Systemy Baz Danych 2023/24

Sebastian Kasiński, Jakub Karczewski, Rafał Chrzanowski

[**Funkcje i użytkownicy systemu 3**](#_qoa7y9woa25a)

[**Użytkownicy bazy danych 3**](#_hgb3zsv36dbw)

[**Funkcje systemu 3**](#_nqi25n8bldln)

[Konta i uprawnienia 3](#_5xanaznjepe3)

[Koszyk 3](#_ui26y9g627w1)

[Webinary 3](#_hkg5efxifrpl)

[Kursy 3](#_7mga13xlmase)

[Studia 4](#_ufmzhiu8ii4g)

[Raporty 4](#_55wyranqd5ea)

[Pozostałe funkcje 5](#_nadxm3hquu4t)

[**Schemat bazy danych 6**](#_t701rvhjf28y)

[**Opis tabel 7**](#_2k520jbf5bcu)

[Użytkownicy 7](#_367c7tc6abu3)

[Users 7](#_eo8fxtqx7w15)

[Students 7](#_7f61aowtlrb)

[Coordinators 8](#_1lp1t0tqwskw)

[Tutors 8](#_btjwuyfbudnw)

[Translators 8](#_clp2wwqspmpv)

[Countries 9](#_5odpa1k2xx4u)

[Cities 9](#_tqz04favuel)

[Addresses 10](#_rhobj2rjft2g)

[Aktywności 10](#_2e6g3iaff2wr)

[Activities 10](#_3ufjy3fhk4t5)

[Spotkania 11](#_olyect3awxf)

[Meetings 11](#_6bpekr4ve2kv)

[Rooms 11](#_mh08jq6e4ah6)

[On-Site Meetings 11](#_vsd7ullud0o3)

[Online Platforms 12](#_4abtjy8r34pq)

[Online Synchronous Meetings 12](#_1gy7ow5cn60x)

[Online Asynchronous Meetings 13](#_h93qoc9mnndk)

[Internships 13](#_jb4mh53lxd3i)

[Meeting Schedule 14](#_bahcfn9i4o09)

[Meeting Presence 14](#_ijmji64pl7ee)

[Meeting Presence Make Up 14](#_ehslewv7dbda)

[Kursy 15](#_9p6klrfmdzxz)

[Courses 15](#_n659yrym5cu)

[Course Modules 15](#_6psike5efx5t)

[Course Meetings 16](#_sn4r024fgis6)

[Studia 16](#_9zqjr4tc3hej)

[Studies 16](#_9htp0sh4veb)

[Study Sessions 16](#_x3lag56fmqes)

[Study Modules 17](#_69jvv2b7hdak)

[Study Meetings 17](#_jal9fz2pwagw)

[Study Module Passes 18](#_fzxk5ipyenku)

[Tłumaczenia 18](#_s4vakr5o6o29)

[Languages 18](#_setf88ld5k8t)

[Translators Languages 19](#_bdra3bpcxxwv)

[Meeting Translations 19](#_7syz1sav4zs7)

[Orders 20](#_ehlhfdpb2mqr)

[Products 20](#_kdndpz20lwh)

[Shopping Cart 20](#_4ekjvxa8nbg4)

[Orders 21](#_m75mnb8dtaza)

[Order Statuses 21](#_euui4jxze8lf)

[Order Details 21](#_vmf6rz7qiow9)

[Advance Payments 22](#_9vzk1ubafqm0)

# Funkcje i użytkownicy systemu

## Użytkownicy bazy danych

* Dyrektor
* Administrator
* Pracownik sekretariatu
* Koordynator
* Prowadzący
* Tłumacz
* Uczestnik
* Gość

## Funkcje systemu

### Konta i uprawnienia

* Założenie konta uczestnika - Gość
* Usunięcie konta uczestnika - Uczestnik
* Ustawienie adresu korespondencyjnego - Uczestnik
* Założenie i dezaktywacja konta tłumacza - Pracownik sekretariatu
* Założenie i dezaktywacja konta prowadzącego - Pracownik sekretariatu
* Założenie i dezaktywacja konta koordynatora - Pracownik sekretariatu
* Dodawanie i usuwanie pracownika sekretariatu - Dyrektor
* Dodawanie i usuwanie administratora - Dyrektor

### Koszyk

* Dodawanie produktów do koszyka - Uczestnik
* Usuwanie produktów z koszyka - Uczestnik
* Wyliczanie wartości koszyka - System
* Tworzenie zamówienia - Uczestnik
* Generowanie linku do płatności - System za pomocą systemu płatności
* Rejestracja udanej opłaty - System za pomocą systemu płatności

### Webinary

* Dodawanie webinaru - Pracownik sekretariatu
* Przeglądanie listy webinarów - Gość
* Dostęp do szczegółowych danych webinaru - Uczestnik po zapisie
* Modyfikacja webinaru - Prowadzący
* Usuwanie webinaru - Administrator
* Zapis uczestnika na webinar - System po udanym opłaceniu całości kwoty
* Przypisanie tłumacza do webinaru - Prowadzący
* Umożliwienie dostępu bez opłaty - Dyrektor
* Dodawanie tłumaczenia - Tłumacz

### Kursy

* Dodawanie kursu - Pracownik sekretariatu
* Przeglądanie listy kursów oraz jego modułów - Gość
* Dostęp do szczegółowych danych kursu - Uczestnik po zapisie
* Modyfikacja kursu - Koordynator
* Modyfikowanie harmonogramu - Pracownik sekretariatu
* Dodawanie, modyfikacja i usuwanie modułów kursu - Koordynator
* Usuwanie kursu - Administrator
* Zapis uczestnika na kurs - System po udanym opłaceniu zaliczki, jeśli limit miejsc nie został przekroczony
* Przypisanie tłumacza do modułu kursu - Prowadzący
* Udzielenie zapisanemu uczestnikowi dostępu do kursu - System po udanym opłaceniu całości kwoty najpóźniej trzy dni przed rozpoczęciem
* Umożliwienie dostępu bez opłaty - Dyrektor
* Ustalanie jakiego kurs jest typu - System używając informacji o typach modułów
* Dodawanie tłumaczenia - Tłumacz
* Zaliczanie kursów - System
* Wyliczanie czy na kurs są jeszcze wolne miejsca - System
* Sprawdzanie obecności - Prowadzący
* Generowanie dyplomów po zaliczeniu kursu - Pracownik sekretariatu

### Studia

* Dodawanie studiów - Pracownik sekretariatu
* Przeglądanie listy studiów i jego modułów - Gość
* Dostęp do szczegółowych danych studiów - Uczestnik po zapisie
* Modyfikacja studiów - Pracownik sekretariatu
* Modyfikowanie harmonogramu - Pracownik sekretariatu
* Dodawanie przedmiotów - Pracownik sekretariatu
* Dodawanie praktyk - Pracownik sekretariatu
* Zapis uczestnika na studia - System po udanym opłaceniu wpisowego
* Zapis uczestnika na zjazd - System po udanym opłaceniu kwoty najpóźniej trzy dni przed rozpoczęciem
* Przypisanie tłumacza do zajęć prowadzonych w ramach studiów - Prowadzący
* Umożliwienie dostępu bez opłaty - Dyrektor
* Wystawianie zaliczeń - Koordynator
* Sprawdzanie obecności - Prowadzący
* Wyliczanie maksymalnej ilości osób mogących wziąć udział w całym cyklu studiów - System
* Zapisywanie na pojedyncze spotkanie uczestnika nie będącego częścią studium - System po udanej opłacie przez uczestnika
* Dodawanie tłumaczenia - Tłumacz
* Wskazanie uczestnikowi wydarzenia, którym może odrobić nieobecności - Prowadzący
* Wyliczanie czy na studiach są jeszcze wolne miejsca - System
* Generowanie dyplomów po zaliczeniu studiów - Pracownik sekretariatu

### Raporty

* Zestawienie przychodów dla każdego webinaru, kursu i studiów - Pracownik sekretariat
* Lista osób, które skorzystały z usług, ale nie opłaciły całej kwoty - Pracownik sekretariatu
* Lista ilości osób zapisanych na przyszłe wydarzenia z informacją jakiego typu to wydarzenie i czy jest stacjonarnie czy zdalnie - Pracownik sekretariatu
* Informacja dotycząca frekwencji na zakończonych wydarzeniach - Pracownik sekretariatu
* Lista obecności dla każdego wydarzenia z datą, imieniem, nazwiskiem i informacją o obecności - Pracownik sekretariatu
* Lista osób, które są zapisane na przyszłe wydarzenia, które ze sobą kolidują - Pracownik sekretariatu
* Lista stałych klientów - Pracownik sekretariatu
* Wykaz wydarzeń na które uczestnik jest zapisany - Uczestnik
* Wykaz prowadzonych wydarzeń - Prowadzący
* Wykaz tłumaczonych wydarzeń - Tłumacz

### Pozostałe funkcje

* Wykonywanie automatycznych backupów - System
* Wykonywanie backupów na żądanie - Administrator
* Odtwarzanie bazy danych z backupów - Administrator
* Wysyłanie przypomnień o wymaganej zapłacie drugiej raty do wykupionych produktów - System

# Schemat bazy danych

# 

## Opis tabel

### Użytkownicy

##### Users

Tabela *users* przechowuje dane dotyczące każdego użytkownika zapisanego w bazie, czyli jego ID w kolumnie *id*, adres e-mail w kolumnie *email*, hasło jako wynik funkcji skrótu SHA256 w kolumnie *password*, imię w kolumnie *first\_name*, nazwisko w kolumnie *last\_name* oraz opcjonalnie telefon w kolumnie *phone*. Kluczem głównym tej tabeli jest kolumna *id*.

Warunki integralności:

* Adres e-mail musi być unikalny
* Adres e-mail musi być w formacie *‘%\_@\_%.\_%’*
* Domyślna wartość *phone* = *NULL*
* Numer telefonu może być *NULL* albo składać się tylko z cyfr (funkcja *ISNUMERIC*), przy czym dopuszczalne jest podanie pierwszego znaku ‘+’

| **CREATE** **TABLE** **users** (  **id** int **NOT** NULL **IDENTITY**(1, 1),  email **nvarchar**(64) **NOT** NULL,  **password** **nvarchar**(64) **NOT** NULL,  first\_name **nvarchar**(64) **NOT** NULL,  last\_name **nvarchar**(64) **NOT** NULL,  phone **nvarchar**(16) NULL **DEFAULT** NULL,  **CONSTRAINT** unique\_email **UNIQUE** (email),  **CONSTRAINT** users\_valid\_email **CHECK** ([email] **LIKE** '%\_@\_%.\_%'),  **CONSTRAINT** users\_valid\_phone **CHECK** (  [phone] **IS** NULL **OR**  **ISNUMERIC**([phone]) = 1 **OR**  **LEFT**([phone], 1) = '+' **AND**  **ISNUMERIC**(**SUBSTRING**([phone], 2, **LEN**([phone]))) = 1  ),  **CONSTRAINT** users\_pk PRIMARY **KEY** (**id**) ); |
| --- |

##### Students

Tabela *students* w kolumnie *user\_id* przechowuje ID użytkowników, którzy są studentami lub uczęszczają na kursy lub biorą udział w webinarach. Kluczem głównym tej tabeli jest kolumna *user\_id*.

| **CREATE** **TABLE** students (  user\_id int **NOT** NULL,  **CONSTRAINT** students\_pk PRIMARY **KEY** (user\_id) );  **ALTER** **TABLE** students **ADD** **CONSTRAINT** students\_users  FOREIGN **KEY** (user\_id)  **REFERENCES** **users** (**id**); |
| --- |

##### Coordinators

Tabela *coordinators* w kolumnie *user\_id* przechowuje ID użytkowników, którzy są koordynatorami kursów czy przedmiotów na studiach. Dodatkowo w celu uniknięcia problemów z brakiem dostępności informacji na temat nieaktywnych kont koordynatorów, przechowywana jest informacja o jego aktywności. Kluczem głównej tej tabeli jest kolumna *user\_id*.

Warunki integralności:

* Domyślna wartość *active* = *1*

| **CREATE** **TABLE** coordinators (  user\_id int **NOT** NULL,  active bit **NOT** NULL **DEFAULT** 1,  **CONSTRAINT** coordinators\_pk PRIMARY **KEY** (user\_id) );  **ALTER** **TABLE** coordinators **ADD** **CONSTRAINT** coordinators\_users  FOREIGN **KEY** (user\_id)  **REFERENCES** **users** (**id**); |
| --- |

##### Tutors

Tabela *tutors* w kolumnie *user\_id* przechowuje ID użytkowników, którzy są prowadzącymi kursów, webinarów lub zajęć na studiach. Podobnie jak w tabeli *coordinators*, w kolumnie *active* przechowywana jest informacja o aktywności konta. Kluczem głównym tabeli jest kolumna *user\_id*.

Warunki integralności:

* Domyślna wartość *active* = *1*

| **CREATE** **TABLE** tutors (  user\_id int **NOT** NULL,  active bit **NOT** NULL **DEFAULT** 1,  **CONSTRAINT** tutors\_pk PRIMARY **KEY** (user\_id) );  **ALTER** **TABLE** tutors **ADD** **CONSTRAINT** tutors\_users  FOREIGN **KEY** (user\_id)  **REFERENCES** **users** (**id**); |
| --- |

##### Translators

Tabela *translators* w kolumnie *user\_id* przechowuje ID użytkowników, którzy są tłumaczami. Podobnie jak w tabeli *coordinators*, w kolumnie *active* przechowywana jest informacja o aktywności konta. Kluczem głównym tabeli jest kolumna *user\_id*.

Warunki integralności:

* Domyślna wartość *active* = *1*

| **CREATE** **TABLE** translators (  user\_id int **NOT** NULL,  active bit **NOT** NULL **DEFAULT** 1,  **CONSTRAINT** translators\_pk PRIMARY **KEY** (user\_id) );  **ALTER** **TABLE** translators **ADD** **CONSTRAINT** translators\_users  FOREIGN **KEY** (user\_id)  **REFERENCES** **users** (**id**); |
| --- |

##### Countries

Tabela *countries* jest tabelą słownikową przechowującą ID krajów w kolumnie *id* oraz ich nazwy w kolumnie *name*. Kluczem głównym tabeli jest kolumna *id*.

Warunki integralności:

* Nazwa kraju musi być unikalna

| **CREATE** **TABLE** countries (  **id** int **NOT** NULL **IDENTITY**(1, 1),  **name** **nvarchar**(128) **NOT** NULL,  **CONSTRAINT** country\_ak\_1 **UNIQUE** (**name**),  **CONSTRAINT** countries\_pk PRIMARY **KEY** (**id**) ); |
| --- |

##### Cities

Tabela *cities* jest tabelą słownikową przechowującą ID miast w kolumnie *id*, nazwę miasta w kolumnie *name* oraz ID kraju w kolumnie *country\_id*, w którym dane miasto się znajduje. Kluczem głównym tabeli jest kolumna *id*.

* Kombinacja nazwy miasta i ID kraju musi być unikalna

| **CREATE** **TABLE** cities (  **id** int **NOT** NULL **IDENTITY**(1, 1),  country\_id int **NOT** NULL,  **name** **nvarchar**(128) **NOT** NULL,  **CONSTRAINT** cities\_ak\_1 **UNIQUE** (**name**, country\_id),  **CONSTRAINT** cities\_pk PRIMARY **KEY** (**id**) );  **ALTER** **TABLE** cities **ADD** **CONSTRAINT** cities\_countries  FOREIGN **KEY** (country\_id)  **REFERENCES** countries (**id**); |
| --- |

##### Addresses

Tabela *addresses* przechowuje adres korespondencyjny studenta o ID w kolumnie *student\_id*. Ulica znajduje się w kolumnie *street*, kod pocztowy w kolumnie *zip\_code* i ID miasta w kolumnie *city\_id*. Kluczem głównym tej tabeli jest *student\_id*.

Warunki integralności:

* Kod pocztowy musi składać się tylko z cyfr (funkcja *ISNUMERIC*)

| **CREATE** **TABLE** addresses (  student\_id int **NOT** NULL,  street **nvarchar**(128) **NOT** NULL,  zip\_code **nvarchar**(8) **NOT** NULL,  city\_id int **NOT** NULL,  **CONSTRAINT** addresses\_valid\_zip\_code **CHECK** (**ISNUMERIC**([zip\_code]) = 1),  **CONSTRAINT** addresses\_pk PRIMARY **KEY** (student\_id) );  **ALTER** **TABLE** addresses **ADD** **CONSTRAINT** addresses\_cities  FOREIGN **KEY** (city\_id)  **REFERENCES** cities (**id**);  **ALTER** **TABLE** addresses **ADD** **CONSTRAINT** addresses\_students  FOREIGN **KEY** (student\_id)  **REFERENCES** students (user\_id); |
| --- |

### Aktywności

##### Activities

Tabela *activities* przechowuje dane dotyczące każdej z aktywności zapisanych w bazie, czyli jej ID w kolumnie *id*, tytuł w kolumnie *title* i opis w kolumnie *description*. Kluczem głównym tej tabeli jest *id*.

Warunki integralności:

* Tytuł musi być unikalny

| **CREATE** **TABLE** activities (  **id** int **NOT** NULL **IDENTITY**(1, 1),  title **nvarchar**(128) **NOT** NULL,  description **nvarchar**(**max**) NULL,  **CONSTRAINT** activities\_ak\_1 **UNIQUE** (title),  **CONSTRAINT** activities\_pk PRIMARY **KEY** (**id**) ); |
| --- |

#### Spotkania

##### Meetings

Tabela *meetings* w kolumnie *activity\_id* przechowuje ID aktywności, które są spotkaniami. Każde spotkanie posiada przypisanego prowadzącego poprzez pole *tutor\_id*. Kluczem głównym tej tabeli jest *activity\_id*.

| **CREATE** **TABLE** meetings (  activity\_id int **NOT** NULL,  tutor\_id int **NOT** NULL,  **CONSTRAINT** meetings\_pk PRIMARY **KEY** (activity\_id) );  **ALTER** **TABLE** meetings **ADD** **CONSTRAINT** meetings\_activities  FOREIGN **KEY** (activity\_id)  **REFERENCES** activities (**id**);  **ALTER** **TABLE** meetings **ADD** **CONSTRAINT** meetings\_tutors  FOREIGN **KEY** (tutor\_id)  **REFERENCES** tutors (user\_id); |
| --- |

##### Rooms

Tabela *rooms* jest tabelą słownikową przechowującą dane dotyczące pokoi, w których mogą odbywać się spotkania, czyli jego ID w kolumnie *id*, nazwę w kolumnie *room\_name* oraz limit miejsc w kolumnie *place\_limit*. Kluczem głównym tej tabeli jest *id*.

Warunki integralności:

* Nazwa pokoju musi być unikalna
* Limit miejsc musi być większy od zera

| **CREATE** **TABLE** rooms (  **id** int **NOT** NULL **IDENTITY**(1, 1),  room\_name **nvarchar**(64) **NOT** NULL,  place\_limit int **NOT** NULL,  **CONSTRAINT** rooms\_ak\_1 **UNIQUE** (room\_name),  **CONSTRAINT** rooms\_valid\_place\_limit **CHECK** ([place\_limit] > 0),  **CONSTRAINT** rooms\_pk PRIMARY **KEY** (**id**) ); |
| --- |

##### On-Site Meetings

Tabela *on\_site\_meetings* w kolumnie *meeting\_id* przechowuje ID spotkań, które są prowadzone stacjonarnie. Do każdego takiego spotkania w kolumnie *room\_id* jest przypisany pokój, w którym spotkanie się odbywa. Kluczem głównym tej tabeli jest *meeting\_id*.

| **CREATE** **TABLE** on\_site\_meetings (  meeting\_id int **NOT** NULL,  room\_id int **NOT** NULL,  **CONSTRAINT** on\_site\_meetings\_pk PRIMARY **KEY** (meeting\_id) );  **ALTER** **TABLE** on\_site\_meetings **ADD** **CONSTRAINT** on\_site\_meetings\_meetings  FOREIGN **KEY** (meeting\_id)  **REFERENCES** meetings (activity\_id);  **ALTER** **TABLE** on\_site\_meetings **ADD** **CONSTRAINT** rooms\_on\_site\_meetings  FOREIGN **KEY** (room\_id)  **REFERENCES** rooms (**id**); |
| --- |

##### Online Platforms

Tabela *online\_platforms* jest tabelą słownikową przechowującą dane dotyczące platform chmurowych, na których mogą być prowadzone spotkania, czyli jej ID w kolumnie *id* oraz nazwę w kolumnie *name*. Kluczem głównym tej tabeli jest *id*.

Warunki integralności:

* Nazwa platformy musi być unikalna

| **CREATE** **TABLE** online\_platforms (  **id** int **NOT** NULL **IDENTITY**(1, 1),  **name** **nvarchar**(64) **NOT** NULL,  **CONSTRAINT** online\_platforms\_ak\_1 **UNIQUE** (**name**),  **CONSTRAINT** online\_platforms\_pk PRIMARY **KEY** (**id**) ); |
| --- |

##### Online Synchronous Meetings

Tabela *online\_synchronous\_meetings* w kolumnie *meeting\_id* przechowuje ID spotkań, które są prowadzone zdalnie synchronicznie. Do każdego takiego spotkania przypisana jest platforma chmurowa poprzez kolumnę *platform\_id*. Przed rozpoczęciem spotkania w polu *meeting\_url* powinien zostać wpisany link do spotkania, a po jego zakończeniu w pole *recording\_url* powinien zostać wpisany link do jego nagrania. Kluczem głównym tej tabeli jest *meeting\_id*.

Warunki integralności:

* Link do spotkania musi być unikalny (*UNIQUE NONCLUSTERED INDEX*)
* Domyślna wartość *meeting\_url* = *NULL*
* Link do nagrania musi być unikalny (*UNIQUE NONCLUSTERED INDEX*)
* Domyślna wartość *recording\_url* = *NULL*

| **CREATE** **TABLE** online\_synchronous\_meetings (  meeting\_id int **NOT** NULL,  platform\_id int **NOT** NULL,  meeting\_url **nvarchar**(128) NULL **DEFAULT** NULL,  recording\_url **nvarchar**(128) NULL **DEFAULT** NULL,  **CONSTRAINT** online\_synchronous\_meetings\_pk PRIMARY **KEY** (meeting\_id) );  **CREATE** **UNIQUE** NONCLUSTERED **INDEX** online\_synchronous\_meetings\_idx\_1 **on** online\_synchronous\_meetings (meeting\_url **ASC**)  **WHERE** [meeting\_url] **IS** **NOT** NULL ;  **CREATE** **UNIQUE** NONCLUSTERED **INDEX** online\_synchronous\_meetings\_idx\_2 **on** online\_synchronous\_meetings (recording\_url **ASC**)  **WHERE** [recording\_url] **IS** **NOT** NULL ;  **ALTER** **TABLE** online\_synchronous\_meetings **ADD** **CONSTRAINT** online\_synchronous\_meetings\_meetings  FOREIGN **KEY** (meeting\_id)  **REFERENCES** meetings (activity\_id);  **ALTER** **TABLE** online\_synchronous\_meetings **ADD** **CONSTRAINT** online\_synchronous\_meetings\_online\_platforms  FOREIGN **KEY** (platform\_id)  **REFERENCES** online\_platforms (**id**); |
| --- |

##### Online Asynchronous Meetings

Tabela *online\_asynchronous\_meetings* w kolumnie *meeting\_id* przechowuje ID spotkań, które są prowadzone zdalnie asynchronicznie. Do każdego takiego spotkania należy przypisać nagranie poprzez pole *recording\_url*. Kluczem głównym tej tabeli jest *meeting\_id*.

| **CREATE** **TABLE** online\_asynchronous\_meetings (  meeting\_id int **NOT** NULL,  recording\_url **nvarchar**(128) **NOT** NULL,  **CONSTRAINT** online\_asynchronous\_meetings\_pk PRIMARY **KEY** (meeting\_id) );  **ALTER** **TABLE** online\_asynchronous\_meetings **ADD** **CONSTRAINT** online\_asynchronous\_meetings\_meetings  FOREIGN **KEY** (meeting\_id)  **REFERENCES** meetings (activity\_id); |
| --- |

##### Internships

Tabela *internships* w kolumnie *meeting\_id* przechowuje ID spotkań, które są praktykami. Kluczem głównym tej tabeli jest *meeting\_id*.

| **CREATE** **TABLE** internships (  meeting\_id int **NOT** NULL,  **CONSTRAINT** internships\_pk PRIMARY **KEY** (meeting\_id) );  **ALTER** **TABLE** internships **ADD** **CONSTRAINT** internships\_meetings  FOREIGN **KEY** (meeting\_id)  **REFERENCES** meetings (activity\_id); |
| --- |

##### Meeting Schedule

Tabela *meeting\_schedule* w kolumnie *meeting\_id* przechowuje ID spotkań, które wymagają umieszczenia w harmonogramie. Do każdego takiego spotkania należy przypisać datę i czas rozpoczęcia w kolumnie *start\_time* i datę i czas zakończenia w kolumnie *end\_time*. Kluczem głównym tej tabeli jest *meeting\_id*.

Warunki integralności:

* Czas rozpoczęcia musi być przed czasem zakończenia

| **CREATE** **TABLE** meeting\_schedule (  meeting\_id int **NOT** NULL,  start\_time datetime **NOT** NULL,  end\_time datetime **NOT** NULL,  **CONSTRAINT** meeting\_schedule\_valid\_time **CHECK** ([start\_time] <= [end\_time]),  **CONSTRAINT** meeting\_schedule\_pk PRIMARY **KEY** (meeting\_id) );  **ALTER** **TABLE** meeting\_schedule **ADD** **CONSTRAINT** meeting\_schedule\_meetings  FOREIGN **KEY** (meeting\_id)  **REFERENCES** meetings (activity\_id); |
| --- |

##### Meeting Presence

Tabela *meeting\_presence* przechowuje dane na temat obecności studentów na spotkaniach poprzez powiązanie ID spotkania w kolumnie *meeting\_id* z ID studenta w kolumnie *student\_id*. Kluczami głównymi tej tabeli są *meeting\_id* i *student\_id*.

| **CREATE** **TABLE** meeting\_presence (  meeting\_id int **NOT** NULL,  student\_id int **NOT** NULL,  **CONSTRAINT** meeting\_presence\_pk PRIMARY **KEY** (meeting\_id,student\_id) );  **ALTER** **TABLE** meeting\_presence **ADD** **CONSTRAINT** meeting\_presence\_meetings  FOREIGN **KEY** (meeting\_id)  **REFERENCES** meetings (activity\_id);  **ALTER** **TABLE** meeting\_presence **ADD** **CONSTRAINT** meeting\_presence\_students  FOREIGN **KEY** (student\_id)  **REFERENCES** students (user\_id); |
| --- |

##### Meeting Presence Make Up

Tabela *meeting\_presense\_make\_up* przechowuje dane na temat jakie zajęcia lub kurs został wyznaczony przez prowadzącego w celu odrobienia nieobecności na spotkaniu poprzez powiązanie ID spotkania w kolumnie *meeting\_id* i ID studenta w kolumnie *student\_id* z ID wyznaczonej aktywności w kolumnie *activity\_id*. Kluczami głównymi tej tabeli są *meeting\_id* i *student\_id*.

| **CREATE** **TABLE** meeting\_presence\_make\_up (  meeting\_id int **NOT** NULL,  student\_id int **NOT** NULL,  activity\_id int **NOT** NULL,  **CONSTRAINT** meeting\_presence\_make\_up\_pk PRIMARY **KEY** (meeting\_id,student\_id) );  **ALTER** **TABLE** meeting\_presence\_make\_up **ADD** **CONSTRAINT** meeting\_absence\_make\_up\_activities  FOREIGN **KEY** (activity\_id)  **REFERENCES** activities (**id**);  **ALTER** **TABLE** meeting\_presence\_make\_up **ADD** **CONSTRAINT** meeting\_absence\_make\_up\_meeting\_presence  FOREIGN **KEY** (meeting\_id,student\_id)  **REFERENCES** meeting\_presence (meeting\_id,student\_id); |
| --- |

#### Kursy

##### Courses

Tabela *courses* w kolumnie *activity\_id* przechowuje ID aktywności, które są kursami. Każdy kurs posiada przypisanego koordynatora w polu *coordinator\_id*. Kluczem głównym tej tabeli jest *activity\_id*.

| **CREATE** **TABLE** courses (  activity\_id int **NOT** NULL,  coordinator\_id int **NOT** NULL,  **CONSTRAINT** courses\_pk PRIMARY **KEY** (activity\_id) );  **ALTER** **TABLE** courses **ADD** **CONSTRAINT** courses\_activities  FOREIGN **KEY** (activity\_id)  **REFERENCES** activities (**id**);  **ALTER** **TABLE** courses **ADD** **CONSTRAINT** courses\_coordinators  FOREIGN **KEY** (coordinator\_id)  **REFERENCES** coordinators (user\_id); |
| --- |

##### Course Modules

Tabela *course\_modules* w kolumnie *activity\_id* przechowuje ID aktywności, które są modułami kursów. Każdy moduł kursu ma przyporządkowany kurs w kolumnie *course\_id*. Kluczem głównym tej tabeli jest *activity\_id*.

| **CREATE** **TABLE** course\_modules (  activity\_id int **NOT** NULL,  course\_id int **NOT** NULL,  **CONSTRAINT** course\_meetings\_pk PRIMARY **KEY** (activity\_id) );  **ALTER** **TABLE** course\_modules **ADD** **CONSTRAINT** course\_meetings\_activities  FOREIGN **KEY** (activity\_id)  **REFERENCES** activities (**id**);  **ALTER** **TABLE** course\_modules **ADD** **CONSTRAINT** course\_meetings\_courses  FOREIGN **KEY** (course\_id)  **REFERENCES** courses (activity\_id); |
| --- |

##### Course Meetings

Tabela *course\_meetings* przechowuje dane na temat spotkań w kolumnie *meeting\_id* powiązanych z danym modułem kursu w kolumnie *module\_id*. Kluczem głównym tej tabeli jest *meeting\_id*.

| **CREATE** **TABLE** course\_modules (  activity\_id int **NOT** NULL,  course\_id int **NOT** NULL,  **CONSTRAINT** course\_meetings\_pk PRIMARY **KEY** (activity\_id) ); |
| --- |

#### Studia

##### Studies

Tabela *studies* w kolumnie *activity\_id* przechowuje ID aktywności, które są studiami. Każde studia posiadają limit miejsc zapisany w kolumnie *place\_limit*. Kluczem głównym tej tabeli jest *activity\_id*.

Warunki integralności:

* Limit miejsc musi być większy od zera.

| **CREATE** **TABLE** studies (  activity\_id int **NOT** NULL,  place\_limit int **NOT** NULL,  **CONSTRAINT** studies\_place\_limit **CHECK** ([place\_limit] > 0),  **CONSTRAINT** studies\_pk PRIMARY **KEY** (activity\_id) );  **ALTER** **TABLE** studies **ADD** **CONSTRAINT** studies\_activities  FOREIGN **KEY** (activity\_id)  **REFERENCES** activities (**id**); |
| --- |

##### Study Sessions

Tabela *study\_sessions* w kolumnie *activity\_id* przechowuje ID aktywności, które są zjazdami. Każdy zjazd jest przyporządkowany do dokładnie jednych studiów poprzez kolumnę *study\_id*. Kluczem głównym tej tabeli jest *activity\_id*.

| **CREATE** **TABLE** study\_sessions (  activity\_id int **NOT** NULL,  study\_id int **NOT** NULL,  **CONSTRAINT** study\_sessions\_pk PRIMARY **KEY** (activity\_id) );  **ALTER** **TABLE** study\_sessions **ADD** **CONSTRAINT** study\_sessions\_activities  FOREIGN **KEY** (activity\_id)  **REFERENCES** activities (**id**);  **ALTER** **TABLE** study\_sessions **ADD** **CONSTRAINT** study\_sessions\_studies  FOREIGN **KEY** (study\_id)  **REFERENCES** studies (activity\_id); |
| --- |

##### Study Modules

Tabela *study\_modules* w kolumnie *activity\_id* przechowuje ID aktywności, które są modułami (przedmiotami) na studiach. Każdy moduł ma przyporządkowanego koordynatora poprzez kolumnę *coordinator\_id* oraz jest przyporządkowany do dokładnie jednych studiów przez kolumnę *study\_id*. Kluczem głównym tej tabeli jest *activity\_id*.

| **CREATE** **TABLE** study\_modules (  activity\_id int **NOT** NULL,  study\_id int **NOT** NULL,  coordinator\_id int **NOT** NULL,  **CONSTRAINT** study\_modules\_pk PRIMARY **KEY** (activity\_id) );  **ALTER** **TABLE** study\_modules **ADD** **CONSTRAINT** study\_modules\_activities  FOREIGN **KEY** (activity\_id)  **REFERENCES** activities (**id**);  **ALTER** **TABLE** study\_modules **ADD** **CONSTRAINT** study\_modules\_coordinators  FOREIGN **KEY** (coordinator\_id)  **REFERENCES** coordinators (user\_id);  **ALTER** **TABLE** study\_modules **ADD** **CONSTRAINT** study\_modules\_studies  FOREIGN **KEY** (study\_id)  **REFERENCES** studies (activity\_id); |
| --- |

##### Study Meetings

Tabela *study\_meetings* w kolumnie *meeting\_id* przechowuje ID spotkań, które są realizowane w ramach studiów. Każde takie spotkanie jest przyporządkowane do zjazdu przez kolumnę *session\_id* i modułu przez kolumnę *module\_id*. Kluczem głównym tej tabeli jest *meeting\_id*.

| **CREATE** **TABLE** study\_meetings (  meeting\_id int **NOT** NULL,  session\_id int **NOT** NULL,  module\_id int **NOT** NULL,  **CONSTRAINT** study\_meetings\_pk PRIMARY **KEY** (meeting\_id) );  **ALTER** **TABLE** study\_meetings **ADD** **CONSTRAINT** study\_meetings\_meetings  FOREIGN **KEY** (meeting\_id)  **REFERENCES** meetings (activity\_id);  **ALTER** **TABLE** study\_meetings **ADD** **CONSTRAINT** study\_modules\_study\_meetings  FOREIGN **KEY** (module\_id)  **REFERENCES** study\_modules (activity\_id);  **ALTER** **TABLE** study\_meetings **ADD** **CONSTRAINT** study\_sessions\_study\_meetings  FOREIGN **KEY** (session\_id)  **REFERENCES** study\_sessions (activity\_id); |
| --- |

##### Study Module Passes

Tabela *study\_module\_passes* zawiera informacje o zaliczeniach modułów studiów przez studentów. Dany student o ID w kolumnie *student\_id* zaliczył dany moduł o ID w kolumnie *module\_id* wtedy i tylko wtedy, gdy znajduje się w tej tabeli. Kluczami głównymi tej tabeli są *student\_id* i *module\_id*.

| **CREATE** **TABLE** study\_module\_passes (  student\_id int **NOT** NULL,  module\_id int **NOT** NULL,  **CONSTRAINT** study\_course\_passes\_pk PRIMARY **KEY** (student\_id,module\_id) );  **ALTER** **TABLE** study\_module\_passes **ADD** **CONSTRAINT** study\_module\_passes\_students  FOREIGN **KEY** (student\_id)  **REFERENCES** students (user\_id);  **ALTER** **TABLE** study\_module\_passes **ADD** **CONSTRAINT** study\_module\_passes\_study\_modules  FOREIGN **KEY** (module\_id)  **REFERENCES** study\_modules (activity\_id); |
| --- |

### Tłumaczenia

##### Languages

Tabela *languages* jest tabelą słownikową przechowującą dane dotyczące języków, czyli ich ID w kolumnie *id* oraz nazwę w kolumnie *name*. Kluczem głównym tej tabeli jest id.

| **CREATE** **TABLE** languages (  **id** int **NOT** NULL **IDENTITY**(1, 1),  **name** **nvarchar**(64) **NOT** NULL,  **CONSTRAINT** languages\_pk PRIMARY **KEY** (**id**) ); |
| --- |

##### Translators Languages

Tabela *translators\_languages* zawiera języki w kolumnie *language\_id*, którymi posługują się tłumacze w kolumnach translator\_id. Kluczami głównymi tej tabeli są *translator\_id* i *language\_id*.

| **CREATE** **TABLE** translators\_languages (  translator\_id int **NOT** NULL,  language\_id int **NOT** NULL,  **CONSTRAINT** translators\_languages\_pk PRIMARY **KEY** (translator\_id,language\_id) );  **ALTER** **TABLE** translators\_languages **ADD** **CONSTRAINT** translators\_languages\_languages  FOREIGN **KEY** (language\_id)  **REFERENCES** languages (**id**);  **ALTER** **TABLE** translators\_languages **ADD** **CONSTRAINT** translators\_languages\_translators  FOREIGN **KEY** (translator\_id)  **REFERENCES** translators (user\_id); |
| --- |

##### Meeting Translators

Tabela *meeting\_translators* dla spotkania o ID w kolumnie *meeting\_id* przechowuje w kolumnie *language\_id* informację w jakim języku odbywa się dane spotkanie. Informacja ta jest umieszczana tylko wtedy, gdy spotkanie nie odbywa się w języku polskim. Każde spotkanie w obcym języku ma przypisanego tłumacza w kolumnie *translator\_id*. Kluczem głównym tej tabeli jest *meeting\_id*.

| **CREATE** **TABLE** meeting\_translators (  meeting\_id int **NOT** NULL,  translator\_id int **NOT** NULL,  language\_id int **NOT** NULL,  **CONSTRAINT** translations\_pk PRIMARY **KEY** (meeting\_id) );  **ALTER** **TABLE** meeting\_translators **ADD** **CONSTRAINT** meeting\_translations\_languages  FOREIGN **KEY** (language\_id)  **REFERENCES** languages (**id**);  **ALTER** **TABLE** meeting\_translators **ADD** **CONSTRAINT** meeting\_translations\_translators  FOREIGN **KEY** (translator\_id)  **REFERENCES** translators (user\_id);  **ALTER** **TABLE** meeting\_translators **ADD** **CONSTRAINT** meetings\_meeting\_translations  FOREIGN **KEY** (meeting\_id)  **REFERENCES** meetings (activity\_id); |
| --- |

### Orders

##### Products

Tabela *products* w kolumnie *activity\_id* przechowuje ID aktywności, które są produktami, czyli są dostępne do kupienia przez studenta. Każdy produkt ma przypisaną cenę w kolumnie *price*, która domyślnie jest zerowa. Produkt można dezaktywować poprzez ustawienie wartości kolumny *active* na 0. Kluczem głównym tej tabeli jest *activity\_id*.

Warunki integralności:

* Cena nie może być ujemna
* Domyślna wartość *price* = *0*
* Domyślna wartość *active* = *1*

| **CREATE** **TABLE** products (  activity\_id int **NOT** NULL,  price money **NOT** NULL **DEFAULT** 0,  active bit **NOT** NULL **DEFAULT** 1,  **CONSTRAINT** products\_valid\_price **CHECK** ([price] >= 0),  **CONSTRAINT** products\_pk PRIMARY **KEY** (activity\_id) );  **ALTER** **TABLE** products **ADD** **CONSTRAINT** products\_activities  FOREIGN **KEY** (activity\_id)  **REFERENCES** activities (**id**); |
| --- |

##### Shopping Cart

Tabela *shopping\_cart* w kolumnie *product\_id* przechowuje produkty dodane do koszyka przez studenta o ID w kolumnie *student\_id*. Kluczami głównymi tej tabeli są *product\_id* i *student\_id*.

| **CREATE** **TABLE** shopping\_cart (  product\_id int **NOT** NULL,  student\_id int **NOT** NULL,  **CONSTRAINT** shopping\_cart\_pk PRIMARY **KEY** (product\_id,student\_id) );  **ALTER** **TABLE** shopping\_cart **ADD** **CONSTRAINT** shopping\_cart\_products  FOREIGN **KEY** (product\_id)  **REFERENCES** products (activity\_id);  **ALTER** **TABLE** shopping\_cart **ADD** **CONSTRAINT** shopping\_cart\_students  FOREIGN **KEY** (student\_id)  **REFERENCES** students (user\_id); |
| --- |

##### Orders

Tabela *orders* przechowuje zamówienia o ID w kolumnie *id* złożone przez studentów o ID w kolumnie *student\_id*. Każde zamówienie, które ma niezerową wartość, ma przypisany unikalny link do płatności w kolumnie *payment\_url*, który służy do wnoszenia wpłat za produkty. Ten sam link służy także do płatności ratalnej. Każde zamówienie posiada także datę złożenia zamówienia w kolumnie *order\_date*. Kluczem głównym tej tabeli jest *id*.

Warunki integralności:

* Link do płatności musi być unikalny
* Domyślna wartość *order\_date* = *GETDATE()*

| **CREATE** **TABLE** orders (  **id** int **NOT** NULL **IDENTITY**(1, 1),  student\_id int **NOT** NULL,  payment\_url **nvarchar**(128) NULL,  order\_date date **NOT** NULL **DEFAULT** **GETDATE**(),  **CONSTRAINT** orders\_ak\_1 **UNIQUE** (payment\_url),  **CONSTRAINT** orders\_pk PRIMARY **KEY** (**id**) );  **ALTER** **TABLE** orders **ADD** **CONSTRAINT** orders\_students  FOREIGN **KEY** (student\_id)  **REFERENCES** students (user\_id); |
| --- |

##### Order Statuses

Tabela *order\_statuses* jest tabelą słownikową przechowującą dane dotyczące statusu zamówienia produktu, czyli ID statusu w kolumnie *id* oraz nazwę statusu zamówienia w kolumnie *name*. Szczególnym przypadkiem statusu zamówienia jest odroczenie płatności za pozwoleniem dyrektora. Kluczem głównym tej tabeli jest *id*.

| **CREATE** **TABLE** order\_statuses (  **id** int **NOT** NULL **IDENTITY**(1, 1),  **name** **nvarchar**(64) **NOT** NULL,  **CONSTRAINT** order\_statuses\_pk PRIMARY **KEY** (**id**) ); |
| --- |

##### Order Details

Tabela *order\_details* w kolumnie *product\_id* przechowuje ID produktów, które zostały kupione w ramach zamówienia o ID w kolumnie *order\_id*. Każdy z tych produktów także posiada w kolumnie *price* cenę, za którą został kupiony, oraz status dotyczący jego opłacenia w kolumnie *status\_id*. Kluczami głównymi tej tabeli są *order\_id* i *product\_id*.

Warunki integralności:

* Cena nie może być ujemna

| **CREATE** **TABLE** order\_details (  order\_id int **NOT** NULL,  product\_id int **NOT** NULL,  price money **NOT** NULL,  status\_id int **NOT** NULL,  **CONSTRAINT** order\_details\_valid\_price **CHECK** ([price] >= 0),  **CONSTRAINT** order\_details\_pk PRIMARY **KEY** (order\_id,product\_id) );  **ALTER** **TABLE** order\_details **ADD** **CONSTRAINT** order\_details\_orders  FOREIGN **KEY** (order\_id)  **REFERENCES** orders (**id**);  **ALTER** **TABLE** order\_details **ADD** **CONSTRAINT** order\_details\_payment\_status  FOREIGN **KEY** (status\_id)  **REFERENCES** order\_statuses (**id**);  **ALTER** **TABLE** order\_details **ADD** **CONSTRAINT** order\_details\_products  FOREIGN **KEY** (product\_id)  **REFERENCES** products (activity\_id); |
| --- |

##### Advance Payments

Tabela *advance\_payments* zawiera informację na temat jak wysokie zaliczki należy wpłacić pomiędzy podanymi datami. W kolumnie *id* do danej zaliczki jest przypisane ID zaliczki, w kolumnie *value* znajduje się informacja jaką część całości kwoty wynosi zaliczka, w kolumnie *start\_date* znajduje się informacja od kiedy zaliczka obowiązuje, a w kolumnie *end\_date* do kiedy obowiązuje. Jeśli wartość *end\_date* wynosi *NULL*, to dana zaliczka obowiązuje bezterminowo. Kluczem głównym tej tabeli jest *id*.

Warunki integralności:

* Data początkowa musi być unikalna
* Data końcowa musi być unikalna
* Data początkowa nie może wystąpić po dacie końcowej
* Wartość zaliczki musi być z przedziału od 0 do 1
* Domyślna wartość *end\_date* = *NULL*

| **CREATE** **TABLE** advance\_payments (  **id** int **NOT** NULL **IDENTITY**(1, 1),  **value** decimal(3,3) **NOT** NULL,  start\_date date **NOT** NULL,  end\_date date NULL **DEFAULT** NULL,  **CONSTRAINT** advance\_payments\_ak\_1 **UNIQUE** (start\_date),  **CONSTRAINT** advance\_payments\_ak\_2 **UNIQUE** (end\_date),  **CONSTRAINT** advance\_payments\_valid\_dates **CHECK** ([start\_date] <= [end\_date]),  **CONSTRAINT** advance\_payments\_valid\_value **CHECK** (0 <= [**value**] **AND** [**value**] <= 1),  **CONSTRAINT** advance\_payments\_pk PRIMARY **KEY** (**id**) ); |
| --- |