

1. Завантажте дані:

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

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

Імпортуйте [дані](#)

```
CREATE DATABASE IF NOT EXISTS pandemic;
USE pandemic;
DROP TABLE IF EXISTS infectious_cases_original;
CREATE TABLE infectious_cases_original (
    Entity VARCHAR(255),
    Code VARCHAR(10),
    Year INT,
    Number_yaws DECIMAL(20, 8),
    polio_cases DECIMAL(20, 8),
    cases_guinea_worm DECIMAL(20, 8),
    Number_rabies DECIMAL(20, 8),
    Number_malaria DECIMAL(20, 8),
    Number_hiv DECIMAL(20, 8),
    Number_tuberculosis DECIMAL(20, 8),
    Number_smallpox DECIMAL(20, 8),
    Number_cholera_cases DECIMAL(20, 8)
);
```

The screenshot shows a SQL command-line interface. At the top, there are two lines of code: `USE pandemic;` and `select * from infectious_cases_original`. Below this, a table titled "infectious_cases_original 1" is displayed. The table has 9 columns: Entity, Code, Year, Number_yaws, polio_cases, cases_guinea_worm, Number_rabies, Number_malaria, and Number_hiv. The data for Afghanistan (AFG) spans from 1980 to 1985. The "Year" column is highlighted in red, indicating it is the current column being edited or selected.

	AZ Entity	AZ Cod	123 Yea	123 Number_yav	123 polio_cases	123 cases_guinea_w	123 Number_rabies	123 Number_malaria
1	Afghanistan	AFG	1,980	[NULL]	6,160	0	[NULL]	[NU]
2	Afghanistan	AFG	1,981	[NULL]	5,859	0	[NULL]	[NU]
3	Afghanistan	AFG	1,982	[NULL]	9,730	0	[NULL]	[NU]
4	Afghanistan	AFG	1,983	[NULL]	13,937	0	[NULL]	[NU]
5	Afghanistan	AFG	1,984	[NULL]	3,864	0	[NULL]	[NU]
6	Afghanistan	AFG	1,985	[NULL]	13,867	0	[NULL]	[NU]

```

USE pandemic;

select count(*) from infectious_cases_original

```

Results 1 × Statistics 1

select count(*) from infectious_cases_original | Enter a SQL expression to

	123 count(*)
1	10,521

2. Нормалізуйте таблицю infectious_cases до 3ї нормальної форми.
Збережіть у цій же схемі дві таблиці з нормалізованими даними.

```

USE pandemic;
DROP TABLE IF EXISTS Cases;
DROP TABLE IF EXISTS Countries;
CREATE TABLE Countries (
    country_code VARCHAR(3) PRIMARY KEY,
    entity_name VARCHAR(255) NOT NULL
);
CREATE TABLE Cases (
    country_code VARCHAR(3) NOT NULL,
    `year` INT NOT NULL,
    number_yaws DECIMAL(20, 8),
    polio_cases DECIMAL(20, 8),
    cases_guinea_worm DECIMAL(20, 8),
    number_rabies DECIMAL(20, 8),
    number_malaria DECIMAL(20, 8),
    number_hiv DECIMAL(20, 8),
    number_tuberculosis DECIMAL(20, 8),
    number_smallpox DECIMAL(20, 8),
    number_cholera_cases DECIMAL(20, 8),

    PRIMARY KEY (country_code, `year`),
    FOREIGN KEY (country_code) REFERENCES Countries(country_code)
)

```

```

USE pandemic;
INSERT INTO Countries (country_code, entity_name)
SELECT DISTINCT
    Code,
    Entity
FROM

```

```

infectious_cases_original
WHERE
Code IS NOT NULL AND Code != " AND LENGTH(Code) = 3;
INSERT INTO Cases (
country_code,
`year`,
number_yaws,
polio_cases,
cases_guinea_worm,
number_rabies,
number_malaria,
number_hiv,
number_tuberculosis,
number_smallpox,
number_cholera_cases
)
SELECT
Code,
`Year`,
Number_yaws,
polio_cases,
cases_guinea_worm,
Number_rabies,
Number_malaria,
Number_hiv,
Number_tuberculosis,
Number_smallpox,
Number_cholera_cases
FROM
infectious_cases_original
WHERE
Code IS NOT NULL AND `Year` IS NOT NULL AND LENGTH(Code) = 3;

```

The screenshot shows a MySQL command-line interface. The SQL code entered is:

```

USE pandemic;
select count(*) from cases

```

The results window shows the output of the query:

	123 count(*)
1	9,284

```
USE pandemic;
select count(*) from countries
```

Entity	Code
1	208

Some records were filtered out during the transfer from the original table to the two related tables.

Below you can see a list of such 37 records.

It looks it represents aggregates and countries without cases or countries with non-standard 3-char code.

```
USE pandemic;
SELECT DISTINCT
    Entity,
    Code
FROM
    infectious_cases_original
WHERE
    Code IS NULL OR Code = '' OR LENGTH(Code) != 3;
```

Entity	Code
31	South-East Asia Region (WHO)
32	Sub-Saharan Africa (WB)
33	Wales
34	Western Pacific
35	Western Pacific Region (WHO)
36	World
37	Yugoslavia

3. Проаналізуйте дані:

Для кожної унікальної комбінації Entity та Code або їх id порахуйте середнє, мінімальне, максимальне значення та суму для атрибута Number_rabies.

 Врахуйте, що атрибут Number_rabies може містити порожні значення

— вам попередньо необхідно їх відфільтрувати.

Результат відсортуйте за порахованим середнім значенням у порядку спадання.

Оберіть тільки 10 рядків для виведення на екран.

Option 1. For original table

```
USE pandemic;
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 total_rabies
FROM
    infectious_cases_original
WHERE
    Number_rabies IS NOT NULL
GROUP BY
    Entity, Code
ORDER BY
    avg_rabies DESC
LIMIT 10;
```

```

USE pandemic;

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 total_rabies
FROM
    infectious_cases_original
WHERE
    Number_rabies IS NOT NULL
GROUP BY
    Entity, Code
ORDER BY
    avg_rabies DESC
LIMIT 10;

```

infectious_cases_original 1 × Statistics 1

SELECT Entity, Code, AVG(Number_rabies) Enter a SQL expression to filter results (use Ctrl+Space)

	A-Z Entity	A-Z Code	123 avg_rabies	123 min_rabies	123 max_rabies	123 total_rabies
1	World	OWID_WRL	20,192,3703666667	14,075.508	24,744.658	605,771.111
2	Lower Middle Income		15,193,9593833333	10,202.528	19,182.795	455,818.7815
3	South Asia (WB)		11,729,8887533333	7,271.2754	15,361.878	351,896.6626
4	South-East Asia and Pacific		11,424,3268933333	6,806.0127	15,641.958	342,729.8068
5	G20		10,189,0460033333	6,339.0796	13,164.881	305,671.3801
6	India	IND	8,599.17328	5,425.8726	11,121.142	257,975.384

refresh ⌂ Save ⌂ Cancel ⌂ Export data 200 10 ... 10 row(s) fetched

Option 2. For normalised tables

```

USE pandemic;
SELECT
    c.country_code,
    c.entity_name,
    AVG(cs.number_rabies) AS avg_rabies,
    MIN(cs.number_rabies) AS min_rabies,
    MAX(cs.number_rabies) AS max_rabies,
    SUM(cs.number_rabies) AS total_rabies
FROM
    Countries AS c
JOIN
    Cases AS cs ON c.country_code = cs.country_code
WHERE
    cs.number_rabies IS NOT NULL
GROUP BY
    c.country_code, c.entity_name
ORDER BY
    avg_rabies DESC
LIMIT 10;

```

```

USE pandemic;

SELECT
    c.country_code,
    c.entity_name,
    AVG(cs.number_rabies) AS avg_rabies,
    MIN(cs.number_rabies) AS min_rabies,
    MAX(cs.number_rabies) AS max_rabies,
    SUM(cs.number_rabies) AS total_rabies
FROM
    Countries AS c
JOIN
    Cases AS cs ON c.country_code = cs.country_code
WHERE
    cs.number_rabies IS NOT NULL
GROUP BY
    c.country_code, c.entity_name
ORDER BY
    avg_rabies DESC
LIMIT 10;

```

Countries 1 × Statistics 1 | Enter a SQL expression to filter results (use Ctrl+Space)

	A-Z country_code	A-Z entity_name	123 avg_rabies	123 min_rabies	123 max_rabies	123 total_rabies
1	IND	India	8,599.17328	5,425.8726	11,121.142	257,
2	PAK	Pakistan	1,582.1696266667	1,177.1449	1,881.7257	47,
3	NGA	Nigeria	1,335.4154866667	1,073.9458	1,441.1188	40,
4	CHN	China	1,247.906733	744.251	2,075.593	37,4
5	ETH	Ethiopia	1,145.1725223333	758.0497	1,323.4026	34,3

refresh ⌂ Save ⌂ Cancel ⌂ Export data ⌂ 200 ⌂ 10 ⌂ 10 row(s) fetched

4. Побудуйте колонку різниці в роках.

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

атрибут, що створює дату першого січня відповідного року,

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

атрибут, що дорівнює поточній даті,

атрибут, що дорівнює різниці в роках двох вищезгаданих колонок.

```

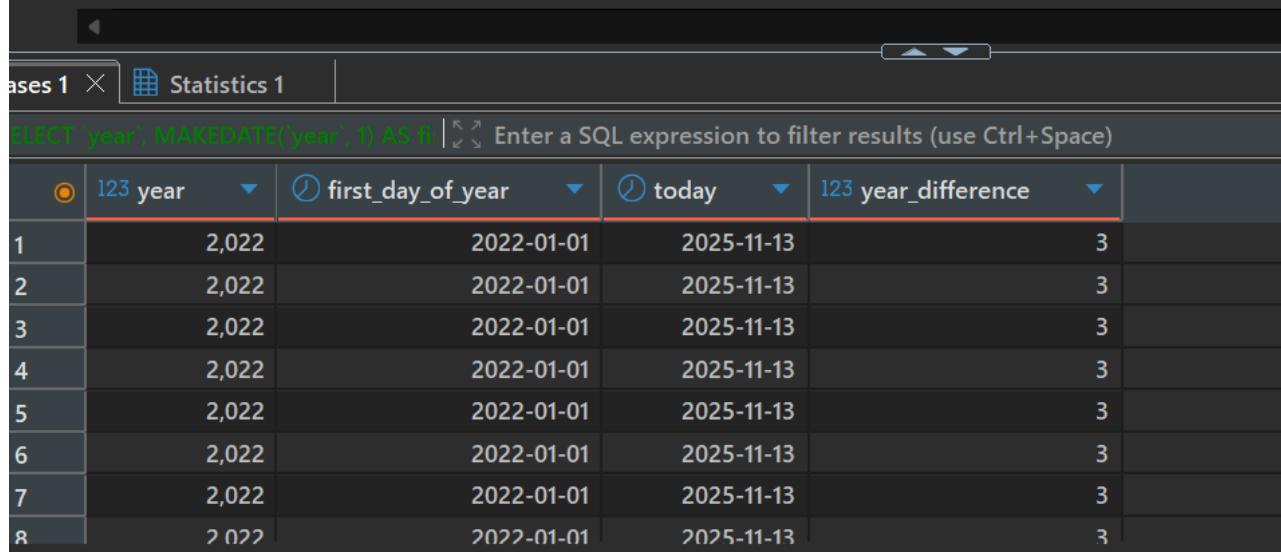
USE pandemic;

SELECT
    `year`,
    MAKEDATE(`year`, 1) AS first_day_of_year,
    CURRENT_DATE() AS today,
    TIMESTAMPDIFF(YEAR, MAKEDATE(`year`, 1), CURRENT_DATE()) AS year_difference
FROM
    Cases

```

```
ORDER BY  
    `year` DESC  
LIMIT 100;
```

```
USE pandemic;  
  
SELECT  
    `year`,  
    MAKEDATE(`year`, 1) AS first_day_of_year,  
    CURRENT_DATE() AS today,  
    TIMESTAMPDIFF(YEAR, MAKEDATE(`year`, 1), CURRENT_DATE()) AS year_difference  
FROM  
    Cases  
ORDER BY  
    `year` DESC  
LIMIT 100;
```



①	123 year	⌚ first_day_of_year	⌚ today	123 year_difference	
1	2,022	2022-01-01	2025-11-13	3	
2	2,022	2022-01-01	2025-11-13	3	
3	2,022	2022-01-01	2025-11-13	3	
4	2,022	2022-01-01	2025-11-13	3	
5	2,022	2022-01-01	2025-11-13	3	
6	2,022	2022-01-01	2025-11-13	3	
7	2,022	2022-01-01	2025-11-13	3	
8	2 022	2022-01-01	2025-11-13	3	

5. Побудуйте власну функцію.

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

```
USE pandemic;  
DELIMITER //  
DROP FUNCTION IF EXISTS GetYearDifference//  
CREATE FUNCTION GetYearDifference(input_year INT)  
RETURNS INT  
DETERMINISTIC  
BEGIN  
    DECLARE first_day_of_input_year DATE;
```

```

DECLARE today_date DATE;
SET first_day_of_input_year = MAKEDATE(input_year, 1);
SET today_date = CURRENT_DATE();
RETURN TIMESTAMPDIFF(YEAR, first_day_of_input_year, today_date);
END//
DELIMITER ;

```

```

USE pandemic;
SELECT
    `year`,
    MAKEDATE(`year`, 1) AS first_day_of_year,
    CURRENT_DATE() AS today,
    GetYearDifference(`year`) AS calculated_year_difference
FROM
    Cases
ORDER BY
    `year` DESC
LIMIT 100;

```

```

USE pandemic;

SELECT
    `year`,
    MAKEDATE(`year`, 1) AS first_day_of_year,
    CURRENT_DATE() AS today,
    GetYearDifference(`year`) AS calculated_year_difference
FROM
    Cases
ORDER BY
    `year` DESC
LIMIT 100;

```

cases 1 × Statistics 1

SELECT `year`, MAKEDATE(`year`, 1) AS fi | Enter a SQL expression to filter results (use Ctrl+Space)

	123 year	first_day_of_year	today	123 calculated_year_difference
1	2,022	2022-01-01	2025-11-13	3
2	2,022	2022-01-01	2025-11-13	3
3	2,022	2022-01-01	2025-11-13	3
4	2,022	2022-01-01	2025-11-13	3
5	2,022	2022-01-01	2025-11-13	3
6	2,022	2022-01-01	2025-11-13	3
7	2,022	2022-01-01	2025-11-13	3
8	2,022	2022-01-01	2025-11-13	3
9	2,022	2022-01-01	2025-11-13	3