

Фінальний проект РБД

Завдання 1, 2

Завантаження даних та нормалізація таблиць до 3 нормальної форми

The screenshot displays a SQL IDE interface with the following components:

- Left Panel (Navigator):** Shows the database structure for 'books'. The 'pandemic' database is selected, showing tables like 'diseasecases', 'entities', and 'infectious_cases'.
- Top Panel (Query Editor):** Contains a SQL script for 'Query 1'. The script includes:
 - Line 1: `USE pandemic;`
 - Line 2: Comment: `-- Виділення таблиці Entities для країн та їх кодів, щоб уникнути дублювання`
 - Line 4: `CREATE TABLE Entities (`
 - Line 5: `EntityID INT AUTO_INCREMENT PRIMARY KEY,`
 - Line 6: `Entity VARCHAR(255) NOT NULL,`
 - Line 7: `Code VARCHAR(10) NOT NULL`
 - Line 8: `);`
 - Line 9: `INSERT INTO Entities (Entity, Code)`
 - Line 10: `SELECT DISTINCT Entity, Code`
 - Line 11: `FROM infectious_cases;`
 - Line 12: `SELECT * FROM Entities;`
 - Line 13: Comment: `-- Виділення таблиці DiseaseCases, яка містить роки та кількості випадків захворювань для кожної сутності`
 - Line 17: `CREATE TABLE DiseaseCases (`
 - Line 18: `CaseID INT AUTO_INCREMENT PRIMARY KEY,`
 - Line 19: `EntityID INT NOT NULL,`
 - Line 20: `Year YEAR NOT NULL,`
 - Line 21: `Number_yaws INT,`
 - Line 22: `polio_cases INT,`
 - Line 23: `cases_guinea_worm INT,`
 - Line 24: `Number_rabies INT,`
 - Line 25: `Number_malaria INT,`
 - Line 26: `Number_hiv INT,`
 - Line 27: `Number_tuberculosis INT,`
 - Line 28: `Number_smallpox INT,`
 - Line 29: `Number_cholera_cases INT,`
 - Line 30: `FOREIGN KEY (EntityID) REFERENCES Entities(EntityID)`
 - Line 31: `--`
- Bottom Panel (Result Grid):** Shows the results of the query `SELECT * FROM Entities;`. The grid has columns 'EntityID', 'Entity', and 'Code'. The data is as follows:

EntityID	Entity	Code
1	Afghanistan	AFG
2	Albania	ALB
3	Algeria	DZA
4	Andorra	AND
5	Angola	AGO
6	Antigua and Barbuda	ATG
7	Argentina	ARG

[illegible]

Заповнення та перевірка даних

```
34 * INSERT INTO DiseaseCases (
35     EntityID, Year, Number_years, polio_cases, cases_guinea_worm,
36     Number_rabies, Number_malaria, Number_hiv,
37     Number_tuberculosis, Number_smallpox, Number_cholera_cases
38 )
39
40 SELECT
41     e.EntityID,
42     ic.Year,
43     NULLIF(ic.Number_years, '') AS Number_years,
44     NULLIF(ic.polio_cases, '') AS polio_cases,
45     NULLIF(ic.cases_guinea_worm, '') AS cases_guinea_worm,
46     NULLIF(ic.Number_rabies, '') AS Number_rabies,
47     NULLIF(ic.Number_malaria, '') AS Number_malaria,
48     NULLIF(ic.Number_hiv, '') AS Number_hiv,
49     NULLIF(ic.Number_tuberculosis, '') AS Number_tuberculosis,
50     NULLIF(ic.Number_smallpox, '') AS Number_smallpox,
51     NULLIF(ic.Number_cholera_cases, '') AS Number_cholera_cases
52 FROM Infectious_cases ic
53 JOIN Entities e ON ic.EntityID = e.EntityID
54
55 -- Перевірка даних
56 * SELECT e.Entity, e.Code, dc.Year, dc.Number_rabies
57 FROM DiseaseCases dc
58 JOIN Entities e ON dc.EntityID = e.EntityID
59
60 * SELECT Entity, COUNT(*)
61 FROM Entities
62 GROUP BY Entity
63
64
65 -- Використовуємо таблиці DiseaseCases і Entities. Для кожного Entity та Code виконаємо агреговані функції (AVG, MIN, MAX, SUM) для атрибута Number_rabies
66 * SELECT
67     e.Entity AS Entity,
68     e.Code AS Code,
69     AVG(dc.Number_rabies) AS Avg_Number_rabies,
70     MIN(dc.Number_rabies) AS Min_Number_rabies,
71     MAX(dc.Number_rabies) AS Max_Number_rabies,
72     SUM(dc.Number_rabies) AS Sum_Number_rabies
73 FROM DiseaseCases dc
74 JOIN Entities e ON dc.EntityID = e.EntityID
75 GROUP BY e.Entity, e.Code
```

Output

Action	Time	Message
39 14:57:59 SELECT e.Entity, e.Code AS Code, AVG(dc.Number_rabies) AS Avg_Number_rabies, MIN(dc.Number_rabies) AS Min_Number_rabies, MAX(dc.Number_rabies) AS Max_Number_rabies, SUM(dc.Number_rabies) AS Sum_Number_rabies FROM DiseaseCases dc JOIN Entities e ON dc.EntityID = e.EntityID	15:01:06	10 rows returned
40 15:01:06 SELECT Entity, COUNT(*) FROM Entities GROUP BY Entity	15:01:06	10 rows returned
41 15:03:01 INSERT INTO DiseaseCases (EntityID, Year, Number_years, polio_cases, cases_guinea_worm, Number_rabies, Number_malaria, Number_hiv, Number_tuberculosis, Number_smallpox, Number_cholera_cases)	15:03:01	7271 rows affected Records: 7271 Duplicates: 0 Warnings: 0

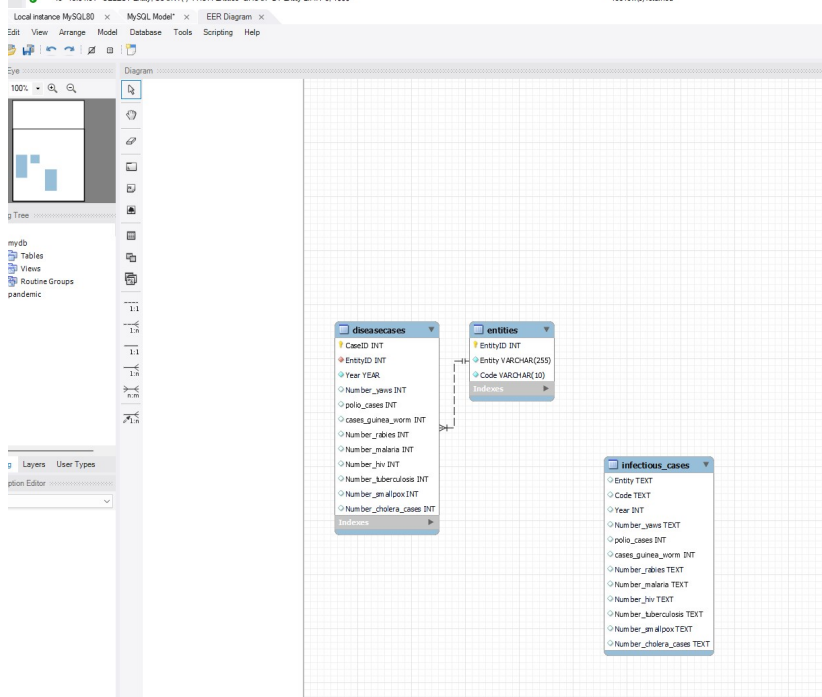
```
54
55 -- Перевірка даних
56 * SELECT e.Entity, e.Code, dc.Year, dc.Number_rabies
57 FROM DiseaseCases dc
58 JOIN Entities e ON dc.EntityID = e.EntityID
59
60 * SELECT Entity, COUNT(*)
61 FROM Entities
62 GROUP BY Entity
63
64
65 -- Використовуємо таблиці DiseaseCases і Entities. Для кожного Entity та Code виконаємо агреговані функції (AVG, MIN, MAX, SUM) для атрибута Number_rabies
66 * SELECT
67     e.Entity AS Entity,
68     e.Code AS Code,
69     AVG(dc.Number_rabies) AS Avg_Number_rabies,
```

Result Grid

Entity	COUNT(*)
Afghanistan	1
Albania	1
Algeria	1
Andorra	1
Angola	1
Antigua and Barbuda	1
Argentina	1
Armenia	1
Australia	1

Output

Action	Time	Message
41 15:03:01 INSERT INTO DiseaseCases (EntityID, Year, Number_years, polio_cases, cases_guinea_worm, Number_rabies, Number_malaria, Number_hiv, Number_tuberculosis, Number_smallpox, Number_cholera_cases)	15:03:01	7271 rows affected Records: 7271 Duplicates: 0 Warnings: 0
42 15:04:27 SELECT e.Entity, e.Code, dc.Year, dc.Number_rabies FROM DiseaseCases dc JOIN Entities e ON dc.EntityID = e.EntityID LIMIT 0, 1000	15:04:27	1000 rows returned
43 15:04:51 SELECT Entity, COUNT(*) FROM Entities GROUP BY Entity LIMIT 0, 1000	15:04:51	195 rows returned



Завдання 3

Аналіз даних

```
62 GROUP BY Entity;
63
64 -- Використовуємо таблиці DiseaseCases і Entities. Для кожного Entity та Code виконаємо агрегуючі функції (AVG, MIN, MAX, SUM) для атрибута Number_rabies
65 SELECT
66     e.Entity AS Entity,
67     e.Code AS Code,
68     AVG(dc.Number_rabies) AS Avg_Number_rabies,
69     MIN(dc.Number_rabies) AS Min_Number_rabies,
70     MAX(dc.Number_rabies) AS Max_Number_rabies,
71     SUM(dc.Number_rabies) AS Sum_Number_rabies
72 FROM DiseaseCases dc
73 JOIN Entities e ON dc.EntityID = e.EntityID
74 WHERE dc.Number_rabies IS NOT NULL
75 GROUP BY e.Entity, e.Code
76 ORDER BY Avg_Number_rabies DESC
77 LIMIT 10;
78
```

Entity	Code	Avg_Number_rabies	Min_Number_rabies	Max_Number_rabies	Sum_Number_rabies
World	OWID_WRL	20192.3667	14076	24745	1211542
India	IND	8599.2333	5426	11121	515954
Pakistan	PAK	1582.2000	1177	1882	94932
Nigeria	NGA	1335.4667	1074	1441	80128
China	CHN	1248.0000	744	2076	74880
Ethiopia	ETH	1145.2333	758	1323	68714
Myanmar	MMR	972.9000	561	1450	58374
Bangladesh	BGD	785.7333	180	1735	47144
Nepal	NPL	661.1667	338	1042	39670

Result 25 x

Output

Action Output

#	Time	Action	Message
42	15:04:27	SELECT e.Entity, e.Code, dc.Year, dc.Number_rabies FROM DiseaseCases dc JOIN Entities e ON dc.EntityID = e.EntityID LIMIT 0, 1000	1000 row(s) returned
43	15:04:51	SELECT Entity, COUNT(*) FROM Entities GROUP BY Entity LIMIT 0, 1000	195 row(s) returned
44	15:05:44	SELECT e.Entity AS Entity, e.Code AS Code, AVG(dc.Number_rabies) AS Avg_Number_rabies, MIN(dc.Number_rabies) AS Min_Number_rabies, MAX(dc.Number_rabies) AS Max_Number_rabies, SUM(dc.Number_rabies) AS Sum_Number_rabies FROM DiseaseCases dc JOIN Entities e ON dc.EntityID = e.EntityID WHERE dc.Number_rabies IS NOT NULL GROUP BY e.Entity, e.Code ORDER BY Avg_Number_rabies DESC LIMIT 10;	10 row(s) returned

Завдання 4

Різниця в роках

```
78
79
80 SELECT DISTINCT Year FROM DiseaseCases;
81
82 SELECT
83     Year,
84     STR_TO_DATE(CONCAT('Year', '-01-01'), '%Y-%m-%d') AS Date_1st_Jan,
85     CURDATE() AS Current_Date,
86     TIMESTAMPDIF(YEAR, STR_TO_DATE(CONCAT('Year', '-01-01'), '%Y-%m-%d'), CURDATE()) AS Year_Difference
87 FROM DiseaseCases
88 LIMIT 10;
89
90
91
92
```

Year	Date_1st_Jan	Current_Date	Year_Difference
1980	1980-01-01	2024-12-01	44
1981	1981-01-01	2024-12-01	43
1982	1982-01-01	2024-12-01	42
1983	1983-01-01	2024-12-01	41
1984	1984-01-01	2024-12-01	40
1985	1985-01-01	2024-12-01	39
1986	1986-01-01	2024-12-01	38
1987	1987-01-01	2024-12-01	37
1988	1988-01-01	2024-12-01	36

Result 35 x

Output

Action Output

#	Time	Action	Message
5	15:23:26	SELECT Year, STR_TO_DATE(CONCAT('Year', '-01-01'), '%Y-%m-%d') AS Date_1st_Jan, CURDATE() AS Current_Date, TIMESTAMPDIF(YEAR, STR_TO_DATE(CONCAT('Year', '-01-01'), '%Y-%m-%d'), CURDATE()) AS Year_Difference FROM DiseaseCases	10 row(s) returned
6	15:23:57	SELECT DISTINCT Year FROM DiseaseCases LIMIT 0, 1000	41 row(s) returned
7	15:24:08	SELECT Year, STR_TO_DATE(CONCAT('Year', '-01-01'), '%Y-%m-%d') AS Date_1st_Jan, CURDATE() AS Current_Date, TIMESTAMPDIF(YEAR, STR_TO_DATE(CONCAT('Year', '-01-01'), '%Y-%m-%d'), CURDATE()) AS Year_Difference FROM DiseaseCases LIMIT 10;	10 row(s) returned

Завдання 5

Створення функцій

Функція, яка приймає рік і повертає різницю в роках між поточною датою та 1 січня цього року

The screenshot shows a SQL IDE interface with a query editor and a results pane. The query editor contains the following SQL code:

```
-- Створимо функцію, яка прийматиме рік і повертатиме різницю в роках між поточною датою та 1 січня цього року
DELIMITER //
CREATE FUNCTION YearDifference(inputYear INT)
RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE firstJanDate DATE;
    DECLARE yearDifference INT;

    -- Створимо дату 1 січня відповідного року
    SET firstJanDate = STR_TO_DATE(CONCAT(inputYear, '-01-01'), '%Y-%m-%d');

    -- Обчислимо різницю в роках між поточною датою та цією датою
    SET yearDifference = TIMESTAMPOFF(YEAR, firstJanDate, CURDATE());

    RETURN yearDifference;
END //
DELIMITER ;
SELECT YearDifference(1996) AS Difference;
```

The results pane shows a single row with the value 28 for the column Difference.

#	Time	Action	Message
8	16:48:28	CREATE FUNCTION YearDifference(inputYear INT) RETURNS INT DETERMINISTIC BEGIN DECLARE firstJanDate DATE; DECLARE yearDiffer...	0 row(s) affected
9	16:48:39	CREATE FUNCTION YearDifference(inputYear INT) RETURNS INT DETERMINISTIC BEGIN DECLARE firstJanDate DATE; DECLARE yearDiffer...	Error Code: 1304. FUNCTION YearDifference already exists
10	16:49:31	SELECT YearDifference(1996) AS Difference LIMIT 0, 1000	1 row(s) returned

Функція, яка обчислює середню кількість захворювань за заданий період на основі річних даних

The screenshot shows the SQL Developer interface with a query window titled 'Query 1' containing the following SQL code:

```
113
114 -- Функція, яка обчислює середню кількість захворювань за заданий період на основі річних даних
115
116 DELIMITER //
117
118 CREATE FUNCTION CalculatePeriodAverage(yearlyCases INT, divisor INT)
119 RETURNS FLOAT
120 DETERMINISTIC
121 BEGIN
122 -- Перевірка на нульовий дільник, щоб уникнути помилки
123 IF divisor = 0 THEN
124 RETURN NULL;
125 END IF;
126 -- Обчислення середньої кількості захворювань
127 RETURN yearlyCases / divisor;
128
129
130 DELIMITER ;
131
132 SELECT CalculatePeriodAverage(120, 12) AS MonthlyAverage;
133
134 SELECT
135 'Year',
136 'Number_rabies',
137 CalculatePeriodAverage('Number_rabies', 12) AS Rabies_MonthlyAverage,
138 CalculatePeriodAverage('Number_rabies', 4) AS Rabies_QuarterlyAverage,
139 CalculatePeriodAverage('Number_rabies', 2) AS Rabies_HalfYearlyAverage,
140
141 'Number_malaria',
```

The 'Result Grid' shows the output of the first query:

MonthlyAverage
10

The 'Action Output' pane shows the execution results:

#	Time	Action	Message
12	16:54:22	SELECT CalculatePeriodAverage(120, 12) AS MonthlyAverage LIMIT 0, 1000	1 row(s) returned
13	16:56:49	SELECT 'Year', 'Number_rabies', CalculatePeriodAverage('Number_rabies', 12) AS Rabies_MonthlyAverage, CalculatePeriodAverage('Number_rabies', 4) AS Rabies_QuarterlyAverage, CalculatePeriodAverage('Number_rabies', 2) AS Rabies_HalfYearlyAverage, 'Number_malaria',	10 row(s) returned
14	16:57:54	SELECT CalculatePeriodAverage(120, 12) AS MonthlyAverage LIMIT 0, 1000	1 row(s) returned

The screenshot shows the SQL Developer interface with a query window titled 'Query 1' containing the following SQL code:

```
120
130 DELIMITER ;
131
132 SELECT CalculatePeriodAverage(120, 12) AS MonthlyAverage;
133
134 -- функція, яка обчислює середню кількість захворювань за заданий період на основі річних даних
135
136 SELECT
137 'Year',
138 'Number_rabies',
139 CalculatePeriodAverage('Number_rabies', 12) AS Rabies_MonthlyAverage,
140 CalculatePeriodAverage('Number_rabies', 4) AS Rabies_QuarterlyAverage,
141 CalculatePeriodAverage('Number_rabies', 2) AS Rabies_HalfYearlyAverage,
142
143 'Number_malaria',
144 CalculatePeriodAverage('Number_malaria', 12) AS Malaria_MonthlyAverage,
145 CalculatePeriodAverage('Number_malaria', 4) AS Malaria_QuarterlyAverage,
146 CalculatePeriodAverage('Number_malaria', 2) AS Malaria_HalfYearlyAverage,
147
148 'Number_hiv',
149 CalculatePeriodAverage('Number_hiv', 12) AS HIV_MonthlyAverage,
150 CalculatePeriodAverage('Number_hiv', 4) AS HIV_QuarterlyAverage,
151 CalculatePeriodAverage('Number_hiv', 2) AS HIV_HalfYearlyAverage,
152
153 FROM DiseaseCases
154 WHERE 'Number_rabies' IS NOT NULL
155 OR 'Number_malaria' IS NOT NULL
156 OR 'Number_hiv' IS NOT NULL
157 LIMIT 10;
```

The 'Result Grid' shows the output of the second query:

Year	Number_rabies	Rabies_MonthlyAverage	Rabies_QuarterlyAverage	Rabies_HalfYearlyAverage	Number_malaria	Malaria_MonthlyAverage	Malaria_QuarterlyAverage	Malaria_HalfYearlyAverage	Number_hiv	HIV_MonthlyAverage	HIV_QuarterlyAverage
1990	2	0.166667	0.5	1	121846	101654	304952	609923	88	7.33333	22
1991	2	0.166667	0.5	1	1132782	94398.5	283196	566391	104	8.66667	26
1995	3	0.25	0.75	1.5	725788	60479.8	181440	362879	144	12	36
1997	3	0.25	0.75	1.5	605853	50487.8	151463	302926	161	13.4167	40.25
1998	4	0.333333	1	2	547045	45587.1	136761	273522	165	13.75	41.25
1999	4	0.333333	1	2	502899	41908.2	125725	251490	169	14.0833	42.25
2000	4	0.333333	1	2	491615	40967.9	122504	245008	178	14.8333	44.5
2001	4	0.333333	1	2	513626	42801.5	127906	255813	193	16.0833	48.25

The 'Action Output' pane shows the execution results:

#	Time	Action	Message
13	16:56:49	SELECT 'Year', 'Number_rabies', CalculatePeriodAverage('Number_rabies', 12) AS Rabies_MonthlyAverage, CalculatePeriodAverage('Number_rabies', 4) AS Rabies_QuarterlyAverage, CalculatePeriodAverage('Number_rabies', 2) AS Rabies_HalfYearlyAverage, 'Number_malaria',	10 row(s) returned
14	16:57:54	SELECT CalculatePeriodAverage(120, 12) AS MonthlyAverage LIMIT 0, 1000	1 row(s) returned
15	16:59:36	SELECT 'Year', 'Number_rabies', CalculatePeriodAverage('Number_rabies', 12) AS Rabies_MonthlyAverage, CalculatePeriodAverage('Number_rabies', 4) AS Rabies_QuarterlyAverage, CalculatePeriodAverage('Number_rabies', 2) AS Rabies_HalfYearlyAverage, 'Number_malaria',	10 row(s) returned