Dokumentacja bazy danych i relacji między danymi

Założenia projektu

Cel projektu

Celem projektu jest stowrzenie aplikacji desktopowej do zarządania wycieczkami w firmie zajmujacej się turystyką rowerową. Fianlny produkt ma zapenwniać spójność danych oraz usprawnienie sposobu obsługi klientów i dokonywania rezerwacji w hotelach.

Wymagania użytkownika oraz funkcje

Program pozwala ustawić wycieczki o podanej trasie wraz z wyborem miast przez które przechodzi dana wycieczke. Można dodawać hotele dla danych miast i tworzyć listy hoteli dla klientów, a wpływ dla danej wycieczki liczony jest automatycznie.

Tabele

---

1. typy_wycieczek

Typy dostępnych wycieczek

Nazwa kolumny	Typ danych	Ograniczenie	Opis
typ	VARCHAR(3)	PRIMARY KEY	3-literowy kod typu wycieczki
liczba_nocy	INTEGER	NOT NULL	Liczba noclegów w ramach danego typu
ceny	INTEGER	FOREIGN KEY	Klucz obcy do tabeli `ceny`

---

2. ceny

Ceny na każdej wycieczce

Nazwa kolumny	Typ danych	Ograniczenie	Opis
id	INTEGER	PRIMARY KEY	Unikalny identyfikator rekordu
typ_wycieczki	VARCHAR(3)	FOREIGN KEY	Klucz obcy do tabeli `typy_wycieczek`
pok_1	NUMERIC(10, 2)	NOT NULL	Cena za 1-osobowy pokój od osoby
pok_2	NUMERIC(10, 2)	NOT NULL	Cena za 2-osobowy pokój od osoby
pok_3	NUMERIC(10, 2)	NOT NULL	Cena za 3-osobowy pokój od osoby
pok_4	NUMERIC(10, 2)	NOT NULL	Cena za 4-osobowy pokój od osoby
ulga_dziecko	INTEGER		Ulga dla dzieci jako wartosc w liczniku dzielenia przez 100
rower	NUMERIC(10, 2)		Koszt wynajmu roweru
e_bike	NUMERIC(10, 2)		Koszt wynajmu roweru elektrycznego
dodatkowa_noc	NUMERIC(10, 2)		Koszt dodatkowego noclegu
hb	NUMERIC(10, 2)		Cena opcji “half-board” (śniadanie + kolacja)

3. miasta

Miasta gdzie są noclegi wycieczki

Nazwa kolumny	Typ danych	Ograniczenie	Opis
miasto	VARCHAR(100)	PRIMARY KEY	Nazwa miasta

4. miasta_wycieczek

Łączy typ wycieczki z miastem oraz numerem nocy w tej wycieczce

Nazwa kolumny	Typ danych	Ograniczenie	Opis
id	INTEGER	PRIMARY KEY	Unikalny identyfikator miasta wycieczki
typ_wycieczki	VARCHAR(3)	FOREIGN KEY	Typ wycieczki z tabeli `typy_wycieczek`
miasto	VARCHAR(100)	FOREIGN KEY	Miasto z tabeli `miasta`
nr_nocy	INTEGER		Nr nocy tej wycieczki która jest w tym mieści

5. wycieczki

Wyceiczki jako poszczególne grupy z datami początku i końca oraz innymi informacjami

Nazwa kolumny	Typ danych	Ograniczenie	Opis
wycieczka	VARCHAR(5)	PRIMARY KEY	Unikalny kod wycieczki jako 3 kodu wyceiczki i numer wycieczki (np. VAR05)
typ_wycieczki	VARCHAR(3)	FOREIGN KEY	Typ wycieczki z tabeli `typy_wycieczek`
początek	DATE	NOT NULL	Data rozpoczęcia wycieczki
koniec	DATE	NOT NULL	Data zakończenia wycieczki
il_uczestnikow	INTEGER		Liczba uczestników wycieczki
wpływ	MONEY		Całkowity wpływ z wycieczki

6. hotele

Wsyzstkie hotele z których można korzystać na trasach

Nazwa kolumny	Typ danych	Ograniczenie	Opis
kod	VARCHAR(6)	PRIMARY KEY	Kod hotelu jako unikalne 6 znkaów
nazwa	VARCHAR(100)	NOT NULL	Nazwa hotelu
miasto	VARCHAR(100)	FOREIGN KEY	Miasto, w którym znajduje się hotel
adres	VARCHAR(100)		Adres hotelu
mail	VARCHAR(100)		Adres e-mail hotelu
nr_tel	VARCHAR(13)		Numer telefonu do hotelu

7. listy_hoteli

Lista hoteli dla danego pokoju z danej wycieczki

Nazwa kolumny	Typ danych	Ograniczenie	Opis
pokój	INTEGER	FOREIGN KEY	Pokój z tabeli `pokoje`
miasto_wycieczki	INTEGER	FOREIGN KEY	Miasto z tabeli `miasta_wycieczek`
hotel	VARCHAR(6)	FOREIGN KEY	Hotel z tabeli `hotele`

8. pokoje

Łączy pokój z wycieczką i daje możliwość przypisania do niego uczestników

Nazwa kolumny	Typ danych	Ograniczenie	Opis
id	INTEGER	PRIMARY KEY	Unikalny identyfikator pokoju
wycieczka	VARCHAR(5)	FOREIGN KEY	Wycieczka z tabeli `wycieczki`
typ_pokoju	VARCHAR(3)	NOT NULL	Typ pokoju (np. 1-os., 2-os.)
il_klientow	INTEGER		obecna ilość osób w pokoju
il_miejsc	INTEGER		pojemność pokoku
czy_lista_hoteli	BOOLEAN		prawda jeśli dla pokoju jest ustawiona lista hoteli

9. klienci

Klienci wyceiczek z wszystkimi potrzebnymi informacjami

Nazwa kolumny	Typ danych	Ograniczenie	Opis
id	INTEGER	PRIMARY KEY	Unikalny identyfikator klienta
imię	VARCHAR(100)	NOT NULL	Imię klienta
nazwisko	VARCHAR(100)	NOT NULL	Nazwisko klienta
wycieczka	VARCHAR(5)	FOREIGN KEY	Wycieczka, w której bierze udział
typ_pokoju	VARCHAR(3)	NOT NULL	Typ pokoju przydzielony klientowi
pokój	INTEGER	FOREIGN KEY	Pokój przypisany klientowi z tabeli `pokoje`
ulga	BOOLEAN		Czy klient korzysta z ulgi?
rower	BOOLEAN		Czy klient wypożycza rower?
e_bike	BOOLEAN		Czy klient wypożycza e-bike?
nocleg_przed	BOOLEAN		Czy klient ma nocleg przed wycieczką?
nocleg_po	BOOLEAN		Czy klient ma nocleg po wycieczce?
hb	BOOLEAN		Czy klient wybrał opcję half-board?
do_zaplaty	numeric(10, 2)		Koszt dla danego klienta

Triggery dla dodawania klientów

Poniżej znajdują się funkcje, które aktualizują ilość klientów danej wycieczki oraz ilość osób w danym pokoju po dodaniu lub usunięciu klienta.

```
CREATE OR REPLACE FUNCTION update_pokoje_on_add()
RETURNS TRIGGER AS $$
BEGIN
UPDATE pokoje
SET il_klientow = il_klientow + 1
WHERE id = NEW.pokoj;

            RETURN NEW;
        END;
    $$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION update_wycieczki_on_add()
RETURNS TRIGGER AS $$
BEGIN
UPDATE wycieczki
SET il_uczestnikow = il_uczestnikow + 1
WHERE wycieczka = NEW.wycieczka;

            RETURN NEW;
        end;
    $$ LANGUAGE plpgsql;


CREATE OR REPLACE FUNCTION update_pokoje_on_delete()
RETURNS TRIGGER AS $$
BEGIN
UPDATE pokoje
SET il_klientow = il_klientow - 1
WHERE id = OLD.pokoj;

    RETURN OLD;
END;
$$ LANGUAGE plpgsql;


CREATE OR REPLACE FUNCTION update_wycieczki_on_delete()
RETURNS TRIGGER AS $$
BEGIN
UPDATE wycieczki
SET il_uczestnikow = il_uczestnikow - 1
WHERE wycieczka = OLD.wycieczka;

    RETURN OLD;
end;
$$ LANGUAGE plpgsql;
```

Tu mamy dodawanie powuższych funkcji do trigerów

```
CREATE TRIGGER trigger_on_add_klinet_pokoje
    AFTER INSERT
    ON klienci
    FOR EACH ROW
EXECUTE FUNCTION update_pokoje_on_add();


CREATE TRIGGER trigger_on_add_klinet_wycieczki
    AFTER INSERT
    ON klienci
    FOR EACH ROW
EXECUTE FUNCTION update_wycieczki_on_add();


CREATE TRIGGER trigger_on_delete_klinet_pokoje
    AFTER DELETE
    ON klienci
    FOR EACH ROW
EXECUTE FUNCTION update_pokoje_on_delete();


CREATE TRIGGER trigger_on_delete_klinet_wycieczki
    AFTER DELETE
    ON klienci
    FOR EACH ROW
EXECUTE FUNCTION update_wycieczki_on_delete();
```

Poniżej mamy to samo, ale aktualizujemy wpływ z danej wycieczki po dodaniu lub usunięciu klienta

```
CREATE OR REPLACE FUNCTION update_ceny_on_add()
RETURNS TRIGGER AS $$
    BEGIN
        UPDATE wycieczki
        SET wplyw = wplyw + NEW.do_zaplaty
        WHERE wycieczka = NEW.wycieczka;

        RETURN NEW;
    end;
    $$ LANGUAGE plpgsql;


CREATE OR REPLACE FUNCTION update_ceny_on_delete()
    RETURNS TRIGGER AS $$
BEGIN
    UPDATE wycieczki
    SET wplyw = wplyw - OLD.do_zaplaty
    WHERE wycieczka = OLD.wycieczka;

    RETURN OLD;
end;
$$ LANGUAGE plpgsql;


CREATE TRIGGER on_add_klient_ceny AFTER INSERT
    ON klienci FOR EACH ROW
    EXECUTE FUNCTION update_ceny_on_add();


CREATE TRIGGER on_delete_klient_ceny AFTER DELETE
    ON klienci FOR EACH ROW
EXECUTE FUNCTION update_ceny_on_delete();
```

Widok zwracający dane potrzebne do listy hoteli

```
CREATE OR REPLACE VIEW v_lista_nocy_hoteli AS
SELECT
    mw.nr_nocy AS noc,
    ( wy.poczatek + (mw.nr_nocy - 1) * INTERVAL '1 day' ) AS data,
    m.miasto AS miasto,
    h.nazwa AS hotel
FROM listy_hoteli lh
         JOIN pokoje p
              ON lh.pokoj = p.id
         JOIN wycieczki wy
              ON p.wycieczka = wy.wycieczka
         JOIN miasta_wycieczek mw
              ON lh.miasto_wycieczki = mw.id
         JOIN miasta m
              ON mw.miasto = m.miasto
         JOIN hotele h
              ON lh.hotel = h.kod;
```

Użyta technologia i opis użytkowania

Aplikacja jest napisana w całości w języku JAVA. Po stronie frontendu wykorzystano Javafx, a po stronie backendu użyto framework SprinBoot. Aplikacja podłącza się do bazy danych i pozawala na wykonywania operacji na tej bazie. Okna składa się z 3 części, głównego panelu oraz lewgo i prawego paska z przyciskami. Na lewym panelu wybieramy co chcemy wyświetlić i w zależności od wybranego elemntu pojawią się przyciski na prawym pasku zależene od wybranego przycisku z lewego paska. Aby dodać klienta należy mieć już dodane wolne pokoje dla wycieczki na którą jedzie klient. Ze względu na docelowy sposób użytkowania dane dodawane są tylko ręcznie porzez dostarczony interfejs. Aplikacja jest zabezpieczona przed większością możliwych pomyłek ze strony użytkownika.