# CSC 343 Phase 2 Report

Connor Burns and Ekagra Luthra
burnsco2, luthraek

## Design Decisions:

### Feedback from TA:

The only negative feedback we got for Phase 1 was related to our investigative questions and not our design, which does not really affect this Phase, as this phase pertains mostly to implementing our schema and the design decisions we make along the way.

### Addition of foreign key constraints:

Although we did not remove or change anything already in our schema, we decided to add to it, specifically new foreign key constraints. We chose for the keys of the emissions table and the pollution deaths by location table to reference the key of population, as this intuitively made sense to us. We want to have population data for everything in our schema, so this step ensures that this is true for the emissions and pollution deaths by location table. The data in our deaths-rates-from-air-pollution.csv file only states deaths per 100,000, meaning that in order to get the exact number of deaths in a country in a given year due to air pollution we would NEED that country's population in that year. This foregin key is not really applicable to the pollution deaths by sdi table since that does not involve location, and rather involves the sdi rank.

### Separation of SDI and location from death-rates-from-air-pollution.csv:

The death-rates-from-air-pollution.csv file contained two types of entities in the leftmost column: locations and (Socio-demographic Index) SDI ranks. This means that some of the entries will say 'Low SDI' or 'Low-medium SDI' (or some variation of this) and others will have a country name. We decided to separate this information into two separate tables, as this would allow us to separately view differences in death rates geographically, as well as socio-demographically (i.e we are comparing low SDIs with high SDIs, and countries with other countries, rather than comparing low SDIs with some country - which tells us very little).

### Removal of 'code' attribute in final tables:

Our co2-emissions.csv and death-rates-from-air-pollution.csv files contained a 'code' attribute, where the entries are country codes. In our final (non-temporary) tables, we made sure that this column does not exist because we have no use for it, as country names serve as an easier to understand alternative to country codes. Further, only 2 out of our 3 csv files contain country codes, whereas all 3 contain country names. On top of this, some locations do not have country codes, so that would be a null entry, which is not useful to us.

## Cleaning Process:

### Temporary tables:

The first step in importing the data to clean, was to create three temporary tables, that take the exact structure of the three csv files which we import. After doing this, we use the temporary tables to run queries which clean the data, and then project the attributes which we want in the final tables.

### Removal of top row of csv files:

We decided to remove the top row of all of our CSV files because the \copy command would have read that first line and thrown an error as the titles are all strings while the entries in the cells under that title might be a float or an int.

### Removal of duplicate row:

The csv file containing the population of each location for various years had a few duplicate rows. In order to remove these duplicate rows (which violated our key constraint), we selected distinct location, year and grouped by all attributes from the temporary table which we imported the population data into:

```
select distinct location, year, totalPopulation from temp_pop where
variant='Medium' group by location, year, temp_pop.totalpopulation;
```

### Removal of rows which violate the new foreign key constraints:

In order to uphold the new foreign key constraints, we chose to join the temp emissions table with the population table on location and year, so that we could project the emissions data into the final table of rows which have a corresponding row in population. We did the same thing for the pollution deaths by location table. After doing this, we can insert the data inside of these views into the final tables without violating the new foreign key constraints:

```
    CREATE VIEW ValidEmissions as SELECT temp_emissions.entity,
    temp_emissions.year, temp_emissions.annualEmissions FROM
    temp_emissions JOIN population on temp_emissions.entity =
    population.location
    AND temp_emissions.year = population.year;
    CREATE VIEW ValidPollutionDeaths as SELECT temp_pollution_deaths.entity,
    temp_pollution_deaths.year,
    temp_pollution_deaths.airpollutiondeathsper100k, outdoor, indoor, ozone
    FROM
```

```
      temp_pollution_deaths JOIN population on temp_pollution_deaths.entity =
      population.location
      AND temp_pollution_deaths.year = population.year;
```

Note we project entity, year, airPollutionDeathsPer100K, indoor, outdoor, ozone when we insert into the final tables.

In order to insert into pollution deaths by sdi, we insert the data from the temporary pollution deaths table, where the entity is equal to an SDI rank:

```
insert into PollutionDeathsBySDI select entity, year, airPollutionDeathsPer100K,
indoor, outdoor, ozone from temp_pollution_deaths WHERE entity='Low SDI' OR
entity='Low-middle SDI' OR entity='Middle SDI' OR entity='Middle SDI' OR
entity='High-middle SDI' OR entity='High SDI';
```