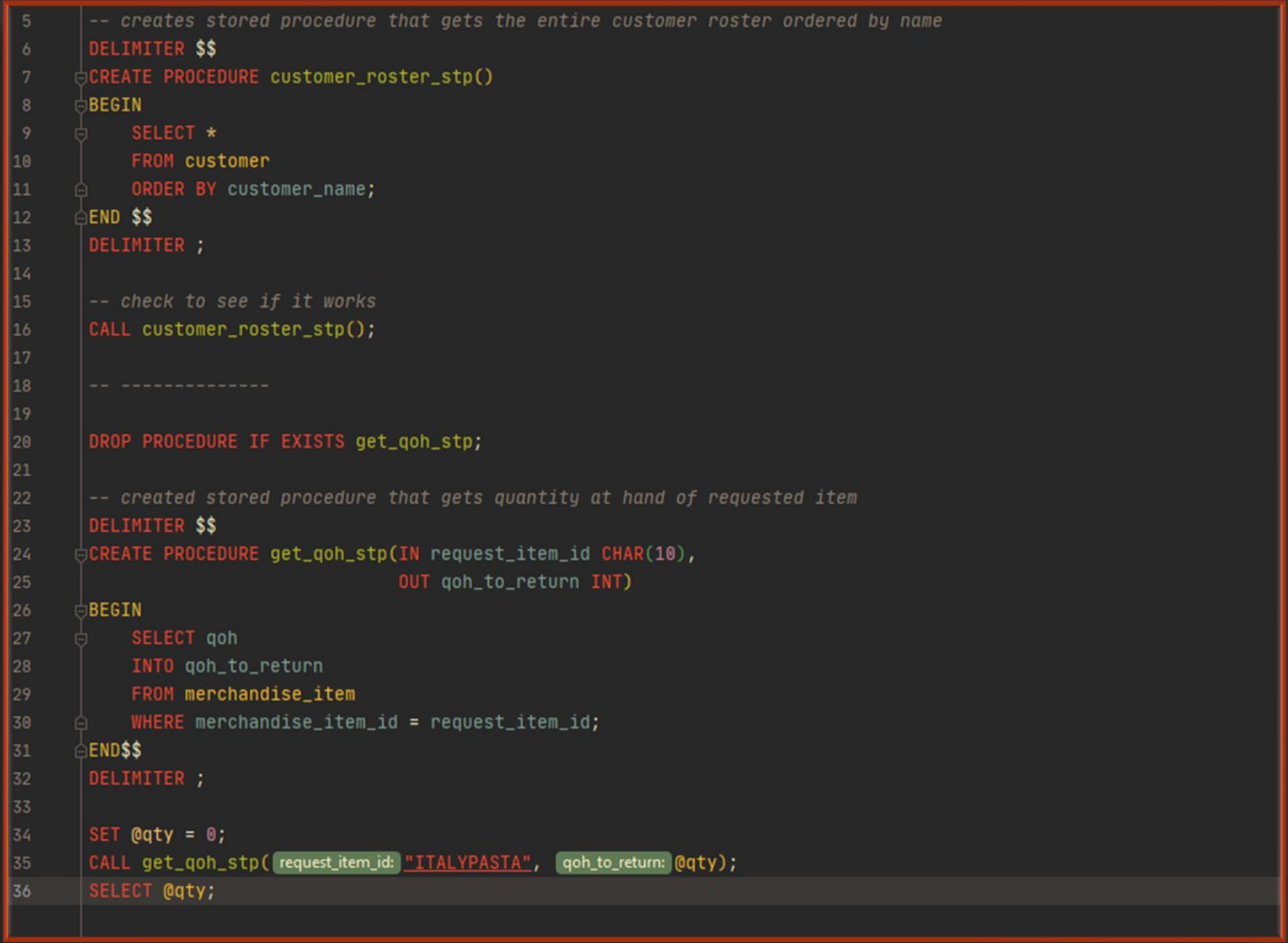
SQL Query Samples

By Efran Himel

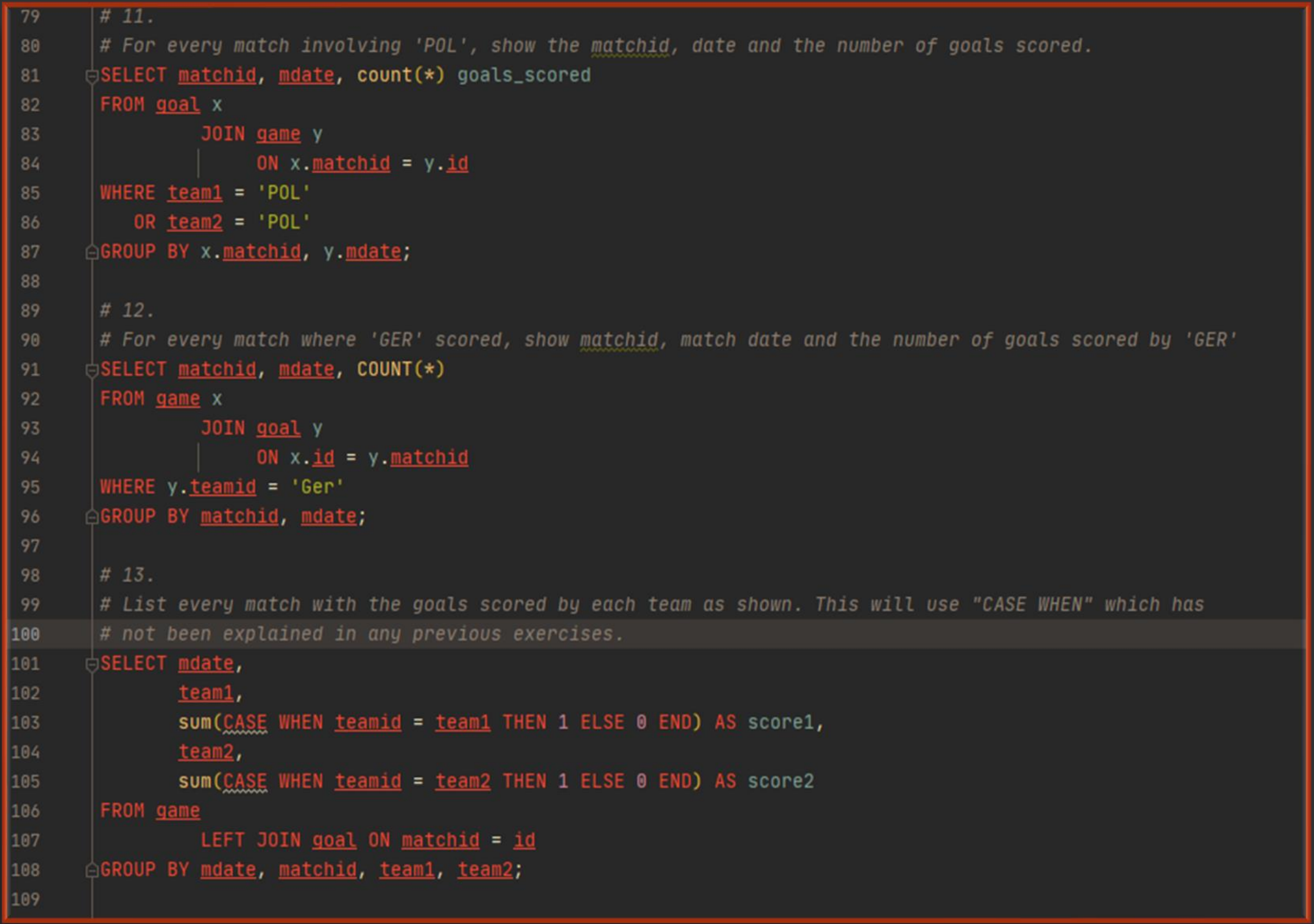
Table of Contents

Selects and Stored Procedures



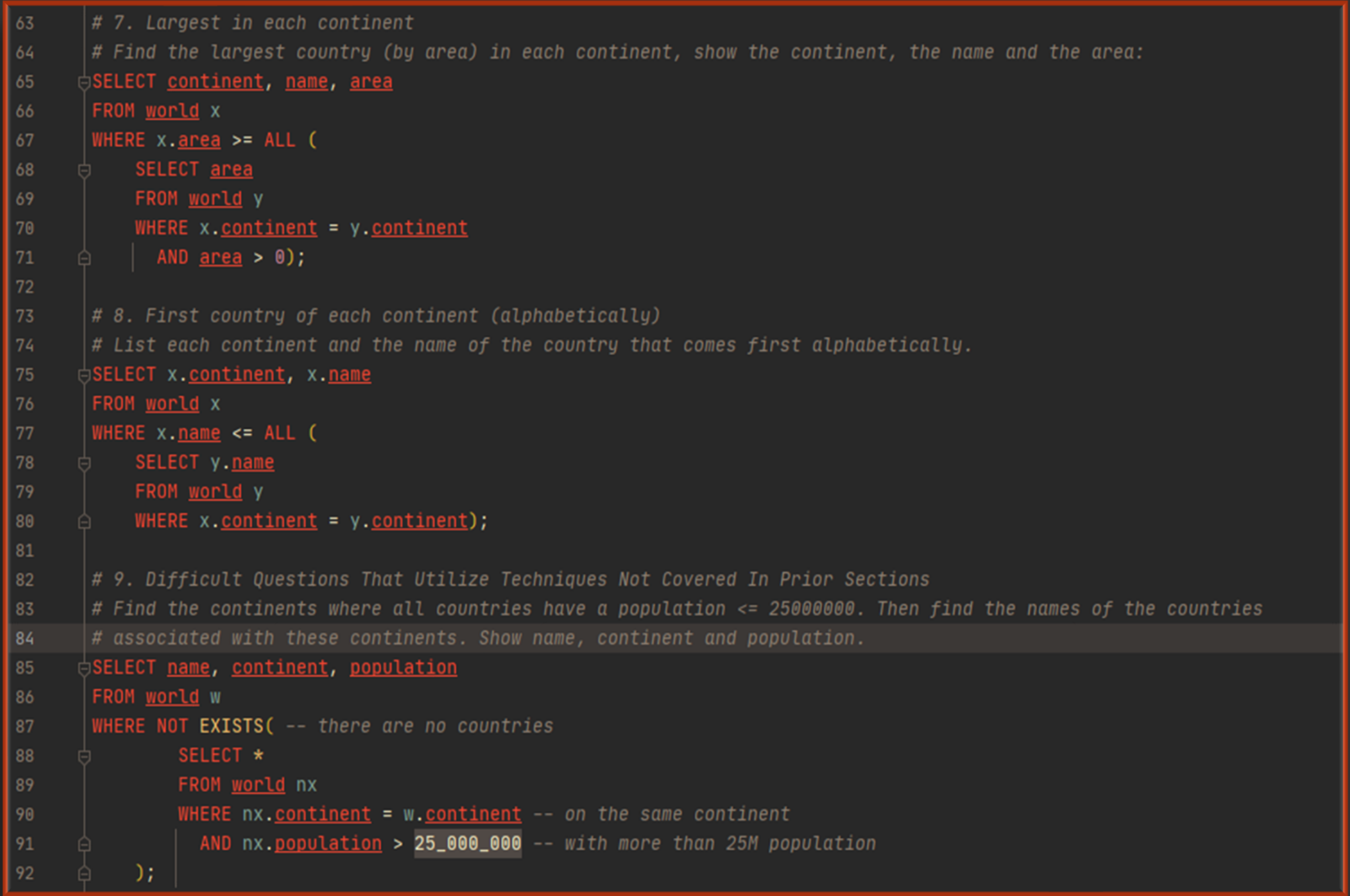
```
-- creates stored procedure that gets the entire customer roster ordered by name
DELIMITER $$
CREATE PROCEDURE customer_roster_stp()
BEGIN
    SELECT *
    FROM customer
    ORDER BY customer_name;
END $$
DELIMITER ;

-- check to see if it works
CALL customer_roster_stp();

-- -------------

DROP PROCEDURE IF EXISTS get_qoh_stp;

-- created stored procedure that gets quantity at hand of requested item
DELIMITER $$
CREATE PROCEDURE get_qoh_stp(IN request_item_id CHAR(10),
                             OUT qoh_to_return INT)
BEGIN
    SELECT qoh
    INTO qoh_to_return
    FROM merchandise_item
    WHERE merchandise_item_id = request_item_id;
END$$
DELIMITER ;

SET @qty = 0;
CALL get_qoh_stp( request_item_id: "ITALYPASTA", qoh_to_return: @qty);
SELECT @qty;
```

Joins



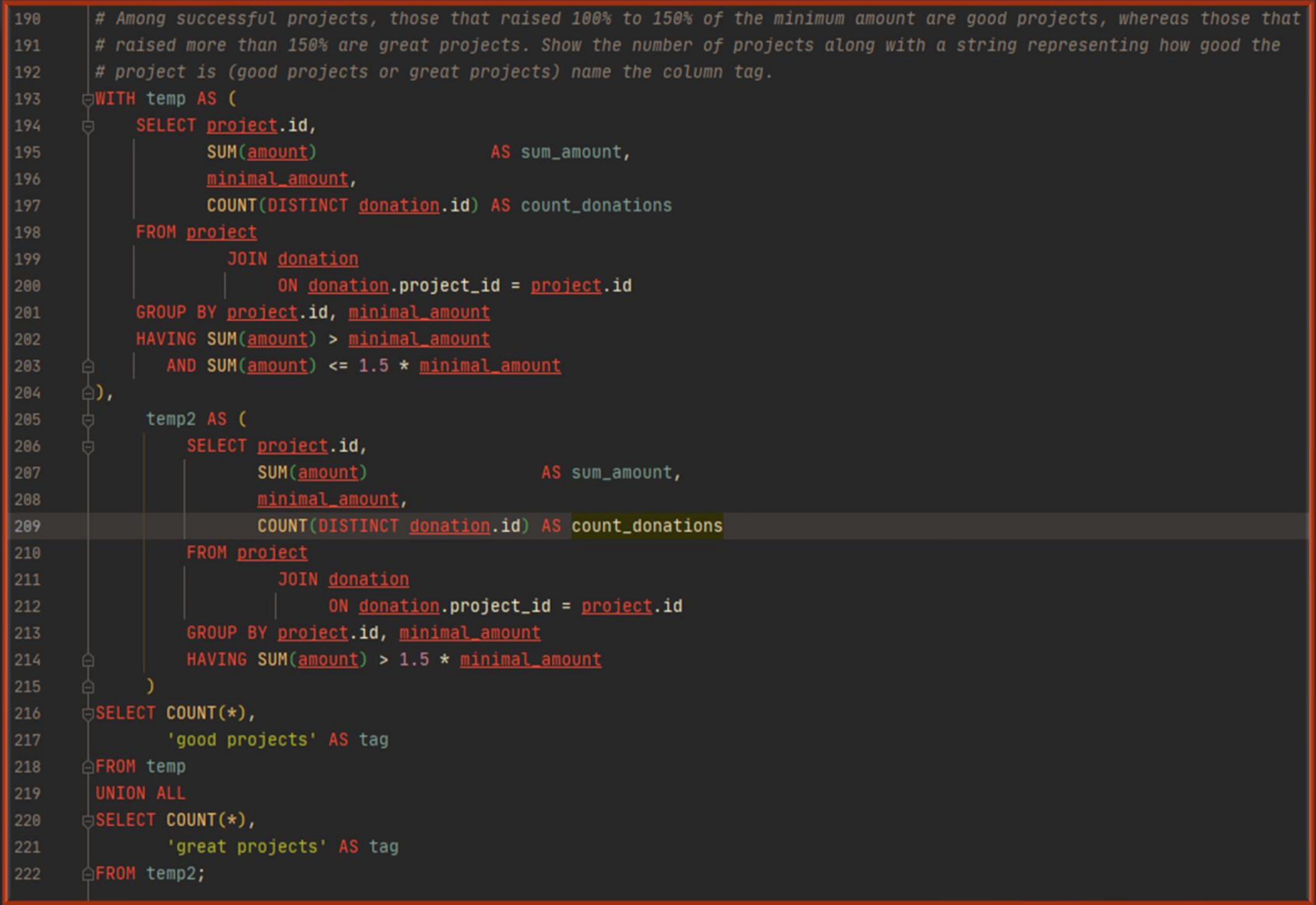
```
# 11.
# For every match involving 'POL', show the matchid, date and the number of goals scored.
SELECT matchid, mdate, count(*) goals_scored
FROM goal x
         JOIN game y
              ON x.matchid = y.id
WHERE team1 = 'POL'
   OR team2 = 'POL'
GROUP BY x.matchid, y.mdate;

# 12.
# For every match where 'GER' scored, show matchid, match date and the number of goals scored by 'GER'
SELECT matchid, mdate, COUNT(*)
FROM game x
         JOIN goal y
              ON x.id = y.matchid
WHERE y.teamid = 'Ger'
GROUP BY matchid, mdate;

# 13.
# List every match with the goals scored by each team as shown. This will use "CASE WHEN" which has
# not been explained in any previous exercises.
SELECT mdate,
       team1,
       sum(CASE WHEN teamid = team1 THEN 1 ELSE 0 END) AS score1,
       team2,
       sum(CASE WHEN teamid = team2 THEN 1 ELSE 0 END) AS score2
FROM game
         LEFT JOIN goal ON matchid = id
GROUP BY mdate, matchid, team1, team2;
```

Subqueries



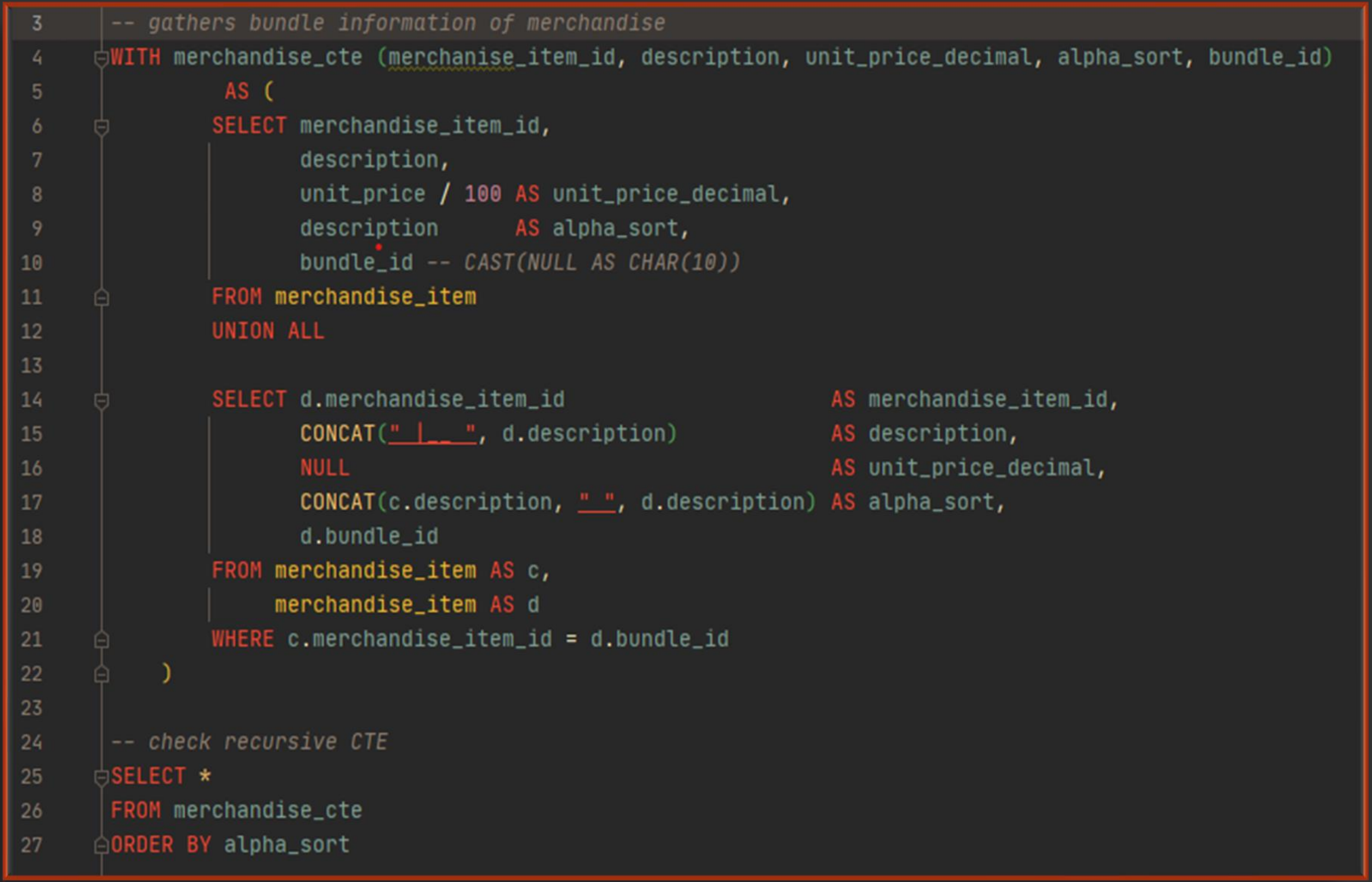
```sql
# 7. Largest in each continent
# Find the largest country (by area) in each continent, show the continent, the name and the area:
SELECT continent, name, area
FROM world x
WHERE x.area >= ALL (
    SELECT area
    FROM world y
    WHERE x.continent = y.continent
      AND area > 0);

# 8. First country of each continent (alphabetically)
# List each continent and the name of the country that comes first alphabetically.
SELECT x.continent, x.name
FROM world x
WHERE x.name <= ALL (
    SELECT y.name
    FROM world y
    WHERE x.continent = y.continent);

# 9. Difficult Questions That Utilize Techniques Not Covered In Prior Sections
# Find the continents where all countries have a population <= 25000000. Then find the names of the countries
# associated with these continents. Show name, continent and population.
SELECT name, continent, population
FROM world w
WHERE NOT EXISTS( -- there are no countries
        SELECT *
        FROM world nx
        WHERE nx.continent = w.continent -- on the same continent
          AND nx.population > 25_000_000 -- with more than 25M population
    );
```

Common Table Expression (CTE)



```
# Among successful projects, those that raised 100% to 150% of the minimum amount are good projects, whereas those that
# raised more than 150% are great projects. Show the number of projects along with a string representing how good the
# project is (good projects or great projects) name the column tag.
WITH temp AS (
    SELECT project.id,
           SUM(amount)                  AS sum_amount,
           minimal_amount,
           COUNT(DISTINCT donation.id) AS count_donations
    FROM project
             JOIN donation
                  ON donation.project_id = project.id
    GROUP BY project.id, minimal_amount
    HAVING SUM(amount) > minimal_amount
       AND SUM(amount) <= 1.5 * minimal_amount
),
     temp2 AS (
         SELECT project.id,
                SUM(amount)                  AS sum_amount,
                minimal_amount,
                COUNT(DISTINCT donation.id) AS count_donations
         FROM project
                  JOIN donation
                       ON donation.project_id = project.id
         GROUP BY project.id, minimal_amount
         HAVING SUM(amount) > 1.5 * minimal_amount
     )
SELECT COUNT(*),
       'good projects' AS tag
FROM temp
UNION ALL
SELECT COUNT(*),
       'great projects' AS tag
FROM temp2;
```

Recursive CTE



```sql
-- gathers bundle information of merchandise
WITH merchandise_cte (merchanise_item_id, description, unit_price_decimal, alpha_sort, bundle_id)
         AS (
        SELECT merchandise_item_id,
               description,
               unit_price / 100 AS unit_price_decimal,
               description      AS alpha_sort,
               bundle_id -- CAST(NULL AS CHAR(10))
        FROM merchandise_item
        UNION ALL

        SELECT d.merchandise_item_id                          AS merchandise_item_id,
               CONCAT(" |__ ", d.description)                 AS description,
               NULL                                           AS unit_price_decimal,
               CONCAT(c.description, " ", d.description)      AS alpha_sort,
               d.bundle_id
        FROM merchandise_item AS c,
             merchandise_item AS d
        WHERE c.merchandise_item_id = d.bundle_id
    )

-- check recursive CTE
SELECT *
FROM merchandise_cte
ORDER BY alpha_sort
```