```
00.
CREATE DATABASE geography_db;

Download and import download the 03-Exercises-Built-in-Functions-geography_db.sql

01.
CREATE VIEW view_river_info
AS
SELECT CONCAT(
      'The river',
      ' ',
      river_name,
      ' ',
      'flows into the',
      ' ',
      outflow,
      ' ',
      'and is',
      ' ',
      "length",
      ' ',
      'kilometers long.'
)
AS "River Information"
FROM rivers
ORDER BY river_name ASC;

SELECT * FROM view_river_info;

02.
CREATE VIEW view_continents_countries_currencies_details
AS
SELECT
      CONCAT(continents.continent_name, ': ', continents.continent_code) AS
"Continent Details",
      CONCAT(countries.country_name, ' - ', countries.capital, ' - ',
countries.area_in_sq_km, ' - ', 'km2') AS "Country Information",
      CONCAT(currencies.description, ' (', currencies.currency_code, ')') AS
"Currencies"
FROM continents
JOIN countries ON continents.continent_code = countries.continent_code
JOIN currencies ON countries.currency_code = currencies.currency_code
ORDER BY
"Country Information" ASC,
"Currencies" ASC;

03.
ALTER TABLE countries
ADD COLUMN capital_code CHAR(2);

UPDATE countries
SET capital_code = SUBSTRING(capital, 1, 2);

04.
UPDATE currencies
SET description = SUBSTRING(description, 5, LENGTH(description));

SELECT description AS "substring"
FROM currencies;
```

OR BELOW:

```
SELECT
      SUBSTRING(description, 5, LENGTH(description)) AS "substring"
FROM currencies;

05.
SELECT
SUBSTRING("River Information" FROM '([0-9]{1,4})')
AS
"river_length"
FROM view_river_info;

OR

SELECT
    (REGEXP_MATCHES("River Information", '([0-9]{1,4})'))[1] AS river_length
FROM
    view_river_info;

06.
SELECT
      REPLACE(mountain_range, 'a', '@') AS "replace_a",
      REPLACE (mountain_range, 'A', '$') AS "replace_A"
FROM mountains;

07.
SELECT capital,
      TRANSLATE(capital, '', 'aaaceinou') AS "translated_name"
FROM countries;

08.
SELECT
      continent_name,
      TRIM(LEADING FROM continent_name)
FROM continents;

OR

SELECT
      continent_name,
      LTRIM(continent_name) AS trim
FROM continents

09.
SELECT continent_name,
      TRIM(TRAILING FROM continent_name)
FROM continents;

10.
SELECT TRIM(LEADING 'M' FROM peak_name) AS "Left Trim",
TRIM(TRAILING 'm' FROM peak_name) AS "Right Trim"
FROM peaks;

11.
SELECT
      CONCAT(mountains.mountain_range, ' ', peaks.peak_name) AS "Mountain
Information",
```

```
      LENGTH(CONCAT(mountains.mountain_range, ' ', peaks.peak_name)) AS "Characters
Length",
      BIT_LENGTH(CONCAT(mountains.mountain_range, ' ', peaks.peak_name)) AS "Bits
of a String"
FROM mountains
JOIN peaks ON mountains.id = peaks.mountain_id;

12.
SELECT
      CAST(population AS VARCHAR),
      LENGTH(CAST(population AS VARCHAR)) AS length
FROM countries

13.
SELECT peak_name,
      LEFT(peak_name, 4) AS "Positive Left",
      LEFT(peak_name, -4) AS "Negative Left"
FROM peaks;

14.
SELECT peak_name,
RIGHT(peak_name, 4) AS "Positive Right",
RIGHT(peak_name, -4) AS "Negative Right"
FROM peaks;

15.
UPDATE countries
      SET iso_code = UPPER(LEFT(country_name, 3))
WHERE iso_code IS NULL;

16.
UPDATE countries
SET
country_code = LOWER(REVERSE(country_code));

17.
SELECT
CONCAT(elevation, ' ', REPEAT('-', 3), REPEAT('>', 2), ' ', peak_name) AS
"Elevation -->> Peak Name"
FROM peaks
WHERE elevation >= 4884;

18.
CREATE TABLE bookings_calculation
AS SELECT
      bookings.booked_for
FROM bookings
WHERE apartment_id = 93;

ALTER TABLE bookings_calculation
ADD COLUMN multiplication NUMERIC,
ADD COLUMN modulo NUMERIC;

UPDATE bookings_calculation
SET multiplication = booked_for * 50,
      modulo = booked_for % 50;

19.
SELECT
```

```
      latitude,
      ROUND(latitude, 2) AS round,
      TRUNC(latitude, 2) AS trunc
FROM apartments

20.
SELECT
      longitude,
      ABS(longitude)
FROM apartments

21.
ALTER TABLE bookings
ADD COLUMN billing_day TIMESTAMPTZ DEFAULT CURRENT_TIMESTAMP;

SELECT
         TO_CHAR(billing_day, 'DD "Day" MM "Month" YYYY "Year" HH24:MI:SS')
         AS "Billing Day"
FROM bookings;

22.
SELECT
         EXTRACT(YEAR FROM booked_at) AS "YEAR",
         EXTRACT(MONTH FROM booked_at) AS "MONTH",
         EXTRACT(DAY FROM booked_at) AS "MONTH",
         EXTRACT(HOUR FROM booked_at) AS "HOUR",
         EXTRACT(MINUTE FROM booked_at) AS "MINUTE",
         CEIL(EXTRACT(SECOND FROM booked_at)) AS "SECOND"
FROM bookings;

23.
SELECT user_id,
       AGE(starts_at, booked_at) AS "Early Birds"
FROM bookings
WHERE AGE(starts_at, booked_at) > INTERVAL '10 months';

24.
SELECT
companion_full_name,
email
FROM users
WHERE LOWER(companion_full_name) LIKE '%and%'
AND email NOT LIKE '%@gmail';

25.
SELECT
LEFT(first_name, 2) AS "initials",
COUNT('initials') AS "user_count"
FROM users
GROUP BY initials
ORDER BY
user_count DESC,
initials ASC;

26.
SELECT SUM(booked_for) AS "total_value"
FROM bookings
WHERE apartment_id = 90;
```

27.

```
SELECT AVG(multiplication) AS "average_value"
FROM bookings_calculation;
```