**GUIA BÁSICA SQL**

1. **CREATE TABLA (CREATE TABLE)**

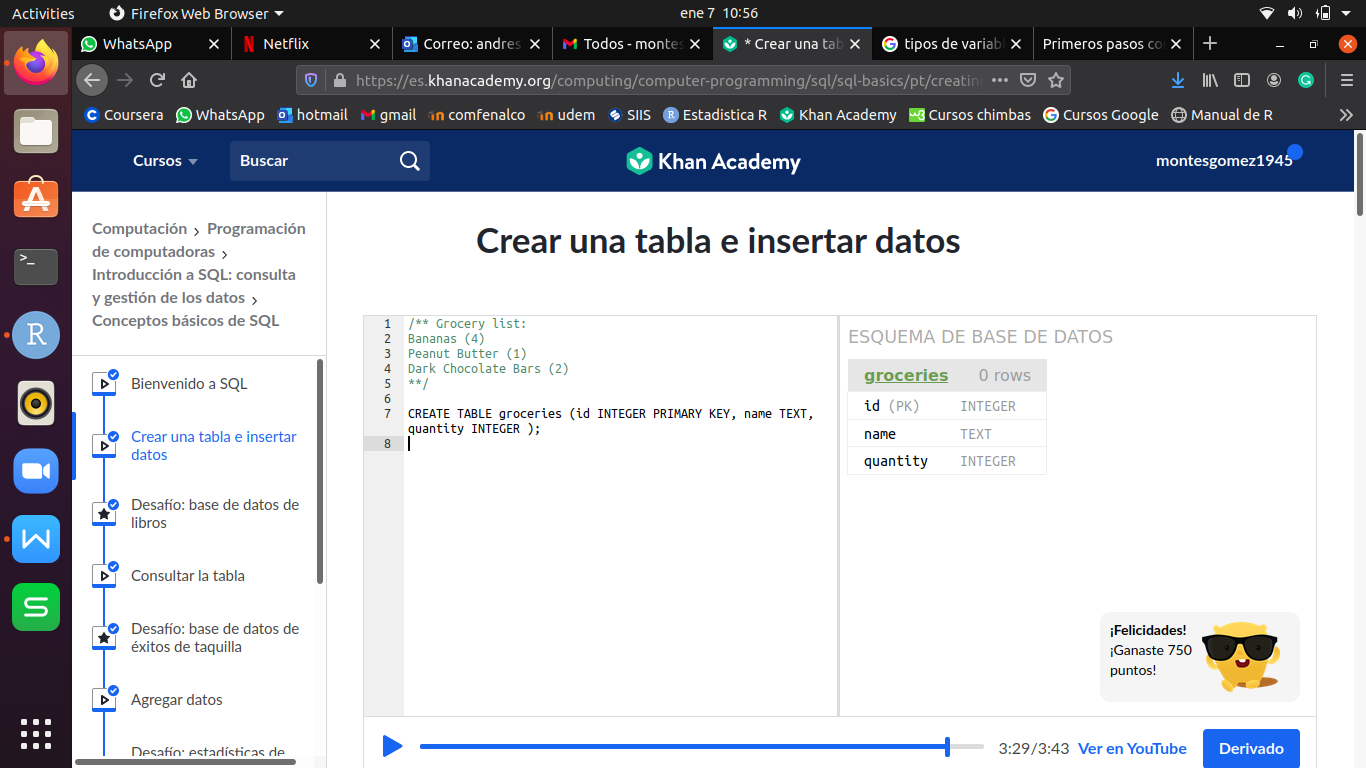
CREATE TABLE nom\_table (id INTEGER PRIMARY KEY, col1\_name TYPE, col2\_name TYPE, ...., coln\_name TYPE);

- It is better start with an id as **PRIMARY KEY**.

- Most frequently **TYPE** are: **TEXT** , **INTEGER** (number without decimals), **REAL** and **BLOB** (keep the input format).

**EXAMPLE:**

CREATE TABLE groceries (id INTEGER PRIMARY KEY, name TEXT, quantity INTEGER );



1. **INSERT VALUES IN A TABLE (INSERT INTO)**

INSERT INTO nom\_table VALUES (‘text’, 5, 2,7, ....);

- For each line, we have to make an INSERT.

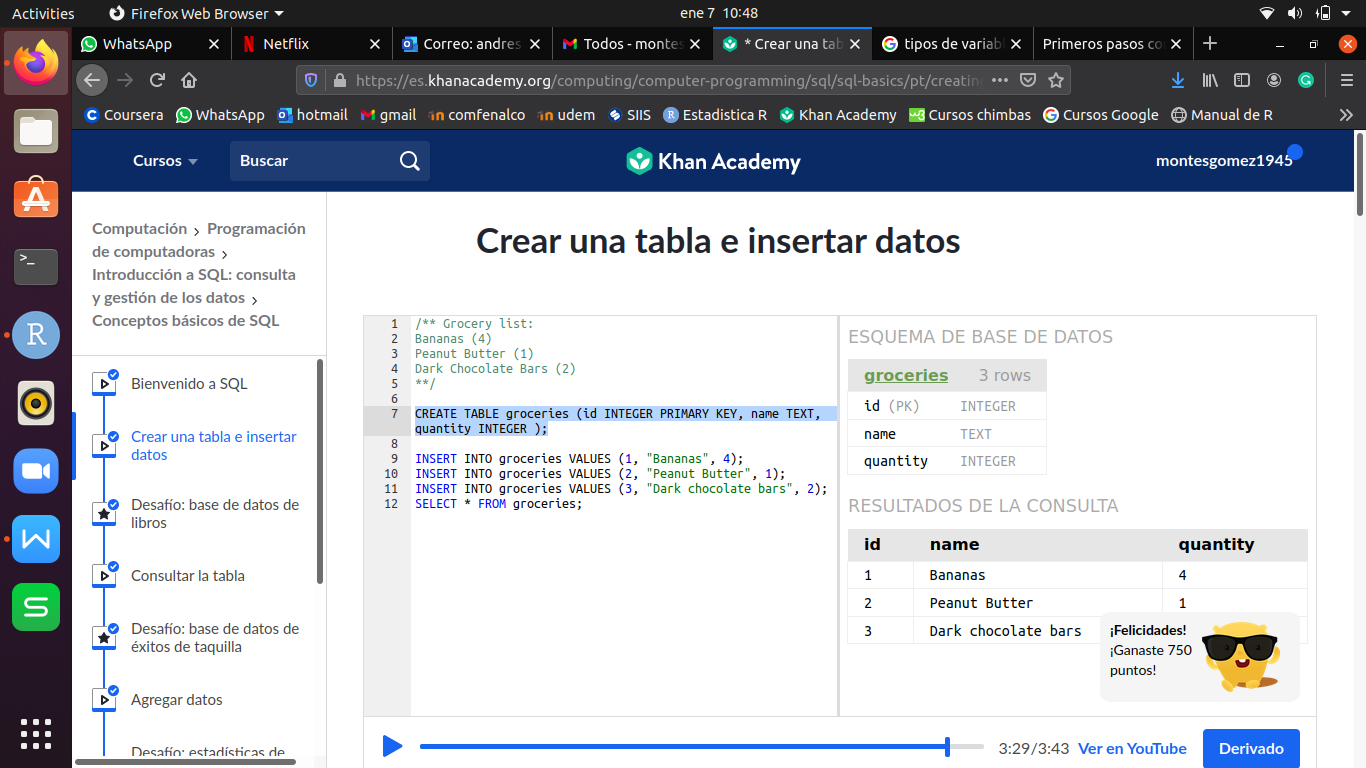
- Text between quotation marks and can be double (“ ”) o simple (‘ ’), SQL accept both.

**EXAMPLE:**

INSERT INTO groceries VALUES (1, "Bananas", 4);

INSERT INTO groceries VALUES (2, "Peanut Butter", 1);

INSERT INTO groceries VALUES (3, "Dark chocolate bars", 2);

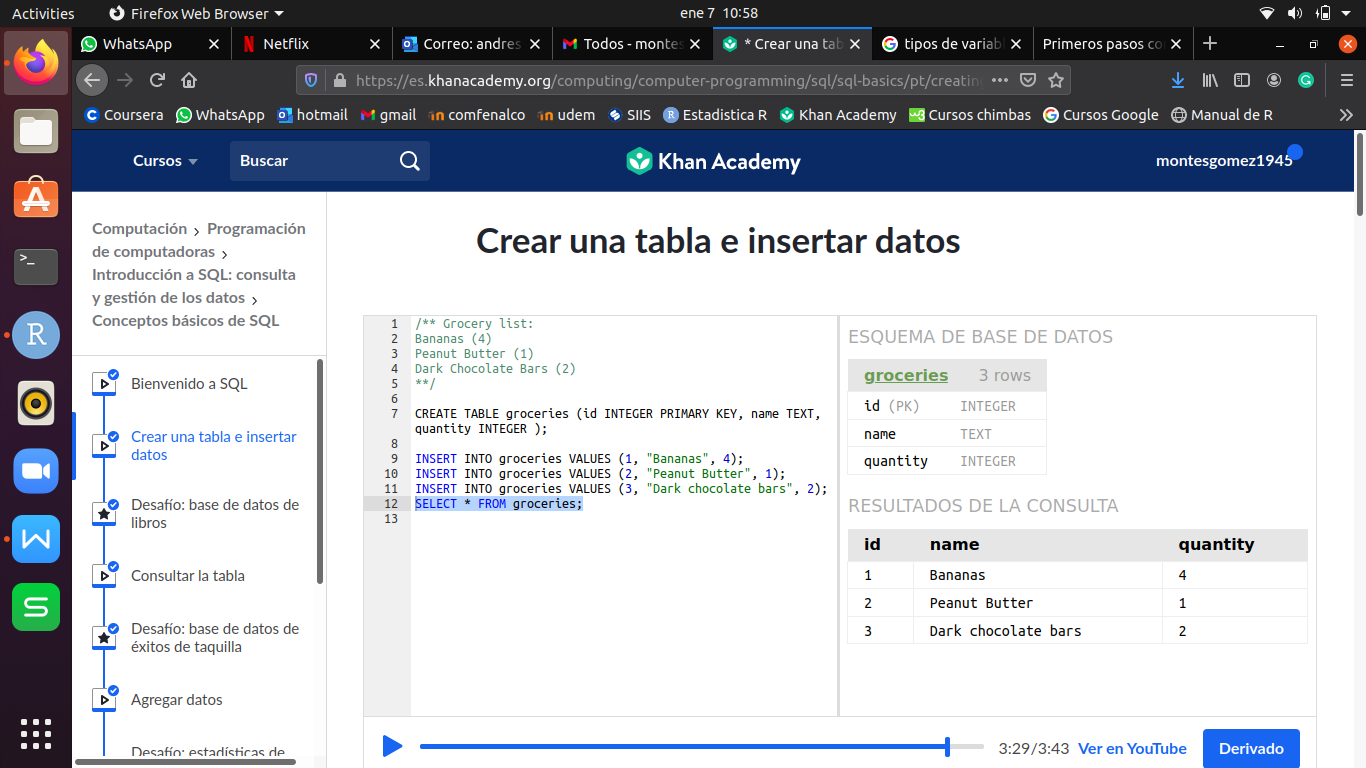


1. **VIEW TABLE (SELECT)**

SELECT \* FROM nom\_table;

**EXAMPLE:**

SELECT \* FROM groceries;

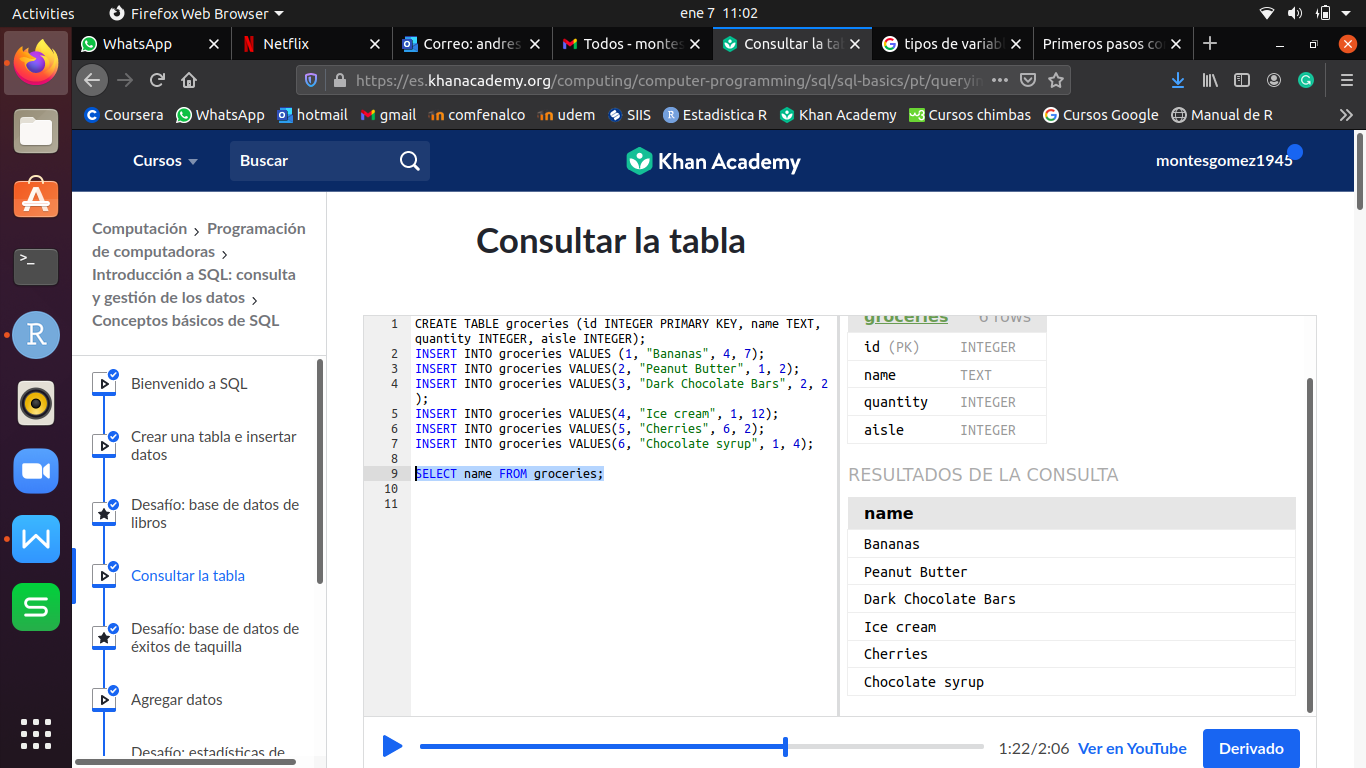


1. **VIEW A COLUMN**

SELECT nom\_col FROM nom\_table;

**EXAMPLE:**

SELECT name FROM groceries;

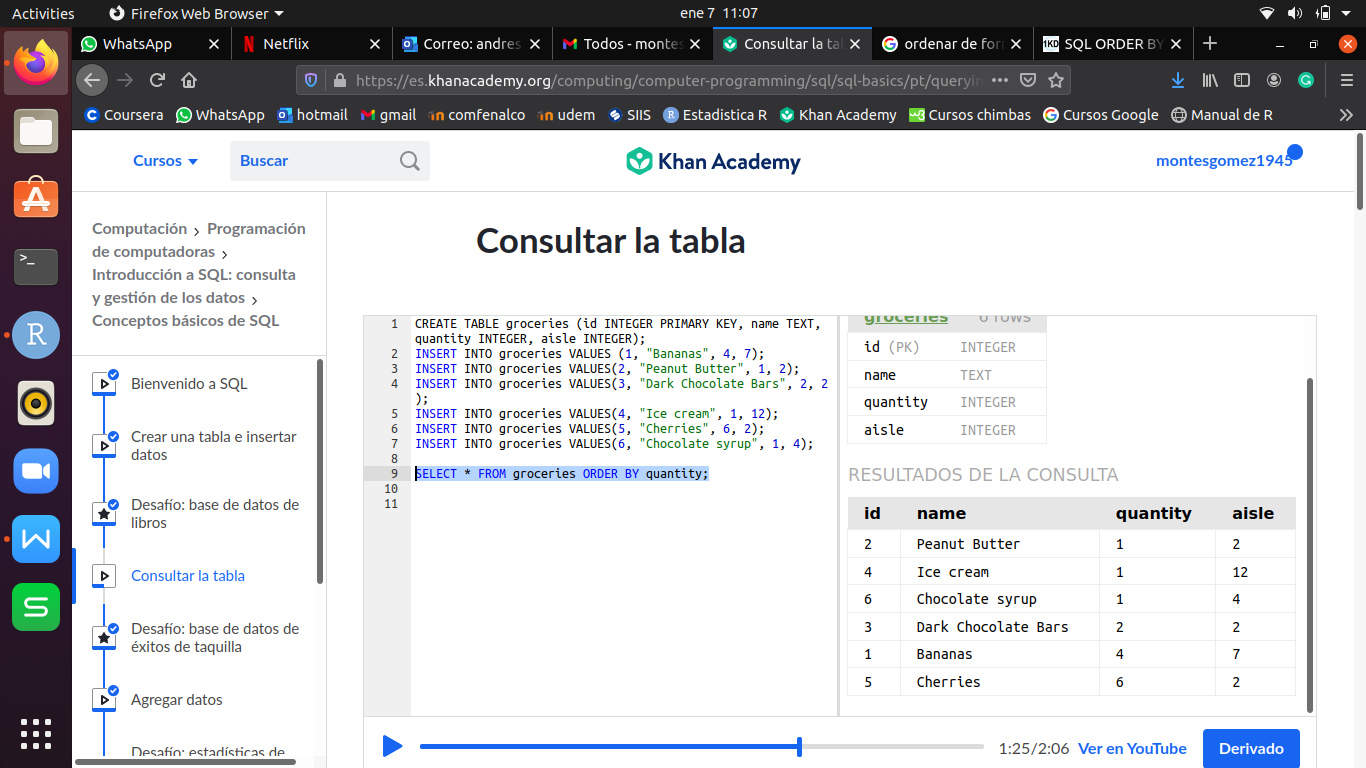


1. **ARRANGE TABLE ACCORDING TO A COLUMN (ORDER BY)**

SELECT \* FROM nom\_table ORDER BY nom\_col DESC ó ASC;

**EXAMPLE:**

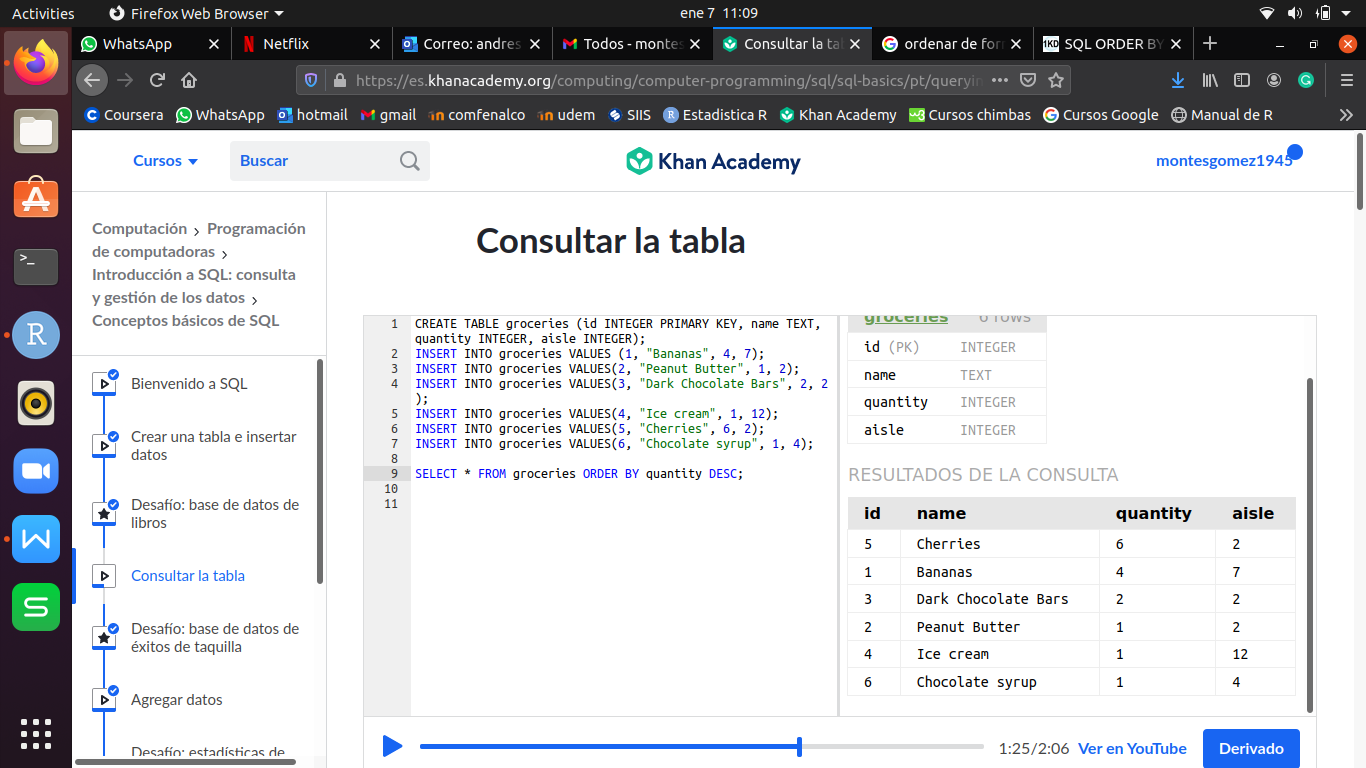
SELECT \* FROM groceries ORDER BY quantity;



- By default SQL arrange ascending, if we want to order descendingly we have to add a **DESC** at the end.

**EXAMPLE:**

SELECT \* FROM groceries ORDER BY quantity DESC;



1. **FILTER (WHERE)**

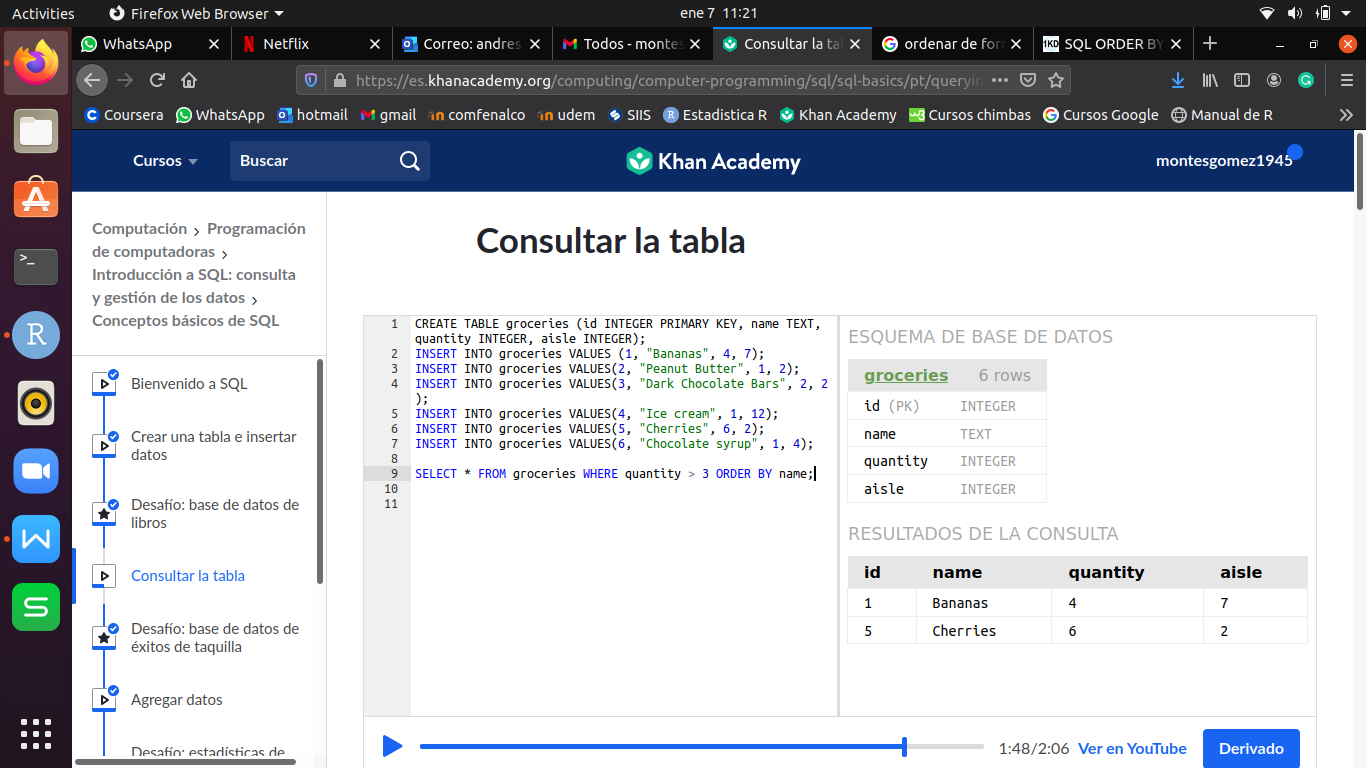
... WHERE operador 5, 3.2, ‘text’, ....

- Operador could be >, <, =, >=, <=, ...

- ORDER BY always at the end.

**EXAMPLE:**

SELECT \* FROM groceries WHERE quantity > 3 ORDER BY name;



- It is possible apply filter with most of one condition adding **OR, AND.**

**EXAMPLE:**

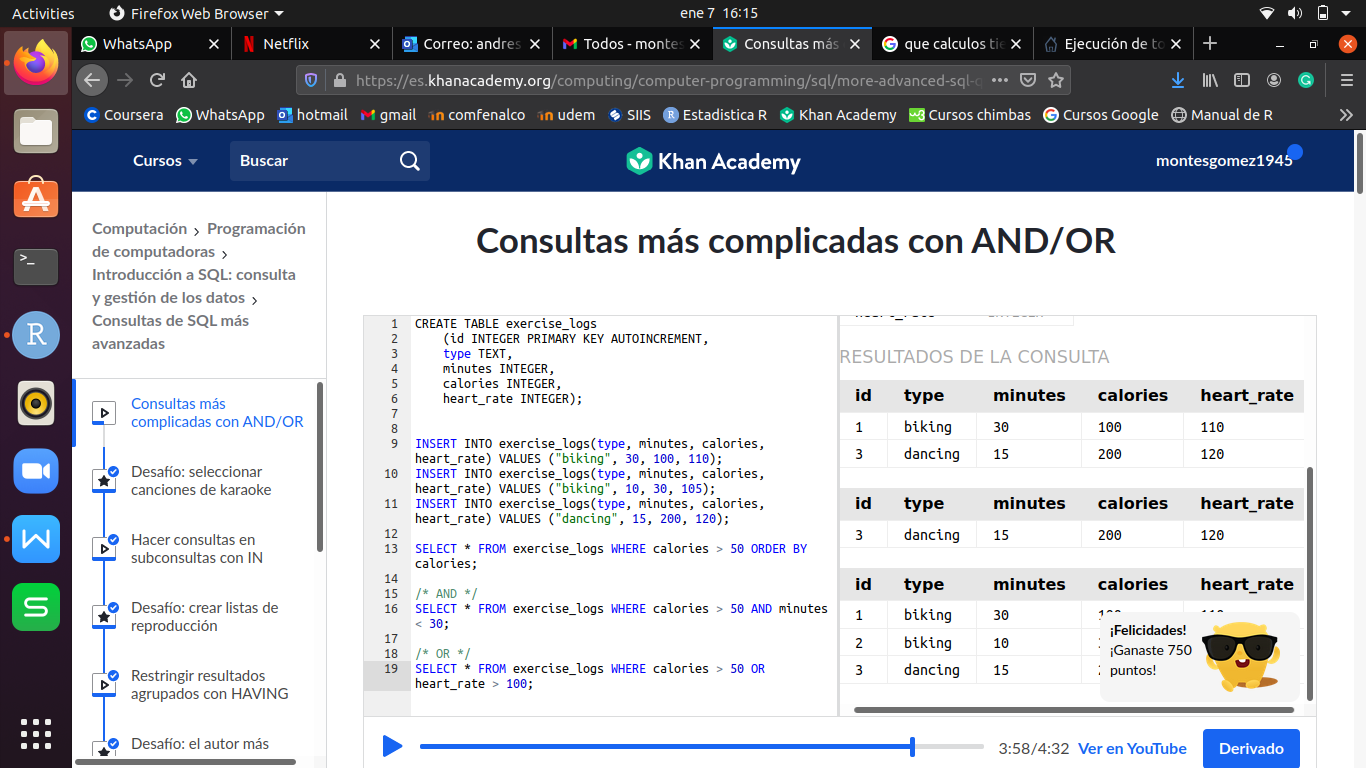
CREATE TABLE exercise\_logs (id INTEGER PRIMARY KEY AUTOINCREMENT, type TEXT, minutes INTEGER, calories INTEGER, heart\_rate INTEGER);

INSERT INTO exercise\_logs(type, minutes, calories, heart\_rate) VALUES ("biking", 30, 100, 110);

INSERT INTO exercise\_logs(type, minutes, calories, heart\_rate) VALUES ("biking", 10, 30, 105);

INSERT INTO exercise\_logs(type, minutes, calories, heart\_rate) VALUES ("dancing", 15, 200, 120);

SELECT \* FROM exercise\_logs WHERE calories > 50 AND minutes < 30;



SELECT \* FROM exercise\_logs WHERE calories > 50 OR heart\_rate > 100;



1. **FUNCTIONS**

SELECT FUNCTION(nom\_col) FROM nom\_table;

- **FUNCTION** can be **SUM**, **AVG**(Mean), **MAX, MIN**.

INSERT INTO groceries VALUES (1, "Bananas", 56, 7);

INSERT INTO groceries VALUES(2, "Peanut Butter", 1, 2);

INSERT INTO groceries VALUES(3, "Dark Chocolate Bars", 2, 2);

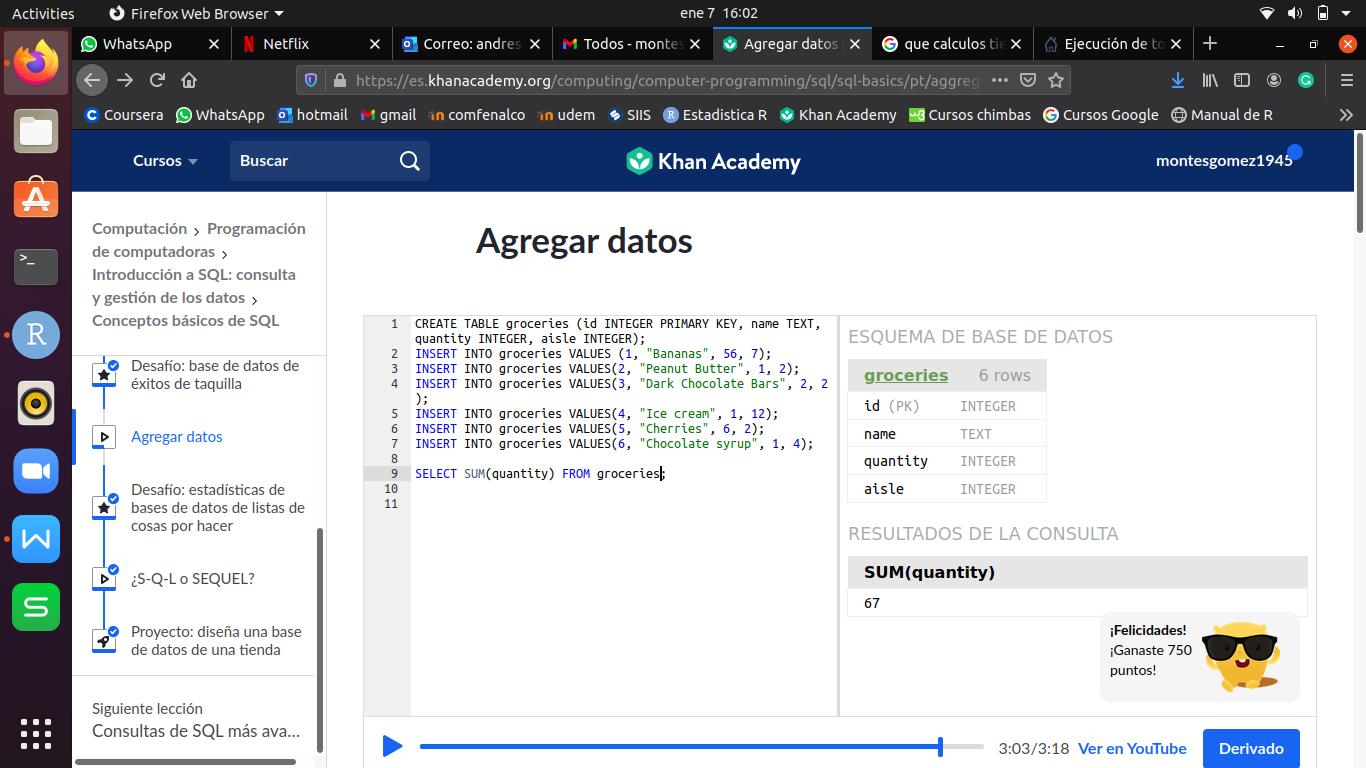
INSERT INTO groceries VALUES(4, "Ice cream", 1, 12);

INSERT INTO groceries VALUES(5, "Cherries", 6, 2);

INSERT INTO groceries VALUES(6, "Chocolate syrup", 1, 4);

**EXAMPLE:**

SELECT aisle, SUM(quantity) FROM groceries GROUP BY aisle;

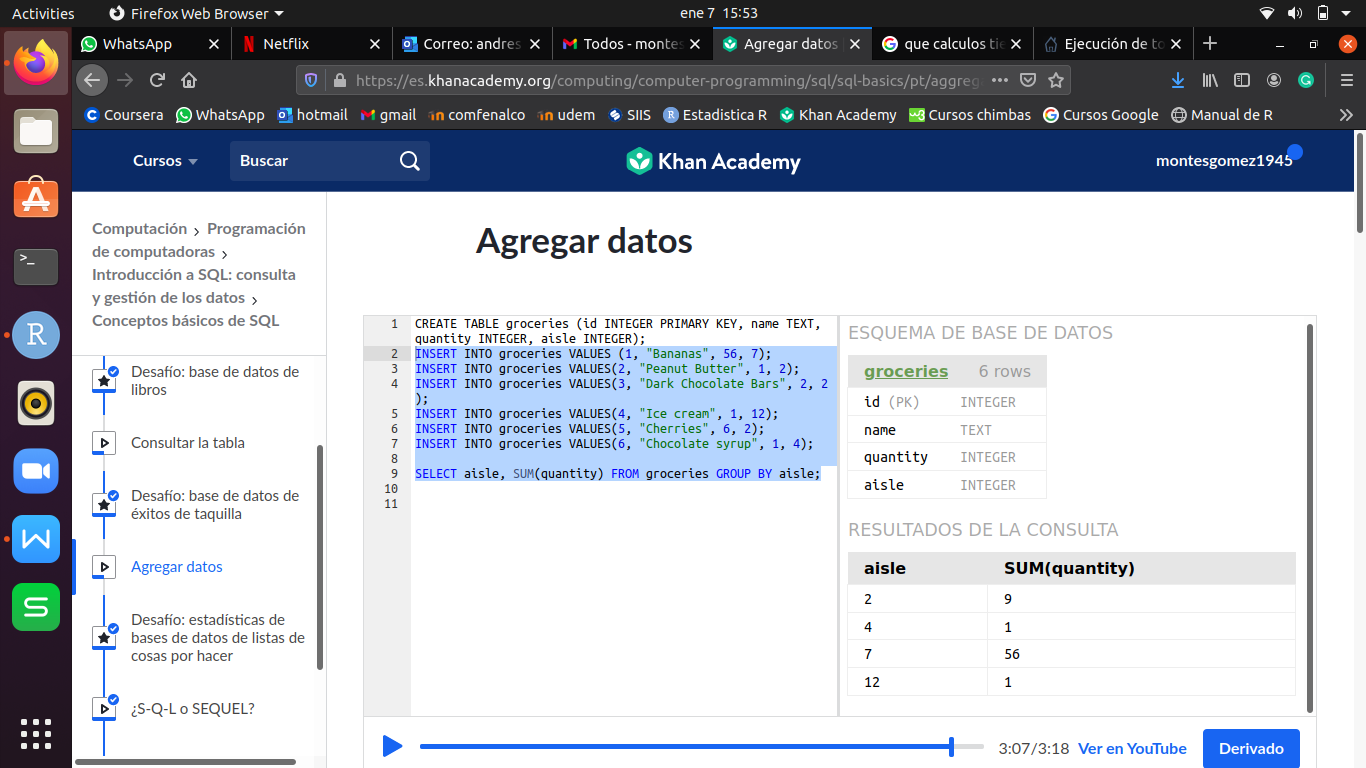


1. **OPERATIONS WITH FILTERS**

SELECT ***col\_filter\_name*** FUNCTION (nom\_col\_num) FROM nom\_table GROUP BY ***col\_filter\_name;***

**EXAMPLE:**

SELECT **aisle**, SUM(quantity) FROM groceries GROUP BY **aisle**;



1. **MULTIPLE FILTERS - (IN, NOT IN)**

SELECT \* FROM nom\_table WHERE col\_num\_name IN (‘texto1’, ‘texto2’, ...);

**EXAMPLE:**

CREATE TABLE exercise\_logs (id INTEGER PRIMARY KEY AUTOINCREMENT, type TEXT, minutes INTEGER, calories INTEGER, heart\_rate INTEGER);

INSERT INTO exercise\_logs(type, minutes, calories, heart\_rate) VALUES ("biking", 30, 100, 110);

INSERT INTO exercise\_logs(type, minutes, calories, heart\_rate) VALUES ("biking", 10, 30, 105);

INSERT INTO exercise\_logs(type, minutes, calories, heart\_rate) VALUES ("dancing", 15, 200, 120);

INSERT INTO exercise\_logs(type, minutes, calories, heart\_rate) VALUES ("tree climbing", 30, 70, 90);

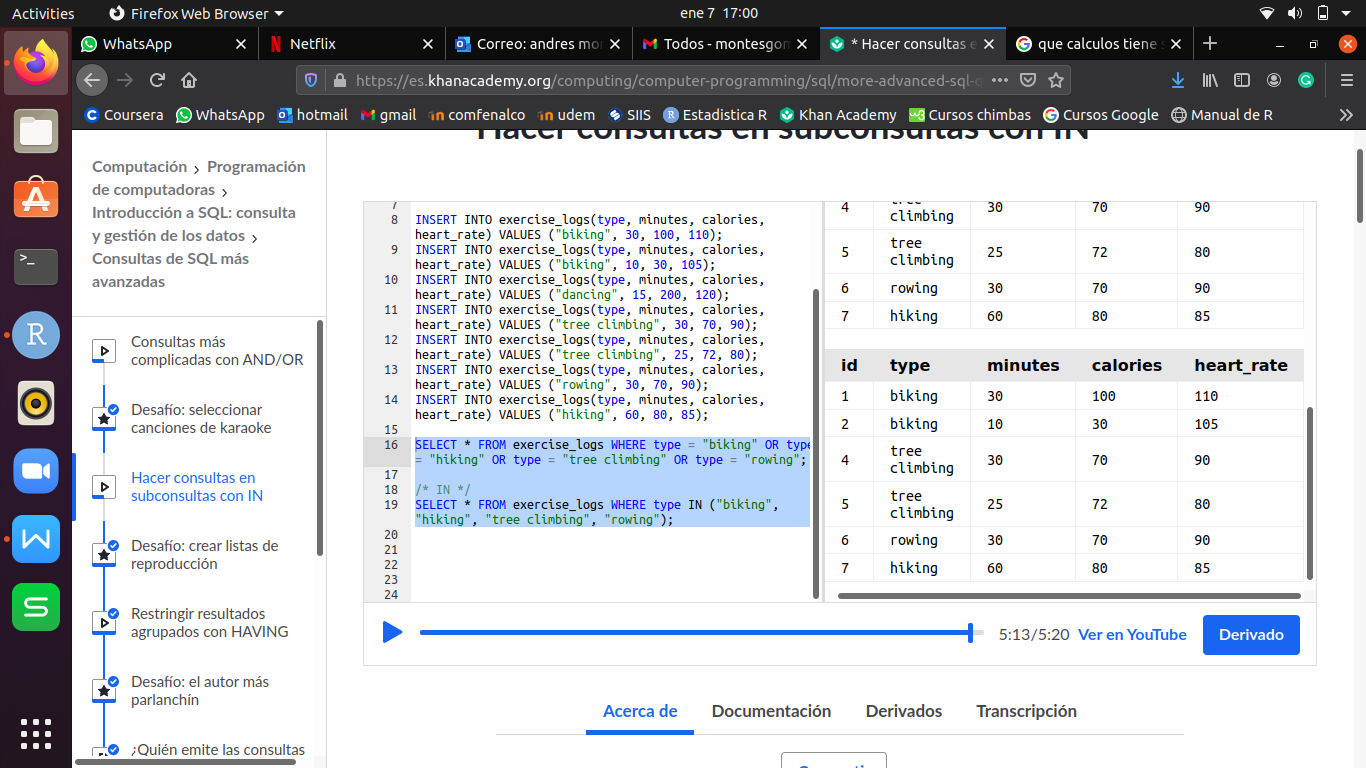
INSERT INTO exercise\_logs(type, minutes, calories, heart\_rate) VALUES ("tree climbing", 25, 72, 80);

INSERT INTO exercise\_logs(type, minutes, calories, heart\_rate) VALUES ("rowing", 30, 70, 90);

INSERT INTO exercise\_logs(type, minutes, calories, heart\_rate) VALUES ("hiking", 60, 80, 85);

SELECT \* FROM exercise\_logs WHERE type = "biking" OR type = "hiking" OR type = "tree climbing" OR type = "rowing";

SELECT \* FROM exercise\_logs WHERE type IN ("biking", "hiking", "tree climbing", "rowing");



- **IN** is used to avoid multiple **OR** and make it easier for the user.

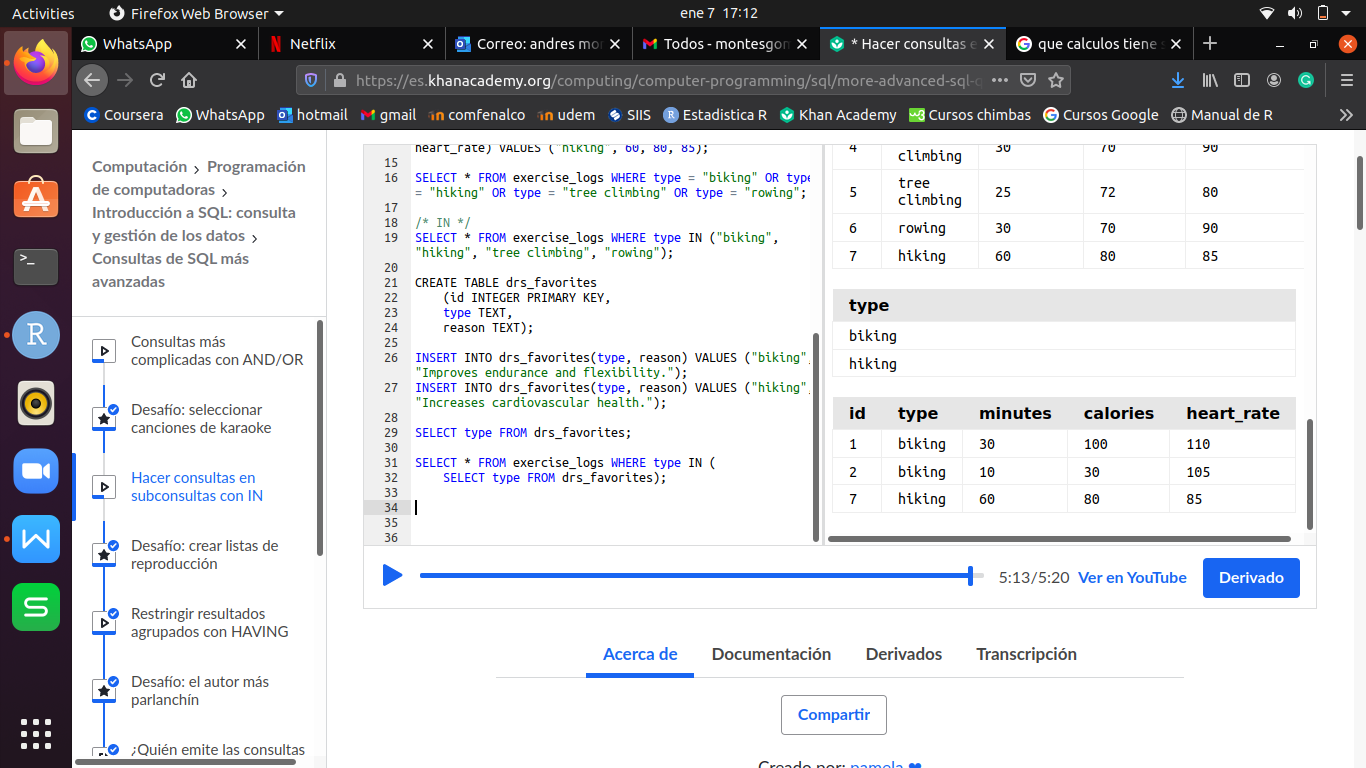
**EXAMPLE:**

CREATE TABLE drs\_favorites (id INTEGER PRIMARY KEY, type TEXT, reason TEXT);

INSERT INTO drs\_favorites(type, reason) VALUES ("biking", "Improves endurance and flexibility.");

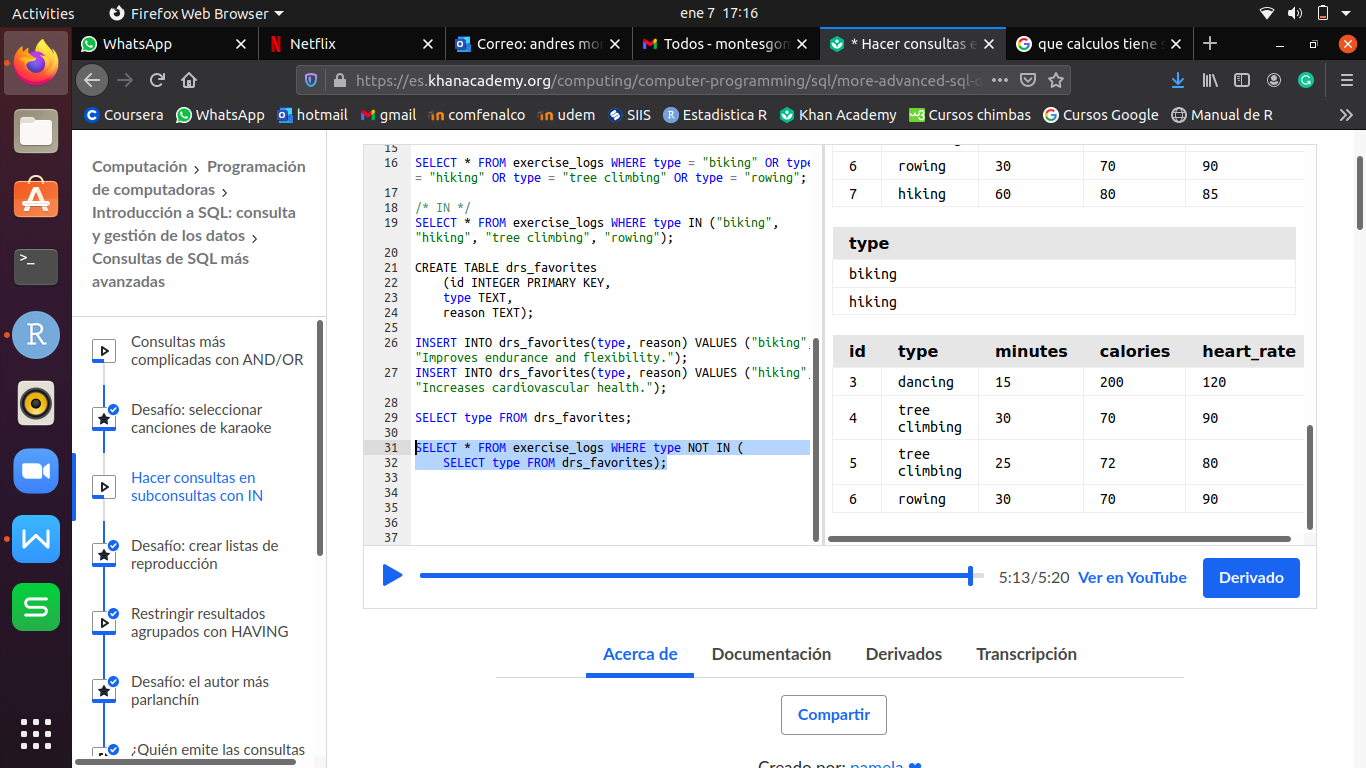
INSERT INTO drs\_favorites(type, reason) VALUES ("hiking", "Increases cardiovascular health.");

SELECT \* FROM exercise\_logs WHERE type IN (SELECT type FROM drs\_favorites);



**EXAMPLE:**

SELECT \* FROM exercise\_logs WHERE type NOT IN (SELECT type FROM drs\_favorites);



1. **QUERIES WITH DESIRED VALUE (LIKE)**

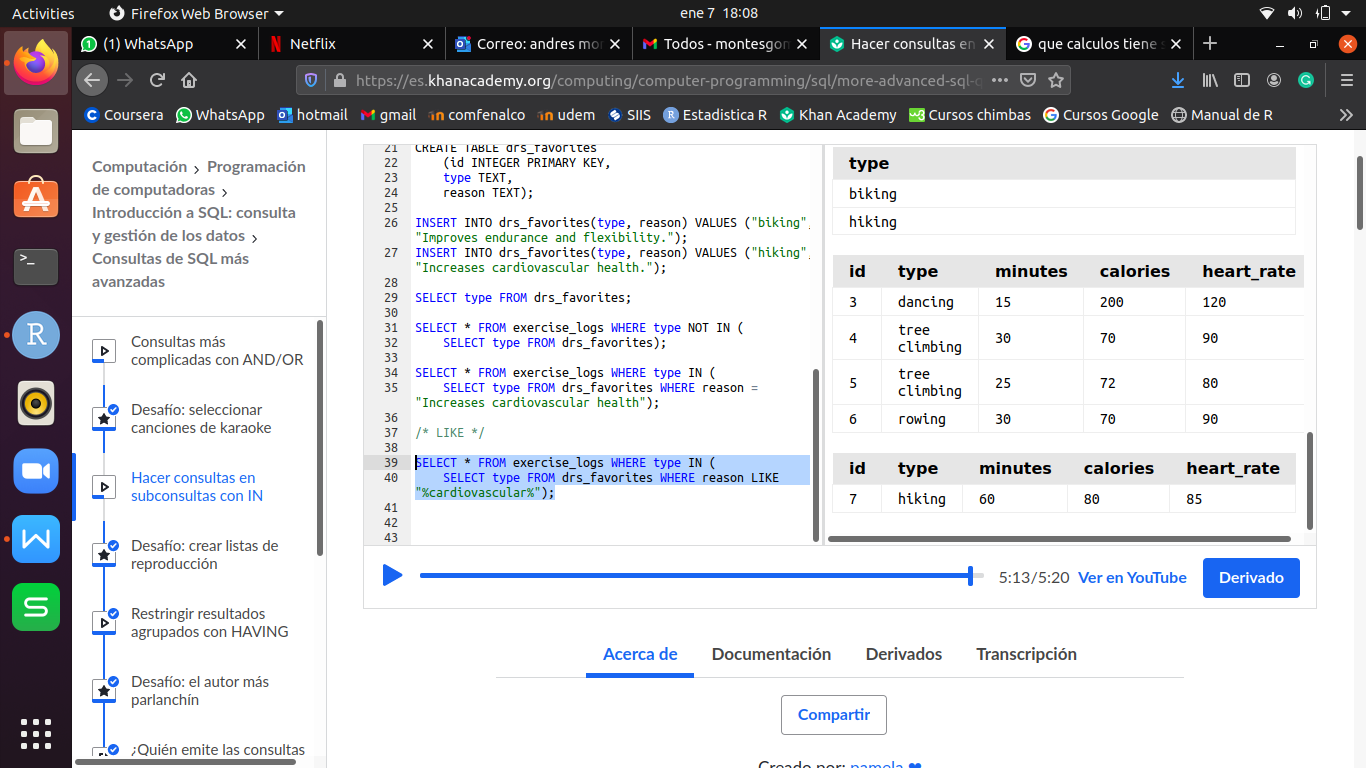
SELECT \* FROM nom\_table1 WHERE nom\_col IN(SELECT nom\_col FROM nom\_table2 WHERE col\_search\_name LIKE ‘%valor\_bucado%);

- The secound part of **SELECT** is an other related table by nom\_col.

- Before **IN** goes the wanted filter.

**EXAMPLE:**

SELECT \* FROM exercise\_logs WHERE type IN (SELECT type FROM drs\_favorites WHERE reason LIKE "%cardiovascular%");



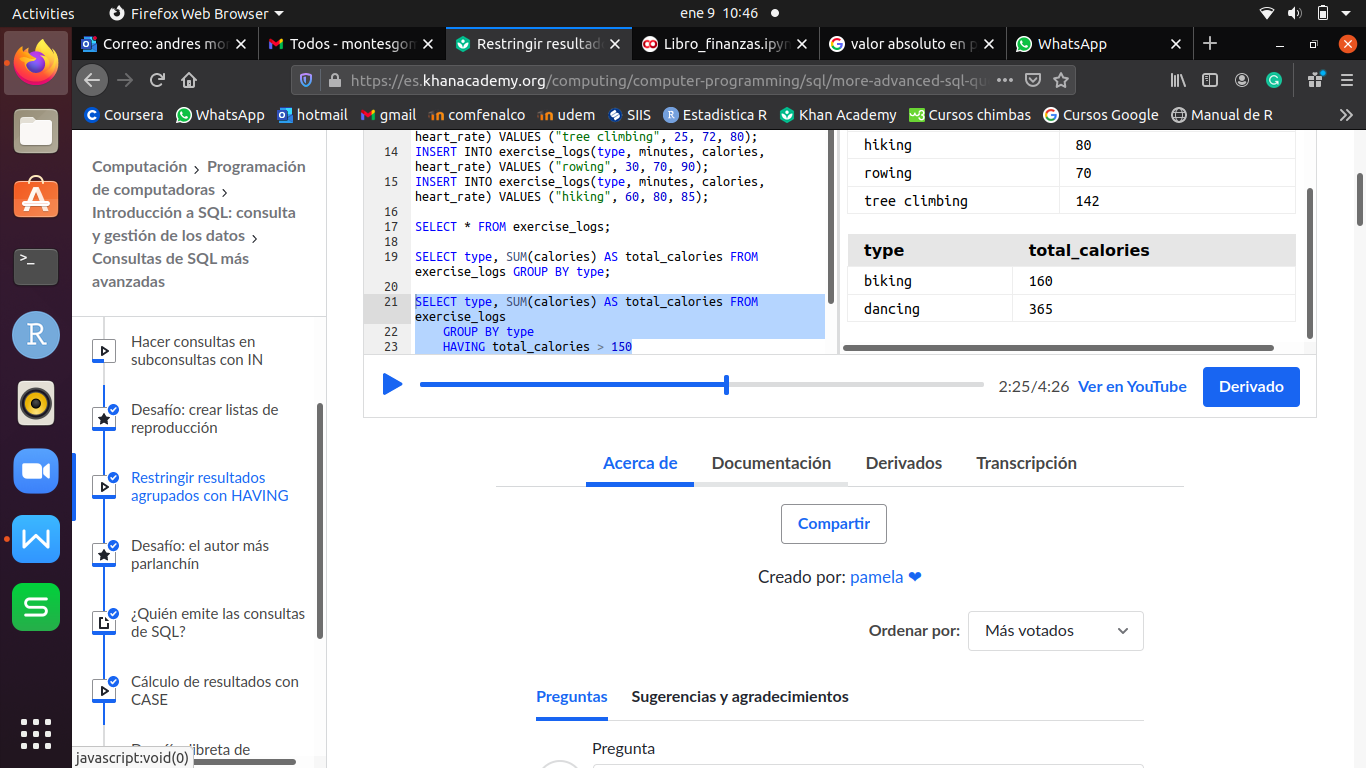
1. **MAKE A FILTER TO AN OPERATION (HAVING)**

SELECT nom\_col FUNCION(nom\_col\_num) AS nom\_col\_resul FROM nom\_tabla GROUP BY nom\_col HAVING nom\_result;

**EXAMPLE:**

SELECT type, SUM(calories) AS total\_calories FROM exercise\_logs

GROUP BY type HAVING total\_calories > 150



1. **CASE**

SELECT OPERATION(\*)

CASE

WHEN name\_col\_num **operador** # or operation THEN ‘text 1’

WHEN name\_col\_num **operador** # or operation THEN ‘text 2’ ...

ELSE ‘text if donesn’t fulfill the conditions’

END AS ‘col\_search\_name’

FROM nom\_table GROUP BY col\_search\_name;

- **CASE** is like a **SWITCH**.

- **CASE** create a new column ‘col\_search\_name’ accordin to given ranges (**WHERE; ELSE**). Wuth this new column is easier to do the searching.

**EXAMPLE:**

SELECT COUNT(\*),

CASE

WHEN heart\_rate > 220-30 THEN "above max"

WHEN heart\_rate > ROUND(0.90 \* (220-30)) THEN "above target"

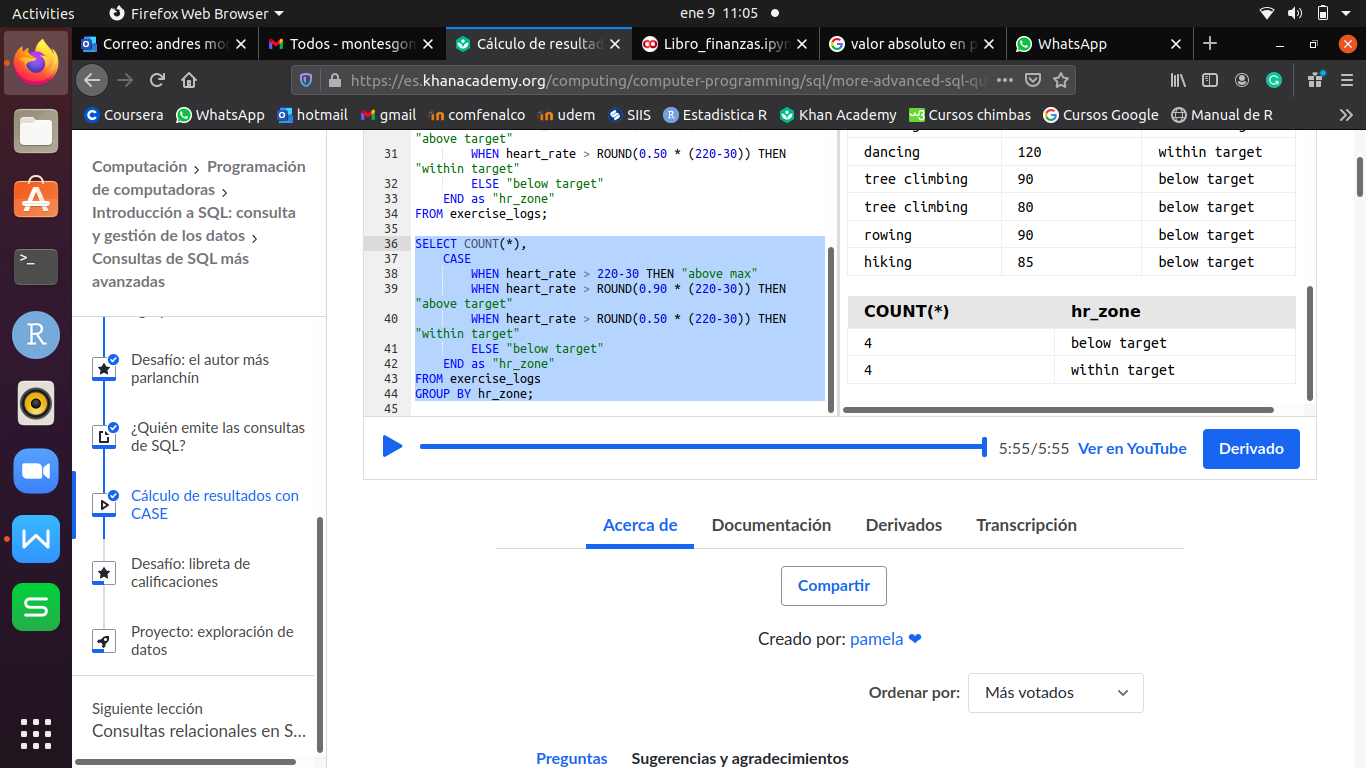
WHEN heart\_rate > ROUND(0.50 \* (220-30)) THEN "within target"

ELSE "below target"

END as "hr\_zone"

FROM exercise\_logs

GROUP BY hr\_zone;



1. **EXPLICIT INNER JOIN**

SELECT nom\_table.nom\_col1, nom\_table.nom\_col2, ... FROM table1

JOIN table2 ON table1.id(PK) = table2.col\_id;

- It is very important input at the beginning of the table (nom\_table) where we are going to take the column (nom\_col), because it is possible that other tables has columns with the same name.

**EXAMPLE:**

CREATE TABLE students (id INTEGER PRIMARY KEY, first\_name TEXT, last\_name TEXT, email TEXT, phone TEXT, birthdate TEXT);

INSERT INTO students (first\_name, last\_name, email, phone, birthdate)

VALUES ("Peter", "Rabbit", "peter@rabbit.com", "555-6666", "2002-06-24");

INSERT INTO students (first\_name, last\_name, email, phone, birthdate)

VALUES ("Alice", "Wonderland", "alice@wonderland.com", "555-4444", "2002-07-04");

CREATE TABLE student\_grades (id INTEGER PRIMARY KEY, student\_id INTEGER, test TEXT, grade INTEGER);

INSERT INTO student\_grades (student\_id, test, grade) VALUES (1, "Nutrition", 95);

INSERT INTO student\_grades (student\_id, test, grade) VALUES (2, "Nutrition", 92);

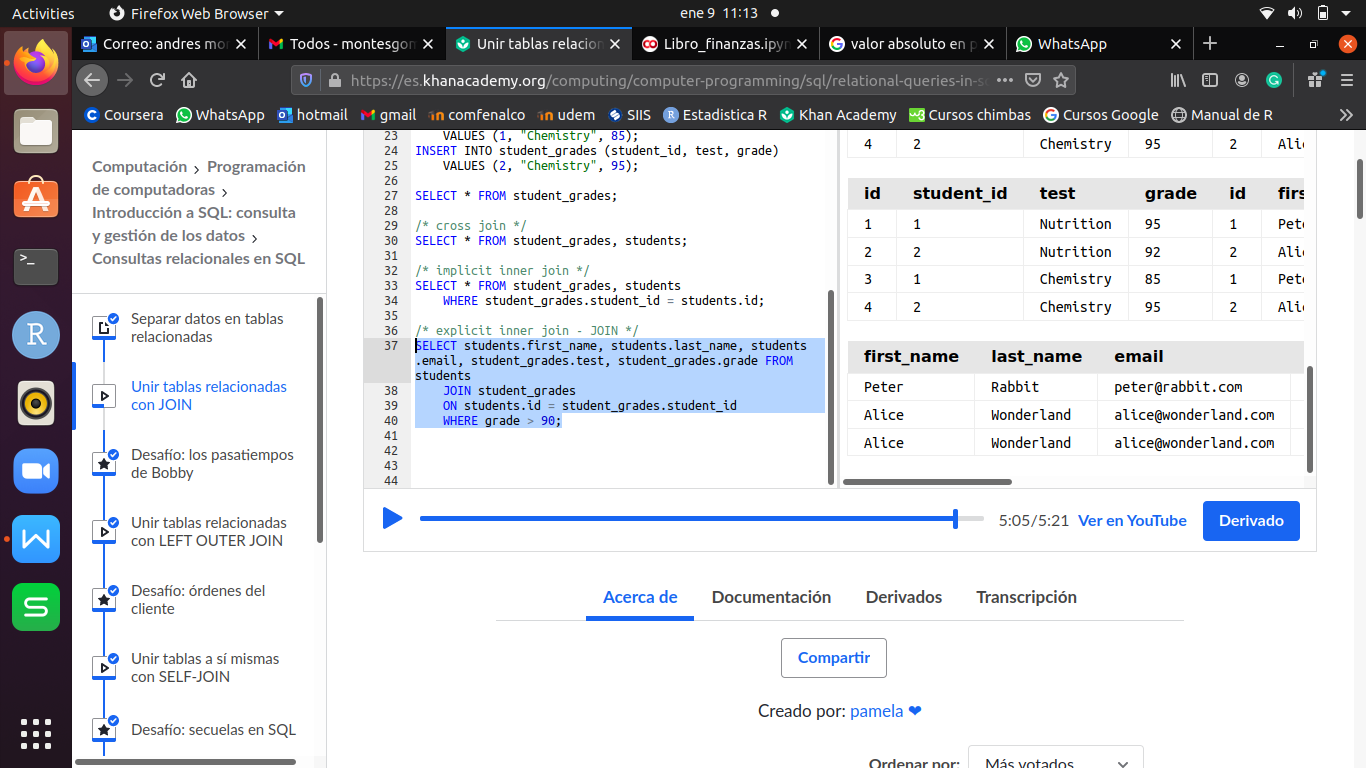
INSERT INTO student\_grades (student\_id, test, grade) VALUES (1, "Chemistry", 85);

INSERT INTO student\_grades (student\_id, test, grade) VALUES (2, "Chemistry", 95);

SELECT students.first\_name, students.last\_name, students.email, student\_grades.test, student\_grades.grade FROM students

JOIN student\_grades ON students.id = student\_grades.student\_id

WHERE grade > 90;



1. **LEFT OUTER JOIN**

SELECT nom\_col1, nom\_col2, ... FROM tabla1 LEFT OUTER JOIN tabla2

ON tabla1.id(PK) = tabla2.col\_id;

- **LEFT OUTER JOIN** is used to when we want to show all the values of the columns selected, inclusive this hasn’t registers in the other table.

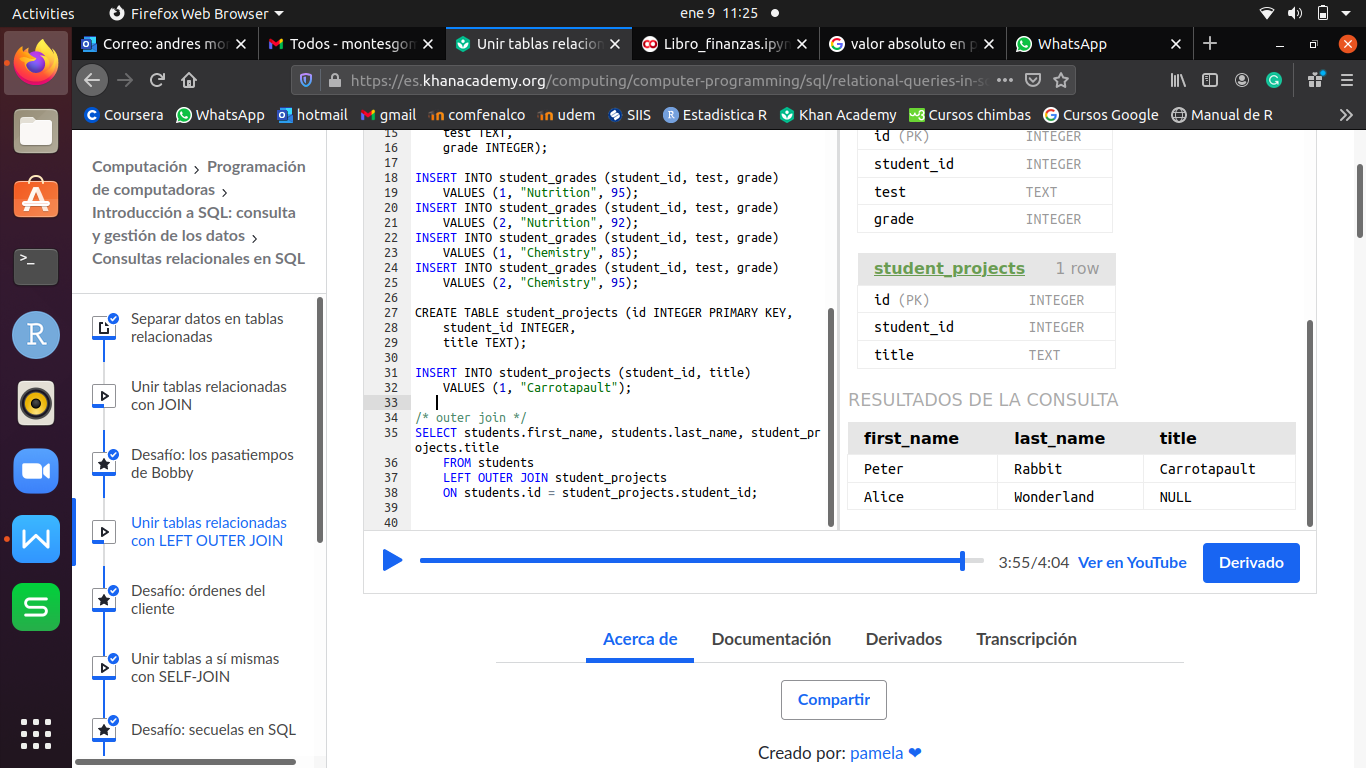
**EXAMPLE:**

CREATE TABLE student\_projects (id INTEGER PRIMARY KEY, student\_id INTEGER, title TEXT);

INSERT INTO student\_projects (student\_id, title) VALUES (1, "Carrotapault");

SELECT students.first\_name, students.last\_name, student\_projects.title FROM students LEFT OUTER JOIN student\_projects

ON students.id = student\_projects.student\_id;



1. **SELF JOIN**

SELECT nom\_col1, nom\_col2, ..., alias.nom\_col AS nom\_com\_wanted FROM table JOIN table alias

ON table.col\_id = alias.id(PK);

- Used to link columns from the same table.

**EXAMPLE:**

CREATE TABLE students (id INTEGER PRIMARY KEY AUTOINCREMENT, first\_name TEXT, last\_name TEXT, email TEXT, phone TEXT, birthdate TEXT,

buddy\_id INTEGER);

INSERT INTO students VALUES (1, "Peter", "Rabbit", "peter@rabbit.com", "555-6666", "2002-06-24", 2);

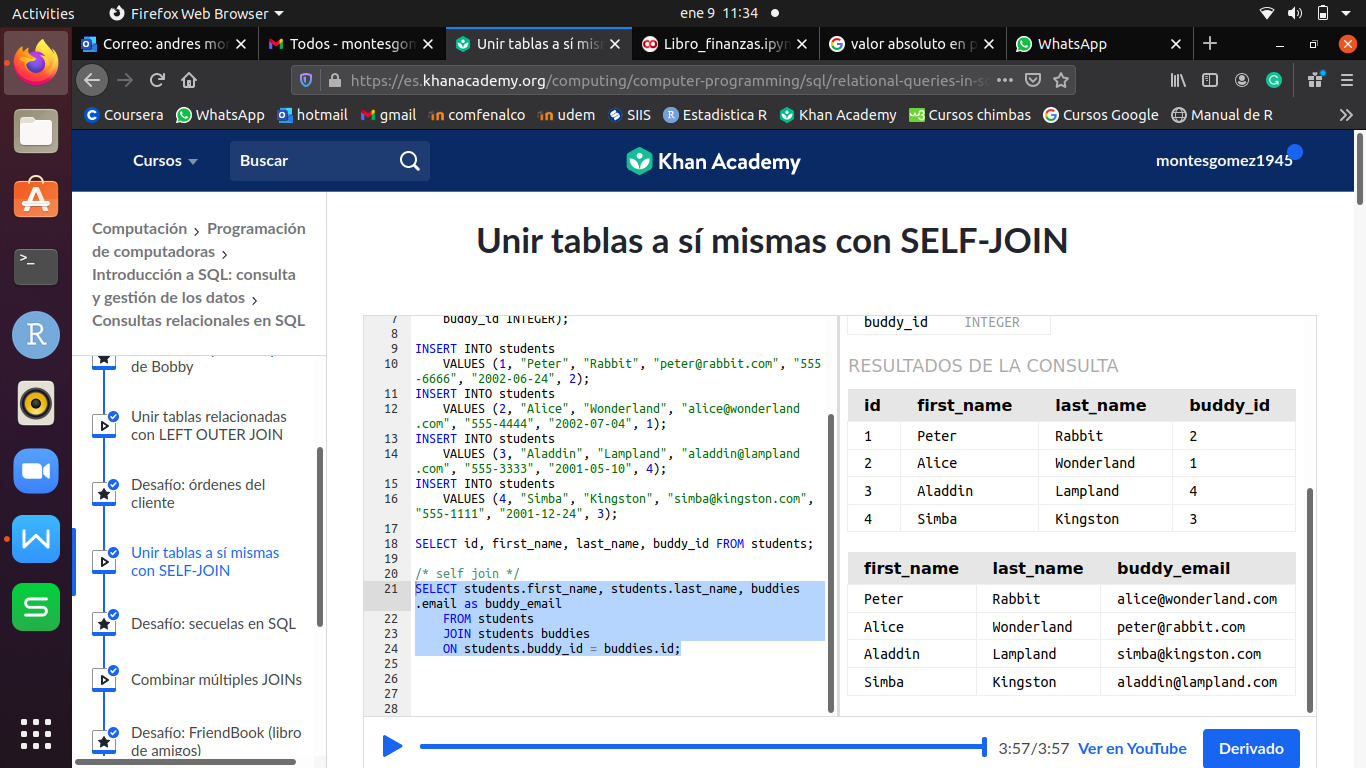
INSERT INTO students VALUES (2, "Alice", "Wonderland", "alice@wonderland.com", "555-4444", "2002-07-04", 1);

INSERT INTO students VALUES (3, "Aladdin", "Lampland", "aladdin@lampland.com", "555-3333", "2001-05-10", 4);

INSERT INTO students VALUES (4, "Simba", "Kingston", "simba@kingston.com", "555-1111", "2001-12-24", 3);

SELECT students.first\_name, students.last\_name, buddies.email as buddy\_email FROM students JOIN students buddies

ON students.buddy\_id = buddies.id;



1. **MULTIPLE JOIN**

SELECT a.nom\_col, b.nom\_col FROM table2

JOIN table1 a ON tabla2.col\_id = a.id(PK)

JOIN table1 b ON tabla2.col\_id = b.id(PK)

**EXAMPLE:**

CREATE TABLE students (id INTEGER PRIMARY KEY, first\_name TEXT, last\_name TEXT, email TEXT, phone TEXT, birthdate TEXT);

INSERT INTO students (first\_name, last\_name, email, phone, birthdate)

VALUES ("Peter", "Rabbit", "peter@rabbit.com", "555-6666", "2002-06-24");

INSERT INTO students (first\_name, last\_name, email, phone, birthdate)

VALUES ("Alice", "Wonderland", "alice@wonderland.com", "555-4444", "2002-07-04");

INSERT INTO students (first\_name, last\_name, email, phone, birthdate)

VALUES ("Aladdin", "Lampland", "aladdin@lampland.com", "555-3333", "2001-05-10");

INSERT INTO students (first\_name, last\_name, email, phone, birthdate)

VALUES ("Simba", "Kingston", "simba@kingston.com", "555-1111", "2001-12-24");

CREATE TABLE student\_projects (id INTEGER PRIMARY KEY, student\_id INTEGER, title TEXT);

INSERT INTO student\_projects (student\_id, title) VALUES (1, "Carrotapault");

INSERT INTO student\_projects (student\_id, title) VALUES (2, "Mad Hattery");

INSERT INTO student\_projects (student\_id, title) VALUES (3, "Carpet Physics");

INSERT INTO student\_projects (student\_id, title) VALUES (4, "Hyena Habitats");

CREATE TABLE project\_pairs (id INTEGER PRIMARY KEY, project1\_id INTEGER, project2\_id INTEGER);

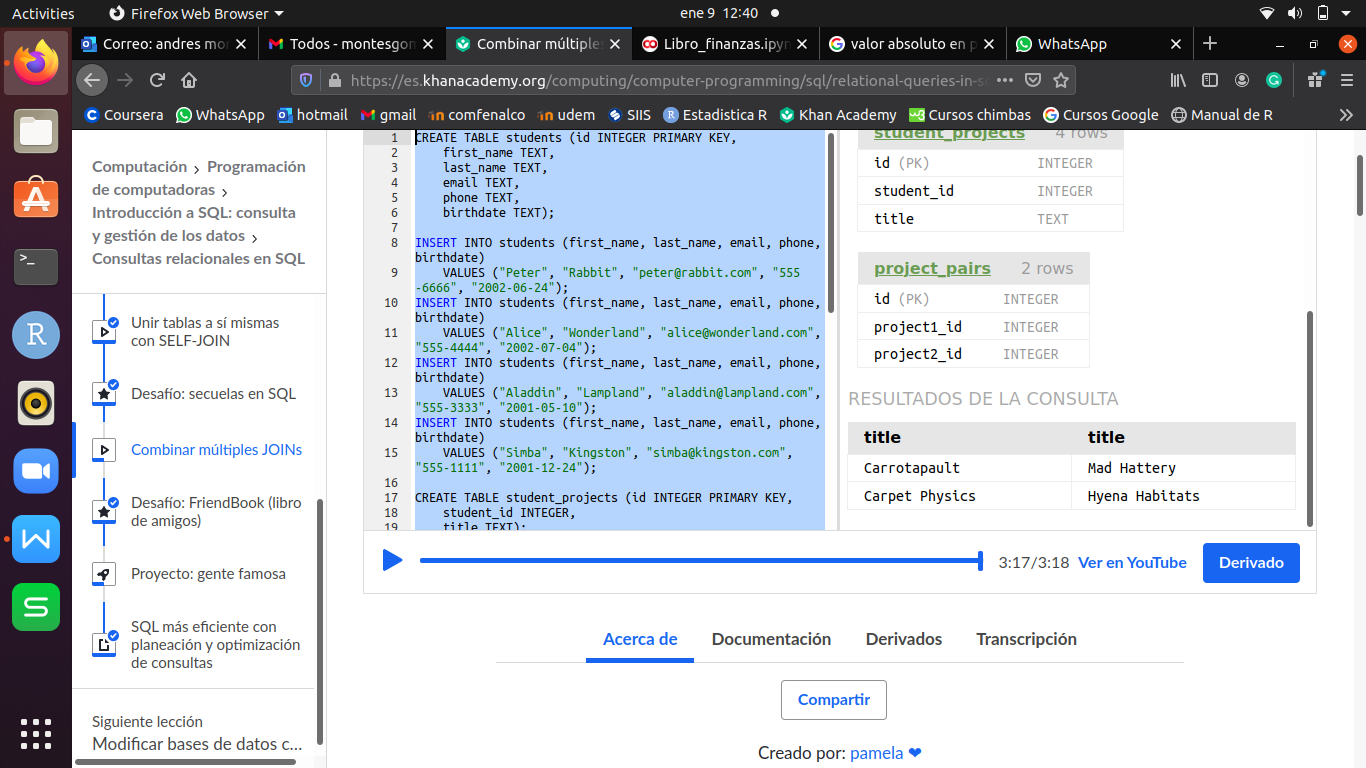
INSERT INTO project\_pairs (project1\_id, project2\_id) VALUES(1, 2);

INSERT INTO project\_pairs (project1\_id, project2\_id) VALUES(3, 4);

SELECT a.title, b.title FROM project\_pairs

JOIN student\_projects a ON project\_pairs.project1\_id = a.id

JOIN student\_projects b ON project\_pairs.project2\_id = b.id;



1. **UPDATE**

UPDATE table SET nom\_col = ‘wath I want to update’ WHERE nom.colx = # ó ‘char’ AND nom\_coly = # ó ‘char’ o simply the id(PK) if I know it.

**EXAMPLE:**

CREATE TABLE users (id INTEGER PRIMARY KEY, name TEXT);

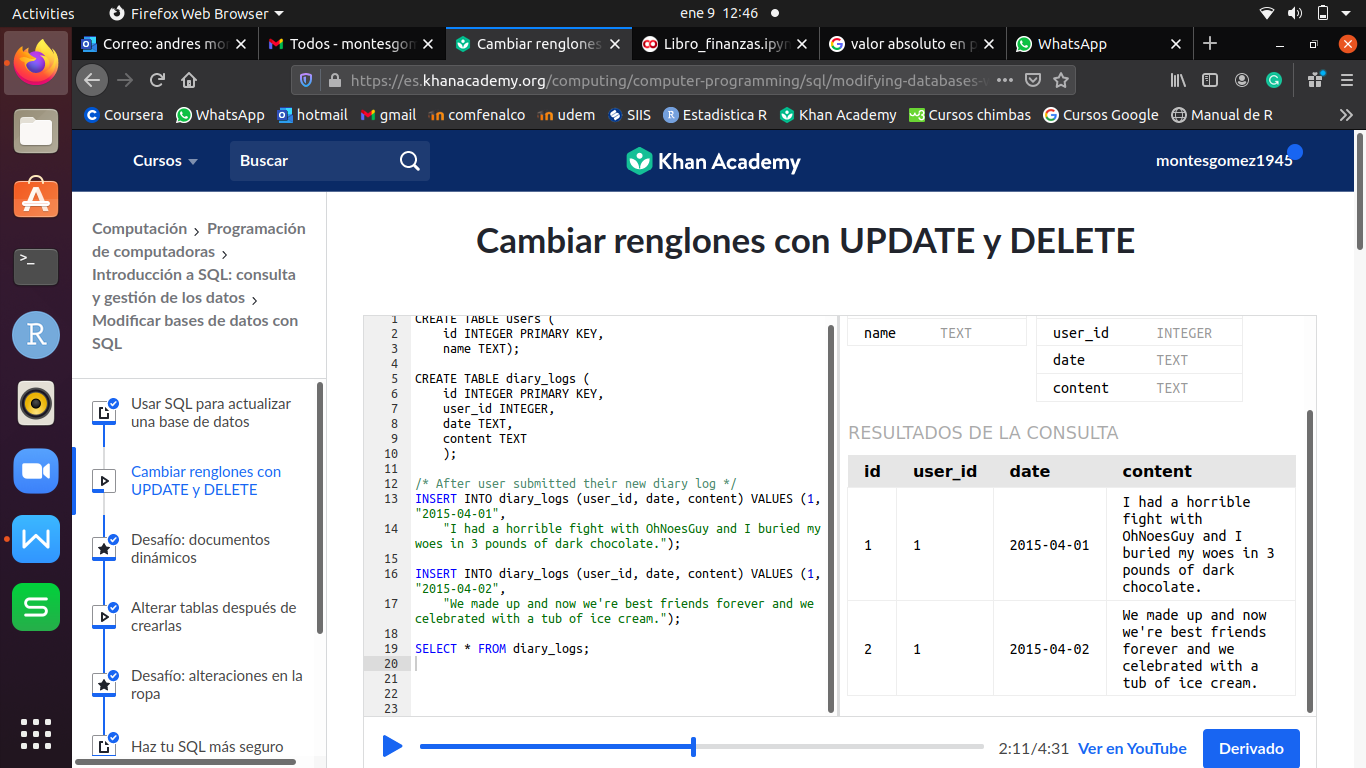
CREATE TABLE diary\_logs (id INTEGER PRIMARY KEY, user\_id INTEGER, date TEXT, content TEXT);

INSERT INTO diary\_logs (user\_id, date, content) VALUES (1, "2015-04-01",

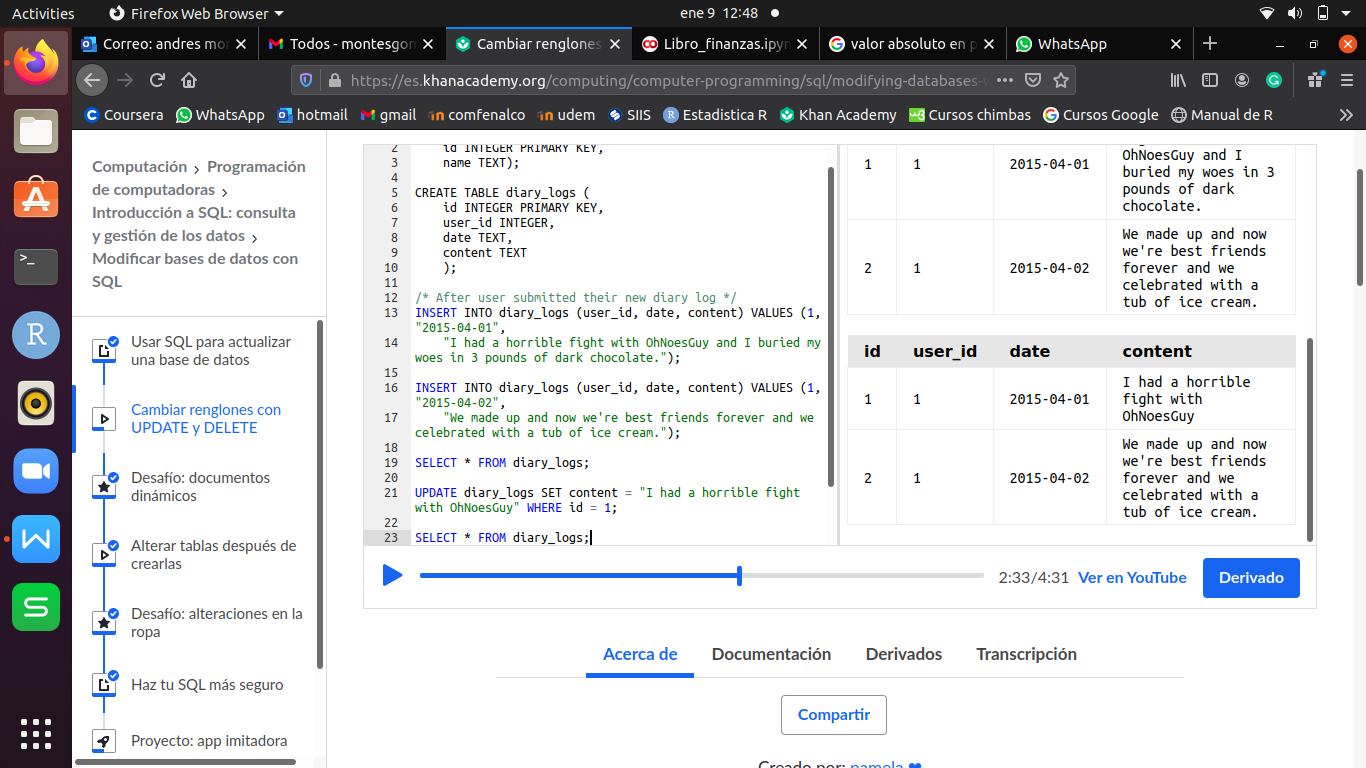
"I had a horrible fight with OhNoesGuy and I buried my woes in 3 pounds of dark chocolate.");

INSERT INTO diary\_logs (user\_id, date, content) VALUES (1, "2015-04-02",

"We made up and now we're best friends forever and we celebrated with a tub of ice cream.");



UPDATE diary\_logs SET content = "I had a horrible fight with OhNoesGuy" WHERE user\_id=1 AND date = "2015-04-01";

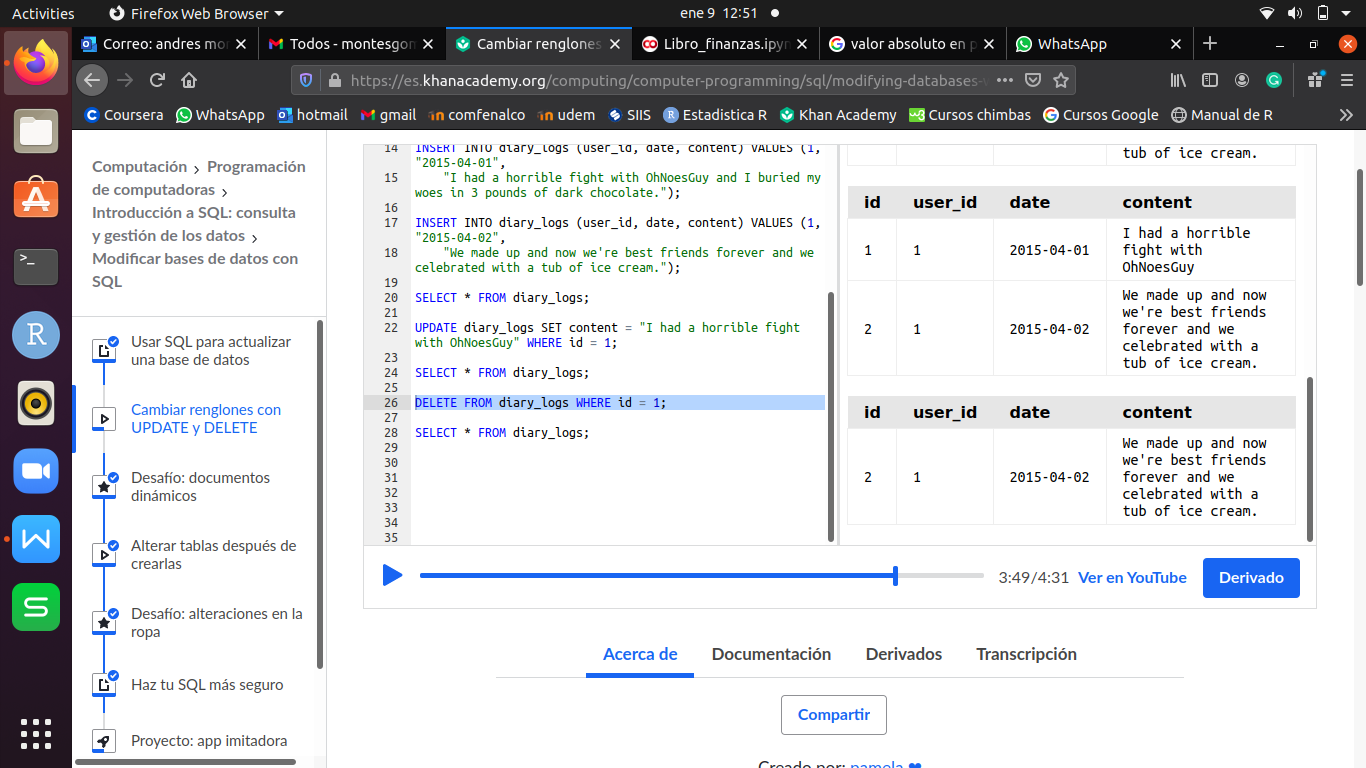


1. **DELETE**

DELETE FROM table WHERE id(PK) = # ó filtrando;

**EXAMPLE:**

DELETE FROM diary\_logs WHERE id = 1;

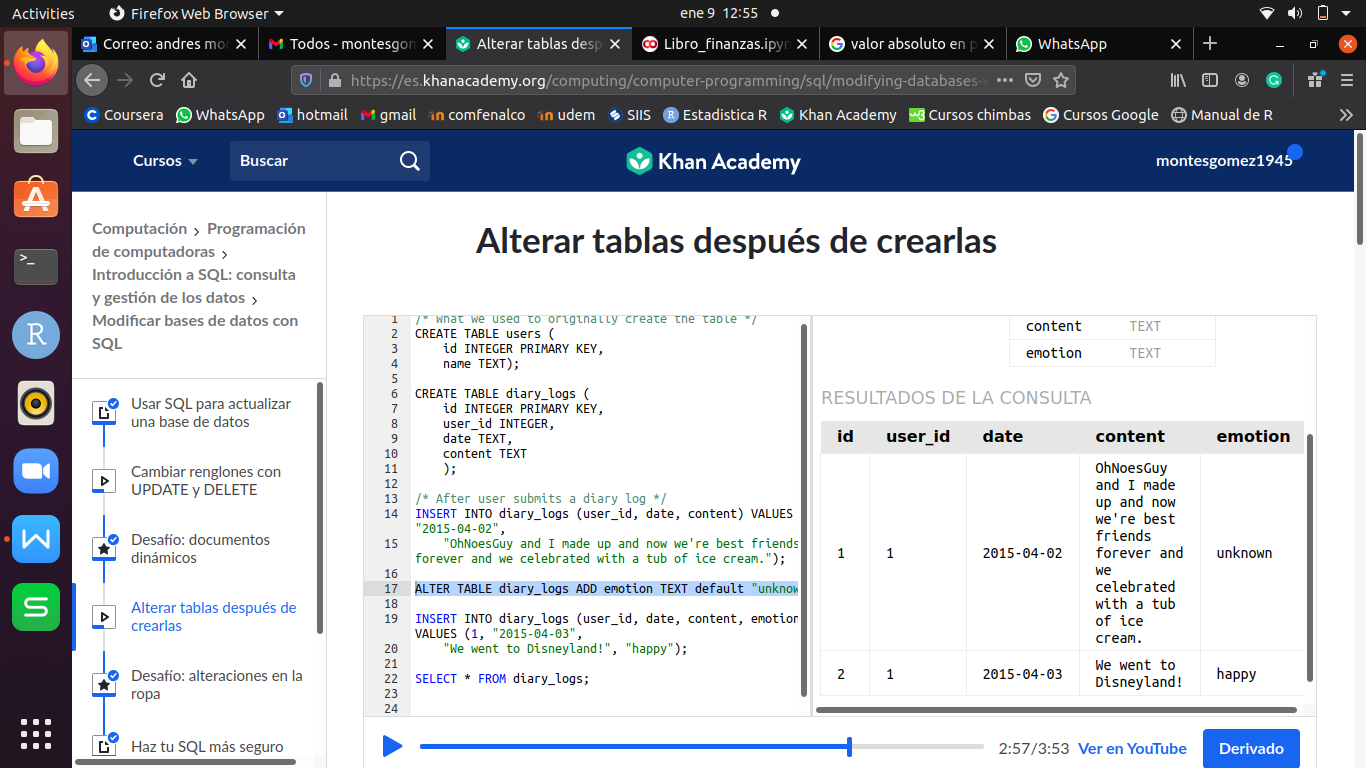


1. **ADD A NEW COLUMN (ALTER TABLE)**

ALTER TABLE tabla ADD nom\_col TIPO default ‘new content’;

**EXAMPLE:**

ALTER TABLE diary\_logs ADD emotion TEXT default "unknown";



1. **DELETE TABLE (DROP TABLE)**

DROP TABLE tabla;