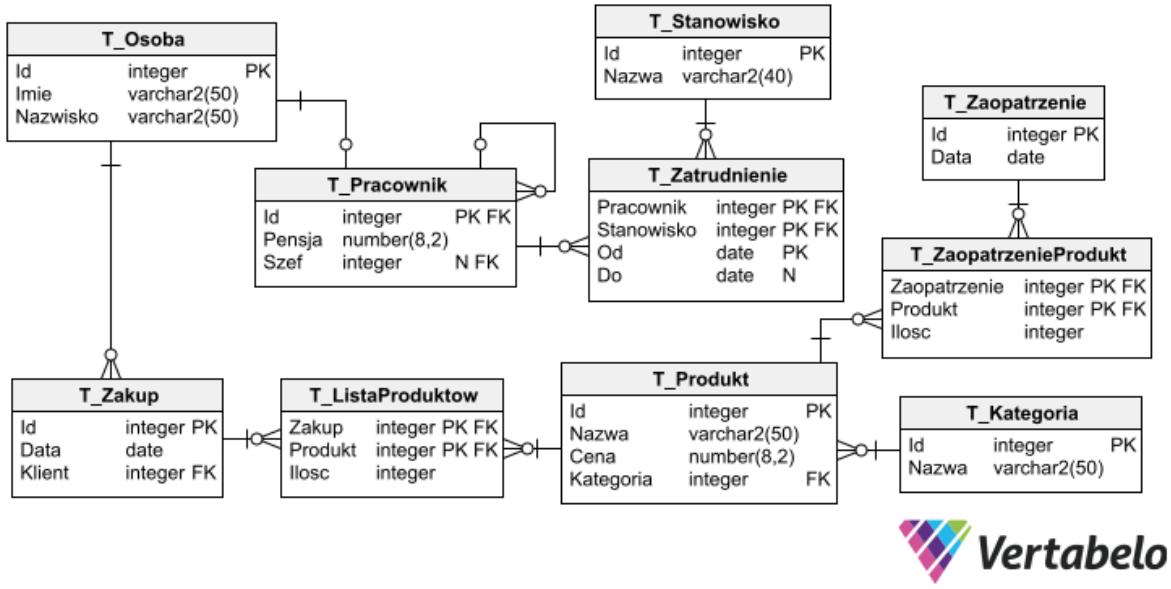
SBD Lab12
T-SQL Wyzwalacze

Link do wygenerowania poniższej bazy danych: link.
Link do skryptu drop: link.



Zad 1

Napisz prosty wyzwalacz, który nie pozwoli nam na usunięcie rekordów z tabeli *T_ListaProduktow*. Wypisz błąd „*Nie można usuwać rekordów z tabeli T_ListaProduktow*” poprzez *raise_application_error*.

Przed utworzeniem wyzwalacza przetestuj usuwanie rekordu, zadziała:

```
DELETE FROM T_ListaProduktow
WHERE zakup = 55 AND produkt = 4;
```

Następnie, po utworzeniu wyzwalacza, spróbuj usunąć inny rekord:

```
DELETE FROM T_ListaProduktow
WHERE zakup = 55 AND produkt = 5;
```

Oczekiwany wynik:

```
[72000][20001]
ORA-20001: Nie można usuwać rekordów z tabeli T_ListaProduktow
ORA-06512: przy "PEUCESTAS.TR_LISTAPRODUKTOW_DELETE", linia 2
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'PEUCESTAS.TR_LISTAPRODUKTOW_DELETE'
Position: 12
```

Zad 2

Przekształć wyzwalacz z zadania 1 tak, aby dodatkowo wypisywał informację: *„Usuwanie rekordu dla zakupu= {IdZakupu} i produktu= {IdProduktu} nie powiodło się”*. Wykorzystaj w tym celu odwołanie *:OLD* (aby zadziałało wyzwalacz musi być wierszowy).

Po zaktualizowaniu wyzwalacza spróbuj jeszcze raz usunąć rekord z tabeli:

```
DELETE FROM T_ListaProduktow
WHERE zakup = 55 AND produkt = 5;
```

Oczekiwany wynik:

```
[72000][20001]
ORA-20001: Nie można usuwać rekordów z tabeli T_ListaProduktow.
Usuwanie rekordu dla zakupu= 55 i produktu= 5 nie powiodło się
ORA-06512: przy "PEUCESTAS.TR_LISTAPRODUKTOW_DELETE", linia 2
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'PEUCESTAS.TR_LISTAPRODUKTOW_DELE ...
```

Zad 3

Usuń wyzwalacz z zadania 1 i 2, a następnie napisz wyzwalacz *AFTER DELETE dla tabeli T_Zakup*, który będzie usuwać wszystkie rekordy z *T_ListaProduktow* powiązane z usuwanym zakupem. Dodatkowo chcemy wypisywać informację: *„Usunięto zakup o id = {id}”*.

Przed utworzeniem wyzwalacza, spróbuj usunąć zakup o id = 1;

```
DELETE FROM T_Zakup WHERE id = 1;
```

Oczekiwany wynik:

```
[23000][2292] ORA-02292: naruszono więzy spójności (PEUCESTAS.T_LISTAPRODUKTOW_T_ZAKUP) - znaleziono rekord podrzędny
Position: 0
```

Po utworzeniu wyzwalacza, spróbuj jeszcze raz usunąć zakup o id = 1;

```
DELETE FROM T_Zakup WHERE id = 1;
```

Oczekiwany wynik:

```
[2024-01-08 16:54:22] 1 row affected in 27 ms
Usunięto zakup o id= 1
```

Zad 4

Utwórz wyzwalacz *BEFORE*, który przy wstawianiu lub modyfikowaniu danych w tabeli *T_Pracownik* sprawdzi czy nowe zarobki (wstawiane lub modyfikowane) są mniejsze niż 10 000. Jeśli warunek nie zostanie spełniony wyzwalacz powinien zgłosić błąd poprzez *raise_application_error* i nie dopuścić do wstawienia rekordu.

Uwaga: W tym zadaniu używamy wyzwalacza jedynie w celach treningowych, ponieważ taką funkcjonalność najlepiej byłoby zrealizować poprzez założenie więzów spójności typu *CHECK* na kolumnę pensja w następujący sposób:

```
ALTER TABLE T_Pracownik
ADD CONSTRAINT CHK_Pensja CHECK (Pensja < 10000);
```

Po utworzeniu wyzwalacza spróbuj zmodyfikować pensję pracownika na większą niż 10 000:

```
UPDATE T_Pracownik
SET pensja = 40000
WHERE id =2;
```

Oczekiwany wynik:

```
[72000][20001]
ORA-20001: Pensja za duża, operacja DML nie powiodła się
ORA-06512: przy "PEUCESTAS.T_PRACOWNIK_BIUR", linia 2
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'PEUCESTAS.T_PRACOWNIK_BIUR'
Position: 7
```

Po utworzeniu wyzwalacza spróbuj dodać nowego pracownika z pensją większą niż 10 000:

```
INSERT INTO T_Pracownik(Id, pensja) VALUES (7, 40000)
```

Oczekiwany wynik:

```
[72000][20001]
ORA-20001: Pensja za duża, operacja DML nie powiodła się
ORA-06512: przy "PEUCESTAS.T_PRACOWNIK_BIUR", linia 2
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'PEUCESTAS.T_PRACOWNIK_BIUR'
Position: 7
```

Zad 5

Napisz wyzwalacz *BEFORE UPDATE OF cena*, który nie pozwoli na obniżenie ceny produktu. Przy próbie zmniejszenia ceny podnosimy błąd: „*Nie można zmniejszać ceny”.*
Po utworzeniu wyzwalacza spróbuj zmniejszyć cenę jednego z produktów:

```
UPDATE T_Produkt
SET cena = 0.01
WHERE id = 2;
```

Oczekiwany wynik:

```
[72000][20001]
ORA-20001: Nie mozna zmniejszać ceny
ORA-06512: przy "PEUCESTAS.TR_PRODUKT_BUR", linia 2
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'PEUCESTAS.TR_PRODUKT_BUR'
Position: 7
```

Zad 6

Usuń wyzwalacz z zadania 5, a następnie napisz jeden wyzwalacz dla T_Produkt, który:
- nie pozwoli na dodanie produktu z ceną większą niż 100 (w *INSERT*)
- nie pozwoli na zwiększenie ceny produktu (w *UPDATE*)
- przy usuwaniu produktu usunie wszystkie rekordy dla danego produktu z tabeli *T_ListaProduktow.*

Zad 7

Napisz wyzwalacz *BEFORE INSERT*, który nie pozwoli na dodanie nowej osoby do tabeli *T_Osoba* jeśli istnieje już osoba o takim samym nazwisku. Jeśli natomiast takie nazwisko jeszcze nie istnieje, wtedy dodajemy nową osobę wraz z informacją: „*{nazwisko} został pomyślnie dodany*”.

Po utworzeniu wyzwalacza spróbuj dodać osobę której nazwisko już istnieje w *T_Osoba*:

```
INSERT INTO T_Osoba (id, imie, nazwisko) VALUES (11, 'Tim',
'Theramenes');
```

Oczekiwany wynik:

```
[72000][20001]
ORA-20001: Osoba o podanym nazwisku już istnieje
ORA-06512: przy "PEUCESTAS.TR_OSOBA_BIR", linia 7
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'PEUCESTAS.TR_OSOBA_BIR'
Position: 12
```

Po utworzeniu wyzwalacza spróbuj dodać osobę której nazwisko nie istnieje w *T_Osoba*:

```
INSERT INTO T_Osoba (id, imie, nazwisko) VALUES (11, 'Tim',
'Thrasybulus');
```

Oczekiwany wynik:

```
Thrasybulus został pomyślnie dodany
```

Po utworzeniu wyzwalacza spróbuj dodać kilka osób w jednym poleceniu DML i wyciągnij wnioski:

```
INSERT ALL
    INTO T_Osoba (id, imie, nazwisko) VALUES (12, 'Liam', 'Thrasyllus')
    INTO T_Osoba (id, imie, nazwisko) VALUES (13, 'Keith', 'Conon')
    INTO T_Osoba (id, imie, nazwisko) VALUES (14, 'Reece',
'Callicratidas')
SELECT 1 FROM DUAL;
```

Oczekiwany wynik:

```
[42000][4091]
ORA-04091: tabela PEUCESTAS.T_OSOBA ulega mutacji, wyzwalacz/funkcja może tego nie widzieć
ORA-06512: przy "PEUCESTAS.TR_OSOBA_BIR", linia 4
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'PEUCESTAS.TR_OSOBA_BIR'
Position: 21
```

Zad 8

Napisz wyzwalacz *BEFORE UPDATE* dla tabeli *T_Zatrudnienie*, który nie pozwoli na modyfikowanie wartości żadnej z kolumn z wyjątkiem kolumny *Do*. Dodatkowo kolumnę *Do* można modyfikować tylko wtedy gdy jej wartość jest *NULL*em i nie może mieć przypisanej daty *Do* wcześniejszej niż data *Od*.

Zad 9

Utwórz tabelę *T_SprzedaneProdukty* z jedną kolumną „*wartosc*”, która będzie przechowywała wartość wszystkich sprzedanych produktów i będzie zawsze zawierała tylko jeden wiersz. Utwórz jeden wyzwalacz, który będzie pilnował, aby wartość w tabeli *T_SprzedaneProdukty* była zawsze aktualna. Przy wszystkich operacjach aktualizujących tabelę *T_ListaProduktow* (*INSERT*, *UPDATE*, *DELETE*), wyzwalacz powinien aktualizować wartość w tabeli *T_SprzedaneProdukty*.

Tworzenie tabeli *T_SprzedaneProdukty* i przypisanie wartości do kolumny „*Wartosc*”:

```
CREATE TABLE T_SprzedaneProdukty(
    Wartosc number(8,2) not null
);

DECLARE
    v_wartosc number(8,2);
BEGIN
    SELECT SUM(cena * ilosc) INTO v_wartosc FROM T_ListaProduktow lp
JOIN T_Produkt p ON lp.produkt = p.id;
    INSERT INTO T_SprzedaneProdukty VALUES (v_wartosc);
END;
```

Zad 10

Stwórz perspektywę, która będzie zawierać imię i nazwisko pracownika, jego pensję oraz stanowisko. Następnie stwórz wyzwalacz *INSTEAD OF* dla tej perspektywy, który będzie dodawał rekord do bazy danych. Jeśli osoba o podanym imieniu i nazwisku nie istnieje, to

dodajemy ją do tabeli *T_Osoba*, tak samo postępujemy z pracownikiem. Jeśli pracownik już istnieje to aktualizujemy jego pensję. Jeśli stanowisko nie istnieje to tworzymy nowe stanowisko. Jeśli pracownik nie jest aktualnie zatrudniony na dane stanowisko to przypisujemy mu je, wcześniej wypisując go z poprzedniego stanowiska z dzisiejszą datą (jeśli je ma).

Tworzenie perspektywy:

```
CREATE VIEW V_Pracownik (Imie, Nazwisko, Pensja, Stanowisko)
AS
SELECT o.imie, o.nazwisko, p.pensja, s.nazwa
FROM T_Osoba o JOIN T_Pracownik p ON o.id = p.id
JOIN T_Zatrudnienie z ON z.pracownik = p.id
JOIN T_Stanowisko s ON s.id = z.stanowisko
WHERE z.do IS NULL;
```

Test 1:

```
INSERT INTO V_PRACOWNIK VALUES ('Mark', 'Clearchus', 9999, 'cashier');
```

Oczekiwany wynik:

```
Zaktualizowano pensję pracownika o id= 2
Dany pracownik jest już zatrudniony na tym stanowisku
```

Test 2:

```
INSERT INTO V_PRACOWNIK VALUES ('Mark', 'Clearchus', 9999, 'boss');
```

Oczekiwany wynik:

```
Zaktualizowano pensję pracownika o id= 2
Utworzono nowe stanowisko o nazwie= boss
Zaktualizowano historie zatrudnienia pracownika o id= 2
```

Test 3:

```
INSERT INTO V_PRACOWNIK VALUES ('Matthew', 'Mindarus', 8675,
'janitor');
```

Oczekiwany wynik:

```
Dodano nowego człowieka o id= 11
Dodano nowego pracownika o id= 11
Zaktualizowano historie zatrudnienia pracownika o id= 11
```

Test 4:

```
INSERT INTO V_PRACOWNIK VALUES ('Matthew', 'Mindarus', 8675,
'security');
```

Oczekiwany wynik:

```
Zaktualizowano pensję pracownika o id= 11
Utworzono nowe stanowisko o nazwie= security
Zaktualizowano historie zatrudnienia pracownika o id= 11
```

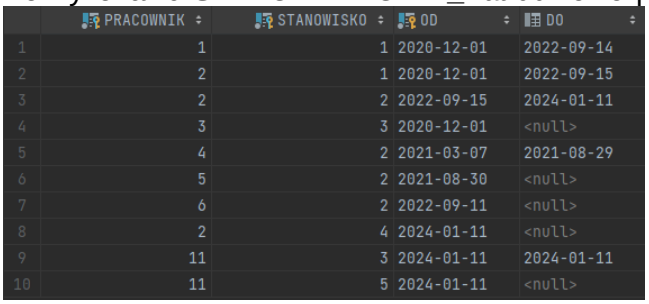
Test 5:

```
INSERT INTO V_PRACOWNIK VALUES ('Matthew', 'Mindarus', 9999,
'security');
```

Oczekiwany wynik:

```
Zaktualizowano pensję pracownika o id= 11
Dany pracownik jest już zatrudniony na tym stanowisku
```

Po wykonaniu *SELECT * FROM T_Zatrudnienie* powinniśmy widzieć następujący wynik:



	PRACOWNIK	STANOWISKO	OD	DO
1	1	1	2020-12-01	2022-09-14
2	2	1	2020-12-01	2022-09-15
3	2	2	2022-09-15	2024-01-11
4	3	3	2020-12-01	<null>
5	4	2	2021-03-07	2021-08-29
6	5	2	2021-08-30	<null>
7	6	2	2022-09-11	<null>
8	2	4	2024-01-11	<null>
9	11	3	2024-01-11	2024-01-11
10	11	5	2024-01-11	<null>