**Podstawy Baz Danych**

**Projekt: System zarządzania konferencjami**

**Dokumentacja**

Mateusz Jarosz, Zbigniew Królikowski

**Opis problemu :**

Projekt dotyczy systemu wspomagania działalności firmy organizującej konferencje:

**Ogólne informacje**

Firma organizuje konferencje, które mogą być jedno- lub kilkudniowe. Klienci

powinni móc rejestrować się na konferencje za pomocą systemu www. Klientami mogą być

zarówno indywidualne osoby jak i firmy, natomiast uczestnikami konferencji są osoby (firma

nie musi podawać od razu przy rejestracji listy uczestników - może zarezerwować

odpowiednią ilość miejsc na określone dni oraz na warsztaty, natomiast na 2 tygodnie przed

rozpoczęciem musi te dane uzupełnić - a jeśli sama nie uzupełni do tego czasu, to pracownicy

dzwonią do firmy i ustalają takie informacje). Każdy uczestnik konferencji otrzymuje

identyfikator imienny (+ ew. informacja o firmie na nim). Dla konferencji kilkudniowych,

uczestnicy mogą rejestrować się na dowolne z tych dni.

**Warsztaty**

Ponadto z konferencją związane są warsztaty, na które uczestnicy także mogą się

zarejestrować - muszą być jednak zarejestrowani tego dnia na konferencję, aby móc w nich

uczestniczyć. Kilka warsztatów może trwać równocześnie, ale uczestnik nie może

zarejestrować się na więcej niż jeden warsztat, który trwa w tym samym czasie. Jest także

ograniczona ilość miejsc na każdy warsztat i na każdy dzień konferencji. Część warsztatów

może być płatna, a część jest darmowa.

**Opłaty**

Opłata za udział w konferencji zależy nie tylko od wykupionych usług, ale także od

terminu ich wykupienia - jest kilka progów ceny (progi ceny dotyczą tylko udziału w

konferencji, cena warsztatów jest stała) i im bliżej rozpoczęcia konferencji, tym cena jest

wyższa (jest także zniżka procentowa dla studentów i wtedy przy rejestracji trzeba podać nr

legitymacji studenckiej). Na zapłatę klienci mają tydzień od rejestracji na konferencję - jeśli

do tego czasu nie pojawi się opłata, rejestracja jest anulowana.

**Raporty**

Dla organizatora najbardziej istotne są listy osobowe uczestników na każdy dzień

konferencji i na każdy warsztat, a także informacje o płatnościach klientów. Ponadto

organizator chciałby mieć informację o klientach, którzy najczęściej korzystają z jego usług.

**Specyfika firmy**

Firma organizuje średnio 2 konferencje w miesiącu, każda z nich trwa zwykle 2-3 dni,

w tym średnio w każdym dniu są 4 warsztaty. Na każdą konferencję średnio rejestruje się

200 osób. Stworzona baza danych powinna zostać wypełniona w takim stopniu, aby

odpowiadała 3-letniej działalności firmy.

**Aktorzy :**

Pracownik Administrator:

- Pełny dostęp do całego systemu.

- Przesuwanie terminu uzupełnienia danych/ płatności.

Pracownik:

- Dodawanie/modyfikacja danych warsztatów, konferencji.

- Dodawanie/modyfikacja danych dni warsztatów, konferencji.

- Dodawanie/modyfikacja progów cenowych warsztatów.

- Dodawanie/modyfikcja rezerwacji konferencji, warsztatów.

- Dodawanie/modyfikacja danych klienta.

- Dodawanie/modyfikacja danych uczestników.

- Generowanie danych płatności na rzecz danego warszatu/konferencji/klienta.

- Anulowanie konferencji.

- Anulowanie warsztatów.

- Anulowanie rezerwacji w przypadku niespełnia warunków (niesprecyzowanie uczestników/niepełna płatność).

Klient (firma lub indywidualny):

- Dodawnie rezerwacji konferencji/warsztatów, na określone dni, na określone warsztaty, na określoną ilość uczestników normalnych/studentów.

- Anulowanie rezerwacji.

- Dodawanie/edycja danych uczestników zgłoszonych od niego.

- Odczyt danych dotyczących płatności.

Uczestnik:

- Dostęp do danych warsztatów/konferencji na które jest zapisany.

- Ma podgląd do swoich danych w celu weryfikacji zgodności.

**Rejestracja klienta (firmy) na konferencję**

**1** Klient specyfikuje na które dni konferencji oraz warsztaty chce się zapisać. Do każdego dnia i warsztatu podaje liczbę uczestników (normalnych i ulgowych).

--Klient nie może przekroczyć ilości wolnych miejsc na dany dzień konferencji.

--Dostępne są tylko te warsztaty, które odbywają się w wybrane dni konferencji.

--Klient nie może przekroczyć ilości wolnych miejsc na dany warsztat.

--W przypadku rezerwacji we wczesnym terminie zostaje ustalona zniżka.

**2** Po dokonaniu rezerwacji klient informowany jest o terminie płatności, kwocie (podawane są dane do przelewu/formularz bankowy) oraz uzupełnienia danych.

**3** Klient uzupełnia dane uczestników maksymalnie na 2 tygodnie przed rozpoczęciem konferencji.

**3.1** Klient nie uzupełnił danych uczestników. Pracownik odpowiedzialny za obsługę tego klienta (lub inny wyznaczony przez przełożonego) stara się skontaktować z klientem.

**3.1.1** Nie udało się skontaktować z klientem. Pracownik informuje przełożonego a ten podejmuje decyzję o unieważnieniu rezerwacji.

**3.1.1.1** Rezerwacja zostaje unieważniona. Zwracane są wszelkie płatności wykonane przez klienta.

**3.1.1.2** Termin uzupełnia danych przesuwa się o dzień. Maksymalnie do 1 tygodnia przed konferencją.

**3.1.2** Udało sie skontaktować z klientem i uzupełniono liczbę osób.

**3.2** Klient uzupełnił dane dane uczestników. Przygotowujemy identyfikatory immienne do rozesłania (lub odebrania).

**4.** Nadchodzi czas na odpłatę (1 tydzień przed konferencją).

--Podliczane są zniżki.

--Studenci którzy nie mieli uzupełnionych numerów indeksów nie dostają żniżki.

**4.1** Klient płaci w terminie całość.

**4.2** Klient płaci część:

**4.2.1** Przełożony informowany jest o sytuacji i podejmuje decyzję.

**4.2.1.1** Anuluje się rejestrację klienta. Pieniądze są zwracane z uwzględnieniem poniesionych kosztów.

**4.2.1.2** Termin odpłaty przełożony jest jest na termin określony przez przełożonego. Niekonieczne jest odpłacenie przez rozpoczęciem konferencji, osoba podejmująca decyzję bierzę na siebie odpowiedzialność.

**4.3.** Klient nie płaci nic.

**4.3.1** Przełożony informowany jest o sytuacji i podejmuje decyzję.

**4.3.1.1** Anuluje się rejestrację klient.

**4.3.1.2** Termin odpłaty przełożony jest jest na termin określony przez przełożonego. Niekonieczne jest odpłacenie przez rozpoczęciem konferencji, osoba podejmująca decyzję bierzę na siebie odpowiedzialność.

--Wszelkie rejestracje typu last minute są dokonywane prez pracowników tylko po kontakcie telefonicznym na conajmniej 6 dni przed konferencją klient ma tylko jeden dzień na wniesienie opłat i podanie danych uczestników.

**Rejestracja klienta (osoby) na konferencję**

**1** Klient specyfikuje na które dni konferencji oraz warsztaty chce się zapisać.

--Uzupełnia swoje dane jeżeli nie był nigdy rejestrowany w systemie.

--Jeżeli był studentem zaznacza czy dalej nim jest.

--Klient nie może zarejestrować się na konferencje na których nie ma miejsc.

--Dostępne są tylko te warsztaty, które odbywają się w wybrane dni konferencji.

--Klient nie może zarejestrować się na warsztaty na których nie ma miejsc.

--W przypadku rezerwacji we wczesnym terminie zostaje ustalona zniżka.

**2** Po dokonaniu rezerwacji klient informowany jest o terminie płatności oraz kwocie (podane są dane do przelewu/formularz bankowy). Drukujowany i wysyłany(jeśli potrzeba) jest identyfikator imienny.

**3** Nadchodzi czas na odpłatę (1 tydzień przed konferencją).

--Podliczane są zniżki.

**3.1** Klient płaci w terminie całość.

**3.2** Klient płaci część:

**3.2.1** Przełożony informowany jest o sytuacji i podejmuje decyzję.

**3.2.1.1** Anuluje się rejestrację klienta. Pieniądze są zwracane z uwzględnieniem poniesionych kosztów.

**3.2.1.2** Termin odpłaty przełożony jest jest na termin określony przez przełożonego. Niekonieczne jest odpłacenie przez rozpoczęciem konferencji, osoba podejmująca decyzję bierzę na siebie odpowiedzialność.

**3.3** Klient nie płaci nic.

**3.3.1** Przełożony informowany jest o sytuacji i podejmuje decyzję.

**3.3.1.1** Anuluje się rejestrację klient.

**3.3.1.2** Termin odpłaty przełożony jest jest na termin określony przez przełożonego. Niekonieczne jest odpłacenie przez rozpoczęciem konferencji, osoba podejmująca decyzję bierzę na siebie odpowiedzialność.

--Wszelkie rejestracje typu last minute są dokonywane prez pracowników tylko po kontakcie telefonicznym na conajmniej 6 dni przed konferencją klient ma tylko jeden dzień na wniesienie opłat i podanie danych uczestników.

**ODWOLANIE KONFERENCJI**: W przypadku odwołania konferencji przez organizatora, z przyczyn losowych, wszelkie wniesione opłaty zostają zwrócone klientom.

**Shemat i opis tabel :**

**(schemat na następnej stonie)**

**Tabela [dbo].[Klienci]**

Opisuje zarównno klientów indywidualnych jak i firmy. W zależności od tego jaki jest typ klient bit [CzyFirma] osiąga wartości 1 lub 0. Względem tej właśności zbudowane są procedury oraz widoki.

Klucz główny: [id]

Klucze obce: [OsobaID]

Nałożone warunki integralnościowe to:

* unikalność [id]
* unikalność [Login]
* unikalność [Email]

**CREATE** **TABLE** [dbo].[Klienci](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[OsobaID] [**int**] **NULL**,

[CzyFirma] [**bit**] **NOT** **NULL**,

[Login] [**varchar**](30) **NOT** **NULL**,

[**Password**] [**varchar**](20) **NOT** **NULL**,

[Email] [**varchar**](50) **NOT** **NULL**,

**CONSTRAINT** [PK\_Klienci] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**],

**CONSTRAINT** [UnikalnyLogin] **UNIQUE** **NONCLUSTERED**

(

[Login] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**],

**CONSTRAINT** [UnikalnyMail] **UNIQUE** **NONCLUSTERED**

(

[Email] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**ALTER** **TABLE** [dbo].[Klienci] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Klienci\_Osoby] **FOREIGN** **KEY**([OsobaID])

**REFERENCES** [dbo].[Osoby] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Klienci] **CHECK** **CONSTRAINT** [FK\_Klienci\_Osoby]

**GO**

**Tabela [dbo].[Firmy]**

Doprecyzowuje klienta jeżeli ten posiada [CzyFirma] = 1, głównie dodając dane kontaktowe.

Klucz główny: [id]

Klucze obce: [OsobaID] , [AdresID]

Nałożone warunki integralnościowe to:

* unikalność [id]

**CREATE** **TABLE** [dbo].[Firmy](

[KlientID] [**int**] **NOT** **NULL**,

[Nazwa Firmy] [**varchar**](25) **NULL**,

[Telefon Kontaktowy] [**varchar**](12) **NULL**,

[Fax] [**varchar**](12) **NULL**,

[AdresID] [**int**] **NOT** **NULL**,

**CONSTRAINT** [PK\_Firmy] **PRIMARY** **KEY** **CLUSTERED**

(

[KlientID] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**ALTER** **TABLE** [dbo].[Firmy] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Firmy\_Adresy] **FOREIGN** **KEY**([AdresID])

**REFERENCES** [dbo].[Adresy] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Firmy] **CHECK** **CONSTRAINT** [FK\_Firmy\_Adresy]

**GO**

**ALTER** **TABLE** [dbo].[Firmy] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Firmy\_Klienci] **FOREIGN** **KEY**([KlientID])

**REFERENCES** [dbo].[Klienci] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Firmy] **CHECK** **CONSTRAINT** [FK\_Firmy\_Klienci]

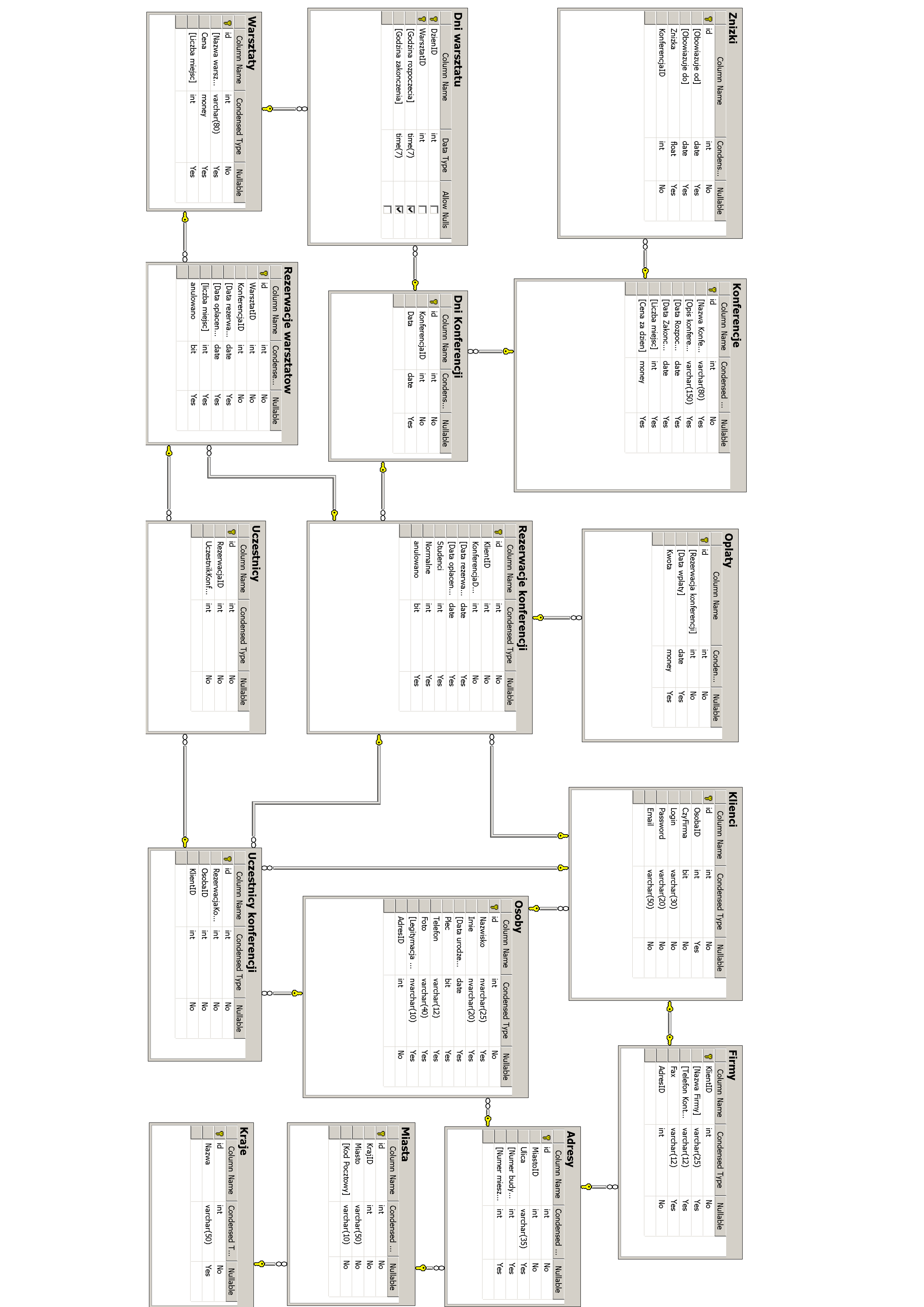
**GO**

**Tabela [dbo].[Kraje]**

Opisuje ona kraje dostępne dla naszych danych adresowych. Ważny jest warunek unikalności nazwy, ze względu na to, że bez niego mogłyby występować w bazie dwa wiersze odnoszące się do tego samego kraju.

Klucz główny: [id]

Brak kluczy obcych.

Nałożone warunki integralnościowe to:

* unikalność [id]
* unikalność [Nazwa]

**CREATE** **TABLE** [dbo].[Kraje](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[Nazwa] [**varchar**](50) **NULL**,

**CONSTRAINT** [PK\_Kraje] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**],

**CONSTRAINT** [UnikalnaNazwa] **UNIQUE** **NONCLUSTERED**

(

[Nazwa] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**SET** ANSI\_PADDING **OFF**

**GO**

**Tabela [dbo].[Miasta]**

Opisuje, w których krajach znajdują się dane miasta.

Klucz główny: [id]

Klucze obce: [KrajID]

Nałożone warunki integralnościowe:

* unikalność [id]

**CREATE** **TABLE** [dbo].[Miasta](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[KrajID] [**int**] **NOT** **NULL**,

[Miasto] [**varchar**](50) **NOT** **NULL**,

[Kod Pocztowy] [**varchar**](10) **NOT** **NULL**,

**CONSTRAINT** [PK\_Miasta] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**SET** ANSI\_PADDING **OFF**

**GO**

**ALTER** **TABLE** [dbo].[Miasta] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Miasta\_Kraje] **FOREIGN** **KEY**([KrajID])

**REFERENCES** [dbo].[Kraje] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Miasta] **CHECK** **CONSTRAINT** [FK\_Miasta\_Kraje]

**GO**

**Tabela [dbo].[Adresy]**

Zawiera informacje odnośnie adresu. Korzystają z niej osoby a także firmy. Jest zabezpieczona przed obecnością w niej dwóch takich samych wierszy poprzed unikalność kombinacji elementów.

Klucz główny: [id]

Klucze obce: [MiastoID]

Nałożone warunki integralnościowe to: unikalność [id], oraz unikalność krotki ([MiastoID], [Ulica], [Numer budynku], [Numer mieszkania])

**CREATE** **TABLE** [dbo].[Adresy](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[MiastoID] [**int**] **NOT** **NULL**,

[Ulica] [**varchar**](35) **NULL**,

[Numer budynku] [**int**] **NULL**,

[Numer mieszkania] [**int**] **NULL**,

**CONSTRAINT** [PK\_Adresy] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**SET** ANSI\_PADDING **OFF**

**GO**

**ALTER** **TABLE** [dbo].[Adresy] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Adresy\_Miasta] **FOREIGN** **KEY**([MiastoID])

**REFERENCES** [dbo].[Miasta] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Adresy] **CHECK** **CONSTRAINT** [FK\_Adresy\_Miasta]

**GO**

**ALTER** **TABLE** [dbo].[Adresy] **WITH** **CHECK** **ADD** **CONSTRAINT** [Numer budynku>0] **CHECK** (([Numer budynku]>(0)))

**GO**

**ALTER** **TABLE** [dbo].[Adresy] **CHECK** **CONSTRAINT** [Numer budynku>0]

**GO**

**ALTER** **TABLE** [dbo].[Adresy] **WITH** **CHECK** **ADD** **CONSTRAINT** [Numer mieszkania >0] **CHECK** (([Numer mieszkania]>(0)))

**GO**

**ALTER** **TABLE** [dbo].[Adresy] **CHECK** **CONSTRAINT** [Numer mieszkania >0]

**GO**

**Tabela [dbo].[Konferencje]**

Zawiera informacje odnośnie konferencji.

Klucz główny: [id]

Brak kluczy obcych.

Nałożone warunki integralnościowe:

* unikalność [id]
* odpowiednia kolejność [Data Rozpoczecia] oraz [Data Zakonczenia]
* dodatnia wartość [Liczba miejsc] i [Cena za dzien]

**CREATE** **TABLE** [dbo].[Konferencje](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[Nazwa Konferencji] [**varchar**](80) **NULL**,

[Opis konferencji] [**varchar**](150) **NULL**,

[**Data** Rozpoczecia] [**date**] **NULL**,

[**Data** Zakonczenia] [**date**] **NULL**,

[Liczba miejsc] [**int**] **NULL**,

[Cena za dzien] [money] **NULL**,

**CONSTRAINT** [PK\_Konferencje] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**SET** ANSI\_PADDING **OFF**

**GO**

**ALTER** **TABLE** [dbo].[Konferencje] **WITH** **CHECK** **ADD** **CONSTRAINT** [CzyDodatniaLiczbaMiejsc] **CHECK** (([Liczba Miejsc]>(0)))

**GO**

**ALTER** **TABLE** [dbo].[Konferencje] **CHECK** **CONSTRAINT** [CzyDodatniaLiczbaMiejsc]

**GO**

**Tabela [dbo].[Dni konferencji]**

Określa dni, w których będzie odbywać się konferencja.

Klucz główny: [id]

Klucz obcy: [KonferencjaID]

Nałożone warunki integralnościowe:

* unikalność [id]
* zawieranie się [Data] w przedziale trawnia konferencji (Trigger)
* nie może być kolizji pomiędzy innym dniem konferencji w ramach tej samej konferencji (Trigger)

**CREATE** **TABLE** [dbo].[Dni Konferencji](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[KonferencjaID] [**int**] **NOT** **NULL**,

[**Data**] [**date**] **NULL**,

**CONSTRAINT** [PK\_Dni Konferencji] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**ALTER** **TABLE** [dbo].[Dni Konferencji] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Dni Konferencji\_Konferencje] **FOREIGN** **KEY**([KonferencjaID])

**REFERENCES** [dbo].[Konferencje] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Dni Konferencji] **CHECK** **CONSTRAINT** [FK\_Dni Konferencji\_Konferencje]

**GO**

**Tabela [dbo].[Znizki]**

Opisuje jakie znizki będą obowiązywać dla odpowiednich dat rezerwacji.

Klucz główny: [id]

Klucz obcy: [KonferencjaID]

Nałożone warunki integralnościowe:

* unikalność [id]
* zniżka musi zawierać się w przedziale (0,1)

**CREATE** **TABLE** [dbo].[Znizki](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[Obowiazuje od] [**date**] **NULL**,

[Obowiazuje **do**] [**date**] **NULL**,

[Znizka] [**float**] **NULL**,

[KonferencjaID] [**int**] **NOT** **NULL**,

**CONSTRAINT** [PK\_ceny konferencji] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**ALTER** **TABLE** [dbo].[Znizki] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_ceny konferencji\_Konferencje] **FOREIGN** **KEY**([KonferencjaID])

**REFERENCES** [dbo].[Konferencje] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Znizki] **CHECK** **CONSTRAINT** [FK\_ceny konferencji\_Konferencje]

**GO**

**ALTER** **TABLE** [dbo].[Znizki] **WITH** **CHECK** **ADD** **CONSTRAINT** [ZnizkaNaKorzysc] **CHECK** (([Znizka]>=(0) **AND** [Znizka]<=(1)))

**GO**

**ALTER** **TABLE** [dbo].[Znizki] **CHECK** **CONSTRAINT** [ZnizkaNaKorzysc]

**GO**

**Tabela [dbo].[Warsztaty]**

Opisuje warsztaty wraz w ceną za miejsce oraz liczbą miejsc.

Klucz główny: [id]

Brak kluczów obcych.

Nałożone warunki integralnościowe:

* unikalność [id]
* [Liczba miejsc] musi być większa od 0.

**CREATE** **TABLE** [dbo].[Warsztaty](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[Nazwa warsztatu] [**varchar**](80) **NULL**,

[Cena] [money] **NULL**,

[Liczba miejsc] [**int**] **NULL**,

**CONSTRAINT** [PK\_Warsztaty] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**SET** ANSI\_PADDING **OFF**

**GO**

**Tabela [dbo].[Dni Warsztatu]**

Określają poszczególne dni warsztatu. Jest to tabela łącznikowa, także posiad klucz główny złożony z dwóch kluczy obcych.

Klucz główny: para ([DzienID], [WarsztatID])

Klucz obcy: [DzienID], [WarsztatID]

Nałożone warunki integralnościowe:

* unikalność [id]
* godzina rozpoczęcia musi być przed godziną zakończenia

**CREATE** **TABLE** [dbo].[Dni warsztatu](

[DzienID] [**int**] **NOT** **NULL**,

[WarsztatID] [**int**] **NOT** **NULL**,

[Godzina rozpoczecia] [**time**](7) **NULL**,

[Godzina zakonczenia] [**time**](7) **NULL**,

**CONSTRAINT** [PK\_Dni warsztatu] **PRIMARY** **KEY** **CLUSTERED**

(

[DzienID] **ASC**,

[WarsztatID] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**ALTER** **TABLE** [dbo].[Dni warsztatu] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Dni warsztatu\_Dni Konferencji] **FOREIGN** **KEY**([DzienID])

**REFERENCES** [dbo].[Dni Konferencji] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Dni warsztatu] **CHECK** **CONSTRAINT** [FK\_Dni warsztatu\_Dni Konferencji]

**GO**

**ALTER** **TABLE** [dbo].[Dni warsztatu] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Dni warsztatu\_Warsztaty] **FOREIGN** **KEY**([WarsztatID])

**REFERENCES** [dbo].[Warsztaty] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Dni warsztatu] **CHECK** **CONSTRAINT** [FK\_Dni warsztatu\_Warsztaty]

**GO**

**Tabela [dbo].[Rezerwacje konferencji]**

Określa rezerwacje na dany dzień konferencji. Rezerwacje można anulować

Klucz główny: [id]

Klucz obcy: [KlientID], [KonferencjaDzienId]

Nałożone warunki integralnościowe:

* Unikalność [id]
* [Data rezerwacji] w momencie dodawania wiersza musi być na 14 dni przed dzisiejszą datą (trigger)
* [Studenci]+[Normalne] musi być większa od zera

**CREATE** **TABLE** [dbo].[Rezerwacje konferencji](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[KlientID] [**int**] **NOT** **NULL**,

[KonferencjaDzienId] [**int**] **NOT** **NULL**,

[**Data** rezerwacji] [**date**] **NULL**,

[**Data** oplacenia] [**date**] **NULL**,

[Studenci] [**int**] **NULL**,

[Normalne] [**int**] **NULL**,

[anulowano] [**bit**] **NULL**,

**CONSTRAINT** [PK\_rezerwacje konferencje] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**ALTER** **TABLE** [dbo].[Rezerwacje konferencji] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_rezerwacje konferencje\_Dni Konferencji] **FOREIGN** **KEY**([KonferencjaDzienId])

**REFERENCES** [dbo].[Dni Konferencji] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Rezerwacje konferencji] **CHECK** **CONSTRAINT** [FK\_rezerwacje konferencje\_Dni Konferencji]

**GO**

**ALTER** **TABLE** [dbo].[Rezerwacje konferencji] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_rezerwacje konferencje\_Klienci] **FOREIGN** **KEY**([KlientID])

**REFERENCES** [dbo].[Klienci] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Rezerwacje konferencji] **CHECK** **CONSTRAINT** [FK\_rezerwacje konferencje\_Klienci]

**GO**

**Tabela [dbo].[Rezerwacje warsztatow]**

Reprezentuje rezerwacje warsztatów. Ze względu na niemożność częściowego uczestnictwa w warsztacie, jedna rezerwacja warsztatu w odniesieniu do jednej rezerwacji konferencji oznacza, że osoby przypisane do danej rezerwacji warsztatu będą uczestniczyć, we wszystkich dniach warsztatu.

Klucz główny: [id]

Klucz obcy: [WarsztatID], [KonferencjaID]

Nałożone warunki integralnościowe:

* Unikalność [id]
* [Data rezerwacji] w momencie dodawania wiersza, musi być na 14 dni przed dzisiejszą datą (trigger)
* [liczba miejsc] musi być większa od zera

**CREATE** **TABLE** [dbo].[Rezerwacje warsztatow](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[WarsztatID] [**int**] **NOT** **NULL**,

[KonferencjaID] [**int**] **NOT** **NULL**,

[**Data** rezerwacji] [**date**] **NULL**,

[**Data** oplacenia] [**date**] **NULL**,

[liczba miejsc] [**int**] **NULL**,

[anulowano] [**bit**] **NULL**,

**CONSTRAINT** [PK\_Rezerwacje warsztatow] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**ALTER** **TABLE** [dbo].[Rezerwacje warsztatow] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Rezerwacje warsztatow\_Rezerwacje konferencji] **FOREIGN** **KEY**([KonferencjaID])

**REFERENCES** [dbo].[Rezerwacje konferencji] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Rezerwacje warsztatow] **CHECK** **CONSTRAINT** [FK\_Rezerwacje warsztatow\_Rezerwacje konferencji]

**GO**

**ALTER** **TABLE** [dbo].[Rezerwacje warsztatow] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Rezerwacje warsztatow\_Warsztaty] **FOREIGN** **KEY**([WarsztatID])

**REFERENCES** [dbo].[Warsztaty] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Rezerwacje warsztatow] **CHECK** **CONSTRAINT** [FK\_Rezerwacje warsztatow\_Warsztaty]

**GO**

**Tabela [dbo].[Oplaty]**

Wiersze w tej tabeli sygnalizują tylko i wyłącznie wykonane płatności, nie płatności oczekujące. Kwotę należna za daną rezerwację da się utworzyć w dowolnym momencie na bazie wierszy w innych tabelach, w szczególności daty rezerwacji oraz progów cenowych za daną konferencję.

Klucz główny: [id]

Klucz obcy: [Rezerwacja konferencji]

Nałożone warunki integralnościowe:

* Unikalność [id]
* Default [Kwota] = 0

**CREATE** **TABLE** [dbo].[Oplaty](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[Rezerwacja konferencji] [**int**] **NOT** **NULL**,

[**Data** wplaty] [**date**] **NULL**,

[Kwota] [money] **NULL**,

**CONSTRAINT** [PK\_oplaty] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**ALTER** **TABLE** [dbo].[Oplaty] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Oplaty\_Rezerwacje konferencji] **FOREIGN** **KEY**([Rezerwacja konferencji])

**REFERENCES** [dbo].[Rezerwacje konferencji] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Oplaty] **CHECK** **CONSTRAINT** [FK\_Oplaty\_Rezerwacje konferencji]

**GO**

**Tabela [dbo].[Uczestnicy konferencji]**

Reprezentuje uczestnictwo danej osoby w danej konferencji w ramach danej rezerwacji. Wiersz w tabeli sygnalizuje tylko, że dana osoba uczestniczy w danym dniu warsztatu.

Klucz główny: [id]

Klucz obcy: [RezerwacjaKonfID], [OsobaID]

Nałożone warunki integralnościowe:

* Unikalność [id],
* Jeden Klient nie może uczestniczyć w wielu konferencjach odbydwających się w tym samym dniu (trigger).

**CREATE** **TABLE** [dbo].[Uczestnicy konferencji](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[RezerwacjaKonfID] [**int**] **NOT** **NULL**,

[OsobaID] [**int**] **NOT** **NULL**,

[KlientID] [**int**] **NOT** **NULL**,

**CONSTRAINT** [PK\_Uczestnicy konferencji] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**ALTER** **TABLE** [dbo].[Uczestnicy konferencji] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Uczestnicy konferencji\_Klienci] **FOREIGN** **KEY**([KlientID])

**REFERENCES** [dbo].[Klienci] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Uczestnicy konferencji] **CHECK** **CONSTRAINT** [FK\_Uczestnicy konferencji\_Klienci]

**GO**

**ALTER** **TABLE** [dbo].[Uczestnicy konferencji] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Uczestnicy konferencji\_Osoby] **FOREIGN** **KEY**([OsobaID])

**REFERENCES** [dbo].[Osoby] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Uczestnicy konferencji] **CHECK** **CONSTRAINT** [FK\_Uczestnicy konferencji\_Osoby]

**GO**

**ALTER** **TABLE** [dbo].[Uczestnicy konferencji] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Uczestnicy konferencji\_rezerwacje konferencje] **FOREIGN** **KEY**([RezerwacjaKonfID])

**REFERENCES** [dbo].[Rezerwacje konferencji] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Uczestnicy konferencji] **CHECK** **CONSTRAINT** [FK\_Uczestnicy konferencji\_rezerwacje konferencje]

**GO**

**Tabela [dbo].[Uczestnicy] (Uczestnictwo w warsztacie)**

Reprezentuje uczestnictwo danej osoby w warsztacie w ramach uczestnictwa w dniu konferencji. Jden wiersz z tabeli oznacza, że dana osoba uczestniczy we wszystkich dniach warsztatu!

Klucz główny: [id]

Klucz obcy: [RezerwacjaID], [UczestnikKonfID]

Nałożone warunki integralnościowe:

* Unikalność [id]
* Jeden Klient nie może uczestniczyć w wielu warsztatach odbywających się w tym samym czasie, nawet jeżeli dopuszczamy dwie konferencje odbywające się w tym samym czasie. (trigger)

**CREATE** **TABLE** [dbo].[Uczestnicy konferencji](

[**id**] [**int**] **IDENTITY**(1,1) **NOT** **NULL**,

[RezerwacjaKonfID] [**int**] **NOT** **NULL**,

[OsobaID] [**int**] **NOT** **NULL**,

[KlientID] [**int**] **NOT** **NULL**,

**CONSTRAINT** [PK\_Uczestnicy konferencji] **PRIMARY** **KEY** **CLUSTERED**

(

[**id**] **ASC**

)**WITH** (PAD\_INDEX = **OFF**, STATISTICS\_NORECOMPUTE = **OFF**, IGNORE\_DUP\_KEY = **OFF**, ALLOW\_ROW\_LOCKS = **ON**, ALLOW\_PAGE\_LOCKS = **ON**) **ON** [**PRIMARY**]

) **ON** [**PRIMARY**]

**GO**

**ALTER** **TABLE** [dbo].[Uczestnicy konferencji] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Uczestnicy konferencji\_Klienci] **FOREIGN** **KEY**([KlientID])

**REFERENCES** [dbo].[Klienci] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Uczestnicy konferencji] **CHECK** **CONSTRAINT** [FK\_Uczestnicy konferencji\_Klienci]

**GO**

**ALTER** **TABLE** [dbo].[Uczestnicy konferencji] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Uczestnicy konferencji\_Osoby] **FOREIGN** **KEY**([OsobaID])

**REFERENCES** [dbo].[Osoby] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Uczestnicy konferencji] **CHECK** **CONSTRAINT** [FK\_Uczestnicy konferencji\_Osoby]

**GO**

**ALTER** **TABLE** [dbo].[Uczestnicy konferencji] **WITH** **CHECK** **ADD** **CONSTRAINT** [FK\_Uczestnicy konferencji\_rezerwacje konferencje] **FOREIGN** **KEY**([RezerwacjaKonfID])

**REFERENCES** [dbo].[Rezerwacje konferencji] ([**id**])

**GO**

**ALTER** **TABLE** [dbo].[Uczestnicy konferencji] **CHECK** **CONSTRAINT** [FK\_Uczestnicy konferencji\_rezerwacje konferencje]

**GO**

**Opis widoków :**

\* Do każdego widoku załączono nagłówek opcjonalnie z przykładowym wierszem.

\*\* Warto zaznaczy, że niektóre z “widoków” są z teoretycznego punktu widzenia procedurami, ale jedynym celem ich wykonania jest utworzenie widoku.

**[dbo].[BrakujaceZgloszenia] –** Dla każdej rezerwacji dnia konferencji oraz każdej rezerwacji warsztatu pokazuje dane tych rezerwacji, które nie mają uzupełnionych danych użytkowników.

****

**CREATE view [dbo].[BrakujaceZgloszenia]**

-- Widok pokazuje zarowno brakujace zgloszenie konferencji jak i warszatow

**as**

**select** 'Rezerwacja konferencji' **as** 'Typ', rk.**id** **as** 'Numer rezerwacji', rk.KlientID **as** 'Numer klienta',

(rk.Studenci+rk.Normalne) **as** 'Ile miejsc zarezerwowano',

(**select** **count**(\*) **from** [Uczestnicy Konferencji] **as** uk **where** uk.RezerwacjaKonfID = rk.**id**) **as** 'Ile uzupelniono',

(rk.Studenci+rk.Normalne)-(**select** **count**(\*) **from** [Uczestnicy Konferencji] **as** uk **where** uk.RezerwacjaKonfID = rk.**id**) **as** 'Ile brakuje',

dk.**Data** **as** 'Dzien na ktory wykonano rezerwacje'

-- Laczenie tabel

**from** [Rezerwacje konferencji] **as** rk

**inner** **join** [Dni Konferencji] **as** dk

**on** rk.KonferencjaDzienId = dk.**id**

-- Warunek na date (2 tygodnie przed) oraz anulowanie

**where** rk.anulowano = 0 **and** **dateadd**(**day**, -14, **GETDATE**()) <= dk.**Data**

**group** **by** rk.**id**, rk.KlientID, dk.**Data**, (rk.Studenci+rk.Normalne)

-- Warunek na ilosc

**having** (**select** **count**(\*) **from** [Uczestnicy Konferencji] **as** uk **where** uk.RezerwacjaKonfID = rk.**id**) < (rk.Studenci+rk.Normalne)

-- Teraz czesc od warsztatow

**union**

**select** 'Rezerwacja warsztatu' **as** 'Typ', rw.**id** **as** 'Numer rezerwacji', rk.KlientID **as** 'Numer klienta',

rw.[Liczba miejsc] **as** 'Ile miejsc zarezerwowano',

(**select** **count**(\*) **from** [Uczestnicy] **as** u **where** u.RezerwacjaID = rk.**id**) **as** 'Ile uzupelniono',

rw.[Liczba miejsc] - (**select** **count**(\*) **from** [Uczestnicy] **as** u **where** u.RezerwacjaID = rk.**id**) **as** 'Ile brakuje',

dk.**Data**

**from** [Rezerwacje warsztatow] **as** rw

**inner** **join** Warsztaty **as** w

**on** w.**id** = rw.WarsztatID

**inner** **join** [Rezerwacje konferencji] **as** rk

**on** rk.**id** = rw.KonferencjaID

**inner** **join** [Dni Konferencji] **as** dk

**on** rk.KonferencjaDzienId = dk.**id**

-- Warunek na date oraz anulowanie

**where** rw.anulowano = 0 **and** (**dateadd**(**day**, -14, **GETDATE**()) <= (**select** **top** 1 dk.**Data**

**from** [Dni warsztatu] **as** dw

**inner** **join** [Dni Konferencji] **as** dk

**on** dk.**id** = dw.DzienID **where** dw.WarsztatID = w.**id** **order** **by** dk.**Data** **DESC**))

**group** **by** rk.**id**, rw.**id**, rk.KlientID, rw.[Liczba miejsc], dk.**Data**

-- Warunek na ilosc

**having** (**select** **count**(\*) **from** [Uczestnicy] **as** u **where** u.RezerwacjaID = rk.**id**) < rw.[Liczba miejsc]

**GO**

**[dbo].[KlienciFirmy] –** Pokazuje klientów, którzy są firmami

****

**CREATE view [dbo].[KlienciFirmy]**

**as**

**select** k.**id**, f.[Nazwa Firmy], f.[Telefon Kontaktowy], (o.Imie + ' ' + o.Nazwisko) **as** "Osoba do kontaktu", k.Email **from** Klienci **as** k

**inner** **join** Osoby **as** o

**on** k.OsobaID = o.**id**

**inner** **join** Firmy **as** f

**on** k.**id** = f.KlientID

**where** k.CzyFirma = 1

**GO**

**[dbo].[KlienciOsoby] –** Pokazuje klientów, którzy są osobami prywatnymi

****

**CREATE view [dbo].[KlienciOsoby]**

**as**

**select** k.**id**,(o.Imie + ' ' + o.Nazwisko) **as** "Imie i nazwisko", o.Telefon **as** "Numer telefonu", k.Email **from** Klienci **as** k

**inner** **join** Osoby **as** o

**on** k.OsobaID = o.**id**

**where** k.CzyFirma = 0

**GO**

**[dbo].[NajcześciejKorzystajacy] –** Pokazuje klientów, którzy zarezerwowali najwięcej dni w naszej firmie.

****

**CREATE view [dbo].[NajczesciejKorzystajacy]**

**as**

--Najpierw dla firm

**select** **top** 100 k.**id**, f.[Nazwa Firmy] **as** "Nazwa", **count**(rk.**id**) **as** "Ilosc zarezerwowanych dni" **from** Klienci **as** k

**inner** **join** [Rezerwacje konferencji] **as** rk

**on** rk.KlientID = k.**id**

**inner** **join** Firmy **as** f

**on** f.KlientID = k.**id**

**where** k.CzyFirma = 1 **and** rk.anulowano = 0 --Warunek na anulowanie

**group** **by** k.**id**, f.[Nazwa Firmy]

--Teraz dla osob prywatynch

**union**

**select** **top** 100 k.**id**, (o.Imie + ' ' + o.Nazwisko) **as** Nazwa, **count**(rk.**id**) **as** "Ilosc zarezerwowanych dni" **from** Klienci **as** k

**inner** **join** [Rezerwacje konferencji] **as** rk

**on** rk.KlientID = k.**id**

**inner** **join** Osoby **as** o

**on** k.OsobaID = o.**id**

**where** k.CzyFirma = 0 **and** rk.anulowano = 0 --Warunek na anulowanie

**group** **by** k.**id**, (o.Imie + ' ' + o.Nazwisko)

**order by 3 DESC**

**GO**

**[dbo].[NajpopularniejszeKonferencje] –** Pokazuje te konferencje, na które było najwięcej zarezerwowanych miejsc.



**CREATE view [dbo].[NajpopularniejszeKonferencje]**

**as**

**select** **top** 100 k.**id**, k.[Nazwa Konferencji], **sum**(rk.Normalne+rk.Studenci) **as** "Liczba zarezerwowanych miejsc" **from** Konferencje **as** k

**inner** **join** [Dni Konferencji] **as** dk

**on** dk.KonferencjaID = k.**id**

**inner** **join** [Rezerwacje konferencji] **as** rk

**on** rk.KonferencjaDzienId = dk.**id**

**where** rk.anulowano = 0 --Warunek na anulowanie

**group** **by** k.**id**, k.[Nazwa Konferencji]

**order** **by** **sum**(rk.Normalne+rk.Studenci) **DESC**

**GO**

**[dbo].[NajpopularniejszeWarsztaty] –** Pokazuje te warsztaty, na które było najwięcej zarezerwowanych miejsc.

****

**CREATE view [dbo].[NajpopularniejszeWarsztaty]**

**as**

**select** **top** 100 w.**id**, w.[Nazwa warsztatu], **sum**(rw.[Liczba miejsc]) **as** "Liczba zarezerwowanych miejsc"

**from** Warsztaty **as** w

**inner** **join** [Rezerwacje warsztatow] **as** rw

**on** rw.WarsztatID = w.**id**

**where** rw.anulowano = 0 --Warunek na anulowanie

**group** **by** w.**id**, w.[Nazwa warsztatu]

**order** **by** **sum**(rw.[Liczba miejsc]) **DESC**

**GO**

**[dbo].[WolneMiejscaKonf] –** Dla każdego dnia konferencji pokazuje wolne miejsca. Bardzo użyteczne do użytku we front-endzie.

****

**CREATE view [dbo].[WolneMiejscaKonf]**

**as**

**select** 'Dzien konferencji' **as** 'Typ', k.[Nazwa Konferencji] **as** 'Nazwa konferencji', dk.**id** **as** 'Numer dnia', dk.**Data** **as** 'Dzien',

(k.[Liczba miejsc]) **as** 'Liczba miejsc',

(**sum**(rk.normalne) + **sum**(rk.Studenci)) **as** 'Liczba zajetych miejsc',

(k.[Liczba miejsc])-(**sum**(rk.normalne) + **sum**(rk.Studenci)) **as** 'Ile miejsc zostalo'

-- Laczenie tabel

**from** [Dni Konferencji] **as** dk

**inner** **join** [Rezerwacje konferencji] **as** rk

**on** rk.KonferencjaDzienId = dk.**id**

**inner** **join** Konferencje **as** k

**on** k.**id** = dk.KonferencjaID

**where** rk.anulowano = 0

**group** **by** k.[Nazwa Konferencji], dk.**id**, dk.**Data**, k.[Liczba miejsc]

**GO**

**[dbo].[WolneMiejscaWarsztatow]** – Dla każdego warsztatu pokazuje wolne miejsca. Także bardzo użyteczne do użytku we front-endzie.



**CREATE** **view** [dbo].[WolneMiejscaWarsztatow]

**as**

**select** 'Warsztat' **as** 'Typ', w.[Nazwa warsztatu] **as** 'Nazwa warsztatu', w.**id** **as** 'Numer warsztatu',

(w.[Liczba miejsc]) **as** 'Liczba miejsc',

**sum**(rw.[liczba miejsc]) **as** 'Liczba zajetych miejsc',

(w.[Liczba miejsc])-(**sum**(rw.[liczba miejsc])) **as** 'Ile miejsc zostalo'

**from** [Rezerwacje warsztatow] **as** rw

**inner** **join** Warsztaty **as** w

**on** w.**id** = rw.WarsztatID

**where** rw.anulowano = 0

**group** **by** w.**id**, w.[Nazwa warsztatu], (w.[Liczba miejsc])

**GO**

**[dbo].[CenyNaDniKonferencji]** – Dla dni podanej konferencji rozpisuje okresy wraz z obowiązującymi w nich cenami. Jako @param podajemy @id konferencji.

****

**CREATE PROCEDURE [dbo].[CenyNaDniKonferencji]**

@param **varchar**(50) --id konferencji lub nazwa

**AS**

**set** nocount **on**

**SELECT** dk.**Data** **as** 'Dzien', z.[Obowiazuje od], z.[Obowiazuje **do**], k.[Cena za dzien]\*(1-z.Znizka) **as** 'Kwota za dzien, za osobe w podanym terminie' **from** [Dni Konferencji] **as** dk

**inner** **join** Konferencje **as** k

**on** dk.KonferencjaID = k.**id**

**inner** **join** Znizki **as** z

**on** z.KonferencjaID = k.**id**

**where** k.[Nazwa Konferencji] **like** @param **or** k.**id** **like** @param

**GO**

**[dbo].[IdentyfikatoryKonferencji]** – Generuje tablicę identyfikatorów na daną konferencję. Jako parametr przyjmuje @id konferencji.

****

**CREATE** **PROCEDURE** [dbo].[IdentyfikatoryKonferencji]

@param **int** --id konferencji

**AS**

**select** o.Imie, o.Nazwisko, isnull(f.[Nazwa Firmy], 'Osoba prywatna') **as** "Firma", o.Foto **as** "Sciezka do zdjecia" **from** [Uczestnicy konferencji] **as** uk

**inner** **join** Osoby **as** o

**on** o.**id** = uk.OsobaID

**inner** **join** Klienci **as** k

**on** k.**id** = uk.KlientID

**inner** **join** Firmy **as** f

**on** k.**id** = f.KlientID

-- Warunek na konferencje

**where** o.**id** **in** ( **select** o.**id** **from** Osoby **as** o

**inner** **join** [Uczestnicy konferencji] **as** uk

**on** uk.OsobaID = o.**id**

**inner** **join** [Rezerwacje konferencji] **as** rk

**on** rk.**id** = uk.RezerwacjaKonfID

**inner** **join** [Dni Konferencji] **as** dk

**on** dk.**id** = rk.**id**

**inner** **join** Konferencje **as** k

**on** k.**id** = dk.KonferencjaID

**where** k.**id** = @param)

**GO**

**[dbo].[IdentyfikatoryDniaKonferencji]** – Generuje tablicę identyfikatorów na dany dzień konferencji. Jako parametr przyjmuje @id dnia konferencji.

****

**CREATE PROCEDURE [dbo].[IdentyfikatoryDniaKonferencji]**

@param **int** --id dnia konferencji

**AS**

**set** nocount **on**

**select** o.Imie, o.Nazwisko, isnull(f.[Nazwa Firmy], 'Osoba prywatna') **as** "Firma", o.Foto **as** "Sciezka do zdjecia" **from** [Uczestnicy konferencji] **as** uk

**inner** **join** Osoby **as** o

**on** o.**id** = uk.OsobaID

**inner** **join** Klienci **as** k

**on** k.**id** = uk.KlientID

**inner** **join** Firmy **as** f

**on** k.**id** = f.KlientID

-- Warunek na dzien konferencji

**where** o.**id** **in** ( **select** o.**id** **from** Osoby **as** o

**inner** **join** [Uczestnicy konferencji] **as** uk

**on** uk.OsobaID = o.**id**

**inner** **join** [Rezerwacje konferencji] **as** rk

**on** rk.**id** = uk.RezerwacjaKonfID

**inner** **join** [Dni Konferencji] **as** dk

**on** dk.**id** = rk.**id**

**where** dk.**id** = @param)

**GO**

**[dbo].[IdentyfikatoryRezerwacjiKonferencji]** – Generuje tablicę identyfikatorów na daną rezerwację dnia konferencji. Jako parametr przyjmuje @id rezerwacji dnia konferencji.



**CREATE** **PROCEDURE** [dbo].[IdentyfikatoryRezerwacjiKonferencji]

@param **int** --id rezerwacji konferencji

**AS**

**set** nocount **on**

**select** o.Imie, o.Nazwisko, isnull(f.[Nazwa Firmy], 'Osoba prywatna') **as** "Firma", o.Foto **as** "Sciezka do zdjecia" **from** [Uczestnicy konferencji] **as** uk

**inner** **join** Osoby **as** o

**on** o.**id** = uk.OsobaID

**inner** **join** Klienci **as** k

**on** k.**id** = uk.KlientID

**inner** **join** Firmy **as** f

**on** k.**id** = f.KlientID

-- Warunek na rezerwacje konferencji

**where** o.**id** **in** (

**select** o.**id** **from** Osoby **as** o

**inner** **join** [Uczestnicy konferencji] **as** uk

**on** uk.OsobaID = o.**id**

**inner** **join** [Rezerwacje konferencji] **as** rk

**on** rk.**id** = uk.RezerwacjaKonfID

**where** rk.**id** = @param

)

**GO**

**[dbo].[IdentyfikatoryWarsztatu]** – Generuje tablicę identyfikatorów na dany warsztat. Jako parametr przyjmuje @id warsztatu.



CREATE PROCEDURE [dbo].[IdentyfikatoryWarsztatu]

@param **int** --id warsztatu

**AS**

**set** nocount **on**

**select** o.Imie, o.Nazwisko, isnull(f.[Nazwa Firmy], 'Osoba prywatna') **as** "Firma", o.Foto **as** "Sciezka do zdjecia" **from** [Uczestnicy konferencji] **as** uk

**inner** **join** Osoby **as** o

**on** o.**id** = uk.OsobaID

**inner** **join** Klienci **as** k

**on** k.**id** = uk.KlientID

**inner** **join** Firmy **as** f

**on** k.**id** = f.KlientID

-- Warunek na warsztat

**where** o.**id** **in** (

**select** o.**id** **from** Osoby **as** o

**inner** **join** [Uczestnicy konferencji] **as** uk

**on** uk.**id** = uk.OsobaID

**inner** **join** [Uczestnicy] **as** u

**on** u.RezerwacjaID = uk.**id**

**inner** **join** [Rezerwacje warsztatow] **as** wk

**on** wk.**id** = u.RezerwacjaID

**inner** **join** Warsztaty **as** w

**on** w.**id** = wk.**id**

**where** w.**id** = @param

)

**GO**

**[dbo].[IdentyfikatoryRezerwacjiWarsztatu]** – Generuje tablicę identyfikatorów na daną rezerwację warsztatu. Jako parametr przyjmuje @id rezerwacji warsztatu.



**CREATE** **PROCEDURE** [dbo].[**IdentyfikatoryRezerwacjiWarsztatu**]

@param **int** --id warsztatu

**AS**

**set** nocount **on**

**select** o.Imie, o.Nazwisko, isnull(f.[Nazwa Firmy], 'Osoba prywatna') **as** "Firma", o.Foto **as** "Sciezka do zdjecia" **from** [Uczestnicy konferencji] **as** uk

**inner** **join** Osoby **as** o

**on** o.**id** = uk.OsobaID

**inner** **join** Klienci **as** k

**on** k.**id** = uk.KlientID

**inner** **join** Firmy **as** f

**on** k.**id** = f.KlientID

-- Warunek na warsztat

**where** o.**id** **in** (

**select** o.**id** **from** Osoby **as** o

**inner** **join** [Uczestnicy konferencji] **as** uk

**on** uk.**id** = uk.OsobaID

**inner** **join** [Uczestnicy] **as** u

**on** u.RezerwacjaID = uk.**id**

**inner** **join** [Rezerwacje warsztatow] **as** wk

**on** wk.**id** = u.RezerwacjaID

**where** wk.**id** = @param

)

**GO**

**[dbo].[NaCoZapisany]** – Dla podanego @id Osoby podaje wszystkie dni konferencji oraz warsztaty na które jest zapisany.

****

**CREATE** **PROCEDURE** [dbo].[NaCoZapisany]

@param **varchar**(50) --id lub imie nazwisko z tabeli osoby

**AS**

**set** nocount **on**

**SELECT** w.[Nazwa warsztatu], dk.**Data**, dw.[Godzina rozpoczecia], dw.[Godzina zakonczenia]

**from** Osoby **as** o

**inner** **join** [Uczestnicy konferencji] **as** uk

**on** o.**id** = uk.OsobaID

**inner** **join** Uczestnicy **as** u

**on** u.UczestnikKonfID = uk.**id**

**inner** **join** [Rezerwacje warsztatow] **as** rw

**on** u.RezerwacjaID = rw.**id**

**inner** **join** Warsztaty **as** w

**on** w.**id** = rw.WarsztatID

**inner** **join** [Dni warsztatu] **as** dw

**on** dw.WarsztatID = w.**id**

**inner** **join** [Dni Konferencji] **as** dk

**on** dw.DzienID = dk.**id**

**where** (o.**id** **like** @param **or** (o.Imie + o.Nazwisko) **like** @param) **and** rw.anulowano = 0 --Warunek na anulowanie

--sortowanie po dacie

**order** **by** dk.**Data**

**GO**

**[dbo].[OsobyNaKonferencji]** – Pokazuje wszystkie osoby, które mają rezerwację na daną konferencję. Jako parametr przyjmuje @id konferencji.

****

create PROCEDURE OsobyNaKonferencji

@param **varchar**(50) --id konferencji lub nazwa

**AS**

**set** nocount **on**

**SELECT** o.**id**, o.Imie, o.Nazwisko, o.[**Data** urodzenia], o.Plec, o.[Legitymacja studencka nr]

--Laczenie tabel

**from** Osoby **as** o

**inner** **join** [Uczestnicy konferencji] **as** uk

**on** uk.OsobaID = o.**id**

**inner** **join** [Rezerwacje konferencji] **as** rk

**on** uk.RezerwacjaKonfID = rk.**id**

**inner** **join** [Dni Konferencji] **as** dk

**on** rk.KonferencjaDzienId = dk.**id**

**inner** **join** Konferencje **as** k

**on** k.**id** = dk.KonferencjaID

**where** (k.[Nazwa Konferencji] **like** @param **or** k.**id** **like** @param)

**GO**

**[dbo].[OsobyNaDniuKonferencji]** – Pokazuje wszystkie osoby, które mają rezerwację na dany dzień konferencji. Jako parametr przyjmuje @id dnia konferencji.



**CREATE** **PROCEDURE** [dbo].[OsobyNaDniuKonferencji]

@param **int** --id dnia konferencji

**AS**

**set** nocount **on**

**SELECT** o.**id**, o.Imie, o.Nazwisko, o.[**Data** urodzenia], o.Plec, o.[Legitymacja studencka nr]

--Laczenie tabel

**from** Osoby **as** o

**inner** **join** [Uczestnicy konferencji] **as** uk

**on** uk.OsobaID = o.**id**

**inner** **join** [Rezerwacje konferencji] **as** rk

**on** uk.RezerwacjaKonfID = rk.**id**

**inner** **join** [Dni Konferencji] **as** dk

**on** rk.KonferencjaDzienId = dk.**id**

**where** dk.**id** = @param **and** rk.anulowano = 0

**GO**

**[dbo].OsobyNaRezerwacjiKonferencji]** – Pokazuje wszystkie osoby, które są w danej rezerwacji konferencji. Jako parametr przyjmuje @id rezerwacji dnia konferencji.



**CREATE** **PROCEDURE** [dbo].[OsobyNaRezerwacjiKonferencji]

@param **int** --id konferencji

**AS**

**set** nocount **on**

**SELECT** o.**id**, o.Imie, o.Nazwisko, o.[**Data** urodzenia], o.Plec, o.[Legitymacja studencka nr]

--Laczenie tabel

**from** Osoby **as** o

**inner** **join** [Uczestnicy konferencji] **as** uk

**on** uk.OsobaID = o.**id**

**inner** **join** [Rezerwacje konferencji] **as** rk

**on** uk.RezerwacjaKonfID = rk.**id**

**where** rk.**id** = @param **and** rk.anulowano = 0

**GO**

**[dbo].OsobyNaRezerwacjiWarsztatu]** – Pokazuje wszystkie osoby, które znajdują się w danej rezerwacji warsztatu. Jako parametr przyjmuje @id warsztatu.



**CREATE** **PROCEDURE** [dbo].[OsobyNaRezerwacjiWarsztatu]

@param **int** --id warsztatu

**AS**

**set** nocount **on**

**SELECT** o.**id**, o.Imie, o.Nazwisko, o.[**Data** urodzenia], o.Plec, o.[Legitymacja studencka nr]

--Laczenie tabel

**from** Osoby **as** o

**inner** **join** [Uczestnicy konferencji] **as** uk

**on** uk.OsobaID = o.**id**

**inner** **join** [Uczestnicy] **as** u

**on** u.UczestnikKonfID = uk.**id**

**inner** **join** [Rezerwacje warsztatow] **as** rw

**on** rw.**id** = u.RezerwacjaID

**where** rw.**id** **like** @param **and** rw.anulowano = 0

**GO**

**[dbo].OsobyNaWarsztacie]** – Dla podanego @id warsztatu pokazuje wszystkie osoby posiadające rezerwację na dany warsztat.

****

**CREATE PROCEDURE OsobyNaWarsztacie**

@param **varchar**(50) --id warsztatu lub nazwa

**AS**

**set** nocount **on**

**SELECT** o.**id**, o.Imie, o.Nazwisko, o.[**Data** urodzenia], o.Plec, o.[Legitymacja studencka nr]

--Laczenie tabel

**from** Osoby **as** o

**inner** **join** [Uczestnicy konferencji] **as** uk

**on** uk.OsobaID = o.**id**

**inner** **join** [Uczestnicy] **as** u

**on** u.UczestnikKonfID = uk.**id**

**inner** **join** [Rezerwacje warsztatow] **as** rw

**on** rw.**id** = u.RezerwacjaID

**inner** **join** Warsztaty **as** w

**on** w.**id** = rw.WarsztatID

**where** w.[Nazwa warsztatu] **like** @param **or** w.**id** **like** @param

**GO**

**[dbo].[UczestnicyOdFirmy]** – Dla podanego @id firmy pokazuje wszystkich uczestników konferencji przez niego zarejestrowanych.

****

CREATE PROCEDURE [dbo].[UczestnicyOdFirmy]

@param **varchar**(50) --id firmy lub nazwa firmy

**AS**

**set** nocount **on**

**SELECT** o.**id**, o.Imie, o.Nazwisko, o.[**Data** urodzenia], o.Plec, o.[Legitymacja studencka nr] **from** [Uczestnicy konferencji] **as** uk

**inner** **join** Osoby **as** o

**on** uk.OsobaID = o.**id**

**inner** **join** Klienci **as** k

**on** k.**id** = uk.KlientID

**inner** **join** Firmy **as** f

**on** f.KlientID = k.**id**

**where** (k.CzyFirma = 1) **and** (f.[Nazwa Firmy] **like** @param **or** f.KlientID **like** @param)

**and** (**select** anulowano **from** [Rezerwacje konferencji] **as** rk **where** rk.**id** = uk.**id**) = 0 --Warunek na anulowanie

**GO**

**[dbo].[FakturaNaKonf] (na dzień konferencji) *–*** Dla podanego numeru rezerwacji dnia tworzy widok faktury. Za argument podaje się @rezerwacjaID.



CREATE PROCEDURE [dbo].[FakturaNaKonf]

@rezerwacjaID **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**declare** @DoZplacenia money

**declare** @Zplacono money

**declare** @klientID **int**

**declare** @iscomp **bit**

**declare** @adresID **int**

**set** @klientID=(**select** KlientID **from** [Rezerwacje konferencji] **where** **id** = @rezerwacjaID)

**set** @DoZplacenia=(**select** dbo.PodliczRezerwacjafun(@rezerwacjaID))

**set** @Zplacono=isnull((**select** **sum**(Kwota) **from** Oplaty **where** [Rezerwacja konferencji]=@rezerwacjaID),0)

**set** @iscomp=(**select** CzyFirma **from** Klienci **where** **id**=@klientID)

**set** @adresID=(**select** AdresID **from** Osoby **where** **id**=(**select** osobaID **from** Klienci **where** **id**=@klientID))

**if** @iscomp=0

**begin**

**select** (**select** imie **from** Osoby **where** **id**=(**select** osobaID **from** Klienci **where** **id**=@klientID))**as** imie,

(**select** nazwisko **from** Osoby **where** **id**=(**select** osobaID **from** Klienci **where** **id**=@klientID))**as** nazwisko,

(**select** nazwa **from** Kraje **where** **id**=(**select** KrajID **from** miasta **where** **id**=(**select** MiastoID **from** Adresy **where** **id**=@adresID)))**as** kraj,

(**select** miasto **from** miasta **where** **id**=(**select** MiastoID **from** Adresy **where** **id**=@adresID))**as** miasto,

(**select** [Kod Pocztowy] **from** miasta **where** **id**=(**select** MiastoID **from** Adresy **where** **id**=@adresID))**as** [kod pocztowy],

(**select** Ulica **from** Adresy **where** **id**=@adresID) **as** ulica,

(**select** [Numer budynku] **from** Adresy **where** **id**=@adresID) **as** [numer domu],

(**select** [Numer mieszkania] **from** Adresy **where** **id**=@adresID) **as** [numer mieszkania],

@DoZplacenia **as** należność,@Zplacono **as** Zapłacono,@DoZplacenia-@Zplacono **as** saldo

**end**

**else**

**begin**

**select** (**select** imie **from** Osoby **where** **id**=(**select** osobaID **from** Klienci **where** **id**=@klientID))**as** imie,

(**select** nazwisko **from** Osoby **where** **id**=(**select** osobaID **from** Klienci **where** **id**=@klientID))**as** nazwisko,

(**select** [Nazwa Firmy] **from** Firmy **where** KlientID=@klientID) **as** [nazwa firmy],

(**select** nazwa **from** Kraje **where** **id**=(**select** KrajID **from** miasta **where** **id**=(**select** MiastoID **from** Adresy **where** **id**=@adresID)))**as** kraj,

(**select** miasto **from** miasta **where** **id**=(**select** MiastoID **from** Adresy **where** **id**=@adresID))**as** miasto,

(**select** [Kod Pocztowy] **from** miasta **where** **id**=(**select** MiastoID **from** Adresy **where** **id**=@adresID))**as** [kod pocztowy],

(**select** Ulica **from** Adresy **where** **id**=@adresID) **as** ulica,

(**select** [Numer budynku] **from** Adresy **where** **id**=@adresID) **as** [numer domu],

(**select** [Numer mieszkania] **from** Adresy **where** **id**=@adresID) **as** [numer mieszkania],

@DoZplacenia **as** należność,@Zplacono **as** Zapłacono,@DoZplacenia-@Zplacono **as** saldo

**end**

**END**

**GO**

**Opis triggerów :**

**Tabela [dbo].[Konferencje]**

[DataKonferencjiWPrzyszłości] **–** Wykonuje się w momencie kiedy dodajemy nową konferencję. Sprawdza, czy nie staramy się dodawać konferencji z datami, które były w przeszłości, aby uniknąć błędów. W momencie gdy jest rzeczywiście konieczne dodanie takiej konferencji, musi to być wykonane przez administratora z tymczasowo dezaktywowanym tym triggerem.

**CREATE TRIGGER [dbo].[DataKonferencjiWPrzyszłości]**

**ON** [dbo].[Konferencje]

**AFTER** **INSERT**,**UPDATE**

**AS**

**BEGIN**

**SET** NOCOUNT **ON**;

**DECLARE** @Date **date** = (**SELECT** [**Data** Rozpoczecia] **FROM** inserted)

**IF**((**DATEDIFF**(**day**,**GETDATE**(),@Date) <= 0))

**BEGIN** ;

THROW 52000,'The conference is starting today or has already started! You can only specify future conferences.',1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

**Tabela [dbo].[Dni konferencji]**

[SprawdzDwaTeSameDni] **–** Wykonywany w momencie modyfikacji tabeli dni konferencji. Sprawdza czy dwa dni danej konferencji nie mają tej samej daty. Także chroni to przed błędami.

**CREATE TRIGGER [dbo].[SprawdzDwaTeSmeDni]**

**ON** [dbo].[Dni Konferencji]

**AFTER** **INSERT**,**UPDATE**

**AS**

**BEGIN**

**DECLARE** @confID **int** = (**select** KonferencjaID **from** inserted)

**DECLARE** @date **date** = (**select** **Data** **from** inserted)

**IF** ((**SELECT** **COUNT**(**id**) **FROM** [Dni Konferencji] **WHERE** (**Data** = @date) **AND** (KonferencjaID = @confID) ) > 1)

**BEGIN**

**DECLARE** @message **varchar**(100) = 'Day has already been added for this conference' ;

THROW 52000,@message,1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

[dbo].[SprawdzCzyDataSieZgadza] – Sprawdza czy data zawiera się w datach początku I końca konferencji.

**CREATE TRIGGER [dbo].[SprawdzCzyDataSieZgadza]**

**ON** [dbo].[Dni Konferencji]

**AFTER** **INSERT**,**UPDATE**

**AS**

**BEGIN**

**DECLARE** @confID **int** = (**select** KonferencjaID **from** inserted)

**DECLARE** @date **date** = (**select** **Data** **from** inserted)

**Declare** @od **date** =(**SELECT** [**Data** Rozpoczecia] **FROM** Konferencje **WHERE** **id**=@confID)

**Declare** @do **date** =(**SELECT** [**Data** Zakonczenia] **FROM** Konferencje **WHERE** **id**=@confID)

**IF** (@date<=@od **and** @date>=@do)

**BEGIN**

**DECLARE** @message **varchar**(100) = 'Day is out of conference time' ;

THROW 52000,@message,1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

**Tabela [dbo].[Znizki]**

[DataWygasnienciaPoDacieRozpoczencia] **–** Wykonywany w momenci modyfikcaji tabeli. Zastępuje tutaj constraint, sprawdzając poprawność dat obowiązywania cen.

**CREATE TRIGGER [dbo].[DataWygasnienciaPoDacieRozpoczencia]**

**ON** [dbo].[Znizki]

**AFTER** **INSERT**,**UPDATE**

**AS**

**BEGIN**

**DECLARE** @date **date** = (**SELECT** [Obowiazuje **do**] **FROM** inserted)

**DECLARE** @confstartdate **date** = (**select** [**Data** Rozpoczecia] **from** Konferencje **where** **id**=(**select** KonferencjaID **from** inserted))

**IF** ((**SELECT** **DATEDIFF**(**day**,@Date,@confstartdate)) < 0)

**BEGIN**

;THROW 52000, 'This price stage expires after conference has started.',1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

[JednaCenaWTymSamymCzasie] – Sprawdza czy w wyniku błędu nie obowiązują w tym samym czasie dwie różne ceny.

**CREATE** **TRIGGER** [dbo].[JednaCenaWTymSamymCzasie]

**ON** [dbo].[Znizki]

**AFTER** **INSERT**,**UPDATE**

**AS**

**BEGIN**

**Declare** @id **int** = (**select** **id** **from** inserted)

**DECLARE** @date **date** = (**SELECT** [Obowiazuje od] **FROM** inserted)

**DECLARE** @confID **int** = (**select** KonferencjaID **from** inserted)

**IF** **exists**(**SELECT** **id** **FROM** Znizki **WHERE** ((**id** <> @Id)**AND**(@Date <=[Obowiazuje **do**]) )**AND**(KonferencjaID=@confID))

**BEGIN**

;THROW 52000, 'There is already a price named for this conference day that expires at the same time.',1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

**Tabela [dbo].[Dni Warsztatu]**

[GodzinaRozpoczenciaPrzedGodzZakonczenia] – Wykonywany w momencie modyfikacji. Sprawdza poprawność kolejności godzin. Funkcjonuje jako constraint.

**CREATE** **TRIGGER** [dbo].[GodzinaRozpoczenciaPrzedGodzZakonczenia]

**ON** [dbo].[Dni warsztatu]

**AFTER** **INSERT**,**UPDATE**

**AS**

**BEGIN**

**DECLARE** @start **time**(0) = (**SELECT** [Godzina rozpoczecia] **FROM** inserted)

**DECLARE** @end **time**(0) = (**SELECT** [Godzina zakonczenia] **FROM** inserted)

**IF**((**SELECT** **DATEDIFF**(**minute**,@start,@end))<5)

**BEGIN** ;

THROW 52000,'Workshop has to last at least 5 minutes.',1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

**Tabela [dbo].[Rezerwacje konferencji]**

[liczbaMiejscNaKonferencji] – Wykonywany w momencie modyfikacji tablele rezerwacji. Sprawdza czy transakcja nie przekroczy liczby miejsc wyznaczonej na konferencję.

**CREATE** **TRIGGER** [dbo].[liczbaMiejscNaKonferencji]

**ON** [dbo].[Rezerwacje konferencji]

**AFTER** **INSERT**,**UPDATE**

**AS**

**BEGIN**

**SET** NOCOUNT **ON**;

**DECLARE** @ConferenceId **int** = (**SELECT** dk.KonferencjaID **FROM** [Dni Konferencji] **as** dk **where** (**select** KonferencjaDzienId **from** inserted)=dk.**id** )

**DECLARE** @liczbaM **int** = (**SELECT** [Liczba miejsc] **FROM** Konferencje **where** **id**=@ConferenceId)

**DECLARE** @liczbaMiejscZajetych **int** = (**SELECT** **sum**(Studenci+Normalne) **FROM** [Rezerwacje konferencji] **where** **id**<>(**select** **id** **from** inserted)**and** KonferencjaDzienId=(**select** KonferencjaDzienId **from** inserted))

**IF** (@liczbaM -@liczbaMiejscZajetych < ( **select** Studenci +Normalne **from** inserted))

**BEGIN**

**DECLARE** @message **varchar**(100) = 'For this conference is only ' +**CAST**(@liczbaM -@liczbaMiejscZajetych **as** **varchar**(3))+'free' ;

THROW 52000,@message,1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

[ZabronBukowacNa14DniPrzed] – Sprawdza czy rezerwacje wykonano w odpowednim czasie przed konferencją. W momencie kiedy ktoś wyjątkowo chce zarezerwować miejsce tuż przed konferencją musi skonsultować to z administracją.

**CREATE** **TRIGGER** [dbo].[ZabronBukowacNa14DniPrzed]

**ON** [dbo].[Rezerwacje konferencji]

**AFTER** **INSERT**,**UPDATE**

**AS**

**BEGIN**

**SET** NOCOUNT **ON**;

**DECLARE** @Conferencedate **date** = (**SELECT** **Data** **FROM** [Dni Konferencji] **as** dk **where** (**select** KonferencjaDzienId **from** inserted)=dk.**id** )

**Declare** @date **date** = (**select** [**Data** rezerwacji] **from** inserted)

**IF** ((**DATEADD**(**DAY**,-14,@Conferencedate)) < @Date)

**BEGIN**

;THROW 53000,'The conference is starting in less than two weeks. It is too late to book or update data.',1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

[ZabronZmniejszacIloscMiejscJesliSaUczestnicy] – Zmniejszanie liczby miejsc na rezerwacji w momencie gdy podano już dane użykowników mogłoby zagrozić integralności danych. Należy najpierw usunąć deklaracje uczestnictwa.

**CREATE** **TRIGGER** [dbo].[ZabronZmniejszacIloscMiejscJesliSaUczestnicy]

**ON** [dbo].[Rezerwacje konferencji]

**AFTER** **UPDATE**

**AS**

**BEGIN**

**DECLARE** @ConferenceDayBookingId **int** = (**SELECT** **id** **FROM** inserted)

**DECLARE** @PlacesWanted **int** = (**SELECT** Normalne+Studenci **FROM** inserted)

**DECLARE** @PlacesSet **int** = (**SELECT** **COUNT**(**id**) **FROM** [Uczestnicy konferencji] **WHERE** RezerwacjaKonfID=@ConferenceDayBookingId)

**IF** (@PlacesSet > @PlacesWanted)

**BEGIN**

**DECLARE** @message **varchar**(100) = 'There is too many participants assigned to this booking. Amount of participants: ' +**CAST**(@PlacesSet **as** **varchar**(10))

;THROW 52000,@message,1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

**Tabela [dbo].[Rezerwacje warsztatow]**

[liczbaMiejscNaWarsztacie] – Analogicznie do rezerwacji konferencji sprawdza ilośc miejsc na warsztacie.

**CREATE** **TRIGGER** [dbo].[liczbaMiejscNaWarsztacie]

**ON** [dbo].[Rezerwacje warsztatow]

**AFTER** **INSERT**,**UPDATE**

**AS**

**BEGIN**

**SET** NOCOUNT **ON**;

**DECLARE** @warsztatId **int** = (**select** WarsztatID **from** inserted)

**DECLARE** @liczbaM **int** = (**SELECT** [Liczba miejsc] **FROM** Warsztaty **where** **id**=@warsztatId)

**DECLARE** @liczbaMiejscZajetych **int** = (**select** **sum**([liczba miejsc]) **from** [Rezerwacje warsztatow] **where** WarsztatID=@warsztatId)

**IF** (@liczbaM -@liczbaMiejscZajetych < ( **select** [liczba miejsc] **from** inserted))

**BEGIN**

**DECLARE** @message **varchar**(100) = 'For this workshop is only ' +**CAST**(@liczbaM -@liczbaMiejscZajetych **as** **varchar**(3))+'free' ;

THROW 52000,@message,1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

**Tabela [dbo].[Uczestnicy konferencji]**

[JednaOsobaMozeBycRylkoRazNatejSamejKonferencji] – Sprawdzamy czy użykownik nie stara się dodać tego samego użytkownika na dany dzień konferencji. Nie ma najmniejszego sensu tego robić.

**CREATE** **TRIGGER** [dbo].[JednaOsobaMozeBycRylkoRazNatejSamejKonferencji]

**ON** [dbo].[Uczestnicy konferencji]

**AFTER** **INSERT**,**UPDATE**

**AS**

**BEGIN**

**DECLARE** @confId **int** = (**SELECT** RezerwacjaKonfID **FROM** inserted)

**DECLARE** @osobaID **int** = (**SELECT** OsobaID **FROM** inserted)

**DECLARE** @HowMany **int** = (**SELECT** **COUNT**(**id**) **FROM** [Uczestnicy konferencji] **WHERE** RezerwacjaKonfID=@confId **and** OsobaID=@osobaID)

**IF** (@HowMany>1)

**BEGIN**

**DECLARE** @message **varchar**(100) = 'You can asign One person only once fore the same boking'

;THROW 52000,@message,1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

[ZabronZmniejszacDodawacZaDuzoUczestnikow] – Sprawdza czy nie dodano za dużo uczestników na daną rezerwację.

**CREATE** **TRIGGER** [dbo].[ZabronZmniejszacDodawacZaDuzoUczestnikow]

**ON** [dbo].[Uczestnicy konferencji]

**AFTER** **INSERT**

**AS**

**BEGIN**

**DECLARE** @Id **int** = (**SELECT** RezerwacjaKonfID **FROM** inserted)

**DECLARE** @PlacesWanted **int** = (**SELECT** Normalne+Studenci **FROM** [Rezerwacje konferencji] **where** **id**=@id)

**DECLARE** @PlacesSet **int** = (**SELECT** **COUNT**(**id**) **FROM** [Uczestnicy konferencji] **WHERE** RezerwacjaKonfID=@id)

**IF** (@PlacesSet > @PlacesWanted)

**BEGIN**

**DECLARE** @message **varchar**(100) = 'There is too many participants assigned to this booking. Amount of participants: ' +**CAST**(@PlacesWanted **as** **varchar**(10))+'you czn bok'

;THROW 52000,@message,1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

**Tabela [dbo].[Uczestnicy] (Uczestnictwo w warsztacie)**

[JednaOsobaMozeBycRylkoRazNatymSamymWarsztacie] – Analogicznie jak w przypadku konferencji sprawdza czy użytkownik nie chce zapisać kogoś dwa razy na ten sam warsztat.

**CREATE** **TRIGGER** [dbo].[JednaOsobaMozeBycRylkoRazNatymSamymWarsztacie]

**ON** [dbo].[Uczestnicy]

**AFTER** **INSERT**,**UPDATE**

**AS**

**BEGIN**

**DECLARE** @RezerwId **int** = (**SELECT** RezerwacjaID **FROM** inserted)

**DECLARE** @UczestnikID **int** = (**SELECT** UczestnikKonfID **FROM** inserted)

**DECLARE** @HowMany **int** = (**SELECT** **COUNT**(**id**) **FROM** Uczestnicy **WHERE** RezerwacjaID=@RezerwId **and** UczestnikKonfID=@UczestnikID)

**IF** (@HowMany>1)

**BEGIN**

**DECLARE** @message **varchar**(100) = 'You can asign One person only once fore the same boking'

;THROW 52000,@message,1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

[ZabronDodawacZaDuzoOsubNaWarsztat] – Analogicznie jak w przypadku konferencji sprawdza ilość użytkowników wprowadzonych na daną rezerwację warsztatu i czy oby nie ma ich za dużo.

**CREATE** **TRIGGER** [dbo].[ZabronDodawacZaDuzoOsubNaWarsztat]

**ON** [dbo].[Uczestnicy]

**AFTER** **INSERT**,**UPDATE**

**AS**

**BEGIN**

**DECLARE** @RezerwId **int** = (**SELECT** RezerwacjaID **FROM** inserted)

**DECLARE** @PlacesWanted **int** = (**SELECT** [liczba miejsc] **FROM** [Rezerwacje warsztatow] **where** **id**=@RezerwId)

**DECLARE** @PlacesSet **int** = (**SELECT** **COUNT**(**id**) **FROM** Uczestnicy **WHERE** RezerwacjaID=@RezerwId)

**IF** (@PlacesSet > @PlacesWanted)

**BEGIN**

**DECLARE** @message **varchar**(100) = 'There is too many participants assigned to this booking. Amount of participants: ' +**CAST**(@PlacesWanted **as** **varchar**(10))+'you can bok'

;THROW 52000,@message,1

**ROLLBACK** **TRANSACTION**

**END**

**END**

**GO**

**Opis procedur :**

**\*** Przy procedurach modyfikujących naturalnie podajemy także @id do danej tabeli.

**[dbo].[DodajAdres] / [dbo].[ZmienAdres] –** Służą do manipulacji na tabelach [dbo].[Adresy], [dbo].[Miasta], [dbo].[Kraje]. Obsługa tych dwóch tabel jest w ten sposób całkowicie odizolowana tak, że mogą pełnić funkcje słownikowe.

**CREATE PROCEDURE [dbo].[DodajAdres]**

@Country **varchar**(50), @Miasto **varchar**(50), @KodPocztowy **varchar**(12), @Ulica **varchar**(35), @NrBud **int**, @NrMiesz **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**DECLARE** @KrajID **int**

**DECLARE** @MiastoID **int**

**DECLARE** @AdresID **int**

**SET** @KrajID=(**select** **id** **from** Kraje **where** Nazwa=@Country)

**if** @KrajID **is** **null**

**begin**

**Insert** **INTO** Kraje (Nazwa) **Values** (@Country)

**SET** @KrajID=@@IDENTITY

**end**

**SET** @MiastoID=(**select** **id** **from** Miasta **where** Miasto=@Miasto **And** [Kod Pocztowy]=@KodPocztowy **and** KrajID=@KrajID)

**if** @MiastoID **is** **null**

**begin**

**Insert** **INTO** Miasta(Miasto,[Kod Pocztowy],KrajID) **Values** (@Miasto,@KodPocztowy,@KrajID)

**SET** @MiastoID=@@IDENTITY

**end**

**SET** @AdresID=(**select** **id** **from** Adresy **where** MiastoID=@MiastoID **and** [Numer budynku]=@NrBud **and** ([Numer mieszkania]=@NrMiesz **or** ([Numer mieszkania] **is** **null** **and** @NrMiesz **is** **null**)) **and** Ulica=@Ulica)

**if** @AdresID **is** **null**

**begin**

**Insert** **INTO** Adresy(MiastoID,[Numer budynku],[Numer mieszkania],Ulica) **Values** (@MiastoID,@NrBud,@NrMiesz,@Ulica)

**end**

**END**

**GO**

**create PROCEDURE [dbo].[ZmienAdres]**

@id **int**, @Country **varchar**(50), @Miasto **varchar**(50),@KodPocztowy **varchar**(12),

@Ulica **varchar**(35), @NrBud **int**, @NrMiesz **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**DECLARE** @KrajID **int**

**DECLARE** @MiastoID **int**

**DECLARE** @AdresID **int**

**SET** @KrajID=(**select** **id** **from** Kraje **where** Nazwa=@Country)

**if** @KrajID **is** **null**

**begin**

**Insert** **INTO** Kraje (Nazwa) **Values** (@Country)

**SET** @KrajID=@@IDENTITY

**end**

**SET** @MiastoID=(**select** **id** **from** Miasta **where** Miasto=@Miasto **And** [Kod Pocztowy]=@KodPocztowy **and** KrajID=@KrajID)

**if** @MiastoID **is** **null**

**begin**

**Insert** **INTO** Miasta(Miasto,[Kod Pocztowy],KrajID) **Values** (@Miasto,@KodPocztowy,@KrajID)

**SET** @MiastoID=@@IDENTITY

**end**

**SET** @AdresID=(**select** **id** **from** Adresy **where** **id** = @id)

**if** @AdresID **is** **not** **null**

**begin**

**update** Adresy

**set** MiastoID = @MiastoID,

[Numer Budynku] = @NrBud,

[Numer mieszkania] = @NrMiesz,

Ulica = @Ulica

**where** @AdresID=**id**

**end**

**END**

**GO**

**[dbo].[DodajOsobe] / [dbo].[ZmienOsobe] *-*** Służą do manipulacji na tabelach [dbo].[Osoby], [dbo].[Adresy], [dbo].[Miasta], [dbo].[Kraje]. Obsługa tych trzech tabel jest obsługiwna poprzez inne procedury.

**CREATE PROCEDURE [dbo].[DodajOsobe]**

@Name **varchar**(50), @LastName **varchar**(50), @Phone **varchar**(12), @DateOfBirth **Date**,

@Sex **bit**, @NrLeg nvarchar(10), @Country **varchar**(50), @Miasto **varchar**(50),

@KodPocztowy **varchar**(12), @Ulica **varchar**(35), @NrBud **int**, @NrMiesz **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**DECLARE** @AdresID **int**

**EXEC** dbo.DodajAdres @Country,@Miasto,@KodPocztowy,@Ulica,@NrBud,@NrMiesz

**SET** @AdresID =(**select** **A**.**id** **from** Adresy **as** **A**

**inner** **join** Miasta **as** M **on** **A**.MiastoID=M.**id**

**inner** **join** Kraje **as** K **on** K.**id**=M.KrajID

**Where** **A**.[Numer budynku]=@NrBud **and** isnull(**A**.[Numer mieszkania],0) = isnull(@NrMiesz,0) **and** **A**.Ulica=@Ulica **and** M.Miasto=@Miasto **And** M.[Kod Pocztowy]=@KodPocztowy **and** K.Nazwa=@Country)

**Insert** **into** Osoby (Imie,Nazwisko,Telefon,[**Data** urodzenia],Plec,[Legitymacja studencka nr],AdresID)**Values**(@Name,@LastName,@Phone,@DateOfBirth,@Sex,@NrLeg,@AdresID)

**END**

**GO**

**create PROCEDURE [dbo].[ZmienOsobe]**

@id **int**, @Name **varchar**(50), @LastName **varchar**(50), @Phone **varchar**(12),

@DateOfBirth **Date**, @Sex **bit**, @NrLeg nvarchar(10), @Country **varchar**(50), @Miasto **varchar**(50),

@KodPocztowy **varchar**(12), @Ulica **varchar**(35), @NrBud **int**, @NrMiesz **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**DECLARE** @AdresID **int**

**EXEC** dbo.DodajAdres @Country,@Miasto,@KodPocztowy,@Ulica,@NrBud,@NrMiesz

--To znajduje adres, ktory dodalismy

**SET** @AdresID =(**select** **A**.**id** **from** Adresy **as** **A**

**inner** **join** Miasta **as** M **on** **A**.MiastoID=M.**id**

**inner** **join** Kraje **as** K **on** K.**id**=M.KrajID

**Where** **A**.[Numer budynku]=@NrBud **and** isnull(**A**.[Numer mieszkania],0) = isnull(@NrMiesz,0) **and** **A**.Ulica=@Ulica **and** M.Miasto=@Miasto **And** M.[Kod Pocztowy]=@KodPocztowy **and** K.Nazwa=@Country)

**update** Osoby

**set** Imie = @Name,

Nazwisko = @LastName,

Telefon=@Phone,

[**Data** urodzenia]=@DateOfBirth,

Plec=@Sex,

[Legitymacja studencka nr]=@NrLeg,

AdresID = @AdresID

**where** **id** = @id

**END**

GO

**[dbo].[DodajKlientaFirm] / [dbo].[ZmienKlientaFirm] *-*** Służą do manipulacji na tabelach [dbo].[Firmy], [dbo].[Klienci], [dbo].[Osoby], [dbo].[Adresy], [dbo].[Miasta], [dbo].[Kraje]. Dodawana jest także osoba reprezentująca oraz adres firmy.

**CREATE** **PROCEDURE** [dbo].[DodajKlientaFirm]

@Name **varchar**(50), @LastName **varchar**(50), @Phone **varchar**(12), @DateOfBirth **Date**,

@Sex **bit**, @NrLeg nvarchar(10), @Country **varchar**(50), @Miasto **varchar**(50),

@KodPocztowy **varchar**(12), @Ulica **varchar**(35), @NrBud **int**, @NrMiesz **int**,

@NazwaFirmy **varchar**(25), @TelFirm **varchar**(12), @Fax **varchar**(12),

@Email **varchar**(50), @Login **varchar**(30), @Password **varchar**(20), @IsCompany **bit**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**DECLARE** @OsobaID **int**

**DECLARE** @AdresID **int**

**DECLARE** @KlientID **int**

**Exec** dbo.DodajOsobe @Name ,@LastName ,@Phone ,@DateOfBirth ,@Sex,@NrLeg,@Country,@Miasto,@KodPocztowy,@Ulica,@NrBud,@NrMiesz

**SET** @OsobaID =(**Select** **Max**(**id**) **from** Osoby)

**Insert** **into** Klienci (Login,**Password**,Email,CzyFirma,OsobaID)**Values**(@Login,@Password,@Email,@IsCompany,@OsobaID)

**SET** @AdresID =(**select** **A**.**id** **from** Adresy **as** **A**

**inner** **join** Miasta **as** M **on** **A**.MiastoID=M.**id**

**inner** **join** Kraje **as** K **on** K.**id**=M.KrajID

**Where** **A**.[Numer budynku]=@NrBud **and** isnull(**A**.[Numer mieszkania],0) = isnull(@NrMiesz,0) **and** **A**.Ulica=@Ulica **and** M.Miasto=@Miasto **And** M.[Kod Pocztowy]=@KodPocztowy **and** K.Nazwa=@Country)

**SET** @KlientID =(**Select** **Max**(**id**) **from** Klienci)

**INSERT** **INTO** Firmy (KlientID,[Nazwa Firmy],[Telefon Kontaktowy],Fax,AdresID)**Values**(@KlientID,@NazwaFirmy,@TelFirm,@Fax,@AdresID)

**END**

**GO**

**[dbo].[DodajKlientaInd] / [dbo].[ZmienKlientaInd] *-*** Służą do manipulacji na tabelach [dbo].[Klienci], [dbo].[Osoby], [dbo].[Adresy], [dbo].[Miasta], [dbo].[Kraje]. Klient dodawany jest także do tabeli [dbo].[Osoby].

**create** **PROCEDURE** [dbo].[ZmienDaneKlientaInd]

@id **int**, --id Klienta

@Name **varchar**(50), @LastName **varchar**(50), @Phone **varchar**(12), @DateOfBirth **Date**,

@Sex **bit**, @NrLeg nvarchar(10), @NazwaFirmy **varchar**(25), @TelFirm **varchar**(12),

@Fax **varchar**(12), @Email **varchar**(50), @Login **varchar**(30), @Password **varchar**(20),

@Country **varchar**(50), @Miasto **varchar**(50), @KodPocztowy **varchar**(12),

@Ulica **varchar**(35), @NrBud **int**, @NrMiesz **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

--Deklaracja zmiennych

**DECLARE** @OsobaID **int**

**DECLARE** @AdresID **int**

**DECLARE** @KlientID **int**

--Sprawdzenie czy to w ogole osoba ind

**if** (**select** CzyFirma **from** Klienci **where** **id** = @id) = 0

**begin**

--Dodanie nowej osoby

**Exec** dbo.DodajOsobe @Name ,@LastName ,@Phone ,@DateOfBirth ,@Sex,@NrLeg,@Country,@Miasto,@KodPocztowy,@Ulica,@NrBud,@NrMiesz

**SET** @OsobaID =(**Select** **id** **from** Osoby **where** @Name = Imie **and** @LastName = Nazwisko **and** Plec = @Sex **and** [Legitymacja studencka nr] = @NrLeg)

--Dodanie nowego adresu

**EXEC** dbo.DodajAdres @Country,@Miasto,@KodPocztowy,@Ulica,@NrBud,@NrMiesz

--Znalezenie nowego adresu

**SET** @AdresID =(**select** **A**.**id** **from** Adresy **as** **A**

**inner** **join** Miasta **as** M **on** **A**.MiastoID=M.**id**

**inner** **join** Kraje **as** K **on** K.**id**=M.KrajID

**Where** **A**.[Numer budynku]=@NrBud **and** isnull(**A**.[Numer mieszkania],0) = isnull(@NrMiesz,0) **and** **A**.Ulica=@Ulica **and** M.Miasto=@Miasto **And** M.[Kod Pocztowy]=@KodPocztowy **and** K.Nazwa=@Country)

--Ustalenie danych w tablicy Klienci

**update** Klienci

**set** Login = @Login,

**Password** = @Password,

Email = @Email,

OsobaID= @OsobaID

**where** **id** = @id

--Koniec

**end**

**END**

**CREATE** **PROCEDURE** [dbo].[DodajKlientaInd]

@Name **varchar**(50), @LastName **varchar**(50), @Phone **varchar**(12),

@DateOfBirth **Date**, @Sex **bit**, @NrLeg nvarchar(10),

@Email **varchar**(50), @Login **varchar**(30), @Password **varchar**(20),

@IsCompany **bit**, @Country **varchar**(50), @Miasto **varchar**(50),

@KodPocztowy **varchar**(12), @Ulica **varchar**(35), @NrBud **int**, @NrMiesz **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**DECLARE** @OsobaID **int**

**Exec** dbo.DodajOsobe @Name ,@LastName ,@Phone ,@DateOfBirth ,@Sex,@NrLeg,@Country,@Miasto,@KodPocztowy,@Ulica,@NrBud,@NrMiesz

**SET** @OsobaID =(**Select** **Max**(**id**) **from** Osoby)

**Insert** **into** Klienci (Login,**Password**,Email,CzyFirma,OsobaID)**Values**(@Login,@Password,@Email,@IsCompany,@OsobaID)

**END**

**GO**

**[dbo].[ZmienHasloKlienta] *-*** Służy do manipulacji na tabeli [dbo].[Klient]. Stare hasło zmieniane jest na nowe. Potrzebna jest znajomość starego hasła.

**create** **PROCEDURE** [dbo].[ZmienHasloKlienta]

@Login **varchar**(30), @OldPassword **varchar**(20), @NewPassword **varchar**(20)

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**if** (**select** **password** **from** Klienci **where** login = @login) = @OldPassword

**begin**

**update** Klienci

**set** **Password** = @NewPassword

**where** (**select** **id** **from** Klienci **where** login = @login) = **id**

--Koniec

**end**

**END**

**GO**

**[dbo].[DodajKonferencje] / [dbo].[ZmienKonferencje] *-*** Służą do manipulacji na tabelach [dbo].[Konferencje]. Klient dodawany jest także do tabeli [dbo].[Osoby].

**CREATE PROCEDURE [dbo].[DodajKonferencje]**

@Nazwa **varchar**(80), @Opis **varchar**(150), @Startdate **Date**,

@EndDate **Date**, @liczbaMiejsc **int**, @Cena money

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**INSERT** **INTO** Konferencje([Nazwa Konferencji],[Opis konferencji],[**Data** Rozpoczecia],[**Data** Zakonczenia],[Liczba miejsc],[Cena za dzien])

**Values**(@Nazwa,@Opis,@Startdate,@EndDate,@liczbaMiejsc,@Cena)

**END**

**GO**

**create PROCEDURE [dbo].[ZmienKonferencje]**

@id **int**, --id procedury

@Nazwa **varchar**(80), @Opis **varchar**(150), @Startdate **Date**,

@EndDate **Date**, @liczbaMiejsc **int**, @Cena money

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**if** (@liczbaMiejsc < (**select** [Liczba miejsc] **from** Konferencje **where** @id = **id**))

**begin**

**update** Konferencje

**set** [Nazwa Konferencji] = @Nazwa,

[Opis konferencji] = @Opis,

[**Data** Rozpoczecia]= @Startdate,

[**Data** Zakonczenia] = @EndDate,

[Liczba miejsc] = @liczbaMiejsc,

[Cena za dzien] = @Cena

**where** @id = **id**

**end**

**END**

**GO**

**[dbo].[DodajDzienKonf] *-*** Służy do manipulacji na tabelach [dbo].[Dni Konferencji].

**CREATE PROCEDURE [dbo].[DodajDzienKonf]**

@KonfID **int**,

@date **Date**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**INSERT** **INTO** [Dni Konferencji] (KonferencjaID,**Data**)**Values**(@KonfID,@date)

**END**

**GO**

**[dbo].[DodajZnizke]  *-*** Służy do manipulacji na tabelach [dbo].[Znizki].

**CREATE PROCEDURE [dbo].[DodajZnizke]**

@KonfID **int**, @Startdate **Date**,

@EndDate **Date**, @zniska **float**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**INSERT** **INTO** Znizki(KonferencjaID,[Obowiazuje od],[Obowiazuje **do**],Znizka)**Values**(@KonfID,@Startdate,@EndDate,@zniska)

**END**

**GO**

**[dbo].[DodajWarsztat] / [dbo].[ZmienWarsztat] *-*** Służą do manipulacji na tabelach [dbo].[Warsztaty].

**CREATE PROCEDURE [dbo].[DodajWarsztat]**

@nazwa **varchar** (80),

@liczbaMiejsc **int**,

@cena money

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**insert** **into** Warsztaty([Nazwa warsztatu],[Liczba miejsc],Cena)**Values**(@nazwa,@liczbaMiejsc,@cena)

**END**

**GO**

**create PROCEDURE [dbo].[ZmienWarsztat]**

@id **int**, @nazwa **varchar** (80), @liczbaMiejsc **int**, @cena money

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**if** (@liczbaMiejsc < (**select** [Liczba miejsc] **from** Warsztaty **where** @id = **id**))

**begin**

**update** Konferencje

**set** [Nazwa Konferencji] = @Nazwa,

[Liczba miejsc] = @liczbaMiejsc,

[Cena za dzien] = @Cena

**where** @id = **id**

**end**

**END**

**GO**

**[dbo].[DodajRezerwacjeKonf] / [dbo].[ZmienRezerwacjeKonf] *-*** Służą do manipulacji na tabelach [dbo].[Rezerwacje konferencji].

**CREATE PROCEDURE [dbo].[DodajRezerwacjeKonf]**

@KlientID **int**, @KonfID **int**, @DataRezerwacji **Date**,

@Studenci **int**, @Normalne **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**insert** **into** [Rezerwacje konferencji] (KlientID,KonferencjaDzienId,[**Data** rezerwacji],Studenci,Normalne,anulowano)**Values**(@KlientID,@KonfID,@DataRezerwacji,@Studenci,@Normalne,0)

**END**

**GO**

**create PROCEDURE [dbo].[ZmienRezerwacjeKonf]**

@KlientID **int**, @KonfID **int**, @DataRezerwacji **Date**,

@Studenci **int**, @Normalne **int**, @anulowano **bit**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**update** [Rezerwacje konferencji] **set** KlientID=@KlientID,KonferencjaDzienId=@KonfID,[**Data** rezerwacji]=@DataRezerwacji,Studenci=@Studenci,Normalne=@Normalne,anulowano=@anulowano

**END**

**GO**

**[dbo].[AnulujRezerwacjeKonferencji]  *-*** Służy do manipulacji na kolumnie *Anulowane* tabeli [dbo].[Rezerwacje konferencji] w celu ustalenia odpowiedniej rezerwacji jako anulowanej.

**create PROCEDURE [dbo].[AnulujRezerwacjeKonferencji]**

@KonfID **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**update** [Rezerwacje konferencji] **set** anulowano=1 **where** **id**=@KonfID

**END**

**GO**

**[dbo].[DodajRezerwacjeWarsztatu] / [dbo].[ZmienRezerwacjeWarsztatu] *-*** Służą do manipulacji na tabelach [dbo].[Rezerwacje warsztatów].

**CREATE PROCEDURE [dbo].[DodajRezerwacjeWarsztatu]**

@WarsztatID **int**, @RezerwacjaKonfID **int**,

@DataRezerwacji **Date**, @LiczbaMiejsc **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**insert** **into** [Rezerwacje warsztatow](WarsztatID,KonferencjaID,[**Data** rezerwacji],[liczba miejsc],anulowano)**Values**(@WarsztatID,@RezerwacjaKonfID,@DataRezerwacji,@LiczbaMiejsc,0)

**END**

**GO**

@WarsztatID **int**, @RezerwacjaKonfID **int**,

@DataRezerwacji **Date**, @LiczbaMiejsc **int**,

@anulowano **bit**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**update** [Rezerwacje warsztatow]**set** WarsztatID=@WarsztatID,KonferencjaID=@RezerwacjaKonfID,[**Data** rezerwacji]=@DataRezerwacji,[liczba miejsc]=@LiczbaMiejsc,anulowano=@anulowano

**END**

**GO**

**[dbo].[AnulujRezerwacjeWarsztatu]  *-*** Służy do manipulacji na kolumnie *Anulowane* tabeli [dbo].[Rezerwacje warsztatów] w celu ustalenia odpowiedniej rezerwacji jako anulowanej.

**create PROCEDURE [dbo].[AnulujRezerwacjeWarsztatu]**

@WarsztatID **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**update** [Rezerwacje warsztatow] **set** anulowano=1 **where** **id**=@WarsztatID

**END**

**GO**

**[dbo].[DodajUczestnikaKonf] *-*** Służy do manipulacji na tabelach [dbo].[Uczestnicy konferencji].

**CREATE PROCEDURE [dbo].[DodajUczestnikaKonf]**

@KlientID **int**, @RezerwacjaKonfID **int**, @OsobaID **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**insert** **into** [Uczestnicy konferencji](KlientID,RezerwacjaKonfID,OsobaID)**Values**(@KlientID,@RezerwacjaKonfID,@OsobaID)

**END**

**GO**

**[dbo].[DodajUczestnika]** (warsztatu) ***-*** Służy do manipulacji na tabelach [dbo].[Uczestnicy].

**CREATE PROCEDURE [dbo].[DodajUczestnika]**

@RezerwacjaWarsztatuID **int**, @UczestnikKonfID **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**insert** **into** Uczestnicy(RezerwacjaID,UczestnikKonfID)**Values**(@RezerwacjaWarsztatuID,@UczestnikKonfID)

**END**

**GO**

**[dbo].[DodajOplate] *-*** Służy do manipulacji na tabelach [dbo].Oplay].

**CREATE PROCEDURE [dbo].[DodajOplate]**

@KonfID **int**, @DataWplaty **Date**, @kwota money

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**insert** **into** Oplaty([Rezerwacja konferencji],[**Data** wplaty],Kwota)**Values**(@KonfID,@DataWplaty,@kwota)

**END**

**GO**

**[dbo].[Podliczkonferencje] *–*** Dla podanego numeru konferencji zwraca wartości wszystkich rezerwacji na daną konferencję, na wszystkie dni.

**create PROCEDURE [dbo].[Podliczkonferencje]**

@konfID **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**Declare** @rezerwacjaID **int**

**Declare** cursor1 **cursor** **for** **select** **id** **from** [Rezerwacje konferencji] **where** KonferencjaDzienId=@konfID

**open** cursor1

**FETCH** **NEXT** **FROM** cursor1

**into** @rezerwacjaID

**WHILE** @@FETCH\_STATUS = 0

**begin**

**EXEC** [dbo].[PodliczRezerwacja] @rezerwacjaID

**FETCH** **NEXT** **FROM** cursor1

**into** @rezerwacjaID

**end**

**close** cursor1;

**DEALLOCATE** cursor1;

**END**

**[dbo].[PodliczRezerwacja] *–*** Dla podanego numeru rezerwacji dnia konferencji zwraca wartość należną.

**create PROCEDURE [dbo].[PodliczRezerwacja]**

@rezerwacjaID **int**

**AS** **BEGIN**

**SET** NOCOUNT **ON**;

**DECLARE** @cenaKonf moneyg**DECLARE** @cenaWarsz money

**Declare** @warsztatID **int**

**Declare** @liczbaOsub **int**

**set** @cenaKonf=(**select** Normalne+Studenci **from** [Rezerwacje konferencji]**where** **id**=@rezerwacjaID)\*(**select** [Cena za dzien] **from** Konferencje **where** **id**=(**select** KonferencjaID **from** [Dni Konferencji] **where** **id**=(**select** KonferencjaDzienId **from** [Rezerwacje konferencji]**where** **id**=@rezerwacjaID)))

**Declare** cursor1 **cursor** **for** **select** WarsztatID,[liczba miejsc] **from** [Rezerwacje warsztatow] **where** KonferencjaID=@rezerwacjaID

**open** cursor1

**FETCH** **NEXT** **FROM** cursor1

**into** @warsztatID,@liczbaOsub

**WHILE** @@FETCH\_STATUS = 0

**begin**

**set** @cenaWarsz=@cenaWarsz+@liczbaOsub\*(**select** Cena **from** Warsztaty)

**FETCH** **NEXT** **FROM** cursor1

**into** @warsztatID,@liczbaOsub

**end**

**close** cursor1;

**DEALLOCATE** cursor1;

**select** @cenaKonf+@cenaWarsz

**END**

**Propozycje przydzielenia widoków i procedur do odpowiednich ról w systemie:**

**[dbo].[BrakujaceZgloszenia] : Administrator, Pracownik**

**[dbo].[KlienciFirmy] : Administrator, Pracownik**

**[dbo].[KlienciOsoby] : Administrator, Pracownik**

**[dbo].[NajcześciejKorzystajacy] : Administrator, Pracownik**

**[dbo].[NajpopularniejszeKonferencje] : Administrator, Pracownik**

**[dbo].[NajpopularniejszeWarsztaty] : Administrator, Pracownik, Klient**

**[dbo].[WolneMiejscaWarsztatow] : Administrator, Pracownik, Klient**

**[dbo].[CenyNaDniKonferencji] : Administrator, Pracownik, Klient**

**[dbo].[IdentyfikatoryKonferencji] : Administrator, Pracownik**

**[dbo].[IdentyfikatoryDniaKonferencji] : Administrator, Pracownik**

**[dbo].[IdentyfikatoryRezerwacjiKonferencji] : Administrator, Pracownik, Klient**

**[dbo].[IdentyfikatoryWarsztatu] : Administrator, Pracownik**

**[dbo].[IdentyfikatoryRezerwacjiWarsztatu] : Administrator, Pracownik, Klient**

**[dbo].[NaCoZapisany] : Administrator, Pracownik, Klient (jego uczestnicy), Uzytkownik (on sam)**

**[dbo].[OsobyNaKonferencji] : Administrator, Pracownik**

**[dbo].[OsobyNaDniuKonferencji] : Administrator, Pracownik**

**[dbo].OsobyNaRezerwacjiKonferencji] : Administrator, Pracownik, Klient**

**[dbo].OsobyNaRezerwacjiWarsztatu] : Administrator, Pracownik, Klient**

**[dbo].OsobyNaWarsztacie]: Administrator, Pracownik**

**[dbo].[UczestnicyOdFirmy] : Administrator, Pracownik, Klient**

**[dbo].[FakturaNaKonf] : Administrator, Pracownik, Klient**

**[dbo].[DodajAdres] : Administrator, Pracownik, Klient**

**[dbo].[ZmienAdres] : Administrator, Pracownik**

**[dbo].[DodajOsobe] : Administrator, Pracownik, Klient**

**[dbo].[DodajKlientaFirm] : Administrator, Pracownik**

**[dbo].[ZmienKlientaFirm] : Administrator, Pracownik**

**[dbo].[DodajKlientaInd] : Administrator, Pracownik**

**[dbo].[ZmienKlientaInd] : Administrator, Pracownik**

**[dbo].[ZmienHasloKlienta] : Administrator, Pracownik, Klient**

**[dbo].[DodajKonferencje] : Administrator, Pracownik**

**[dbo].[ZmienKonferencje] : Administrator, Pracownik**

**[dbo].[DodajDzienKonf] : Administrator, Pracownik**

**[dbo].[DodajZnizke] : Administrator, Pracownik**

**[dbo].[DodajWarsztat] : Administrator, Pracownik**

**[dbo].[ZmienWarsztat] : Administrator, Pracownik**

**[dbo].[DodajRezerwacjeKonf] : Administrator, Pracownik, Klient**

**[dbo].[ZmienRezerwacjeKonf] : Administrator, Pracownik, Klient**

**[dbo].[AnulujRezerwacjeKonferencji] arsztatu] : Administrator, Pracownik, Klient**

**[dbo].[ZmienRezerwacjeWarsztatu] : Administrator, Pracownik, Klient**

**[dbo].[AnulujRezerwacjeWarsztatu] : Administrator, Pracownik, Klient**

**[dbo].[DodajUczestnikaKonf] : Administrator, Pracownik, Klient**

**[dbo].[DodajUczestnika] : Administrator, Pracownik, Klient**

**[dbo].[DodajOplate] : Administrator, Pracownik**

**[dbo].[Podliczkonferencje] :** **Administrator, Pracownik, Klient**

**[dbo].[PodliczRezerwacja] : Administrator, Pracownik, Klient**

**Opis generatora bazy danych:**

Generator został sporządzony w języku Ruby z użyciem biblioteki Faker tworzącej dane teleadresowe. Dodatkowo zebraliśmy zbiór nazw dla warsztatów oraz konferencji obejmujących ponad 300 linii.

Generator zgromadzony był w plikach:

*Address.rb, Clients.rb, Conference.rb, NazwyKonferencji, NazwyWarsztatow Payment.rb, Person.rb, Reservation.rb, Run2.rb, Run.rb, Supporting.rb, Workshop.rb*

Pliki Run.rb oraz Run2.rb będacy udoskonaloną wersją tego pierwszego to pliki rdzenne dla całej aplikacji. Reszta plików to pliki opisujące relacje w bazie danych.

Aplikacja generuje pliki .sql ze składnią zgodną z naszymi procedurami, które można potem wykonać w SQL Studio. Pliki posiadają niekiedy dane z kilku tabel, dzieląc aplikacje na kilka obszarów.

*Klienci.sql, Konferencje.sql, Osoby.sql , RezerwacjeKonf.sql, RezerwacjeWarsz.sql, UczestnicyKonf.sql UczestnicyWarsz.sql Warsztaty.sql*

Aplikacja posiada rozległe możliwości konfiguracyjne:

**Clients.rb**

PASSW\_LENGTH\_MIN = 10

PASSW\_LENGTH\_MAX = 20

**Conference.rb**

#How many days backward can history go

CONFERENCE\_START = 1000

#Seed for length

CONFERENCE\_DAYS\_MAX\_DIFF = 12

CONFERENCE\_DAYS\_BASIC = 4

#Seed for places

PLACES\_ROUNDING = -1

BASIC\_PLACES = 300

VARIABLE\_PLACES = 200

#Seed for price

PRICE\_ROUNDING = -1

BASIC\_PRICE = 10

VARIABLE\_PRICE = "dependant"

MAX\_PRICE = 100

#There are always 3 discount options

DISCOUNT\_DAYS = [60, 120, 180, 360]

DISCOUNT\_DAYS\_MAX\_DIFF = 10

DISCOUNT\_AMMOUNTS = [0.05, 0.10, 0.15]

DISCOUNT\_AMMOUNTS\_MAX\_DIFF = 0.05

**Person.rb**

TELEPHONE\_N\_LENGTH = 9

PERSON\_MIN\_AGE = 18

PERSON\_MAX\_AGE = 65

STUDENT\_QUOTA = 0.1 # How many of the people should be students

STUDENT\_NR\_LENGTH = 6

**Reservation.rb**

*RESERVATION\_BASE\_QUOTA = 0.03*

*RESERVATION\_VARY\_QUOTA = 0.03*

*RESERVATION\_BASE\_QUOTA\_WORKSHOP = 0.2*

*RESERVATION\_VARY\_QUOTA\_WORKSHOP = 0.1*

*# This represent how long ago the reservation had been made*

*RESERVATION\_DATE\_BASE = 14 # In days*

*RESERVATION\_DATE\_VARY = 500 # In days*

*STUDENT\_PROPORTION = 0.1*

**Workshop.rb**

*WORKSHOP\_PLACES\_ROUNDING = -1*

*WORKSHOP\_BASIC\_PLACES = 50*

*WORKSHOP\_VARIABLE\_PLACES = 40*

*WORKSHOP\_PRICE\_ROUNDING = -1*

*WORKSHOP\_BASIC\_PRICE = 20*

*WORKSHOP\_VARIABLE\_PRICE = "dependant"*

*WORKSHOP\_MAX\_PRICE = 200*

*#How many workshop can there be per conference day*

*WORKSHOP\_PER\_CONF\_DAY\_BASE = 2*

*WORKSHOP\_PER\_CONF\_DAY\_MAX\_DIFF = 2*

*#Maximal time worshop can take*

*WORKSHOP\_MAX\_LENGTH = 5*

*#In what time windows can workshops occur*

*WORK\_HOUR\_MIN = Time.new(2000, 01, 01, 8, 0, 0, "+01:00")*

*WORK\_HOUR\_MAX = Time.new(2000, 01, 01, 19, 0, 0, "+01:00")*

*#Maximal and minimal workshop length*

*WORK\_TIME\_MIN = 60\*60 # in second*

*WORK\_TIME\_MAX = 240\*60 # in second*

**Run.rb**

*NR\_PEOPLE = 5000 #default 5000*

*NR\_FIRMS = 400 #default 400*

*NR\_IND = 1000 #default 1000*

*NR\_CONFERENCES = 5 #default 72*

*NR\_WOKSHOPS\_PER\_CON\_BASE = 5 #default 5*

*NR\_WOKSHOPS\_PER\_CON\_VAR = 5 #default 5*

*NR\_RESERVATIONS\_PER\_CON\_FIRMS\_BASE = 10 #default 10*

*NR\_RESERVATIONS\_PER\_CON\_FIRMS\_VAR = 10 #default 10*

*NR\_RESERVATIONS\_PER\_CON\_IND\_BASE = 60#60 #default 30*

*NR\_RESERVATIONS\_PER\_CON\_IND\_VAR = 60 #60 #default 30*

*NR\_RESERVATIONS\_PER\_WORK\_BASE = 60 #default 20*

*NR\_RESERVATIONS\_PER\_WORK\_VAR = 20 #default 10*

*#The ratio by which workshops are attended in a given conference*

*WORKSHOPS\_ATTENDANCE\_BASE = 0.5 #default 0.5*

*WORKSHOPS\_ATTENDANCE\_VARY = 0.5 #default 0.5*