



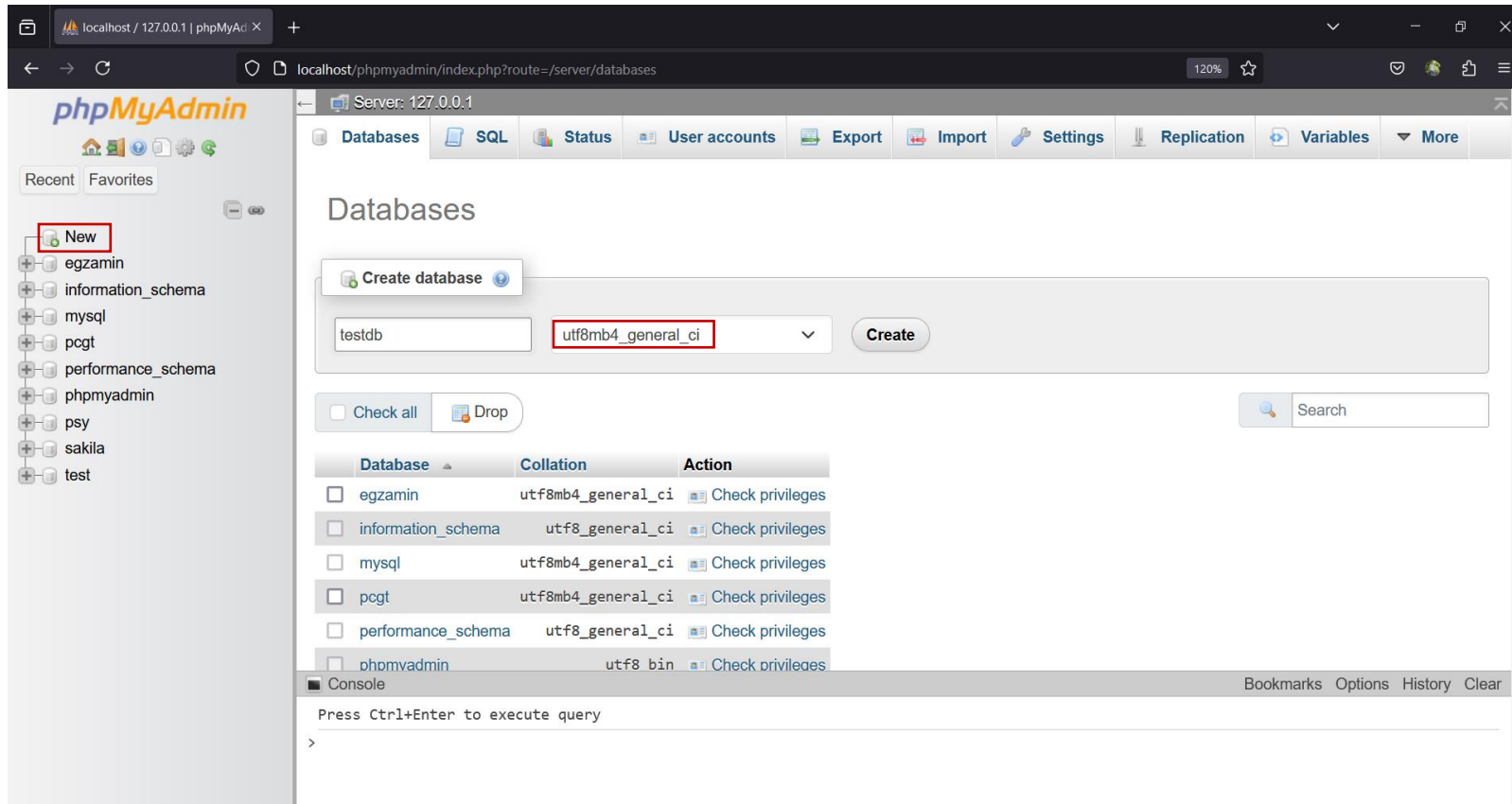
# **DDL**

**(Data Definition Language)**

[dimon.work/kurs.html](http://dimon.work/kurs.html)

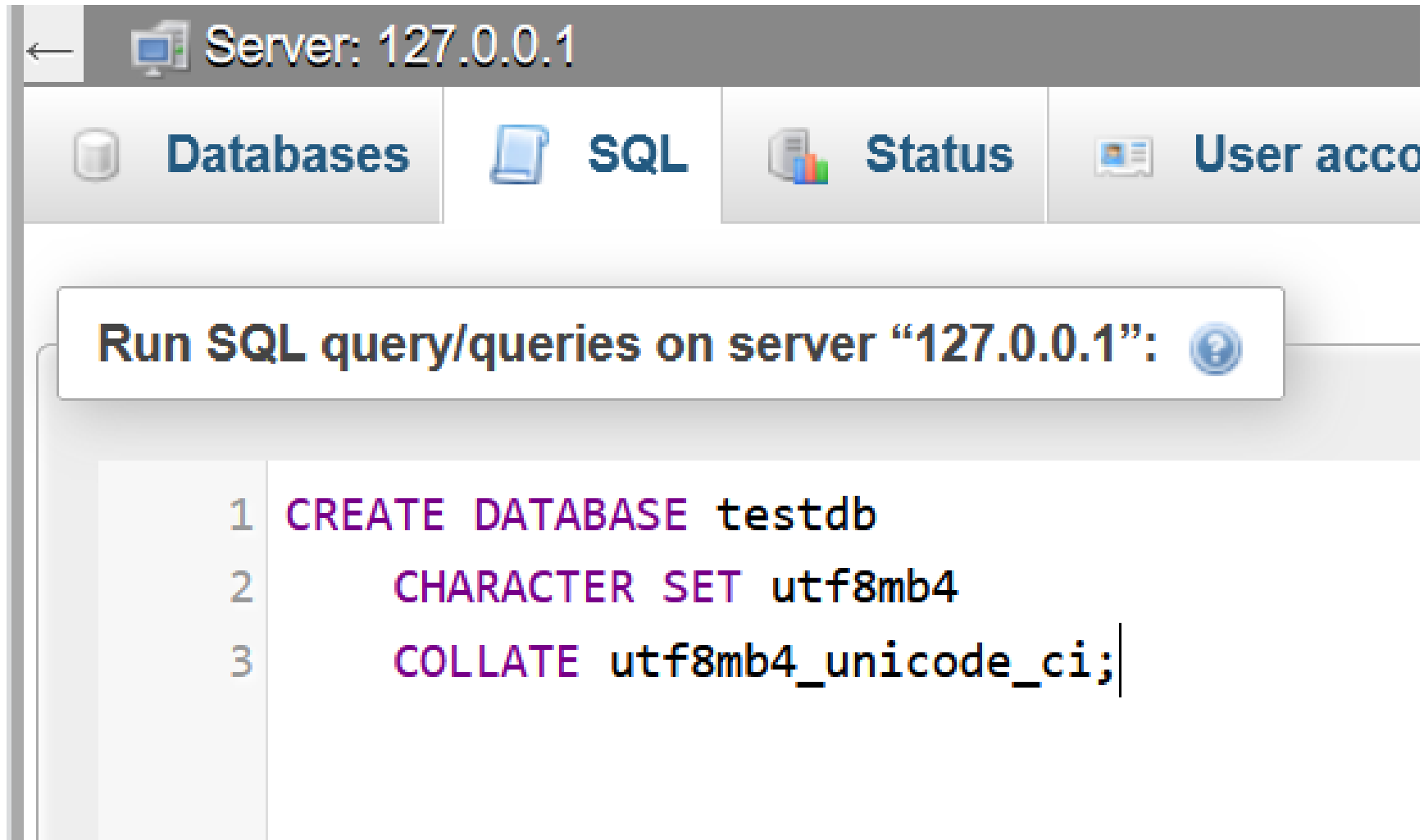
- **Data Definition Language, DDL** – grupa instrukcji w języku [SQL](#), które służą do definiowania struktur danych. Możemy do nich zaliczyć polecenia takie jak **CREATE**, **ALTER**, **DROP**. Za pomocą instrukcji DDL nie manipulujemy bezpośrednio danymi, a ich strukturą. Można zdefiniować kolumny tabel, zmienić typy danych, czy usunąć obiekt taki jak widok czy tabela.

# Jak stworzyć BD



- **utf8mb4\_unicode\_ci** jest oparty na standardzie Unicode do sortowania i porównywania ciągów, który sortuje ciągi dokładniej w szerokim zakresie języków/alfabetach.
- **utf8mb4\_general\_ci** nie implementuje wszystkich reguł sortowania Unicode, co często prowadzi do niepożądanych wyników w niektórych sytuacjach dla niektórych języków/znaków.

# Jak stworzyć BD

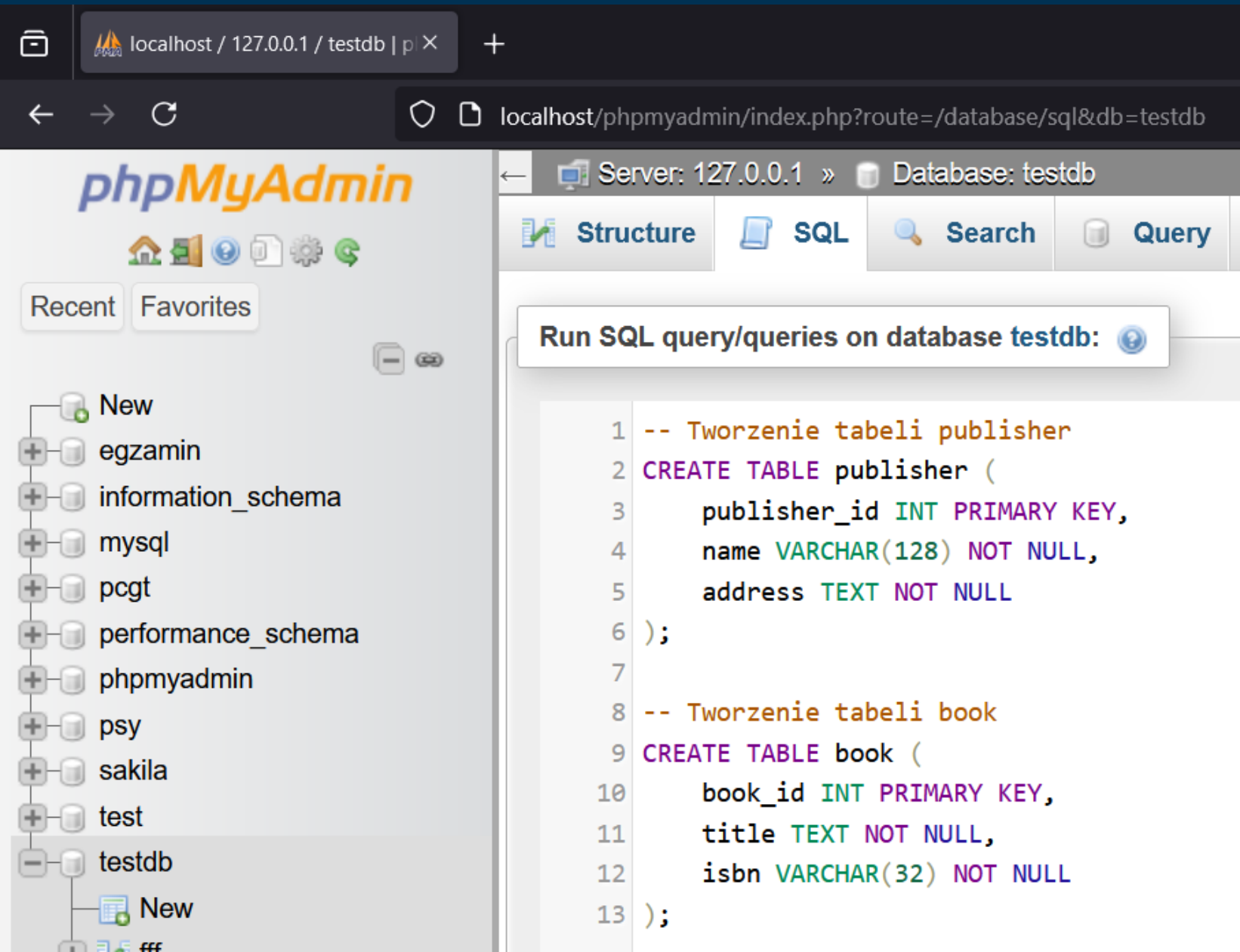


The screenshot shows a MySQL management tool interface. At the top, there is a header bar with a back arrow, a server icon, and the text "Server: 127.0.0.1". Below this are four tabs: "Databases", "SQL", "Status", and "User accounts". The "SQL" tab is active. A white box with a blue question mark icon contains the text "Run SQL query/queries on server '127.0.0.1':". Below this, a text area contains the following SQL code:

```
1 CREATE DATABASE testdb
2     CHARACTER SET utf8mb4
3     COLLATE utf8mb4_unicode_ci;
```

# Jak dodać tabelę w BD

dimon.work



The screenshot shows the phpMyAdmin interface. The browser address bar indicates the URL is localhost/phpmyadmin/index.php?route=/database/sql&db=testdb. The interface shows the 'testdb' database selected in the left sidebar. The main area displays the SQL editor with the following code:

```
1 -- Tworzenie tabeli publisher
2 CREATE TABLE publisher (
3     publisher_id INT PRIMARY KEY,
4     name VARCHAR(128) NOT NULL,
5     address TEXT NOT NULL
6 );
7
8 -- Tworzenie tabeli book
9 CREATE TABLE book (
10    book_id INT PRIMARY KEY,
11    title TEXT NOT NULL,
12    isbn VARCHAR(32) NOT NULL
13 );
```

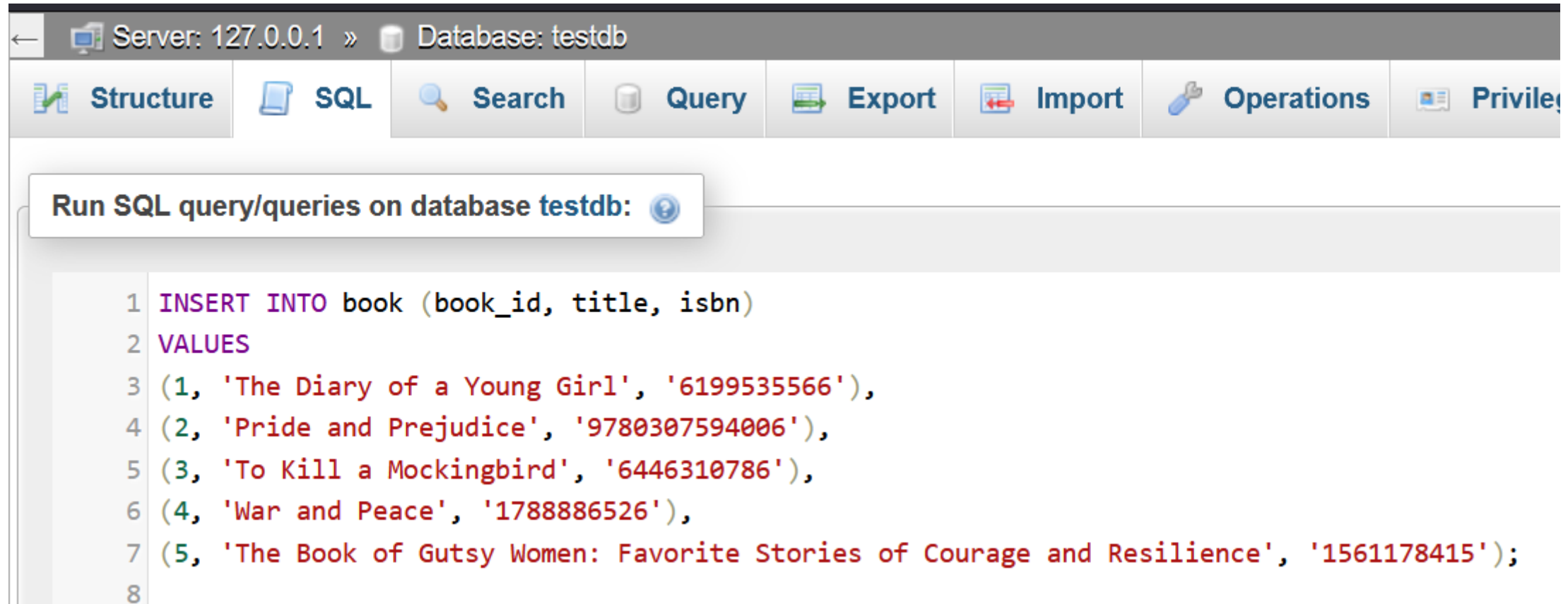
Dla tabeli publisher :

- **publisher\_id INT PRIMARY KEY** - jest to klucz główny, który jednoznacznie identyfikuje każdego wydawcę.
- **name VARCHAR(128) NOT NULL** - nazwa wydawcy z limitem 128 znaków.
- **address TEXT NOT NULL** - pole tekstowe do przechowywania adresu.

Dla tabeli book:

- **book\_id INT PRIMARY KEY** - klucz główny dla tabeli book.
- **title TEXT NOT NULL** - tytuł książki.
- **isbn VARCHAR(32) NOT NULL** - pole na numer ISBN z limitem 32 znaków.

# Wypełnienie tabel danymi



The screenshot shows a database management tool interface. At the top, there is a navigation bar with a back arrow, a server icon, and the text "Server: 127.0.0.1 » Database: testdb". Below this is a menu bar with icons and labels for "Structure", "SQL", "Search", "Query", "Export", "Import", "Operations", and "Privileges". The main area contains a text box with the prompt "Run SQL query/queries on database testdb:" followed by a question mark icon. Below the prompt is a code editor with the following SQL query:

```
1 INSERT INTO book (book_id, title, isbn)
2 VALUES
3 (1, 'The Diary of a Young Girl', '6199535566'),
4 (2, 'Pride and Prejudice', '9780307594006'),
5 (3, 'To Kill a Mockingbird', '6446310786'),
6 (4, 'War and Peace', '1788886526'),
7 (5, 'The Book of Gutsy Women: Favorite Stories of Courage and Resilience', '1561178415');
8
```

To polecenie SQL wstawia pięć rekordów (książek) do tabeli book, określając identyfikator książki (book\_id), tytuł (title) i ISBN (isbn). Każdy rekord zawiera unikalny identyfikator i powiązane informacje o książce.

# Wypełnienie tabel danymi

```
Server: 127.0.0.1 » Database: testdb
Structure SQL Search Query Export Import Operations Privil

Run SQL query/queries on database testdb:

1 INSERT INTO book (book_id, title, isbn)
2 VALUES
3 (1, 'The Diary of a Young Girl', '6199535566'),
4 (2, 'Pride and Prejudice', '9780307594006'),
5 (3, 'To Kill a Mockingbird', '6446310786'),
6 (4, 'War and Peace', '1788886526'),
7 (5, 'The Book of Gutsy Women: Favorite Stories of Courage and Resilience', '1561178415');
8
9 INSERT INTO publisher (publisher_id, name, address)
10 VALUES
11 (1, 'Everyman''s Library', 'NY'),
12 (2, 'Oxford University Press', 'NY'),
13 (3, 'Grand Central Publishing', 'Washington'),
14 (4, 'Simon & Schuster', 'Chicago');
15
```

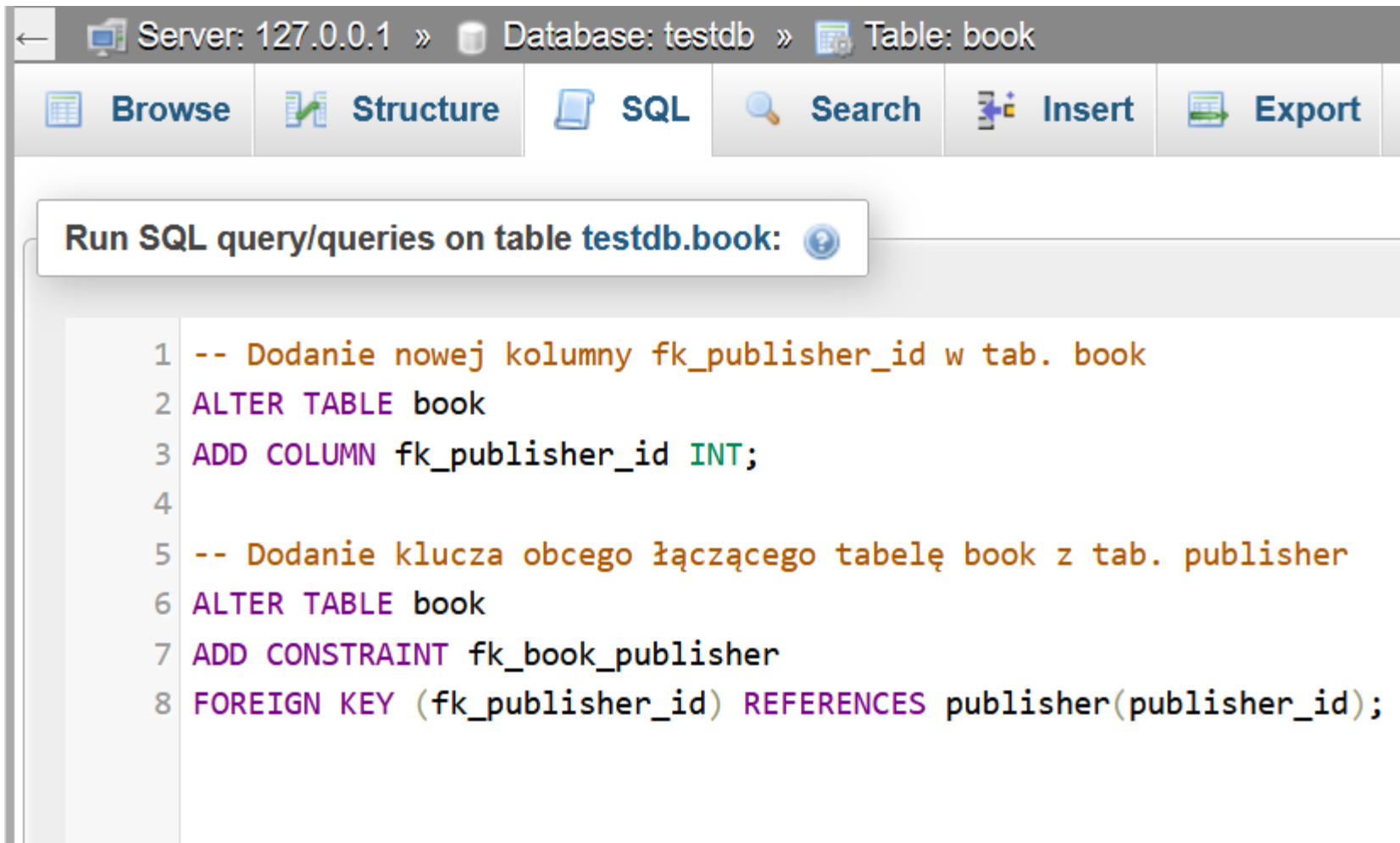
- Pojedynczy apostrof w łańcuchach SQL musi być uniknięty podwójnym apostrofem: 'Everyman''s Library'.
- Każdy rekord jest oddzielony przecinkiem, a średnik (;) jest używany tylko na końcu całego zapytania.

# Relacja 1 do wielu

Publisher				Book			
id	name	address		id	title	ISBN	
1	Everyman"s Library	NY		1	The Diary of a Young Girl	199535566	
2	Oxford University Press	NY		2	Pride and Prejudice	9780307594006	
3	Grand Central Publishing	Washington		3	To Kill a Mockingbird	0446310786	
4	Simon & Schuster	Chicago		4	The Book of Gutsy Women	1501178415	
				5	War and Peace	1788886526	
				1 publisher can publish many books			
		<b>id</b>	<b>name</b>	<b>address</b>	<b>title</b>	<b>ISBN</b>	
		2	Oxford University Press	NY	To Kill a Mockingbird	0446310786	
		2	Oxford University Press	NY	The Book of Gutsy Women	1501178415	
		2	Oxford University Press	NY	War and Peace	1788886526	

Dla połączenia tabel wykorzystamy **klucz obcy**. Dodamy w tabelę book dodatkową kolumnę z indeksem wydania





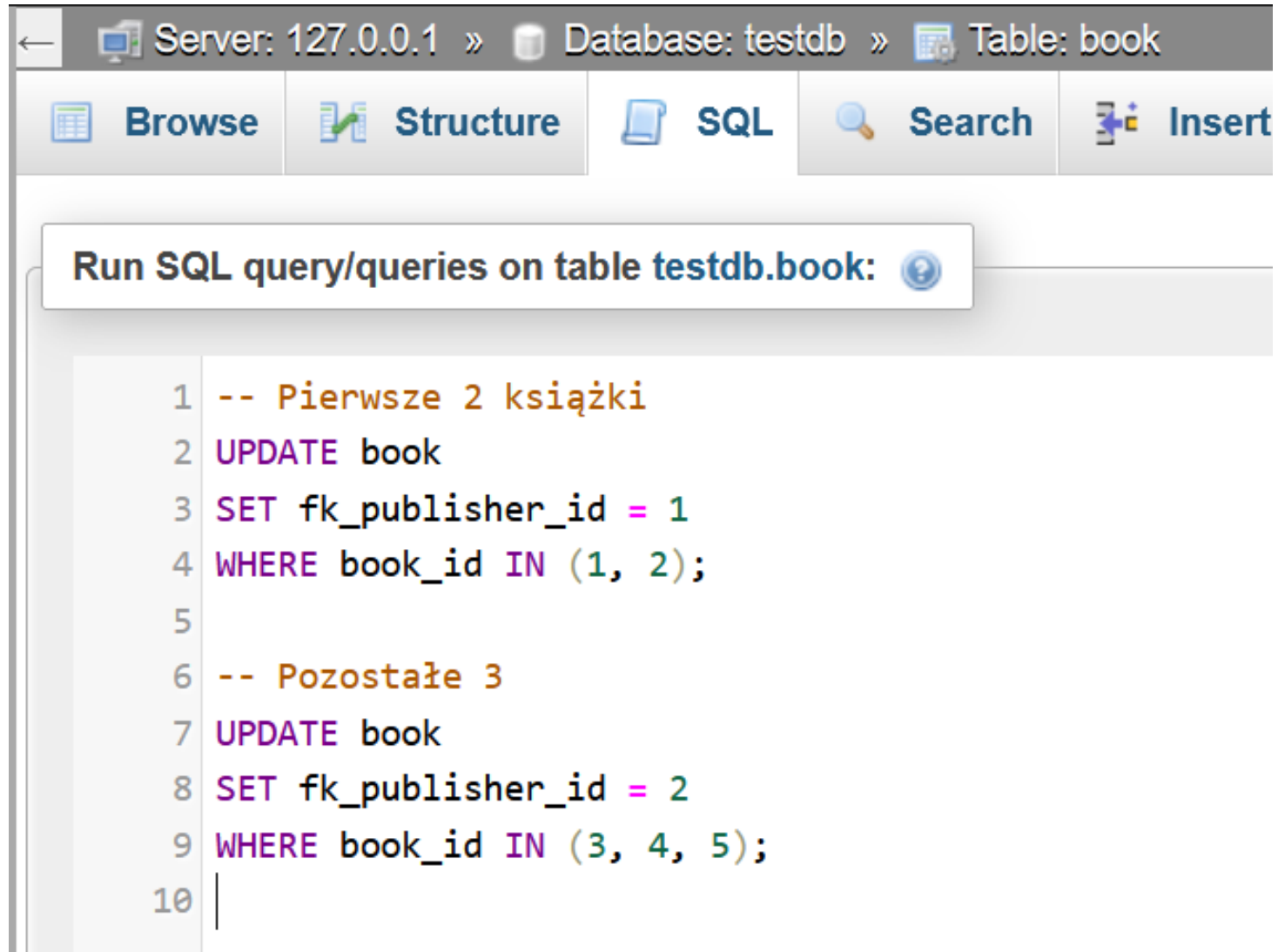
The screenshot shows a database management tool interface. At the top, there is a breadcrumb navigation: "Server: 127.0.0.1 » Database: testdb » Table: book". Below this is a toolbar with buttons for "Browse", "Structure", "SQL", "Search", "Insert", and "Export". A text box contains the prompt "Run SQL query/queries on table testdb.book:". Below the text box is a code editor with the following SQL queries:

```
1 -- Dodanie nowej kolumny fk_publisher_id w tab. book
2 ALTER TABLE book
3 ADD COLUMN fk_publisher_id INT;
4
5 -- Dodanie klucza obcego łączącego tabelę book z tab. publisher
6 ALTER TABLE book
7 ADD CONSTRAINT fk_book_publisher
8 FOREIGN KEY (fk_publisher_id) REFERENCES publisher(publisher_id);
```

- **ADD COLUMN** `fk_publisher_id INT`; - do tabeli `book` dodawana jest nowa kolumna `fk_publisher_id` typu `INT`, która posłuży jako klucz obcy do połączenia z tabelą `publisher`.
- **ADD CONSTRAINT** `fk_book_publisher` - dodawane jest ograniczenie klucza obcego, które łączy kolumnę `fk_publisher_id` tabeli `book` z kolumną `publisher_id` tabeli `publisher`.

# Books & Publishers

dimon.work



The screenshot shows a database management tool interface. At the top, the breadcrumb navigation indicates the current location: Server: 127.0.0.1 » Database: testdb » Table: book. Below this, there are five main menu items: Browse, Structure, SQL, Search, and Insert. A tooltip is visible over the SQL menu item, displaying the text: "Run SQL query/queries on table testdb.book:". The main area of the interface is a text editor containing SQL code. The code is as follows:

```
1  -- Pierwsze 2 książki
2  UPDATE book
3  SET fk_publisher_id = 1
4  WHERE book_id IN (1, 2);
5
6  -- Pozostałe 3
7  UPDATE book
8  SET fk_publisher_id = 2
9  WHERE book_id IN (3, 4, 5);
10 |
```

book_id	title	isbn	fk_publisher_id
1	The Diary of a Young Girl	6199535566	1
2	Pride and Prejudice	9780307594006	1
3	To Kill a Mockingbird	6446310786	2
4	War and Peace	1788886526	2
5	The Book of Gutsy Women: Favorite Stories of Coura...	1561178415	2

# Relacja 1 do 1

Person			Passport			
id	first_name	last_name	serial_number	registration	fk_person_id	
1	John	Snow	123456	Winterfell	1	
2	Ned	Stark	789012	Winterfell	2	
3	Rob	Baratheon	345678	King's Landing	3	
Полная выборка из объединённых таблиц						
	id	first_name	last_name	serial_number	registration	
	1	John	Snow	123456	Winterfell	
	2	Ned	Stark	789012	Winterfell	
	3	Rob	Baratheon	345678	King's Landing	

# Person & Passport

dimon.work

Server: 127.0.0.1 » Database: testdb

Structure SQL Search Query Export Import

Run SQL query/queries on database testdb: ?

```
1 -- Tworzenie tabeli person
2 CREATE TABLE person (
3     person_id INT PRIMARY KEY,
4     first_name VARCHAR(64) NOT NULL,
5     last_name VARCHAR(64) NOT NULL
6 );
7
8 -- Tworzenie tabeli passport
9 CREATE TABLE passport (
10    passport_id INT PRIMARY KEY,
11    serial_number INT NOT NULL,
12    fk_person_id INT,
13    FOREIGN KEY (fk_person_id) REFERENCES person(person_id)
14 );
```

Server: 127.0.0.1 » Database: testdb

Structure SQL Search Query Export

Run SQL query/queries on database testdb: ?

```
1 INSERT INTO person VALUES (1, 'John', 'Snow');
2 INSERT INTO person VALUES (2, 'Ned', 'Stark');
3 INSERT INTO person VALUES (3, 'Rob', 'Baratheon');
4
```

Server: 127.0.0.1 » Database: testdb

Structure SQL Search Query

Run SQL query/queries on database testdb: ?

```
1 INSERT INTO passport VALUES (1, 123456, 1);
2 INSERT INTO passport VALUES (2, 789012, 2);
3 INSERT INTO passport VALUES (3, 345678, 3);
```

# Person & Passport

dimon.work

```
Server: 127.0.0.1 » Database: testdb » Table: passport
Browse Structure SQL Search Insert
Run SQL query/queries on table testdb.passport:
1 ALTER TABLE passport
2 ADD COLUMN registration VARCHAR(100);
3
4 UPDATE passport
5 SET registration = CASE
6     WHEN passport_id = 1 THEN 'Winterfell'
7     WHEN passport_id = 2 THEN 'Winterfell'
8     WHEN passport_id = 3 THEN 'King''s Landing'
9     END
10 WHERE passport_id IN (1, 2, 3);
11 |
```

- W tym zapytaniu **SET registration = CASE** pozwala określić różne wartości dla rejestracji w zależności od **passport\_id**.
- Warunek **WHERE passport\_id IN (1, 2, 3)** gwarantuje, że aktualizacja zostanie zastosowana tylko do określonych rekordów.

