*Додаток 1*

Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки Кафедра інформатики та програмної інженерії

Звіт

з комп’ютерного практикуму № 1 з дисципліни

«Аналіз даних в інформаційних системах»

на тему: «Створення сховища даних»

Виконав студент ІП-13, Бабашев Олексій Дмитрович

(шифр, прізвище, ім'я, по батькові)

Перевірив Олійник Юрій Олександрович

( прізвище, ім'я, по батькові)

Київ 202 3

Комп’ютерний практикум 1

Тема – створення сховища даних.

Мета – ознайомитись з підходами до створення сховищ даних.

Для виконання даної лабораторної роботи було вибрано даний набір даних:

https://www.kaggle.com/russellyates88/suicide-rates-overview-1985-to-2016 https://www.kaggle.com/datasets/fernandol/countries-of-the-world https://www.kaggle.com/datasets/alcidesoxa/world-happiness-report-2005-2018 https://www.kaggle.com/datasets/madhurpant/world-deaths-and-causes-1990-2019

Даний набір містить інформацію про випадки смертей в країнах світу за певний період часу. Також цей набір даних містить статистичну інформацію про різні аспекти здоров'я, економіки та соціальних показників різних країн світу.

Таблиця annual\_deaths\_by\_causes містить дані про випадки та причини смертей в країнах за певний період часу.

Таблиця suicide rates1985-2016 містить інформацію про випадки суїцидів в країнах світу за різний період часу.

Таблиця countries of the world.csv містить інформацію про країни світу, їх населення, територію і тд.

Таблиця world-happiness-report-2005-2018 містить дані про умови проживання в країнах світу, індекс демократії, тривалість життя та інше.

Форматування даних

Для форматування даних використаємо python скрипт наведений нижче:

import pandas as pd

#DEATHES

df = pd.read\_csv('../dataset/annual\_deaths\_by\_causes.csv')

#Cleaning from extra spaces in columns and data Replace NaN with 0

df.columns = df.columns.str.strip()

df = df.apply(lambda x:x.str.strip() if x.dtype == "object" else x)

df.fillna(0, inplace=True)

#delete extra columns

df.drop('code',axis=1, inplace=True)

#add column sum of deaths

df['total'] = df.iloc[:,2:].sum(axis=1)

# Write the modified DataFrame to a new CSV file

df.to\_csv('C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/cleaned-dataset/deaths1990-2019.csv', index=False)

#%%

#SUICIDES

df = pd.read\_csv('../dataset/suicide rates1985-2016.csv')

#Cleaning from extra spaces in columns and data Replace NaN with 0

df.columns = df.columns.str.strip()

df = df.apply(lambda x:x.str.strip() if x.dtype == "object" else x)

df.fillna(0, inplace=True)

# group by 'country', 'year', 'gdp\_per\_capita' columns and aggregate 'suicides\_no' and 'population' columns

df = df.groupby(['country','year','gdp\_per\_capita ($)'], as\_index=False).agg({'suicides\_no': 'sum', 'population': 'sum'})

#after this you do not have to delete extra columns!

df.to\_csv('C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/cleaned-dataset/suicides1985-2016GROUPED.csv', index=False)

#%%

#HAPPINESS

# Read the CSV file with ';' delimiter

df = pd.read\_csv('../dataset/world-happiness-report-2005-2018.csv', sep=';')

#Cleaning from extra spaces in columns and data Replace NaN with 0

df.columns = df.columns.str.strip()

df = df.apply(lambda x:x.str.strip() if x.dtype == "object" else x)

df.fillna(0, inplace=True)

#delete extra column

df.drop('Year',axis=1, inplace=True)

#making the same name of the same countries

stand\_name = {

'Congo (Brazzaville)':'Congo',

'Congo (Kinshasa)':'Congo'

}

df['Country name'] = df['Country name'].replace(stand\_name)

#group by country name and find average of other data of different years

df = df.groupby(['Country name'],as\_index=False).mean()

# Save the file with ',' delimiter

df.to\_csv('C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/cleaned-dataset/happiness.csv', sep=',', index=False)

#%%

#COUNTRIES

df = pd.read\_csv('../dataset/countries of the world.csv')

#Cleaning from extra spaces in columns and data Replace NaN with 0

df.columns = df.columns.str.strip()

df = df.apply(lambda x:x.str.strip() if x.dtype == "object" else x)

df.fillna(0, inplace=True)

# replace commas with periods in the numeric columns

df = df.replace(',', '.', regex=True)

mapping = {

'ASIA (EX. NEAR EAST)': 'ASIA',

'EASTERN EUROPE': 'EUROPE',

'NORTHERN AFRICA': 'AFRICA',

'WESTERN EUROPE': 'EUROPE',

'SUB-SAHARAN AFRICA': 'AFRICA',

'LATIN AMER. & CARIB': 'LATIN AMERICA',

'C.W. OF IND. STATES': 'ASIA',

'NEAR EAST': 'ASIA',

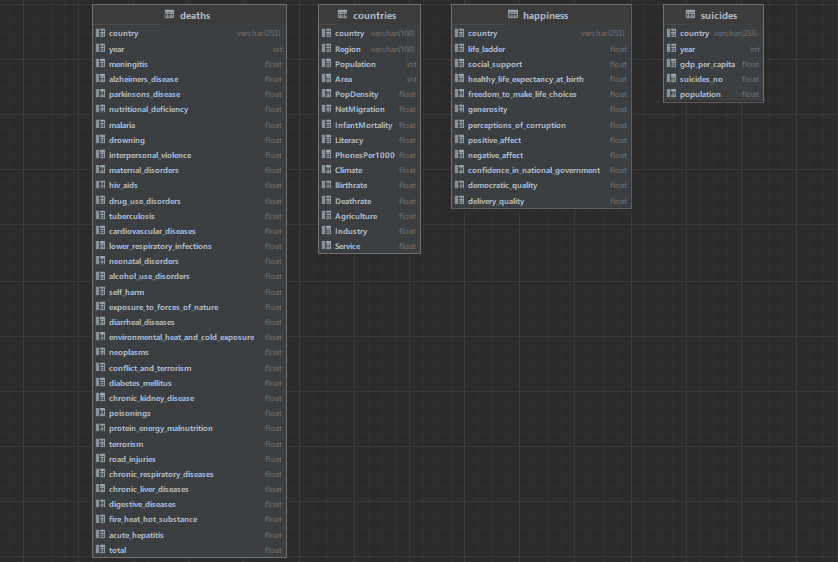
'BALTICS': 'EUROPE'

}

df['Region'] = df['Region'].replace(mapping)

df.to\_csv('C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/cleaned-dataset/countries.csv', index=False)

Stage зона



**Таблиця "deaths":**

"country" - назва країни, де сталися смерті

"year" - рік, коли сталися смерті

Колонки, що містять назви різних хвороб і причин смерті, такі як "meningitis", "alzheimers\_disease", "malaria" і т.д. Кожна з цих колонок містить кількість смертей, пов'язаних з відповідною хворобою або причиною.

"total" - сумарний показник усіх смертей за цей рік.

**Таблиця "suicides":**

"country" - назва країни, де сталися суїциди

"year" - рік, коли сталися суїциди

"gdp\_per\_capita" - ВВП на душу населення

"suicides\_no" - кількість суїцидів

"population" - загальна кількість населення

**Таблиця "happiness":**

"country" - назва країни

"life\_ladder" - загальний показник щастя за допомогою рейтингу

"social\_support" - рівень соціальної підтримки, яку люди отримують від своїх родин, друзів і інших людей

"healthy\_life\_expectancy\_at\_birth" - очікувана тривалість здорового життя при народженні

"freedom\_to\_make\_life\_choices" - рівень свободи людини в прийнятті власних життєвих рішень

"generosity" - рівень щедрості відповідної країни

"perceptions\_of\_corruption" - сприйняття корупції відповідної країни

"positive\_affect" - рівень позитивних емоцій у громадян країни

"negative\_affect" - рівень негативних емоцій у громадян країни

"confidence\_in\_national\_government" - рівень довіри національному уряду

"democratic\_quality" - якість демократії відповідної країни

"delivery\_quality" - якість державних послуг відповідної країни

**Таблиця "countries":**

"country" - назва країни

"Region" - регіон, до якого належить країна

"Population

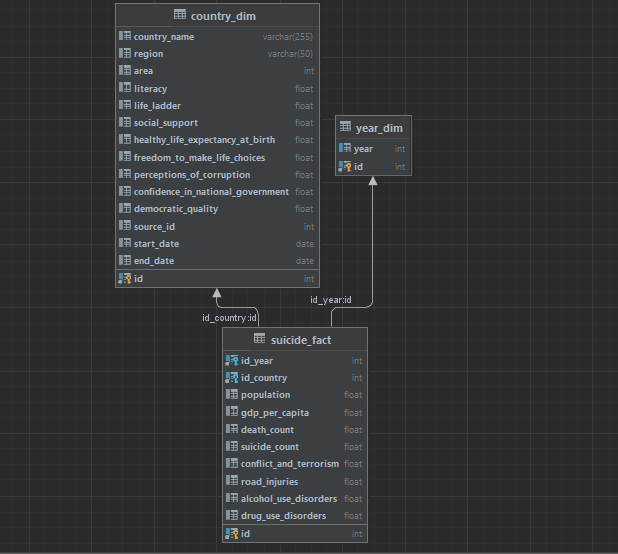
MySQL скрипти для створення stage:

drop database if exists stage;  
  
create database stage;  
  
use stage;  
  
drop table if exists suicides;  
drop table if exists deaths;  
drop table if exists happiness;  
  
  
CREATE TABLE deaths  
(  
 country VARCHAR(255),  
 `year` INT,  
 meningitis FLOAT,  
 alzheimers\_disease FLOAT,  
 parkinsons\_disease FLOAT,  
 nutritional\_deficiency FLOAT,  
 malaria FLOAT,  
 drowning FLOAT,  
 interpersonal\_violence FLOAT,  
 maternal\_disorders FLOAT,  
 hiv\_aids FLOAT,  
 drug\_use\_disorders FLOAT,  
 tuberculosis FLOAT,  
 cardiovascular\_diseases FLOAT,  
 lower\_respiratory\_infections FLOAT,  
 neonatal\_disorders FLOAT,  
 alcohol\_use\_disorders FLOAT,  
 self\_harm FLOAT,  
 exposure\_to\_forces\_of\_nature FLOAT,  
 diarrheal\_diseases FLOAT,  
 environmental\_heat\_and\_cold\_exposure FLOAT,  
 neoplasms FLOAT,  
 conflict\_and\_terrorism FLOAT,  
 diabetes\_mellitus FLOAT,  
 chronic\_kidney\_disease FLOAT,  
 poisonings FLOAT,  
 protein\_energy\_malnutrition FLOAT,  
 terrorism FLOAT,  
 road\_injuries FLOAT,  
 chronic\_respiratory\_diseases FLOAT,  
 chronic\_liver\_diseases FLOAT,  
 digestive\_diseases FLOAT,  
 fire\_heat\_hot\_substance FLOAT,  
 acute\_hepatitis FLOAT,  
 total FLOAT  
);  
  
  
CREATE TABLE suicides  
(  
 country VARCHAR(255),  
 `year` INT,  
 gdp\_per\_capita FLOAT,  
 suicides\_no FLOAT,  
 population FLOAT  
);  
  
  
CREATE TABLE happiness  
(  
 country VARCHAR(255),  
 life\_ladder FLOAT,  
 social\_support FLOAT,  
 healthy\_life\_expectancy\_at\_birth FLOAT,  
 freedom\_to\_make\_life\_choices FLOAT,  
 generosity FLOAT,  
 perceptions\_of\_corruption FLOAT,  
 positive\_affect FLOAT,  
 negative\_affect FLOAT,  
 confidence\_in\_national\_government FLOAT,  
 democratic\_quality FLOAT,  
 delivery\_quality FLOAT  
);  
  
CREATE TABLE countries (  
 country VARCHAR(100),  
 Region VARCHAR(100),  
 Population INT,  
 Area INT,  
 PopDensity FLOAT,  
 NetMigration FLOAT,  
 InfantMortality FLOAT,  
 Literacy FLOAT,  
 PhonesPer1000 FLOAT,  
 Climate FLOAT,  
 Birthrate FLOAT,  
 Deathrate FLOAT,  
 Agriculture FLOAT,  
 Industry FLOAT,  
 Service FLOAT  
);

MySQL скрипт для заповнення stage зони даними:

use stage;  
  
truncate table suicides;  
truncate table deaths;  
truncate table happiness;  
truncate table countries;  
  
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/cleaned-dataset/deaths1990-2019.csv' INTO TABLE deaths  
 FIELDS TERMINATED BY ','  
 ENCLOSED BY '"'  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS;  
  
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/cleaned-dataset/suicides1985-2016GROUPED.csv' INTO TABLE suicides  
 FIELDS TERMINATED BY ','  
 ENCLOSED BY '"'  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS;  
  
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/cleaned-dataset/happiness.csv' INTO TABLE happiness  
 FIELDS TERMINATED BY ','  
 ENCLOSED BY '"'  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS  
 (country, life\_ladder, @skip, social\_support, healthy\_life\_expectancy\_at\_birth, freedom\_to\_make\_life\_choices,  
 generosity, perceptions\_of\_corruption, positive\_affect, negative\_affect, confidence\_in\_national\_government,  
 democratic\_quality, delivery\_quality, @skip, @skip, @skip, @skip, @skip, @skip, @skip, @skip, @skip, @skip,  
 @skip, @skip);  
  
  
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/cleaned-dataset/countries.csv' INTO TABLE countries  
FIELDS TERMINATED BY ','  
ENCLOSED BY '"'  
LINES TERMINATED BY '\n'  
IGNORE 1 ROWS  
(country, Region, Population, Area, PopDensity, @skip, NetMigration, InfantMortality, @skip, Literacy, PhonesPer1000,  
 @skip, @skip, @skip, Climate, Birthrate, Deathrate, Agriculture, Industry, Service);

Main сховище



Фактова таблиця містить зовнішні ключі на таблиці виміри, що містять дані про країни та роки. Також фактова таблиця містить інформацію про кількість смертей, суїцидів, смертей на дорогах, смертей від алкоголю та наркотиків, смертей вніслідок конфліктів та тероризму за конкретний рік в конкретній країні.

MySQL скрипти для створення main warehouse:

drop database if exists warehouse;  
create database warehouse;  
  
use warehouse;  
  
drop table if exists country\_dim;  
drop table if exists year\_dim;  
drop table if exists suicide\_fact;  
  
CREATE TABLE country\_dim  
(  
 id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,  
 country\_name VARCHAR(255),  
 region VARCHAR(50),  
 area INT,  
 literacy FLOAT,  
 life\_ladder FLOAT,  
 social\_support FLOAT,  
 healthy\_life\_expectancy\_at\_birth FLOAT,  
 freedom\_to\_make\_life\_choices FLOAT,  
 perceptions\_of\_corruption FLOAT,  
 confidence\_in\_national\_government FLOAT,  
 democratic\_quality FLOAT,  
 source\_id int default null,  
 start\_date date default null,  
 end\_date date default null  
);  
  
CREATE TABLE year\_dim  
(  
 id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,  
 `year` INT UNIQUE  
);  
  
CREATE TABLE suicide\_fact  
(  
 id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,  
 id\_year INT NOT NULL,  
 id\_country INT NOT NULL,  
 population FLOAT,  
 gdp\_per\_capita FLOAT,  
 death\_count FLOAT,  
 suicide\_count FLOAT,  
 conflict\_and\_terrorism FLOAT,  
 road\_injuries FLOAT,  
 alcohol\_use\_disorders FLOAT,  
 drug\_use\_disorders FLOAT,  
  
 FOREIGN KEY (id\_year) references year\_dim (id),  
 FOREIGN KEY (id\_country) references country\_dim (id)  
);

MySQL скрипти для заповнення даними main warehouse:

use stage;  
  
insert into warehouse.year\_dim(`year`)  
select distinct deaths.year  
from deaths  
 join suicides on deaths.year = suicides.year;  
  
insert into warehouse.country\_dim(country\_name, region, area, literacy, life\_ladder, social\_support,  
 healthy\_life\_expectancy\_at\_birth, freedom\_to\_make\_life\_choices,  
 perceptions\_of\_corruption, confidence\_in\_national\_government, democratic\_quality)  
select distinct deaths.country,  
 countries.Region,  
 countries.Area,  
 countries.Literacy,  
 happiness.life\_ladder,  
 happiness.social\_support,  
 happiness.healthy\_life\_expectancy\_at\_birth,  
 happiness.freedom\_to\_make\_life\_choices,  
 happiness.perceptions\_of\_corruption,  
 happiness.confidence\_in\_national\_government,  
 happiness.democratic\_quality  
from deaths  
 join suicides on suicides.country = deaths.country  
 join happiness on happiness.country = deaths.country  
 join countries on countries.country = deaths.country;  
  
  
insert into warehouse.suicide\_fact(id\_year, id\_country, population, gdp\_per\_capita, death\_count,  
 suicide\_count, conflict\_and\_terrorism, road\_injuries,  
 alcohol\_use\_disorders, drug\_use\_disorders)  
select warehouse.year\_dim.id,  
 warehouse.country\_dim.id,  
 suicides.population,  
 suicides.gdp\_per\_capita,  
 deaths.total,  
 suicides.suicides\_no,  
 deaths.conflict\_and\_terrorism,  
 deaths.road\_injuries,  
 deaths.alcohol\_use\_disorders,  
 deaths.drug\_use\_disorders  
from suicides  
 join warehouse.country\_dim on suicides.country = warehouse.country\_dim.country\_name  
 join warehouse.year\_dim on suicides.year = warehouse.year\_dim.year  
 join deaths on deaths.country = suicides.country and deaths.year = suicides.year;

MySQL скрипти реалізації процедури, прикладу scd.

use warehouse;  
  
drop procedure if exists *slow\_change\_country*;  
  
delimiter //  
create procedure *slow\_change\_country*(old\_name varchar(50), new\_name varchar(50))  
begin  
 declare old\_id int default null;  
 declare old\_region VARCHAR(50);  
 declare old\_area INT;  
 declare old\_literacy FLOAT;  
 declare old\_life\_ladder FLOAT;  
 declare old\_social\_support FLOAT;  
 declare old\_healthy\_life\_expectancy\_at\_birth FLOAT;  
 declare old\_freedom\_to\_make\_life\_choices FLOAT;  
 declare old\_perceptions\_of\_corruption FLOAT;  
 declare old\_confidence\_in\_national\_government FLOAT;  
 declare old\_democratic\_quality FLOAT;  
  
  
 select id,  
 region,  
 area,  
 literacy,  
 life\_ladder,  
 social\_support,  
 healthy\_life\_expectancy\_at\_birth,  
 freedom\_to\_make\_life\_choices,  
 perceptions\_of\_corruption,  
 confidence\_in\_national\_government,  
 democratic\_quality  
 into old\_id,old\_region,old\_area,old\_literacy,old\_life\_ladder,old\_social\_support,old\_healthy\_life\_expectancy\_at\_birth,  
 old\_freedom\_to\_make\_life\_choices,old\_perceptions\_of\_corruption,old\_confidence\_in\_national\_government,old\_democratic\_quality  
 from country\_dim  
 where country\_name = old\_name;  
  
  
 if old\_name is null then  
 signal sqlstate '45000' set message\_text = 'Check country name. This country is not in database.';  
 else  
 insert into country\_dim (country\_name, region, area, literacy, life\_ladder, social\_support,  
 healthy\_life\_expectancy\_at\_birth, freedom\_to\_make\_life\_choices,  
 perceptions\_of\_corruption, confidence\_in\_national\_government, democratic\_quality,  
 source\_id, start\_date)  
 value (new\_name, old\_region, old\_area, old\_literacy, old\_life\_ladder, old\_social\_support,  
 old\_healthy\_life\_expectancy\_at\_birth,  
 old\_freedom\_to\_make\_life\_choices, old\_perceptions\_of\_corruption,  
 old\_confidence\_in\_national\_government,  
 old\_democratic\_quality, old\_id, *CURRENT\_DATE*);  
  
 update country\_dim  
 set end\_date = *CURRENT\_DATE* where id = old\_id;  
  
 end if;  
end //  
delimiter ;  
  
call *slow\_change\_country*('Albania', 'CHECK');  
call *slow\_change\_country*('CHECK', 'Albania');

**Додаткове завдання**

Повільно змінювані розміри (SCD) використовуються в сховищах даних для відстеження змін даних розмірів з часом. Є три поширені типи SCD:

* + SCD типу 1 перезаписує наявні дані у вимірі новими даними. Це означає, що історичні дані не зберігаються, а розмір містить лише найновішу інформацію. Цей тип SCD корисний, коли вам не потрібно відстежувати історичні зміни даних і потрібна лише найновіша інформація.
  + SCD типу 2 створює новий рядок у таблиці розмірів щоразу, коли відбувається зміна даних. Це означає, що історичні дані зберігаються, і ви можете відстежувати зміни в даних з часом. Цей тип SCD корисний, коли потрібно відстежувати історичні зміни даних і підтримувати повну історію розмірності.
  + SCD типу 3 додає нові стовпці до таблиці розмірів для відстеження змін даних. Це означає, що ви можете відстежувати деякі історичні зміни даних, але не всі. Цей тип SCD корисний, коли вам потрібно лише відстежувати обмежену кількість історичних даних і вам не потрібна повна історія виміру.

MySQL скрипт для завантаження нових даних до існуючих, incremental load:

use stage;  
  
INSERT INTO warehouse.year\_dim(`year`)  
SELECT DISTINCT deaths.year  
FROM deaths  
JOIN suicides ON deaths.year = suicides.year  
WHERE NOT *EXISTS* (  
 SELECT 1 FROM warehouse.year\_dim y  
 WHERE y.year = deaths.year  
);  
  
  
INSERT INTO warehouse.country\_dim(country\_name, region, area, literacy, life\_ladder, social\_support,  
 healthy\_life\_expectancy\_at\_birth, freedom\_to\_make\_life\_choices,  
 perceptions\_of\_corruption, confidence\_in\_national\_government, democratic\_quality)  
SELECT DISTINCT deaths.country,  
 countries.Region,  
 countries.Area,  
 countries.Literacy,  
 happiness.life\_ladder,  
 happiness.social\_support,  
 happiness.healthy\_life\_expectancy\_at\_birth,  
 happiness.freedom\_to\_make\_life\_choices,  
 happiness.perceptions\_of\_corruption,  
 happiness.confidence\_in\_national\_government,  
 happiness.democratic\_quality  
FROM deaths  
JOIN suicides ON suicides.country = deaths.country  
JOIN happiness ON happiness.country = deaths.country  
JOIN countries ON countries.country = deaths.country  
WHERE NOT *EXISTS* (  
 SELECT 1  
 FROM warehouse.country\_dim c  
 WHERE c.country\_name = deaths.country  
);  
  
INSERT INTO warehouse.suicide\_fact(id\_year, id\_country, population, gdp\_per\_capita, death\_count,  
 suicide\_count, conflict\_and\_terrorism, road\_injuries,  
 alcohol\_use\_disorders, drug\_use\_disorders)  
SELECT warehouse.year\_dim.id,  
 warehouse.country\_dim.id,  
 suicides.population,  
 suicides.gdp\_per\_capita,  
 deaths.total,  
 suicides.suicides\_no,  
 deaths.conflict\_and\_terrorism,  
 deaths.road\_injuries,  
 deaths.alcohol\_use\_disorders,  
 deaths.drug\_use\_disorders  
FROM suicides  
JOIN warehouse.country\_dim ON suicides.country = warehouse.country\_dim.country\_name  
JOIN warehouse.year\_dim ON suicides.year = warehouse.year\_dim.year  
JOIN deaths ON deaths.country = suicides.country AND deaths.year = suicides.year  
WHERE NOT *EXISTS* (  
 SELECT 1  
 FROM warehouse.suicide\_fact s  
 WHERE s.id\_year = warehouse.year\_dim.id  
 AND s.id\_country = warehouse.country\_dim.id  
);

Висновок

Ознайомився з можливістю проектування сховищ даних, проходячи етапи створення stage зони для завантаження даних та створення основного сховища для розподілення даних зі зв’язками між ними. Було реалізовано можливості slowly changing dimension та incremental load. Перед завантаження у stage дані були обробленні python скриптами за допомогою бібліотеки pandas.