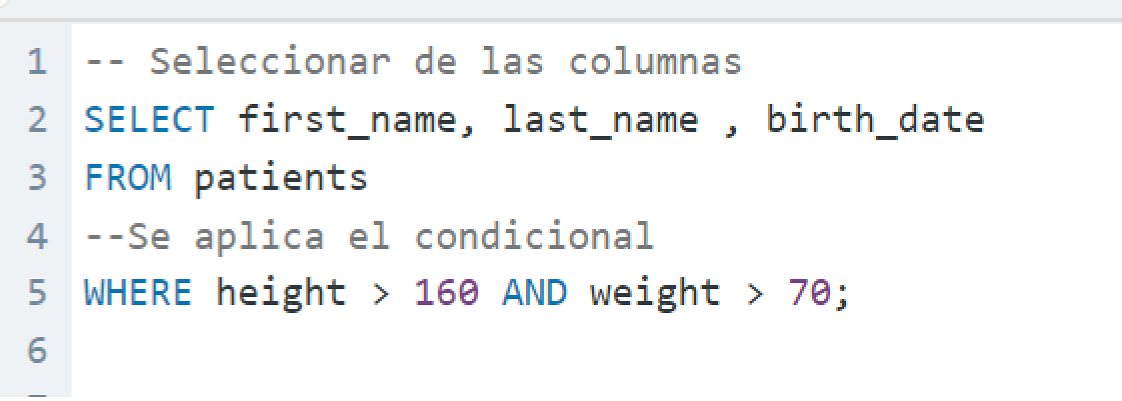
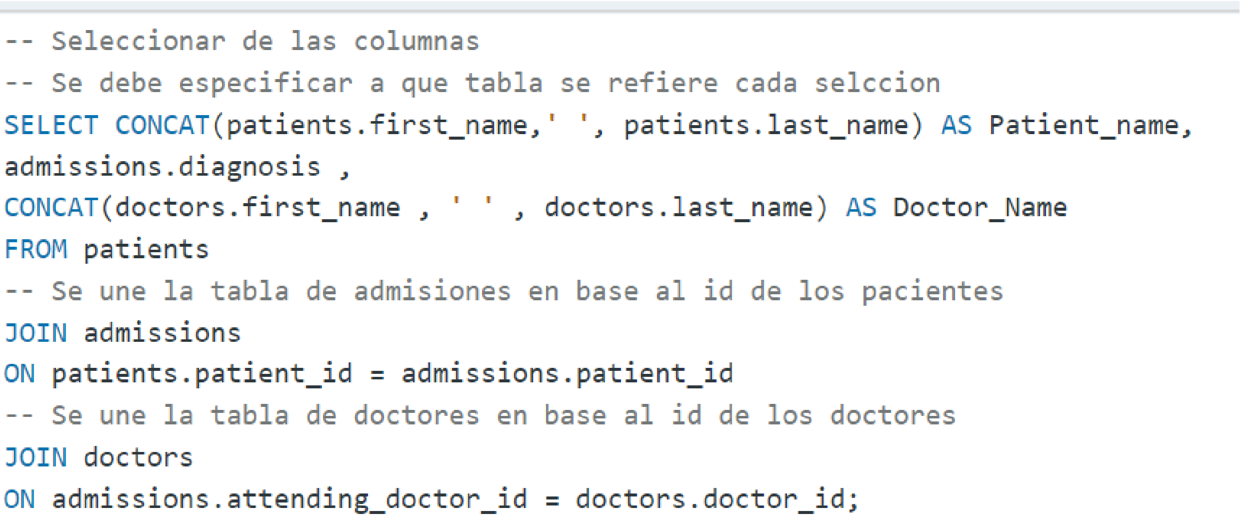
hospital.db:

Write a query to find the first_name, last name and birth date of patients who has height greater than 160 and weight greater than 70



```
-- Seleccionar de las columnas
SELECT first_name, last_name , birth_date
FROM patients
--Se aplica el condicional
WHERE height > 160 AND weight > 70;
```

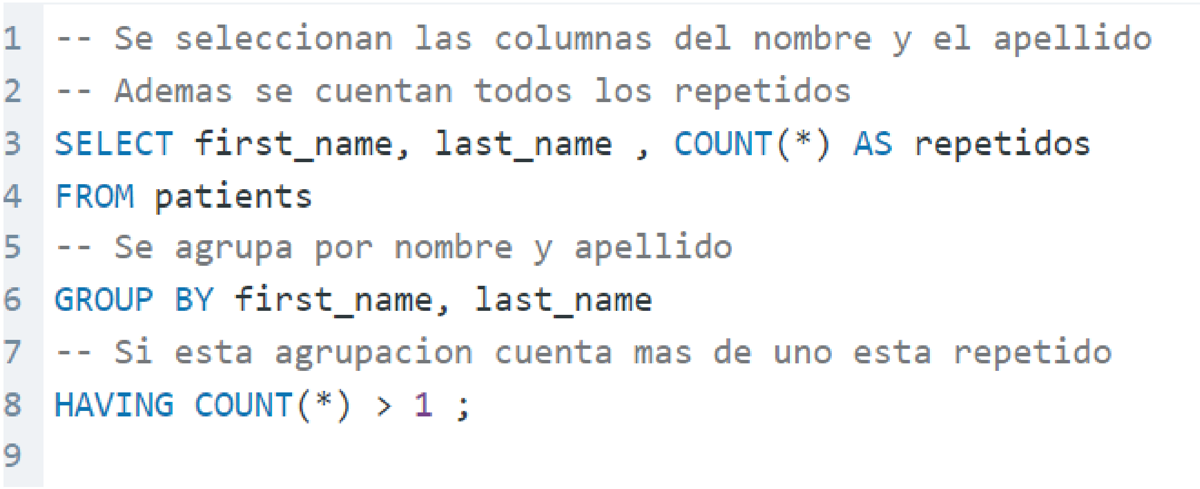
For every admission, display the patient's full name, their admission diagnosis, and their doctor's full name who diagnosed their problem.



```
-- Seleccionar de las columnas
-- Se debe especificar a que tabla se refiere cada selccion
SELECT CONCAT(patients.first_name,' ', patients.last_name) AS Patient_name,
admissions.diagnosis ,
CONCAT(doctors.first_name , ' ' , doctors.last_name) AS Doctor_Name
FROM patients
-- Se une la tabla de admisiones en base al id de los pacientes
JOIN admissions
ON patients.patient_id = admissions.patient_id
-- Se une la tabla de doctores en base al id de los doctores
JOIN doctors
ON admissions.attending_doctor_id = doctors.doctor_id;
```

Display the first name, last name and number of duplicate patients based on their first name and last name.

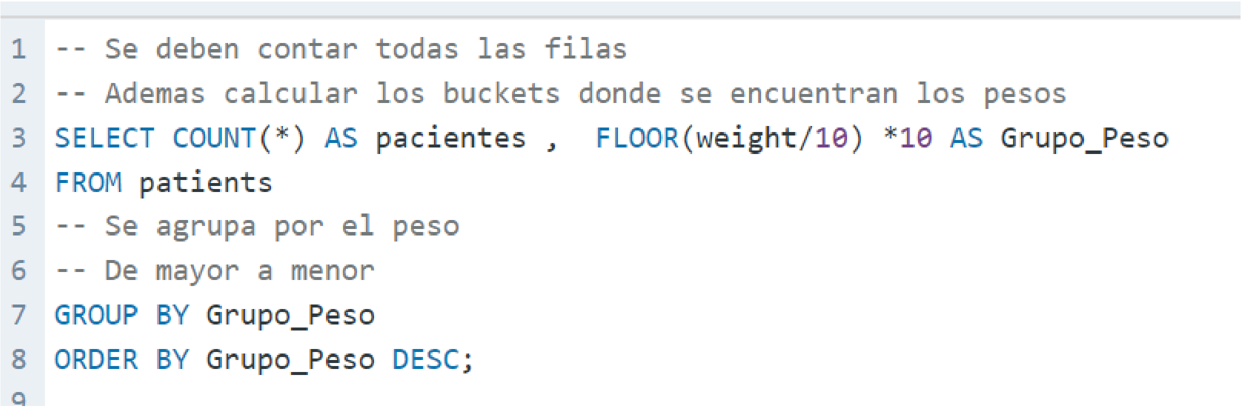
Ex: A patient with an identical name can be considered a duplicate.



```sql
-- Se seleccionan las columnas del nombre y el apellido
-- Ademas se cuentan todos los repetidos
SELECT first_name, last_name , COUNT(*) AS repetidos
FROM patients
-- Se agrupa por nombre y apellido
GROUP BY first_name, last_name
-- Si esta agrupacion cuenta mas de uno esta repetido
HAVING COUNT(*) > 1 ;
```

Show all of the patients grouped into weight groups.
Show the total amount of patients in each weight group.
Order the list by the weight group decending.

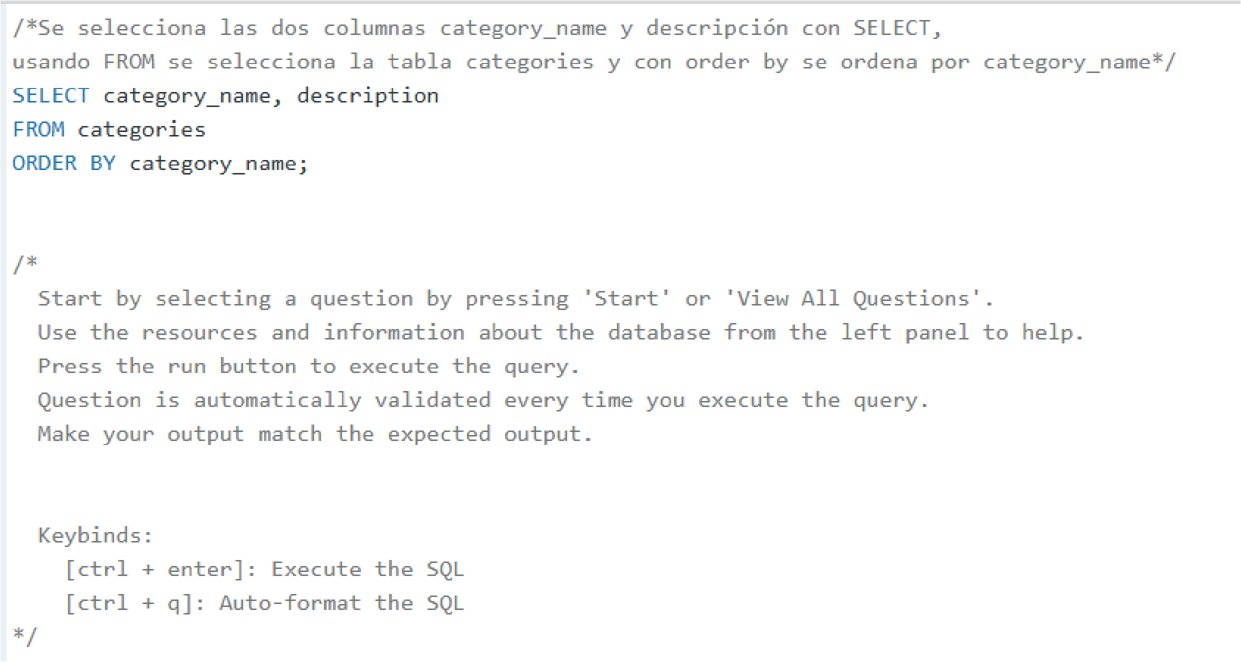
For example, if they weight 100 to 109 they are placed in the 100 weight group, 110-119 = 110 weight group, etc.



```sql
-- Se deben contar todas las filas
-- Ademas calcular los buckets donde se encuentran los pesos
SELECT COUNT(*) AS pacientes ,  FLOOR(weight/10) *10 AS Grupo_Peso
FROM patients
-- Se agrupa por el peso
-- De mayor a menor
GROUP BY Grupo_Peso
ORDER BY Grupo_Peso DESC;
```

<u>northwind.db</u>

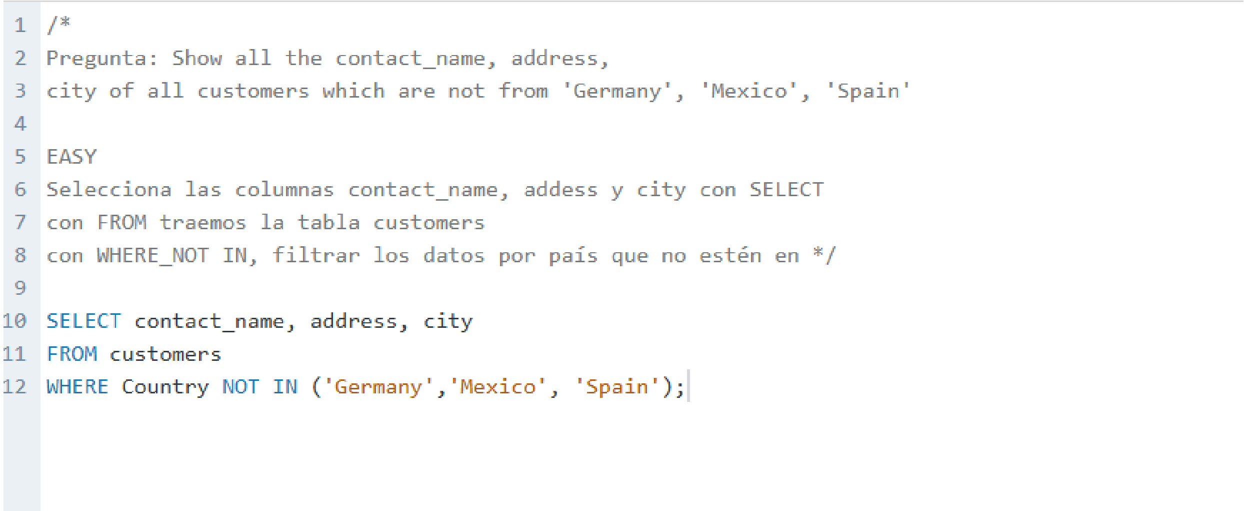
Show the category_name and description from the categories table sorted by category_name.



```
/*Se selecciona las dos columnas category_name y descripción con SELECT,
usando FROM se selecciona la tabla categories y con order by se ordena por category_name*/
SELECT category_name, description
FROM categories
ORDER BY category_name;


/*
  Start by selecting a question by pressing 'Start' or 'View All Questions'.
  Use the resources and information about the database from the left panel to help.
  Press the run button to execute the query.
  Question is automatically validated every time you execute the query.
  Make your output match the expected output.


  Keybinds:
    [ctrl + enter]: Execute the SQL
    [ctrl + q]: Auto-format the SQL
*/
```

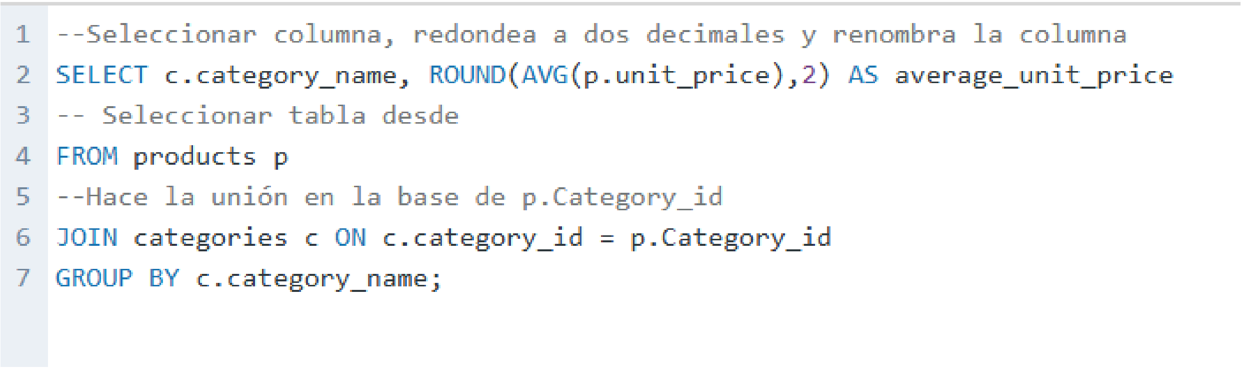


```
/*
Pregunta: Show all the contact_name, address,
city of all customers which are not from 'Germany', 'Mexico', 'Spain'

EASY
Selecciona las columnas contact_name, addess y city con SELECT
con FROM traemos la tabla customers
con WHERE_NOT IN, filtrar los datos por país que no estén en */

SELECT contact_name, address, city
FROM customers
WHERE Country NOT IN ('Germany','Mexico', 'Spain');
```

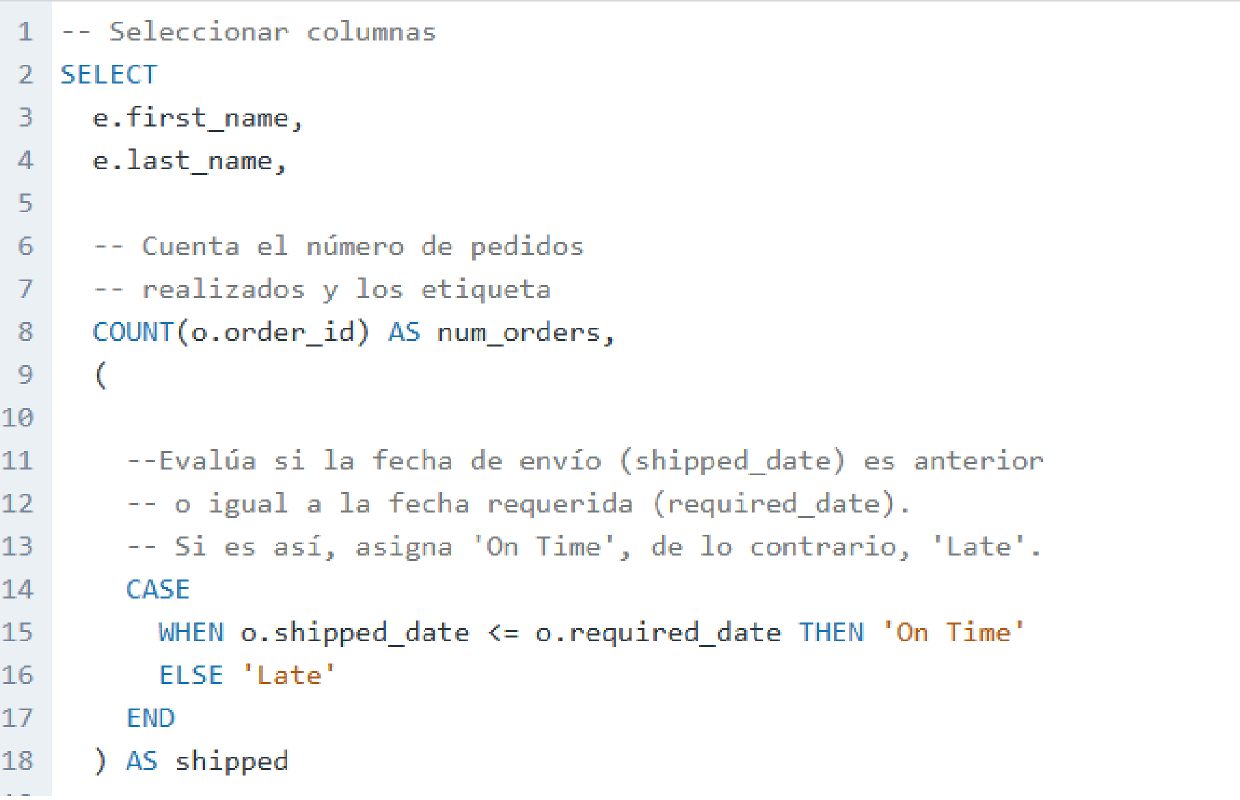
Show the category_name and the average product unit price for each category rounded to 2 decimal places.



```sql
--Seleccionar columna, redondea a dos decimales y renombra la columna
SELECT c.category_name, ROUND(AVG(p.unit_price),2) AS average_unit_price
-- Seleccionar tabla desde
FROM products p
--Hace la unión en la base de p.Category_id
JOIN categories c ON c.category_id = p.Category_id
GROUP BY c.category_name;
```

Show the employee's first_name and last_name, a "num_orders" column with a count of the orders taken, and a column called "Shipped" that displays "On Time" if the order shipped_date is less or equal to the required_date, "Late" if the order shipped late.

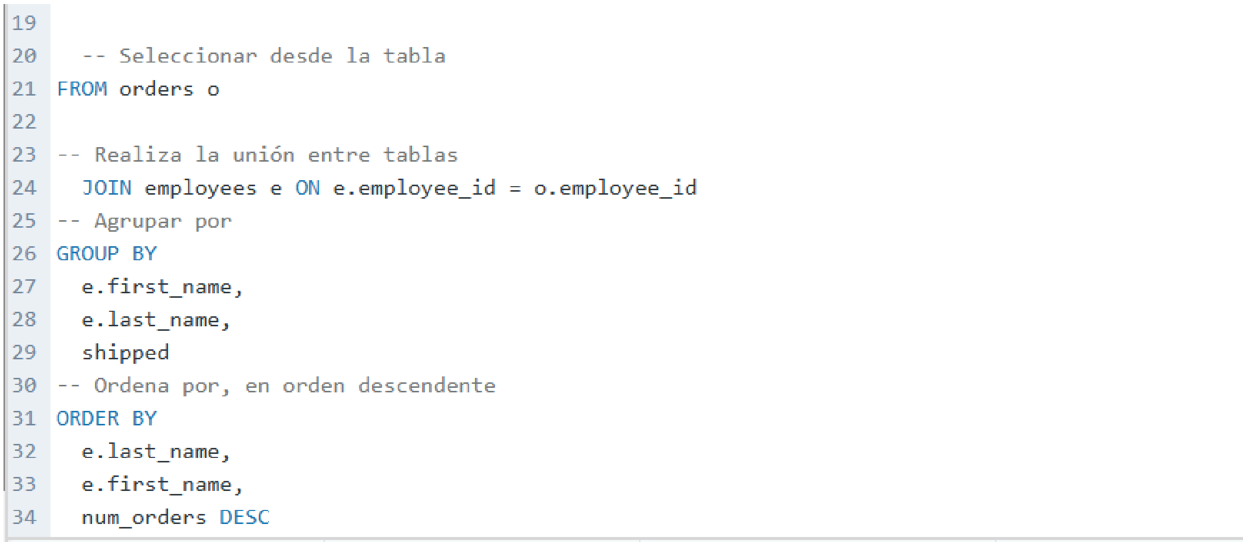
Order by employee last_name, then by first_name, and then descending by number of orders.



```sql
-- Seleccionar columnas
SELECT
  e.first_name,
  e.last_name,

  -- Cuenta el número de pedidos
  -- realizados y los etiqueta
  COUNT(o.order_id) AS num_orders,
  (

    --Evalúa si la fecha de envío (shipped_date) es anterior
    -- o igual a la fecha requerida (required_date).
    -- Si es así, asigna 'On Time', de lo contrario, 'Late'.
    CASE
      WHEN o.shipped_date <= o.required_date THEN 'On Time'
      ELSE 'Late'
    END
  ) AS shipped
```



```sql

  -- Seleccionar desde la tabla
FROM orders o

-- Realiza la unión entre tablas
  JOIN employees e ON e.employee_id = o.employee_id
-- Agrupar por
GROUP BY
  e.first_name,
  e.last_name,
  shipped
-- Ordena por, en orden descendente
ORDER BY
  e.last_name,
  e.first_name,
  num_orders DESC
```