

Опис фінального проєкту

1. Завантажте дані:

Створіть схему `pandemic` у базі даних за допомогою SQL-команди.

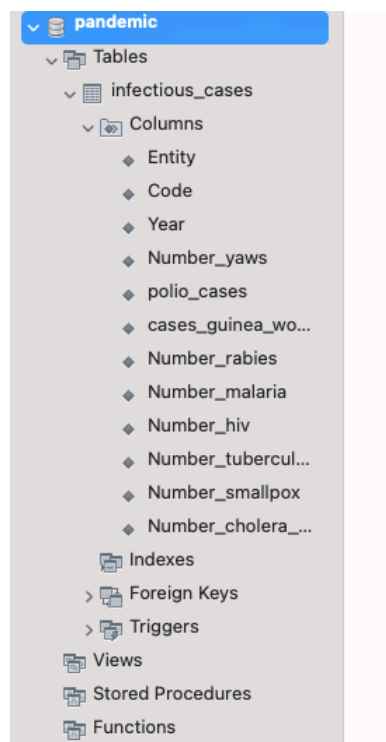
```
(CREATE SCHEMA IF NOT EXISTS pandemic;)
```

Оберіть її як схему за замовчуванням за допомогою SQL-команди.

```
(USE pandemic)
```

Імпортуйте [дані](#) за допомогою Import wizard так, як ви вже робили це у темі 3.

[infectious_cases.csv](#)



Продивіться дані, щоб бути у контексті.



Як бачите, атрибути `Entity` та `Code` постійно повторюються. Позбудьтесь цього за допомогою нормалізації даних.

2. Нормалізуйте таблицю infectious_cases. Збережіть у цій же схемі дві таблиці з нормалізованими даними.

```
CREATE TABLE IF NOT EXISTS entities (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    Entity VARCHAR(255),  
    Code VARCHAR(10)  
);
```

```
INSERT INTO entities (Entity, Code)  
SELECT DISTINCT Entity, Code  
FROM infectious_cases;
```

The screenshot shows a database management interface. The left sidebar contains a tree view of the database schema, with 'pandemic' selected. The main area displays the SQL queries and their execution results. The queries are:

```
1 CREATE TABLE IF NOT EXISTS entities (  
2     id INT AUTO_INCREMENT PRIMARY KEY,  
3     Entity VARCHAR(255),  
4     Code VARCHAR(10)  
5 );  
6  
7 INSERT INTO entities (Entity, Code)  
8 SELECT DISTINCT Entity, Code  
9 FROM infectious_cases;  
10  
11  
12
```

The execution results are shown in a table with columns: Time, Action, Response, and Duration / Fetch Time.

	Time	Action	Response	Duration / Fetch Time
1	21:11:50	CREATE TABLE IF NOT EXISTS entities (id INT AUTO_INCREMENT PRIMARY KEY, Entity VARCHAR(255), Code VARCHAR(10))	0 row(s) affected	0.018 sec
2	21:11:50	INSERT INTO entities (Entity, Code) SELECT DISTINCT Entity, Code FROM infectious_cases	195 row(s) affected Records: 195 Duplicates: 0 Warn...	0.014 sec
3	21:12:04	SELECT * FROM pandemic.entities LIMIT 0, 1000	195 row(s) returned	0.00053 sec / 0.0000...

The screenshot shows a database management interface. The left sidebar contains a tree view of the database schema, with 'pandemic' selected. The main area displays the SQL queries and their execution results. The queries are:

```
1 SELECT * FROM pandemic.entities;
```

The execution results are shown in a table with columns: Time, Action, Response, and Duration / Fetch Time.

	Time	Action	Response	Duration / Fetch Time
1	21:11:50	CREATE TABLE IF NOT EXISTS entities (id INT AUTO_INCREMENT PRIMARY KEY, Entity VARCHAR(255), Code VARCHAR(10))	0 row(s) affected	0.018 sec
2	21:11:50	INSERT INTO entities (Entity, Code) SELECT DISTINCT Entity, Code FROM infectious_cases	195 row(s) affected Records: 195 Duplicates: 0 Warn...	0.014 sec
3	21:12:04	SELECT * FROM pandemic.entities LIMIT 0, 1000	195 row(s) returned	0.00053 sec / 0.0000...

```

CREATE TABLE IF NOT EXISTS cases (
  id INT AUTO_INCREMENT PRIMARY KEY,
  entity_id INT,
  Year INT,
  Number_rabies INT,
  FOREIGN KEY (entity_id) REFERENCES entities(id)
);

INSERT INTO cases (entity_id, Year, Number_rabies)
SELECT e.id, ic.Year, COALESCE(NULLIF(ic.Number_rabies, ''), NULL)
FROM entities e
JOIN infectious_cases ic ON e.Entity = ic.Entity AND e.Code = ic.Code;

```

The screenshot displays a database management interface with two panels. The top panel shows the execution of a SQL script, and the bottom panel shows the resulting data grid.

Top Panel: SQL Query Execution

The SQL script executed is as follows:

```

1 CREATE TABLE IF NOT EXISTS cases (
2   id INT AUTO_INCREMENT PRIMARY KEY,
3   entity_id INT,
4   Year INT,
5   Number_rabies INT,
6   FOREIGN KEY (entity_id) REFERENCES entities(id)
7 );
8
9 INSERT INTO cases (entity_id, Year, Number_rabies)
10 SELECT e.id, ic.Year, COALESCE(NULLIF(ic.Number_rabies, ''), NULL)
11 FROM entities e
12 JOIN infectious_cases ic ON e.Entity = ic.Entity AND e.Code = ic.Code;

```

The execution results are shown in the Action Output table:

Time	Action	Response	Duration / Fetch Time
21:15:45	CREATE TABLE IF NOT EXISTS cases (id INT AUTO_INCREMENT PRIMARY KEY, entity_id INT, Year INT, Number_rabies INT, FORE...	0 row(s) affected	0.029 sec
21:15:45	INSERT INTO cases (entity_id, Year, Number_rabies) SELECT e.id, ic.Year, COALESCE(NULLIF(ic.Number_rabies, ''), NULL) FROM entities e JOI...	7271 row(s) affected Records: 7271 Duplicates: 0 Wa...	0.185 sec
21:15:53	SELECT * FROM pandemic.cases LIMIT 0, 1000	1000 row(s) returned	0.0010 sec / 0.00025...

Bottom Panel: Result Grid

The Result Grid shows the data returned by the final query:

id	entity_id	Year	Number_rabies
6	1	1985	0
7	1	1986	0
8	1	1987	0
9	1	1988	0
10	1	1989	0
11	1	1990	2
12	1	1991	2
13	1	1995	3

The bottom panel also includes a table for the Action Output, which is identical to the one in the top panel.

3. Проаналізуйте дані:

Для кожної унікальної комбінації `Entity` та `Code` або їх `id` порахуйте середнє (`AVG(Number_rabies) AS avg_rabies,`), мінімальне (`MIN(Number_rabies) AS min_rabies,`), максимальне значення (`MAX(Number_rabies) AS max_rabies,`) та суму (`SUM(Number_rabies) AS sum_rabies`) для атрибута `Number_rabies`.

💡 Врахуйте, що атрибут `Number_rabies` може містити порожні значення `''` — вам попередньо необхідно їх відфільтрувати. (`WHERE Number_rabies IS NOT NULL AND Number_rabies != ''`)

Результат відсортуйте за порахованим середнім значенням у порядку спадання. (`ORDER BY avg_rabies DESC`)

Оберіть тільки 10 рядків для виведення на екран. (`LIMIT 10;`)

```
SELECT Entity, Code,
       AVG(Number_rabies) AS avg_rabies,
       MIN(Number_rabies) AS min_rabies,
       MAX(Number_rabies) AS max_rabies,
       SUM(Number_rabies) AS sum_rabies
FROM pandemic.infectious_cases
WHERE Number_rabies IS NOT NULL AND Number_rabies != ''
GROUP BY Entity, Code
ORDER BY avg_rabies DESC
LIMIT 10;
```

The screenshot shows a database management tool interface. On the left is a 'SCHEMAS' sidebar with a tree view containing folders like 'hw_3', 'Tables', 'Views', 'Stored Procedures', 'Functions', 'LibraryManagement', 'mydb', 'pandemic', and 'sys'. The 'pandemic' folder is expanded, showing 'cases', 'entities', and 'infectious_cases'. The main area displays a SQL query in a text editor, which matches the query provided in the text. Below the editor is a 'Result Grid' showing the results of the query. The grid has columns: Entity, Code, avg_rabies, min_rabies, max_rabies, and sum_rabies. It displays 10 rows of data for various countries. At the bottom, there is an 'Action Output' section showing the execution details of the query.

Entity	Code	avg_rabies	min_rabies	max_rabies	sum_rabies
Pakistan	PAK	1582.1696266666667	1177.1449	1881.7257	47465.08879999999
Nigeria	NGA	1335.4154866666667	1073.9458	1441.1188	40062.46460000001
China	CHN	1247.906733	1044.7965	977.9139	37437.20199
Ethiopia	ETH	1145.1725223333333	1008.1017	967.63824	34355.17567
Myanmar	MMR	972.9036866666668	1022.31464	992.23914	29187.110600000004
Bangladesh	BGD	785.7088230000002	1033.1205	911.6814	23571.264690000004
Nepal	NPL	661.2508643333334	1019.73895	997.43427	19837.525930000003
Philippines	PHL	618.6910056666668	358.0602	698.071	18560.730170000003

4. Побудуйте колонку різниці в роках.

Для оригінальної або нормованої таблиці для колонки `Year` побудуйте з використанням вбудованих SQL-функцій:

атрибут, що створює дату першого січня відповідного року, (`Year, CONCAT(Year, '-01-01')`)

💡 Наприклад, якщо атрибут містить значення '1996', то значення нового атрибута має бути '1996-01-01'.

атрибут, що дорівнює поточній даті(`CURDATE()`),

атрибут, що дорівнює різниці в роках двох вищезгаданих колонок. (`TIMESTAMPDIFF(YEAR, CONCAT(Year, '-01-01'), CURDATE())`)

💡 Перераховувати всі інші атрибути, такі як `Number_malaria`, не потрібно.

👉 Для пошуку необхідних вбудованих функцій вам може знадобитися матеріал до теми 7.

SELECT

```
Year, CONCAT(Year, '-01-01') AS start_of_year,  
CURDATE() AS current_date_column,  
TIMESTAMPDIFF(YEAR, CONCAT(Year, '-01-01'), CURDATE()) AS year_difference  
FROM infectious_cases;
```

Або

SELECT

```
Year, CONCAT(Year, '-01-01') AS start_of_year,  
CURDATE() AS current_date_column,  
YEAR(CURDATE()) - Year AS year_difference  
FROM infectious_cases;
```

The screenshot shows a database management interface with a left sidebar containing a tree view of database objects. The main area displays a SQL query and its results in a table format. The query is:

```
1 SELECT  
2   Year, CONCAT(Year, '-01-01') AS start_of_year,  
3   CURDATE() AS current_date_column,  
4   TIMESTAMPDIFF(YEAR, CONCAT(Year, '-01-01'), CURDATE()) AS year_difference  
5 FROM infectious_cases;
```


The results table has the following data:

Year	start_of_year	current_date_c...	year_difference
1982	1982-01-01	2024-04-16	42
1983	1983-01-01	2024-04-16	41
1984	1984-01-01	2024-04-16	40
1985	1985-01-01	2024-04-16	39
1986	1986-01-01	2024-04-16	38
1987	1987-01-01	2024-04-16	37
1988	1988-01-01	2024-04-16	36
1989	1989-01-01	2024-04-16	35

The interface also shows a 'Result Output' section at the bottom with a table of actions and their durations.

5. Побудуйте власну функцію.

Створіть і використайте функцію, що будує такий же атрибут, як і в попередньому завданні: функція має приймати на вхід значення року, а повертати різницю в роках між поточною датою та датою, створеною з атрибута року (1996 рік → '1996-01-01').

 Якщо ви не виконали попереднє завдання, то можете побудувати іншу функцію — функцію, що рахує кількість захворювань за певний період. Для цього треба поділити кількість захворювань на рік на певне число: 12 — для отримання середньої кількості захворювань на місяць, 4 — на квартал або 2 — на півріччя. Таким чином, функція буде приймати два параметри: кількість захворювань на рік та довільний дільник. Ви також маєте використати `IF` — запустити на даних. Оскільки не всі рядки містять число захворювань, вам необхідно буде відсіяти ті, що не мають чисельного значення ($\neq ''$).

```
DROP FUNCTION IF EXISTS CalculateYearDifference;  
DELIMITER //
```

```
CREATE FUNCTION CalculateYearDifference(year_input INT)  
RETURNS INT  
DETERMINISTIC  
NO SQL  
BEGIN  
    DECLARE start_of_year DATE;  
    DECLARE current_year DATE;  
  
    SET start_of_year = CONCAT(year_input, '-01-01');  
    SET current_year = CURDATE();  
  
    RETURN YEAR(current_year) - YEAR(start_of_year);  
END//
```

```
DELIMITER ;
```

```
SELECT  
    Year, CONCAT(Year, '-01-01') AS start_of_year,  
    CURDATE() AS current_date_column,  
    CalculateYearDifference(Year) AS year_difference  
FROM infectious_cases;
```

