1. The university database schema contains various integrity constraints.
   Execute some SQL statements violating them.

Instructor table을 대상으로 SQL statements를 실행했다. 우선 '\d instructor' 를 통해 살펴본 정보들은 아래와 같다.

```
practice2=# \d instructor
                    Table "public.instructor"
  Column   |         Type          | Collation | Nullable | Default
-----------+-----------------------+-----------+----------+---------
 id        | character varying(5)  |           | not null |
 name      | character varying(20) |           | not null |
 dept_name | character varying(20) |           |          |
 salary    | numeric(8,2)          |           |          |
Indexes:
    "instructor_pkey" PRIMARY KEY, btree (id)
Check constraints:
    "instructor_salary_check" CHECK (salary > 9000::numeric)
Foreign-key constraints:
    "instructor_dept_name_fkey" FOREIGN KEY (dept_name) REFERENCES department(dept_name) ON DELETE SET N
ULL
Referenced by:
    TABLE "advisor" CONSTRAINT "advisor_i_id_fkey" FOREIGN KEY (i_id) REFERENCES instructor(id) ON DELET
E SET NULL
    TABLE "teaches" CONSTRAINT "teaches_id_fkey" FOREIGN KEY (id) REFERENCES instructor(id) ON DELETE CA
SCADE
```

(1) Primary key constraints

```
practice2=# insert into instructor(name, dept_name, salary) values('Elon Musk',
'Physics', 10000);
ERROR:  null value in column "id" of relation "instructor" violates not-null constraint
DETAIL:  Failing row contains (null, Elon Musk, Physics, 10000.00).
practice2=# insert into instructor values('10101','Elon Musk', 'Physics', 10000);
ERROR:  duplicate key value violates unique constraint "instructor_pkey"
DETAIL:  Key (id)=(10101) already exists.
```

Instructor table의 primary key인 id는 not null 이면서 unique해야 한다는 primary key constraints를 지닌다.
이로 인해 id가 null 이거나 기존에 이미 존재하는 id와 중복인 경우 insert SQL statement에 에러가 발생해 실행되지 않는다.

(2) Foreign key constraints

```
practice2=# insert into instructor values('22510','Elon Musk','Rocket Science', 10000);
ERROR:  insert or update on table "instructor" violates foreign key constraint "instructor_dept_name_fke
y"
DETAIL:  Key (dept_name)=(Rocket Science) is not present in table "department".
```

instructor 테이블의 foreign key constraints는 FOREIGN KEY (dept_name) REFERENCES department(dept_name) 이므로
이 테이블의 dept_name은 department table에 존재해야 한다. 하지만 'Rocket Science'는 department에 존재하지 않는 dept_name이다.
따라서 이를 삽입하는 insert SQL statement는 foreign key constraints violation에 의해 실행될 수 없다.

(3) Not null constraints

```
practice2=# insert into instructor(id, dept_name, salary) values('22510', 'Physics', 10000);
ERROR:  null value in column "name" of relation "instructor" violates not-null constraint
DETAIL:  Failing row contains (22510, null, Physics, 10000.00).
```

instructor 테이블에서 name attribute는 not null로 지정되어있다. 이로 인해 name이 null인 data는 삽입될 수 없다.

(4) Check constraints

```
practice2=# insert into instructor values('22510','Elon Musk','Physics', 0);
ERROR:  new row for relation "instructor" violates check constraint "instructor_salary_check"
DETAIL:  Failing row contains (22510, Elon Musk, Physics, 0.00).
```

Instruct 테이블 내에 "instrucgtor_salary_check" CHECK (salary > 9000::numeric) 이라는 check constraint가 존재한다.
이로 인해 9000 미만의 value를 salary로 삽입하는 SQL statement는 check constraints violation에 의해 실행될 수 없다.

2. Make two or more concurrently executed transactions, and show they are executed in an isolated manner.

```
practice2=# begin;
BEGIN
practice2=*# select * from instructor;
  id   |    name    | dept_name  |  salary
-------+------------+------------+----------
 10101 | Srinivasan | Comp. Sci. | 65000.00
 12121 | Wu         | Finance    | 90000.00
 15151 | Mozart     | Music      | 40000.00
 22222 | Einstein   | Physics    | 95000.00
 32343 | El Said    | History    | 60000.00
 33456 | Gold       | Physics    | 87000.00
 45565 | Katz       | Comp. Sci. | 75000.00
 58583 | Califieri  | History    | 62000.00
 76543 | Singh      | Finance    | 80000.00
 76766 | Crick      | Biology    | 72000.00
 83821 | Brandt     | Comp. Sci. | 92000.00
 98345 | Kim        | Elec. Eng. | 80000.00
(12 rows)

practice2=*# insert into instructor values('22510','Elon Musk', 'Physics', 99000);
INSERT 0 1
practice2=*# select * from instructor;
  id   |    name    | dept_name  |  salary
-------+------------+------------+----------
 10101 | Srinivasan | Comp. Sci. | 65000.00
 12121 | Wu         | Finance    | 90000.00
 15151 | Mozart     | Music      | 40000.00
 22222 | Einstein   | Physics    | 95000.00
 32343 | El Said    | History    | 60000.00
 33456 | Gold       | Physics    | 87000.00
 45565 | Katz       | Comp. Sci. | 75000.00
 58583 | Califieri  | History    | 62000.00
 76543 | Singh      | Finance    | 80000.00
 76766 | Crick      | Biology    | 72000.00
 83821 | Brandt     | Comp. Sci. | 92000.00
 98345 | Kim        | Elec. Eng. | 80000.00
 22510 | Elon Musk  | Physics    | 99000.00
(13 rows)

practice2=*# delete from instructor;
DELETE 13
practice2=*# select * from instructor;
 id | name | dept_name | salary
----+------+-----------+--------
(0 rows)

practice2=*# rollback;
ROLLBACK
```

```
practice2=# select * from instructor;
  id   |    name    | dept_name  |  salary
-------+------------+------------+----------
 10101 | Srinivasan | Comp. Sci. | 65000.00
 12121 | Wu         | Finance    | 90000.00
 15151 | Mozart     | Music      | 40000.00
 22222 | Einstein   | Physics    | 95000.00
 32343 | El Said    | History    | 60000.00
 33456 | Gold       | Physics    | 87000.00
 45565 | Katz       | Comp. Sci. | 75000.00
 58583 | Califieri  | History    | 62000.00
 76543 | Singh      | Finance    | 80000.00
 76766 | Crick      | Biology    | 72000.00
 83821 | Brandt     | Comp. Sci. | 92000.00
 98345 | Kim        | Elec. Eng. | 80000.00
(12 rows)

practice2=# select * from instructor;
  id   |    name    | dept_name  |  salary
-------+------------+------------+----------
 10101 | Srinivasan | Comp. Sci. | 65000.00
 12121 | Wu         | Finance    | 90000.00
 15151 | Mozart     | Music      | 40000.00
 22222 | Einstein   | Physics    | 95000.00
 32343 | El Said    | History    | 60000.00
 33456 | Gold       | Physics    | 87000.00
 45565 | Katz       | Comp. Sci. | 75000.00
 58583 | Califieri  | History    | 62000.00
 76543 | Singh      | Finance    | 80000.00
 76766 | Crick      | Biology    | 72000.00
 83821 | Brandt     | Comp. Sci. | 92000.00
 98345 | Kim        | Elec. Eng. | 80000.00
(12 rows)

practice2=#
```

왼쪽 terminal에서 begin statement로 transaction을 시작하고,이후 새로운 tuple('22510', 'Elon', 'Physics', 9900)을 instructor table에 삽입했다. 이후 'select * form instructor'를 입력하면 transaction이 실행중인 왼쪽 terminal에서는 삽입한 튜플을 확인할 수 있지만, transaction이 실행되고 있지 않은 오른쪽 터미널에서는 왼쪽 터미널과 같은 select문을 입력해도 삽입된 튜플이 나타나지 않는다.

이후 'delete from instructor' statement로 table의 모든 튜플을 삭제했다. 이 역시 select문을 사용하면 왼쪽 터미널에서는 모든 튜플이 삭제된 것을 확인할 수 있지만, 오른쪽 터미널에서는 기존의 테이블이 삭제되지 않은 채 그대로 나타났다.

```
practice2=*# rollback;
ROLLBACK
practice2=# select * from instructor;
  id   |    name    | dept_name  |  salary
-------+------------+------------+----------
 10101 | Srinivasan | Comp. Sci. | 65000.00
 12121 | Wu         | Finance    | 90000.00
 15151 | Mozart     | Music      | 40000.00
 22222 | Einstein   | Physics    | 95000.00
 32343 | El Said    | History    | 60000.00
 33456 | Gold       | Physics    | 87000.00
 45565 | Katz       | Comp. Sci. | 75000.00
 58583 | Califieri  | History    | 62000.00
 76543 | Singh      | Finance    | 80000.00
 76766 | Crick      | Biology    | 72000.00
 83821 | Brandt     | Comp. Sci. | 92000.00
 98345 | Kim        | Elec. Eng. | 80000.00
(12 rows)
```

한편, 왼쪽 터미널에서 rollback statement를 입력하면, 앞서 transaction을 시작한 이후 concurrent하게 실행되었던 insert, delete 명령이 모두 취소된다. 이에 따라 select 문을 입력했을 때 삭제했던 모든 튜플들이 다시 원상복구됨과 동시에, 앞서 추가했던 ('22510', 'Elon', 'Physics', 9900) 튜플 역시 사라진 원래의 상태로 되돌아온다.

3. Make users and set up a few authorization rules; Show some non-authorized accesses.

```
practice2=# create user levy;
CREATE ROLE
practice2=# create view levy_takes as
practice2-# (select course_id, sec_id, semester, year, grade
practice2(# from student natural join takes
practice2(# where student.name = 'Levy');
CREATE VIEW
practice2=# grant select on levy_takes to levy;
GRANT
practice2=#
```

학생 중 하나인 levy를 유저로 생성한다. 그리고 levy에게 본인이 수강한 성적을 확인할 수 있는 'levy_takes' 라는 view에 대해서 select 권한을 부여한다.

```
postgres=> \c practice2
You are now connected to database "practice2" as user "levy".
practice2=> select * from takes;
ERROR:  permission denied for table takes
practice2=> select * from levy_takes;
 course_id | sec_id | semester | year | grade
-----------+--------+----------+------+-------
 CS-101    | 1      | Fall     | 2017 | F
 CS-101    | 1      | Spring   | 2018 | B+
 CS-319    | 1      | Spring   | 2018 | B
(3 rows)
```

이후 levy로 user를 전환해 'select * from takes' 명령문을 실행하면, takes 테이블에 대한 접근이 불가능하다.
한편 앞서 권한을 부여했던 'levy_takes'라는 view에 대해서는 select 문이 실행 가능하다.

```
practice2=# create user professor;
CREATE ROLE
practice2=# grant select, insert, update on takes to professor;
GRANT
```

다음으로는 professor라는 유저를 생성한다.
professor에 대해서는 takes라는 테이블에 대해 select, insert, update 권한을 부여한다.

```
practice1=> \c practice2;
You are now connected to database "practice2" as user "professor".
practice2=> select * from takes where id = '45678';
  id   | course_id | sec_id | semester | year | grade
-------+-----------+--------+----------+------+-------
 45678 | CS-101    | 1      | Fall     | 2017 | F
 45678 | CS-101    | 1      | Spring   | 2018 | B+
 45678 | CS-319    | 1      | Spring   | 2018 | B
(3 rows)

practice2=> update takes
practice2-> set grade = 'A'
practice2-> where id = '45678' and course_id = 'CS-101' and year = 2018;
UPDATE 1
practice2=> select * from takes where id = '45678';
  id   | course_id | sec_id | semester | year | grade
-------+-----------+--------+----------+------+-------
 45678 | CS-101    | 1      | Fall     | 2017 | F
 45678 | CS-101    | 1      | Spring   | 2018 | A
 45678 | CS-319    | 1      | Spring   | 2018 | B
(3 rows)

practice2=> delete from takes
practice2-> where grade = 'F';
ERROR:  permission denied for table takes
practice2=>
```

이후 professor로 유저를 전환하면 takes 테이블에 대해 select와 update가 가능하다.
그런데 앞서 delete 권한은 부여하지 않았기 때문에 마지막 부분의 delete from taeks where grade = 'F' 라는 명령문은 실행되지 않는다.

4. Create some views and show how view maintenance works and how view update is processed.

(1) view maintenance (= update of source relation)

view의 source relation에 변화가 생겼을 때 그것이 view에도 반영되는 것을 의미한다. 이를 아래와 같이 확인해보았다.

```
practice2=# create materialized view levy_takes as
(select course_id, sec_id, semester, year, grade
from student natural join takes
where student.name = 'Levy');
SELECT 3
practice2=# select * from levy_takes;
 course_id | sec_id | semester | year | grade
-----------+--------+----------+------+-------
 CS-101    | 1      | Fall     | 2017 | F
 CS-319    | 1      | Spring   | 2018 | B
 CS-101    | 1      | Spring   | 2018 | A
(3 rows)
```

먼저 levy 라는 이름의 학생이 수강한 section의 course_id, sec_id, year, grade를 표시하는 levy_takes 를 materialized view로 생성한다.

```
practice2=# insert into takes values('45678', 'CS-347', 1, 'Fall', 2017, 'A');
INSERT 0 1
```

다음으로 takes 테이블에 levy의 수강 정보를 추가로 삽입한다.

```
practice2=# refresh materialized view levy_takes;
REFRESH MATERIALIZED VIEW
practice2=# select * from levy_takes;
 course_id | sec_id | semester | year | grade
-----------+--------+----------+------+-------
 CS-101    | 1      | Fall     | 2017 | F
 CS-319    | 1      | Spring   | 2018 | B
 CS-101    | 1      | Spring   | 2018 | A
 CS-347    | 1      | Fall     | 2017 | A
(4 rows)
```

이후 levy_takes를 갱신(refresh) 한다. 이렇게 하면 DB가 자동으로 추가된 정보를 반영해 levy_takes가 update된다.

(2) view update

반대로 view에서 발생한 update가 소스에 반영되는 것을 의미한다. 이것은 해당 뷰가 simple view일 때만 가능하다.

```
practice2=# create view faculty as
practice2-# select id, name, dept_name from instructor;
CREATE VIEW
practice2=# select * from faculty;
  id   |    name    | dept_name
-------+------------+------------
 10101 | Srinivasan | Comp. Sci.
 12121 | Wu         | Finance
 15151 | Mozart     | Music
 22222 | Einstein   | Physics
 32343 | El Said    | History
 33456 | Gold       | Physics
 45565 | Katz       | Comp. Sci.
 58583 | Califieri  | History
 76543 | Singh      | Finance
 76766 | Crick      | Biology
 83821 | Brandt     | Comp. Sci.
 98345 | Kim        | Elec. Eng.
(12 rows)
```

예를 들어 위와 같은 ‘faculty’라는 view는 하나의 source relation에서만 attribute(id, name, dept_name)를 (distinct 같은 별도의 조건 없이) 순수하게 select한다. 이때 select 되지 않은 attribute인 salary는 null로 설정 가능한 attribute다.

```
practice2=# insert into faculty values('22510', 'Elon', 'Physics');
INSERT 0 1
practice2=# select * from faculty;
  id   |    name    | dept_name
-------+------------+------------
 10101 | Srinivasan | Comp. Sci.
 12121 | Wu         | Finance
 15151 | Mozart     | Music
 22222 | Einstein   | Physics
 32343 | El Said    | History
 33456 | Gold       | Physics
 45565 | Katz       | Comp. Sci.
 58583 | Califieri  | History
 76543 | Singh      | Finance
 76766 | Crick      | Biology
 83821 | Brandt     | Comp. Sci.
 98345 | Kim        | Elec. Eng.
 22510 | Elon       | Physics
(13 rows)
```

```
practice2=# select * from instructor;
  id   |    name    | dept_name  |  salary
-------+------------+------------+----------
 10101 | Srinivasan | Comp. Sci. | 65000.00
 12121 | Wu         | Finance    | 90000.00
 15151 | Mozart     | Music      | 40000.00
 22222 | Einstein   | Physics    | 95000.00
 32343 | El Said    | History    | 60000.00
 33456 | Gold       | Physics    | 87000.00
 45565 | Katz       | Comp. Sci. | 75000.00
 58583 | Califieri  | History    | 62000.00
 76543 | Singh      | Finance    | 80000.00
 76766 | Crick      | Biology    | 72000.00
 83821 | Brandt     | Comp. Sci. | 92000.00
 98345 | Kim        | Elec. Eng. | 80000.00
 22510 | Elon       | Physics    |
(13 rows)
```

이때 이 faculty 라는 view에 새로운 튜플을 추가하면, 이같으 update가 view에 표시되는 것은 물론, 해당 view의 source relation인 instructor에도 자동으로 반영된다. 이때 view에는 존재하지 않지만, source realtion에 존재하는 attribute는 null로 표시된다.

한편 하나가 아닌 둘 이상의 relation을 source로 지니는 즉 simple 하지 않은 view의 경우, view update가 자동으로 이루어지지 않는다.
예를 들어 앞서 살펴본 levy_take (여기서는 materialized view가 아닌 단순 view로 가정한다)는 student와 takes를 source realtion으로 하기 때문에 simple view가 아니다. 따라서 아래와 같이 해당 view에서 특정 튜플을 제거하는 delete 명령을 입력해도 그것이 실행되지 않는다.

```
practice2=# select * from levy_takes;
 course_id | sec_id | semester | year | grade
-----------+--------+----------+------+-------
 CS-101    | 1      | Fall     | 2017 | F
 CS-319    | 1      | Spring   | 2018 | B
 CS-101    | 1      | Spring   | 2018 | A
 CS-347    | 1      | Fall     | 2017 | A
(4 rows)

practice2=# delete from levy_takes where course_id = 'CS-347';
ERROR:  cannot delete from view "levy_takes"
DETAIL:  Views that do not select from a single table or view are not automatically updatable.
```