

**AKADEMIA HANDLOWA  
NAUK STOSOWANYCH W RADOMIU**



**RADOM  
ACADEMY OF ECONOMICS**

## **Wydział Studiów Strategicznych i Technicznych**

**Kierunek: Informatyka, rok II, semestr III (2021/2022)**

### **BAZ DANYCH (SQL)**

**Prowadzący: dr hab. Filip Rudziński**

**Autor sprawozdania:**

**Magdalena Szafrńska, nr albumu: 18345**

# Spis treści

<b>Wstęp</b>	<b>3</b>
Wykonanie projektu bazy danych	3
Uwaga	3
<b>Cel projektu</b>	<b>3</b>
Użyte narzędzia	4
Projekt bazy danych z użyciem diagramów ERD	4
Wstępna struktura bazodanowa	4
Uruchomienie serwera bazodanowego w narzędziu XAMPP	5
Baza danych w serwerze lokalnym XAMPP	6
Uruchomienie narzędzia phpMyAdmin	6
Utworzenie bazy danych	6
Generowanie podstawowego kodu w MySQL	6
Stworzenie encji	6
Stworzenie relacji pomiędzy encjami	8
Diagram relacji pomiędzy tabelami	9
<b>Wprowadzenie danych do bazy</b>	<b>9</b>
<b>Przykładowe zapytania SQL</b>	<b>11</b>
Influencerzy posortowani malejąco wg liczby followersów (ORDER BY)	11
Akcje wykonane przez influencerów w listopadzie (BETWEEN)	11
Wszystkie produkty z "Reductor" w nazwie (LIKE)	12
Jaka wysyłka, o jakiej wartości i kiedy trafiła do influencerów (AS)	12
Funkcje wbudowane SQL	13
ilość wszystkich produktów (COUNT)	13
średni koszt możliwych do wykonania przez influencerów akcji	13
Wysyłki konkretnych produktów do influencerów	14
wszyscy influencerzy, którym wysłano (LIKE + AND)	14
ograniczenie ilości do 5 (LIMIT)	14
Wszystkie wysyłki do określonego influencerów	15
Aktualizacja imienia influencerów (UPDATE)	15
Influencerzy spoza Polski (NOT IN)	16
Akcje wykonane przez influencerów	17
influencerzy, którzy nie wykonali żadnej akcji (LEFT JOIN)	17
influencerzy, którzy wykonali jakąkolwiek akcję (INNER JOIN)	17
Wyszukanie produktów nigdy nie wysłanych (NOT EXISTS)	18
<b>Widoki</b>	<b>19</b>
<b>Funkcje</b>	<b>20</b>
Połączenie imienia i nazwiska	20
Etykieta dla influencerów wg followersów	20
Wartość wysyłki w kursie euro	22
<b>Procedury</b>	<b>23</b>
Wyświetlanie danych o influencerach	24
Wyświetlanie działań wszystkich influencerów	25
Wyświetlanie działań konkretnego influencerów	26

Wyświetlanie wynagrodzenia dla influencera w danym miesiącu	28
Zmniejszenie ilość towarów w magazynie	29
<b>Obsługa wyjątków</b>	<b>31</b>
<b>Wyzwalacze</b>	<b>32</b>
Wywołanie procedury zmniejszania ilości magazynu	33
Dodanie nowego rekordu	35
<b>Przydzielanie uprawnień użytkownikom</b>	<b>37</b>
Tworzenie nowego użytkownika z uprawnieniami READONLY	38
Tworzenie nowego użytkownika z uprawnieniami 'full access'	41
<b>Pełny skrypt SQL</b>	<b>44</b>

# Wstęp

Podstawowym etapem powstawania bazy danych jest tworzenie jej projektu. Od decyzji podjętych na etapie projektowania zależy jakość i użyteczność stworzonej bazy danych. Baza danych, podobnie jak większość projektów komputerowych, jest modelem wycinka świata rzeczywistego, utworzonym tak, aby był możliwy do zapamiętania przez maszynę cyfrową i zawierał optymalną ilość informacji do zastosowania, jakiemu będzie służył.

Przy modelowaniu baz danych możemy posłużyć się notacją graficzną modelowania danych – diagramem związków encji ERD (ang. Entity-Relationship Diagram). Jest to model sieciowy opisujący na wysokim poziomie abstrakcji dane, które są przechowywane w systemie.

Każda nowo utworzona baza danych wymaga konsekwentnego etapowego działania. Cały proces projektowania bazy danych możemy podzielić na kilka etapów:

- planowanie bazy danych,
  - określenie występujących zbiorów encji,
  - określenie atrybutów przypisanych do poszczególnych encji
  - określenie dziedziny poszczególnych atrybutów
- tworzenie modelu koncepcyjnego (np. diagramu ERD),
- transformacja modelu koncepcyjnego na model relacyjny,
- proces normalizacji bazy danych,
- wybór struktur i określenie zasad dostępu do bazy danych.

## Wykonanie projektu bazy danych

Zaprojektowałam bazę danych do serwisu monitorującego współpracę firmy z influencerami. Model związków encji zawiera 8 encji. Zaprojektowaną bazę planuję w przyszłości rozszerzyć i realnie wykorzystać w mojej pracy na co dzień, gdyż takiego właśnie narzędzia monitorującego brakuje w mojej obecnej firmie.

## Uwaga

Do pracy użyłam bazy danych tworzonej przeze mnie w tym samym czasie na laboratoriach z przedmiotu “Bazy danych (SQL)” pod przewodnictwem mgr. Tomasza Marka. W trakcie pracy nad pierwotną bazą, potrzeby, jakie niniejsza baza miała spełnić, odbiegały nieco od wersji pierwotnej przeznaczonej na laboratoria. W celu dokończenia projektu zaliczeniowego w obrębie konwersatoriów postanowiłam skopiować zawartość do nowej bazy pracując od tego momentu jedynie na niej. Nową bazę nazwałam “db\_influencers\_konw”. Z tego właśnie powodu w początkowych poleceniach i screenach pojawia się nazwa bazy “db\_influencers”, a w późniejszych “db\_influencers\_konw”.

## Cel projektu

Świat cyfrowy daje coraz więcej możliwości do współpracy online. Firmy zawierają porozumienia z osobami obecnymi w social mediach (tzw. influencerami) wynagradzając ich za promowanie produktów marki, wysyłając im swoje produkty do testowania czy choćby nadając im specjalne kody rabatowe na zniżkę w sklepie internetowym. Moim celem jest przygotowanie modelu bazy danych, który będzie umożliwiał m.in.:

- dodawanie i usuwanie influencerów,
- gromadził informację jakie działania marketingowe na rzecz marki wykonał dany influencer,
- jakie wynagrodzenie marka powinna wypłacić influencerowi za określone działania,
- kiedy i jakie produkty zostały wysłane do influencera,
- użycia jego kodu rabatowego,
- przeszukanie osób (influencerów) generujących dla marki największe zyski.

## Użyte narzędzia

- język zapytań SQL
- MySQL – ogólnodostępny system zarządzania relacyjnymi bazami danych, który pomaga użytkownikom przechowywać, uporządkowywać i później pobierać dane
- lokalny serwer XAMPP w wersji v3.3.0
- silnik bazy danych MariaDB w wersji 10.4.21
- phpMyAdmin Version information: 5.1.1 - narzędzie służące do przetwarzania informacji znajdujących się w bazie danych i łatwego zarządzania bazą
- edytor GenMyModel do projektowania baz danych (<https://www.genmymodel.com/>)

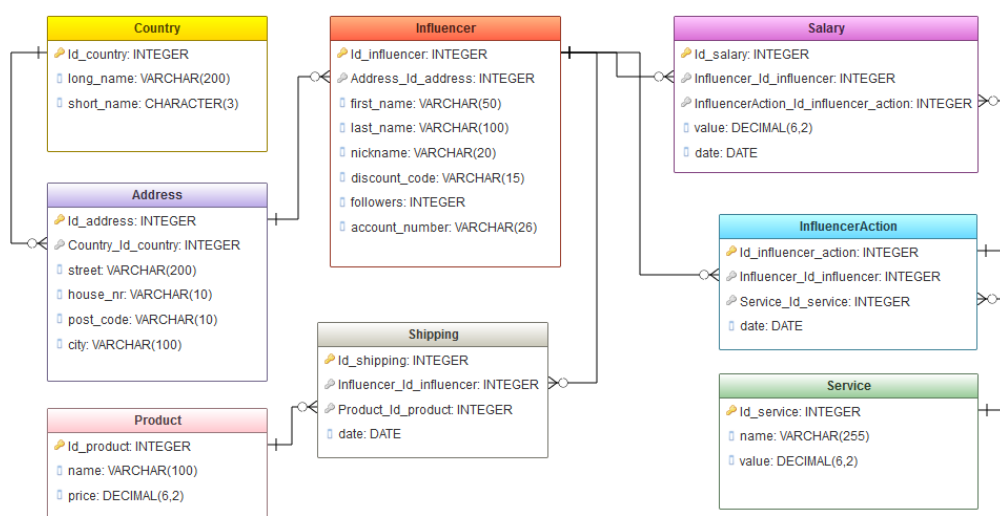
## Projekt bazy danych z użyciem diagramów ERD

Aby wstępnie zobrazować budowę bazy danych, opracowałam diagram związków encji, który będzie jednoznacznie i przejrzysto przedstawiał wymagania firmy w zakresie przetwarzanych przez nią danych oraz umożliwiał zbudowanie na jego podstawie relacyjnej bazy danych.

Do wykonania diagramu ERD użyłam edytora GenMyModel (<https://www.genmymodel.com/>) umożliwiającego projektowanie baz danych online na poziomie tabel i odniesień. Encje przedstawione są za pomocą prostokątów zawierających listę atrybutów. Klucze główne oznaczone są przez żółtą ikonę klucza.

Po wprowadzeniu relacji pomiędzy tabelami zostały dodane klucze obce. Poniżej diagram po wprowadzeniu relacji wraz z opcjonalnością związku oraz pokazaniem rodzaju relacji.

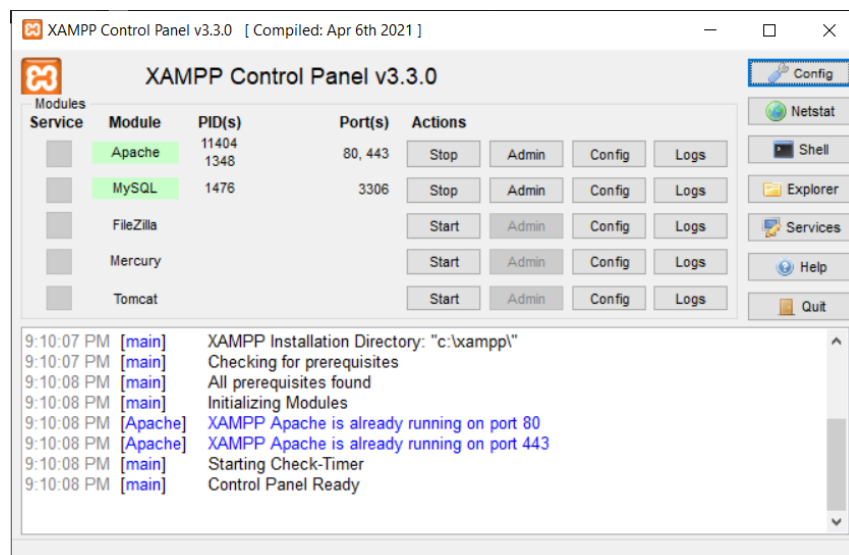
### 1. Wstępna struktura bazodanowa



Tak przygotowany diagram ERD pozwala na późniejszą weryfikację i optymalizację bazy danych, a także stanowi podstawową dokumentację projektowanej bazy danych.

## 2. Uruchomienie serwera bazodanowego w narzędziu XAMPP

W panelu kontrolnym pakietu XAMPP uruchamiam usługę "Apache" oraz "MySQL". Po upewnieniu się, że serwer jest włączony wraz z usługą "MySQL", zamykam panel bo on i tak pozostanie działający w tle. Następnie przechodzę do pakietu phpMyAdmin.



## Baza danych w serwerze lokalnym XAMPP

### 1. Uruchomienie narzędzia phpMyAdmin

Do wykonania niniejszego projektu użyłam zainstalowanego na moim komputerze serwera lokalnego. XAMPP symuluje właśnie taki lokalny serwer dzięki specjalnemu adresowi sieciowemu w mojej karcie sieciowej: 127.0.0.1 (localhost). Wszystkie operacje odbywać się będą lokalnie na moim dysku, bez żadnych opóźnień.

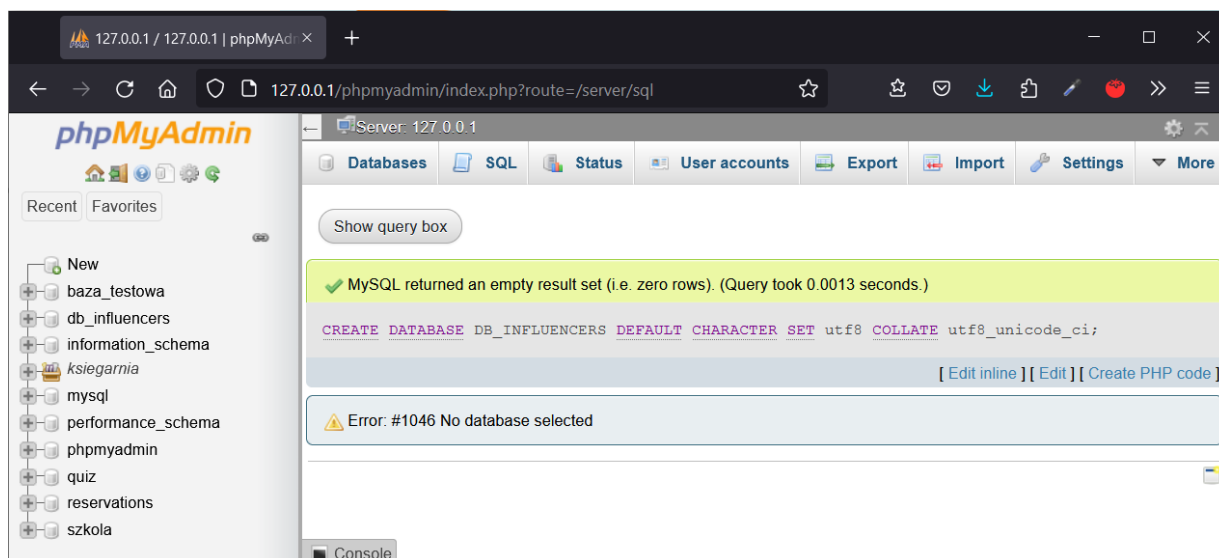
Taki lokalny serwer posłuży mi idealnie do nauki treści z wykładów i wykonania projektu zaliczeniowego. Nic nie stoi na przeszkodzie aby później gotowy serwis przenieść z mojego lokalnego serwera na serwer działający w Internecie i dostępny już dla wszystkich internautów.

### 2. Utworzenie bazy danych

W narzędziu phpMyAdmin utworzyłam nową bazę danych o nazwie DB\_Influencers. Aby zapewnić poprawność wyświetlania polskich znaków zastosowałam dodatkowe polecenia SQL:

```
CREATE DATABASE DB_Influencers DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci;
```

Baza danych została prawidłowo utworzona.



### 3. Generowanie podstawowego kodu w MySQL

#### Stworzenie encji

Skrypt wygenerowany w genMyModel.com podzieliłam na dwie części. Najpierw w oknie SQL wykonałam skrypt dotyczący tworzenia tabel. Kod przedstawiam poniżej. Po zatwierdzeniu tabele zostały pomyślnie utworzone, co również przedstawia poniższy zrzut ekranu.

```

# Create tables
CREATE TABLE IF NOT EXISTS Influencer
(
    Id_influencer INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    Address_Id_address INT,
    first_name VARCHAR(50),
    last_name VARCHAR(100),
    nickname VARCHAR(20),
    discount_code VARCHAR(15),
    followers INT,
    account_number VARCHAR(26)
);

CREATE TABLE IF NOT EXISTS Shipping
(
    Id_shipping INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    Influencer_Id_influencer INT,
    Product_Id_product INT,
    date DATE
);

CREATE TABLE IF NOT EXISTS Country
(
    Id_country INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    long_name VARCHAR(200),
    short_name CHARACTER(3)
);

CREATE TABLE IF NOT EXISTS Salary
(
    Id_salary INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    Influencer_Id_influencer INT,
    InfluencerAction_Id_influencer_action INT,
    value DECIMAL(6, 2),
    date DATE
);

CREATE TABLE IF NOT EXISTS Service
(
    Id_service INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255),
    value DECIMAL(6, 2)
);

CREATE TABLE IF NOT EXISTS InfluencerAction
(
    Id_influencer_action INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    Influencer_Id_influencer INT,
    Service_Id_service INT,
    date DATE
);

CREATE TABLE IF NOT EXISTS Product
(
    Id_product INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100),
    price DECIMAL(6, 2)
);

CREATE TABLE IF NOT EXISTS Address
(
    Id_address INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    Country_Id_country INT,
    street VARCHAR(200),
    house_nr VARCHAR(10),
    post_code VARCHAR(10),
    city VARCHAR(100)
);

```



```

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0130 seconds.)

# Create tables CREATE TABLE IF NOT EXISTS Influencer ( Id_influencer INT NOT NULL PRIMARY KEY AUTO_INCREMENT, Address_Id_address INT,
first_name VARCHAR(50), last_name VARCHAR(100), nickname VARCHAR(20), discount_code VARCHAR(15), followers INT, account_number VARCHAR(26) );

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0118 seconds.)

CREATE TABLE IF NOT EXISTS Shipping ( Id_shipping INT NOT NULL PRIMARY KEY AUTO_INCREMENT, Influencer_Id_influencer INT, Product_Id_product
INT, date DATE );

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0115 seconds.)

CREATE TABLE IF NOT EXISTS Country ( Id_country INT NOT NULL PRIMARY KEY AUTO_INCREMENT, long_name VARCHAR(200), short_name CHARACTER(3) );

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0177 seconds.)

CREATE TABLE IF NOT EXISTS Salary ( Id_salary INT NOT NULL PRIMARY KEY AUTO_INCREMENT, Influencer_Id_influencer INT,
InfluencerAction_Id_influencer_action INT, value DECIMAL(6, 2), date DATE );

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0112 seconds.)

CREATE TABLE IF NOT EXISTS Service ( Id_service INT NOT NULL PRIMARY KEY AUTO_INCREMENT, name VARCHAR(255), value DECIMAL(6, 2) );

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0100 seconds.)

CREATE TABLE IF NOT EXISTS InfluencerAction ( Id_influencer_action INT NOT NULL PRIMARY KEY AUTO_INCREMENT, Influencer_Id_influencer INT,
Service_Id_service INT, date DATE );

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0101 seconds.)

CREATE TABLE IF NOT EXISTS Product ( Id_product INT NOT NULL PRIMARY KEY AUTO_INCREMENT, name VARCHAR(100), price DECIMAL(6, 2) );

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0102 seconds.)

CREATE TABLE IF NOT EXISTS Address ( Id_address INT NOT NULL PRIMARY KEY AUTO_INCREMENT, Country_Id_country INT, street VARCHAR(200), house_nr
VARCHAR(10), post_code VARCHAR(10), city VARCHAR(100) );

[ Edit inline ] [ Edit ] [ Create PHP code ]

```

## Stworzenie relacji pomiędzy encjami

Wprowadziłam skrypt odpowiadający za utworzenie relacji pomiędzy tabelami. Kod poniżej. Po zatwierdzeniu relacje pomiędzy encjami zostały pomyślnie utworzone, co również przedstawia poniższy zrzut ekranu.

```

# Create FKs

ALTER TABLE Influencer
  ADD FOREIGN KEY (Address_Id_address) REFERENCES Address(Id_address)
;

ALTER TABLE Shipping
  ADD FOREIGN KEY (Influencer_Id_influencer) REFERENCES Influencer(Id_influencer),
  ADD FOREIGN KEY (Product_Id_product) REFERENCES Product(Id_product)
;

ALTER TABLE Salary
  ADD FOREIGN KEY (InfluencerAction_Id_influencer_action) REFERENCES InfluencerAction(Id_influencer_action),
  ADD FOREIGN KEY (Influencer_Id_influencer) REFERENCES Influencer(Id_influencer)
;

ALTER TABLE InfluencerAction
  ADD FOREIGN KEY (Influencer_Id_influencer) REFERENCES Influencer(Id_influencer),
  ADD FOREIGN KEY (Service_Id_service) REFERENCES Service(Id_service)
;

ALTER TABLE Address
  ADD FOREIGN KEY (Country_Id_country) REFERENCES Country(Id_country)
;

```

```

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0452 seconds)

+ Create FKs ALTER TABLE Influencer ADD FOREIGN KEY (Address_Id_address) REFERENCES Address(Id_address);

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0502 seconds)

ALTER TABLE Shipping ADD FOREIGN KEY (Influencer_Id_influencer) REFERENCES Influencer(Id_influencer), ADD FOREIGN KEY (Product_Id_product) REFERENCES Product(Id_product);

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0431 seconds)

ALTER TABLE Salary ADD FOREIGN KEY (InfluencerAction_Id_influencer_action) REFERENCES InfluencerAction(Id_influencer_action), ADD FOREIGN KEY (Influencer_Id_influencer) REFERENCES Influencer(Id_influencer);

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0450 seconds)

ALTER TABLE InfluencerAction ADD FOREIGN KEY (Influencer_Id_influencer) REFERENCES Influencer(Id_influencer), ADD FOREIGN KEY (Service_Id_service) REFERENCES Service(Id_service);

[ Edit inline ] [ Edit ] [ Create PHP code ]

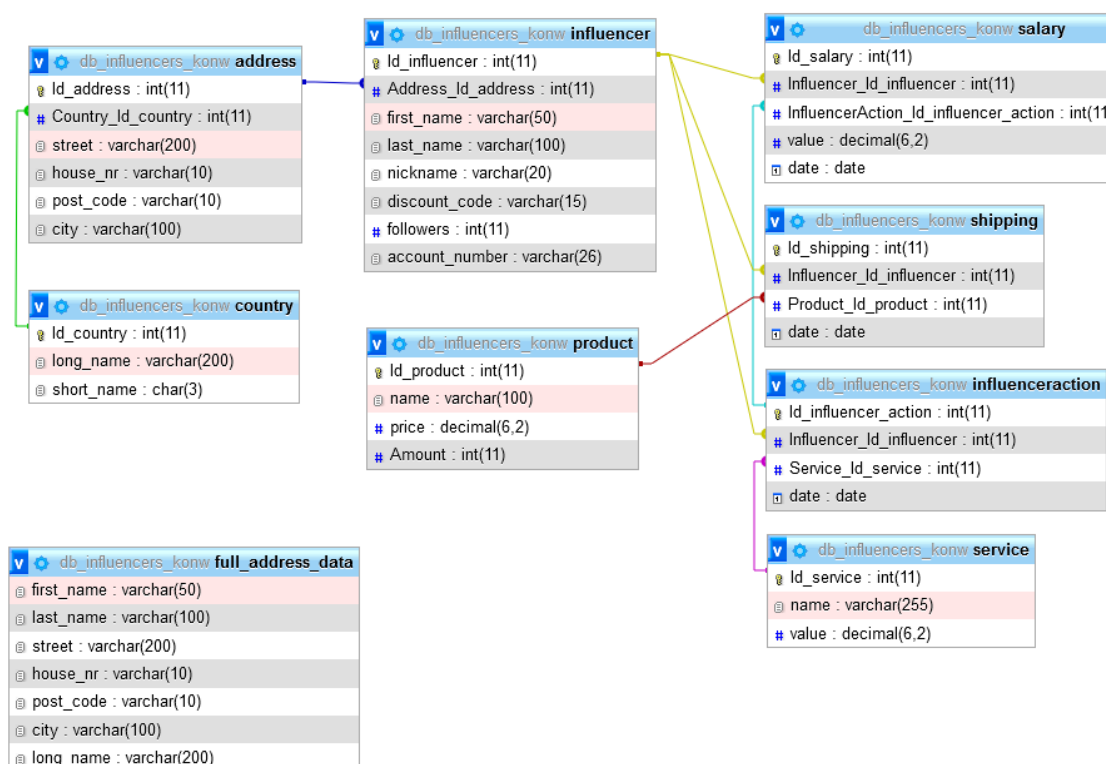
✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0423 seconds)

ALTER TABLE Address ADD FOREIGN KEY (Country_Id_country) REFERENCES Country(Id_country);

[ Edit inline ] [ Edit ] [ Create PHP code ]

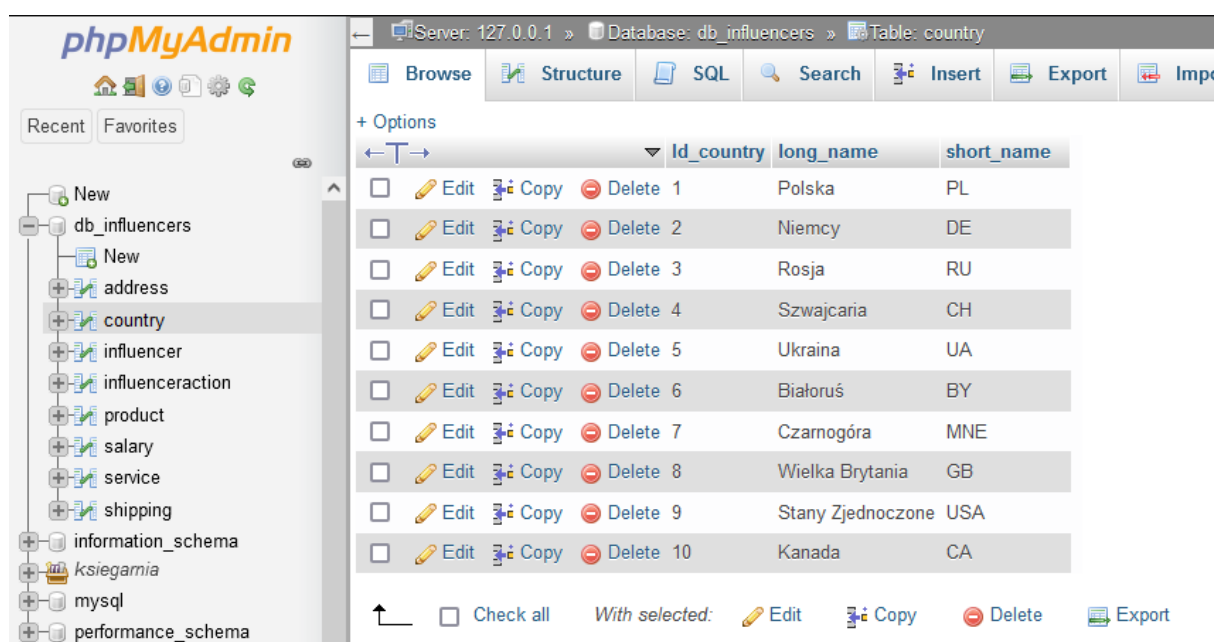
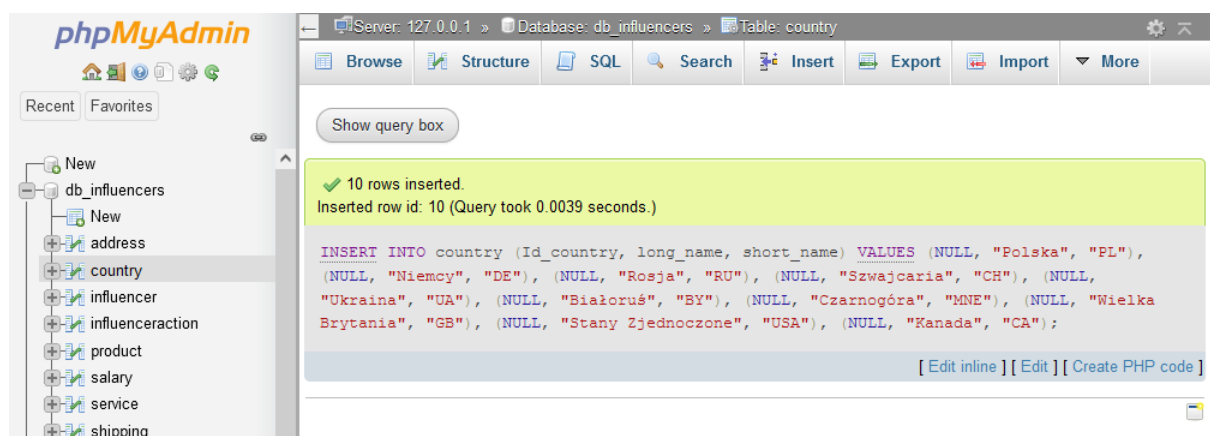
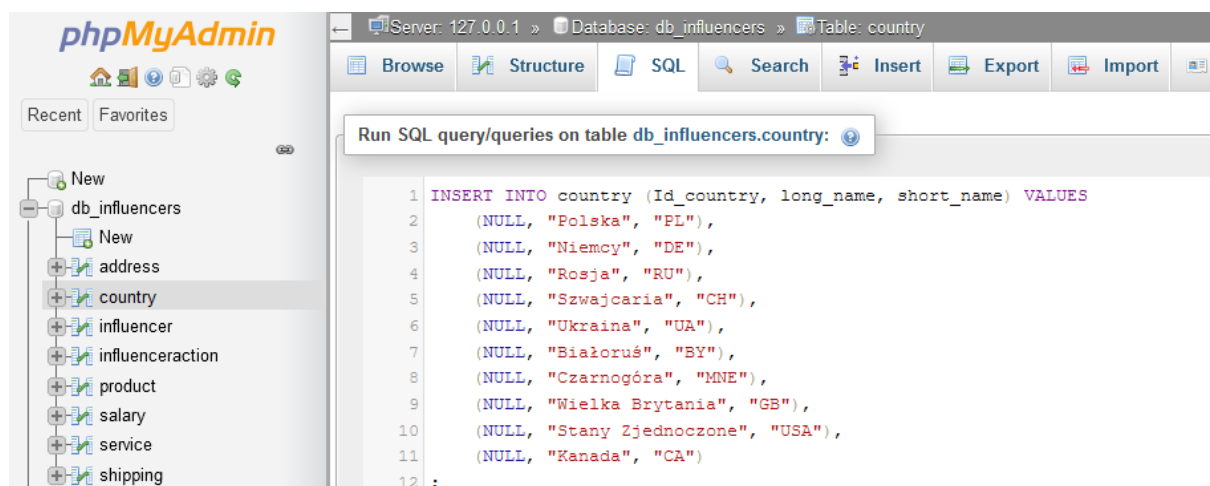
```

## Diagram relacji pomiędzy tabelami



## Wprowadzenie danych do bazy

Używając języka SQL uzupełniłam zaprojektowane tabele danymi. Do każdej z nich wprowadzam co najmniej po 10 rekordów. Poniżej przykładowa procedura uzupełniania tabeli "Country". Dla pozostałych tabel proces był analogiczny więc zamieszczę jedynie skrypt SQL dla każdej z nich. Pełny kod znajduje się na końcu sprawozdania w zakładce [Pełny skrypt SQL](#).



## Przykładowe zapytania SQL

Używając języka SQL stworzyłam ponad 10 dowolnych zapytań do zaprojektowanej bazy. Podczas tworzenia zapytań złożonych dostarczyłam listę wszystkich relacji, które zachodzą pomiędzy używanymi w tym zapytaniu tabelami. Tę listę umieszczałam po klauzuli WHERE.

W przykładach użyłam zarówno zapytań złożonych jak i skorelowanych.

## 1. Influencerzy posortowani malejąco wg liczby followersów (ORDER BY)

### POLECENIE SQL:

```
SELECT inf.first_name, inf.last_name, inf.followers FROM influencer AS inf  
ORDER BY inf.followers DESC;
```

Showing rows 0 - 9 (10 total, Query took 0.0004 seconds.) [followers: 264000... - 632...]

```
SELECT inf.first_name, inf.last_name, inf.followers FROM influencer AS inf ORDER BY inf.followers DESC;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

+ Options

	first_name	last_name	followers
<input type="checkbox"/> Edit Copy Delete	Paulina	Guzińska	264000
<input type="checkbox"/> Edit Copy Delete	Aleksander	Mąkosa	89000
<input type="checkbox"/> Edit Copy Delete	Marcin	Wiesiuk	65002
<input type="checkbox"/> Edit Copy Delete	Maria	Wrześniak	53000
<input type="checkbox"/> Edit Copy Delete	Ismena	Stelmaszczyk	47600
<input type="checkbox"/> Edit Copy Delete	John	Smith	45000
<input type="checkbox"/> Edit Copy Delete	Rafał	Piotrowski	7522
<input type="checkbox"/> Edit Copy Delete	Klaudia	Michalak	4123
<input type="checkbox"/> Edit Copy Delete	Bartosz	Smęda	801
<input type="checkbox"/> Edit Copy Delete	Aleksa	Woźnicki	632

## 2. Akcje wykonane przez influencerów w listopadzie (BETWEEN)

### POLECENIE SQL:

```
SELECT * FROM influenceraction WHERE influenceraction.date  
BETWEEN "2021-11-01" AND "2021-11-30";
```

Showing rows 0 - 3 (4 total, Query took 0.0013 seconds.)

```
SELECT * FROM influenceraction WHERE influenceraction.date BETWEEN "2021-11-01" AND "2021-11-30";
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

+ Options

	Id_influencer_action	Influencer_Id_influencer	Service_Id_service	date
<input type="checkbox"/> Edit Copy Delete	5	4	10	2021-11-02
<input type="checkbox"/> Edit Copy Delete	6	3	10	2021-11-14
<input type="checkbox"/> Edit Copy Delete	8	7	5	2021-11-02
<input type="checkbox"/> Edit Copy Delete	9	10	9	2021-11-03

☐ Check all | With selected: Edit Copy Delete Export

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

## 3. Wszystkie produkty z "Reductor" w nazwie (LIKE)

### POLECENIE SQL:

```
SELECT * FROM product WHERE product.name LIKE "%Reductor%";
```

✓ Showing rows 0 - 2 (3 total, Query took 0.0004 seconds.)

```
SELECT * FROM product WHERE product.name LIKE "%Reductor%";
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

				Id_product	name	price
<input type="checkbox"/>	Edit	Copy	Delete	1	Modify Reductor	129.00
<input type="checkbox"/>	Edit	Copy	Delete	7	Pełna kuracja Modify Reductor	387.00
<input type="checkbox"/>	Edit	Copy	Delete	9	Zestaw świąteczny Modify Reductor	199.00

↑ ☐ Check all With selected: Edit Copy Delete

#### 4. Jaka wysyłka, o jakiej wartości i kiedy trafiła do influencerów (AS)

##### POLECENIE SQL:

```
SELECT sh.Id_shipping, FullName(inf.first_name, inf.last_name)
AS FullName, sh.Product_Id_product, sh.date, pr.price
FROM influencer AS inf, shipping AS sh, product AS pr
WHERE inf.Id_influencer = sh.Influencer_Id_influencer
AND pr.Id_product = sh.Product_Id_product;
```

✓ Showing rows 0 - 11 (12 total, Query took 0.0008 seconds.)

```
SELECT sh.Id_shipping, FullName(inf.first_name, inf.last_name) AS FullName,
sh.Product_Id_product, sh.date, pr.price FROM influencer AS inf, shipping
AS sh, product AS pr WHERE inf.Id_influencer = sh.Influencer_Id_influencer
AND pr.Id_product = sh.Product_Id_product;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

Id_shipping	FullName	Product_Id_product	date	price
1	Ismenka Stelmaszczyk	7	2021-09-08	387.00
2	Paulina Guzińska	7	2021-11-07	387.00
3	Bartosz Smęda	9	2021-08-18	199.00
4	John Smith	1	2021-11-19	129.00
5	Maria Wrześniak	2	2021-10-18	139.00
6	Maria Wrześniak	5	2021-10-18	19.00
7	Klaudia Michalak	10	2020-12-12	256.00
8	Aleksander Mąkosa	9	2021-11-17	199.00
9	Aleksa Woźnicki	2	2021-07-11	139.00
10	Rafał Piotrowski	1	2021-11-06	129.00
11	Marcin Wiesiuk	9	2021-11-15	199.00
12	Ismenka Stelmaszczyk	4	2021-11-08	49.00

☐ Show all | Number of rows: 25 | Filter rows: Search this table

## 5. Funkcje wbudowane SQL

### a. ilość wszystkich produktów (COUNT)

#### POLECENIE SQL:

```
SELECT COUNT(*) FROM product;
```

Your SQL query has been executed successfully.

```
SELECT COUNT(*) FROM product;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

+ Options

COUNT(\*)

10

### b. średni koszt możliwych do wykonania przez influencera akcji

#### POLECENIE SQL:

```
SELECT AVG(value) FROM service;
```

✓ Showing rows 0 - 0 (1 total, Query took 0.0004 seconds.)

```
SELECT AVG(value) FROM service;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

AVG(value)

177.000000

## 6. Wysyłki konkretnych produktów do influencerów

### a. wszyscy influencerzy, którym wysłano (LIKE + AND)

#### POLECENIE SQL:

```
SELECT inf.first_name, inf.last_name, p.name, sh.date
FROM influencer AS inf, product AS p, shipping AS sh
WHERE (p.name LIKE "%Reductor%" OR p.name LIKE "%Femibra%")
AND inf.Id_influencer = sh.Influencer_Id_influencer
AND sh.Product_Id_product = p.Id_product
LIMIT 5;
```

✓ Showing rows 0 - 9 (10 total, Query took 0.0005 seconds.)

```
SELECT inf.first_name, inf.last_name, p.name, sh.date FROM influencer
AS inf, product AS p, shipping AS sh WHERE (p.name LIKE "%Reductor%"
OR p.name LIKE "%Femibra%") AND inf.Id_influencer =
sh.Influencer_Id_influencer AND sh.Product_Id_product = p.Id_product;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

first_name	last_name	name	date
Ismena	Stelmaszczyk	Pełna kuracja Modify Reductor	2021-09-08
Paulina	Guzińska	Pełna kuracja Modify Reductor	2021-11-07
Bartosz	Smęda	Zestaw świąteczny Modify Reductor	2021-08-18
John	Smith	Modify Reductor	2021-11-19
Maria	Wrześniak	Modify Femibra	2021-10-18
Klaudia	Michalak	Zestaw świąteczny Modify Femibra	2020-12-12
Aleksander	Mąkosa	Zestaw świąteczny Modify Reductor	2021-11-17
Aleksa	Woźnicki	Modify Femibra	2021-07-11
Rafał	Piotrowski	Modify Reductor	2021-11-06
Marcin	Wiesiuk	Zestaw świąteczny Modify Reductor	2021-11-15

☐ Show all | Number of rows: 25 | Filter rows: Search this table

b. ograniczenie ilości do 5 (LIMIT)

## POLECENIE SQL:

```
SELECT inf.first_name, inf.last_name, p.name, sh.date
FROM influencer AS inf, product AS p, shipping AS sh
WHERE (p.name LIKE "%Reductor%" OR p.name LIKE "%Femibra%")
AND inf.Id_influencer = sh.Influencer_Id_influencer
AND sh.Product_Id_product = p.Id_product
LIMIT 5;
```

✓ Showing rows 0 - 4 (5 total, Query took 0.0004 seconds.)

```
SELECT inf.first_name, inf.last_name, p.name, sh.date FROM influencer AS inf, product AS p,
shipping AS sh WHERE (p.name LIKE "%Reductor%" OR p.name LIKE "%Femibra%") AND
inf.Id_influencer = sh.Influencer_Id_influencer AND sh.Product_Id_product = p.Id_product LIMIT
5;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

+ Options

first_name	last_name	name	date
Ismenka	Stelmaszczyk	Pełna kuracja Modify Reductor	2021-09-08
Paulina	Guzińska	Pełna kuracja Modify Reductor	2021-11-07
Bartosz	Smęda	Zestaw świąteczny Modify Reductor	2021-08-18
John	Smith	Modify Reductor	2021-11-19
Maria	Wrześniak	Modify Femibra	2021-10-18

Query results operations

## 7. Wszystkie wysyłki do określonego influencera

### POLECENIE SQL:

```
SELECT inf.first_name, inf.last_name, sh.date, p.name
FROM influencer AS inf, shipping AS sh, product AS p
WHERE inf.Id_influencer = 1
AND inf.Id_influencer = sh.Influencer_Id_influencer
AND sh.Product_Id_product = p.Id_product;
```

✓ Showing rows 0 - 1 (2 total, Query took 0.0006 seconds.)

```
SELECT inf.first_name, inf.last_name, sh.date, p.name FROM influencer AS inf,
shipping AS sh, product AS p WHERE inf.Id_influencer = 1 AND
inf.Id_influencer = sh.Influencer_Id_influencer AND sh.Product_Id_product =
p.Id_product;
```

☐ Profiling [ [Edit inline](#) ] [ [Edit](#) ] [ [Explain SQL](#) ] [ [Create PHP code](#) ] [ [Refresh](#) ]

☐ Show all | Number of rows: 25 | Filter rows:

+ Options

first_name	last_name	date	name
Ismena	Stelmaszczyk	2021-09-08	Pełna kuracja Modify Reductor
Ismena	Stelmaszczyk	2021-11-08	Gumka do włosów - Iniana XL

☐ Show all | Number of rows: 25 | Filter rows:

## 8. Aktualizacja imienia influencera (UPDATE)

### POLECENIE SQL:

```
UPDATE influencer SET influencer.first_name = "Ismenka"
WHERE influencer.Id_influencer = 1;
```

✓ 1 row affected. (Query took 0.0033 seconds.)

```
UPDATE influencer SET influencer.first_name = "Ismenka"
WHERE influencer.Id_influencer = 1;
```

[ [Edit inline](#) ] [ [Edit](#) ] [ [Create PHP code](#) ]

Wykonanie ponownie skryptu z poprzedniego punktu dla sprawdzenia.



✓ Showing rows 0 - 1 (2 total, Query took 0.0007 seconds.)

```
SELECT inf.first_name, inf.last_name, sh.date, p.name FROM influencer AS inf,
shipping AS sh, product AS p WHERE inf.Id_influencer = 1 AND inf.Id_influencer
= sh.Influencer_Id_influencer AND sh.Product_Id_product = p.Id_product;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

first_name	last_name	date	name
Ismenka	Stelmaszczyk	2021-09-08	Pełna kuracja Modify Reductor
Ismenka	Stelmaszczyk	2021-11-08	Gumka do włosów - Iniana XL

☐ Show all | Number of rows: 25 | Filter rows: Search this table

## 9. Influencerzy spoza Polski (NOT IN)

### POLECENIE SQL:

```
SELECT inf.first_name, inf.last_name, a.street, a.house_nr, a.post_code, a.city,
c.long_name, c.short_name
FROM influencer AS inf, country AS c, address AS a
WHERE c.short_name NOT IN ("PL")
AND inf.Address_Id_address = a.Id_address
AND a.Country_Id_country = c.Id_country;
```

✓ Showing rows 0 - 2 (3 total, Query took 0.0005 seconds.)

```
SELECT inf.first_name, inf.last_name, a.street, a.house_nr, a.post_code, a.city, c.long_name,
c.short_name FROM influencer AS inf, country AS c, address AS a WHERE c.short_name NOT IN ("PL")
AND inf.Address_Id_address = a.Id_address AND a.Country_Id_country = c.Id_country;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

first_name	last_name	street	house_nr	post_code	city	long_name	short_name
Paulina	Guzińska	Isern-Hinnerk-Weg	14B	22457	Hamburg	Niemcy	DE
John	Smith	High Park Road	20A	PR9 7QL	Southport	Wielka Brytania	GB
Aleksa	Woźnicki	Saxonbury Way	94	PE2 9FB	Cambridgeshire	Wielka Brytania	GB

☐ Show all | Number of rows: 25 | Filter rows: Search this table

## 10. Akcje wykonane przez influencerów

a. influencerzy, którzy nie wykonali żadnej akcji (LEFT JOIN)

### POLECENIE SQL:

```
SELECT i.Id_influencer, i.first_name, i.last_name
FROM influencer AS i LEFT JOIN influenceraction AS ia
ON i.Id_influencer = ia.Influencer_Id_influencer
WHERE ia.Influencer_Id_influencer IS NULL;
```

✓ Showing rows 0 - 1 (2 total, Query took 0.0004 seconds.)

```
SELECT i.Id_influencer, i.first_name, i.last_name FROM influencer AS i LEFT JOIN influenceraction AS ia
ON i.Id_influencer = ia.Influencer_Id_influencer WHERE ia.Influencer_Id_influencer IS NULL;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

+ Options

Id_influencer	first_name	last_name
6	Klaudia	Michalak
9	Rafał	Piotrowski

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

b. influencerzy, którzy wykonali jakąkolwiek akcję (INNER JOIN)

### POLECENIE SQL:

```
SELECT * FROM influencer AS i
INNER JOIN influenceraction AS ia
ON i.Id_influencer = ia.Influencer_Id_influencer;
```

✓ Showing rows 0 - 9 (10 total, Query took 0.0004 seconds.) [Id\_influencer: 1... - 10...]

```
SELECT i.Id_influencer, i.first_name, i.last_name, ia.Service_Id_service, ia.date FROM influencer AS i
INNER JOIN influenceraction AS ia ON i.Id_influencer = ia.Influencer_Id_influencer ORDER BY
`i`.`Id_influencer` ASC
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

Id_influencer	first_name	last_name	Service_Id_service	date
1	Ismenka	Stelmaszczyk	8	2021-09-30
1	Ismenka	Stelmaszczyk	8	2021-09-30
1	Ismenka	Stelmaszczyk	1	2021-09-15
2	Paulina	Guzińska	7	2021-10-13
3	Bartosz	Smęda	10	2021-11-14
4	John	Smith	10	2021-11-02
5	Maria	Wrześniak	3	2021-10-19
7	Aleksander	Makosa	5	2021-11-02
8	Aleksa	Woźnicki	6	2021-08-10
10	Marcin	Wiesiuk	9	2021-11-03

☐ Show all | Number of rows: 25 | Filter rows: Search this table

## 11. Wyszukanie produktów nigdy nie wysłanych (NOT EXISTS)

### POLECENIE SQL:

```
SELECT p.name, p.price FROM product AS p WHERE NOT EXISTS
(SELECT * FROM shipping AS s WHERE s.Product_Id_product = p.Id_product);
```

✓ Showing rows 0 - 2 (3 total, Query took 0.0014 seconds.)

```
SELECT p.name, p.price FROM product AS p WHERE NOT EXISTS (SELECT * FROM shipping AS s WHERE s.Product_Id_product = p.Id_product);
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

+ Options

		name	price
<input type="checkbox"/>	Edit	Gumka do włosów - Iniana	29.00
<input type="checkbox"/>	Edit	Gumka do włosów - welurowa XL	29.00
<input type="checkbox"/>	Edit	Pełna kuracja Modify Femibra	417.00

↑ ☐ Check all With selected: Edit Copy Delete Export

## Widoki

Widoki (perspektywy) w języku SQL to wirtualne tabele tworzone na podstawie zapytań. Składają się z kolumn i wierszy pobranych z prawdziwych tabel. Pokazywane w widoku dane są zawsze aktualne, ponieważ widoki są tworzone w momencie wykonania zapytania. Widoki nie przechowują zapisanych w tabelach danych.

Aby zaprezentować tę funkcjonalność stworzyłam widok z odpowiednich kolumn oparty na następujących tabelach:

- tabela "influencer"
  - kolumna "first\_name"
  - kolumna "last\_name"
- tabela "address"
  - kolumna "street"
  - kolumna "house\_nr"
  - kolumna "post\_code"
  - kolumna "city"
- tabela "country"
  - kolumna "long\_name"

Kod do wygenerowanie widoku:

```
SELECT i.first_name AS first_name, i.last_name AS last_name, a.street AS street,
a.house_nr AS house_nr, a.post_code AS post_code, a.city AS city,
c.long_name AS long_name
FROM ((db_influencers_konw.influencer i left join db_influencers_konw.address a
on(i.Address_Id_address = a.Id_address)) left join db_influencers_konw.country c
on(a.Country_Id_country = c.Id_country))
```

Stworzony widok wygenerował tabelę jak na poniższym zrzucie ekranu.

### POLECENIE SQL:

```
SELECT * FROM full_address_data;
```

Showing rows 0 - 9 (10 total, Query took 0.0005 seconds)

```
SELECT * FROM `full_address_data`
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

	first_name	last_name	street	house_nr	post_code	city	long_name
<input type="checkbox"/> Edit Copy Delete	Ismenka	Stelmaszczyk	Jurajska	4/59	02-699	Warszawa	Polska
<input type="checkbox"/> Edit Copy Delete	Paulina	Guzińska	Isern-Hinnerk-Weg	14B	22457	Hamburg	Niemcy
<input type="checkbox"/> Edit Copy Delete	Bartosz	Smęda	Kadłubka	42/5	71-524	Szczecin	Polska
<input type="checkbox"/> Edit Copy Delete	John	Smith	High Park Road	20A	PR9 7QL	Southport	Wielka Brytania
<input type="checkbox"/> Edit Copy Delete	Maria	Wrześniak	Puławska	38/15	03-232	Warszawa	Polska
<input type="checkbox"/> Edit Copy Delete	Klaudia	Michalak	Aleja Krakowska	49	05-090	Sękocin Stary	Polska
<input type="checkbox"/> Edit Copy Delete	Aleksander	Mąkosa	Nowowiejska	15	06-500	Mława	Polska
<input type="checkbox"/> Edit Copy Delete	Aleksa	Woźnicki	Saxonbury Way	94	PE2 9FB	Cambridgeshire	Wielka Brytania
<input type="checkbox"/> Edit Copy Delete	Rafał	Piotrowski	Marii Dąbrowskiej	96/9	97-300	Piotrków Trybunalski	Polska
<input type="checkbox"/> Edit Copy Delete	Marcin	Wiesiuk	Niemeńska	66	60-412	Poznań	Polska

☐ Check all | With selected: Edit Copy Delete Export

## Funkcje

Funkcji używa się, aby tworzyć dedykowane rozwiązania. Funkcjom, tak jak procedurom, można przekazać pewną liczbę parametrów. Funkcja jednak nie tylko wykonuje pewne operacje, ale także zwraca obliczony na podstawie przekazanych parametrów wynik.

### 1. Połączenie imienia i nazwiska

Funkcja wyświetlająca imię i nazwisko jako jedną frazę. Często używam tej funkcji w bazie więc zdecydowałam się na funkcję skracającą mi czas wywoływania imienia i nazwiska osobno za każdym razem.

#### POLECENIE SQL:

```
CREATE FUNCTION FullName(first_name VARCHAR(20), last_name VARCHAR(30))
RETURNS VARCHAR(50)
DETERMINISTIC
RETURN CONCAT(first_name, ' ', last_name);
```

Showing rows 0 - 9 (10 total, Query took 0.0007 seconds)

```
SELECT FullName(first_name, last_name) FROM influencer;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25

+ Options

**FullName(first\_name, last\_name)**

- Ismenka Stelmaszczyk
- Paulina Guzińska
- Bartosz Smęda
- John Smith
- Maria Wrześniak
- Klaudia Michalak
- Aleksander Mąkosa
- Aleksa Woźnicki
- Rafał Piotrowski
- Marcin Wiesiuk

## 2. Etykieta dla influencera wg followersów

Przypisanie influencerowi odpowiedniej “etykiety” w zależności od liczby jego followersów.

### POLECENIE SQL:

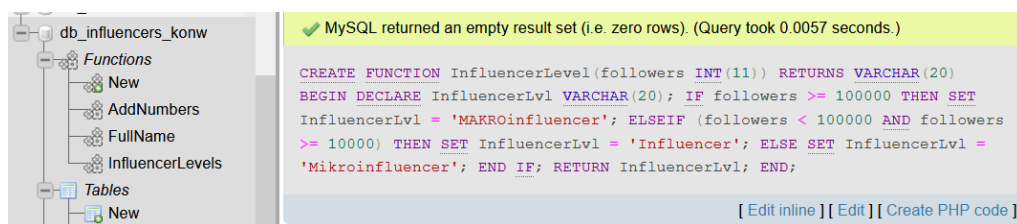
```
DELIMITER //
CREATE FUNCTION InfluencerLevel(followers INT(11))
RETURNS VARCHAR(20)

BEGIN
    DECLARE InfluencerLvl VARCHAR(20);

    IF followers >= 100000 THEN SET InfluencerLvl = 'MAKROinfluencer';
    ELSEIF (followers < 100000 AND followers >= 10000) THEN SET InfluencerLvl
= 'Influencer';
    ELSE SET InfluencerLvl = 'Mikroinfluencer';
    END IF;

    RETURN InfluencerLvl;
END //
DELIMITER ;
```

Kod został zaimplementowany pomyślnie (screen poniżej).



Poniżej dodatkowo podgląd wyeksportowanej funkcji.



Połączenie dwóch powyższych funkcji pozwoliło mi wygenerować zapytanie złożone. Korzystając w nim z pierwszej funkcji wyświetliłam imiona i nazwiska wszystkich influencerów z tabeli Influencerzy. Druga funkcja natomiast przyporządkowała danego influencera w osobnej kolumnie *fn* do jednej z trzech grup liczności (w zależności od liczby followersów tegoż influencera).

## POLECENIE SQL:

```
SELECT FullName(influencer.first_name, influencer.last_name),  
InfluencerLevel(influencer.followers) AS fn FROM `influencer`;
```

✓ Showing rows 0 - 9 (10 total, Query took 0.0012 seconds.)

```
SELECT FullName(influencer.first_name, influencer.last_name),  
InfluencerLevel(influencer.followers) AS fn FROM `influencer`;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

FullName(influencer.first_name, influencer.last_name)	fn
Ismenka Stelmaszczyk	Influencer
Paulina Guzińska	MAKROinfluencer
Bartosz Smeđa	Mikroinfluencer
John Smith	Influencer
Maria Wrześniak	Influencer
Klaudia Michalak	Mikroinfluencer
Aleksander Mąkosa	Influencer
Aleksa Woźnicki	Mikroinfluencer
Rafał Piotrowski	Mikroinfluencer
Marcin Wiesiuk	Influencer

☐ Show all | Number of rows: 25 | Filter rows: Search this table

### 3. Wartość wysyłki w kursie euro

W funkcji przekazuję wartość wysyłki w złotych i podaję na bieżąco kurs euro. W przyszłości planuję rozwinąć funkcjonalność i pobierać bieżący kurs euro z ogólnodostępnych źródeł.

## POLECENIE SQL:

```
CREATE FUNCTION ValueToEuro(prodValue decimal(6,2), newValue decimal(6,2))  
RETURNS decimal(6,2)  
RETURN prodValue / newValue;
```

Kod został zaimplementowany pomyślnie (screen poniżej).

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0059 seconds.)

```
CREATE FUNCTION ValueToEuro(prodValue decimal(6,2), newValue decimal(6,2))  
RETURNS decimal(6,2) RETURN prodValue / newValue;
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

Poniżej dodatkowo podgląd wyeksportowanej funkcji.

```
Export of routine `ValueToEuro`

1 DELIMITER $$
2 CREATE DEFINER='root'@'localhost' FUNCTION `ValueToEuro`(prodValue decimal(6,2), newValue decimal(6,2)) RETURNS decimal(6,2)
3 RETURN prodValue / newValue$$
4 DELIMITER ;
```

Close

Połączenie pierwszej oraz niniejszej funkcji pozwoliło mi wygenerować zapytanie złożone. Wyświetliłam jaka wysyłka, o jakiej wartości i kiedy trafiła do danego influencera. Korzystając z pierwszej funkcji wyświetliłam imiona i nazwiska wszystkich influencerów z tabeli Influencerzy w osobnej kolumnie *FullName*. Ostatnio stworzona funkcja natomiast policzyła wartość wysyłki w innej walucie niż pierwotnie, czyli zamieniła złotówki na euro z dokładnością do 2 miejsc po przecinku.

```
SELECT sh.Id_shipping, FullName(inf.first_name, inf.last_name) AS FullName,
sh.Product_Id_product, sh.date, pr.price, ValueToEuro(pr.price, 4.53) AS EUR
FROM influencer AS inf, shipping AS sh, product AS pr
WHERE inf.Id_influencer = sh.Influencer_Id_influencer
AND pr.Id_product = sh.Product_Id_product;
```

✓ Showing rows 0 - 11 (12 total, Query took 0.0009 seconds.)

```
SELECT sh.Id_shipping, FullName(inf.first_name, inf.last_name) AS FullName,
sh.Product_Id_product, sh.date, pr.price, ValueToEuro(pr.price, 4.53) AS EUR
FROM influencer AS inf, shipping AS sh, product AS pr WHERE inf.Id_influencer
= sh.Influencer_Id_influencer AND pr.Id_product = sh.Product_Id_product;
```

☐ Profiling [ [Edit inline](#) ] [ [Edit](#) ] [ [Explain SQL](#) ] [ [Create PHP code](#) ] [ [Refresh](#) ]

☐ Show all | Number of rows: 25 v | Filter rows:

+ Options

Id_shipping	FullName	Product_Id_product	date	price	EUR
1	Ismenka Stelmaszczyk	7	2021-09-08	387.00	85.43
2	Paulina Guzińska	7	2021-11-07	387.00	85.43
3	Bartosz Smęda	9	2021-08-18	199.00	43.93
4	John Smith	1	2021-11-19	129.00	28.48
5	Maria Wrześniak	2	2021-10-18	139.00	30.68
6	Maria Wrześniak	5	2021-10-18	19.00	4.19
7	Klaudia Michalak	10	2020-12-12	256.00	56.51
8	Aleksander Mąkosa	9	2021-11-17	199.00	43.93
9	Aleksa Woźnicki	2	2021-07-11	139.00	30.68
10	Rafał Piotrowski	1	2021-11-06	129.00	28.48
11	Marcin Wiesiuk	9	2021-11-15	199.00	43.93
12	Ismenka Stelmaszczyk	4	2021-11-08	49.00	10.82

## Procedury

Procedury ograniczają liczbę danych, które są przesyłane między serwerem bazy danych a klientem. Ograniczają więc również obciążenie serwera bazy danych, gdyż serwer taki realizuje mniej połączeń.

Zarówno funkcja jak i procedura to ręcznie wykonywane programy. Można je wyjaśnić na podstawie analogii: jeśli chciałabym odczytać pensję influencera, to skorzystałabym z

funkcji. Jeśli jednak chciałabym ją zmodyfikować, to skorzystałabym z procedury. To tak jak *getter* i *setter* w językach programowania.

Funkcje od procedur różnią się głównie tym, że funkcja musi zwracać wartość (w klauzuli `return`), a procedura może (za pomocą parametru `out`) ale nie musi. Dodatkowo, funkcje mogą być wywoływane z procedury ale procedura nie może być wywołana z funkcji.

Procedury mogą wykorzystywać/wykonywać polecenia SQL, a funkcje niekoniecznie.

## 1. Wyświetlanie danych o influencerach

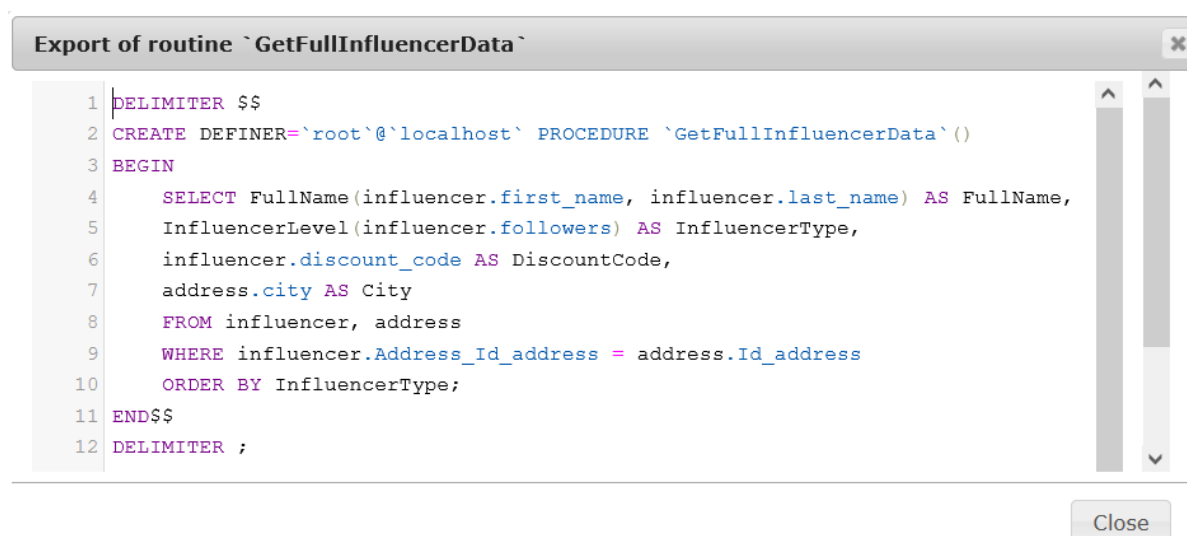
W oparciu o wcześniej utworzone funkcje, procedura ta wyświetla podstawowe dane o wszystkich influencerach z tabeli `Influencerzy`, imię i nazwisko (korzystając z funkcji `FullName()`), etykietę influencera (korzystając z funkcji `InfluencerLevel()`), jego kod zniżkowy oraz miasto, z którego pochodzi.

Procedura `GetFullInfluencerData()` nic nie pobiera ani nie zwraca. Wyświetla jedynie dane za pomocą poleceń SQL umieszczonych w jej wnętrzu.

### POLECENIE SQL:

```
DELIMITER //
CREATE PROCEDURE GetFullInfluencerData()
BEGIN
    SELECT FullName(influencer.first_name, influencer.last_name) AS FullName,
    InfluencerLevel(influencer.followers) AS InfluencerType,
    influencer.discount_code AS DiscountCode,
    address.city AS City
    FROM influencer, address
    WHERE influencer.Address_Id_address = address.Id_address
    ORDER BY InfluencerType;
END//
DELIMITER ;
```

Poniżej dodatkowo podgląd wyeksportowanej funkcji.

The screenshot shows a window titled "Export of routine `GetFullInfluencerData`". Inside, the SQL code for the procedure is displayed with line numbers from 1 to 12. The code is: 1 DELIMITER \$\$, 2 CREATE DEFINER='root'@'localhost' PROCEDURE `GetFullInfluencerData`() , 3 BEGIN, 4 SELECT FullName(influencer.first\_name, influencer.last\_name) AS FullName, , 5 InfluencerLevel(influencer.followers) AS InfluencerType, , 6 influencer.discount\_code AS DiscountCode, , 7 address.city AS City , 8 FROM influencer, address , 9 WHERE influencer.Address\_Id\_address = address.Id\_address , 10 ORDER BY InfluencerType; , 11 END\$\$, 12 DELIMITER ;. There are vertical scroll bars on the right side of the code area. A "Close" button is located at the bottom right of the window.

```
1 DELIMITER $$
2 CREATE DEFINER='root'@'localhost' PROCEDURE `GetFullInfluencerData`()
3 BEGIN
4     SELECT FullName(influencer.first_name, influencer.last_name) AS FullName,
5     InfluencerLevel(influencer.followers) AS InfluencerType,
6     influencer.discount_code AS DiscountCode,
7     address.city AS City
8     FROM influencer, address
9     WHERE influencer.Address_Id_address = address.Id_address
10    ORDER BY InfluencerType;
11 END$$
12 DELIMITER ;
```

Utworzoną procedurę wywołałam poniższym poleceniem.

```
CALL GetFullInfluencerData();
```



Zrzut wyniku działania procedury GetFullInfluencerData() poniżej.

✓ Showing rows 0 - 9 (10 total, Query took 0.0014 seconds.)

CALL GetFullInfluencerData();

[ Edit inline ] [ Edit ] [ Create PHP code ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

FullName	InfluencerType	DiscountCode	City
Marcin Wiesiuk	Influencer	marcinek15	Poznań
Ismenka Stelmaszczyk	Influencer	ismena15	Warszawa
Aleksander Mąkosa	Influencer	alexi15	Mława
John Smith	Influencer	john15	Southport
Maria Wrześniak	Influencer	marysia15	Warszawa
Paulina Guzińska	MAKROinfluencer	paulina15	Hamburg
Klaudia Michałak	Mikroinfluencer	klaudia15	Sękocin Stary
Bartosz Smeđa	Mikroinfluencer	triathlon15	Szczecin
Aleksa Woźnicki	Mikroinfluencer	woznicki15	Cambridgeshire
Rafał Piotrowski	Mikroinfluencer	rafal15	Piotrków Trybunalski

☐ Show all | Number of rows: 25 | Filter rows: Search this table

## 2. Wyświetlanie działań wszystkich influencerów

Korzystając z wcześniej utworzonych funkcji, procedura ta wyświetla wszystkie usługi jakie wykonali wszyscy influencerzy i sortuje je rosnąco (domyślnie) według daty świadczenia usługi.

Procedura AllInfluencerActions() wyświetla dane za pomocą poleceń SQL umieszczonych w jej wnętrzu, nic nie pobiera ani nie zwraca.

### POLECENIE SQL:

```
DELIMITER //
CREATE PROCEDURE AllInfluencerActions()
BEGIN
    SELECT influencer.Id_influencer AS InfluencerID,
    FullName(influencer.first_name, influencer.last_name) AS FullName,
    service.name AS service,
    influenceraction.date AS Date
    FROM influencer, influenceraction, service
    WHERE influencer.Id_influencer = influenceraction.Influencer_Id_influencer
    AND influenceraction.Service_Id_service = service.Id_service
    ORDER BY Date;
END//
DELIMITER ;
```

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0058 seconds.)

```
CREATE PROCEDURE AllInfluencerActions() BEGIN SELECT influencer.Id_influencer AS InfluencerID,
FullName(influencer.first_name, influencer.last_name) AS FullName, service.name AS service,
influenceraction.date AS Date FROM influencer, influenceraction, service WHERE influencer.Id_influencer =
influenceraction.Influencer_Id_influencer AND influenceraction.Service_Id_service = service.Id_service
ORDER BY Date; END;
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

Poniżej dodatkowo podgląd wyeksportowanej funkcji.

Export of routine `AllInfluencerActions`

```
1 DELIMITER $$
2 CREATE DEFINER=`root`@`localhost` PROCEDURE `AllInfluencerActions`()
3 BEGIN
4     SELECT influencer.Id_influencer AS InfluencerID,
5         FullName(influencer.first_name, influencer.last_name) AS FullName,
6         service.name AS service,
7         influenceraction.date AS Date
8     FROM influencer, influenceraction, service
9     WHERE influencer.Id_influencer = influenceraction.Influencer_Id_influencer
10    AND influenceraction.Service_Id_service = service.Id_service
11    ORDER BY Date;
12 END$$
13 DELIMITER ;
```

Close

Utworzoną procedurę wywołałam poniższym poleceniem.

```
CALL AllInfluencerActions();
```

Zrzut wyniku działania procedury GetFullInfluencerData().

✓ Showing rows 0 - 11 (12 total, Query took 0.0013 seconds.)

CALL AllInfluencerActions();

[\[ Edit inline \]](#) [\[ Edit \]](#) [\[ Create PHP code \]](#)

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

InfluencerID	FullName	service	Date
8	Aleksa Woźnicki	Logo marki na stronie	2021-08-10
1	Ismenka Stelmaszczyk	Relacja	2021-09-15
1	Ismenka Stelmaszczyk	Użycie kodu influencera	2021-09-30
1	Ismenka Stelmaszczyk	Użycie kodu influencera	2021-09-30
2	Paulina Guzińska	Oznaczenie na story	2021-10-13
5	Maria Wrześniak	Post bez produktu	2021-10-19
7	Aleksander Mąkosa	Udział w evencie marki	2021-11-02
4	John Smith	Pakiet (post + relacja + swipe)	2021-11-02
10	Marcin Wiesiuk	Polecenie zakupu bez kodu	2021-11-03
3	Bartosz Smęda	Pakiet (post + relacja + swipe)	2021-11-14
1	Ismenka Stelmaszczyk	Relacja	2021-12-01
4	John Smith	Post z produktem	2021-12-04

☐ Show all | Number of rows: 25 | Filter rows: Search this table

### 3. Wyświetlanie działań konkretnego influencera

Udoskonalenie poprzedniej procedury o wyświetlanie świadczonych usług zawężonych do jednego konkretnego influencera. Procedura InfluencerActions() pobiera tym razem ID influencera.

## POLECENIE SQL:

```
DELIMITER //
CREATE PROCEDURE InfluencerActions(IN idInf INT)
BEGIN
    SELECT influencer.Id_influencer AS InfluencerID,
        FullName(influencer.first_name, influencer.last_name) AS FullName,
        service.name AS service,
        influenceraction.date AS Date
    FROM influencer, influenceraction, service
    WHERE influencer.Id_influencer = idInf
    AND influencer.Id_influencer = influenceraction.Influencer_Id_influencer
    AND influenceraction.Service_Id_service = service.Id_service
    ORDER BY Date;
END//
DELIMITER ;
```

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0058 seconds.)

```
CREATE PROCEDURE InfluencerActions(IN idInf INT) BEGIN SELECT influencer.Id_influencer AS InfluencerID,
FullName(influencer.first_name, influencer.last_name) AS FullName, service.name AS service, influenceraction.date AS
Date FROM influencer, influenceraction, service WHERE influencer.Id_influencer = idInf AND influencer.Id_influencer =
influenceraction.Influencer_Id_influencer AND influenceraction.Service_Id_service = service.Id_service ORDER BY Date;
END;
```

[\[ Edit inline \]](#) [\[ Edit \]](#) [\[ Create PHP code \]](#)

Poniżej dodatkowo podgląd wyeksportowanej funkcji.

Export of routine `AllInfluencerActions`

```
1 DELIMITER $$
2 CREATE DEFINER=`root`@`localhost` PROCEDURE `AllInfluencerActions`()
3 BEGIN
4     SELECT influencer.Id_influencer AS InfluencerID,
5         FullName(influencer.first_name, influencer.last_name) AS FullName,
6         service.name AS service,
7         influenceraction.date AS Date
8     FROM influencer, influenceraction, service
9     WHERE influencer.Id_influencer = influenceraction.Influencer_Id_influencer
10    AND influenceraction.Service_Id_service = service.Id_service
11    ORDER BY Date;
12 END$$
13 DELIMITER ;
```

Close

Wywołałam procedurę z argumentem 1 do wyświetlenia usług influencer'a o ID równym 1.

```
CALL InfluencerActions(1);
```

✓ Showing rows 0 - 3 (4 total, Query took 0.0043 seconds.)

`call InfluencerActions(1);`

[ Edit inline ] [ Edit ] [ Create PHP code ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

InfluencerID	FullName	service	Date
1	Ismenka Stelmaszczyk	Relacja	2021-09-15
1	Ismenka Stelmaszczyk	Użycie kodu influencera	2021-09-30
1	Ismenka Stelmaszczyk	Użycie kodu influencera	2021-09-30
1	Ismenka Stelmaszczyk	Relacja	2021-12-01

☐ Show all | Number of rows: 25 | Filter rows: Search this table

## 4. Wyświetlanie wynagrodzenia dla influencera w danym miesiącu

### POLECENIE SQL:

```
DELIMITER //
CREATE PROCEDURE InfluencerSalary(IN idInf INT)
BEGIN
    SELECT influencer.Id_influencer AS InfluencerID,
           FullName(influencer.first_name, influencer.last_name) AS FullName,
           service.name AS service,
           salary.value AS PLN,
           salary.date AS Date
    FROM influencer, influenceraction, service, salary
    WHERE influencer.Id_influencer = idInf
    AND influencer.Id_influencer = influenceraction.Influencer_Id_influencer
    AND influenceraction.Service_Id_service = service.Id_service
    AND influencer.Id_influencer = salary.Influencer_Id_influencer
    AND salary.InfluencerAction_Id_influencer_action =
    influenceraction.Id_influencer_action
    ORDER BY Date;
END//
DELIMITER ;
```

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0059 seconds.)

```
CREATE PROCEDURE InfluencerSalary(IN idInf INT) BEGIN SELECT influencer.Id_influencer AS InfluencerID,
FullName(influencer.first_name, influencer.last_name) AS FullName, service.name AS service, salary.value AS PLN,
salary.date AS Date FROM influencer, influenceraction, service, salary WHERE influencer.Id_influencer = idInf
AND influencer.Id_influencer = influenceraction.Influencer_Id_influencer AND influenceraction.Service_Id_service
= service.Id_service AND influencer.Id_influencer = salary.Influencer_Id_influencer AND
salary.InfluencerAction_Id_influencer_action = influenceraction.Id_influencer_action ORDER BY Date; END;
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

Poniżej dodatkowo podgląd wyeksportowanej funkcji.

```
Export of routine `InfluencerSalary`

1 DELIMITER $$
2 CREATE DEFINER=`root`@`localhost` PROCEDURE `InfluencerSalary`(IN `idInf` INT)
3 BEGIN
4     SELECT influencer.Id_influencer AS InfluencerID,
5     FullName(influencer.first_name, influencer.last_name) AS FullName,
6     influenceraction.Id_influencer_action AS ActionID,
7     service.name AS service,
8     salary.value AS PLN,
9     salary.date AS Date
10    FROM influencer, influenceraction, service, salary
11   WHERE influencer.Id_influencer = idInf
12     AND influencer.Id_influencer = influenceraction.Influencer_Id_influencer
13     AND influenceraction.Service_Id_service = service.Id_service
14     AND influencer.Id_influencer = salary.Influencer_Id_influencer
15     AND salary.InfluencerAction_Id_influencer_action = influenceraction.Id_influencer_action
16   ORDER BY Date;
17 END$$
18 DELIMITER ;
```

Close

Wywołałam procedurę z argumentem 1 do wyświetlenia usług influencera o ID równym 1.

```
call InfluencerSalary(4);
```

Zrzut wyniku działania procedury dla zarobków influencera o ID równym 4.

✓ Showing rows 0 - 3 (4 total, Query took 0.0007 seconds.)

```
call InfluencerSalary(4);
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

InfluencerID	FullName	ActionID	service	PLN	Date
4	John Smith	5	Pakiet (post + relacja + swipe)	350.00	2021-11-12
4	John Smith	13	Post z produktem	250.00	2021-12-04
4	John Smith	15	Relacja	50.00	2022-01-13
4	John Smith	16	Post z produktem	250.00	2022-01-13

☐ Show all | Number of rows: 25 | Filter rows: Search this table

## 5. Zmniejszenie ilość towarów w magazynie

Podaję jako parametry wejściowe ID produktu oraz jego ilość. Procedura aktualizuje ilość produktu o danym ID zmniejszając jego ilość w magazynie (tabela *product*) o taką, jaką podamy w drugim parametrze.

### POLECENIE SQL:

```
DELIMITER //
CREATE PROCEDURE ProductStock(IN productID INT, IN amount INT)
BEGIN
    UPDATE product
    SET product.Amount = product.Amount - amount
```

```
WHERE product.Id_product = productID;
END//
DELIMITER ;
```

Poniżej podgląd wyeksportowanej funkcji.

Export of routine `ProductStock`

```
1 DELIMITER $$
2 CREATE DEFINER=`root`@`localhost` PROCEDURE `ProductStock`(IN productID INT, IN amount INT)
3 BEGIN
4     UPDATE product
5     SET product.Amount = product.Amount - amount
6     WHERE product.Id_product = productID;
7 END$$
8 DELIMITER ;
```

Close

Dla przetestowania działania procedury wywołałam ją na jednym z produktów. Najpierw poniżej screen przed wywołaniem.

Showing rows 0 - 9 (10 total, Query took 0.0004 seconds.)

SELECT \* FROM `product`

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

+ Options

			Id_product	name	price	Amount
<input type="checkbox"/>	Edit	Copy	Delete	1	Modify Reductor	129.00 100
<input type="checkbox"/>	Edit	Copy	Delete	2	Modify Femibra	139.00 68
<input type="checkbox"/>	Edit	Copy	Delete	3	Gumka do włosów - Iniana	29.00 20
<input type="checkbox"/>	Edit	Copy	Delete	4	Gumka do włosów - Iniana XL	49.00 30
<input type="checkbox"/>	Edit	Copy	Delete	5	Gumka do włosów - welurowa	19.00 20
<input type="checkbox"/>	Edit	Copy	Delete	6	Gumka do włosów - welurowa XL	29.00 20
<input type="checkbox"/>	Edit	Copy	Delete	7	Pełna kuracja Modify Reductor	387.00 10
<input type="checkbox"/>	Edit	Copy	Delete	8	Pełna kuracja Modify Femibra	417.00 10
<input type="checkbox"/>	Edit	Copy	Delete	9	Zestaw świąteczny Modify Reductor	199.00 15
<input type="checkbox"/>	Edit	Copy	Delete	10	Zestaw świąteczny Modify Femibra	256.00 15

Wywołałam procedurę z argumentami: 1 jako ID produktu (Modify Reductor) oraz 1 jako ilością tego produktu do zmniejszenia.

```
CALL ProductStock(1, 1);
```

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0037 seconds.)

CALL ProductStock(1, 1);

[ Edit inline ] [ Edit ] [ Create PHP code ]

Zrzut po wywołaniu procedury zmniejszającej ilość produktu o 1. Zgodnie z oczekiwaniami ilość danego produktu zmniejszyła się o 1: ze 100 na 99.

Showing rows 0 - 9 (10 total, Query took 0.0002 seconds.)

```
SELECT * FROM `product` WHERE 1;
```

☐ Profiling [\[ Edit inline \]](#) [\[ Edit \]](#) [\[ Explain SQL \]](#) [\[ Create PHP code \]](#) [\[ Refresh \]](#)

☐ Show all | Number of rows:  | Filter rows:  | Sort by key:

+ Options

					Id_product	name	price	Amount
<input type="checkbox"/>					1	Modify Reductor	129.00	99
<input type="checkbox"/>					2	Modify Femibra	139.00	68
<input type="checkbox"/>					3	Gumka do włosów - Iniana	29.00	20
<input type="checkbox"/>					4	Gumka do włosów - Iniana XL	49.00	30
<input type="checkbox"/>					5	Gumka do włosów - welurowa	19.00	20
<input type="checkbox"/>					6	Gumka do włosów - welurowa XL	29.00	20
<input type="checkbox"/>					7	Pełna kuracja Modify Reductor	387.00	10
<input type="checkbox"/>					8	Pełna kuracja Modify Femibra	417.00	10
<input type="checkbox"/>					9	Zestaw świąteczny Modify Reductor	199.00	15
<input type="checkbox"/>					10	Zestaw świąteczny Modify Femibra	256.00	15

☐ Check all | With selected: Edit Copy Delete Export

☐ Show all | Number of rows:  | Filter rows:  | Sort by key:

Powyższą procedurę można poprawić dodając do niej obsługę wyjątku. Obecnie, w momencie, kiedy stan magazynu jest równy 0, procedura i tak zmniejszy ilość produktu do wartości ujemnej. Aby temu zapobiec, dodam do kodu fragment przechwytyjący taką sytuację - informujący o błędzie i nie zmniejszający ilości produktu.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `ProductStock`(IN `productID` INT, IN
`amount` INT)
BEGIN
    SET @MESSAGE_TEXT = 'The state of a stock is 0 for this product!';
    SET @GET_PRODUCT_AMOUNT = (SELECT product.Amount FROM product WHERE
product.Id_product = productID LIMIT 1);

    IF @GET_PRODUCT_AMOUNT - amount <= 0 THEN
    SELECT @MESSAGE_TEXT;
    ELSE
    UPDATE product
    SET product.Amount = product.Amount - amount
    WHERE product.Id_product = productID;
    END IF;
END$$
DELIMITER ;
```

```
Export of routine `ProductStock`

1 DELIMITER $$
2 CREATE DEFINER=`root`@`localhost` PROCEDURE `ProductStock`(IN `productID` INT, IN `amount` INT)
3 BEGIN
4     SET @MESSAGE_TEXT = 'The state of a stock is 0 for this product!';
5     SET @GET_PRODUCT_AMOUNT = (SELECT product.Amount FROM product WHERE product.Id_product = productID LIMIT 1);
6
7     IF @GET_PRODUCT_AMOUNT - amount <= 0 THEN
8         SELECT @MESSAGE_TEXT;
9     ELSE
10        UPDATE product
11        SET product.Amount = product.Amount - amount
12        WHERE product.Id_product = productID;
13    END IF;
14    END$$
15 DELIMITER ;
```

Zmieniłam ilość produktu o ID równym 10 na 1 aby pokazać działanie procedury.

Showing rows 0 - 9 (10 total, Query took 0.0003 seconds.)

SELECT \* FROM `product`

Profiling

[ Edit inline ]

[ Edit ]

[ Explain SQL ]

[ Create PHP code ]

[ Refresh ]

Show all

Number of rows: 25

Filter rows: Search this table

Sort by key: None

+ Options

						Id_product	name	price	Amount
<input type="checkbox"/>				1		Modify Reductor		129.00	94
<input type="checkbox"/>				2		Modify Femibra		139.00	67
<input type="checkbox"/>				3		Gumka do włosów - Iniana		29.00	20
<input type="checkbox"/>				4		Gumka do włosów - Iniana XL		49.00	30
<input type="checkbox"/>				5		Gumka do włosów - welurowa		19.00	20
<input type="checkbox"/>				6		Gumka do włosów - welurowa XL		29.00	20
<input type="checkbox"/>				7		Pełna kuracja Modify Reductor		387.00	10
<input type="checkbox"/>				8		Pełna kuracja Modify Femibra		417.00	10
<input type="checkbox"/>				9		Zestaw Świąteczny Modify Reductor		199.00	15
<input type="checkbox"/>				10		Zestaw Świąteczny Modify Femibra		256.00	1

Za pierwszym razem, kiedy wywołuję procedurę na produkcie o ID=10 oraz z ilością 1, ze stanu magazynu zostaje zdjęta oczekiwana ilość produktu i zostaje na stanie 0 sztuk. Screen poniżej.

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0017 seconds.)

CALL ProductStock(10,1);

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ Showing rows 0 - 9 (10 total, Query took 0.0003 seconds.)

SELECT \* FROM `product`

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

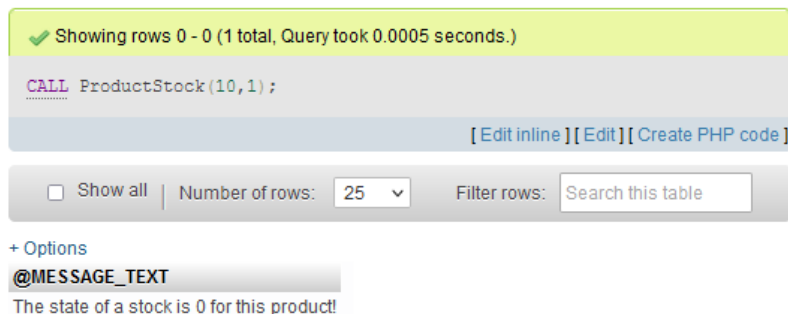
☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

+ Options

					Id_product	name	price	Amount
<input type="checkbox"/>				Delete	1	Modify Reductor	129.00	94
<input type="checkbox"/>				Delete	2	Modify Femibra	139.00	67
<input type="checkbox"/>				Delete	3	Gumka do włosów - Iniana	29.00	20
<input type="checkbox"/>				Delete	4	Gumka do włosów - Iniana XL	49.00	30
<input type="checkbox"/>				Delete	5	Gumka do włosów - welurowa	19.00	20
<input type="checkbox"/>				Delete	6	Gumka do włosów - welurowa XL	29.00	20
<input type="checkbox"/>				Delete	7	Pełna kuracja Modify Reductor	387.00	10
<input type="checkbox"/>				Delete	8	Pełna kuracja Modify Femibra	417.00	10
<input type="checkbox"/>				Delete	9	Zestaw świąteczny Modify Reductor	199.00	15
<input type="checkbox"/>				Delete	10	Zestaw świąteczny Modify Femibra	256.00	0



Kiedy na tym samym produkcie wywołałam tę samą procedurę ponownie, tym razem stan magazynu jest równy zero dla produktu o ID równym 10 a więc otrzymam odpowiedni komunikat z procedury. Jak widać w kodzie procedury, jeśli spróbowałabym zdjąć ze stanu dowolną ilość większą niż jest, również otrzymam komunikat błędu.



Stan magazynu dla tego produktu nie uległ zmianie. Obsługa wyjątków działa prawidłowo.

## Wyzwalacze

Zdarzenie, inaczej trigger, wyzwalacz, to skrypt (fragment kodu) wykonywany automatycznie w przypadku zajścia jakiegoś zdarzenia w bazie danych (np. dodania danych, ich modyfikacji, czy usunięcia).

Istnieje kilka typów wyzwalaczy. najpowszechniejsze to BEFORE i AFTER. Dla każdego typu istnieją trzy zdarzenia powodujące wykonanie wyzwalacza i są to:

- AFTER DELETE – wykonanie wyzwalacza po operacji usunięcia rekordu
- AFTER INSERT – wykonanie wyzwalacza po dodaniu rekordu
- AFTER UPDATE – wykonanie wyzwalacza po zmodyfikowaniu rekordu
- BEFORE DELETE – wykonanie wyzwalacza przed operacją usunięcia rekordu
- BEFORE INSERT – wykonanie wyzwalacza przed dodaniem rekordu
- BEFORE UPDATE – wykonanie wyzwalacza przed zmodyfikowaniem rekordu

### 1. Wywołanie procedury zmniejszania ilości magazynu

Za każdym razem, kiedy stworzona zostanie nowa wysyłka (rekord w tabeli *shipping*), wyzwalacz wywoła procedurę zmniejszającą w magazynie ilość wysyłanego produktu.

Skorzystałam w wcześniej stworzonej procedury ProductStock, która przyjmuje w parametrach ID produktu oraz jego ilość. Procedura ta aktualizuje ilość produktu o danym ID zmniejszając jego ilość w magazynie (tabela *product*) o taką, jaką podamy w drugim parametrze.

Następnie utworzyłam wyzwalacz do uruchomienia tej procedury po każdorazowym dodaniu rekordu do tabeli *shipping*.

#### POLECENIE SQL:

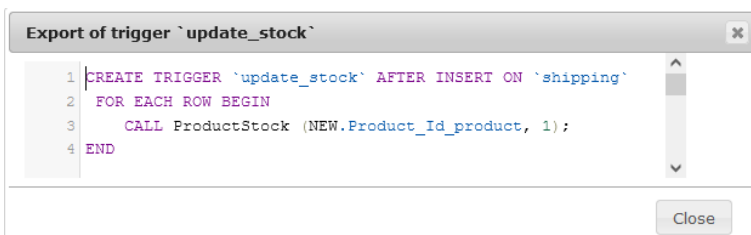
```
DELIMITER //
CREATE TRIGGER update_stock
AFTER INSERT ON shipping
FOR EACH ROW
BEGIN
    CALL ProductStock (NEW.Product_Id_product, 1);
END
```

```
END//  
DELIMITER ;
```

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0073 seconds.)

```
CREATE TRIGGER update_stock AFTER INSERT ON shipping  
FOR EACH ROW BEGIN CALL ProductStock  
(NEW.Product_Id_product, 1); END;
```

[\[ Edit inline \]](#) [\[ Edit \]](#) [\[ Create PHP code \]](#)



Po pomyślnym utworzeniu triggera wywołującego procedurę, sprawdziłam działanie. Utworzyłam wysyłkę do influencera z konkretnym produktem, czyli dodaję rekord w tabeli *shipping*. Po dodaniu rekordu powinien wywołać się trigger uruchamiający procedurę zmniejszania ilości magazynu (tabeli *product*) o wysłany produkt. Screeny z tego działania w krokach poniżej.

Widok tabeli *shipping* i *product* PRZED operacją.

✓ Showing rows 0 - 11 (12 total, Query took 0.0004 seconds.) [Product\_Id\_product: 10... - 1...]

```
SELECT * FROM `shipping` ORDER BY `Product_Id_product` DESC
```

☐ Profiling [\[ Edit inline \]](#) [\[ Edit \]](#) [\[ Explain SQL \]](#) [\[ Create PHP code \]](#) [\[ Refresh \]](#)

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: Product\_Id\_product (DESC)

+ Options

	Id_shipping	Influencer_Id_influencer	Product_Id_product	date
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	7	6	10	2020-12-12
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	11	10	9	2021-11-15
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	3	9	2021-08-18
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	8	7	9	2021-11-17
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	7	2021-09-08
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	2	7	2021-11-07
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	6	5	5	2021-10-18
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	12	1	4	2021-11-08
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	5	5	2	2021-10-18
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	9	8	2	2021-07-11
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	4	1	2021-11-19
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	10	9	1	2021-11-06

↑ ☐ Check all | With selected: ☐ Edit ☐ Copy ☐ Delete ☐ Export

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: Product\_Id\_product (DESC)

Showing rows 0 - 9 (10 total, Query took 0.0003 seconds.)

```
SELECT * FROM `product`
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

+ Options

				Id_product	name	price	Amount
<input type="checkbox"/>	Edit	Copy	Delete	1	Modify Reductor	129.00	99
<input type="checkbox"/>	Edit	Copy	Delete	2	Modify Femibra	139.00	68
<input type="checkbox"/>	Edit	Copy	Delete	3	Gumka do włosów - Iniana	29.00	20
<input type="checkbox"/>	Edit	Copy	Delete	4	Gumka do włosów - Iniana XL	49.00	30
<input type="checkbox"/>	Edit	Copy	Delete	5	Gumka do włosów - welurowa	19.00	20
<input type="checkbox"/>	Edit	Copy	Delete	6	Gumka do włosów - welurowa XL	29.00	20
<input type="checkbox"/>	Edit	Copy	Delete	7	Pełna kuracja Modify Reductor	387.00	10
<input type="checkbox"/>	Edit	Copy	Delete	8	Pełna kuracja Modify Femibra	417.00	10
<input type="checkbox"/>	Edit	Copy	Delete	9	Zestaw świąteczny Modify Reductor	199.00	15
<input type="checkbox"/>	Edit	Copy	Delete	10	Zestaw świąteczny Modify Femibra	256.00	15

Check all | With selected: Edit Copy Delete Export

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Dodałam wysyłkę, a więc rekord do tabeli *shipping*.

1 row inserted.  
Inserted row id: 13

```
INSERT INTO `shipping` (`Id_shipping`, `Influencer_Id_influencer`, `Product_Id_product`, `date`)
VALUES (NULL, '6', '2', '2022-01-20');
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

Widok tabeli *product* PO operacji. Ilość wysyłanego produktu zmniejszyła się w magazynie zgodnie z oczekiwaniami.

Showing rows 0 - 9 (10 total, Query took 0.0003 seconds.)

```
SELECT * FROM `product`
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

+ Options

				Id_product	name	price	Amount
<input type="checkbox"/>	Edit	Copy	Delete	1	Modify Reductor	129.00	99
<input type="checkbox"/>	Edit	Copy	Delete	2	Modify Femibra	139.00	67
<input type="checkbox"/>	Edit	Copy	Delete	3	Gumka do włosów - Iniana	29.00	20
<input type="checkbox"/>	Edit	Copy	Delete	4	Gumka do włosów - Iniana XL	49.00	30
<input type="checkbox"/>	Edit	Copy	Delete	5	Gumka do włosów - welurowa	19.00	20
<input type="checkbox"/>	Edit	Copy	Delete	6	Gumka do włosów - welurowa XL	29.00	20
<input type="checkbox"/>	Edit	Copy	Delete	7	Pełna kuracja Modify Reductor	387.00	10
<input type="checkbox"/>	Edit	Copy	Delete	8	Pełna kuracja Modify Femibra	417.00	10
<input type="checkbox"/>	Edit	Copy	Delete	9	Zestaw świąteczny Modify Reductor	199.00	15
<input type="checkbox"/>	Edit	Copy	Delete	10	Zestaw świąteczny Modify Femibra	256.00	15

Check all | With selected: Edit Copy Delete Export

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

## 2. Dodanie nowego rekordu

Dodaję trigger tworzenia nowego rekordu w tabeli *salary*. Będzie się on wywoływać za każdym razem, gdy dodam akcję do influencera, czyli nowy rekord w tabeli *influenceraction*.

Automatycznie do tabeli salary

## POLECENIE SQL:

```
CREATE TRIGGER `new_salary`  
AFTER INSERT ON `influenceraction`  
FOR EACH ROW  
INSERT INTO `salary` (`Id_salary`, `Influencer_Id_influencer`,  
`InfluencerAction_Id_influencer_action`, `value`, `date`)  
VALUES (NULL, NEW.Influencer_Id_influencer, NEW.Id_influencer_action, (SELECT  
value FROM `service` WHERE Id_service = NEW.Service_Id_service), DATE(NOW()))
```

✓ Trigger `new\_salary` has been created.

```
CREATE TRIGGER `new_salary` AFTER INSERT ON `influenceraction` FOR EACH ROW INSERT INTO `salary`  
(`Id_salary`, `Influencer_Id_influencer`, `InfluencerAction_Id_influencer_action`, `value`,  
`date`) VALUES (NULL, NEW.Influencer_Id_influencer, NEW.Id_influencer_action, (SELECT value FROM  
`service` WHERE Id_service = NEW.Service_Id_service), DATE(NOW()))
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

### Triggers

Name	Action	Time	Event
<input type="checkbox"/> new_salary	Edit  Export  Drop	AFTER	INSERT

Dla sprawdzenia działania dodałam rekord w tabeli “influenceraction”. Screen tabel *influenceraction* i *salary* przed operacją:

		Id_influencer_action	Influencer_Id_influencer	Service_Id_service	date
<input type="checkbox"/>	Edit  Copy  Delete	1	1	8	2021-09-30
<input type="checkbox"/>	Edit  Copy  Delete	2	1	8	2021-09-30
<input type="checkbox"/>	Edit  Copy  Delete	3	1	1	2021-09-15
<input type="checkbox"/>	Edit  Copy  Delete	14	1	9	2022-01-13
<input type="checkbox"/>	Edit  Copy  Delete	11	1	1	2021-12-01
<input type="checkbox"/>	Edit  Copy  Delete	4	2	7	2021-10-13
<input type="checkbox"/>	Edit  Copy  Delete	6	3	10	2021-11-14
<input type="checkbox"/>	Edit  Copy  Delete	15	4	1	2022-01-01
<input type="checkbox"/>	Edit  Copy  Delete	13	4	4	2021-12-04
<input type="checkbox"/>	Edit  Copy  Delete	16	4	4	2022-01-04
<input type="checkbox"/>	Edit  Copy  Delete	5	4	10	2021-11-02
<input type="checkbox"/>	Edit  Copy  Delete	7	5	3	2021-10-19
<input type="checkbox"/>	Edit  Copy  Delete	8	7	5	2021-11-02
<input type="checkbox"/>	Edit  Copy  Delete	10	8	6	2021-08-10
<input type="checkbox"/>	Edit  Copy  Delete	9	10	9	2021-11-03

↑ ☐ Check all With selected: Edit Copy Delete Export

				Id_salary	Influencer_Id_influencer	InfluencerAction_Id_influencer_action	value	date
<input type="checkbox"/>	Edit	Copy	Delete	1	1	1	30.00	2021-09-30
<input type="checkbox"/>	Edit	Copy	Delete	2	1	2	30.00	2021-09-30
<input type="checkbox"/>	Edit	Copy	Delete	3	1	3	50.00	2021-09-15
<input type="checkbox"/>	Edit	Copy	Delete	4	2	4	100.00	2021-10-13
<input type="checkbox"/>	Edit	Copy	Delete	5	4	5	350.00	2021-11-12
<input type="checkbox"/>	Edit	Copy	Delete	6	3	6	350.00	2021-11-14
<input type="checkbox"/>	Edit	Copy	Delete	7	5	7	100.00	2021-10-19
<input type="checkbox"/>	Edit	Copy	Delete	8	7	8	500.00	2021-11-02
<input type="checkbox"/>	Edit	Copy	Delete	9	10	9	40.00	2021-11-03
<input type="checkbox"/>	Edit	Copy	Delete	10	8	10	200.00	2021-08-10
<input type="checkbox"/>	Edit	Copy	Delete	13	4	13	250.00	2021-12-04
<input type="checkbox"/>	Edit	Copy	Delete	14	1	14	40.00	2022-01-13
<input type="checkbox"/>	Edit	Copy	Delete	15	4	15	50.00	2022-01-13
<input type="checkbox"/>	Edit	Copy	Delete	16	4	16	250.00	2022-01-13

☐ Check all    With selected:    Edit    Copy    Delete    Export

oraz po dodaniu rekordu w tabeli *influenceraction*:

✓ 1 row inserted.

Inserted row id: 17

```
INSERT INTO `influenceraction` (`Id_influencer_action`, `Influencer_Id_influencer`, `Service_Id_service`, `date`)
VALUES (NULL, '2', '8', NULL);
```

[\[ Edit inline \]](#)
[\[ Edit \]](#)
[\[ Create PHP code \]](#)

Run SQL query/queries on table db\_influencers\_konw.influenceraction:

```
1 INSERT INTO `influenceraction` (`Id_influencer_action`,
`Influencer_Id_influencer`, `Service_Id_service`, `date`) VALUES (NULL,
'2', '8', NULL);
```

Id\_influencer\_action

Influencer\_Id\_influencer

Service\_Id\_service

date

widać, że w tabeli *influenceraction* dodano 1 rekord (o ID=17)

				Id_influencer_action	Influencer_Id_influencer	Service_Id_service	date
<input type="checkbox"/>	Edit	Copy	Delete	1	1	8	2021-09-30
<input type="checkbox"/>	Edit	Copy	Delete	2	1	8	2021-09-30
<input type="checkbox"/>	Edit	Copy	Delete	3	1	1	2021-09-15
<input type="checkbox"/>	Edit	Copy	Delete	4	2	7	2021-10-13
<input type="checkbox"/>	Edit	Copy	Delete	5	4	10	2021-11-02
<input type="checkbox"/>	Edit	Copy	Delete	6	3	10	2021-11-14
<input type="checkbox"/>	Edit	Copy	Delete	7	5	3	2021-10-19
<input type="checkbox"/>	Edit	Copy	Delete	8	7	5	2021-11-02
<input type="checkbox"/>	Edit	Copy	Delete	9	10	9	2021-11-03
<input type="checkbox"/>	Edit	Copy	Delete	10	8	6	2021-08-10
<input type="checkbox"/>	Edit	Copy	Delete	11	1	1	2021-12-01
<input type="checkbox"/>	Edit	Copy	Delete	13	4	4	2021-12-04
<input type="checkbox"/>	Edit	Copy	Delete	14	1	9	2022-01-13
<input type="checkbox"/>	Edit	Copy	Delete	15	4	1	2022-01-01
<input type="checkbox"/>	Edit	Copy	Delete	16	4	4	2022-01-04
<input type="checkbox"/>	Edit	Copy	Delete	17	2	8	NULL

☐ Check all    With selected:    Edit    Copy    Delete    Export

a tym samym automatycznie w tabeli *salary* stworzony został rekord o wynagrodzeniu dla influencera za dodaną wyżej akcję

					Id_salary	Influencer_Id_influencer	InfluencerAction_Id_influencer_action	value	date
<input type="checkbox"/>	Edit	Copy	Delete	1	1		1	30.00	2021-09-30
<input type="checkbox"/>	Edit	Copy	Delete	2	1		2	30.00	2021-09-30
<input type="checkbox"/>	Edit	Copy	Delete	3	1		3	50.00	2021-09-15
<input type="checkbox"/>	Edit	Copy	Delete	4	2		4	100.00	2021-10-13
<input type="checkbox"/>	Edit	Copy	Delete	5	4		5	350.00	2021-11-12
<input type="checkbox"/>	Edit	Copy	Delete	6	3		6	350.00	2021-11-14
<input type="checkbox"/>	Edit	Copy	Delete	7	5		7	100.00	2021-10-19
<input type="checkbox"/>	Edit	Copy	Delete	8	7		8	500.00	2021-11-02
<input type="checkbox"/>	Edit	Copy	Delete	9	10		9	40.00	2021-11-03
<input type="checkbox"/>	Edit	Copy	Delete	10	8		10	200.00	2021-08-10
<input type="checkbox"/>	Edit	Copy	Delete	13	4		13	250.00	2021-12-04
<input type="checkbox"/>	Edit	Copy	Delete	14	1		14	40.00	2022-01-13
<input type="checkbox"/>	Edit	Copy	Delete	15	4		15	50.00	2022-01-13
<input type="checkbox"/>	Edit	Copy	Delete	16	4		16	250.00	2022-01-13
<input type="checkbox"/>	Edit	Copy	Delete	17	2		17	30.00	2022-01-20

☐ Check all    With selected:    Edit    Copy    Delete    Export

Wyzwalacz wygenerował w tabeli "Salary" rekord z automatycznie zaciągniętą wartością usługi z tabeli "service" co świadczy o poprawności działania skryptu zdarzenia.

## Przydzielanie uprawnień użytkownikom

System MySQL jak każdy inny system do zarządzania bazami danych pomaga użytkownikom przechowywać, uporządkowywać i pobierać dane. Ze względów bezpieczeństwa zalecane jest tworzenie różnego typu użytkowników MySQL o odrębnych uprawnieniach w celu ograniczenia dostępu i uniknięcia niepożądanych zmian w bazach danych.

### Sprawdzenie bieżących uprawnień

Przed tworzeniem nowych użytkowników i nadawaniu im uprawnień upewniłam się, że sama mam do tego uprawnienia będąc na obecnym koncie.

#### POLECENIE SQL:

```
SHOW GRANTS FOR CURRENT_USER;
```

Your SQL query has been executed successfully.

```
show grants for CURRENT_USER;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Create PHP code ] [ Refresh ]

+ Options

Grants for root@localhost

```
GRANT ALL PRIVILEGES ON *.* TO 'root'@'localhost' ...
```

```
GRANT PROXY ON '@'%' TO 'root'@'localhost' WITH G...
```

## Tworzenie nowego użytkownika z uprawnieniami READONLY

Utworzyłam nowego użytkownika przez polecenia w MySQL:

## POLECENIE SQL:

```
CREATE USER 'UserAdmin'@'%' IDENTIFIED BY 'AdminPassword';
```

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0037 seconds.)

```
CREATE USER 'UserAdmin'@'%' IDENTIFIED BY 'AdminPassword';
```

[ [Edit inline](#) ] [ [Edit](#) ] [ [Create PHP code](#) ]

## Nadawanie uprawnień

W tym miejscu nowy użytkownik o nazwie “UserAdmin” nie jest jeszcze upoważniony do wniesienia jakichkolwiek zmian w bazę danych. W rzeczywistości, jeśli nowy użytkownik spróbowałby się na tym etapie zalogować (z hasłem 'AdminPassword'), nie będzie on mógł dostać się do linii poleceń MySQL. Dlatego, najpierw muszę mu zapewnić dostęp do informacji, do której będzie mógł się dostać. Nadaję mu 2 uprawnienia typu readonly.

## POLECENIE SQL:

```
GRANT SELECT, SHOW VIEW ON `db_influencers_konw`.* TO `UserAdmin`@'%' ;
```

- Uprawnienia SELECT, SHOW VIEW umożliwią użytkownikom MySQL dostęp do tych właśnie poleceń (i żadnych innych),
- W powyższym wyrażeniu ‘%’ oznacza, że użytkownik będzie miał dostęp ze wszystkich adresów IP,
- \* oznacza wszystkie.

Poniżej widoczne dodatkowo nadane uprawnienia tylko do odczytu.

Database Login Information

Edit privileges: User account 'UserAdmin'@'%' - Databases db\_influencers\_konw

Database-specific privileges [Check all](#)

Note: MySQL privilege names are expressed in English.

Data	Structure	Administration
<input checked="" type="checkbox"/> SELECT	<input type="checkbox"/> CREATE	<input type="checkbox"/> GRANT
<input type="checkbox"/> INSERT	<input type="checkbox"/> ALTER	<input type="checkbox"/> LOCK TABLES
<input type="checkbox"/> UPDATE	<input type="checkbox"/> INDEX	<input type="checkbox"/> REFERENCES
<input type="checkbox"/> DELETE	<input type="checkbox"/> DROP	
	<input type="checkbox"/> CREATE TEMPORARY TABLES	
	<input checked="" type="checkbox"/> SHOW VIEW	
	<input type="checkbox"/> CREATE ROUTINE	
	<input type="checkbox"/> ALTER ROUTINE	
	<input type="checkbox"/> EXECUTE	
	<input type="checkbox"/> CREATE VIEW	
	<input type="checkbox"/> EVENT	
	<input type="checkbox"/> TRIGGER	

Upewniam się, jakie przywileje zostały nadane.

## POLECENIE SQL:

```
SHOW GRANTS FOR UserAdmin;
```

Your SQL query has been executed successfully.

```
SHOW GRANTS FOR UserAdmin;  
.....
```

☐ Profiling [ [Edit inline](#) ] [ [Edit](#) ] [ [Create PHP code](#) ] [ [Refresh](#) ]

☐ Profiling [ [Edit inline](#) ] [ [Edit](#) ] [ [Create PHP code](#) ] [ [Refresh](#) ]

☐ Profiling [\[ Edit inline \]](#) [\[ Edit \]](#) [\[ Create PHP code \]](#) [\[ Refresh \]](#)

+ Options

**Grants for UserAdmin@%**

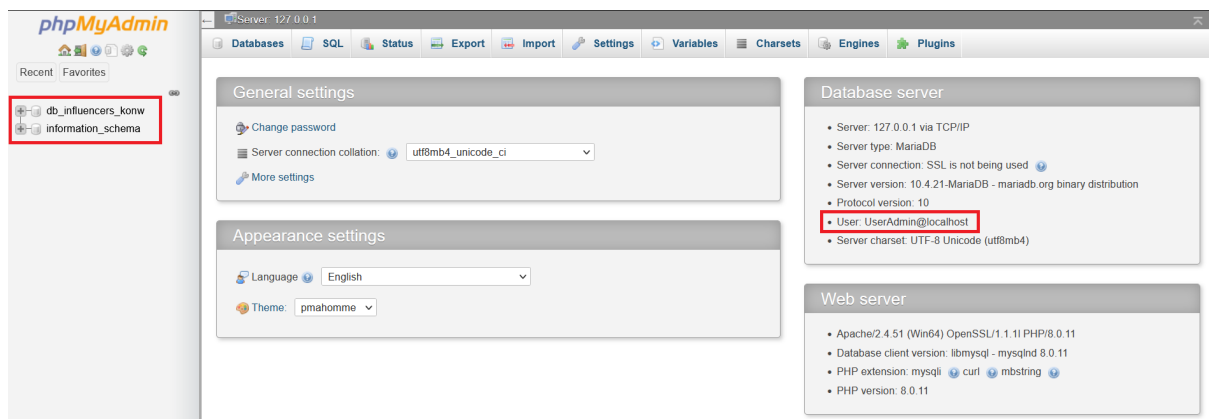
```
GRANT USAGE ON *.* TO 'UserAdmin'@`%` IDENTIFIED BY B...
GRANT SELECT, SHOW VIEW ON `db_influencers_konw`.*...
```

```
GRANTS for UserAdmin@%
GRANT USAGE ON *.* TO `UserAdmin`@`%` IDENTIFIED B...
GRANT SELECT, SHOW VIEW ON `db_influencers_konw`.*
```

```
GRANT USAGE ON *.* TO 'UserAdmin'@'%' IDENTIFIED BY 'UserAdmin';
GRANT SELECT, SHOW VIEW ON `db_influencers_konw`.* TO 'UserAdmin'@'%' IDENTIFIED BY 'UserAdmin';
```

```
GRANT SELECT, SHOW VIEW ON `db_influencers_konw`.*...
```

Loguję się jako nowo utworzony użytkownik. Widać, że jedyna baza, do której jest dostęp to ta, do której udzieliłam dostępu czyli "db\_influencers\_konw".



Uprawnień udzieliłam jedynie dla polecenia SELECT oraz SHOW VIEW. Jak widać na poniższym screenie, polecenie SELECT działa bez zarzutu.

Showing rows 0 - 9 (10 total, Query took 0.0005 seconds.)

SELECT \* FROM influencer;

Profiling

[Edit inline]

[Edit]

[Explain SQL]

[Create PHP code]

[Refresh]

Show all

Number of rows: 25

Filter rows: Search this table

Sort by key: None

+ Options

				Id_influencer	Address_Id_address	first_name	last_name	nickname	discount_code	followers	account_number
<input type="checkbox"/>				1	4	Ismenka	Stelmaszczyk	ismena_stelmaszczyk	ismena15	47600	NULL
<input type="checkbox"/>				2	2	Paulina	Guzińska	paulina_guzinska	paulina15	264000	NULL
<input type="checkbox"/>				3	5	Bartosz	Smęda	smeda.triathlon	triathlon15	801	NULL
<input type="checkbox"/>				4	3	John	Smith	john_smith	john15	45000	12445588237595145212547719
<input type="checkbox"/>				5	1	Maria	Wrześniak	mariaaa	marysia15	53000	NULL
<input type="checkbox"/>				6	6	Klaudia	Michalak	klaudia_michalak	klaudia15	4123	NULL
<input type="checkbox"/>				7	7	Aleksander	Makosa	alexi	alexi15	89000	NULL
<input type="checkbox"/>				8	8	Aleksa	Woźnicki	aleksa_woznikci	woznikci15	632	NULL
<input type="checkbox"/>				9	9	Rafał	Piotrowski	rafal_p	rafal15	7522	NULL
<input type="checkbox"/>				10	10	Marcin	Wiesiuk	marcinek_w	marcinek15	65002	NULL

↑

☐ Check all

With selected:

Edit

Copy

Delete

Export

Bez zarzutu działa również “czytanie” definicji utworzonych widoków. To druga z opcji typu readonly, która działa według oczekiwań, ponieważ nadany został taki przywilej dla tego konta.



## POLECENIE SQL:

```
SHOW CREATE VIEW full_address_data;
```

View	Create View	character_set_client	collation_connection
full_address_data	CREATE ALGORITHM=UNDEFINED DEFINER='root'@'localho...	utf8mb4	utf8mb4_unicode_ci

Tym razem spróbuję dodać jakiś rekord do dowolnej tabeli. Polecenie INSERT nie było nadane dla tego konta w przywilejach. Jak widać poniżej, jest odmowa wykonania go dla tego użytkownika. Wszystko działa zgodnie z założeniami.

**Error**

SQL query: [Copy](#) [Edit](#)

```
INSERT INTO `influencer` (`Id_influencer`, `Addr`
```

MySQL said:

```
#1142 - INSERT command denied to user
'UserAdmin'@'localhost' for table
'influencer'
```

Tworzenie nowego użytkownika z uprawnieniami 'full access'

Utworzyłam nowego użytkownika przez polecenia w MySQL:

## POLECENIE SQL:

```
CREATE USER 'UserAdmin2'@'%' IDENTIFIED BY 'Admin2Password';
```

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0021 seconds.)

```
CREATE USER 'UserAdmin2'@'%' IDENTIFIED BY 'Admin2Password';
```

[\[ Edit inline \]](#) [\[ Edit \]](#) [\[ Create PHP code \]](#)

Nadawanie uprawnień

Tak jak poprzednio, zapewniam użytkownikowi najpierw dostęp do informacji, do której będzie mógł się dostać. Tym razem nadaję mu pełny dostęp do bazy danych 'db\_influencers\_konw' - poza opcjami administrującymi.

### POLECENIE SQL:

```
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX, ALTER, CREATE  
TEMPORARY TABLES, CREATE VIEW, EVENT, TRIGGER, SHOW VIEW, CREATE ROUTINE, ALTER  
ROUTINE, EXECUTE ON `db_influencers_konw`.* TO 'UserAdmin2'@'%';
```

Poniżej widoczne dodatkowo nadane wszystkie uprawnienia - poza administracyjnymi.

Database Table Routine Login Information

Edit privileges: User account 'UserAdmin2'@'%' - Database db\_influencers\_konw

Database-specific privileges ☒ Check all

Note: MySQL privilege names are expressed in English.

☒ Data

- ☒ SELECT
- ☒ INSERT
- ☒ UPDATE
- ☒ DELETE

☒ Structure

- ☒ CREATE
- ☒ ALTER
- ☒ INDEX
- ☒ DROP
- ☒ CREATE TEMPORARY TABLES
- ☒ SHOW VIEW
- ☒ CREATE ROUTINE
- ☒ ALTER ROUTINE
- ☒ EXECUTE
- ☒ CREATE VIEW
- ☒ EVENT
- ☒ TRIGGER

☐ Administration

- ☐ GRANT
- ☐ LOCK TABLES
- ☐ REFERENCES

Go

Upewniam się, jakie przywileje zostały nadane.

### POLECENIE SQL:

```
SHOW GRANTS FOR UserAdmin;
```

Your SQL query has been executed successfully.

```
SHOW GRANTS FOR UserAdmin2;
```

☐ Profiling [ Edit inline ] [ Edit ] [ Create PHP code ] [ Refresh ]

+ Options

**Grants for UserAdmin2@%**

```
GRANT USAGE ON *.* TO 'UserAdmin2'@'%' IDENTIFIED ...  
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP...
```

Loguję się jako nowo utworzony użytkownik UserAdmin2.



Welcome to phpMyAdmin

Language

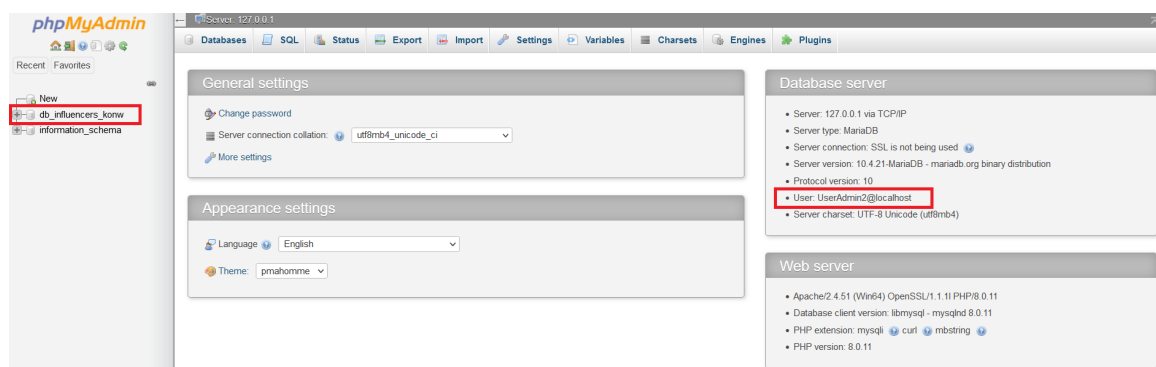
English

Log in

Username: UserAdmin2
Password:

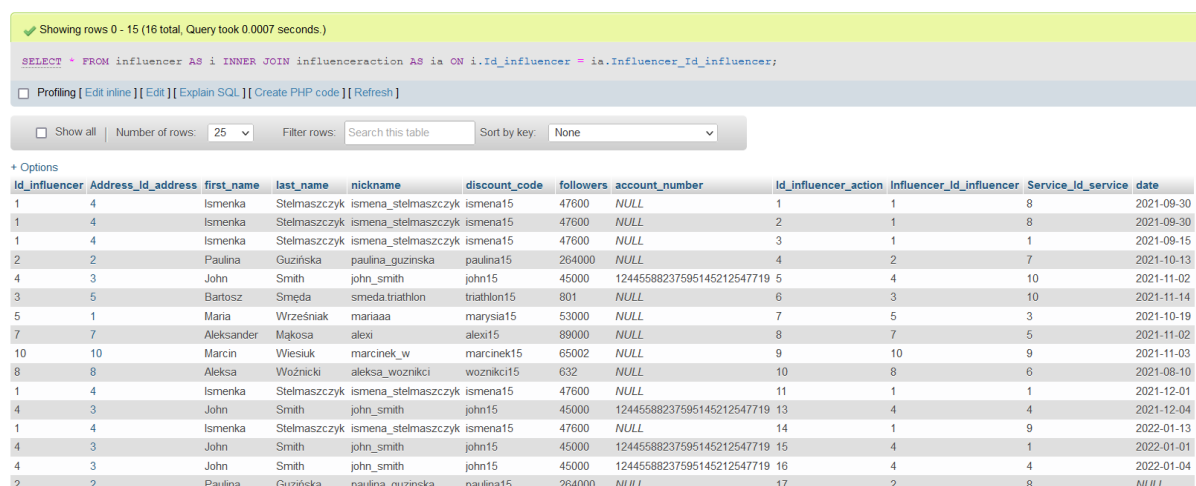
Go

Widać, że jedyna baza, do której jest dostęp to ta, do której udzieliłam dostępu czyli “db\_influencers\_konw”.



The screenshot shows the phpMyAdmin interface. On the left, the 'New' dropdown menu is open, and 'db\_influencers\_konw' is highlighted. The main panel shows the 'General settings' and 'Database server' sections. In the 'Database server' section, the 'User' is set to 'UserAdmin2@localhost'.

Uprawnień udzieliłam na wszystkie polecenia zapisu, odczytu i wszelkich modyfikacji danych (poza administracyjnymi). Jak widać na poniższym screenie, polecenie SELECT działa bez zarzutu.



The screenshot shows the SQL query results in phpMyAdmin. The query is a JOIN operation between the 'influencer' and 'influenceraction' tables. The results are displayed in a table with 16 rows and 12 columns. The columns are: Id\_influencer, Address\_Id\_address, first\_name, last\_name, nickname, discount\_code, followers, account\_number, Id\_influencer\_action, Influencer\_Id\_influencer, Service\_Id\_service, and date.

Id_influencer	Address_Id_address	first_name	last_name	nickname	discount_code	followers	account_number	Id_influencer_action	Influencer_Id_influencer	Service_Id_service	date
1	4	Ismenka	Stelmaszczyk	ismena_stelmaszczyk	ismena15	47600	NULL	1	1	8	2021-09-30
1	4	Ismenka	Stelmaszczyk	ismena_stelmaszczyk	ismena15	47600	NULL	2	1	8	2021-09-30
1	4	Ismenka	Stelmaszczyk	ismena_stelmaszczyk	ismena15	47600	NULL	3	1	1	2021-09-15
2	2	Paulina	Guzińska	paulina_guzinska	paulina15	264000	NULL	4	2	7	2021-10-13
4	3	John	Smith	john_smith	john15	45000	12445588237595145212547719	5	4	10	2021-11-02
3	5	Bartosz	Smeda	smeda.triathlon	triathlon15	801	NULL	6	3	10	2021-11-14
5	1	Maria	Wrześniak	mariaaa	marysia15	53000	NULL	7	5	3	2021-10-19
7	7	Aleksander	Makosa	alexi	alexi15	89000	NULL	8	7	5	2021-11-02
10	10	Marcin	Wiesiuk	marcinek_w	marcinek15	65002	NULL	9	10	9	2021-11-03
8	8	Aleksa	Woźnicki	aleksa_woznicki	woznicki15	632	NULL	10	8	6	2021-08-10
1	4	Ismenka	Stelmaszczyk	ismena_stelmaszczyk	ismena15	47600	NULL	11	1	1	2021-12-01
4	3	John	Smith	john_smith	john15	45000	12445588237595145212547719	13	4	4	2021-12-04
1	4	Ismenka	Stelmaszczyk	ismena_stelmaszczyk	ismena15	47600	NULL	14	1	9	2022-01-13
4	3	John	Smith	john_smith	john15	45000	12445588237595145212547719	15	4	1	2022-01-01
4	3	John	Smith	john_smith	john15	45000	12445588237595145212547719	16	4	4	2022-01-04
2	2	Paulina	Guzińska	paulina_guzinska	paulina15	264000	NULL	17	2	8	NULL

Bez zarzutu działa również polecenie INSERT, aby dodać jakiś rekord do dowolnej tabeli.

## POLECENIE SQL:


































```
INSERT INTO product (Id_product, name, price, Amount) VALUES (NULL, 'Nowy produkt dla User2', '100', '5');
```

Inserted row id: 14 (Query took 0.0020 seconds.)

[\[ Edit inline \]](#) [\[ Edit \]](#) [\[ Create PHP code \]](#)

☐ Profiling
 [\[ Edit inline \]](#)
[\[ Edit \]](#)
[\[ Explain SQL \]](#)
[\[ Create PHP code \]](#)
[\[ Refresh \]](#)

☐ Show all
 Number of rows: 25
 Filter rows: Search this table
 Sort by key: None

+ Options			▼ Id_product	name	price	Amount
<input type="checkbox"/>	 Edit	 Copy	 Delete	1	Modify Reductor	129.00 94
<input type="checkbox"/>	 Edit	 Copy	 Delete	2	Modify Femibra	139.00 67
<input type="checkbox"/>	 Edit	 Copy	 Delete	3	Gumka do włosów - Iniana	29.00 20
<input type="checkbox"/>	 Edit	 Copy	 Delete	4	Gumka do włosów - Iniana XL	49.00 30
<input type="checkbox"/>	 Edit	 Copy	 Delete	5	Gumka do włosów - welurowa	19.00 20
<input type="checkbox"/>	 Edit	 Copy	 Delete	6	Gumka do włosów - welurowa XL	29.00 20
<input type="checkbox"/>	 Edit	 Copy	 Delete	7	Pełna kuracja Modify Reductor	387.00 10
<input type="checkbox"/>	 Edit	 Copy	 Delete	8	Pełna kuracja Modify Femibra	417.00 10
<input type="checkbox"/>	 Edit	 Copy	 Delete	9	Zestaw świąteczny Modify Reductor	199.00 15
<input type="checkbox"/>	 Edit	 Copy	 Delete	10	Zestaw świąteczny Modify Femibra	256.00 1
<input type="checkbox"/>	 Edit	 Copy	 Delete	14	Nowy produkt dla User2	100.00 5

Jak widać, stworzony użytkownik posiada przywileje, które zostały mu nadane.

## Pełny skrypt SQL

```
-- phpMyAdmin SQL Dump
-- version 5.1.1
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Generation Time: Jan 22, 2022 at 03:49 PM
-- Server version: 10.4.21-MariaDB
-- PHP Version: 8.0.11

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `db_influencers_konw`
--

DELIMITER $$
```

```

-- phpMyAdmin SQL Dump
-- version 5.1.1
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Generation Time: Jan 22, 2022 at 06:54 PM
-- Server version: 10.4.21-MariaDB
-- PHP Version: 8.0.11

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `db_influencers_konw`
--

CREATE DATABASE IF NOT EXISTS `db_influencers_konw` DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci;
USE `db_influencers_konw`;

DELIMITER $$
--
-- Procedures
--

DROP PROCEDURE IF EXISTS `AllInfluencerActions`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `AllInfluencerActions` () BEGIN
    SELECT influencer.Id_influencer AS InfluencerID,
        FullName(influencer.first_name, influencer.last_name) AS FullName,
        service.name AS service,
        influenceraction.date AS Date
    FROM influencer, influenceraction, service
    WHERE influencer.Id_influencer = influenceraction.Influencer_Id_influencer
    AND influenceraction.Service_Id_service = service.Id_service
    ORDER BY Date;
END$$

DROP PROCEDURE IF EXISTS `GetFullInfluencerData`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `GetFullInfluencerData` () BEGIN
    SELECT FullName(influencer.first_name, influencer.last_name) AS FullName,
        InfluencerLevel(influencer.followers) AS InfluencerType,
        influencer.discount_code AS DiscountCode,
        address.city AS City
    FROM influencer, address
    WHERE influencer.Address_Id_address = address.Id_address
    ORDER BY InfluencerType;
END$$

DROP PROCEDURE IF EXISTS `InfluencerActions`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `InfluencerActions` (IN `idInf` INT) BEGIN
    SELECT influencer.Id_influencer AS InfluencerID,
        FullName(influencer.first_name, influencer.last_name) AS FullName,
        service.name AS service,
        influenceraction.date AS Date
    FROM influencer, influenceraction, service
    WHERE influencer.Id_influencer = idInf
    AND influencer.Id_influencer = influenceraction.Influencer_Id_influencer
    AND influenceraction.Service_Id_service = service.Id_service
    ORDER BY Date;
END$$

DROP PROCEDURE IF EXISTS `InfluencerSalary`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `InfluencerSalary` (IN `idInf` INT) BEGIN
    SELECT influencer.Id_influencer AS InfluencerID,
        FullName(influencer.first_name, influencer.last_name) AS FullName,
        influenceraction.Id_influencer_action AS ActionID,
        service.name AS service,
        salary.value AS PLN,
        salary.date AS Date
    FROM influencer, influenceraction, service, salary
    WHERE influencer.Id_influencer = idInf
    AND influencer.Id_influencer = influenceraction.Influencer_Id_influencer
    AND influenceraction.Service_Id_service = service.Id_service
    AND influencer.Id_influencer = salary.Influencer_Id_influencer

```

```

        AND salary.InfluencerAction_Id_influencer_action =
influenceraction.Id_influencer_action
        ORDER BY Date;
END$$

DROP PROCEDURE IF EXISTS `ProductStock`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `ProductStock` (IN `productID` INT, IN
`amount` INT) BEGIN
    SET @MESSAGE_TEXT = 'The state of a stock is 0 for this product!';
    SET @GET_PRODUCT_AMOUNT = (SELECT product.Amount FROM product WHERE
product.Id_product = productID LIMIT 1);

    IF @GET_PRODUCT_AMOUNT - amount <= 0 THEN
        SELECT @MESSAGE_TEXT;
    ELSE
        UPDATE product
        SET product.Amount = product.Amount - amount
        WHERE product.Id_product = productID;
    END IF;
END$$

--
-- Functions
--
DROP FUNCTION IF EXISTS `AddNumbers`$$
CREATE DEFINER=`root`@`localhost` FUNCTION `AddNumbers` (`val1` INT, `val2` INT) RETURNS
INT(11) RETURN val1 + val2$$

DROP FUNCTION IF EXISTS `FullName`$$
CREATE DEFINER=`root`@`localhost` FUNCTION `FullName` (`first_name` VARCHAR(20),
`last_name` VARCHAR(30)) RETURNS VARCHAR(50) CHARSET utf8 COLLATE utf8_unicode_ci RETURN
CONCAT(first_name, ' ', last_name)$$

DROP FUNCTION IF EXISTS `InfluencerLevel`$$
CREATE DEFINER=`root`@`localhost` FUNCTION `InfluencerLevel` (`followers` INT(11))
RETURNS VARCHAR(20) CHARSET utf8 COLLATE utf8_unicode_ci BEGIN
    DECLARE InfluencerLvl VARCHAR(20);

    IF followers >= 100000 THEN SET InfluencerLvl = 'MAKROinfluencer';
    ELSEIF (followers < 100000 AND followers >= 10000) THEN SET InfluencerLvl =
'Influencer';
    ELSE SET InfluencerLvl = 'Mikroinfluencer';
    END IF;

    RETURN InfluencerLvl;
END$$

DROP FUNCTION IF EXISTS `ValueToEuro`$$
CREATE DEFINER=`root`@`localhost` FUNCTION `ValueToEuro` (`prodValue` DECIMAL(6,2),
`newValue` DECIMAL(6,2)) RETURNS DECIMAL(6,2) RETURN prodValue / newValue$$

DELIMITER ;

-- -----
--
-- Table structure for table `address`
--
DROP TABLE IF EXISTS `address`;
CREATE TABLE `address` (
  `Id_address` int(11) NOT NULL,
  `Country_Id_country` int(11) DEFAULT NULL,
  `street` varchar(200) COLLATE utf8_unicode_ci DEFAULT NULL,
  `house_nr` varchar(10) COLLATE utf8_unicode_ci DEFAULT NULL,
  `post_code` varchar(10) COLLATE utf8_unicode_ci DEFAULT NULL,
  `city` varchar(100) COLLATE utf8_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

--
-- Dumping data for table `address`
--
INSERT INTO `address` (`Id_address`, `Country_Id_country`, `street`, `house_nr`,
`post_code`, `city`) VALUES
(1, 1, 'Puławska', '38/15', '03-232', 'Warszawa'),
(2, 2, 'Isern-Hinnerk-Weg', '14B', '22457', 'Hamburg'),
(3, 8, 'High Park Road', '20A', 'PR9 7QL', 'Southport'),
(4, 1, 'Jurajska', '4/59', '02-699', 'Warszawa'),

```

```

(5, 1, 'Kadłubka', '42/5', '71-524', 'Szczecin'),
(6, 1, 'Aleja Krakowska', '49', '05-090', 'Sękocin Stary'),
(7, 1, 'Nowowiejska', '15', '06-500', 'Mława'),
(8, 8, 'Saxonbury Way', '94', 'PE2 9FB', 'Cambridgeshire'),
(9, 1, 'Marii Dąbrowskiej', '96/9', '97-300', 'Piotrków Trybunalski'),
(10, 1, 'Niemeńska', '66', '60-412', 'Poznań'),
(11, NULL, 'Oakstreet', '23', '1', '890FR');

-- -----

--
-- Table structure for table `country`
--

DROP TABLE IF EXISTS `country`;
CREATE TABLE `country` (
  `Id_country` int(11) NOT NULL,
  `long_name` varchar(200) COLLATE utf8_unicode_ci DEFAULT NULL,
  `short_name` char(3) COLLATE utf8_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

--
-- Dumping data for table `country`
--

INSERT INTO `country` (`Id_country`, `long_name`, `short_name`) VALUES
(1, 'Polska', 'PL'),
(2, 'Niemcy', 'DE'),
(3, 'Rosja', 'RU'),
(4, 'Szwajcaria', 'CH'),
(5, 'Ukraina', 'UA'),
(6, 'Białoruś', 'BY'),
(7, 'Czarnogóra', 'MNE'),
(8, 'Wielka Brytania', 'GB'),
(9, 'Stany Zjednoczone', 'USA'),
(10, 'Kanada', 'CA');

-- -----

--
-- Stand-in structure for view `full_address_data`
-- (See below for the actual view)
--

DROP VIEW IF EXISTS `full_address_data`;
CREATE TABLE `full_address_data` (
  `first_name` varchar(50)
, `last_name` varchar(100)
, `street` varchar(200)
, `house_nr` varchar(10)
, `post_code` varchar(10)
, `city` varchar(100)
, `long_name` varchar(200)
);

-- -----

--
-- Table structure for table `influencer`
--

DROP TABLE IF EXISTS `influencer`;
CREATE TABLE `influencer` (
  `Id_influencer` int(11) NOT NULL,
  `Address_Id_address` int(11) DEFAULT NULL,
  `first_name` varchar(50) COLLATE utf8_unicode_ci DEFAULT NULL,
  `last_name` varchar(100) COLLATE utf8_unicode_ci DEFAULT NULL,
  `nickname` varchar(20) COLLATE utf8_unicode_ci DEFAULT NULL,
  `discount_code` varchar(15) COLLATE utf8_unicode_ci DEFAULT NULL,
  `followers` int(11) DEFAULT NULL,
  `account_number` varchar(26) COLLATE utf8_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

--
-- Dumping data for table `influencer`
--

INSERT INTO `influencer` (`Id_influencer`, `Address_Id_address`, `first_name`,
`last_name`, `nickname`, `discount_code`, `followers`, `account_number`) VALUES
(1, 4, 'Ismenka', 'Stelmaszczyk', 'ismena_stelmaszczyk', 'ismena15', 47600, NULL),

```

```

(2, 2, 'Paulina', 'Guzińska', 'paulina_guzinska', 'paulina15', 264000, NULL),
(3, 5, 'Bartosz', 'Smeda', 'smeda.triathlon', 'triathlon15', 801, NULL),
(4, 3, 'John', 'Smith', 'john_smith', 'john15', 45000, '12445588237595145212547719'),
(5, 1, 'Maria', 'Wrześniak', 'mariaaa', 'marysia15', 53000, NULL),
(6, 6, 'Klaudia', 'Michalak', 'klaudia_michalak', 'klaudia15', 4123, NULL),
(7, 7, 'Aleksander', 'Makosa', 'alexi', 'alexi15', 89000, NULL),
(8, 8, 'Aleksa', 'Woznicki', 'aleksa_woznikci', 'woznikci15', 632, NULL),
(9, 9, 'Rafał', 'Piotrowski', 'rafal_p', 'rafal15', 7522, NULL),
(10, 10, 'Marcin', 'Wiesiuk', 'marcinek_w', 'marcinek15', 65002, NULL);

-- -----

--
-- Table structure for table `influenceraction`
--

DROP TABLE IF EXISTS `influenceraction`;
CREATE TABLE `influenceraction` (
  `Id_influencer_action` int(11) NOT NULL,
  `Influencer_Id_influencer` int(11) DEFAULT NULL,
  `Service_Id_service` int(11) DEFAULT NULL,
  `date` date DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

--
-- Dumping data for table `influenceraction`
--

INSERT INTO `influenceraction` (`Id_influencer_action`, `Influencer_Id_influencer`,
`Service_Id_service`, `date`) VALUES
(1, 1, 8, '2021-09-30'),
(2, 1, 8, '2021-09-30'),
(3, 1, 1, '2021-09-15'),
(4, 2, 7, '2021-10-13'),
(5, 4, 10, '2021-11-02'),
(6, 3, 10, '2021-11-14'),
(7, 5, 3, '2021-10-19'),
(8, 7, 5, '2021-11-02'),
(9, 10, 9, '2021-11-03'),
(10, 8, 6, '2021-08-10'),
(11, 1, 1, '2021-12-01'),
(13, 4, 4, '2021-12-04'),
(14, 1, 9, '2022-01-13'),
(15, 4, 1, '2022-01-01'),
(16, 4, 4, '2022-01-04'),
(17, 2, 8, NULL);

--
-- Triggers `influenceraction`
--

DROP TRIGGER IF EXISTS `new_salary`;
DELIMITER $$
CREATE TRIGGER `new_salary` AFTER INSERT ON `influenceraction` FOR EACH ROW INSERT INTO
`salary` (`Id_salary`, `Influencer_Id_influencer`,
`InfluencerAction_Id_influencer_action`, `value`, `date`) VALUES (NULL,
NEW.Influencer_Id_influencer, NEW.Id_influencer_action, (SELECT value FROM `service`
WHERE Id_service = NEW.Service_Id_service), DATE(NOW()))
$$
DELIMITER ;

-- -----

--
-- Table structure for table `product`
--

DROP TABLE IF EXISTS `product`;
CREATE TABLE `product` (
  `Id_product` int(11) NOT NULL,
  `name` varchar(100) COLLATE utf8_unicode_ci DEFAULT NULL,
  `price` decimal(6,2) DEFAULT NULL,
  `Amount` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

--
-- Dumping data for table `product`
--

INSERT INTO `product` (`Id_product`, `name`, `price`, `Amount`) VALUES

```



```

(1, 'Modify Reductor', '129.00', 94),
(2, 'Modify Femibra', '139.00', 67),
(3, 'Gumka do włosów - lniana ', '29.00', 20),
(4, 'Gumka do włosów - lniana XL', '49.00', 30),
(5, 'Gumka do włosów - welurowa', '19.00', 20),
(6, 'Gumka do włosów - welurowa XL', '29.00', 20),
(7, 'Pełna kuracja Modify Reductor', '387.00', 10),
(8, 'Pełna kuracja Modify Femibra', '417.00', 10),
(9, 'Zestaw świąteczny Modify Reductor', '199.00', 15),
(10, 'Zestaw świąteczny Modify Femibra', '256.00', 1),
(14, 'Nowy produkt dla User2', '100.00', 5);

-- -----

--
-- Table structure for table `salary`
--

DROP TABLE IF EXISTS `salary`;
CREATE TABLE `salary` (
  `Id_salary` int(11) NOT NULL,
  `Influencer_Id_influencer` int(11) DEFAULT NULL,
  `InfluencerAction_Id_influencer_action` int(11) DEFAULT NULL,
  `value` decimal(6,2) DEFAULT NULL,
  `date` date DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

--
-- Dumping data for table `salary`
--

INSERT INTO `salary` (`Id_salary`, `Influencer_Id_influencer`,
`InfluencerAction_Id_influencer_action`, `value`, `date`) VALUES
(1, 1, 1, '30.00', '2021-09-30'),
(2, 1, 2, '30.00', '2021-09-30'),
(3, 1, 3, '50.00', '2021-09-15'),
(4, 2, 4, '100.00', '2021-10-13'),
(5, 4, 5, '350.00', '2021-11-12'),
(6, 3, 6, '350.00', '2021-11-14'),
(7, 5, 7, '100.00', '2021-10-19'),
(8, 7, 8, '500.00', '2021-11-02'),
(9, 10, 9, '40.00', '2021-11-03'),
(10, 8, 10, '200.00', '2021-08-10'),
(13, 4, 13, '250.00', '2021-12-04'),
(14, 1, 14, '40.00', '2022-01-13'),
(15, 4, 15, '50.00', '2022-01-13'),
(16, 4, 16, '250.00', '2022-01-13'),
(17, 2, 17, '30.00', '2022-01-20');

-- -----

--
-- Table structure for table `service`
--

DROP TABLE IF EXISTS `service`;
CREATE TABLE `service` (
  `Id_service` int(11) NOT NULL,
  `name` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `value` decimal(6,2) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

--
-- Dumping data for table `service`
--

INSERT INTO `service` (`Id_service`, `name`, `value`) VALUES
(1, 'Relacja ', '50.00'),
(2, 'Relacja + swipe', '150.00'),
(3, 'Post bez produktu', '100.00'),
(4, 'Post z produktem', '250.00'),
(5, 'Udział w evencie marki', '500.00'),
(6, 'Logo marki na stronie', '200.00'),
(7, 'Oznaczenie na story', '100.00'),
(8, 'Użycie kodu influencer', '30.00'),
(9, 'Polecenie zakupu bez kodu', '40.00'),
(10, 'Pakiet (post + relacja + swipe)', '350.00');

-- -----

```

```

--
-- Table structure for table `shipping`
--

DROP TABLE IF EXISTS `shipping`;
CREATE TABLE `shipping` (
  `Id_shipping` int(11) NOT NULL,
  `Influencer_Id_influencer` int(11) DEFAULT NULL,
  `Product_Id_product` int(11) DEFAULT NULL,
  `date` date DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

--
-- Dumping data for table `shipping`
--

INSERT INTO `shipping` (`Id_shipping`, `Influencer_Id_influencer`, `Product_Id_product`,
`date`) VALUES
(1, 1, 7, '2021-09-08'),
(2, 2, 7, '2021-11-07'),
(3, 3, 9, '2021-08-18'),
(4, 4, 1, '2021-11-19'),
(5, 5, 2, '2021-10-18'),
(6, 5, 5, '2021-10-18'),
(7, 6, 10, '2020-12-12'),
(8, 7, 9, '2021-11-17'),
(9, 8, 2, '2021-07-11'),
(10, 9, 1, '2021-11-06'),
(11, 10, 9, '2021-11-15'),
(12, 1, 4, '2021-11-08'),
(13, 6, 2, '2022-01-20');

--
-- Triggers `shipping`
--

DROP TRIGGER IF EXISTS `update_stock`;
DELIMITER $$
CREATE TRIGGER `update_stock` AFTER INSERT ON `shipping` FOR EACH ROW BEGIN
  CALL ProductStock (NEW.Product_Id_product, 1);
END
$$
DELIMITER ;

--
-----

--
-- Structure for view `full_address_data` exported as a table
--

DROP TABLE IF EXISTS `full_address_data`;
CREATE TABLE `full_address_data` (
  `first_name` varchar(50) COLLATE utf8_unicode_ci DEFAULT NULL,
  `last_name` varchar(100) COLLATE utf8_unicode_ci DEFAULT NULL,
  `street` varchar(200) COLLATE utf8_unicode_ci DEFAULT NULL,
  `house_nr` varchar(10) COLLATE utf8_unicode_ci DEFAULT NULL,
  `post_code` varchar(10) COLLATE utf8_unicode_ci DEFAULT NULL,
  `city` varchar(100) COLLATE utf8_unicode_ci DEFAULT NULL,
  `long_name` varchar(200) COLLATE utf8_unicode_ci DEFAULT NULL
);

--
-- Indexes for dumped tables
--

--
-- Indexes for table `address`
--

ALTER TABLE `address`
  ADD PRIMARY KEY (`Id_address`),
  ADD KEY `Country_Id_country` (`Country_Id_country`);

--
-- Indexes for table `country`
--

ALTER TABLE `country`
  ADD PRIMARY KEY (`Id_country`);

--
-- Indexes for table `influencer`

```

```

--
ALTER TABLE `influencer`
  ADD PRIMARY KEY (`Id_influencer`),
  ADD KEY `Address_Id_address` (`Address_Id_address`);

--
-- Indexes for table `influenceraction`
--
ALTER TABLE `influenceraction`
  ADD PRIMARY KEY (`Id_influencer_action`),
  ADD KEY `Influencer_Id_influencer` (`Influencer_Id_influencer`),
  ADD KEY `Service_Id_service` (`Service_Id_service`);

--
-- Indexes for table `product`
--
ALTER TABLE `product`
  ADD PRIMARY KEY (`Id_product`);

--
-- Indexes for table `salary`
--
ALTER TABLE `salary`
  ADD PRIMARY KEY (`Id_salary`),
  ADD KEY `InfluencerAction_Id_influencer_action`
(`InfluencerAction_Id_influencer_action`),
  ADD KEY `Influencer_Id_influencer` (`Influencer_Id_influencer`);

--
-- Indexes for table `service`
--
ALTER TABLE `service`
  ADD PRIMARY KEY (`Id_service`);

--
-- Indexes for table `shipping`
--
ALTER TABLE `shipping`
  ADD PRIMARY KEY (`Id_shipping`),
  ADD KEY `Influencer_Id_influencer` (`Influencer_Id_influencer`),
  ADD KEY `Product_Id_product` (`Product_Id_product`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `address`
--
ALTER TABLE `address`
  MODIFY `Id_address` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=12;

--
-- AUTO_INCREMENT for table `country`
--
ALTER TABLE `country`
  MODIFY `Id_country` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=11;

--
-- AUTO_INCREMENT for table `influencer`
--
ALTER TABLE `influencer`
  MODIFY `Id_influencer` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=11;

--
-- AUTO_INCREMENT for table `influenceraction`
--
ALTER TABLE `influenceraction`
  MODIFY `Id_influencer_action` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=18;

--
-- AUTO_INCREMENT for table `product`
--
ALTER TABLE `product`
  MODIFY `Id_product` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=15;

--
-- AUTO_INCREMENT for table `salary`
--

```

```

ALTER TABLE `salary`
  MODIFY `Id_salary` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=18;

--
-- AUTO_INCREMENT for table `service`
--
ALTER TABLE `service`
  MODIFY `Id_service` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=11;

--
-- AUTO_INCREMENT for table `shipping`
--
ALTER TABLE `shipping`
  MODIFY `Id_shipping` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=14;

--
-- Constraints for dumped tables
--

--
-- Constraints for table `address`
--
ALTER TABLE `address`
  ADD CONSTRAINT `address_ibfk_1` FOREIGN KEY (`Country_Id_country`) REFERENCES
`country` (`Id_country`);

--
-- Constraints for table `influencer`
--
ALTER TABLE `influencer`
  ADD CONSTRAINT `influencer_ibfk_1` FOREIGN KEY (`Address_Id_address`) REFERENCES
`address` (`Id_address`);

--
-- Constraints for table `influenceraction`
--
ALTER TABLE `influenceraction`
  ADD CONSTRAINT `influenceraction_ibfk_1` FOREIGN KEY (`Influencer_Id_influencer`)
REFERENCES `influencer` (`Id_influencer`),
  ADD CONSTRAINT `influenceraction_ibfk_2` FOREIGN KEY (`Service_Id_service`) REFERENCES
`service` (`Id_service`);

--
-- Constraints for table `salary`
--
ALTER TABLE `salary`
  ADD CONSTRAINT `salary_ibfk_1` FOREIGN KEY (`InfluencerAction_Id_influencer_action`)
REFERENCES `influenceraction` (`Id_influencer_action`),
  ADD CONSTRAINT `salary_ibfk_2` FOREIGN KEY (`Influencer_Id_influencer`) REFERENCES
`influencer` (`Id_influencer`);

--
-- Constraints for table `shipping`
--
ALTER TABLE `shipping`
  ADD CONSTRAINT `shipping_ibfk_1` FOREIGN KEY (`Influencer_Id_influencer`) REFERENCES
`influencer` (`Id_influencer`),
  ADD CONSTRAINT `shipping_ibfk_2` FOREIGN KEY (`Product_Id_product`) REFERENCES
`product` (`Id_product`);
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```