Baza danych – biuro podróży

Sprawozdanie

Prowadzący: dr. Inż. Jarosław Koszela

Grupa: WCY19IY1S1

Wykonał: Damian Subzda 75157

2. Spis treści:

3. Treść zadania:

Zaproponować i zaimplementować implementacje bazy danych na wybrany przez siebie temat. Wymagania:

- Minimum 10 tabel
- Minimum 3 widoki (w tym jeden zmaterializowany)
- Minimum 3 funkcje wbudowane
- Minimum 3 procedury wbudowane
- Minimum 3 wyzwalacze
- Mechanizm kursora i transakcji w jednej funkcji i jednej procedurze
- Minimum 3 użytkowników o różnych uprawnieniach

4. Opis dotyczący bazy danych:

Do implementacji posłużył Sybase Central oraz projekt konceptualny jak i fizyczny został zaprojektowany w PowerDesigner.

Baza przechowuje dane dotyczące zamówień, które są realizowane przez klientów. Klient może zamówić więcej niż jedno zamówienie ale zamówienie musi mieć jednego klienta.

Każde zamówienie musi mieć co najmniej jednego uczestnika i uczestnik jeśli jest w bazie musi mieć przypisane zamówienie.

Do encji zamówienia została połączona encja zakwaterowania która może mieć jeden lub więcej zamówień ale zamówienie musi mieć co najmniej jedno zakwaterowanie.

Zamówienie musi mieć wybraną wycieczkę, a wycieczka może być w więcej niż jednym zamówieniu.

Każda wycieczka może mieć maksymalnie jednego przewodnika a przewodnik może mieć wiele wycieczek oraz każda wycieczka musi mieć transport, a transport może mieć jedną lub więcej wycieczek.

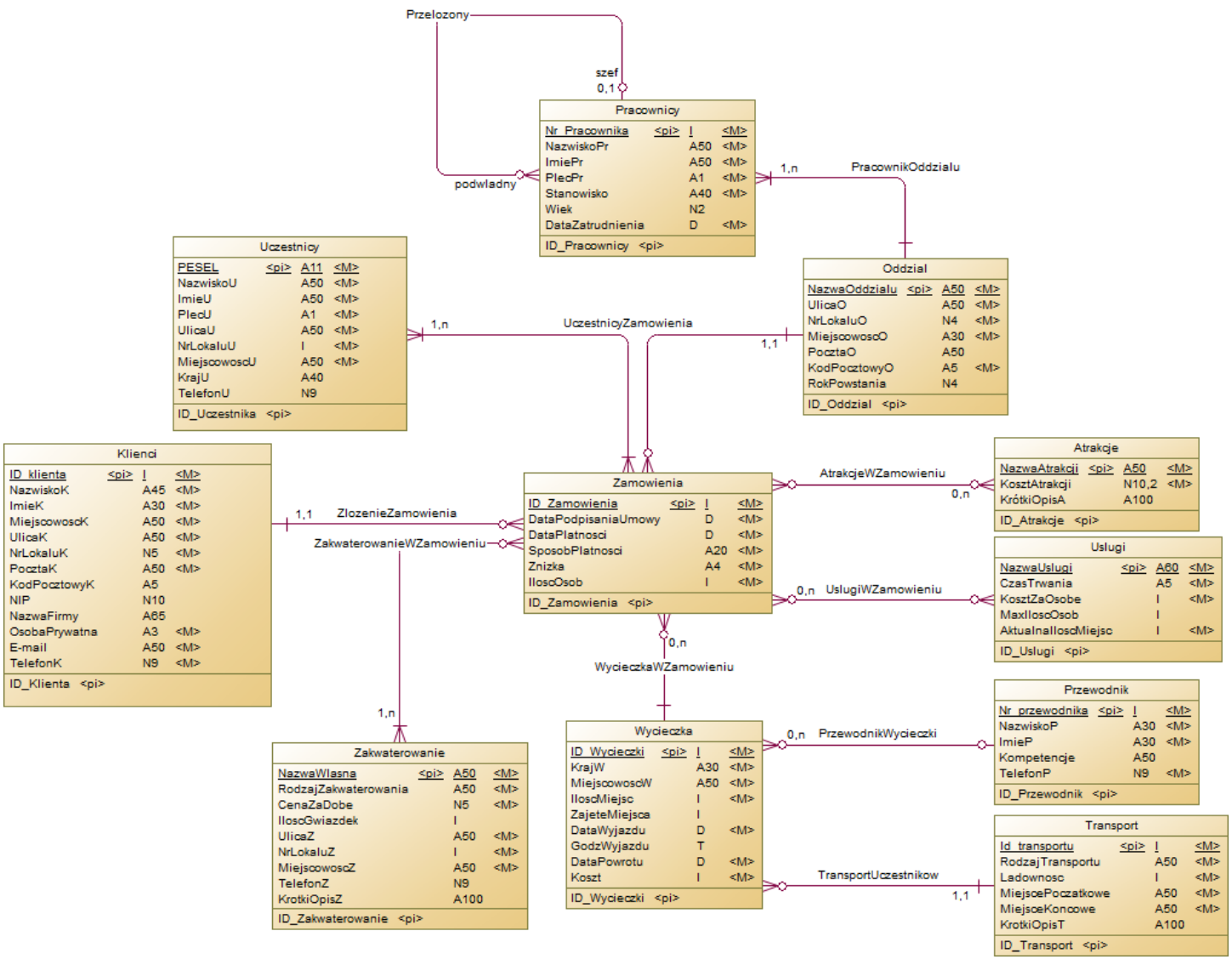
Zamówienie posiada również dowiązanie do atrakcji i usług gdzie w obu przypadkach jest to wiele zamówień może mieć wiele atrakcji/usług.

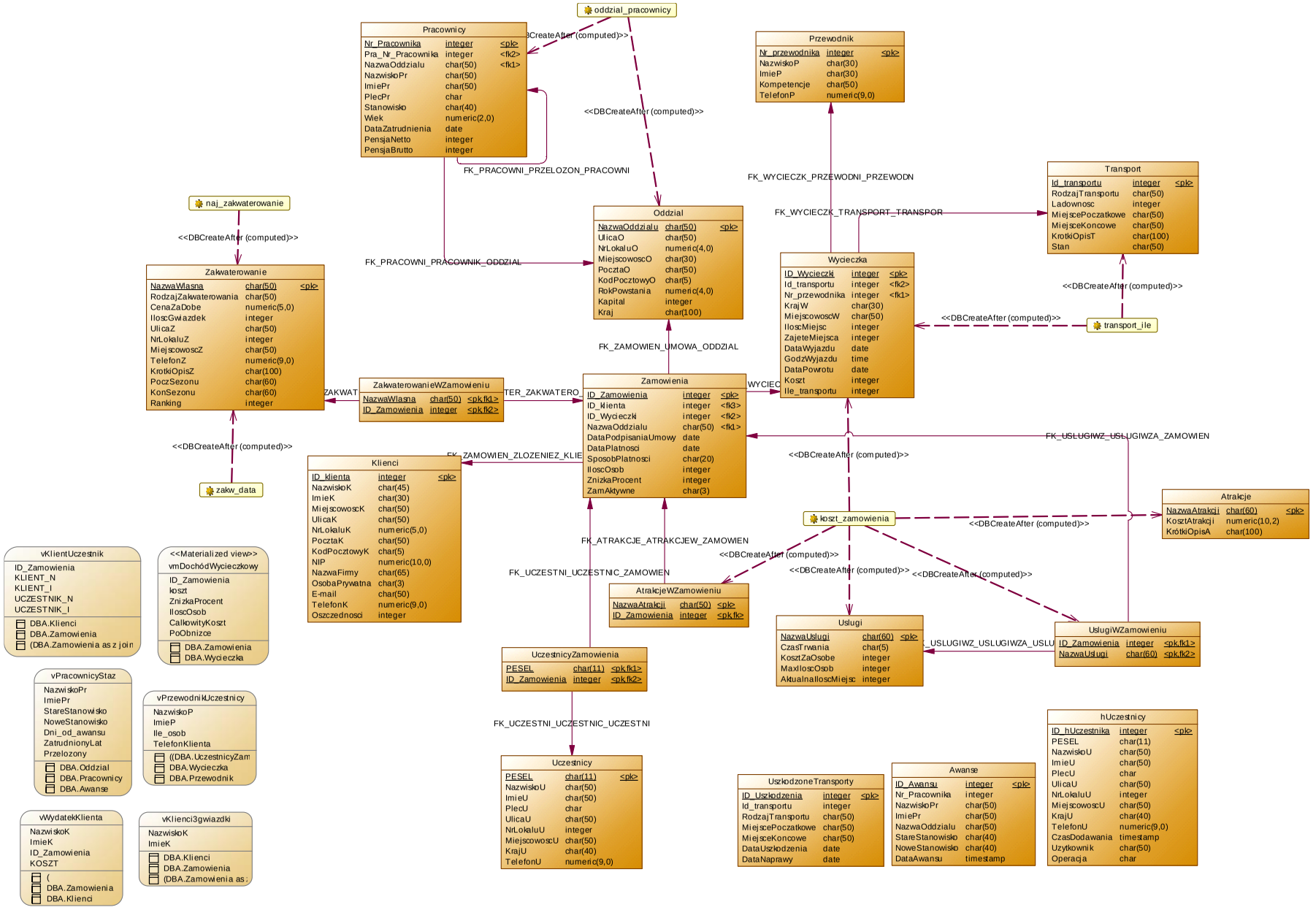
Zamówienie jest realizowane w danym oddziale oraz oddział ma możliwość realizacji wielu zamówień.

Oddział posiada pracowników, którzy są w relacji jeden oddział posiada jednego bądź wielu pracowników. Każdy pracownik musi być dopisany do jednego oddziału.

Pracownicy posiadają przełożonego, czyli jest to dowiązanie się encji pracownicy z samą sobą i jeśli pracownik jest szefem (przełożonym) to w miejscu przełożonego ma NULL.

5. Model konceptualny:




oddzial_pracownicy
Pracownicy
Nr_Pracownika integer <pk>
Pra_Nr_Pracownika integer <fk2>
NazwaOddzialu char(50) <fk1>
NazwiskoPr char(50)
ImiePr char(50)
PlecPr char
Stanowisko char(40)
Wiek numeric(2,0)
DataZatrudnienia date
PensjaNetto integer
PensjaBrutto integer
<<DBCreateAfter (computed)>>
FK_PRACOWNI_PRZELOZON_PRACOWNI
Przewodnik
Nr_przewodnika integer <pk>
NazwiskoP char(30)
ImieP char(30)
Kompetencje char(50)
TelefonP numeric(9,0)
FK_WYCIECZK_PRZEWODNI_PRZEWODN
Transport
Id_transportu integer <pk>
RodzajTransportu char(50)
Ladownosc integer
MiejscePoczatkowe char(50)
MiejsceKoncowe char(50)
KrotkiOpisT char(100)
Stan char(50)
FK_WYCIECZK_TRANSPORT_TRANSPOR
naj_zakwaterowanie
<<DBCreateAfter (computed)>>
Oddzial
NazwaOddzialu char(50) <pk>
UlicaO char(50)
NrLokaluO numeric(4,0)
MiejscowoscO char(30)
PosztaO char(50)
KodPocztowyO char(5)
RokPowstania numeric(4,0)
Kapital integer
Kraj char(100)
FK_PRACOWNI_PRACOWNIK_ODDZIAL
Wycieczka
ID_Wycieczki integer <pk>
Id_transportu integer <fk2>
Nr_przewodnika integer <fk1>
KrajW char(30)
MiejscowoscW char(50)
IloscMiejsc integer
ZajeteMiejsca integer
DataWyjazdu date
GodzWyjazdu time
DataPowrotu date
Koszt integer
Ile_transportu integer
<<DBCreateAfter (computed)>>
transport_ile
Zakwaterowanie
NazwaWlasna char(50) <pk>
RodzajZakwaterowania char(50)
CenaZaDobe numeric(5,0)
IloscGwiazdek integer
UlicaZ char(50)
NrLokaluZ integer
MiejscowoscZ char(50)
TelefonZ numeric(9,0)
KrotkiOpisZ char(100)
PoczSezonu char(60)
KonSezonu char(60)
Ranking integer
ZakwaterowanieWZamowieniu
NazwaWlasna char(50) <pk,fk1>
ID_Zamowienia integer <pk,fk2>
Zamowienia
ID_Zamowienia integer <pk>
ID_klienta integer <fk3>
ID_Wycieczki integer <fk2>
NazwaOddzialu char(50) <fk1>
DataPodpisaniaUmowy date
DataPlatnosci date
SposobPlatnosci char(20)
IloscOsob integer
ZnizkaProcent integer
ZamAktywne char(3)
FK_ZAMOWIEN_UMOWA_ODDZIAL
FK_USLUGIWZ_USLUGIWZA_ZAMOWIEN
<<DBCreateAfter (computed)>>
<<DBCreateAfter (computed)>>
zakw_data
Klienci
ID_klienta integer <pk>
NazwiskoK char(45)
ImieK char(30)
MiejscowoscK char(50)
UlicaK char(50)
NrLokaluK numeric(5,0)
Poczta K char(50)
KodPocztowyK char(5)
NIP numeric(10,0)
NazwaFirmy char(65)
OsobaPrywatna char(3)
E-mail char(50)
TelefonK numeric(9,0)
Oszczednosci integer
FK_ATRAKCJE_ATRAKCJEW_ZAMOWIEN
FK_UCZESTNI_UCZESTNIC_ZAMOWIEN
koszt_zamowienia
Atrakcje
NazwaAtrakcji char(60) <pk>
KosztAtrakcji numeric(10,2)
KrótkiOpisA char(100)
<<DBCreateAfter (computed)>>
AtrakcjeWZamowieniu
NazwaAtrakcji char(50) <pk>
ID_Zamowienia integer <pk,fk>
Uslugi
NazwaUslugi char(60) <pk>
CzasTrwania char(5)
KosztZaOsobe integer
MaxIloscOsob integer
AktualnaIloscMiejsc integer
UslugiWZamowieniu
ID_Zamowienia integer <pk,fk1>
NazwaUslugi char(60) <pk,fk2>
UczestnicyZamowienia
PESEL char(11) <pk,fk1>
ID_Zamowienia integer <pk,fk2>
FK_UCZESTNI_UCZESTNIC_UCZESTNI
Uczestnicy
PESEL char(11) <pk>
NazwiskoU char(50)
ImieU char(50)
PlecU char
UlicaU char(50)
NrLokaluU integer
MiejscowoscU char(50)
KrajU char(40)
TelefonU numeric(9,0)
UszkodzoneTransporty
ID_Uszkodzenia integer <pk>
Id_transportu integer
RodzajTransportu char(50)
MiejscePoczatkowe char(50)
MiejsceKoncowe char(50)
DataUszkodzenia date
DataNaprawy date
Awanse
ID_Awansu integer <pk>
Nr_Pracownika integer
NazwiskoPr char(50)
ImiePr char(50)
NazwaOddzialu char(50)
StareStanowisko char(40)
NoweStanowisko char(40)
DataAwansu timestamp
hUczestnicy
ID_hUczestnika integer <pk>
PESEL char(11)
NazwiskoU char(50)
ImieU char(50)
PlecU char
UlicaU char(50)
NrLokaluU integer
MiejscowoscU char(50)
KrajU char(40)
TelefonU numeric(9,0)
CzasDodawania timestamp
Uzytkownik char(50)
Operacja char
vKlientUczestnik
ID_Zamowienia
KLIENT_N
KLIENT_I
UCZESTNIK_N
UCZESTNIK_I
DBA.Klienci
DBA.Zamowienia
(DBA.Zamowienia as z join
<<Materialized view>>
vmDochódWycieczkowy
ID_Zamowienia
koszt
ZnizkaProcent
IloscOsob
CalkowityKoszt
PoObnizce
DBA.Zamowienia
DBA.Wycieczka
vPracownicyStaz
NazwiskoPr
ImiePr
StareStanowisko
NoweStanowisko
Dni_od_awansu
ZatrudnionyLat
Przelozony
DBA.Oddzial
DBA.Pracownicy
DBA.Awanse
vPrzewodnikUczestnicy
NazwiskoP
ImieP
Ile_osob
TelefonKlienta
((DBA.UczestnicyZam
DBA.Wycieczka
DBA.Przewodnik
vWydatekKlienta
NazwiskoK
ImieK
ID_Zamowienia
KOSZT
(
DBA.Zamowienia
DBA.Klienci
vKlienci3gwiazdki
NazwiskoK
ImieK
DBA.Klienci
DBA.Zamowienia
(DBA.Zamowienia as

7. Widoki:

7.1. **vPrzewodnikUczestnicy**

Widok przedstawia nazwisko i imię przewodnika wycieczek, oraz liczbę osób jaką posiada podczas danej wycieczki oraz numer do klienta czyli osoby, która zamówiła konkretną wycieczkę.

Kod SQL:

```
ALTER VIEW "DBA"."vPrzewodnikUczestnicy"
as
select P.NazwiskoP, P.ImieP, count(UZ.Pesel) Ile_osob, K.TelefonK
TelefonKlienta
from ((((UczestnicyZamowienia UZ right join Zamowienia Z on
UZ.ID_Zamowienia = Z.ID_Zamowienia)
        left join Klienci K on Z.ID_klienta = K.ID_klienta)
        left join Wycieczka W on Z.ID_Wycieczki = W.ID_Wycieczki)
        right join Przewodnik P on w.Nr_przewodnika =
P.Nr_przewodnika)
where datediff(day, now(), W.DataWyjazdu) < 31
group by P.NazwiskoP, P.ImieP, K.TelefonK
having Ile_osob > 0
order by P.NazwiskoP, P.ImieP, Ile_osob desc
```

Do wywołania widoku użyjemy poniższego zapytania:

```
select * from vPrzewodnikUczestnicy
```

Uzyskany wynik:

Results

	NazwiskoP	ImieP	Ile_osob	TelefonKlienta
1	Maruk	Marian	2	234 937 302
2	Maruk	Marian	2	123 456 789

7.2. **vPracownicyStaz**

Widok przedstawia nazwisko i imię pracownika oraz jego stare i nowe stanowisko. Dni od awansu, ile lat jest zatrudniony oraz jego przełożonego. Jeśli stare stanowisko to ‘szef’ to nie wyświetlamy pracownika oraz jeśli nie awansował to także nie wyświetlamy.

Kod SQL:

```
ALTER VIEW "DBA"."vPracownicyStaz"
as
select p.NazwiskoPr, p.ImiePr, StareStanowisko, NoweStanowisko,
datediff(day, DataAwansu ,now()) Dni_od_awansu,
datediff(year, p.DataZatrudnienia, now()) ZatrudnionyLat,
        case
            when (select NazwiskoPr
                  from Pracownicy
                  where Nr_Pracownika = p.Pra_Nr_Pracownika) IS NULL
            then 'brak'
            else (select NazwiskoPr
                  from Pracownicy
                  where Nr_Pracownika = p.Pra_Nr_Pracownika)
            end as Przelozony
from (Oddzial o inner join Pracownicy p on o.NazwaOddzialu =
p.NazwaOddzialu), Awanse
where p.NazwiskoPr = Awanse.NazwiskoPr AND p.ImiePr = Awanse.ImiePr
      AND Dni_od_awansu is not null AND StareStanowisko not in ('szef')
order by PlecPr, p.NazwiskoPr, p.ImiePr, Dni_od_awansu desc
```

Do wywołania widoku użyjemy poniższego zapytania:

```
select * from vPracownicyStaz
```

Uzyskany wynik:

Results

	NazwiskoPr	ImiePr	StareStanowisko	NoweStanowisko	Dni_od_awansu	ZatrudnionyLat	Przelozony
1	Wieczorek	Weronika	konsultant	kierownik	39	1	Iksiński
2	Majeranek	Klaudia	konsultant	kierownik	2	2	Iksiński
3	Majeranek	Klaudia	kierownik	szef	2	2	Iksiński

7.3. **vmDochódWycieczkowy**

Widok zmaterializowany przedstawia dochód biura podróży z poszczególnej wycieczki. W zależności od ilości osób oraz koszcie wycieczki naliczane są dodatkowe zniżki grupowe dla klientów. Wliczane są wyłącznie zamówienia, które zastały już zakończone.

Kod SQL:

```
CREATE MATERIALIZED VIEW "DBA"."vmDochódWycieczkowy"
as
select Z.ID_Zamowienia, W.koszt ,Z.ZnizkaProcent, z.IloscOsob,
Z.IloscOsob*w.Koszt CalkowityKoszt,
case
    when W.koszt >100
    then
    (
    case
        when Z.IloscOsob >=5
        then Z.IloscOsob*w.Koszt*(100-z.znizkaProcent-2)/100
        when Z.IloscOsob >=10
        then Z.IloscOsob*w.Koszt*(100-z.znizkaProcent-5)/100
        else Z.IloscOsob*w.Koszt*(100-z.znizkaProcent)/100
        end
    )
    ELSE Z.IloscOsob*w.Koszt*(100-z.znizkaProcent)/100
    end as PoObnizce
from (Zamowienia Z left join Wycieczka W on Z.ID_Wycieczki =
W.ID_Wycieczki)
where ZamAktywne = 'Nie'
order by z.ID_Zamowienia
```

Do wywołania widoku użyjemy poniższego zapytania:

```
select * from vmDochódWycieczkowy
```

Uzyskany wynik:

Results

	ID_Zamowienia	koszt	ZnizkaProcent	IloscOsob	CalkowityKoszt	PoObnizce
1	1	999	5	3	2 997	2 847
2	2	999	7	5	4 995	4 545

8. Funkcje:

8.1. **koszt_zamowienia**

Funkcja ta dla podanego ID_Zamowienia wylicza łączny jego koszt obciążający klienta (koszt atrakcji, usług, wycieczki) i zwraca wartość.

Kod SQL:

```
ALTER FUNCTION "DBA"."koszt_zamowienia"( IN zamowienieNr int )
RETURNS INT
DETERMINISTIC
BEGIN
       DECLARE "wartosc" INT;

    set wartosc = (select sum(a.KosztAtrakcji)
                   from ((Zamowienia z left join AtrakcjeWZamowieniu awz on
                          z.ID_Zamowienia = awz.ID_Zamowienia)
                   left join Atrakcje a on awz.NazwaAtrakcji = a.NazwaAtrakcji)
                   where z.ID_Zamowienia = zamowienieNr
                   group by z.ID_Zamowienia);
    set wartosc = wartosc + (select Sum(u.KosztZaOsobe*z.IloscOsob)
                             from ((Zamowienia z left join UslugiWZamowieniu uwz on
                                    z.ID_Zamowienia = uwz.ID_Zamowienia)
                             left join Uslugi u on uwz.NazwaUslugi = u.NazwaUslugi)
                             where z.ID_Zamowienia = zamowienieNr
                             group by z.ID_Zamowienia
                             order by z.ID_Zamowienia);
    set wartosc = wartosc + (select w.Koszt
                             from (Zamowienia z inner join Wycieczka w on
                                    z.ID_Wycieczki = w.ID_Wycieczki)
                             where z.ID_Zamowienia = zamowienieNr);
       RETURN "wartosc";
END
```

Do wywołania funkcji użyjemy poniższego zapytania:

```
select koszt_zamowienia(1)
```

Uzyskany wynik:

	koszt_zamowienia(1)
1	2 220

8.2. **oddzial_pracownicy**

Funkcja zwraca ilość miesięcy jaką dany oddział jest w stanie zatrudniać wszystkich pracowników posiadając aktualny kapitał, oraz uwzględniając stopy procentowe zależne od danego kraju. Funkcja uaktualnia pensje brutto pracowników. Kursor zlicza wymogi netto pracowników dla danego oddziału. W przypadku niepowodzenia czyli wpisania złej nazwy oddziału wykonuje się ROLLBACK aby przywrócić aktualizowanie rekordów pracowników.

Kod SQL:

```
ALTER FUNCTION "DBA"."oddzial_pracownicy"( IN NazwaOdd char(100) )
RETURNS INTEGER
NOT DETERMINISTIC
BEGIN
    DECLARE "miesiecy" INTEGER;
    DECLARE pieniadze INTEGER;
    DECLARE wymogiPr float;

    set miesiecy = 0;
    set pieniadze = 0;
    set pieniadze = (select Kapital from Oddzial where NazwaOddzialu  = NazwaOdd);
    set wymogiPr = 0;
    SAVEPOINT svp1;
    FOR petla as kursor CURSOR FOR
        SELECT PensjaNetto
        FROM Pracownicy
        where NazwaOddzialu = NazwaOdd
    DO
        set wymogiPr = wymogiPr + PensjaNetto;
    END FOR;

    IF (select Kraj from Oddzial where NazwaOddzialu = NazwaOdd) = 'Polska'
        THEN
            set wymogiPr = wymogiPr*123/100;
            update Pracownicy
            set PensjaBrutto = PensjaNetto *123/100
            where NazwaOddzialu in (select NazwaOddzialu
                                    from Oddzial
                                    where Kraj = 'Polska');
        ELSEIF (select Kraj
                from Oddzial
                where NazwaOddzialu = NazwaOdd) = 'Niemcy'
        THEN
            set wymogiPr = wymogiPr*119/100;
            update Pracownicy
            set PensjaBrutto = PensjaNetto *119/100
            where NazwaOddzialu in (select NazwaOddzialu
                                    from Oddzial
                                    where Kraj = 'Niemcy');
        ELSE
            set wymogiPr = wymogiPr*110/100;
            update Pracownicy
            set PensjaBrutto = PensjaNetto *123/100
            where NazwaOddzialu not in (select NazwaOddzialu
```

```
                                        from Oddzial
                                        where Kraj = 'Polska' OR Kraj = 'Niemcy');
        ENDIF;
    IF pieniadze IS NULL
        THEN
            set miesiecy = -999;
    ROLLBACK to savepoint svp1;
        ELSE
            WHILE pieniadze > 0
            loop
                set pieniadze = pieniadze - wymogiPr;
                set miesiecy = miesiecy +1;
            END loop;
    ENDIF;
    set miesiecy = miesiecy - 1;
      RETURN "miesiecy";
END
```

Do wywołania funkcji użyjemy poniższego zapytania:

```
select oddzial_pracownicy('Biuro Podróży Itaka oddział Warszawski')
```

Uzyskany wynik:

Results

	oddzial_pracownicy('Biuro Podróży Itaka oddział Warszawski')
1	3

8.3. **zakw_data**

Funkcja zwracająca randomową nazwę zakwaterowania w zależności od daty przyjazdu. Wpisujemy datę w jakiej chcemy się zakwaterować jako argument funkcji i sprawdzany jest dostęp do zakwaterowania (czy a tym czasie dany obiekt jest otwarty).

Kod SQL:

```
ALTER FUNCTION "DBA"."zakw_data"( IN dataPrzyjazdu date)
RETURNS CHAR(100)
NOT DETERMINISTIC
BEGIN
      DECLARE "nazwa" CHAR(100);
      DECLARE "max_data" date;
      IF month(dataPrzyjazdu) > 1
    then
```

```
        IF month(dataPrzyjazdu) > 2
        then
            IF month(dataPrzyjazdu) > 3
            then
                IF month(dataPrzyjazdu) > 4
                then
                    IF month(dataPrzyjazdu) > 5
                    then
                        IF month(dataPrzyjazdu) > 6
                        then
                            IF month(dataPrzyjazdu) > 7
                            then
                                IF month(dataPrzyjazdu) > 8
                                then
                                    IF month(dataPrzyjazdu) > 9
                                    then
                                        IF month(dataPrzyjazdu) > 10
                                        then
                                            IF month(dataPrzyjazdu) > 11
                                            then
                                            set nazwa = (SELECT top 1
NazwaWlasna
                                                        FROM Zakwaterowanie
                                                        WHERE PoczSezonu =
'gr%'
                                                        order by rand());
                                            ELSE
                                            set nazwa = (SELECT top 1
NazwaWlasna
                                                        FROM Zakwaterowanie
                                                        WHERE PoczSezonu IN
('gr%','lis%')
                                                        order by rand());
                                            endif;
                                        ELSE
                                        set nazwa = (SELECT top 1 NazwaWlasna
                                                    FROM Zakwaterowanie
                                                    WHERE PoczSezonu IN
('gr%','lis%','p%')
                                                    order by rand());
                                        endif;
                                    ELSE
                                    set nazwa = (SELECT top 1 NazwaWlasna
                                                FROM Zakwaterowanie
                                                WHERE PoczSezonu IN
('gr%','lis%','p%','w%')
                                                order by rand());
                                    endif;
                                ELSE
                                set nazwa = (SELECT top 1 NazwaWlasna
                                            FROM Zakwaterowanie
                                            WHERE PoczSezonu IN
('gr%','lis%','p%','w%','si%')
                                            order by rand());
                                endif;
                            ELSE
                            set nazwa = (SELECT top 1 NazwaWlasna
                                        FROM Zakwaterowanie
```

```
                                        WHERE PoczSezonu IN
('gr%','li%','p%','w%','si%')
                                        order by rand());
                            endif;
                        ELSE
                        set nazwa = (SELECT top 1 NazwaWlasna
                                     FROM Zakwaterowanie
                                     WHERE PoczSezonu IN
('gr%','li%','p%','w%','si%','c%')
                                     order by rand());
                        endif;
                    ELSE
                    set nazwa = (SELECT top 1 NazwaWlasna
                                  FROM Zakwaterowanie
                                  WHERE PoczSezonu IN
('gr%','li%','p%','w%','si%','c%','m%')
                                  order by rand());
                    endif;
                ELSE
                set nazwa = (SELECT top 1 NazwaWlasna
                              FROM Zakwaterowanie
                              WHERE PoczSezonu IN
('gr%','li%','p%','w%','si%','c%','m%','k%')
                              order by rand());
                endif;
            ELSE
            set nazwa = (SELECT top 1 NazwaWlasna
                          FROM Zakwaterowanie
                          WHERE PoczSezonu IN
('gr%','li%','p%','w%','si%','c%','m%','k%','m%')
                          order by rand());
            endif;
        ELSE
        set nazwa = (SELECT top 1 NazwaWlasna
                      FROM Zakwaterowanie
                      WHERE PoczSezonu IN
('gr%','l%','p%','w%','si%','c%','m%','k%')
                      order by rand());
        endif;
    ELSE
    set nazwa = (SELECT top 1 NazwaWlasna
                  FROM Zakwaterowanie
                  order by rand());
    ENDIF;
       RETURN "nazwa";
END
```

Do wywołania funkcji użyjemy poniższego zapytania:

```
select zakw_data('2020-01-01')
```

Uzyskany wynik:

Results

	zakw_data('2020-01-01')
1	Hotel u Marka

9. Triggery:

9.1. **trg_pracownik_awans**

Trigger jest wyzwalany przy aktualizacji pracowników. Jeśli zostało zmienione stanowisko to zapisuj je do tabeli ‘Awanse’, a jeśli nie to nie rób nic.

Kod SQL:

```
ALTER TRIGGER "trg_pracownik_awans" AFTER UPDATE
ORDER 1 ON "DBA"."Pracownicy"
REFERENCING OLD AS old_n NEW AS new_n
FOR EACH ROW WHEN( new_n.Stanowisko<>old_n.Stanowisko )
BEGIN
     insert into Awanse(Nr_Pracownika, NazwiskoPr, ImiePr,
                NazwaOddzialu,           StareStanowisko,
NoweStanowisko, DataAwansu)
    values(new_n.Nr_Pracownika,new_n.NazwiskoPr,
new_n.ImiePr,
                new_n.NazwaOddzialu,    old_n.Stanowisko,
new_n.Stanowisko,now())
END
```

Do wywołania triggera użyjemy poniższego zapytania:

```
update Pracownicy
set Stanowisko = 'kierownik'
where Nr_Pracownika = 13
```

Oraz

```
select * from Awanse
```

Uzyskany wynik:

5	5	13	Gwiazda	Julia	Biuro Podróży Itaka oddział Niemiecki	konsultant	kierownik	2021-02-10 11:23:14.392

9.2. **trg_Uczestnicy_M**

Trigger zapisuje do tabeli ‘hUczestnicy’ uczestników którzy zostali usunięci, dodani albo zaktualizowani.

Kod SQL:

```
ALTER TRIGGER "trg_Uczestnicy_M" AFTER INSERT, DELETE, UPDATE
ORDER 1 ON "DBA"."Uczestnicy"
REFERENCING OLD AS old_n NEW AS new_n
FOR EACH ROW
BEGIN
    declare pOperacja char(1);
    set pOperacja = 'I';
    IF DELETING then
        set pOeracja = 'D';
    ELSEIF UPDATING then
        set pOperacja = 'U';
    endif;
     insert into hUczestnicy(PESEL, NazwiskoU, ImieU,PlecU, UlicaU,
NrLokaluU,
                MiejscowoscU,KrajU,TelefonU,
CzasDodawania,Uzytkownik,Operacja)
    values(new_n.PESEL,  new_n.NazwiskoU,  new_n.ImieU,new_n.PlecU,
new_n.UlicaU, new_n.NrLokaluU,
                new_n.MiejscowoscU,new_n.KrajU,new_n.TelefonU,
now(),connection_property('userID'), pOperacja)
END
```

Do wywołania triggera użyjemy poniższego zapytania:

```
update Uczestnicy
set NrLokaluU = 22
where NrLokaluU = 21
```

Oraz

```
select * from hUczestnicy
```

Uzyskany wynik:

2	4	12345678901	Icicki	Wojciech	M	Sosnowa	22	Warszawa	Polska	123 456 789	2021-02-10 11:35:10.247	DBA	U

9.3. **trg_uszkodzonyTransport**

Trigger przy aktualizowaniu transportu sprawdza jego stan, jeśli zmienił się na ‘uszkodzony’, wtedy wyznacza długość naprawy jaka jest potrzebna w zależności od wielkości (ładowności). Wynik wpisywany jest do tabeli ‘UszkodzoneTransporty’.

Kod SQL:

```
ALTER TRIGGER "trg_uszkodzonyTransport" AFTER UPDATE
ORDER 1 ON "DBA"."Transport"
REFERENCING OLD AS old_n NEW AS new_n
FOR EACH ROW WHEN( (new_n.Stan = 'uszkodzony')OR(new_n.Stan =
'Uszkodzony') )
BEGIN
DECLARE naprawa INTEGER;
set naprawa = 5;
    IF (old_n.Ladownosc BETWEEN 10 AND 100) THEN
       set naprawa = 15;
    ELSEIF (old_n.Ladownosc BETWEEN 101 AND 1000) Then
       set naprawa = 25;
    ELSEIF (old_n.Ladownosc > 1000) then
       set naprawa = 40;
    endif;
    insert UszkodzoneTransporty(Id_transportu, RodzajTransportu,
MiejscePoczatkowe,
           MiejsceKoncowe,DataUszkodzenia ,DataNaprawy)
    values      (old_n.Id_transportu,      old_n.RodzajTransportu,
old_n.MiejscePoczatkowe, old_n.MiejsceKoncowe,
           getdate(), getdate() + naprawa)
END
```

Do wywołania triggera użyjemy poniższego zapytania:

```
update Transport
set Stan = 'uszkodzony'
where Id_transportu = 3
```

Oraz

```
select * from UszkodzoneTransporty
```

Uzyskany wynik:

2	6	9	Autobus	Augustowo	Gdańsk	2021-01-27	2021-02-11

10. Procedury:

10.1. **naj_zakwaterowanie()**

Procedura dodająca do tabeli ‘zakwaterowania’ w kolumnie ‘ranking’ wartości kolejno numerowane od 1 do n mające na celu wyznaczyć który obiekt ma najbardziej atrakcyjną ofertę bazując na koszcie dobowym i gwiazdkach.

Kod SQL:

```
ALTER PROCEDURE "DBA"."naj_zakwaterowanie"()
BEGIN
    DECLARE position INT;
    set position = 0;
    savepoint mysv;
    FOR petla as kursor CURSOR FOR
        select NazwaWlasna a, CenaZaDobe, IloscGwiazdek,
        case
        when IloscGwiazdek = 0 then CenaZaDobe/0.5
        else CenaZaDobe/IloscGwiazdek
        END as aRanking
        from Zakwaterowanie
        order by aRanking
    DO
        set position = position + 1;
        update Zakwaterowanie set Ranking = position
        where NazwaWlasna = a;
    END FOR;
    IF (select count(NazwaWlasna) from Zakwaterowanie) = 0
    Then
    ROLLBACK to savepoint mysv;
    ENDIF;
END
```

Do wywołania procedury posłużymy się komendą:

```
call naj_zakwaterowanie()
```

Uzyskany wynik:

CenaZaDobe	IloscGwiazdek	UlicaZ	NrLokaluZ	MiejscowoscZ	TelefonZ	KrotkiOpisZ	PoczSezonu	KonSezonu	Ranking
75	0	Sloneczna	12	Otwock	923 123 123	*(NULL)*	lipiec	pazdziernik	3
180	4	Kwiecista	1	Warszawa	923 121 323	Widok na Stadion Narodowy z okna	czerwiec	wrzesien	1
120	1	Akacjowa	21	Kalisz	213 932 123	*(NULL)*	marzec	grudzien	2

10.2. **transport_ile()**

Procedura wyliczająca ile transportów danego rodzaju należy użyć aby przemieścić ludzi zapisanych na wycieczkę. Dana ta zostaje wpisana do tablicy ‘Wycieczka’ do kolumny ‘Ile_transportu’.

Kod SQL:

```
ALTER PROCEDURE "DBA"."transport_ile"()
BEGIN
    FOR petla as kursor CURSOR FOR
        SELECT ID_Wycieczki idw, Id_transportu  t, ZajeteMiejsca z
        FROM Wycieczka
    DO
        update Wycieczka set Ile_transportu = ceiling(z/(select Ladownosc
                                                from Transport
                                                where Id_transportu = t)+1)
        where ID_Wycieczki = idw
    END FOR;
END
```

Do wywołania procedury posłużymy się komendą:

```
call transport_ile()
```

Uzyskany wynik:

	Ile_transportu
1	1
2	1
3	1
4	1
5	1
6	1

11. Użytkownicy:

- DBmanager – ma dostęp do wszystkich elementów bazy danych, może w pełni administrować bazą tworzyć obiekty, wykonywać back-up'y oraz wszystkie możliwe operacje na bazie.
- Szef – wpisywanie, usuwanie, z bazy danych oraz wykonywanie analizy dziennika żądań.
- Uzytkownik – odczytywanie danych z bazy danych