Obligatorisk aflevering 1

Af Kristofer, Michael, Søren og Mads

Explore the data and formulate considerations about a hosting database.

To create a database, we must first explore and understand the different types of data in each dataset, as well as how the different datasets are connected to each other with the data they contain. In this section we will go over the different datasets and look at what columns they contain, and for each column we will look through the data contained within to determine what type of value they are. We will also try to give each a quick description to help us understand the relationships between all of the datasets.

Data exploration:

2016_-_Cities_Emissions_Reduction_Targets_20240207.csv:

Organisation: Looking at the column of data we can see that they are strings writing in the native language where the organization resides. The data are names of different government organizations overseeing a city. The value is unique.
Account No: Account number or id in non-sequential order. The value is unique to each organization.
Country: The country of which the organization is located within. The values are string and contain names. One country can have multiple organizations within it. Unlike the values of Organisation the values are all written in english.
City Short Name: The shortened name of the city where the organization resides. The values are string and some contain special characters.
C40: **Currently unknown!** Some rows of the column “C40” contain the value of “C40”. While the dataset data of this column is a string containing both letters and numbers, the value could be stored as a bool to reduce memory. What “C40” means is currently unknown in relation to the rest of the dataset.
Reporting Year: The year the organization gave the report from which the data of the row is gathered from. The data is an integer.
Sector: From which part of the organization the data has been gathered from. The values first read as an enum but with a few rows of data where multiple choices are included means the value should be a list of enums.
Target boundary: The limit where emission data has been collected within. The value is a string and is used as a description. The value in each row varies with some rows missing data.

Baseline year: The year where the baseline measurements were taken. The value is an integer.
Baseline emissions (metric tonnes CO2e): The emission value measured as the baseline. The value is a float and represents metric tons.
Percentage reduction target: The desired reduction of the baseline emission measured. The value is a float and represents percentage.
Target date: The year in which the organization aims to have the baseline emission reduced by the Percentage reduction target. The value is an integer.
Comment: String value containing a comment from the rapport. Some rows have missing values.
City Location: Coordinates of the city's location which most likely refers to the center of the city. The value is a string considering the use of parentheses. Within the parentheses there are two numbers separated by a comma: latitude and longitude.
Country Location: Coordinates of the county’s location which most likely refers to the center of the country. The value is a string considering the use of parentheses. Within the parentheses there are two numbers separated by a comma: latitude and longitude.

2016_-_Citywide_GHG_Emissions_20240207.csv:

Account Number: Account number or id in non-sequential order.
City Name: The full name of the city. The value is a string.
Country: The country of which the organization is located within. The values are string and contain names. One country can have multiple organizations within it. Unlike the values of Organisation the values are all written in english.
City Short Name: The shortened name of the city where the organization resides. The values are string and some contain special characters.
C40: The value type is a string and represents if a city can be labeled as a C40 city.
Reporting Year: The year the organization gave the report from which the data of the row is gathered from. The data is an integer.
Measurement Year: The year where the measurements were taken. The value is a date.
Boundary: The limit where emission data has been collected within. The value is a string and is used as a description. The value in each row varies with some rows missing data.
Primary Methodology: Name of the methodology used for the data collection. The value is a string.
Methodology Details: A description of the Primary Methodology used. The value is a string.
Gases included: What type of gases are included in the measurements. The value can contain multiple choices which could be considered enums since the gases that can be included appear to be static.
Total City-wide Emissions (metric tonnes CO2e): The value is a float and represents metric tons.
Total Scope 1 Emissions (metric tonnes CO2e): The value is a float and represents metric tons.
Total Scope 2 Emissions (metric tonnes CO2e): The value is a float and represents metric tons.
Increase/Decrease from last year If there have been any changes in their measurements compared to last year. The value is an enum which represents “=, >, <” or if it's their first measurement.

Reason for increase/decrease in emissions: A description given in the report. The value is a string.
Current Population Year: The year of the Current Population. The value is an integer.
Current Population: The measured number of people living in the country. The value is an integer
City GDP: The Gross Domestic Product of the market value within a city. The value is an integer.
GDP Currency: What type of currency the City CDP refers to. The value is a string which can be separated into the long name and short name of the currency
Year of GDP: The year of the City GDP was measured. The value is an integer.
GDP Source: Where the information regarding the GDP was gathered. The value is a string and can both be a name of an organization or a link to a website.
Average annual temperature (in Celsius): The average annual temperature of the city in celsius. The value is a float.
Land area (in square km): The size of the city in square kilometers. The value is a float.
Average altitude (m): The average altitude in meters of the city. The value is an integer.
City Location: Coordinates of the city's location which most likely refers to the center of the city. The value is a string considering the use of parentheses. Within the parentheses there are two numbers separated by a comma: latitude and longitude.
Country Location: Coordinates of the county’s location which most likely refers to the center of the country. The value is a string considering the use of parentheses. Within the parentheses there are two numbers separated by a comma: latitude and longitude.

2017_-_Cities_Community_Wide_Emissions.csv:

Account number: Account number or id in non-sequential order. The value is unique to each organization.
Organization: Looking at the column of data we can see that they are strings writing in the native language where the organization resides. The data are names of different government organizations overseeing a city. The value is unique.
City: The full name of the city. The value is a string.
County: The country of which the organization is located within. The values are string and contain names. One country can have multiple organizations within it. Unlike the values of Organisation the values are all written in english.
Region: Where on the world map the country is located. The value is a string and represents the different names of the world regions.
C40: The value type is a string and represents if a city can be labeled as a C40 city.
Access: If the report is accessible to the public or private. It would be reasonable to assume that there would be reports which are or would be marked as private then we can reflect that the data is or is not public. The value is a bool representing public or private.
Reporting year: The year the organization gave the report from which the data of the row is gathered from. The data is an integer.
Accounting year: The period of time where the data gathered in the report originated from. The value is a string containing two dates(YYYY-MM-DD) separated by a “-” and represents a period of time.
Boundary: The limit where data has been collected within. The value is a string and is used as a description. The value in each row varies with some rows missing data.
Protocol: The name of the protocol used for the gathering of information. The value is a string.
Protocol column: A description of how the protocol was used to gather information for the report. The value is a string.
Gases included: A description of the type of gases included when gathering information. The value is a string and contains mostly a list of the chemical name of the gases but also contains a description.

Total emissions (metric tonnes CO2e): The total measured emissions of CO2 in tons. The value is float. Most of the values in the column could be integers but the few who aren't forces the others.
Scopes Included: What scopes are included in the report. The value is a string.
Total Scope 1 Emissions (metric tonnes CO2e): The value is a float and represents metric tons.
Total Scope 2 Emissions (metric tonnes CO2e): The value is a float and represents metric tons.
Comment: A comment in there may be in the report. The value is a string and can contain empty values.
Increase/Decrease from last year: If there have been any changes in their measurements compared to last year. The value is an enum which represents "=, >, <" or if it's their first measurement.
Reason for increase/decrease in emissions: A description given in the report. The value is a string.
Population:% The measured number of people living in the country. The value is an integer
Population year: The year of the Population. The value is an integer.
GDP: The Gross Domestic Product of the market value within a city. The value is an integer.
GDP Currency: What type of currency the City CDP refers to. The value is a string which can be separated into the long name and short name of the currency
GDP Year: The year of the City GDP was measured. The value is an integer.
GDP Source: Where the information regarding the GDP was gathered. The value is a string and can both be a name of an organization or a link to a website.
Average annual temperature (in Celsius): The average annual temperature of the city in celsius. The value is a float.
Average altitude (m): The average altitude in meters of the city. The value is an integer.
Land area (in square km): The size of the city in square kilometers. The value is a float.
City Location: Coordinates of the city's location which most likely refers to the center of the city. The

value is a string considering the use of parentheses. Within the parentheses there are two numbers separated by a comma: latitude and longitude.
Country Location: Coordinates of the county’s location which most likely refers to the center of the country. The value is a string considering the use of parentheses. Within the parentheses there are two numbers separated by a comma: latitude and longitude.

2017_-_Cities_Emissions_Reduction_Targets_20240207.csv:

Account No: Account number or id in non-sequential order.
Organisation: Looking at the column of data we can see that they are strings writing in the native language where the organization resides. The data are names of different government organizations overseeing a city. The value is unique.
City: The full name of the city. The value is a string.
Country: The country of which the organization is located within. The values are string and contain names. One country can have multiple organizations within it. Unlike the values of Organisation the values are all written in english.
Region: Where on the world map the country is located. The value is a string and represents the different names of the world regions.
Access: If the report is accessible to the public or private. It would be reasonable to assume that there would be reports which are or would be marked as private then we can reflect that the data is or is not public. The value is a bool representing public or private.
C40: The value type is a string and represents if a city can be labeled as a C40 city.
Reporting year: The year the organization gave the report from which the data of the row is gathered from. The data is an integer.
Type of target: The value type is an enum: “Absolute target” “Base year intensity target” “Baseline scenario (business as usual) target”
Sector: The value type is string. It would be an enum if not for “Other” which offers a string description.
Baseline year: The year where the baseline measurements were taken. The value is an integer.
Baseline emissions (metric tonnes CO2e): The emission value measured as the baseline. The value is a float and represents metric tons.
Percentage reduction target: The desired reduction of the baseline emission measured. The value is a float and represents percentage.

Target date: The year in which the organization aims to have the baseline emission reduced by the Percentage reduction target. The value is an integer.
Estimated business as usual absolute emissions in target year (metric tonnes CO2e): How much CO2 is estimated to be produced as a byproduct of business. The type is an integer and can be empty.
Intensity unit (emissions per): The value type is a string and is a description of what “per” the measurements are taking.
Comment: String value containing a comment from the rapport. Some rows have missing values.
Population: The measured number of people living in the country. The value is an integer
Population Year: The year of the Population. The value is an integer.
City Location: Coordinates of the city's location which most likely refers to the center of the city. The value is a string considering the use of parentheses. Within the parentheses there are two numbers separated by a comma: latitude and longitude.
Country Location: Coordinates of the county’s location which most likely refers to the center of the country. The value is a string considering the use of parentheses. Within the parentheses there are two numbers separated by a comma: latitude and longitude.

2023_Cities_Climate_Risk_and_Vulnerability_Assessments_20240207.csv:

Questionnaire: The value is a string. All the values within the column are the same: “Cities 2023”.
Organization Number: Account number or id in non-sequential order.
Organization Name: Name of the organization. The value is a string and the names within are written in the languages of the country it resides in.
City: Name of the city the organization resides in. The value is a string and some values are empty.
Country/Area: The country the city and organization resides in. The value is a string and the names are written in english.
CDP Region: The state/region relevant CDP reporting platform. The value is a string and contains the name of a region.
C40 City: The value is a bool.
GCoM City: The value is a bool.
Access: If the report is accessible to the public or private. It would be reasonable to assume that there would be reports which are or would be marked as private then we can reflect that the data is or is not public. The value is a bool representing public or private.
Assessment attachment and/or direct link A PDF of download link for an assessment document. The value may be meant to contain a pdf file but all values within the column is a string with some being a filename and extension name, and some being a web link.
Confirm attachment/link provided Confirmation of which type of attachment was given. The values indicate an enum of “PDF”, “Link” but also contains “Other” where a description can be given, therefore making the value a string.
Boundary of assessment relative to jurisdiction boundary: The value is an enum indicating the scale of the report. “Smaller - covers only part of the jurisdiction, please explain exclusions: Área referente ao bairro de Porto de Pedra, correspondendo ao mapeamento de risco de movimento de massa e maior abrangência nas ruas Maria de Souza, Dom Marcos de Noronha, Senador Álvaro Uchoa.”

“Partial - covers part of the jurisdiction and adjoining areas, please explain exclusions/additions: Zona precordillerana y cordillerana de Peñalolén. Incluye la principal cuenca de la comuna, abarcando comuna aledaña” “Same - covers entire jurisdiction and nothing else” “Larger - covers the whole jurisdiction and adjoining areas, please explain additions: This is a county-wide plan”
Year of publication or approval: The year where the report was either publicized or approved. The value is an integer.
Factors considered in assessment: A description of what was considered during the assessment. The value is a string.
Primary author(s) of assessment: Who oversaw the assessment. The value is a string and can be empty.
Does the city have adaptation goal(s) and/or an adaptation plan?: The data indicates that the value is of type enum. “Adaptation goal(s) and adaptation plan” “Adaptation plan” “Adaptation goal(s)” “Incomplete report”
Population: The measured number of people living in the country. The value is an integer
Population Year: The year of the Population. The value is an integer.
City Location: The coordinates of the city. The value is a string. The coordinates are within two parentheses and separated with a space.
Last update: This value contains a date and represents when the values of the row was last updated.

Dataset relationship:

Here we will describe how the different datasets are connected to each other through their columns, values, data and a description of the overall meaning of the different reports represented in the datasets. We will also describe what each dataset represents.

From here each dataset will be know as:

Reduc16:

- 2016_-_Cities_Emissions_Reduction_Targets_20240207.csv

GHG:

- 2016_-_Citywide_GHG_Emissions_20240207.csv

Comm:

- 2017_-_Cities_Community_Wide_Emissions.csv

Reduc17:

- 2017_-_Cities_Emissions_Reduction_Targets_20240207.csv

RiskAndVul:

- 2023_Cities_Climate_Risk_and_Vulnerability_Assessments_20240207

Firstly we can see how they relate to each other by doing a quick comparison of some of the different columns in each dataset. We can see that many of them share a lot of data even if the names of the columns don't entirely match across the board. From doing this quick comparison we can also see that they all share an account number which we can use to easily create relationships within our database.

	GHG	Reduc16	Reduc17	Comm	RiskAndVul
Account number	Account Number	Account No	Account No	Account number	Organization Number
Organisation			Organisation	Organization	Organization Name
City	City Name		City	City	City
Country	Country	Country	Country	Country	Country/Area
Access			Access	Access	Access
C40	C40	C40	C40	C40	C40 City
Reporting year	Reporting Year	Reporting Year	Reporting year	Reporting year	
Baseline year	Measurement Year	Baseline year	Baseline year	Accounting year	
Baseline emission	Total City-wide Emissions (metric	Baseline emissions (metric tonnes CO2e)	Baseline emissions (metric tonnes CO2e)	Total emissions (metric tonnes CO2e)	

	tonnes CO2e)				

Organization account number:
Through each of the datasets there is a column, though with a different column name, that represents the account number of an organization that has submitted a report to at least one of the datasets. We can therefore connect all the different reports in the different datasets through their unique account number.

City, County and Coordinates:
The data can give us a picture of where the work against climate change is strongest. Throughout all five datasets there is a consistent representation of where in the world the data is coming from. The consistency only breaks in the newest dataset RiskAndVul, where the setup of the coordinates are different from the other datasets, though the data when extracted is still useful. In RiskAndVul the way the data has been save was: *"POINT (113.813 22.9175)"* and also with the possibility of empty data, where as the other datasets all setup the data as *"*(56.168393, 10.137373)*"* with no empty data.
Since the data is consistent over the different datasets we don't need to consider this difference as long as we save the data so that when the coordinates are extracted as longitude and latitude numbers correctly.

City names and organization:
Looking at the data where both a column of the organization names and city names, can we see that each organization appears to be a government organization tied directly to the city it resides in.

Reduc16 and Reduc17:
Looking at the data within these two datasets and their names, we can see that they are of the same type of report separated by a year. We can however also see that the dataset from 2017 includes new columns of data which separates them from being completely identical to each other.

Formulation:

Here we will give a little overview of how we have chosen to set up our database.

Normal form:

In the database we have adhered to the first 3 normalization rules. This will reduce redundancies which means reducing duplicated data, reducing storage space and improved data integrity.

The first normal form says that each row needs a unique primary key. This is done in example our table “city” where we have used the account no/organization no as a primary key. By using this id, which is consistent within several tables, we are able to extract relevant information from several and place them in a singular table. In the “case of cities community wide emissions” table, that a single city could have several lines of data which would mean using the account no is not a viable strategy as they would not be unique anymore. This is why it has a key that is auto incremented. In the case with “country”, the name for each country is unique therefore the name of the country was used as its primary key.

2NF is focused on eliminating redundancy and ensuring database integrity by requiring that all non-key attributes are fully functional and dependent on the primary key. For a database to adhere to 2NF, every column that is not part of the primary key must depend entirely on the primary key for its existence. Examples of this can be seen in our city table, while it could be easier to implement by having the countries data in the same table or row as the city. As the country is not dependent on the city, to follow the 2NF we will have to make a separate table.

3NF is a design that ensures that every non-key attribute in a table directly depends on the primary key and not on any other non- key attribute. This means a table is organized in a way to avoid indirect relationships within. This can be seen in examples of “gdp” and “populations”. If we used a similar structure to the original dataset, then columns on gdp and population will be in tables “cities_wide_emissions” and “ghg_emissions”. This would break 2NF as the data on population and GDP are not dependent on the city, which is not a primary key. It was then chosen to make separate tables for gdp and population.

Replicated:

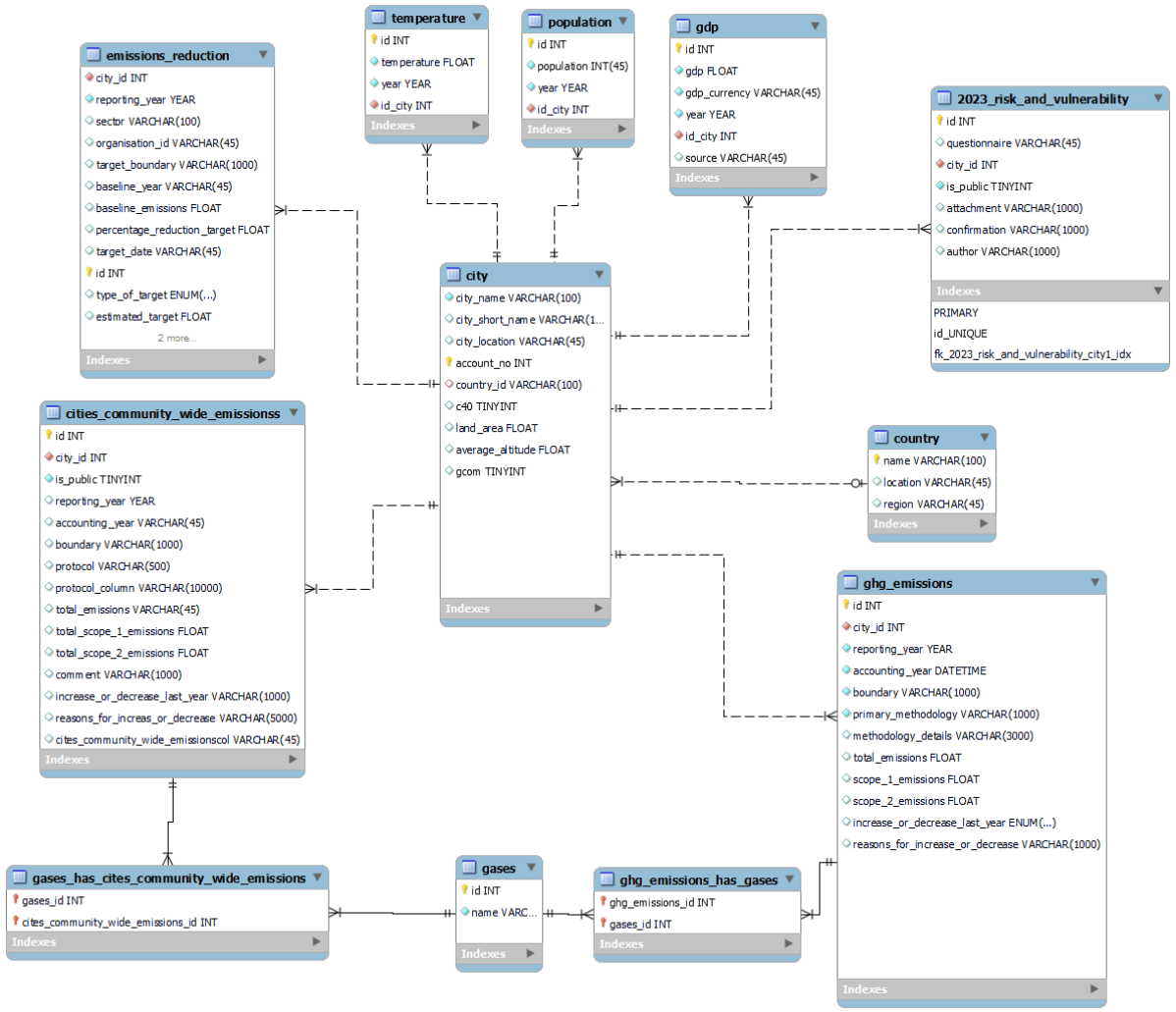
We have decided to use transactional replication, primarily for load balancing and high availability. Load balancing can improve performance by distributing operations across multiple servers. Creating several replicas of the main database, ensures minimal downtime in the event of a primary database failure, which improves availability.

We have set up our database to use transactional replication, with a publisher/distributor server and 3 subscriber servers. We have initially created a snapshot of the publisher database, which is replicated to the subscribers, and from this point the database is set up to replicate any transaction received by the publisher to the subscribers as well. We have achieved this by creating separate users with specific roles and access rights, for example the snapshot user has full control, where the logreader only has read access, ensuring that the replication process operates smoothly and securely without compromising the integrity of the data or the performance of the system. This is set up in the Server Agent to handle the continued replication.

Design and develop proper database structure of the requested type.

SQL:

Here you can see the EER diagram which we have developed and which we use to generate a script for setting up or resetting our database for the development process.



Ingest the data into the database, including pre-processing of it, if necessary.

Here we will discuss how we introduce the data from the five datasets into our database. We will go over any data that we may or may not exclude, due to for example being unnecessary, and we will go over any data that may be the same in different datasets but with different data value types and how we will handle it.

Deciding include/exclude:

The first thing we need to push to our database is the organization's names and account number. We first need to determine whether or not we will include organizations with only an account number and no name tied to it. The reason for this is because only four of five of the datasets include a column representing the organization's name whilst they all include account number. It is in the *GHG* dataset that we have no column containing the names.We would first push the four datasets with names included so that there's only one row per unique account number with a name tied to it, and thereafter we could push from *GHG* any account number that doesn't already exist within the database. This would ensure all the data from the different datasets are included but could also lead to incomplete data when pulled for use. There is also the chance that we will find no new account number in *GHG* which hadn't already been pushed before from the other datasets. It should be noted that while they don't all contain the data of the names they do all include the name of the city it resides in, meaning the organization can be identified later if the name of the city and the country the city resides within are included. We have chosen to store organizations and cities together in one table called "City" with a primary key from the account number. This ensures that all account numbers are present even if they don't contain all of the names associated, such as organization name, city name or city short name.

In the datasets *Reduc17* and *RiskAndVul* we see the column "Access" which we view as showing whether or not the general public have access to the report. The data within both datasets is however always "Public" which means we don't need to push this data to the database as we can generalize that all data, from all five datasets, pushed and pulled to the database is public. We have however chosen to still include it in our database because we have determined it would be better for future proofing the database. While no data currently have any other value than "Public", we would rather not run the risk of future reports missing this important difference should there be any that contain the value "Private".

When we have to handle the overlapping data between the datasets we need to determine how we will handle each report type from the datasets. *Reduc16* and *Reduc17* both are the same type of report with the latter including new columns. With them being literally the same report but from different years mean we could store them together in the same table. This however would introduce many empties for the *Reduc16* data arriving from the new columns introduced from *Reduc17*. We have chosen to store the reports from the two datasets together in one table within our database.

Translating data type to new type:

For nearly every instance where a column contains data representing a year of the calendar they have a single integer number, but there are instances where the data type is represented by a variable type of date. This would introduce conflicts when pushing the data to the database. When deciding to translate an integer number to a date or a date to an integer number, we need to determine if the added information from the data that already exists as a date is important enough to store as translating an integer would not give us any new information regarding month, day or time. The added data wouldn't be of much use considering that the reports are yearly based, so even if there are multiple reports from the same organizations there wouldn't be multiple of the same target focus during one year. We can therefore discard the information of month, day and time and push only the year to our database.

In the dataset *RiskAndVul* we can see the column C40 being of type boolean whereas in other datasets with the column C40 being a string which can be empty. Since the string data are always empty or contain a string of "C40" then we can translate the strings to a boolean where an empty string equals false and a string of "C40" equals true.

Ingest script overview:

Here we will go over our scripts which we will be used to load the data of the datasets to our database. We will present a quick overview in sequential order of operations from start to end of our script. The order or execution is important as during it non-sequentially will result in errors when uploading to our sql database. The error occurs due to foreign primary keys, where "City "have a dependency on "Country" and nearly every other table depends on the table "City".

- **Country:**

Upload the combined overlapping data from all five datasets that revolve specifically around countries.

- **Organizations and cities:**

Upload the data of the organizations and the cities to the database from all five datasets.

- **Population, Temperature, Gases, GDP:**

Upload the data of population, temperature, gases to their relevant tables.

- **Reduc16 and Reduc17:**

Upload the data specific to the datasets Reduc16 and Reduc17 reports.

- **GHG:**

Upload the data specific to the datasets GHG reports.

- **Comm:**

Upload the data specific to the datasets Comm reports.

- **RiskAndVul:**

Upload the data specific to the datasets RiskAndVul reports.

Design and develop operations for maintenance of the database.

Considering that the type of data we will be storing within our database is not something that should change over time after it's first upload to our database and that any reports stored shouldn't be deleted unless there is a specific call to it, say if an organization is remove or replace due to politics within a country or city then it would make much sense to spend time or resources within our database around constantly be checking the data within each table for value changes. Instead should any data need changing or deletion then a person should be overseeing the process.
Therefore we believe that time and resources would be better spent optimizing the speed for the request it would receive during its up time.

Indexing:

Considering the data from the datasets is mainly focused on where in the world it is coming from a city and country focus, we see a lot of data based on string values, primarily names. We can expect that a lot of data requests that our database would receive would be based on these names, so setting up indexing based on the different names of the organizations, cities, countries and even the account numbers, though they aren't strings, would speed up the request time.
The same can also be done on each of the different report tables which would often accompany the cities. Here it would mainly be on the account numbers which could be grouped based on their account number to speed up the gathering of report data for requests.

We now have to consider when we will maintain the indexing within our database with respect to when new data could be uploaded and when it is uploaded. Considering the type of data within the datasets we can see that most cities uploaded yearly reports, with a few uploading more than once per year if the area of data gathering were different. Then we would consider the number of cities and countries that appear to be part of these reports. Overall we believe that indexing when new data is uploaded would be the best solution as the number of uploads is relatively few in comparison to how many requests there could appear should the database be used in a public scenario.

Formulate ten relevant questions for extracting information from the database, design and develop database functionality for implementing the information extraction.

1. All C40/GCOM city:

One question which may be often requested from the database is which of the many cities stored within our database is both labeled as being a C40 city and part of the GCoM.

SELECT * FROM mydb.city WHERE c40 = 1 && gcom = 1;

```
1 •    SELECT * FROM mydb.city WHERE c40 = 1 && gcom = 1;
```

2. Decrease/Increase by city:

SELECT
c.city_name, ghg.increase_or_decrease_last_year as ghg_increase_decrease,
cce.increase_or_decrease_last_year as cce_increase_decrease
FROM city c
JOIN ghg_emissions ghg ON ghg.city_id = c.account_no
JOIN cites_community_wide_emissions cce ON cce.city_id = c.account_no
WHERE ghg.increase_or_decrease_last_year IS NOT NULL AND
cce.increase_or_decrease_last_year IS NOT NULL;

```
SELECT c.city_name, ghg.increase_or_decrease_last_year as ghg_increase_decrease, cce.increase_or_decrease_last_year as cce_increase_dec
JOIN ghg_emissions ghg ON ghg.city_id = c.account_no
JOIN cites_community_wide_emissions cce ON cce.city_id = c.account_no
WHERE ghg.increase_or_decrease_last_year IS NOT NULL AND cce.increase_or_decrease_last_year IS NOT NULL
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

city_name	ghg_increase_decrease	cce_increase_decrease
Ayuntamiento de Madrid	Decreased	Decreased
New York City	Increased	Decreased
Ville de Montreal	Stayed the same	Decreased
City of Burlington	Decreased	Decreased
Wellington City Council	Decreased	Decreased

3. Percentage reduction target over ~80%:

```
SELECT city_id, city_name, city_short_name, percentage_reduction_target
from mydb.emissions_reduction
left join mydb.city
    on mydb.city.account_no = mydb.emissions_reduction.city_id
where mydb.emissions_reduction.percentage_reduction_target > 80
```

city_id	city_name	city_short_name	percentage_reduction_target
54408	Aarhus Kommune	Aarhus Kommune	100
50154	City of Turku	Turku	100
3429	City of Stockholm	Stockholm	100
58489	Hoeje-Taastrup Kommune	Hoeje-Taastrup Kommune	98
54443	Landeshauptstadt Magdeburg	Landeshauptstadt Magdeburg	95
54443	Landeshauptstadt Magdeburg	Landeshauptstadt Magdeburg	86
58488	Sonderborg Kommune	Sonderborg Kommune	100
35449	Stadt Zürich	Stadt Zürich	82
14088	City of Oslo	Oslo	95
43930	The Hague	The Hague	100
31151	Basel-Stadt	Basel-Stadt	100
31009	City of Copenhagen	Copenhagen	100

4. Percentage reduction target under ~20% including comment:

SELECT emissions_reduction.city_id, city_name, city_short_name, percentage_reduction_target, emissions_reduction.comment
from emissions_reduction
left join city
    on city.account_no = emissions_reduction.city_id
where emissions_reduction.percentage_reduction_target < 20;

```
20 •    SELECT emissions_reduction.city_id, city_name,
21      city_short_name, percentage_reduction_target, emissions_reduction.comment
22      from emissions_reduction
23      left join city
24          on city.account_no = emissions_reduction.city_id
25      where emissions_reduction.comment IS NOT NULL AND
26      emissions_reduction.percentage_reduction_target < 20;
27
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

city_id	city_name	city_short_name	percentage_reduction_target	comment
62855	Egedal Municipality	Egedal Municipality	7	The municipality are increasing in terms of citize...
58537	Tarnów	Tarnów	8.46	Data based on plan mentioned in 7.1. a - row 3
35858	City of Cape Town	Cape Town	13	The targets are set s in the City's Energy and Cl...
36254	Comune di Venezia	Venezia	13.1	SEAP action "free-14 [1/2] - extimated CO2 red...
31110	Roma Capitale	Roma	0	Emission for tertiary are forecasted increasing o...

5. Baseline emission and percentage reduction target by sector:

```
2 •  SELECT city_id, city_name, baseline_emissions, baseline_year, target_date, sector, percentage_reduction_target, comment
3    from mydb.emissions_reduction
4    join mydb.city on mydb.city.account_no = mydb.emissions_reduction.city_id
5    where mydb.emissions_reduction.sector = 'transport'
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

city_id	city_name	baseline_emissions	baseline_year	target_date	sector	percentage_reduction_target	comment
63616	Abasan Al-Kabira Municipality	6893	2010	2020	Transport	6	NULL
1499	Ajuntament de Barcelona	1540	2007	2020	Transport	27	On road transportation, port and airport
54637	Alcaldía de Cuenca	11000	2017	NULL	Transport	NULL	Tranvía, ciclovías, sensibilización sobre el uso d...
31171	Ayuntamiento de Madrid	3162	2012	2030	Transport	50	NULL
60588	City of Alba-Iulia	44216	2008	2020	Transport	23	NULL
1093	City of Atlanta	27266	2009	2020	Transport	20	Sharing mobility strategies, TOD, and parking st...
36159	City of Lisbon	1567000	2002	2030	Transport	40	Target of 40% based on total city wide emissio...
50551	City of Long Beach	NULL	2007	2020	Transport	10	NULL
35877	City of Pittsburgh	NULL	2003	2030	Transport	50	NULL
58569	City of Podgorica	299	2008	2020	Transport	20	NULL
31113	City of Yokohama	4340000	2005	2020	Transport	50	NULL
59180	Middelfart Kommune	104276	2010	2025	Transport	NULL	NULL
60002	Municipality of Cainta	123940	2015	2016	Transport	20	NULL
56276	New Taipei City Government	4807480	2005	2030	Transport	25	NULL

6. Cities with increase in emission - including country, gdp, population, comments etc:

```
SELECT
c.country_id, c.city_name,
p.population,
gd.gdp,
cce.increase_or_decrease_last_year as increase_decrease,
cce.comment
FROM city c
JOIN gdp gd ON gd.id_city = c.account_no
JOIN cites_community_wide_emissions cce ON cce.city_id = c.account_no
JOIN population p ON c.account_no = p.id_city
JOIN gdp gdp ON gdp.id_city = c.account_no
WHERE cce.increase_or_decrease_last_year = "Increased"
AND cce.comment IS NOT NULL
;
```

```
5  SELECT
6      c.country_id, c.city_name,
7      p.population,
8      gd.gdp,
9      cce.increase_or_decrease_last_year as increase_decrease,
10     cce.comment
11     FROM city c
12     JOIN gdp gd ON gd.id_city = c.account_no
13     JOIN cites_community_wide_emissions cce ON cce.city_id = c.account_no
14     JOIN population p ON c.account_no = p.id_city
15     JOIN gdp gdp ON gdp.id_city = c.account_no
16     WHERE cce.increase_or_decrease_last_year = "Increased"
17     AND cce.comment IS NOT NULL
18     ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

country_id	city_name	population	gdp	increase_decrease	comment
Hong Kong	Government of ...	7346100	320679000000	Increased	95% level of confidence
Hong Kong	Government of ...	7346100	320679000000	Increased	95% level of confidence
USA	City of Benicia	27450	23000000	Increased	A third party data anomaly in the 2010 inventor...
Indonesia	Bogor City Gov...	1030720	117412000000000	Increased	Finished
Indonesia	Bogor City Gov...	1047922	117412000000000	Increased	Finished

Design and implement a model for scaling the database, considering ACID and/or CAP theorem rules.

ACID:

Atomicity:
In the future when new reports could be added to the database we need to ensure that all relevant data is uploaded in the correct order so that all our primary key relationships are maintained. When new reports are to be uploaded a transaction should be made where in any countries not already in our database is set to be uploaded first, secondly any city or organization not already in our database should be uploaded and then when both country and city is set up then the report can be uploaded.

Consistency:
We are limited to how much we can enforce a consistency with the data within our database since many of the columns in the different datasets either allow for null values or have a few values which completely changes how the data can be stored. At most the only consistency we can and must enforce is the need for any report that would be uploaded to include an account number of type integer and a string representing the name of a country.

Isolation:
Due to the need for countries and account numbers to be unique and with the expected low number of uploads within a short time frame, we should make each upload happen in order instead of at the same time. This will help insure the database should a new organization upload multiple reports at the same time which may result in errors if two or more attempted to upload a new unique account number to the city table.

Durability:
To ensure the data within our database is safe incase of things like crashes we have our replicated database which should contain a full copy of the data within our database.

CAP:

Consistency:
We ensure data consistency by only having one active database at one time with a replicated database ready with all the data from the active database. Should the active database fail, the replicated database will be ready to take its place.

Availability:
Should our database for whatever reason be unavailable then an api which all the requests should go through would return the request with an exception message.

Partition tolerance:
Since our database would be set up on multiple separate partitions using replication partitions, our database would have a certain tolerance should one of the partitions fail for whatever reason.

Implementation:

In our database we, as previously stated under formulation, decided to use transactional replication wherein we have set up a publisher/distributor server together with three subscriber servers.

Firstly this ensures our primary focus with the data from the datasets which is to ensure that the data is stored correctly and secondly that it is available to everyone who makes requests to it.

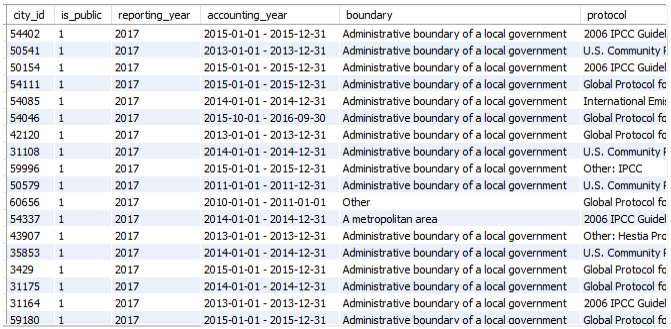
Validate and test all database operations.

Check data using Select:

Here we would go over the different tables within our database and manually create different select commands in a query to check whether or not the data has been correctly uploaded to our database. Knowing whether or not the data from the five datasets have been properly uploaded to the database is something we can already see to some extent when uploading the data as value type mismatch, among other things, is something that we would be quick to discover when creating the upload scripts.

Example:

```
select * from mydb.country;
select * from mydb.city;
select * from mydb.population;
select * from mydb.temperature;
select * from mydb.gdp;
select * from mydb.gases;
select * from mydb.emissions_reduction;
select * from mydb.cites_community_wide_emissions;
select * from mydb.ghg_emissions;
select * from mydb.2023_risk_and_vulnerability;
```



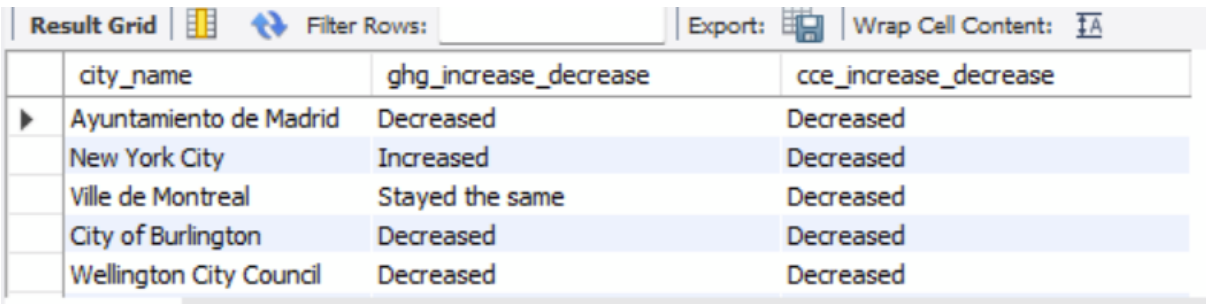
city_id	is_public	reporting_year	accounting_year	boundary	protocol
54402	1	2017	2015-01-01 - 2015-12-31	Administrative boundary of a local government	2006 IPCC Guidel
50541	1	2017	2013-01-01 - 2013-12-31	Administrative boundary of a local government	U.S. Community F
50154	1	2017	2015-01-01 - 2015-12-31	Administrative boundary of a local government	2006 IPCC Guidel
54111	1	2017	2015-01-01 - 2015-12-31	Administrative boundary of a local government	Global Protocol fo
54085	1	2017	2014-01-01 - 2014-12-31	Administrative boundary of a local government	International Emi
54046	1	2017	2015-10-01 - 2016-09-30	Administrative boundary of a local government	Global Protocol fo
42120	1	2017	2013-01-01 - 2013-12-31	Administrative boundary of a local government	Global Protocol fo
31108	1	2017	2014-01-01 - 2014-12-31	Administrative boundary of a local government	U.S. Community F
59996	1	2017	2015-01-01 - 2015-12-31	Administrative boundary of a local government	Other: IPCC
50579	1	2017	2011-01-01 - 2011-12-31	Administrative boundary of a local government	U.S. Community F
60656	1	2017	2010-01-01 - 2011-01-01	Other	Global Protocol fo
54337	1	2017	2014-01-01 - 2014-12-31	A metropolitan area	2006 IPCC Guidel
43907	1	2017	2013-01-01 - 2013-12-31	Administrative boundary of a local government	Other: Hestia Pro
35853	1	2017	2014-01-01 - 2014-12-31	Administrative boundary of a local government	U.S. Community F
3429	1	2017	2015-01-01 - 2015-12-31	Administrative boundary of a local government	Global Protocol fo
31175	1	2017	2014-01-01 - 2014-12-31	Administrative boundary of a local government	Global Protocol fo
31164	1	2017	2013-01-01 - 2013-12-31	Administrative boundary of a local government	2006 IPCC Guidel
59180	1	2017	2015-01-01 - 2015-12-31	Administrative boundary of a local government	Global Protocol fo

Check key constraints using Join:

One thing that would not be checked during the upload process is the ways we can connect the different data to each other from different tables based on a chain of relationship constraints. Joining together different tables may be used when comparing the data from different types of reports to gather a greater picture of the world.
To check the constraints and how they may be used, we can set up some examples as to how some of these requests may look like.

Example:

```
SELECT c.city_name, ghg.increase_or_decrease_last_year as ghg_increase_decrease, cce.increase_or_decrease_last_year as cce_increase_dec
JOIN ghg_emissions ghg ON ghg.city_id = c.account_no
JOIN cites_community_wide_emissions cce ON cce.city_id = c.account_no
WHERE ghg.increase_or_decrease_last_year IS NOT NULL AND cce.increase_or_decrease_last_year IS NOT NULL
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

city_name	ghg_increase_decrease	cce_increase_decrease
Ayuntamiento de Madrid	Decreased	Decreased
New York City	Increased	Decreased
Ville de Montreal	Stayed the same	Decreased
City of Burlington	Decreased	Decreased
Wellington City Council	Decreased	Decreased

Evaluate the database's performance and suggest measures for improving it.

Profiler:

To test the performance of our database we can use our previous ten relevant questions we made. They each contain a SQL query which we then can use the built in profiler to get the individual request time for each request. For this we used MySQL workbench built in profiller which shows the duration time for a request in seconds but for our purpose we would be better with recalculating it to milliseconds. For the queries we made for the ten relevant questions we end with a duration time in milliseconds as such:

Query:	Time:
select * from city where c40 = 1 && gcom = 1;	0,60875
SELECT c.city_name, ghg.increase_or_decrease_last_year as ghg_increase_decrease, cce.increase_or_decrease_last_year as cce_increase_decrease FROM city c JOIN ghg_emissions ghg ON ghg.city_id = c.account_no JOIN cites_community_wide_emissions cce ON cce.city_id = c.account_no WHERE ghg.increase_or_decrease_last_year IS NOT NULL AND cce.increase_or_decrease_last_year IS NOT NULL;	1,137
Select city_id, city_name, percentage_reduction_target from emissions_reduction left join city on city.account_no = emissions_reduction.city_id where emissions_reduction.percentage_reduction_target > 80;	0,52125
SELECT emissions_reduction.city_id, city_name, city_short_name, percentage_reduction_target, emissions_reduction.comment from emissions_reduction left join city on city.account_no = emissions_reduction.city_id WHERE emissions_reduction.percentage_reduction_target < 20;	0,85625
SELECT city_id, city_name, baseline_emissions, baseline_year, target_date, sector, percentage_reduction_target, comment FROM emissions_reduction JOIN city on city.account_no = emissions_reduction.city_id WHERE emissions_reduction.sector = 'transport';	0,661
SELECT c.country_id, c.city_name, p.population, gd.gdp, cce.increase_or_decrease_last_year as increase_decrease, cce.comment FROM city c JOIN gdp gd ON gd.id_city = c.account_no JOIN cites_community_wide_emissions cce ON cce.city_id = c.account_no JOIN population p ON c.account_no = p.id_city JOIN gdp gdp ON gdp.id_city = c.account_no WHERE cce.increase_or_decrease_last_year = "Increased";	1,25925

Improvements:

In a real world scenario where in our database could see regular daily use, it would be an improvement to introduce caching for data, so that constantly access data requested by the same means, like a static hard coded sql string within a front end application, would not need to constantly use resources on the database for a search that would inevitably end with the same data results that the previous same request returned. Caching data would decrease the load on our database while also decreasing the wait time for the user. To cache the result data from the requests, we could use Redis which has the built-in functionality of storing results on a specific timer.

Formulate conclusions and recommendations.

We have gone through the five datasets given to us and from those have explored the data they contained, had discussions and made our conclusions as to how we best could store the data within a database. From our conclusions regarding the datasets have we discussed our options for setting up a database where our conclusion is the result of our work, with a EER diagram of the structure of the database which we used to setup a local instance and created scripts for uploading the data from the five datasets into the database instance. From there we have made discussions, evaluations and conclusions for future improvement recommendations that could be made to our database as well as some improvements which we have implemented into our database.