JEGYZŐKÖNYV

Adatbázis rendszerek II.

Féléves feladat

Étterem nyilvántartása

Készítette Kerekes Krisztofer

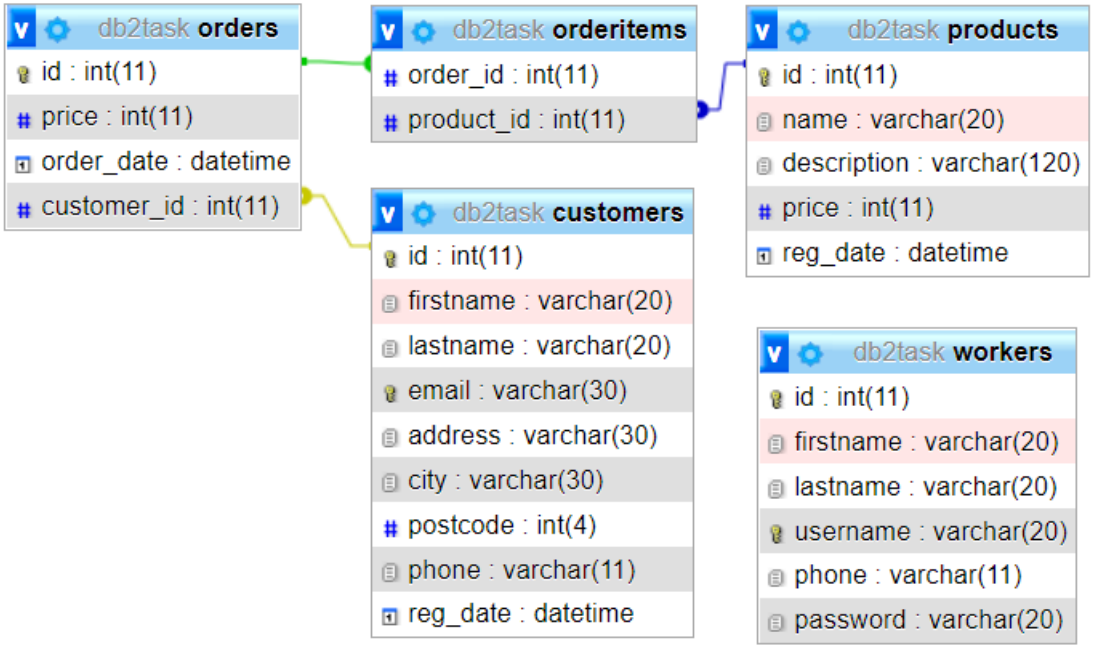
Neptunkód: TRNA8A

Gyakorlatvezető: Dr. Bednarik László

A feladat leírása

Ebben a feladatban az előzőleg elkészített étterem nyilvántartó programnak az adattábláit és adatait használom. A legtöbb esetben a ’products’ táblán alkalmazom a feladatokat, de egyes feladatokat kiterjesztek a többi táblára is.

Adatbázis séma



A táblákat létrehozó SQL parancsok

```
CREATE TABLE Workers (
  id NUMBER NOT NULL,
  firstname VARCHAR2(20) NOT NULL,
  lastname VARCHAR2(20) NOT NULL,
  username VARCHAR2(20) UNIQUE NOT NULL,
  phone VARCHAR2(11) NOT NULL,
  password VARCHAR2(20) NOT NULL,
  PRIMARY KEY (id)
);
CREATE TABLE Customers (
  id NUMBER NOT NULL,
  firstname VARCHAR2(20) NOT NULL,
  lastname VARCHAR2(20) NOT NULL,
```

```
  email VARCHAR2(30) UNIQUE NOT NULL,
  address VARCHAR2(30),
  city VARCHAR2(30),
  postcode NUMBER(4),
  phone VARCHAR2(11) NOT NULL,
  reg_date TIMESTAMP NOT NULL,
  PRIMARY KEY (id)
);
CREATE TABLE Products (
  id NUMBER NOT NULL,
  name VARCHAR2(20) NOT NULL,
  description VARCHAR2(120),
  price NUMBER NOT NULL,
  reg_date TIMESTAMP NOT NULL,
  PRIMARY KEY (id)
);
CREATE TABLE Orders (
  id NUMBER NOT NULL,
  price NUMBER NOT NULL,
  order_date TIMESTAMP NOT NULL,
  customer_id NUMBER NOT NULL,
  PRIMARY KEY (id),
  FOREIGN KEY (customer_id) REFERENCES Customers(id)
);
CREATE TABLE OrderItems (
  order_id NUMBER NOT NULL,
  product_id NUMBER NOT NULL,
  FOREIGN KEY (order_id) REFERENCES Orders(id),
  FOREIGN KEY (product_id) REFERENCES Products(id)
);
```

A táblákat feltöltő SQL parancsok

Minden, az adatbázisban tárolt adat kitalált, bármilyen egyezés a valósággal csak a véletlen műve.

```
INSERT INTO workers(firstname, lastname, username, phone, password) VALUES
('Krisztofer','Kerekes','krisz00','06706762436','alma');

INSERT INTO workers(firstname, lastname, username, phone, password) VALUES ('Teszt
Fiók','','teszt','06203763855','12345');

INSERT INTO products(name, description, price, reg_date) VALUES
('Pizza Margareta','paradicsomszósz, sajt, oregano',1590,NOW());

INSERT INTO products(name, description, price, reg_date) VALUES
('Pizza Prosciutto','paradicsomszósz, sonka, sajt, oregano',1990,NOW());

INSERT INTO products(name, description, price, reg_date) VALUES
('Pizza Salami','paradicsomszósz,sajt,szalámi,oregano',1990,NOW());

INSERT INTO products(name, description, price, reg_date) VALUES
('Pizza Hawaii','paradicsomszósz, sajt, sonka, ananász, oregano',2100,NOW());

INSERT INTO products(name, description, price, reg_date) VALUES
('Pizza Magyaros','paradicsomszósz, sajt, szalámi, bacon, hagyma, erőspaprika,
oregano',2190,NOW());

INSERT INTO products(name, description, price, reg_date) VALUES
('Pizza Négysajtos','paradicsomszósz,sajt,füstölt sajt,trappista
sajt,mozzarella,parmezán,oregano',2200,NOW());

INSERT INTO products(name, description, price, reg_date) VALUES
('Coca Cola 1l',NULL,450,NOW());

INSERT INTO products(name, description, price, reg_date) VALUES
('Pepsi Cola 1l',NULL,450,NOW());

INSERT INTO products(name, description, price, reg_date) VALUES
('Ásványvíz',NULL,300,NOW());

INSERT INTO customers(firstname, lastname, email, address, city, postcode, phone, reg_date)
VALUES
('Cintia','Vass','vasscintia@gmail.com','Ady Endre utca 43','Emőd',3432,'06708435622',NOW());

INSERT INTO customers(firstname, lastname, email, address, city, postcode, phone, reg_date)
VALUES
('Nikoletta','Kovács','kovacsnikoletta@gmail.com','Balassi Bálint utca
22','Nyékládháza',3433,'06308467367',NOW());

INSERT INTO customers(firstname, lastname, email, address, city, postcode, phone, reg_date)
VALUES
('Dávid','Szabó','szabodavid@gmail.com','Vasút utca 10','Nyékládháza',3433,'06709463726',NOW());

INSERT INTO customers(firstname, lastname, email, address, city, postcode, phone, reg_date)
VALUES
('Márton','Fehér','fehermarton@gmail.com','Kossuth Lajos utca
9','Nyékládháza',3433,'06709557888',NOW());
```

PL/SQL

Tárolt eljárás adatok felvitelére

```
create or replace procedure newproduct(p_id number, p_name char, p_description char, p_price char) is
exsisterror exception;
rows_found number;
begin
    select count(*)
    into rows_found
    from products
    where id = p_id;

    if rows_found != 0 then raise exsisterror;
    else
        insert into products values(p_id, p_name, p_description, p_price, current_timestamp);
    end if;
exception
    when exsisterror then
    dbms_output.put_line('Ez az azonosító már létezik!');
end;
```

A ’newproduct’ eljárásban meg kell adni a termék azonosítóját, elnevezését, leírását és árát. Az eljárás ellenőrzi hogy az azonosító foglalt-e, ha nem akkor elvégzi a beszúrást.

Meghívás:

```
begin
    newproduct(8, 'Alma', 'Fán terem', '130');
end;

PL/SQL procedure successfully completed.
```

Tárolt eljárás adatok módosítására

```
create or replace procedure updateproduct(p_id number, p_newname char, p_newdescription char, p_newprice char) is
notexsisterror exception;
rows_found number;
begin
    select count(*)
    into rows_found
    from products
    where id = p_id;

    if rows_found = 0 then raise notexsisterror;
    else
        update products set name = p_newname, description = p_newdescription, price = p_newprice where id = p_id;
    end if;
exception
    when notexsisterror then
    dbms_output.put_line('Ilyen azonosítójú termék nem létezik!');
end;
```

Ebben az eljárásban meg kell adni a módosítandó termék azonosítóját és az új értékeit. Ha nem létező azonosító lett megadva, nem végzi el a módosítást.

Meghívás:

```
begin
    updateproduct(8, 'Körte', 'Fán terem', '140');
end;

PL/SQL procedure successfully completed.
```

Tárolt eljárás adatok módosítására mezőnként

```
create or replace procedure updateproductname(p_id number, p_newname char) is
notexsisterror exception;
nameequalerror exception;
rows_found number;
current_name char(20);
begin
    select count(*)
    into rows_found
    from products
    where id = p_id;

    if rows_found !=0 then
        select name
        into current_name
        from products
        where id = p_id;
    end if;

    if rows_found = 0 then raise notexsisterror;
    elsif p_newname = current_name then raise nameequalerror;
    else
        update products set name = p_newname where id = p_id;
    end if;
exception
    when notexsisterror then
    dbms_output.put_line('Ilyen azonosítójú termék nem létezik!');
    when nameequalerror then
    dbms_output.put_line('Ugyanaz a név van megadva!');
end;
```

Meghívás:

```
begin
    updateproductname(8, 'Szilva');
end;

PL/SQL procedure successfully completed.
```

Az eljárás ellenőrzi, hogy a megadott érték megegyezik-e az eddigivel. Ebben az esetben nem végzi el a módosítást.

```
create or replace procedure updateproductdescription(p_id number, p_newdescription char) is
notexsisterror exception;
rows_found number;
begin
    select count(*)
    into rows_found
    from products
    where id = p_id;

    if rows_found = 0 then raise notexsisterror;
    else
        update products set description = p_newdescription where id = p_id;
    end if;
exception
    when notexsisterror then
    dbms_output.put_line('Ilyen azonosítójú termék nem létezik!');
end;
```

Meghívás:

```
begin
    updateproductdescription(8, 'Ez egy gyümölcs');
end;

PL/SQL procedure successfully completed.
```

```
create or replace procedure updateproductprice(p_id number, p_newprice number) is
notexsisterror exception;
priceisnotvalid exception;
rows_found number;
begin
    select count(*)
    into rows_found
    from products
    where id = p_id;

    if rows_found = 0 then raise notexsisterror;
    elsif p_newprice < 1 then raise priceisnotvalid;
    else
        update products set price = p_newprice where id = p_id;
    end if;
exception
    when notexsisterror then
    dbms_output.put_line('Ilyen azonosítójú termék nem létezik!');
    when priceisnotvalid then
    dbms_output.put_line('A megadott ár nem megfelelő!');
end;
```

Meghívás:

```
begin
    updateproductprice(8, 200);
end;

PL/SQL procedure successfully completed.
```

Tárolt eljárás adatok törlésére

```
create or replace procedure deleteproductbyid(p_id number) is
notexsisterror exception;
rows_found number;
begin
    select count(*)
    into rows_found
    from products
    where id = p_id;

    if rows_found = 0 then raise notexsisterror;
    else
        delete from products where id = p_id;
    end if;
exception
    when notexsisterror then
    dbms_output.put_line('Ilyen azonosítójú termék nem létezik!');
end;
```

Ez az eljárás paraméterként kéri a törölni kívánt termék azonosítóját.

Meghívás:

```
begin
    deleteproductbyid(8);
end;

PL/SQL procedure successfully completed.
```

```
create or replace procedure deleteproductbyname(p_name char) is
notexsisterror exception;
rows_found number;
begin
    select count(*)
    into rows_found
    from products
    where name = p_name;

    if rows_found = 0 then raise notexsisterror;
    else
        delete from products where name = p_name;
    end if;
exception
    when notexsisterror then
    dbms_output.put_line('Ilyen termék nem létezik!');
end;
```

Ez az eljárás paraméterként kéri a törölni kívánt termék elnevezését.

Meghívás:

```
begin
    deleteproductbyname('Körte');
end;

PL/SQL procedure successfully completed.
```

Tárolt függvény adott feltételű rekordok aggregált értékének lekérdezésére

```
create or replace function productcount return int as f_result int;
begin
    select count(id) into f_result from products;
    return(f_result);
end;
```

Ez a függvény visszaadja, mennyi termék van a rendszerben.

Meghívás:

```
select productcount() from products;
```

	PRODUCTCOUNT()
1	5

```
create or replace function totalpricegreatherthan(f_price in int) return int as f_result int;
begin
    select sum(price) into f_result from products where price > f_price;
    return(f_result);
end;
```

Ez a függvény pedig megadja a paraméterben megadott értéknél nagyobb árú termékek összárát.

Meghívás:

```
select totalpricegreaterthan(1000) from products;
```

	TOTALPRICEGREATERTHAN(1000)
1	6540

megjegyzés: futtatáskor nem volt az összes adat a táblában!

Tárolt csomag egy tábla funkcióinak összefogására

```
create or replace package productPackage is
    procedure newProduct(p_name char, p_description char, p_price char);
    procedure updateProductName(p_id number, p_newname char);
    procedure updateProductDescription(p_id number, p_newdescription char);
    procedure updateProductPrice(p_id number, p_newprice number);
    procedure deleteProductById(p_id number);
    procedure deleteProductByName(p_name char);
    function totalpricegreaterthan(f_price in int) return int;
end productPackage;
```

Ebben a csomagban az előző feladatokban létrehozott eljárásokat és függvényeket hozom össze.

```
create or replace package body productPackage is
    procedure newProduct(p_name char, p_description char, p_price char) is
    begin
        insert into products values(null, p_name, p_description, p_price, current_timestamp);
    end;

    procedure updateProductName(p_id number, p_newname char) is
    notexsisterror exception;
    nameequalerror exception;
    rows_found number;
    current_name char(20);
    begin
        select count(*)
        into rows_found
        from products
        where id = p_id;

        if rows_found !=0 then
            select name
            into current_name
            from products
            where id = p_id;
        end if;

        if rows_found = 0 then raise notexsisterror;
        elsif p_newname = current_name then raise nameequalerror;
        else
            update products set name = p_newname where id = p_id;
        end if;
    exception
        when notexsisterror then
        dbms_output.put_line('Ilyen azonosítójú termék nem létezik!');
        when nameequalerror then
        dbms_output.put_line('Ugyanaz a név van megadva!');
    end;

    procedure updateProductDescription(p_id number, p_newdescription char) is
    notexsisterror exception;
    rows_found number;
    begin
        select count(*)
        into rows_found
        from products
        where id = p_id;

        if rows_found = 0 then raise notexsisterror;
        else
            update products set description = p_newdescription where id = p_id;
        end if;
    exception
        when notexsisterror then
        dbms_output.put_line('Ilyen azonosítójú termék nem létezik!');
    end;
```

```
    procedure updateProductPrice(p_id number, p_newprice number) is
    notexsisterror exception;
    priceisnotvalid exception;
    rows_found number;
    begin
        select count(*)
        into rows_found
        from products
        where id = p_id;

        if rows_found = 0 then raise notexsisterror;
        elsif p_newprice < 1 then raise priceisnotvalid;
        else
            update products set price = p_newprice where id = p_id;
        end if;
    exception
        when notexsisterror then
        dbms_output.put_line('Ilyen azonosítójú termék nem létezik!');
        when priceisnotvalid then
        dbms_output.put_line('A megadott ár nem megfelelő!');
    end;

    procedure deleteProductById(p_id number) is
    notexsisterror exception;
    rows_found number;
    begin
        select count(*)
        into rows_found
        from products
        where id = p_id;

        if rows_found = 0 then raise notexsisterror;
        else
            delete from products where id = p_id;
        end if;
    exception
        when notexsisterror then
        dbms_output.put_line('Ilyen azonosítójú termék nem létezik!');
    end;
```

```
    procedure deleteProductByName(p_name char) is
    notexsisterror exception;
    rows_found number;
    begin
        select count(*)
        into rows_found
        from products
        where name = p_name;

        if rows_found = 0 then raise notexsisterror;
        else
            delete from products where name = p_name;
        end if;
    exception
        when notexsisterror then
        dbms_output.put_line('Ilyen termék nem létezik!');
    end;

    function totalPriceGreaterThan(f_price in int) return int as f_result int;
    begin
        select sum(price) into f_result from products where price > f_price;
        return(f_result);
    end;
end productPackage;
```

Ezen eljárások és függvények meghívása a csomag nevével történik. pl.:

```
select productPackage.totalPriceGreaterThan(1000) from products;
begin
    productPackage.deleteProductByName('Körte');
end;
```

Triggerek

Első körben létrehoztam egy ’data_log’ táblát.

```
CREATE TABLE DATA_LOG (
  log_date TIMESTAMP,
  log_description varchar2(30),
  log_table varchar (20)
);
```

Trigger módosítási események naplózására

```
create or replace trigger updatelogproduct before update on products for each row
begin
    insert into data_log values(current_timestamp, 'módosítás', 'products');
end;
```

Trigger kulcs érték automatikus megadására

Létrehoztam egy szekvenciát.

```
CREATE SEQUENCE products_seq;
```

Ezután pedig magát a triggert.

```
CREATE OR REPLACE TRIGGER products_insert
  BEFORE INSERT ON products
  FOR EACH ROW
BEGIN
  SELECT products_seq.nextval
  INTO :new.id
  FROM dual;
END;
```

Ebből kifolyólag készítettem egy új feltöltőmetódust, amit ’newproduct2’-nek neveztem el. Ebben már nem kell külön megadni az azonosítót.

```
create or replace procedure newproduct2(p_name char, p_description char, p_price char) is
begin
    insert into products values(null, p_name, p_description, p_price, current_timestamp);
end;
```

Trigger a módosítások kontrollálására

```
create or replace trigger productupdatechecktrigger before update on products for each row
begin
    if not ((:new.price / (:old.price / 100)) >= 80 and (:new.price / (:old.price / 100)) <= 120) then
        if :old.price < :new.price then
            :new.price := :old.price * 1.2;
        end if;
        if :new.price < :old.price then
            :new.price := :old.price * 0.8;
        end if;
        dbms_output.put_line('A módosítás mértéke meghaladja a 20%-ot, ezért korlátozva lett!');
    end if;
end;
```

Ez a trigger ellenőrzi a termék árának módosítása előtt az új értéket. Ha az 20%-ot meghaladó mértékben tér el az eredeti értéktől, a módosítást felülírja és csak 20%-kal módosítja az árát a módosítás irányának megfelelően. (Ha a módosítás pozitív irányban haladja meg a 20%-ot, akkor ennek megfelelően korlátozza a módosítás mértékét.)

Többféle esemény naplózása egyetlen triggerrel (plusz pont)

```
create or replace trigger producttrigger
before insert or delete or update on products for each row
begin
    if inserting then
        insert into data_log values(current_timestamp, 'beszúrás', 'products');
    end if;
    if deleting then
        insert into data_log values(current_timestamp, 'törlés', 'products');
    end if;
    if updating then
        insert into data_log values(current_timestamp, 'módosítás', 'products');
    end if;
end;

create or replace trigger workertrigger
before insert or delete or update on workers for each row
begin
    if inserting then
        insert into data_log values(current_timestamp, 'beszúrás', 'workers');
    end if;
    if deleting then
        insert into data_log values(current_timestamp, 'törlés', 'workers');
    end if;
    if updating then
        insert into data_log values(current_timestamp, 'módosítás', 'workers');
    end if;
end;

create or replace trigger customertrigger
before insert or delete or update on customers for each row
begin
    if inserting then
        insert into data_log values(current_timestamp, 'beszúrás', 'customers');
    end if;
    if deleting then
        insert into data_log values(current_timestamp, 'törlés', 'customers');
    end if;
    if updating then
        insert into data_log values(current_timestamp, 'módosítás', 'customers');
    end if;
end;

create or replace trigger orderrigger
before insert or delete or update on orders for each row
begin
    if inserting then
        insert into data_log values(current_timestamp, 'beszúrás', 'orders');
    end if;
    if deleting then
        insert into data_log values(current_timestamp, 'törlés', 'orders');
    end if;
    if updating then
        insert into data_log values(current_timestamp, 'módosítás', 'orders');
    end if;
end;
```

Kurzor

```
declare
c_name products.name%type;
c_description products.description%type;
c_price products.price%type;
cursor c_products is select name, description, price from products;
begin
    open c_products;
    loop
        fetch c_products into c_name, c_description, c_price;
        exit when c_products%notfound;
        dbms_output.put_line(c_name || ' ' || c_description || ' ' || c_price);
    end loop;
    close c_products;
end;
```

Eredmény:

```
Pizza Margareta paradicsomszósz, sajt, oregano 1590
Pizza Prosciutto paradicsomszósz, sonka, sajt, oregano 1990
Pizza Salami paradicsomszósz,sajt,szalámi,oregano 1990
Pizza Hawaii paradicsomszósz, sajt, sonka, ananász, oregano 2100
Pizza Magyaros paradicsomszósz, sajt, szalámi, bacon, hagyma, erőspaprika, oregano 2190
Pizza Négysajtos paradicsomszósz,sajt,füstölt sajt,trappista sajt,mozzarella,parmezán,oregano 2200
```