AUTORZY

1. ***Paweł Sosnowski***
2. ***Maciej Słowik***

AUTORZY

1. ***Paweł Sosnowski***
2. ***Maciej Słowik***

Użytkownicy systemu

1. Administrator
2. Dyrektor platformy
3. Koordynator webinarów
4. Koordynator studiów
5. Koordynator kursów
6. Pracownik sekretariatu
7. Wykładowcy
8. Tłumacze
9. Uczestnik (osoba z założonym kontem)
10. Gość

Funkcje dla każdego użytkownika

1. **Administrator**
    - Możliwość usuwania nagrań z webinarów
    - Możliwość edytowania danych uzytkowników systemu
    - Możliwość edytowania ofert z webinarami
    - Możliwość tworzenia i odtwarzania kopii zapasowych
2. **Dyrektor platformy**
    - Odraczanie płatności dla wybranych klientów
    - Dostęp do raportów generowanych przez system (raporty finansowe, raporty z "listą dłużników", raport związane z przyszłymi wydarzeniami i frekwencją na nich)
    - Możliwość zwalniania i zatrudniania pracowników
    - Możliwość wyrzucania uczestników z kursów/webinarów/studiów
3. **Koordynator webinarów**
    - Dodawanie i usuwanie webinarów oraz modyfikowanie ich właściwości
    - Przypisywanie prowadzących czy tłumaczy do webinarów
4. **Koordynator kursów**
    - Dodawanie i usuwanie kursów do oferty
    - Możliwość modyfikowania właściwości kursów i wybieranie wykładowców czy tłumaczy oraz określanie typu modułów
    - Możliwość decydowania o zaliczeniu kursu (może zmienić decyzję systemu w szeczólnych przypadkach nawet jak uczestnik zaliczył 80% modułów)
5. **Koordynator studiów**

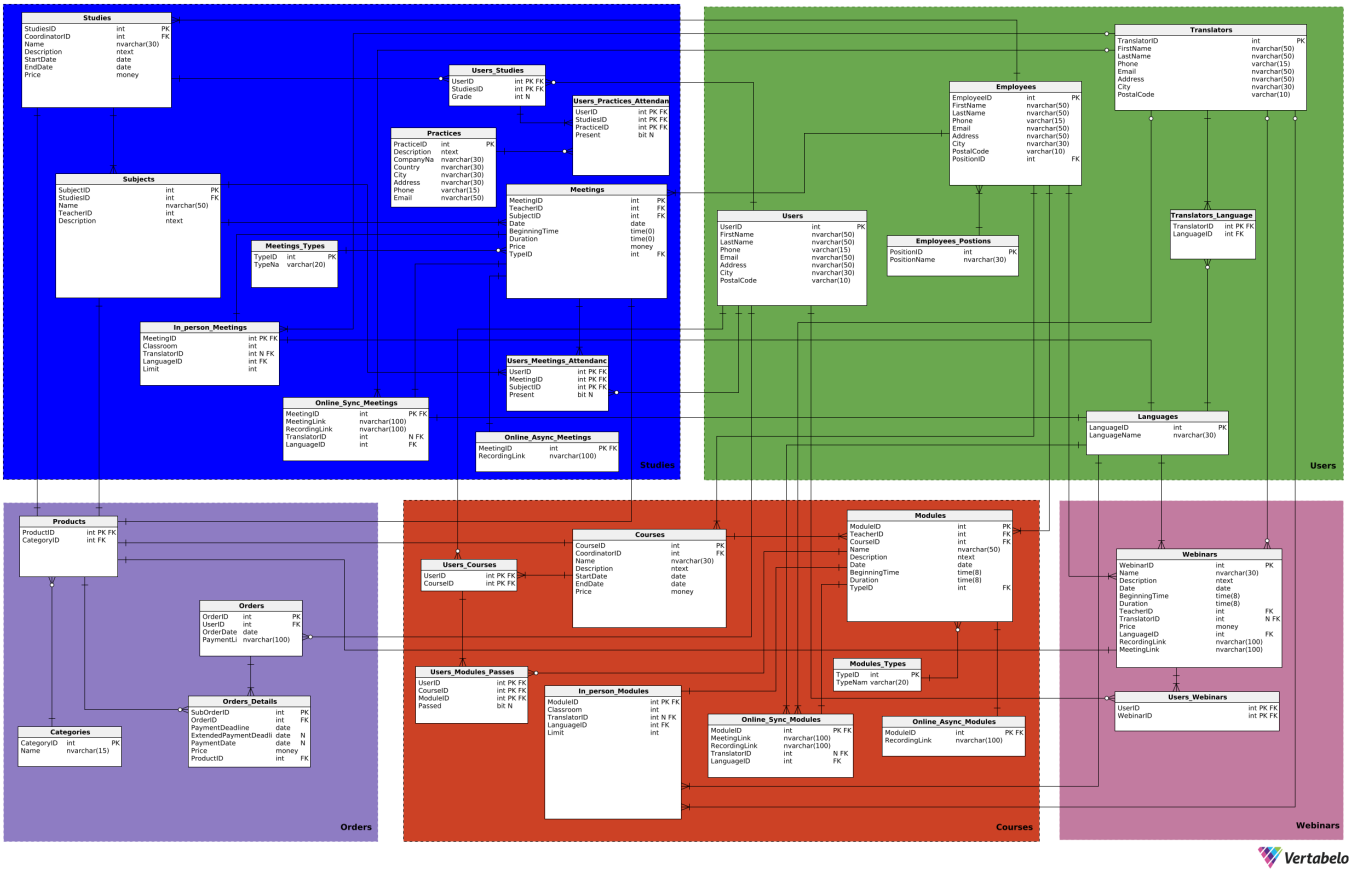
    - Dodawanie i usuwanie studiów do oferty
    - Możliwość tworzenia sylabusa studiów poprzez dodawanie przedmiotów:
        - Możliwość przypisywania wykładowców i tłumaczy do danych przedmiotów
        - Możliwość określenia typu danego spotkania przedmotu (stacjnonarnie, online, hybrydowo)
    - Możliwość dodawania i usuwania uczestników danych studiów
    - Możliwość modyfikowania harmonogramu danych studiów
    - Możliwość wpisania oceny końcowej z egzaminu
6. **Pracownik sekretariatu**
    - Możliwość tworzenia harmonogramu zajęć na studiach/kursach i zarządzanie nim
    - Dostęp do raportów generowanych przez system (raporty finansowe, raporty z "listą dłużników", raport związane z przyszłymi wydarzeniami i frekwencją na nich)
    - Możliwość przyjmowania zgłoszeń od użytkowników systemu
    - Dostęp do danych uczestników kursów/studiów/webinarów i możliwość ich modyfikacji
7. **Wykładowcy**
    - Dostęp do swojego harmonogramu zajęć i ich szczegółów
    - Możliwość modyfikacji frekwencji uczestników na swoich zajęciach
    - Dostęp do raportu dotyczącego nieobecności uczestników na swoich zajęciach
8. **Tłumacze**
    - Dostęp do swojego harmonogramu zajęć i ich szczegółów
9. **Uczestnik (osoba z założonym kontem)**
    - Możliwość zmiany swoich danych osobowych
    - Możliwość dodawania produktów z oferty do koszyka
    - Możliwość wybrania sposobu płatności w przypadku kursu/studiów (zaliczka czy od razu całość)
    - Możliwość przeglądania aktualnych ofert wraz z ich szczegółami
    - Dostęp do harmonogramu kursów/studiów/webinarów, na które wykupił dostęp lub się zapisał
    - Dostęp do nagrań z wykupionych ofert
    - Możliwość zgłoszenia próśb np. o usunięcie z listy uczestników danego kursu/studium
10. **Gość**
    - Możliwość przeglądania aktualnych ofert wraz z ich szczegółami
    - Możliwość stworzenia konta

Funkcje systemu

- Sprawdzanie potwierdzenia płatności (czy udana / czy nieudana)
- Możliwość blokowania dostępu do zasobów w przypadku opoźnienia w płatnościach (nagrania, kursy)
- Rejestrowanie opłat w systemie
- Udzielenie użytkownikowi dostępu do danej usługi w przypadku potwierdzenia płatności
- Możliwość sprawdzenia i zmienienia obecności w wyniku obejrzenia nagrania
- Generowanie raportów:
    - generowanie sprawozdań finansowych, czyli zestawienie przychodów dla każdego kursu/webinaru/studium
    - generowanie listy osób, które zwlekają z opłatą, lista dłużników
    - generowanie listy zapisanych użytkowników na dane wydarzenie oraz informacji o tym wydarzeniu: typ spotkania
    - generowanie raportu o frekwencji dla każdego szkolenia

    - generowanie raportu o kolizjach w harmonogramie zajęć dla uczestników
- Wyznaczanie ilości wolnych miejsc na studiach/kursach i pilnowanie limitów
- Wysyłanie powiadomień o zaległych płatnościach
- Blokowanie dostępu do nagrań z webinarów po upływie 30 dni
- Generowanie linku do płatności dla użytkownika

Diagram bazy danych



Vertabelo

Opis poszczególnych tabel

Kategoria Users

Tabela **Users**

Zawiera ona informacje o użytkownikach:

- **UserID** [int] - klucz główny, identyfikator użytkownika
- **FirstName** [nvarchar(50)] - imię użytkownika
- **LastName** [nvarchar(50)] - nazwisko użytkownika
- **Phone** [varchar(15), unique] - numer telefonu użytkownika
    - warunki: LEN(Phone) = 15 AND ISNUMERIC(Phone) = 1
- **Email** [nvarchar(50), unique] - email użytkownika
    - warunki: Email LIKE '%_@%.%'
- **Address** [nvarchar(50)] - adres użytkownika
- **City** [nvarchar(30)] - miasto użytkownika
- **PostalCode** [varcahr(10)] - kod pocztowy użytkownika

```
CREATE TABLE Users (
   UserID int  NOT NULL,
   FirstName nvarchar(50)  NOT NULL,
   LastName nvarchar(50)  NOT NULL,
   Phone varchar(15)  NOT NULL CHECK (LEN(Phone) = 15 and ISNUMERIC(Phone) = 1),
   Email nvarchar(50)  NOT NULL CHECK (Email LIKE '%_@%.%'),
   Address nvarchar(50)  NOT NULL,
   City nvarchar(30)  NOT NULL,
   PostalCode varchar(10)  NOT NULL,
   CONSTRAINT UserPhone UNIQUE (Phone),
   CONSTRAINT UserEmail UNIQUE (Email),
   CONSTRAINT UserID PRIMARY KEY  (UserID)
);
```

Tabela **Employees**

Zawiera informacje o pracownikach:

- **EmployeeID** [int] - klucz główny, identyfikator pracownika
- **FirstName** [nvarchar(50)] - imię pracownika
- **LastName** [nvarchar(50)] - nazwisko pracownika
- **Phone** [varcahr(15), unique] - numer telefonu pracownika
    - warunki: LEN(Phone) = 15 AND ISNUMERIC(Phone) = 1

- **Email** [nvarchar(50), unique] - email pracownika
    - warunki: Email LIKE '%_@%.%'
- **Address** [nvarchar(50)] - adres pracownika
- **City** [nvarchar(30)] - miasto pracownika
- **PostalCode** [varchar(10)] - kod pocztowy pracownika
- **PositionID** [int] - identyfikator pozycji pracownika

```
CREATE TABLE Employees (
   EmployeeID int  NOT NULL,
   FirstName nvarchar(50)  NOT NULL,
   LastName nvarchar(50)  NOT NULL,
   Phone varchar(15)  NOT NULL CHECK (LEN(Phone) = 15 and ISNUMERIC(Phone) = 1),
   Email nvarchar(50)  NOT NULL CHECK (Email LIKE '%_@%.%'),
   Address nvarchar(50)  NOT NULL,
   City nvarchar(30)  NOT NULL,
   PostalCode varchar(10)  NOT NULL,
   PositionID int  NOT NULL,
   CONSTRAINT EmployeePhone UNIQUE (Phone),
   CONSTRAINT EmployeeEmail UNIQUE (Email),
   CONSTRAINT Employees_pk PRIMARY KEY  (EmployeeID)
);
```

Tabela **Employees_Positions**

Zawiera informacje o możliwych pozycjach pracowników:

- **PositionID** [int] - klucz główny, identyfikator pozycji
- **PositionName** [nvarchar(30)] - nazwa pozycji

```
CREATE TABLE Employees_Postions (
   PositionID int  NOT NULL,
   PositionName nvarchar(30)  NOT NULL,
   CONSTRAINT Employees_Postions_pk PRIMARY KEY  (PositionID)
);
```

Tabela **Translators**

Zwiera informacje o tłumaczach:

- **TranslatorID** [int] - klucz główny, identyfikator tłumacza
- **FirstName** [nvarchar(50)] - imię tłumacza
- **LastName** [nvarchar(50)] - nazwisko tłumacza
- **Phone** [varchar(15), unique] - numer telefonu tłumacza
    - warunki: LEN(Phone) = 15 AND ISNUMERIC(Phone) = 1
- **Email** [nvarchar(50), unique] - email tłumacza
    - warunki: Email LIKE '%_@%.%'

- **Address** [nvarchar(50)] - adres tłumacza
- **City** [nvarchar(30)] - miasto tłumacza
- **PostalCode** [varchar(10)] - kod pocztowy tłumacza

```
CREATE TABLE Translators (
   TranslatorID int  NOT NULL,
   FirstName nvarchar(50)  NOT NULL,
   LastName nvarchar(50)  NOT NULL,
   Phone varchar(15)  NOT NULL CHECK (LEN(Phone) = 15 and ISNUMERIC(Phone) = 1),
   Email nvarchar(50)  NOT NULL CHECK (Email LIKE '%_@%.%'),
   Address nvarchar(50)  NOT NULL,
   City nvarchar(30)  NOT NULL,
   PostalCode varchar(10)  NOT NULL,
   CONSTRAINT TranslatorPhone UNIQUE (Phone),
   CONSTRAINT TranslatorEmail UNIQUE (Email),
   CONSTRAINT UserID PRIMARY KEY  (TranslatorID)
);
```

Tabela **Translator_Languages**

Zawiera informajce o językach jakimi posługują się tłumacze:

- **TranslatorID** [int] - część klucza głównego, identyfikator tłumacza
- **LanguageID** [int] - klucz obcy, identyfikator języka

```
CREATE TABLE Translators_Languages (
   TranslatorID int  NOT NULL,
   LanguageID int  NOT NULL,
   CONSTRAINT Translators_Languages_pk PRIMARY KEY  (TranslatorID)
);
```

Tabela **Languages**

Zawiera możliwe języki tłumaczenia:

- **LanguageID** [int] - klucz główny, identyfikator języka
- **LanguaneName** [nvarchar(30)] - nazwa języka

```
CREATE TABLE Languages (
   LanguageID int  NOT NULL,
   LanguageName nvarchar(30)  NOT NULL,
   CONSTRAINT Languages_pk PRIMARY KEY  (LanguageID)
);
```

Kategoria Courses

Tabela **Courses**

Zawiera informacje dotyczące kursów:

- **CourseID** [int] - klucz główny, identyfikator kursu
- **CoordinatorID** [int] - identyfikator koordynatora kursu
- **Name** [nvarchar(30)] - nazwa kursu
- **Description** [ntext] - opis kursu
- **StartDate** [date] - data rozpoczęcia kursu w formacie 'rok-miesiąc-dzień'
- **EndDate** [date] - data zakończenia kursu w formacie 'rok-miesiąc-dzień' (musi być późniejsza od daty rozpoczęcia kursu)
- **Price** [money] - cena kursu (domyślna 100, musi być dodatnia)

```
CREATE TABLE Courses (
   CourseID int  NOT NULL,
   CoordinatorID int  NOT NULL,
   Name nvarchar(30)  NOT NULL,
   Description ntext  NOT NULL,
   StartDate date  NOT NULL,
   EndDate date  NOT NULL CHECK (EndDate > StartDate),
   Price money  NOT NULL DEFAULT 100 CHECK (Price > 0),
   CONSTRAINT Courses_pk PRIMARY KEY  (CourseID)
);
```

Tabela In-**person_Modules**

Zawiera informacje o modułach odbywających się stacjonarnie:

- **ModuleID** [int] - klucz główny, identyfikator modułu
- **Classroom** [int] - numer sali, w której odbywają się zajęcia
- **TransalatorID** [int] - identyfikator tłumacza
- **Limit** [int] - limit osób mogących uczestniczyć w danych zajęciach

```
CREATE TABLE In-person_Modules (
   ModuleID int  NOT NULL,
   Classroom int  NOT NULL,
   TranslatorID int  NULL,
   Limit int  NOT NULL,
   CONSTRAINT In-person_Modules_pk PRIMARY KEY  (ModuleID)
);
```

Tabela **Modules**

Zawiera informacje dotyczące modułów:

- **ModuleID** [int] - klucz główny, identyfikator modułu
- **CourseID** [int] - identyfikator kursu, do którego należy dany moduł
- **Name** [nvarchar(50)] - nazwa modułu
- **Description** [ntext] - opis modułu
- **TeacherID** [int] - identyfikator prowadzącego dany moduł
- **Date** [date] - termin odbywania się modułu w formacie 'rok-miesiąc-dzień'
- **BeginningTime** [time(8)] - godzina rozpoczęcia modułu w formacie 'godzina:minuty:sekundy'
- **Duration** [time(8)] - czas trwania modułu w formacie 'godziny:minuty:sekundy' (domyślny 1g 30min, musi być większy od 0)
- **TypeID** [int] - identyfikator typu modułu

```
CREATE TABLE Modules (
   ModuleID int  NOT NULL,
   CourseID int  NOT NULL,
   Name nvarchar(50)  NOT NULL,
   Description ntext  NOT NULL,
   TeacherID int  NOT NULL,
   Date date  NOT NULL,
   BeginningTime time(8)  NOT NULL,
   Duration time(8)  NOT NULL DEFAULT '01:30:00' CHECK (Duration > '00:00:00'),
   TypeID int  NOT NULL,
   CONSTRAINT Modules_pk PRIMARY KEY  (ModuleID)
);
```

Tabela **Modules_Types**

Zawiera informację o typie danego modułu:

- **TypeID** [int] - identyfikator typu modułu
- **TypeName** [varchar(20)] - nazwa typu modułu, określająca sposób jego odbywania się (domyślna 'In-person'):
  - 'In-person' - stacjonarnie
  - 'Online Sync' - online synchronicznie
  - 'Online Async' - online asynchronicznie
  - 'Hybrid' - hybrydowo

```
CREATE TABLE Modules_Types (
   TypeID int  NOT NULL,
   TypeName varchar(20)  NOT NULL DEFAULT 'In-person' CHECK (TypeName IN ('In-
person', 'Online Sync', 'Online Async', 'Hybrid')),
   CONSTRAINT TypeID PRIMARY KEY  (TypeID)
);
```

Tabela **Online_Async_Modules**

Zawiera informacje o modułach odbywających się online asynchronicznie:

- **ModuleID** [int] - identyfikator modułu
- **RecordingLink** [nvarchar(100)] - link do nagrania

```
CREATE TABLE Online_Async_Modules (
   ModuleID int  NOT NULL,
   RecordingLink nvarchar(100)  NOT NULL,
   CONSTRAINT Online_Async_Modules_pk PRIMARY KEY  (ModuleID)
);
```

Tabela **Online_Sync_Modules**

Zawiera informacje o modułach odbywających się online synchronicznie:

- **ModuleID** [int] - identyfikator modułu
- **MeetingLink** [nvarchar(100)] - link do spotkania
- **RecordingLink** [nvarchar(100)] - link do nagrania ze spotkania
- **TranslatorID** [int] - identyfikator tłumacza

```
CREATE TABLE Online_Sync_Modules (
   ModuleID int  NOT NULL,
   MeetingLink nvarchar(100)  NOT NULL,
   RecordingLink nvarchar(100)  NOT NULL,
   TranslatorID int  NULL,
   CONSTRAINT Online_Sync_Modules_pk PRIMARY KEY  (ModuleID)
);
```

Tabela **Users_Courses**

Zawiera informacje o tym, na jakie kursy jest zapisany dany użytkownik:

- **UserID** [int] - identyfikator użytkownika
- **CourseID** [int] - identyfikator kursu

```
CREATE TABLE Users_Courses (
   UserID int  NOT NULL,
   CourseID int  NOT NULL,
   CONSTRAINT Users_Courses_pk PRIMARY KEY  (UserID,CourseID)
);
```

Tabela **Users_Modules_Passes**

Zawiera informację o zaliczeniu poszczególnych modułów przez danego użytkownika:

- **UserID** [int] - identyfikator użytkownika
- **CourseID** [int] - identyfikator kursu, do którego należy dany moduł
- **ModuleID** [int] - identyfikator modułu
- **Passed** [bit] - informacja, czy moduł został zaliczony przez danego użytkownika (1 - zaliczony, 0 - niezaliczony)

```
CREATE TABLE Users_Modules_Passes (
   UserID int  NOT NULL,
   CourseID int  NOT NULL,
   ModuleID int  NOT NULL,
   Passed bit  NULL,
   CONSTRAINT UserID PRIMARY KEY  (ModuleID,UserID,CourseID)
);
```

Kategoria Studies

Tabela **Studies**

Zawiera informajce o dostępnych studiach:

- **StudiesID** [int] - klucz główny, identyfikator studiów
- **CoordinatorID** [int] - identyfikator kordynatora studiów
- **Name** [nvarchar(30)] - nazwa studiów
- **Description** [ntext] - obpis studiów
- **StartDate** [date] - data rozpoczęcia studiów
- **EndDate** [date] - data zakończenia studiów
- **Price** [money] - cena wpisowego na studiach
    - warunki: Price > 0
    - wartość domyślna: 1200

```
CREATE TABLE Studies (
   StudiesID int  NOT NULL,
   CoordinatorID int  NOT NULL,
   Name nvarchar(30)  NOT NULL,
   Description ntext  NOT NULL,
   StartDate date  NOT NULL,
   EndDate date  NOT NULL,
   Price money  NOT NULL DEFAULT 1200 CHECK (Price > 0),
   CONSTRAINT Studies_pk PRIMARY KEY  (StudiesID)
);
```

Tabela **Subjects**

Zawiera informajce o przedmiotach:

- **SubjectID** [int] - klucz główny, identyfikator przedmiotu
- **StudiesID** [int] - klucz obcy, identyfikator studiów
- **Name** [nvarchar(50)] - nazwa przedmiotu
- **TeacherID** [int] - identyfikator koordynatora przedmiotu
- **Description** [ntext] - opis przedmiotu

```
CREATE TABLE Subjects (
    SubjectID int  NOT NULL,
    StudiesID int  NOT NULL,
    Name nvarchar(50)  NOT NULL,
    TeacherID int  NOT NULL,
    Description ntext  NOT NULL,
    CONSTRAINT Subjects_pk PRIMARY KEY  (SubjectID)
);
```

Tabela **Meetings**

Zawiera informacje o pojedynczym spotkaniu na studiach z danego przedmiotu:

- **MeetingID** [int] - klucz główny, identyfikator spotkania
- **TeacheID** [int] - klucz obcy, identyfikator nauczyciela
- **SubjectID** [int] - klucz obcy - identyfikator przedmiotu, z którego jest dane spotkanie
- **Date** [date] - data spotkania
- **BeginningTime** [time(0)] - godzina rozpoczęcia spotkania
- **Duration** [time(0)] - czas trwania spotkania
    - warunki: Duration > '00:00:00'
    - wartość domyślna: '01:30:00'
- **Price** [money] - cena pojedynczego spotkania
    - warunki: Price > 0
    - wartość domyślna: 120
- **TypeID** [int] - klucz obcy, identyfikator typu spotkania np. stacjonarne itd.

```
CREATE TABLE Meetings (
    MeetingID int  NOT NULL,
    TeacherID int  NOT NULL,
    SubjectID int  NOT NULL,
    Date date  NOT NULL,
    BeginningTime time(0)  NOT NULL,
    Duration time(0)  NOT NULL DEFAULT 01:30:00 CHECK (Duration > '00:00:00'),
    Price money  NOT NULL DEFAULT 120 CHECK (Price > 0),
```

```
    TypeID int  NOT NULL,
    CONSTRAINT MeetingID PRIMARY KEY  (MeetingID)
);
```

Tabela **Meeting_Types**

Zawiera informacje o rodzajach typów spotkań:

- **TypeID** [int] - klucz główny, identyfikator typu
- **TypeName** [varchar(20)] - nazwa typu

```
CREATE TABLE Meetings_Types (
    TypeID int  NOT NULL,
    TypeName varchar(20)  NOT NULL,
    CONSTRAINT TypeID PRIMARY KEY  (TypeID)
);
```

Tabela **In-person_Meetings**

Zawiera dodatkowe informacje dla spotkach stacjonarnych:

- **MeetingID** [int] - klucz główny, identyfikator spotkania
- **Classroom** [int] - numer sali spotkania
- **TranslatorID** [int, nullable] - identyfikator tłumacza
- **LanguageID** [int] - klucz obcy, identyfikator języka w jakim odbywa się spotkanie
- **Limit** [int] - limit miejsc na spotkaniu
    - warunki: Limit > 0
    - wartość domyślna: 25

```
CREATE TABLE In_person_Meetings (
    MeetingID int  NOT NULL,
    Classroom int  NOT NULL,
    TranslatorID int  NULL,
    LanguageID int  NOT NULL,
    Limit int  NOT NULL DEFAULT 25 CHECK (Limit > 0),
    CONSTRAINT In_person_Meetings_pk PRIMARY KEY  (MeetingID)
);
```

Tabela **Online_Sync_Meetings**

Zaweira dodatkowe informacje dla spotkań online synchronicznie:

- **MeetingID** [int] - klucz główny, identyfikator spotkania

- **MeetingLink** [nvarchar(100)] - link do spotkania
- **RecodringLink** [nvarchar(100)] - link do nagrania spotkania
- **TranslatorID** [int, nullable] - identyfikator tłumacza
- **LanguageID** [int] - identyfiaktor języka w jakim odbywa się spotkanie

```
CREATE TABLE Online_Sync_Meetings (
   MeetingID int  NOT NULL,
   MeetingLink nvarchar(100)  NOT NULL,
   RecordingLink nvarchar(100)  NOT NULL,
   TranslatorID int  NULL,
   LanguageID int  NOT NULL,
   CONSTRAINT Online_Sync_Meetings_pk PRIMARY KEY  (MeetingID)
);
```

Tabela **Online_Async_Meetings**

Zawiera dodatkowe informacje dla spotkań online asynchronicznie:

- **MeetingID** [int] - klucz główny, identyfikator spotkania
- **RecordingLink** [nvarchar(100)] - link do nagrania

```
CREATE TABLE Online_Async_Meetings (
   MeetingID int  NOT NULL,
   RecordingLink nvarchar(100)  NOT NULL,
   CONSTRAINT Online_Async_Meetings_pk PRIMARY KEY  (MeetingID)
);
```

Tabela **User_Meetings_Attendance**

Zawiera informacje o obecności studenta na spotkaniu:

- **UserID** [int] - część klucz głównego, identyfikator studenta
- **MeetingID** [int] - część klucza głównego, identyfikator spotkania
- **SubjectID** [int] - część klucza głównego, identyfikator przedmiotu
- **Present** [bit, nullable] - informacja o obecności studenta na spotkaniu

```
CREATE TABLE Users_Meetings_Attendance (
   UserID int  NOT NULL,
   MeetingID int  NOT NULL,
   SubjectID int  NOT NULL,
   Present bit  NULL,
   CONSTRAINT Users_Meetings_Attendance_pk PRIMARY KEY
(MeetingID,UserID,SubjectID)
);
```

Tabela **Practices**

Zawiera informacje o firmach gdzie mogą być odbywane praktyki:

- **PracticeID** [int] - klucz główny, identyfikator praktyki
- **Description** [ntext] - opis firmy gdzie odbywają się praktyki
- **CompanyName** [nvarchar(30)] - nazwa firmy
- **Country** [nvarchar(30)] - kraj gdzie jest zarejestrowana firma
- **City** [nvarchar(30)] - miasto siedziby firmy
- **Address** [nvarchar(30)] - adres firmy
- **Phone** [varchar(15)] - numer telefonu do firmy
  - warunki: LEN(Phone) = 15 AND ISNUMERIC(Phone) = 1
- **Email** [nvarchar(50)] - email firmy
  - warunki: Email LIKE '%_@%.%'

```
CREATE TABLE Practices (
   PracticeID int  NOT NULL,
   Description ntext  NOT NULL,
   CompanyName nvarchar(30)  NOT NULL,
   Country nvarchar(30)  NOT NULL,
   City nvarchar(30)  NOT NULL,
   Address nvarchar(30)  NOT NULL,
   Phone varchar(15)  NOT NULL CHECK (LEN(Phone) = 15 AND ISNUMERIC(Phone) = 1),
   Email nvarchar(50)  NOT NULL CHECK (Email LIKE '%_@%.%'),
   CONSTRAINT Phone UNIQUE (Phone),
   CONSTRAINT Email UNIQUE (Email),
   CONSTRAINT Practices_pk PRIMARY KEY  (PracticeID)
);
```

Tabela **Users_Practices_Attendance**

Zawiera informacje o zdaniu praktyk przez danego studenta:

- **UserID** [int] - część klucz głównego, identyfikator studenta
- **StudiesID** [int] - część klucza głównego, identyfikator studiów
- **PracticeID** [int] - część klucza głównego, identyfikator praktyk
- **Present** [bit, nullable] - informacja o zdaniu praktyk

```
CREATE TABLE Users_Practices_Attendance (
   UserID int  NOT NULL,
   StudiesID int  NOT NULL,
   PracticeID int  NOT NULL,
   Present bit  NULL,
   CONSTRAINT Users_Practices_Attendance_pk PRIMARY KEY
```

```
(UserID,StudiesID,PracticeID)
);
```

Tabela **Users_Studies**

Zawiera informacje studentach przypisanych do danych studiów:

- **UserID** [int] - część klucza głównego, identyfikator studenta
- **StudiesID** [int] - część klucza głównego, identyfikator studiów
- **Grade** [int] - wartość oceny studenta na koniec studiów
    - warunki: Grade >=2 AND Grade <= 5

```
CREATE TABLE Users_Studies (
   UserID int  NOT NULL,
   StudiesID int  NOT NULL,
   Grade int  NULL CHECK (Grade >=2 AND Grade <= 5),
   CONSTRAINT Users_Studies_pk PRIMARY KEY  (UserID,StudiesID)
);
```

Kategoria Webinars

Tabela **Users_Webinars**

Zawiera informacje o tym, na jakie Webinary jest zapisany dany użytkownik:

- UserID [int] - identyfikator użytkownika
- WebinarID [int] - identyfikator webinaru

```
CREATE TABLE Users_Webinars (
   UserID int  NOT NULL,
   WebinarID int  NOT NULL,
   CONSTRAINT Users_Webinars_pk PRIMARY KEY  (UserID,WebinarID)
);
```

Tabela **Webinars**

Zawiera informacje o Webinarach:

- WebinarID [int] - identyfikator webinaru
- Name [nvarchar(30)] - nazwa webinaru

- Description [ntext] - opis webinaru
- Date [date] - termin odbywania się webinaru w formacie 'rok-miesiąc-dzień'
- BeginningTime [time(8)] - czas rozpoczęcia webinaru w formacie 'godzina:minuty:sekundy'
- Duration [time(8)] - czas trwania webinaru w formacie 'godziny:minuty:sekundy' (domyślny 1g 30min, musi być większy od 0)
- TeacherID [int] - identyfikator prowadzącego dany webinar
- TranslatorID [int] - identyfikator tłumacza
- Price [money] - cena webinaru (domyślna 0, musi być nieujemna)
- LanguageID [int] - identyfikator języka, w którym prowadzony jest webinar
- RecordingLink [nvarchar(100)] - link do nagrania z webinaru
- MeetingLink [nvarchar(100)] - link do webinaru

```
CREATE TABLE Webinars (
   WebinarID int  NOT NULL,
   Name nvarchar(30)  NOT NULL,
   Description ntext  NOT NULL,
   Date date  NOT NULL,
   BeginningTime time(8)  NOT NULL,
   Duration time(8)  NOT NULL DEFAULT '01:30:00' CHECK (Duration > '00:00:00'),
   TeacherID int  NOT NULL,
   TranslatorID int  NULL,
   Price money  NOT NULL DEFAULT 0 CHECK (Price >= 0),
   LanguageID int  NOT NULL,
   RecordingLink nvarchar(100)  NOT NULL,
   MeetingLink nvarchar(100)  NOT NULL,
   CONSTRAINT Webinars_pk PRIMARY KEY  (WebinarID)
);
```

Kategoria Orders

Tabela **Categories**

Zawiera informacje dotyczące kategorii możliwych do zamówienia usług (produktów):

- **CategoryID** [int] - klucz główny, identyfikator kategorii
- **Name** [nvarchar(15)] - nazwa kategorii:
  - 'Course' - kurs
  - 'Meeting' - spotkanie (zjazd)
  - 'Studies' - studia
  - 'Subject' - przedmiot (zajęcia prowadzone w ramach pewnych studiów)
  - 'Webinar' - webinar

```
CREATE TABLE Categories (
   CategoryID int  NOT NULL,
```

```
    Name nvarchar(15)  NOT NULL CHECK (Name IN ('Course', 'Meeting', 'Studies',
'Subject', 'Webinar')),
    CONSTRAINT Categories_pk PRIMARY KEY  (CategoryID)
);
```

Tabela **Orders**

Zawiera informacje dotyczące zamówień złożonych przez użytkowników:

- **OrderID** [int] - klucz główny, identyfikator zamówienia
- **UserID** [int] - identyfikator użytkownika składającego zamówienie
- **OrderDate** [date] - data złożenia zamówienia w formacie 'rok-miesiąc-dzień'
- **PaymentLink** [nvarchar(100)] - link do płatności

```
CREATE TABLE Orders (
    OrderID int  NOT NULL,
    UserID int  NOT NULL,
    OrderDate date  NOT NULL,
    PaymentLink nvarchar(100)  NOT NULL,
    CONSTRAINT Orders_pk PRIMARY KEY  (OrderID)
);
```

Tabela **Orders_Details**

Zawiera informacje szczegółowe dotyczące danego zamówienia oraz jego zamówień składowych:

- **SubOrderID** [int] - klucz główny, identyfikator zamówienia składowego
- **OrderID** [int] - identyfikator zamówienia
- PaymentDeadline [date] - termin, do którego trzeba dokonać **płatności** w formacie 'rok-miesiąc-dzień'
- **ExtendedPaymentDeadline** [date] - odroczony termin, do którego trzeba dokonać płatności w formacie 'rok-miesiąc-dzień' (jeśli jest podany, to musi być późniejszy od poprzedniego terminu płatności)
- **PaymentDate** [date] - data dokonania płatności za dane zamówienie składowe w formacie 'rok-miesiąc-dzień'
- **Price** [money] - wpłacona kwota (musi być dodatnia)
- **ProductID** [int] - identyfikator zamawianego produktu

```
CREATE TABLE Orders_Details (
    SubOrderID int  NOT NULL,
    OrderID int  NOT NULL,
    PaymentDeadline date  NOT NULL,
    ExtendedPaymentDeadline date  NULL CHECK (ExtendedPaymentDeadline >
PaymentDeadline),
    PaymentDate date  NULL,
    Price money  NOT NULL CHECK (Price > 0),
```

```
    ProductID int  NOT NULL,
    CONSTRAINT Orders_Details_pk PRIMARY KEY  (SubOrderID)
);
```

Tabela **Products**

Zawiera informacje o dostępnych produktach (usługach):

- **ProductID** [int] - klucz główny, identyfikator produktu
- **CategoryID** [int] - identyfikator kategorii produktu

```
CREATE TABLE Products (
    ProductID int  NOT NULL,
    CategoryID int  NOT NULL,
    CONSTRAINT Products_pk PRIMARY KEY  (ProductID)
);
```

Zależności między tabelami

```
-- foreign keys
-- Reference: Courses_Users_Courses (table: Users_Courses)
ALTER TABLE Users_Courses ADD CONSTRAINT Courses_Users_Courses
    FOREIGN KEY (CourseID)
    REFERENCES Courses (CourseID);

-- Reference: Employees_Courses (table: Courses)
ALTER TABLE Courses ADD CONSTRAINT Employees_Courses
    FOREIGN KEY (CoordinatorID)
    REFERENCES Employees (EmployeeID);

-- Reference: Employees_Modules (table: Modules)
ALTER TABLE Modules ADD CONSTRAINT Employees_Modules
    FOREIGN KEY (TeacherID)
    REFERENCES Employees (EmployeeID);

-- Reference: Employees_Postions_Employees (table: Employees)
ALTER TABLE Employees ADD CONSTRAINT Employees_Postions_Employees
    FOREIGN KEY (PositionID)
    REFERENCES Employees_Postions (PositionID);

-- Reference: In_person_Meetings_Languages (table: In_person_Meetings)
ALTER TABLE In_person_Meetings ADD CONSTRAINT In_person_Meetings_Languages
    FOREIGN KEY (LanguageID)
    REFERENCES Languages (LanguageID);
```

```sql
-- Reference: In_person_Meetings_Translators (table: In_person_Meetings)
ALTER TABLE In_person_Meetings ADD CONSTRAINT In_person_Meetings_Translators
    FOREIGN KEY (TranslatorID)
    REFERENCES Translators (TranslatorID);

-- Reference: In_person_Modules_Languages (table: In_person_Modules)
ALTER TABLE In_person_Modules ADD CONSTRAINT In_person_Modules_Languages
    FOREIGN KEY (LanguageID)
    REFERENCES Languages (LanguageID);

-- Reference: Meetings_Employees (table: Meetings)
ALTER TABLE Meetings ADD CONSTRAINT Meetings_Employees
    FOREIGN KEY (TeacherID)
    REFERENCES Employees (EmployeeID);

-- Reference: Meetings_In_person_Meetings (table: In_person_Meetings)
ALTER TABLE In_person_Meetings ADD CONSTRAINT Meetings_In_person_Meetings
    FOREIGN KEY (MeetingID)
    REFERENCES Meetings (MeetingID);

-- Reference: Meetings_Online_Async_Meetings (table: Online_Async_Meetings)
ALTER TABLE Online_Async_Meetings ADD CONSTRAINT Meetings_Online_Async_Meetings
    FOREIGN KEY (MeetingID)
    REFERENCES Meetings (MeetingID);

-- Reference: Meetings_Online_Sync_Meetings (table: Online_Sync_Meetings)
ALTER TABLE Online_Sync_Meetings ADD CONSTRAINT Meetings_Online_Sync_Meetings
    FOREIGN KEY (MeetingID)
    REFERENCES Meetings (MeetingID);

-- Reference: Meetings_Subjects (table: Meetings)
ALTER TABLE Meetings ADD CONSTRAINT Meetings_Subjects
    FOREIGN KEY (SubjectID)
    REFERENCES Subjects (SubjectID);

-- Reference: Meetings_Types_Meetings (table: Meetings)
ALTER TABLE Meetings ADD CONSTRAINT Meetings_Types_Meetings
    FOREIGN KEY (TypeID)
    REFERENCES Meetings_Types (TypeID);

-- Reference: Meetings_Users_Meetings_Attendance (table:
Users_Meetings_Attendance)
ALTER TABLE Users_Meetings_Attendance ADD CONSTRAINT
Meetings_Users_Meetings_Attendance
    FOREIGN KEY (MeetingID)
    REFERENCES Meetings (MeetingID);

-- Reference: Modules_Courses (table: Modules)
ALTER TABLE Modules ADD CONSTRAINT Modules_Courses
    FOREIGN KEY (CourseID)
    REFERENCES Courses (CourseID);

-- Reference: Modules_In_person_Modules (table: In_person_Modules)
```

```sql
ALTER TABLE In_person_Modules ADD CONSTRAINT Modules_In_person_Modules
    FOREIGN KEY (ModuleID)
    REFERENCES Modules (ModuleID);

-- Reference: Modules_Online_Async_Modules (table: Online_Async_Modules)
ALTER TABLE Online_Async_Modules ADD CONSTRAINT Modules_Online_Async_Modules
    FOREIGN KEY (ModuleID)
    REFERENCES Modules (ModuleID);

-- Reference: Modules_Online_Sync_Modules (table: Online_Sync_Modules)
ALTER TABLE Online_Sync_Modules ADD CONSTRAINT Modules_Online_Sync_Modules
    FOREIGN KEY (ModuleID)
    REFERENCES Modules (ModuleID);

-- Reference: Modules_Types_Modules (table: Modules)
ALTER TABLE Modules ADD CONSTRAINT Modules_Types_Modules
    FOREIGN KEY (TypeID)
    REFERENCES Modules_Types (TypeID);

-- Reference: Modules_Users_Modules_Passes (table: Users_Modules_Passes)
ALTER TABLE Users_Modules_Passes ADD CONSTRAINT Modules_Users_Modules_Passes
    FOREIGN KEY (ModuleID)
    REFERENCES Modules (ModuleID);

-- Reference: Online_Sync_Meetings_Languages (table: Online_Sync_Meetings)
ALTER TABLE Online_Sync_Meetings ADD CONSTRAINT Online_Sync_Meetings_Languages
    FOREIGN KEY (LanguageID)
    REFERENCES Languages (LanguageID);

-- Reference: Online_Sync_Meetings_Translators (table: Online_Sync_Meetings)
ALTER TABLE Online_Sync_Meetings ADD CONSTRAINT Online_Sync_Meetings_Translators
    FOREIGN KEY (TranslatorID)
    REFERENCES Translators (TranslatorID);

-- Reference: Online_Sync_Modules_Languages (table: Online_Sync_Modules)
ALTER TABLE Online_Sync_Modules ADD CONSTRAINT Online_Sync_Modules_Languages
    FOREIGN KEY (LanguageID)
    REFERENCES Languages (LanguageID);

-- Reference: Order_Details_Orders (table: Orders_Details)
ALTER TABLE Orders_Details ADD CONSTRAINT Order_Details_Orders
    FOREIGN KEY (OrderID)
    REFERENCES Orders (OrderID);

-- Reference: Order_Details_Products (table: Orders_Details)
ALTER TABLE Orders_Details ADD CONSTRAINT Order_Details_Products
    FOREIGN KEY (ProductID)
    REFERENCES Products (ProductID);

-- Reference: Orders_Users (table: Orders)
ALTER TABLE Orders ADD CONSTRAINT Orders_Users
    FOREIGN KEY (UserID)
    REFERENCES Users (UserID);
```

```sql
-- Reference: Products_Category (table: Products)
ALTER TABLE Products ADD CONSTRAINT Products_Category
    FOREIGN KEY (CategoryID)
    REFERENCES Categories (CategoryID);

-- Reference: Products_Courses (table: Products)
ALTER TABLE Products ADD CONSTRAINT Products_Courses
    FOREIGN KEY (ProductID)
    REFERENCES Courses (CourseID);

-- Reference: Products_Meetings (table: Products)
ALTER TABLE Products ADD CONSTRAINT Products_Meetings
    FOREIGN KEY (ProductID)
    REFERENCES Meetings (MeetingID);

-- Reference: Products_Studies (table: Products)
ALTER TABLE Products ADD CONSTRAINT Products_Studies
    FOREIGN KEY (ProductID)
    REFERENCES Studies (StudiesID);

-- Reference: Products_Subjects (table: Products)
ALTER TABLE Products ADD CONSTRAINT Products_Subjects
    FOREIGN KEY (ProductID)
    REFERENCES Subjects (SubjectID);

-- Reference: Products_Webinars (table: Products)
ALTER TABLE Products ADD CONSTRAINT Products_Webinars
    FOREIGN KEY (ProductID)
    REFERENCES Webinars (WebinarID);

-- Reference: Studies_Employees (table: Studies)
ALTER TABLE Studies ADD CONSTRAINT Studies_Employees
    FOREIGN KEY (CoordinatorID)
    REFERENCES Employees (EmployeeID);

-- Reference: Studies_Users_Studies (table: Users_Studies)
ALTER TABLE Users_Studies ADD CONSTRAINT Studies_Users_Studies
    FOREIGN KEY (StudiesID)
    REFERENCES Studies (StudiesID);

-- Reference: Subjects_Studies (table: Subjects)
ALTER TABLE Subjects ADD CONSTRAINT Subjects_Studies
    FOREIGN KEY (StudiesID)
    REFERENCES Studies (StudiesID);

-- Reference: Subjects_Users_Meetings_Attendance (table:
Users_Meetings_Attendance)
ALTER TABLE Users_Meetings_Attendance ADD CONSTRAINT
Subjects_Users_Meetings_Attendance
    FOREIGN KEY (SubjectID)
    REFERENCES Subjects (SubjectID);

-- Reference: Translators_In_person_Modules (table: In_person_Modules)
ALTER TABLE In_person_Modules ADD CONSTRAINT Translators_In_person_Modules
```

```
    FOREIGN KEY (TranslatorID)
    REFERENCES Translators (TranslatorID);

-- Reference: Translators_Languages_Languages (table: Translators_Languages)
ALTER TABLE Translators_Languages ADD CONSTRAINT Translators_Languages_Languages
    FOREIGN KEY (LanguageID)
    REFERENCES Languages (LanguageID);

-- Reference: Translators_Online_Sync_Modules (table: Online_Sync_Modules)
ALTER TABLE Online_Sync_Modules ADD CONSTRAINT Translators_Online_Sync_Modules
    FOREIGN KEY (TranslatorID)
    REFERENCES Translators (TranslatorID);

-- Reference: Translators_Translators_Language (table: Translators_Languages)
ALTER TABLE Translators_Languages ADD CONSTRAINT Translators_Translators_Language
    FOREIGN KEY (TranslatorID)
    REFERENCES Translators (TranslatorID);

-- Reference: Users_Courses_Users (table: Users_Courses)
ALTER TABLE Users_Courses ADD CONSTRAINT Users_Courses_Users
    FOREIGN KEY (UserID)
    REFERENCES Users (UserID);

-- Reference: Users_Meetings_Attendance_Users (table: Users_Meetings_Attendance)
ALTER TABLE Users_Meetings_Attendance ADD CONSTRAINT
Users_Meetings_Attendance_Users
    FOREIGN KEY (UserID)
    REFERENCES Users (UserID);

-- Reference: Users_Modules_Passes_Users_Courses (table: Users_Modules_Passes)
ALTER TABLE Users_Modules_Passes ADD CONSTRAINT Users_Modules_Passes_Users_Courses
    FOREIGN KEY (UserID,CourseID)
    REFERENCES Users_Courses (UserID,CourseID);

-- Reference: Users_Practices_Attendance_Practices (table:
Users_Practices_Attendance)
ALTER TABLE Users_Practices_Attendance ADD CONSTRAINT
Users_Practices_Attendance_Practices
    FOREIGN KEY (PracticeID)
    REFERENCES Practices (PracticeID);

-- Reference: Users_Practices_Attendance_Users_Studies (table:
Users_Practices_Attendance)
ALTER TABLE Users_Practices_Attendance ADD CONSTRAINT
Users_Practices_Attendance_Users_Studies
    FOREIGN KEY (UserID,StudiesID)
    REFERENCES Users_Studies (UserID,StudiesID);

-- Reference: Users_Studies_Users (table: Users_Studies)
ALTER TABLE Users_Studies ADD CONSTRAINT Users_Studies_Users
    FOREIGN KEY (UserID)
    REFERENCES Users (UserID);

-- Reference: Users_Webinars_Users (table: Users_Webinars)
```

```sql
ALTER TABLE Users_Webinars ADD CONSTRAINT Users_Webinars_Users
    FOREIGN KEY (UserID)
    REFERENCES Users (UserID);

-- Reference: Webinars_Employees (table: Webinars)
ALTER TABLE Webinars ADD CONSTRAINT Webinars_Employees
    FOREIGN KEY (TeacherID)
    REFERENCES Employees (EmployeeID);

-- Reference: Webinars_Languages (table: Webinars)
ALTER TABLE Webinars ADD CONSTRAINT Webinars_Languages
    FOREIGN KEY (LanguageID)
    REFERENCES Languages (LanguageID);

-- Reference: Webinars_Translators (table: Webinars)
ALTER TABLE Webinars ADD CONSTRAINT Webinars_Translators
    FOREIGN KEY (TranslatorID)
    REFERENCES Translators (TranslatorID);

-- Reference: Webinars_Users_Webinars (table: Users_Webinars)
ALTER TABLE Users_Webinars ADD CONSTRAINT Webinars_Users_Webinars
    FOREIGN KEY (WebinarID)
    REFERENCES Webinars (WebinarID);
```