Databases: Exercises 8 (16p/16p)

Task 1 [2p/2p]

Create such a trigger in the sample database of the study course http://netisto.fi/oppaat/tietokannat/?id=03 that hometowns with more than one million inhabitants cannot be added to the table with the `INSERT INTO`*-phrase* `cities`*.*
The return must show that the addition of hometowns with more than one million inhabitants is not possible. Remove the trigger you created after testing its functionality.

ANSWER:

To create a trigger, syntax:

```
CREATE TRIGGER trigger_name [BEFORE|AFTER] event ON table_name
trigger_type
BEGIN
  -- trigger_logic
END;
```

specific event e.g., INSERT, UPDATE, or DELETE.

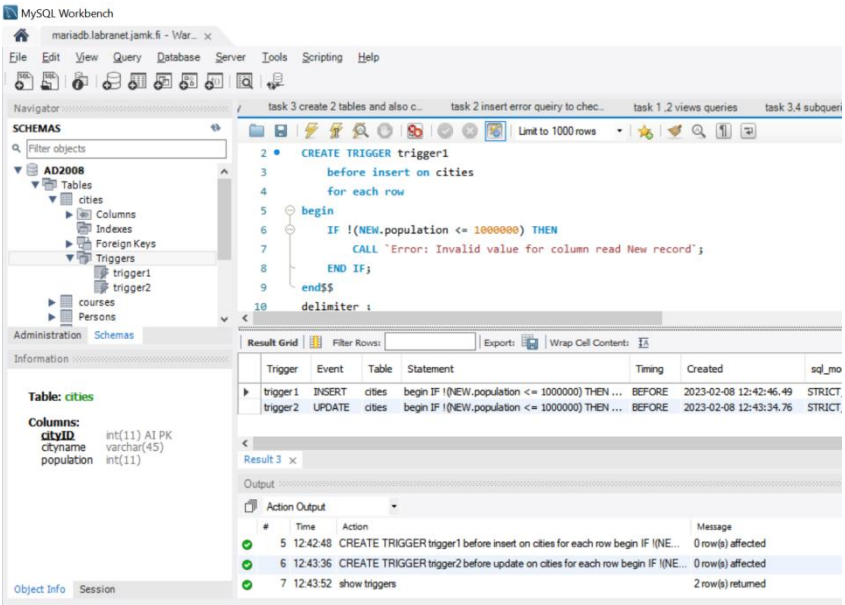
Trigger-types are FOR EACH ROW or FOR EACH STATEMENT

Solution:

The following statement used to create trigger for the above question and the results are shown in the screenshot.

```
DELIMITER $$
CREATE TRIGGER trigger1
      before insert on cities
      for each row
begin
      IF !(NEW.population <= 1000000) THEN
            CALL `Error: Invalid value for column read New record`;
    END IF;
end$$
delimiter ;


DELIMITER $$
CREATE TRIGGER trigger2
      before update on cities
      for each row
begin
      IF !(NEW.population <= 1000000) THEN
            CALL `Error: Invalid value for column read New record`;
    END IF;
```

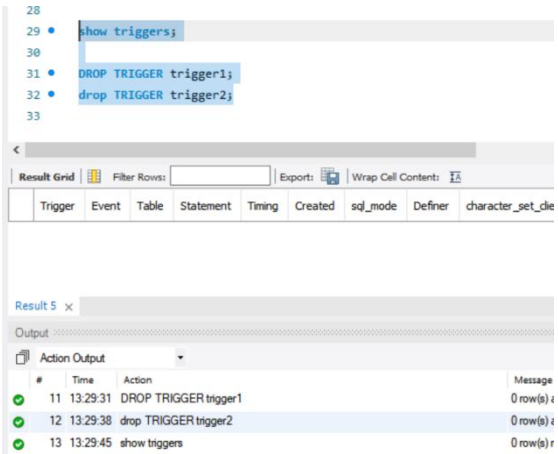
```
end$$
delimiter ;

show triggers;
```



when the event is called it showed an error (trigger is activated when tried to insert the value for the population more than one million) the reults are in the screenshot.

The *DROP TRIGGER* statement deletes a trigger from the database.

Syntax *DROP TRIGGER [IF EXISTS] [schema_name.]trigger_name;*



Task 2 [2p/2p]

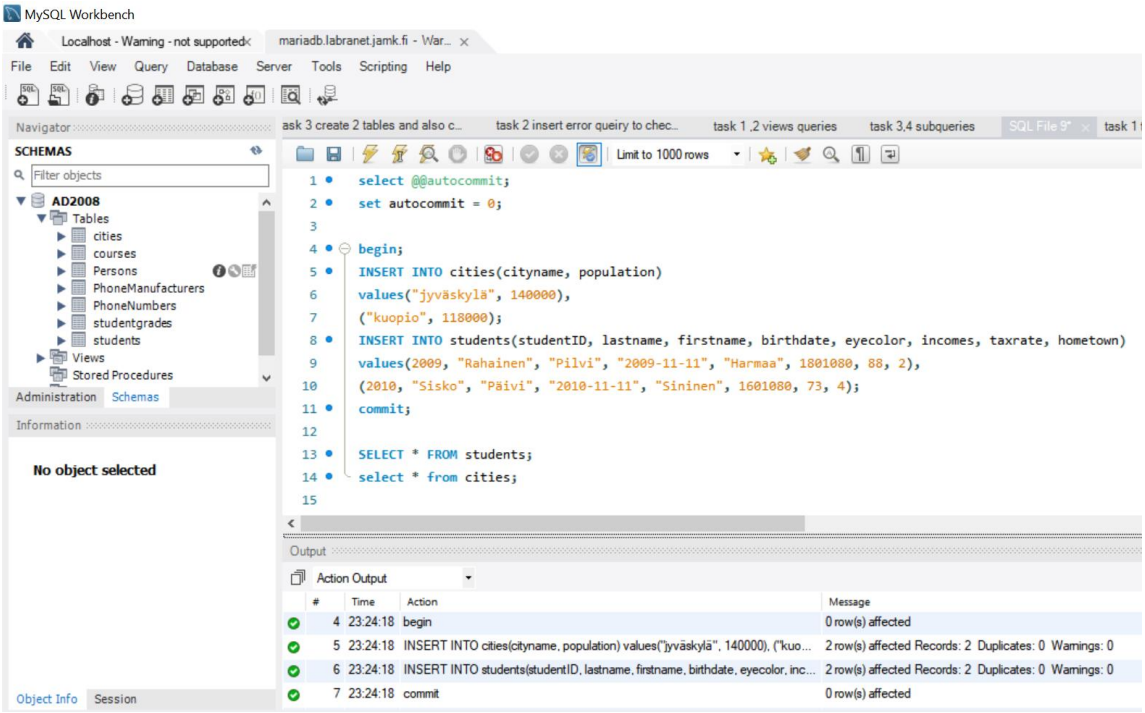
Create a transaction where the sample database of the study course http://netisto.fi/oppaat/tietokannat/?id=03

1) Two new cities are added to the citiestable with one INSERT INTO statement AND
2) two new students are added to the studentstable with one INSERT INTO statement, whose hometown is either of the hometowns added in point A)
The return must show that all additions specified in the aforementioned transaction were successful.

To add multiple rows to a table at once, the following form of the *INSERT* statement is used: syntax

```
INSERT INTO table_name (column_list)VALUES
    (value_list_1),
    (value_list_2),
    ...
    (value_list_n);
```

```
select @@autocommit;
set autocommit = 0;

begin;
INSERT INTO cities(cityname, population)
values("jyväskylä", 140000),
("kuopio", 118000);
INSERT INTO students(studentID, lastname, firstname, birthdate,
eyecolor, incomes, taxrate, hometown)
values(2009, "Rahainen", "Pilvi", "2009-11-11", "Harmaa", 1801080, 88,
2),
(2010, "Sisko", "Päivi", "2010-11-11", "Sininen", 1601080, 73, 4);
commit;
```



Task 3 [2p/2p]

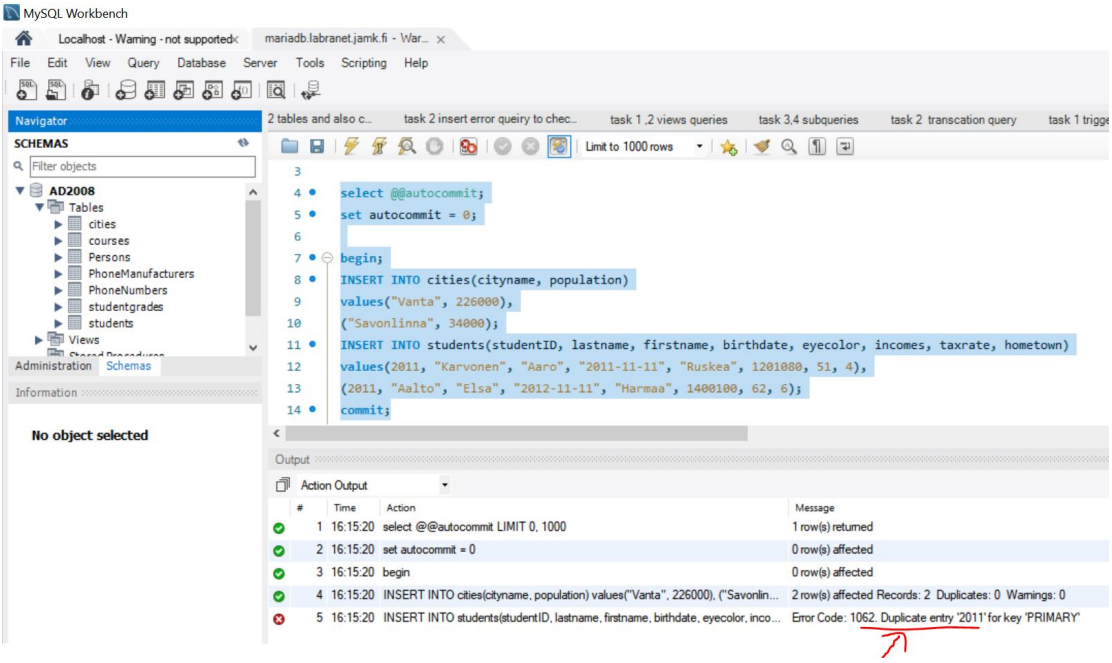
Create a transaction where the sample database of the study course http://netisto.fi/oppaat/tietokannat/?id=03

1) Two new cities are added to the `citiestable` with one `INSERT INTO`-sentence AND
2) two new students are added to the table with one -sentence whose studentID is erroneously the samestudentsINSERT INTO.
The return should show that the addition of cities was successful (despite the transaction) but the addition of students was not.

ANSWER:

It is done as same as previous task ,inserting the student ID with the same studentID which is already exists, then it showed the following error and other insert into cities is successed. The results are in the screen shot.

```
select @@autocommit;
set autocommit = 0;

begin;
INSERT INTO cities(cityname, population)
values("Vanta", 226000),
("Savonlinna", 34000);
INSERT INTO students(studentID, lastname, firstname, birthdate,
eyecolor, incomes, taxrate, hometown)
values(2011, "Karvonen", "Aaro", "2011-11-11", "Ruskea", 1201080, 51,
4),
(2011, "Aalto", "Elsa", "2012-11-11", "Harmaa", 1400100, 62, 6);
commit;
```



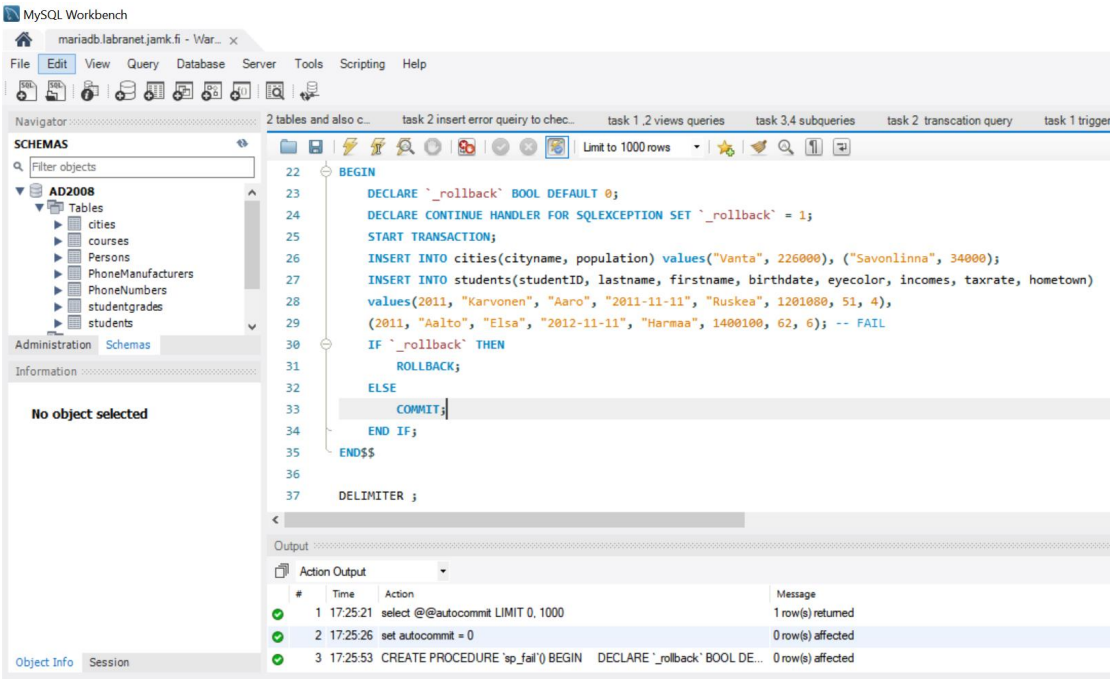
Task 4 [4p/4p]

In connection with the transaction of the previous task, create `sp_fail()` *a stored procedure (stored procedure) which, when called, will roll back all* `INSERT INTO`*-statements if the execution of one of them fails for some reason. If all INSERT INTO statements are executable, the transaction is committed in its entirety (COMMIT).*

Use the stored procedure you created by calling it `CALL sp_fail;`

The return must show that the addition of the cities was also unsuccessful, if the addition of a student was not successful.

```
select @@autocommit;
set autocommit = 0;

DELIMITER $$

CREATE PROCEDURE `sp_fail`()
BEGIN
    DECLARE `_rollback` BOOL DEFAULT 0;
    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;
    START TRANSACTION;
    INSERT INTO cities(cityname, population) values("Vanta", 226000),
("Savonlinna", 34000);
      INSERT INTO students(studentID, lastname, firstname, birthdate,
eyecolor, incomes, taxrate, hometown)
      values(2011, "Karvonen", "Aaro", "2011-11-11", "Ruskea",
1201080, 51, 4),
      (2011, "Aalto", "Elsa", "2012-11-11", "Harmaa", 1400100, 62, 6);
-- FAIL
    IF `_rollback` THEN
        ROLLBACK;
    ELSE
        COMMIT;
    END IF;
END$$

DELIMITER ;
```

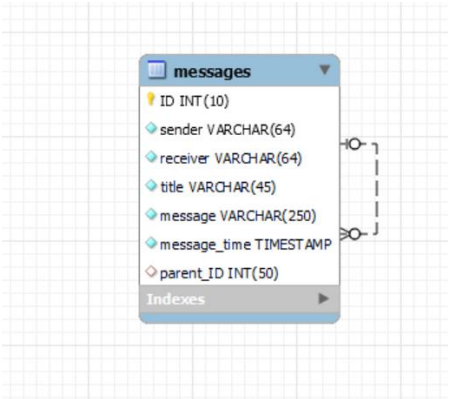


Task 5 [6p/6p]

A) Create an imaginary table for storing the messages of the discussion forum software in a database with a running number (integer) as the primary key. Other fields should be at least for the title, author and content of the message. The time when the message was added must also be recorded. In addition, there must be an INTEGER-type "parent" field, which tells with its number which message it might be the answer to. The value of the "Parent" field is set to NULL if the message is a thread opening message. The "Parent" field should be a reference key to the table's primary key.

ANSWER:

From the given question forum software database is created with a table which has both primary and reference key, where the each primary key ID (messages) has zero or many reference keys (replies). It is shown the EER diagram. Forwared engineer is done for this EER diagram in the MYSQL workbench



Description of the table message is shown in the given screenshot

```
desc messages;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
ID	int(10)	NO	PRI	NULL	auto_increment
sender	varchar(64)	NO		NULL	
receiver	varchar(64)	NO		NULL	
title	varchar(45)	NO		NULL	
message	varchar(250)	NO		NULL	
message_time	timestamp	NO		current_timestamp()	
parent_ID	int(50)	YES	MUL	NULL	

B) Add content to the board so that there are messages on at least three levels, as in the attached example. So there are messages that open a conversation, their answers and answers to answers. Of course.

- *The values are added by Insert Into sql statement, the results are shown in the given screenshot by the following sql statement*

```
select * from forum_software_db.messages;
```

flea_market_db
forum_software_db
Tables
messages
Views
Stored Procedures
Functions

Table: messages

Columns:

ID	int(10) AI PK
sender	varchar(64)
receiver	varchar(64)
title	varchar(45)
message	varchar(250)
message_time	timestamp
parent_ID	int(50)

```
select * from forum_software_db.messages;
```

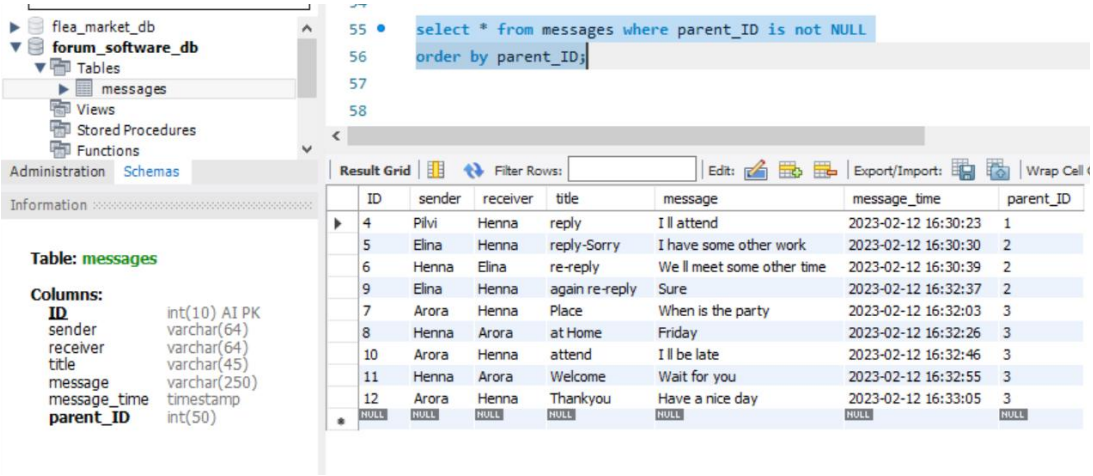
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Conte

ID	sender	receiver	title	message	message_time	parent_ID
1	Henna	Pilvi	invitation	birthday	2023-02-12 16:29:39	NULL
2	Henna	Elina	invitation	birthady	2023-02-12 16:30:10	NULL
3	Henna	Arora	invitation	birthady	2023-02-12 16:30:15	NULL
4	Pilvi	Henna	reply	I ll attend	2023-02-12 16:30:23	1
5	Elina	Henna	reply-Sorry	I have some other work	2023-02-12 16:30:30	2
6	Henna	Elina	re-reply	We ll meet some other time	2023-02-12 16:30:39	2
7	Arora	Henna	Place	When is the party	2023-02-12 16:32:03	3
8	Henna	Arora	at Home	Friday	2023-02-12 16:32:26	3
9	Elina	Henna	again re-reply	Sure	2023-02-12 16:32:37	2
10	Arora	Henna	attend	I ll be late	2023-02-12 16:32:46	3
11	Henna	Arora	Welcome	Wait for you	2023-02-12 16:32:55	3
12	Arora	Henna	Thankyou	Have a nice day	2023-02-12 16:33:05	3
NULL	NULL	NULL	NULL	NULL	NULL	NULL

C) <u>Make a SQL query that retrieves all responses from first-level messages.</u>

The following sql statement retrieves the responses from first level messages

```
select * from messages where parent_ID is not NULL
order by parent_ID;
```



ID	sender	receiver	title	message	message_time	parent_ID
4	Pilvi	Henna	reply	I ll attend	2023-02-12 16:30:23	1
5	Elina	Henna	reply-Sorry	I have some other work	2023-02-12 16:30:30	2
6	Henna	Elina	re-reply	We ll meet some other time	2023-02-12 16:30:39	2
9	Elina	Henna	again re-reply	Sure	2023-02-12 16:32:37	2
7	Arora	Henna	Place	When is the party	2023-02-12 16:32:03	3
8	Henna	Arora	at Home	Friday	2023-02-12 16:32:26	3
10	Arora	Henna	attend	I ll be late	2023-02-12 16:32:46	3
11	Henna	Arora	Welcome	Wait for you	2023-02-12 16:32:55	3
12	Arora	Henna	Thankyou	Have a nice day	2023-02-12 16:33:05	3
NULL	NULL	NULL	NULL	NULL	NULL	NULL