**Projekt bazy danych Biura Podróży**

1. Temat: Baza danych Biura Podróży
2. Projekt wykonali:

Tomasz Łukasik, Dawid Włoszek, Mateusz Szumaczuk.

1. Cel projektu:

Stworzenie bazy danych pozwalającej na sprawne zarządzanie i dopasowywanie oferty do klientów dla konkretnego Biura Podróży.

1. Opis projektu:

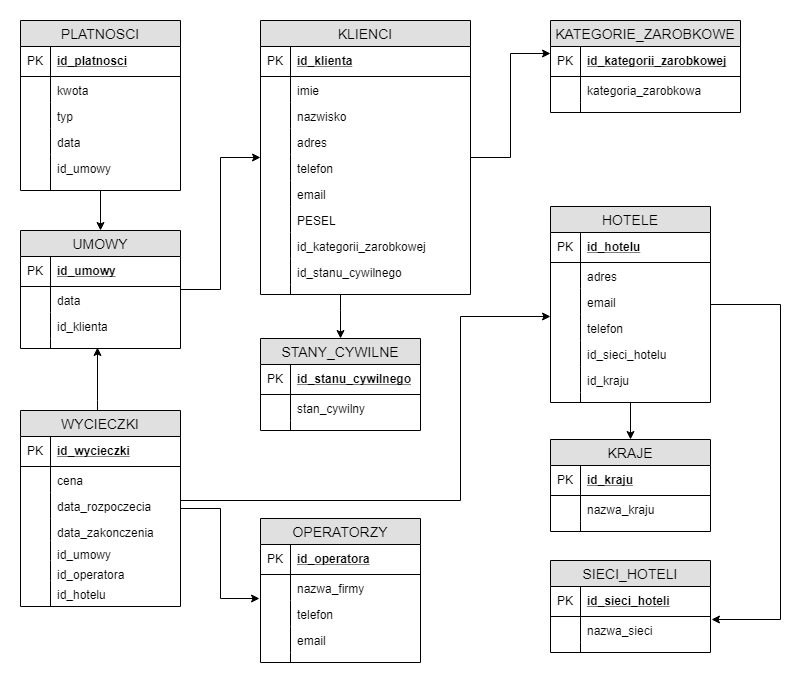
Projekt będzie stanowił główne źródło wiedzy dla Biura Podróży.

Głównym zadaniem biura jest oferowanie i sprzedawanie wycieczek dla potencjalnych klientów. W tym zakresie wymagana jest współpraca z wieloma podmiotami, takimi jak operatorzy wycieczek czy hotele, należące do różnych sieci.

Projekt ma za zadanie stworzenie struktury danych dla takiej firmy, co pozwoli na ich analizę i w konsekwencji lepsze zarządzanie ofertą.

Dlatego została stworzona baza danych z następującymi tabelami:

* Klienci – ich dane, stan cywilny oraz przynależność zarobkowa.
* Stany\_cywilne – tabele z stanami cywilnymi w jakich są klienci.
* Kategorie\_zarobkowe – tabela z kategoriami zarobkowymi, w której klienci klasyfikowani są do jeden z 4 kategorii względem ich dochodów. Zdefiniowane kategorie to : niska, średnia, wysoka i bardzo wysoka.
* Umowy – tabela wiążąca klientów z zakupioną wycieczką.
* Platnosci – informację o dokonanej płatności.
* Wycieczki – oferty wycieczkowe dla klientów.
* Operatorzy – podmioty organizujące wycieczkę.
* Hotele – hotele w których odbywa się zakwaterowanie.
* Kraje – tabela z krajami w których położone są hotele
* Sieci\_hoteli – tabela z sieciami hotelowymi.

1. Wstępny schemat bazy danych:
2. Stworzone widoki:

Głównymi tabelami bazy danych Biura Podróży są:

* wycieczki, jako główny produkt firmy oraz tabela łącząca inne tabele
* klienci, jako grupa do której kierowane są oferty wycieczek
* hotele oraz kraje, jako miejsca atrakcyjne dla klientów
* operatorzy, którzy przeprowadzają wycieczki

Dlatego stworzyliśmy 5 widoków, mających pomóc wyobrażeniu związków między danymi, pokazujących w sposób bardziej zwięzły ważne dane. Utworzone widoki to:

* KLIENCI\_VIEW – wyświetla wszystkie informacje o kliencie wraz z powiązanymi z nim tabelami dotyczącymi do jakiej kategorii zarobkowej należy klient oraz jego stan cywilny.
* HOTELE\_VIEW – wyświetla informacje o hotelu łącząc go z krajem w którym operuje oraz siecią do której należy
* WYCIECZKI\_KRAJE\_VIEW – łączy informacje o wycieczce, takie jak cena, data rozpoczęcia, data zakończenia z krajem do którego obowiązuje.
* WYCIECZKI\_KLIENCI VIEW – łączy informacje o wycieczce, z klientem który ją kupił. Informacje o kliencie wzięte są z widoku KLIENCI\_VIEW.
* WYCIECZKI\_OPERATOROW – łączy informacje o wycieczce, z informacjami o operatorach, którzy ją przeprowadzają.

1. Procedury

W bazie danych zostały stworzone procedury, pozwalające aktualizowanie danych w szybszy sposób. Utworzono następujące procedury:

1. updateKLIENT – procedura odpowiedzialna za aktualizację danych klienta, jeśli opcjonalny parametr nie zostanie przekazany, zostawia poprzednio wpisaną wartość. Parametry jakie przyjmuje:

* @id – główny parametr po którym następuje aktualizacja.
* @imie – opcjonalny.
* @nazwisko – opcjonalny.
* @adres– opcjonalny.
* @PESEL– opcjonalny.
* @telefon – opcjonalny.
* @email – opcjonalny.
* @nr\_dowodu – opcjonalny.
* @id\_kategorii\_zarobkowej – opcjonalny.
* @id\_stanu\_cywilnego – opcjonalny.

1. updateHOTEL – procedura odpowiedzialna za aktualizację danych hotelu, jeśli opcjonalny parametr nie zostanie przekazany, zostawia poprzednio wpisaną wartość. Parametry jakie przyjmuje:

* @id – główny parametr po którym następuje aktualizacja.
* @adres - opcjonalny.
* @email - opcjonalny.
* @telefon - opcjonalny.
* @id\_kraju - opcjonalny.
* @id\_sieci\_hotelu - opcjonalny.

1. updateWYCIECZKA – procedura odpowiedzialna za aktualizację danych wycieczki, jeśli opcjonalny parametr nie zostanie przekazany, zostawia poprzednio wpisaną wartość. Parametry jakie przyjmuje:

* @id – główny parametr po którym następuje aktualizacja.
* @cena – opcjonalny.
* @data\_rozpoczecia – opcjonalny.
* @data\_zakonczenia – opcjonalny.
* @id\_operatora – opcjonalny.
* @id\_umowy – opcjonalny.
* @id\_hotelu – opcjonalny.

1. updateOPERATORA – procedura odpowiedzialna za aktualizację danych operatora, jeśli opcjonalny parametr nie zostanie przekazany, zostawia poprzednio wpisaną wartość. Parametry jakie przyjmuje:

* @id – główny parametr po którym następuje aktualizacja.
* @nazwa\_firmy – opcjonalny.
* @telefon – opcjonalny.
* @email – opcjonalny.

1. updateSIECI\_HOTELI – procedura odpowiedzialna za aktualizację danych sieci hoteli, jeśli opcjonalny parametr nie zostanie przekazany, zostawia poprzednio wpisaną wartość. Parametry jakie przyjmuje:

* @id – główny parametr po którym następuje aktualizacja.
* @nazwa\_sieci – opcjonalny.

1. updateKRAJ – procedura odpowiedzialna za aktualizację danych kraju, jeśli opcjonalny parametr nie zostanie przekazany, zostawia poprzednio wpisaną wartość. Parametry jakie przyjmuje:

* @id – główny parametr po którym następuje aktualizacja.
* @nazwa\_kraju – opcjonalny.

1. Funkcje

W projekcie stworzono funkcje pozwalające na otrzymanie wybranych danych, w zależności od podanych parametrów.

* 1. wycieczkiOperatorowCenaWiekszaOd

Przyjmuje parametr @cena, a zwraca tabelę z wycieczkami organizowanymi przez operatorów, których cena wycieczek jest większa od podanego parametru. Zwraca tabela z kolumnami:

nazwa\_firmy, telefon, email, cena, liczbaDni (jako liczbą dni wycieczki)

* 1. wycieczkiOperatorowCzasTrwania

Przyjmuje parametr @liczbaDni, a zwraca tabelę z wycieczkami organizowanymi przez operatorów, których czas trwania wynosił więcej lub równo podaną liczbę dni.

* 1. przychodOperatora

Przyjmuje parametr @id, jako id operatora dla którego zwracana jest łączna suma cen wycieczek, które prowadził wybrany operator.

* 1. liczbaSprzedanychWycieczekOperatora

Przyjmuje parametr @id, jako id operatora dla którego zwracana jest liczba wszystkich wycieczek, które prowadził wybrany operator.

* 1. liczbaKupionychWycieczekKlienta

Przyjmuje parametr @id, jako id klienta dla którego zwracana jest liczba wszystkich wycieczek, które zakupił dany klient.

1. Audyt

Na potrzeby monitorowania bazy danych i wszystkich zmian wiążących się z głównymi tabelami, została stworzona tabela audytowa o nazwie – AuditTabele, z polami:

* AuditID – auto inkrementujący się klucz tabeli.
* TableName – nazwa tabeli w której zachodzi zmiana.
* ChangedID – id wiersza, którego dotyczy zmiana.
* EventType – rodzaj zmiany.
* LoginName – login osoby dokonującej zmiany.
* AuditDateTime – data wykonanej zmiany.

Poprzez utworzenie na głównych tabelach wyzwalaczy (trigger), zmiany dokonywane na rekordach tych tabel zostają wysyłane do tabeli audytowej, która przechowuje tą informację. Lista tabel dla których utworzono wyzwalacze:

* WYCIECZKI
* HOTELE
* SIECI\_HOTELI
* OPERATORZY
* KLIENCI
* UMOWY

Zostały utworzone po 3 wyzwalacze na każdą tabelę, monitorujące różne rodzaje zmian i odkładające je do tabeli audytowej. Rodzaje zmian jakie zostały obsłużone, to:

* INSERT – wstawienie rekordu.
* UPDATE – aktualizacja rekordu.
* DELETED – usunięcie rekordu.

1. Indeksy

W celu optymalizacji zapytań do bazy danych zostały stworzone indeksy niezgrupowane. Natomiast indeksy zgrupowane zostały utworzone w momencie deklaracji kolumny klucza głównego ID dla każdej z tabel.

Poniżej zostały zawarte indeksy niezgrupowane, które zostały stworzone w bazie danych biura podróży:

* IX\_Klienci\_imie\_nazwisko – indeks został stworzony w tabeli Klienci na kolumnach imię i nazwisko, ponieważ użytkownik bazy danych w zapytaniach do tej tabeli będzie zazwyczaj posługiwał się imieniem i nazwiskiem w celu odnalezieniu konkretnego klienta.
* IX\_Operatorzy\_nazwa\_firmy - indeks został stworzony w tabeli Operatorzy na kolumnie nazwy firmy, ponieważ użytkownik bazy danych w zapytaniach do tej tabeli będzie domyślnie poszukiwał Operatora za pomocą nazwy firmy.
* IX\_Wycieczki\_data – indeks został stworzony w tabeli Wycieczki na kolumnie data tak, aby zapytania kierowane do tabeli Wycieczki, gdzie w klauzuli WHERE wskazana jest data zostały zoptymalizowane.
* IX\_Umowy\_data - indeks został stworzony w tabeli Umowy na kolumnie data tak, aby zapytania kierowane do tabeli Umowy, gdzie w klauzuli WHERE wskazana jest data zostały zoptymalizowane.
* IX\_Platnosci\_data - indeks został stworzony w tabeli Platnosci na kolumnie data tak, aby zapytania, które mają na celu uzyskanie płatności z danego dnia były zoptymalizowane.