Name: Vanness M. Lao
Course, Year, and Section: BSIT 3-1N

Activity 1: Writing SQL Syntax

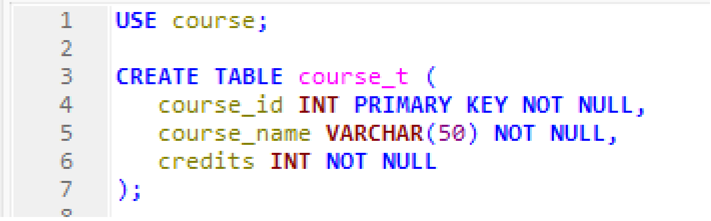
#1
Instructions:

Write the appropriate SQL syntax for the following tasks. Use the examples provided in the lesson as a guide.

1. Create a table named course_t with the following columns:

    - course_id (INTEGER, primary key, not null)
    - course_name (VARCHAR(50), not null)
    - credits (INTEGER, not null

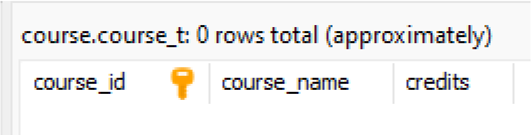
    Query:



```sql
USE course;

CREATE TABLE course_t (
   course_id INT PRIMARY KEY NOT NULL,
   course_name VARCHAR(50) NOT NULL,
   credits INT NOT NULL
);
```

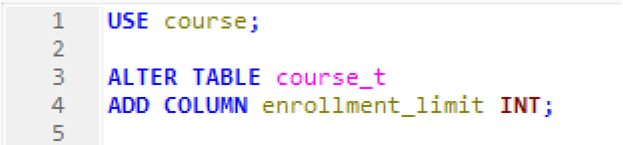
    Output:



```
course.course_t: 0 rows total (approximately)
course_id 🔑 | course_name | credits
```

2. Add a column enrollment_limit (INTEGER) to the course_t table.

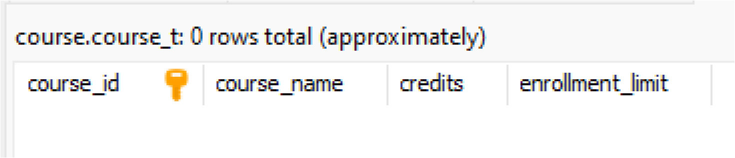
    Query:



```sql
USE course;

ALTER TABLE course_t
ADD COLUMN enrollment_limit INT;
```

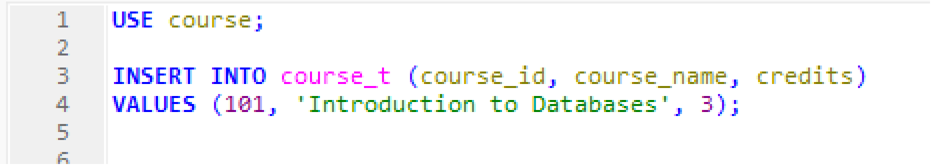
    Output:



```
course.course_t: 0 rows total (approximately)
course_id 🔑 | course_name | credits | enrollment_limit
```

3. Insert the following data into the course_t table:
    - course_id: 101
    - course_name: "Introduction to Databases"
    - credits: 3

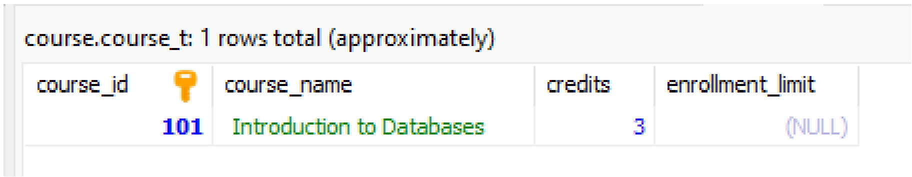
Query:



```
USE course;

INSERT INTO course_t (course_id, course_name, credits)
VALUES (101, 'Introduction to Databases', 3);
```

Output:

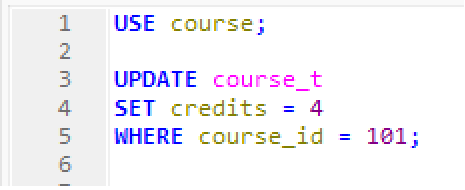


course.course_t: 1 rows total (approximately)

course_id	course_name	credits	enrollment_limit
101	Introduction to Databases	3	(NULL)

4. Update the credits column for the course with course_id 101 to 4.

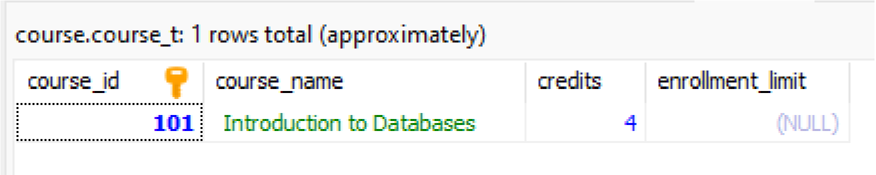
Query:



```
USE course;

UPDATE course_t
SET credits = 4
WHERE course_id = 101;
```

Output:

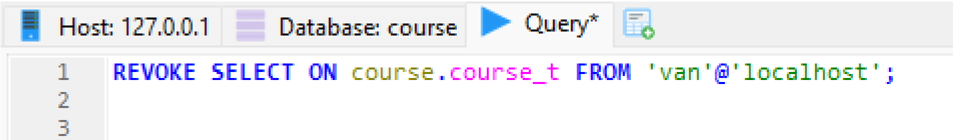


course.course_t: 1 rows total (approximately)

course_id	course_name	credits	enrollment_limit
101	Introduction to Databases	4	(NULL)

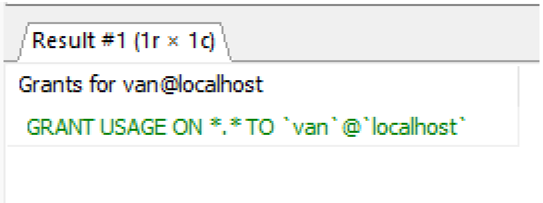
5. Revoke permission for a user to query the course_t table.

Query:



```
REVOKE SELECT ON course.course_t FROM 'van'@'localhost';
```

Output:



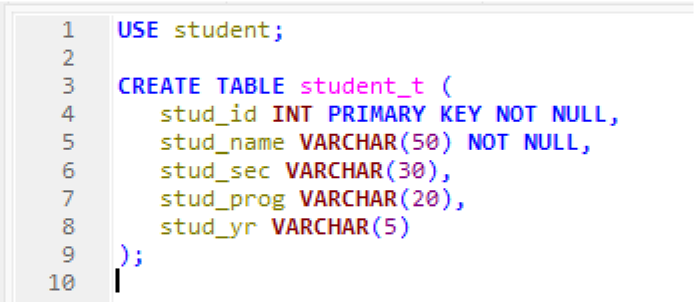
```
Grants for van@localhost
GRANT USAGE ON *.* TO `van`@`localhost`
```

Instructions:

Write the appropriate SQL syntax for the following tasks.

1. Create a table student_t with the following columns:
    - stud_id (INTEGER, Primary Key, NOT NULL)
    - stud_name (VARCHAR(50), NOT NULL)
    - stud_sec (VARCHAR(30))
    - stud_prog (VARCHAR(20))
    - stud_yr (VARCHAR(5))

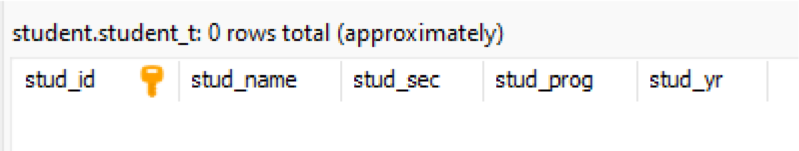
Query:



```sql
USE student;

CREATE TABLE student_t (
    stud_id INT PRIMARY KEY NOT NULL,
    stud_name VARCHAR(50) NOT NULL,
    stud_sec VARCHAR(30),
    stud_prog VARCHAR(20),
    stud_yr VARCHAR(5)
);
```

Output:

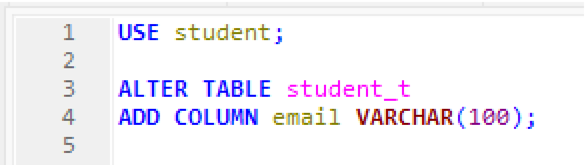


student.student_t: 0 rows total (approximately)

stud_id	stud_name	stud_sec	stud_prog	stud_yr

2. Add a column email (VARCHAR(100)) to the student_t table.

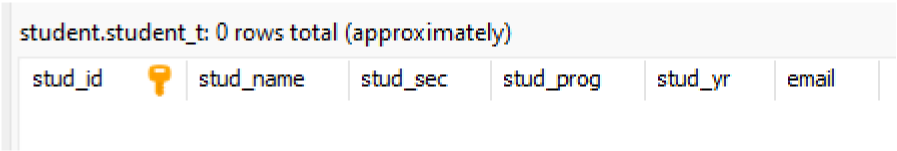
Query:



```sql
USE student;

ALTER TABLE student_t
ADD COLUMN email VARCHAR(100);
```

Output:

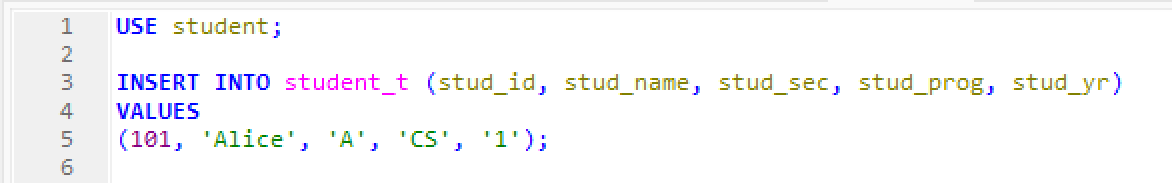


student.student_t: 0 rows total (approximately)

stud_id	stud_name	stud_sec	stud_prog	stud_yr	email

3. Insert the following record into the student_t table:
    - stud_id: 101, stud_name: "Alice", stud_sec: "A", stud_prog: "CS", stud_yr: "1"

Query:

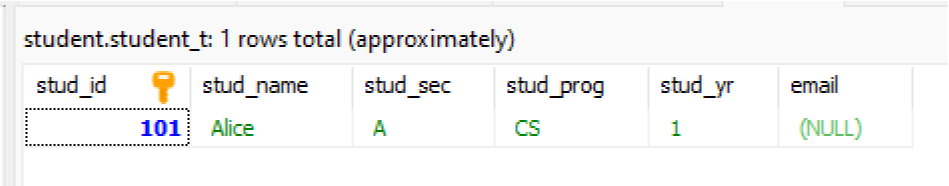


```sql
USE student;

INSERT INTO student_t (stud_id, stud_name, stud_sec, stud_prog, stud_yr)
VALUES
(101, 'Alice', 'A', 'CS', '1');

```

Output:

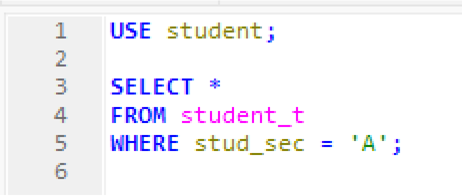


student.student_t: 1 rows total (approximately)

stud_id	stud_name	stud_sec	stud_prog	stud_yr	email
101	Alice	A	CS	1	(NULL)

4. Write a query to retrieve all records from the student_t table where stud_sec is "A".

Query:

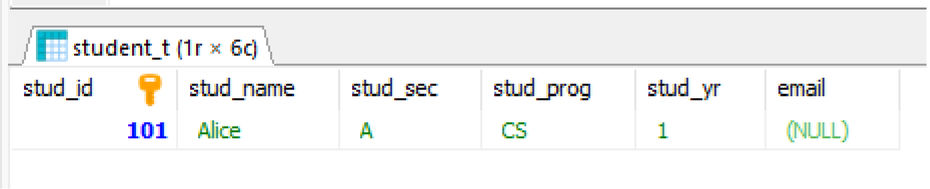


```sql
USE student;

SELECT *
FROM student_t
WHERE stud_sec = 'A';

```

Output:

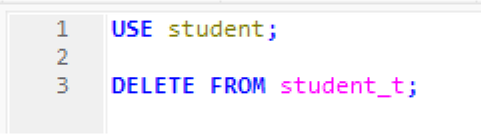


student_t (1r × 6c)

stud_id	stud_name	stud_sec	stud_prog	stud_yr	email
101	Alice	A	CS	1	(NULL)

5. Delete all records from the student_t table but retain the table structure.

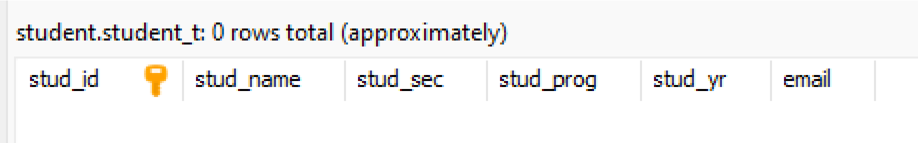
Query:



```sql
USE student;

DELETE FROM student_t;
```

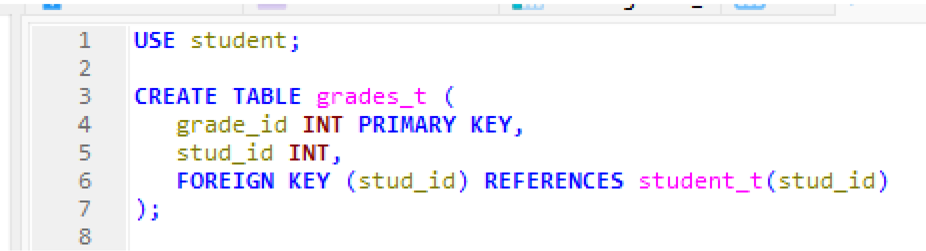
Output:



```
student.student_t: 0 rows total (approximately)
stud_id 🔑 | stud_name | stud_sec | stud_prog | stud_yr | email
```

6. Create a table grades_t with the following constraints:
    - grade_id as the Primary Key
    - stud_id as a Foreign Key referencing student_t

Query:

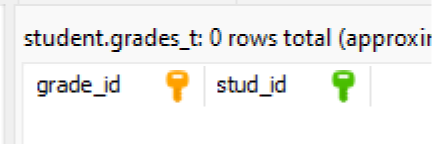


```sql
USE student;

CREATE TABLE grades_t (
   grade_id INT PRIMARY KEY,
   stud_id INT,
   FOREIGN KEY (stud_id) REFERENCES student_t(stud_id)
);

```

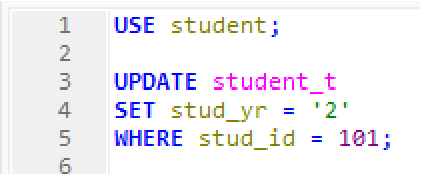
Output:



```
student.grades_t: 0 rows total (approxi
grade_id 🔑 | stud_id 🔑
```

7. Update the stud_yr of the student with stud_id 101 to "2".

Query:

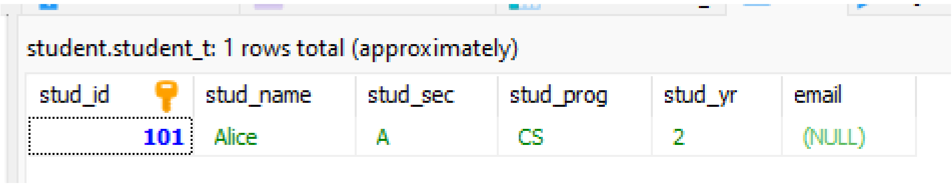


```sql
USE student;

UPDATE student_t
SET stud_yr = '2'
WHERE stud_id = 101;

```

Output:

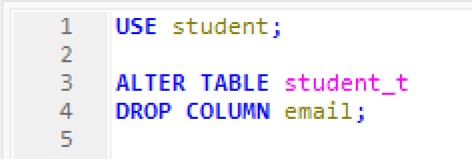


student.student_t: 1 rows total (approximately)

stud_id	stud_name	stud_sec	stud_prog	stud_yr	email
101	Alice	A	CS	2	(NULL)

8. Write the syntax to remove the email column from the student_t table.

Query:

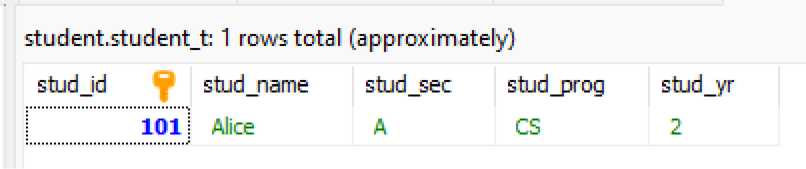


```
USE student;

ALTER TABLE student_t
DROP COLUMN email;

```

Output:

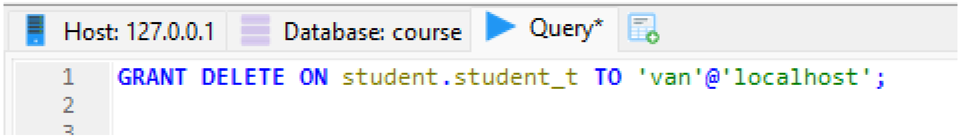


student.student_t: 1 rows total (approximately)

stud_id	stud_name	stud_sec	stud_prog	stud_yr
101	Alice	A	CS	2

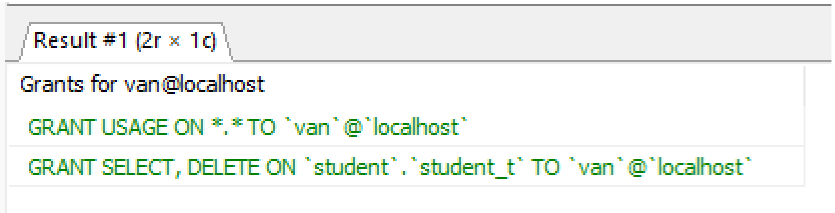
9. Grant permission to a user to query the student_t table.

Query:



```
GRANT DELETE ON student.student_t TO 'van'@'localhost';
```

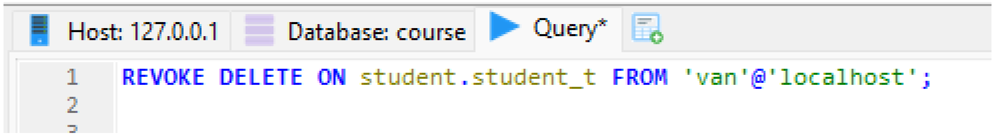
Output:



Grants for van@localhost
GRANT USAGE ON *.* TO `van`@`localhost`
GRANT SELECT, DELETE ON `student`.`student_t` TO `van`@`localhost`

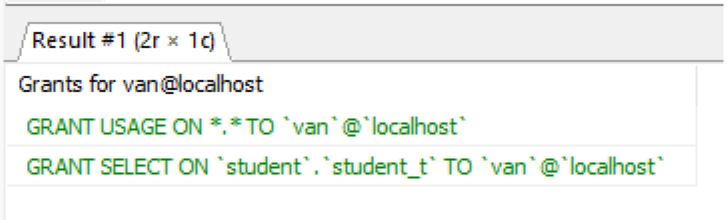
10. Revoke permission for a user to delete records from the student_t table.

Query:



```
REVOKE DELETE ON student.student_t FROM 'van'@'localhost';
```

Output:



```
Grants for van@localhost
GRANT USAGE ON *.* TO `van`@`localhost`
GRANT SELECT ON `student`.`student_t` TO `van`@`localhost`
```