**ADT Lab 4 (10pts)**

**Name: Saiabhinav Chekka(schekka@iu.edu)  
Spring 2024 (Final semester)**

**Q1**

**Q2**

**Q3**

**Q4**

**Highlight the questions you are done with to green colour before you submit**

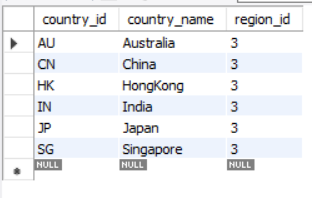
Setup Instructions-

1. Download the files ADT Lab 4.docx, create\_table.sql, insert\_table.sql files from canvas.
2. Create a new database with your IU username (or use the one that you might have created in previous labs)
3. Run the sql commands in create\_table.sql to create new tables.
4. Run the sql commands in insert\_table.sql to populate all the tables.

Questions

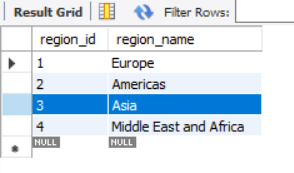
1. Observe the create queries in the file create\_table.sql. There are foreign key constraints on several tables.

Run the quey- SELECT \* FROM countries WHERE region\_id=3;



See that we have several records with region\_id 3. The column region\_id in countries table is a foreign key which references the column region\_id in regions table.

SELECT \* FROM REGIONS;



In country table region\_id corresponds to the region\_name Asia.

Let’s try to delete the region\_id 3 from table country.

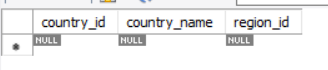
Run the query-

DELETE FROM regions WHERE region\_id=3;

After successfully executing this query, check the countries table again-

SELECT \* FROM countries where region\_id=3;

It should show an empty result



Why did this happen? Even though we deleted the records from regions table all the records in the country table with region\_id=3 vanished.

Write an explanation below with the reason.

# Write your answer below.

The "regions" table and the "countries" table are separated by a foreign key constraint. To be more precise, the "region\_id" in the "regions" table is referenced by the "region\_id" in the "countries" table through the use of a foreign key.

When the region\_id 3 is deleted from the regions table, the corresponding entries in the countries table that have a region\_id of 3 will no longer have a valid reference.

The foreign key field in the child table will be set to NULL for all matching rows when the referenced row in the parent table is modified or removed.

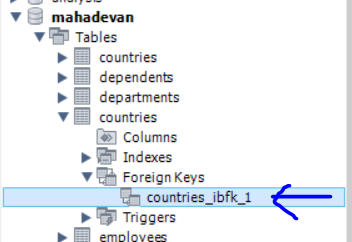
When region\_id 3 was deleted from the regions table, the region\_id for all countries that referenced this region\_id ‘3’ was set to NULL to maintain the integrity of the database.

Hint-

<https://dba.stackexchange.com/questions/74627/difference-between-on-delete-cascade-on-update-cascade-in-mysql> (1st answer)

Research more on Cascade if you need more clarity

1. Now let us drop and create a new foreign key constraint
2. Note the name of the foreign key constraint under the Foreign Keys list

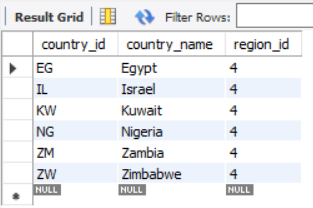


Run the query-

ALTER TABLE countries DROP FOREIGN KEY countries\_ibfk\_1;

SELECT \* FROM countries where region\_id=4;

We see the following entries in the county table for region\_id=4

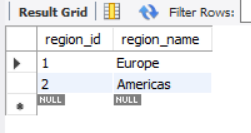


Now let’s delete the region 4 from the regions table using the following query

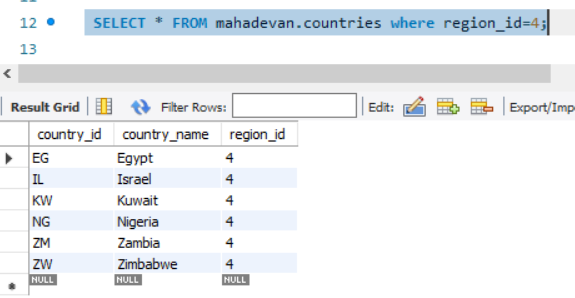
DELETE FROM regions WHERE region\_id=4;

SELECT \* FROM REGIONS;

Once deleted we now only have 2 regions in the regions table-



Now let’s check the countries table-



We still have all the entries corresponding to region\_id 4 in the countries table even after deleting the region\_id 4 in regions table.

Why did this happen?

# Write your answer below.

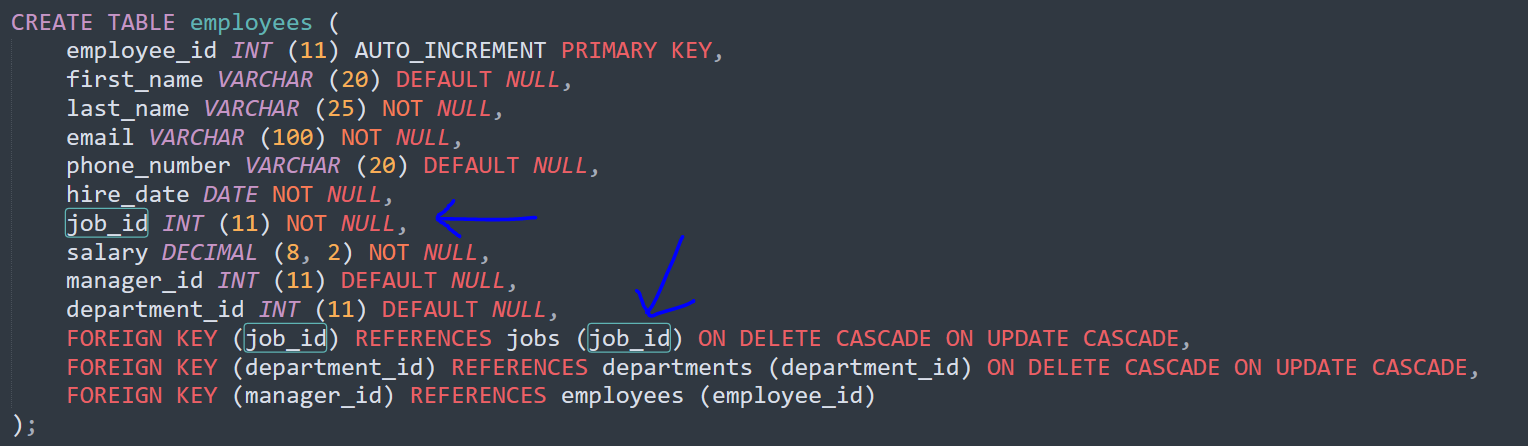
Because the foreign key constraint countries\_ibfk\_1 was dropped, the entries corresponding to region\_id 4 in the nations table remained after the region\_id 4 in the regions table was erased.

The removal of region\_id 4 from the regions table has no effect on the countries table since there is no rule enforcing referential integrity between the regions and countries tables in the absence of a foreign key constraint.

1. Now let’s try to delete some tables

DROP TABLE jobs;

You won’t be allowed to delete this table because of the foreign key constraint that is being used in the employees table.



In order to delete the job table, you first need to drop the foreign\_key constraint and then delete the table jobs

# Delete the jobs table and post the screenshot below

Hint- First drop the all the foreign key constraint on employee’s table. Then drop the jobs table

ALTER TABLE employees DROP FOREIGN KEY employees\_ibfk\_1;

#Sample screenshot with no jobs table-



# Your screenshot

A screenshot of a computer

Description automatically generated

1. Now drop the table regions and countries.

DROP TABLE regions;

If you get an error while deleting the countries table that’s because there is a foreign key in the locations table that’s referencing the column country\_id in countries table.

So first drop the foreign key constraint from locations table

ALTER TABLE locations DROP FOREIGN KEY locations\_ibfk\_1;

Now drop the countries table- Drop table countries;

DROP TABLE countries;

Now recreate the countries and region table with the below queries-

CREATE TABLE regions (

region\_id INT (11) AUTO\_INCREMENT PRIMARY KEY,

region\_name VARCHAR (25) DEFAULT NULL

);

CREATE TABLE countries (

country\_id CHAR (2) PRIMARY KEY,

country\_name VARCHAR (40) DEFAULT NULL,

region\_id INT (11) NOT NULL,

FOREIGN KEY (region\_id) REFERENCES regions (region\_id) ON DELETE RESTRICT

);

INSERT INTO regions(region\_id,region\_name) VALUES (1,'Europe');

INSERT INTO regions(region\_id,region\_name) VALUES (2,'Americas');

INSERT INTO regions(region\_id,region\_name) VALUES (3,'Asia');

INSERT INTO regions(region\_id,region\_name) VALUES (4,'Middle East and Africa');

/\*Data for the table countries \*/

INSERT INTO countries(country\_id,country\_name,region\_id) VALUES ('AR','Argentina',2);

INSERT INTO countries(country\_id,country\_name,region\_id) VALUES ('AU','Australia',3);

INSERT INTO countries(country\_id,country\_name,region\_id) VALUES ('BE','Belgium',1);

INSERT INTO countries(country\_id,country\_name,region\_id) VALUES ('BR','Brazil',2);

INSERT INTO countries(country\_id,country\_name,region\_id) VALUES ('CA','Canada',2);

INSERT INTO countries(country\_id,country\_name,region\_id) VALUES ('CH','Switzerland',1);

INSERT INTO countries(country\_id,country\_name,region\_id) VALUES ('CN','China',3);

INSERT INTO countries(country\_id,country\_name,region\_id) VALUES ('DE','Germany',1);

INSERT INTO countries(country\_id,country\_name,region\_id) VALUES ('DK','Denmark',1);

INSERT INTO countries(country\_id,country\_name,region\_id) VALUES ('EG','Egypt',4);

INSERT INTO countries(country\_id,country\_name,region\_id) VALUES ('FR','France',1);

INSERT INTO countries(country\_id,country\_name,region\_id) VALUES ('HK','HongKong',3);

Now we will try to delete from the regions table-

DELETE FROM regions WHERE region\_id=1;

It will throw an error.

Why did in throw an error?

# Write your explanation here

The error when attempting to delete region\_id=1 from the regions table occurred because there is a FOREIGN KEY constraint with the ON DELETE RESTRICT action on the country\_id column in the countries table.

As a result, any attempt to remove a record from the regions database that a country in the countries table references will be rejected. By forbidding the deletion of a region that a country is currently referencing, the database ensures referential integrity.

Since the countries table is a child table and deleting a child record does not compromise the referential integrity of the parent table (regions), we are able to remove records from it.

Hint- It has something to do with the foreign key constraint that we added when we created the table

But you can safely delete records from the countries table (child table).