GROUP-11

Ulaş Yıldız - 24931

Selin Ceydeli - 28877

Berk Kayabaş - 29485

Mehmet Barış Tekdemir - 29068

Canberk Tahıl - 29515

Our GitHub link: https://github.com/berkkayabas/CS306-PROJECT.git

Question 1: Discovering insights from data to find patterns, trends, and correlations

Part-A: In total, we created 7 views, explaining the patterns and relationships we wanted to explore in our datasets.

View 1:

```
Drop view if exists low_democratized_countries;
Create view low_democratized_countries as
Select D.country, D.avg_liberal_dem_idx
from (Select D2.iso_code as country, AVG(D2.liberal_dem) as avg_liberal_dem_idx
        From Democracies_Governedby D2
        Group by D2.iso_code) as D
Where D.avg_liberal_dem_idx <= 0.408
Group by D.country;
```

This view selects the countries with below-average liberal democracy indexes, which we defined as having a liberal democracy index lower than 0.408 (the threshold of 0.408 corresponds to the average liberal democracy index of all countries). The view displays the country code and the corresponding average liberal democracy index, calculated from the table 'Democracies_Governedby' by grouping the entries by country code.

View 2:

```
Drop view if exists high_schooling_countries;
Create view high_schooling_countries as
Select A.country, A.avg_schooling_idx
From (Select C4.iso_code as country, AVG(C4.schooling_index) as avg_schooling_idx
        From cs306_project.schooling_announces C4
        Group by C4.iso_code) as A
Where A.avg_schooling_idx >= 8.3
Group by A.country;
```

This view selects the countries with above-average schooling indexes, which we defined as having a schooling index higher than 8.3 (the threshold of 8.3 corresponds to the average schooling index of all countries). The view displays the country code and the corresponding average schooling index, calculated from the table 'Schooling_Announces' by grouping the entries by country code.

View 3:

```
Drop view if exists high_development_countries;
Create view high_development_countries as
Select hum.country, hum.avg_hum_dev_idx
From (Select HD.iso_code as country, AVG(HD.hum_dev_index) as avg_hum_dev_idx
        From cs306_project.humdev_reports HD
        Group by HD.iso_code) as hum
Where hum.avg_hum_dev_idx >= 0.711
Group by hum.country;
```

This view selects the countries with above-average human development indexes, which we defined as having a human development index higher than 0.711 (the threshold of 0.711 corresponds to the average human development index of all countries). The view displays the country code and the corresponding average human development index, calculated from the table 'HumDev_Reports' by grouping the entries by country code.

View 4:

```
Drop view if exists high_corr_per_countries;
Create view high_corr_per_countries as
Select c.iso as country, c.aver as avg_corr_per_idx
From (Select Co.iso_code as country, AVG(co.corr_per_index) as aver
        From cs306_project.corruppercep_describes CO
        Group By CO.iso_code ) as C
Where c.aver >= 42.805
Group by c.iso;
```

This view selects the countries with above-average corruption perception indexes, which we defined as having a corruption perception index higher than 42.805 (the threshold of 42.805 corresponds to the average corruption perception index of all countries). The view displays the country code and the corresponding average corruption perception index, calculated from the table 'CorrupPercep_Describes' by grouping the entries by country code.

View 5:

```
Drop view if exists high_bribery_countries;
Create view high_bribery_countries as
Select B.iso as country, B.aver as avg_bribe_payers_index
From (Select BH.iso_code as iso, Avg(BH.bribe_payers_index) as aver
    From cs306_project.bribepayers_has BH
    Group by BH.iso_code) as B
Where b.aver >= 7.872
Group by B.iso;
```

This view selects the countries with above-average bribe payers indexes, which we defined as having a bribe payers index higher than 7.872 (the threshold of 7.872 corresponds to the average bribe payers index of all countries). The view displays the country code and the corresponding average bribe payers index, calculated from the table 'BribePayers_Has' by grouping the entries by country code.

Part-B: We decided to use the INTERSECT and INNER JOIN operators between two views: high_corr_per_countries and low_democratized_countries.

```
SELECT H.country FROM high_corr_per_countries H
INTERSECT
SELECT D.country FROM low_democratized_countries D;

SELECT H.country
FROM high_corr_per_countries H
JOIN low_democratized_countries D
ON H.country = D.country;
```

The first query uses the INTERSECT operator to find the countries that appear in both views. The result is a list of countries that have high levels of corruption perception and low levels of liberal democracy.

The second query uses the JOIN operator to combine the two views on the "country" column and again returns the list of countries that have high levels of corruption perception and low levels of liberal democracy. The output is the same as the first query.

Part-C:

IN Operator:

```
SELECT L.country, L.avg_liberal_dem_idx, H.avg_bribe_payers_index
FROM low_democratized_countries L, high_bribery_countries H
WHERE L.country IN (SELECT H2.country
                    FROM high_bribery_countries H2);
```

This query is finding countries that have both low levels of democratization and high levels of bribery and then displays the average liberal democracy index and average bribe payers index for those countries. The subquery is looking for countries that appear in both "low_democratized_countries" and "high_bribery_countries" tables. In the subquery, we don't need to mention any attribute of the "low_democratized_countires" table.

EXISTS Operator:

```
SELECT L.country, L.avg_liberal_dem_idx, H.avg_bribe_payers_index
FROM low_democratized_countries L, high_bribery_countries H
WHERE EXISTS (SELECT *
                FROM high_bribery_countries H2
                WHERE L.country = H2.country);
```

This query is finding countries that have both low levels of democratization and high levels of bribery and then displays the average liberal democracy index and average bribe payers index for those countries. The subquery is looking for countries that appear in both "low_democratized_countries" and "high_bribery_countries" tables. In the subquery, we do need to mention any attribute of the "low_democratized_countires" table.

Part-D:

Aggregate Operation 1 (Used AVG, MIN, MAX, and COUNT operators):

```
Select C.iso_code, C.name, AVG(D.liberal_dem) as avg_liberal_dem, MIN(D.liberal_dem) as min_liberal_dem,
        MAX(D.liberal_dem) as max_liberal_dem
From Countries C, Democracies_Governedby D
Where C.iso_code = D.iso_code
Group by D.iso_code, C.name
Having COUNT(*) > 1;
```

The above select statement is selecting data from the Countries and Democracies_Governedby tables and calculating the average, minimum, and maximum values of the liberal_dem column for each country that has more than one entry in the Democracies_Governedby table. The output will include the country code, country name, and the corresponding liberal democracy metrics.

Aggregate Operation 2 (Used AVG, MIN, MAX, and COUNT operators):

```
Select C.iso_code, C.name, AVG(CP.corr_per_index) as avg_corr_per_idx,
        MIN(CP.corr_per_index) as min_corr_per_idx, MAX(CP.corr_per_index) as max_corr_per_idx
From Countries C, CorrupPercep_Describes CP
Where C.iso_code = D.iso_code
```

```
Group by D.iso_code, C.name
Having COUNT(*) > 1;
```

The above select statement is selecting data from the Countries and CorrupPercep_Describes tables and calculating the average, minimum, and maximum values of the corr_per_index column for each country that has more than one entry in the CorrupPercep_Describes table. The output will include the country code, country name, and the corresponding corruption perception metrics.

Aggregate Operation 3 (Used SUM operator):

```
Select CP.iso_code, C.name, SUM((CP.bribery_rate)*CP.population)/COUNT(CP.year_) as
        estimated_bribery_incidents
From cs306_project.Countries C, cs306_project.CorrupPercep_Describes CP
Where C.iso_code = CP.iso_code
Group by CP.iso_code, C.name;
```

The above select statement is calculating the estimated number of bribery incidents for each country by multiplying the bribery rate with the population of the country for each year, summing up the products, and dividing by the number of years for which there is data available to find an average. The output shows the country code, country name, and estimated number of bribery incidents for each country that has data available in the "CorrupPercep_Describes" table.

Aggregate Operation 4 (Used MIN and MAX operators):

```
Select C.iso_code, C.name, MIN(H.hum_dev_index), MIN(S.schooling_index), MIN(D.liberal_dem),
        MAX(CP.corr_per_index), MAX(B.bribe_payers_index)
From Countries C, HumDev_Reports H, Schooling_Announces S, Democracies_Governedby D,
        CorrupPercep_Describes CP, BribePayers_Has B
Where C.iso_code = H.iso_code AND C.iso_code = S.iso_code AND C.iso_code = D.iso_code AND
        C.iso_code = CP.iso_code AND C.iso_code = B.iso_code
Group by H.iso_code, S.iso_code, D.iso_code, CP.iso_code, B.iso_code;
```

The above select statement aims to show how corrupt countries are by retrieving the minimum and maximum values of different indices. It displays the minimum values of human development, schooling, and liberal democracy indices and the maximum values of corruption perception and bribe payers indices.

Aggregate Operation 5 (Used MIN and MAX operators):

```
Select C.iso_code, C.name, MAX(H.hum_dev_index), MAX(S.schooling_index), MAX(D.liberal_dem),
        MIN(CP.corr_per_index), MIN(B.bribe_payers_index)
```

```
From Countries C, HumDev_Reports H, Schooling_Announces S, Democracies_Governedby D,
        CorrupPercep_Describes CP, BribePayers_Has B
Where C.iso_code = H.iso_code AND C.iso_code = S.iso_code AND C.iso_code = D.iso_code AND
        C.iso_code = CP.iso_code AND C.iso_code = B.iso_code
Group by H.iso_code, S.iso_code, D.iso_code, CP.iso_code, B.iso_code;
```

Similar to the fourth select statement, this select statement aims to show how virtuous, i.e. not corrupted, countries are by retrieving the minimum and maximum values of different indices. Opposite to the fourth select statement, this statement displays the maximum values of human development, schooling, and liberal democracy indices and the minimum values of corruption perception and bribe payers indices.

Question 2: Constraints and triggers

Constraint:

```
ALTER TABLE cs306_project.corruppercep_describes
ADD CONSTRAINT index_range_brib CHECK ( bribery_rate >= 0 AND bribery_rate <= 84 );
```

After determining the minimum and maximum values for bribery_rate in our corruppercep_describes table, we have created a constraint using ALTER TABLE to check and prevented entering values outside the range in the bribary_rate column. With this constraint, entering values lower than 0 or higher than 84 is prevented.

```
INSERT INTO cs306_project.corruppercep_describes (bribery_rate) VALUES (85);
```

We then checked whether our constraint correctly works or not by trying to insert a value higher than the maximum value and as expected, an error occurred in the log file.

Trigger 1: Check before inserting

```
DELIMITER //
CREATE TRIGGER index_range_check_insert
BEFORE INSERT ON cs306_project.corruppercep_describes
FOR EACH ROW
BEGIN
  IF NEW.bribery_rate < 0 THEN
    SET NEW.bribery_rate = 0;
  ELSEIF NEW.bribery_rate > 84 THEN
    SET NEW.bribery_rate = 84;
  END IF;
```

```
END //
DELIMITER ;
```

Trigger 2: Check before updating

```
DELIMITER //
CREATE TRIGGER index_range_check_update
BEFORE update ON cs306_project.corruppercep_describes
FOR EACH ROW
BEGIN
  IF NEW.bribery_rate < 0 THEN
    SET NEW.bribery_rate = 0;
  ELSEIF NEW.bribery_rate > 84 THEN
    SET NEW.bribery_rate = 84;
  END IF;
END //
DELIMITER ;
```

These triggers check the value being inserted or updated in the “bribery_rate” field, and if it is outside of the allowed range, the triggers set the value being inserted or updated to either the minimum or maximum value.

Comparing the pros and cons of general constraints and trigger methods:

Creating general constraints provides various pros. It is an efficient and easy way to get a wide range of data integrity rules, such as check constraints, unique constraints, and foreign key constraints. Also, there is no need to type extra code to enforce the constraint. It is automatically done by the database management system.

On the other hand, general constraints are inflexible. It can only apply a single rule to a single table. Also, It may be hard to remove those constraints if the table has a vast amount of data.

Triggers also have specific pros and cons. The most important advantage of triggers is that they are more flexible in terms of modification and deletion. It allows the user to type custom codes to perform more advanced checks. Plus, triggers can be used to execute on multiple tables.

However, they are complex structures as they require custom codes. If they are poorly designed and executed frequently there is a big chance that it can cause significant performance overhead.

In summary, general constraints are a simple and effective way to enforce data integrity in a relational database, while triggers offer more flexibility and can be used to enforce more complex rules. The choice between the two methods ultimately depends on the specific requirements of the database and the complexity of the data integrity rules that need to be enforced.

Question 3: Stored Procedure

```
DELIMITER //
CREATE PROCEDURE corr_per_data (IN iso_code CHAR(11))
BEGIN
  IF iso_code = 'TUR' THEN
    SELECT * FROM corruppercep_describes C WHERE C.iso_code = 'TUR;
  ELSEIF iso_code = 'RUS' THEN
    SELECT * FROM corruppercep_describes C WHERE  C.iso_code = 'RUS';
  ELSEIF iso_code = 'USA' THEN
    SELECT * FROM corruppercep_describes C WHERE  C.iso_code = 'USA';
  ELSEIF iso_code = 'SWE' THEN
    SELECT * FROM corruppercep_describes C WHERE  C.iso_code = 'SWE';
  ELSE
    SELECT 'Invalid ISO code';
  END IF;
END //
DELIMITER ;

CALL corr_per_data('RUS');
CALL corr_per_data('TUR');
CALL corr_per_data('USA');
CALL corr_per_data('SWE');
```

This stored procedure takes in one parameter, iso_code, which is the code for a country and returns data related to the corruption perception index and bribery rate of the corresponding country from the Corruppercep_Describes table. It returns data for iso-codes 'TUR', 'USA', 'RUS', and 'SWE' and outputs an error message for any other iso-codes.

Files we have submitted to GitHub:

- **CS306 Project_Step-3 Report_Group-11:** is our step-3 report (pdf file)
- **views.sql:** has all our SQL statements

- **sql_actions_local_Server(step 3).log:** is our log file **(check the lines starting from 6274 for step-3 logs)**