 다음과 같은 정보를 저장하고 확인해봅시다.
  - 각 정보는 이름 / 나이 / 주소 로 구분됩니다.
    - 홍길동 / 30 / 서울
    - 김철수 / 30 / 제주
    - 이호영 / 26 / 인천
    - 박민희 / 29 / 대구
    - 최혜영 / 28 / 전주

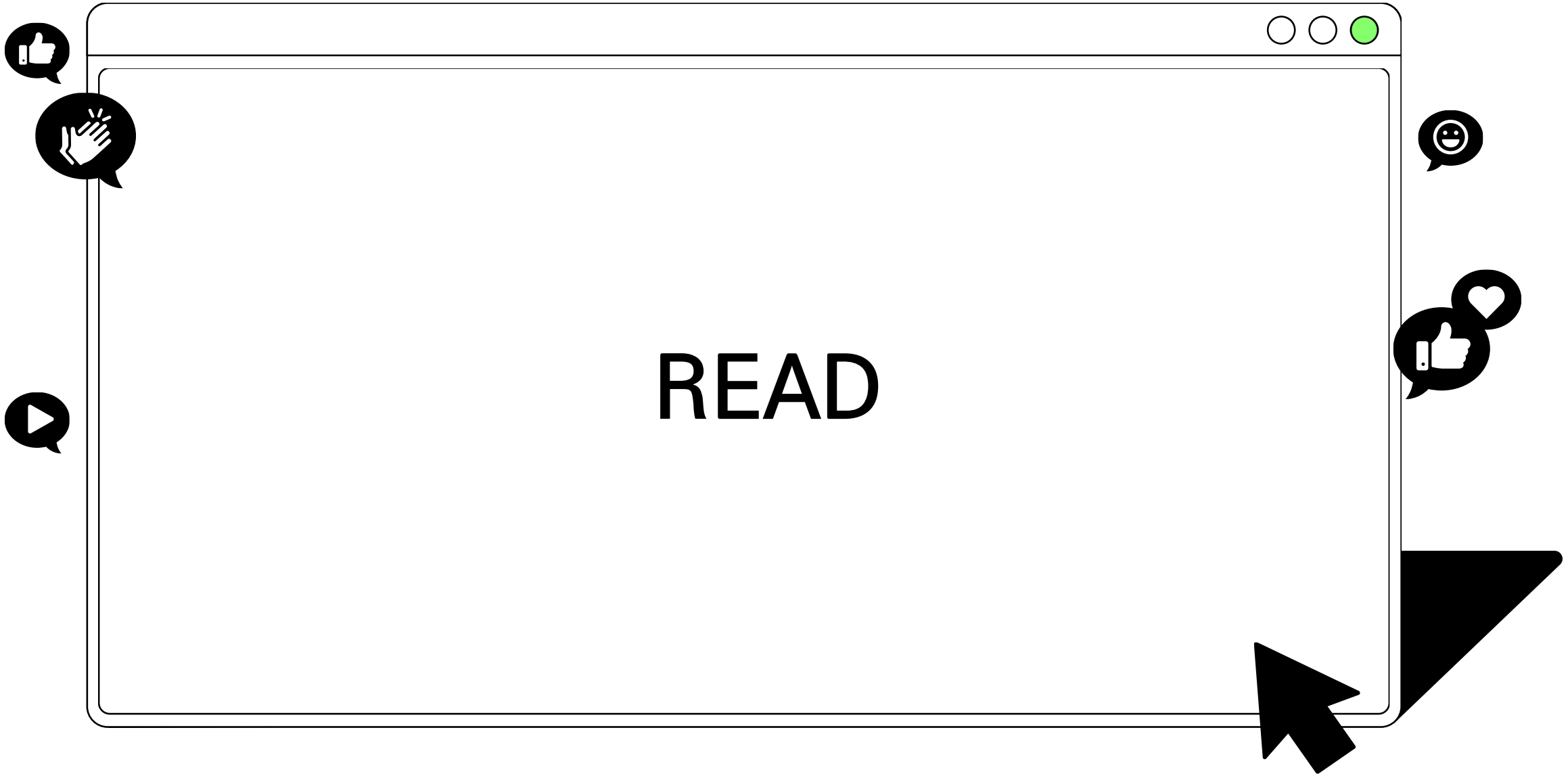
HAPPY HACKING

SQL

- INSERT 직접 해보기 – 해설

```
INSERT INTO classmates VALUES
('홍길동', 30, '서울'),
('김철수', 30, '제주'),
('이호영', 26, '인천'),
('박민희', 29, '대구'),
('최혜영', 28, '전주');
```

HAPPY HACKING



READ

HAPPY HACKING

- **SELECT**
  - “query data from a table”
  - 테이블에서 데이터를 조회
  - SELECT 문은 SQLite에서 가장 기본이 되는 문이며 다양한 절(clause)와 함께 사용
    - `ORDER BY, DISTINCT, WHERE, LIMIT, GROUP BY …`

https://www.sqlitetutorial.net/sqlite-select/

READ

HAPPY HACKING

- **`LIMIT`**
  - “constrain the number of rows returned by a query”
  - 쿼리에서 반환되는 행 수를 제한
  - 특정 행부터 시작해서 조회하기 위해 **`OFFSET`** 키워드와 함께 사용하기도 함
- **`WHERE`**
  - “specify the search condition for rows returned by the query”
  - 쿼리에서 반환된 행에 대한 특정 검색 조건을 지정

https://www.sqlitetutorial.net/sqlite-limit/
https://www.sqlitetutorial.net/sqlite-where/

HAPPY HACKING

- **`SELECT DISTINCT`**
  - “remove duplicate rows in the result set”
  - 조회 결과에서 중복 행을 제거
  - DISTINCT 절은 SELECT 키워드 바로 뒤에 작성해야 함

https://www.sqlitetutorial.net/sqlite-select-distinct

READ

HAPPY HACKING

- Q. classmates 테이블에서 id, name 컬럼 값만 조회하세요.

```
SELECT 컬럼1, 컬럼2, ... FROM 테이블이름;
```

READ

HAPPY HACKING

- Q. classmates 테이블에서 id, name 컬럼 값만 조회하세요.

```
SELECT 컬럼1, 컬럼2, ... FROM 테이블이름;
```

```
SELECT rowid, name FROM classmates;
rowid  name
-----  ----
1      홍길동
2      김철수
3      이호영
4      박민희
5      최혜영
```

READ

HAPPY HACKING

- Q. classmates 테이블에서 id, name 컬럼 값을 하나만 조회하세요.

```
SELECT 컬럼1, 컬럼2, ... FROM 테이블이름 LIMIT 숫자;
```

READ

HAPPY HACKING

- Q. classmates 테이블에서 id, name 컬럼 값을 하나만 조회하세요.

```
SELECT 컬럼1, 컬럼2, ... FROM 테이블이름 LIMIT 숫자;
```

```
SELECT rowid, name FROM classmates LIMIT 1;
rowid  name
-----  ----
1      홍길동
```

READ

HAPPY HACKING

- Q. classmates 테이블에서 id, name 컬럼 값을 세 번째에 있는 하나만 조회하세요.

```
SELECT 컬럼1, 컬럼2, ... FROM 테이블이름 LIMIT 숫자 OFFSET 숫자;
```

READ

HAPPY HACKING

- Q. classmates 테이블에서 id, name 컬럼 값을 세 번째에 있는 하나만 조회하세요.

```
SELECT 컬럼1, 컬럼2, ... FROM 테이블이름 LIMIT 숫자 OFFSET 숫자;
```

```
SELECT rowid, name FROM classmates LIMIT 1 OFFSET 2;
rowid  name
-----  ----
3      이호영
```

READ

HAPPY HACKING

- OFFSET : 처음부터 주어진 요소나 지점까지의 차이를 나타내는 정수형
- 예시
  1. 문자열 'abcedf' 에서 문자 `c`는 시작점 `a`에서 2의 OFFSET을 지님
  2. `SELECT * FROM MY_TABLE LIMIT 10 OFFSET 5`
     - "6번째 행 부터 10개 행을 조회 (6번째 행부터 10개를 출력)"
     - 0부터 시작함

READ

HAPPY HACKING

- Q. classmates 테이블에서 id, name 컬럼 값 중에 주소가 서울인 경우의 데이터를 조회하세요.

```
SELECT 컬럼1, 컬럼2, ... FROM 테이블이름 WHERE 조건;
```

READ

HAPPY HACKING

- Q. classmates 테이블에서 id, name 컬럼 값 중에 주소가 서울인 경우의 데이터를 조회하세요.

```
SELECT 컬럼1, 컬럼2, ... FROM 테이블이름 WHERE 조건;
```

```
SELECT * FROM classmates WHERE address='서울';
name   age   address
----   ---   -------
홍길동    30    서울
```

READ

HAPPY HACKING

- Q. classmates 테이블에서 age값 전체를 중복없이 조회하세요.

```
SELECT DISTINCT 컬럼 FROM 테이블이름;
```

READ

HAPPY HACKING

- Q. classmates 테이블에서 age값 전체를 중복없이 조회하세요.

```
SELECT DISTINCT 컬럼 FROM 테이블이름;
```

```
SELECT DISTINCT age FROM classmates;
age
---
30
26
29
28
```

HAPPY HACKING

DELETE

HAPPY HACKING

- **DELETE**
  - “remove rows from a table”
  - 테이블에서 행을 제거

```
DELETE FROM 테이블이름 WHERE 조건;
```

DELETE

HAPPY HACKING

- 데이터 삭제

```
DELETE FROM 테이블이름 WHERE 조건;
```

중복 불가능한(UNIQUE) 값인 rowid를 기준으로 삭제하자!

DELETE

HAPPY HACKING

- Q. classmates 테이블에 id가 5인 레코드를 삭제하세요.

```
DELETE FROM 테이블이름 WHERE 조건;
```

DELETE

HAPPY HACKING

- Q. classmates 테이블에 id가 5인 레코드를 삭제하세요.

```
DELETE FROM 테이블이름 WHERE 조건;
```

```
DELETE FROM classmates WHERE rowid=5;
rowid  name  age  address
-----  ----  ---  -------
1      홍길동   30   서울
2      김철수   30   제주
3      이호영   26   인천
4      박민희   29   대구
```

DELETE

HAPPY HACKING

- AUTOINCREMENT
  - SQLite가 사용되지 않은 값이나 이전에 삭제된 행의 값을 재사용하는 것을 방지

```
CREATE TABLE students(
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL
);

INSERT INTO students VALUES
(1, '홍길동'),
(2, '김철수'),
(3, '이호영'),
(4, '박민희'),
(5, '최혜영');
```

```
DELETE FROM members WHERE rowid=5;
INSERT INTO members (name) VALUES ('이민호');
SELECT * FROM members;
id  name
--  ----
1   홍길동
2   김철수
3   이호영
4   박민희
6   이민호
```

https://www.sqlitetutorial.net/sqlite-autoincrement/

HAPPY HACKING

UPDATE

HAPPY HACKING

- **UPDATE**
  - “update data of existing rows in the table”
  - 기존 행의 데이터를 수정
  - **SET** clause에서 테이블의 각 열에 대해 새로운 값을 설정

```
UPDATE 테이블이름 SET 컬럼1=값1, 컬럼2=값2, ... WHERE 조건;
```

UPDATE

HAPPY HACKING

- Q. classmates 테이블에 id가 5인 레코드를 수정하세요.
  - 이름을 홍길동으로, 주소를 제주도로 수정합니다.

```
UPDATE 테이블이름 SET 컬럼1=값1, 컬럼2=값2, ... WHERE 조건;
```

UPDATE

HAPPY HACKING

- Q. classmates 테이블에 id가 5인 레코드를 수정하세요.
  - 이름을 홍길동으로, 주소를 제주도로 수정합니다.

```
UPDATE 테이블이름 SET 컬럼1=값1, 컬럼2=값2, ... WHERE 조건;
```

```
UPDATE classmates SET name='홍길동',
address='제주도' WHERE rowid=5;
SELECT rowid, * FROM classmates;
rowid  name  age  address
-----  ----  ---  -------
1      홍길동   30   서울
2      김철수   30   제주
3      이호영   26   인천
4      박민희   29   대구
5      홍길동   40   제주도
```

UPDATE

HAPPY HACKING

	구문	예시
C	INSERT	**INSERT INTO** 테이블이름 (컬럼1, 컬럼2, ...) **VALUES** (값1, 값2);
R	SELECT	**SELECT** * **FROM** 테이블이름 **WHERE** 조건;
U	UPDATE	**UPDATE** 테이블이름 **SET** 컬럼1=값1, 컬럼2=값2 **WHERE** 조건;
D	DELETE	**DELETE FROM** 테이블이름 **WHERE** 조건;