SZABADKAI MŰSZAKI SZAKFŐISKOLA
SZABADKA

VEGYES NAGYKERESKEDÉS ADATBÁZIS

projektum
Adatbázis 2 tárgyból

témavezető: Dr. Simon János
főiskolai tanár

hallgató: Kovács Róbert
leckekönyv: 16218119
szakirány: Műszaki informatika

Szabadka, 2020

Tartalom

Vegyes nagykereskedés

Az adatbázis egy áruház lánc adatait fogja tartalmazni, ami vegyes árukat fog forgalmazni. Ebben az adatbázisban nyomon lehet követni az áruházak forgalmat, raktárát és munkásait.

Az adatbázisban található táblák a következők:

- aruhazak
  - id_aruhaz
  - aruhaz_telepules
  - aruhaz_cim
- raktar
  - id_raktar
  - id_aruhaz
  - id_termek
  - termek_mennyiseg
  - utolso_szallitas
- termekek
  - id_termek
  - termek_nev
  - termek_tipus
  - termek_leiras
  - termek_ar
- penz
  - id_penz
  - id_aruhaz
  - bevetel
  - kiadas
  - datum
- munkasok
  - id_munkas
  - id_aruhaz
  - id_munka
  - jmbg
  - munkas_vezeteknev
  - munkas_keresztnev
  - munkas_telepules
  - munkas_cim
- munkak
  - id_munka
  - munka_nev
  - munka_leiras
  - munka_fizetes
  - munka_ora

Műveletek

Az adatbázisban a következő műveleteket használtam: eljárások, függvények, eseményindítók és nézetek.

Eljárások

vasarlas

Ezzel az eljárással, az adott áruház raktárából kivonja a megfelelő mennyiségű vásárolt terméket, és az aznapi bevételhez hozzáadja az árát.

```
DELIMITER //
CREATE PROCEDURE vasarlas (IN aruhaz_id INT(11), IN termek_id INT(11), IN
mennyiseg INT(3))
BEGIN
    DECLARE t_mennyiseg INT(5);
    DECLARE id_a, id_t, d BOOLEAN;
    SELECT termek_mennyiseg INTO t_mennyiseg
    FROM raktar
    WHERE id_aruhaz = aruhaz_id AND id_termek = termek_id;
    SELECT EXISTS(SELECT id_aruhaz FROM aruhazak WHERE id_aruhaz = aruhaz_id)
    INTO id_a;
    SELECT EXISTS(SELECT id_termek FROM termekek WHERE id_termek = termek_id)
    INTO id_t;
    IF t_mennyiseg >= mennyiseg AND id_a AND id_t THEN
        UPDATE raktar
        SET termek_mennyiseg = termek_mennyiseg - mennyiseg
        WHERE id_aruhaz = aruhaz_id AND id_termek = termek_id;

        SELECT EXISTS(SELECT datum FROM penz
        WHERE id_aruhaz = aruhaz_id AND datum = CURDATE())
        INTO d;
        IF d THEN
            UPDATE penz
            SET bevetel = bevetel + vasarlas_ara(termek_id, mennyiseg)
            WHERE id_aruhaz = aruhaz_id AND datum = CURDATE();
        ELSE
            INSERT INTO penz (id_aruhaz, bevetel, kiadas, datum)
            VALUES (aruhaz_id, vasarlas_ara(termek_id, mennyiseg), 0, CURDATE());
        END IF;
        SELECT raktar.id_aruhaz, raktar.termek_mennyiseg, termekek.termek_nev,
termekek.termek_ar, penz.bevetel, raktar.utolso_szallitas
        FROM raktar
        JOIN termekek ON termekek.id_termek = raktar.id_termek
        JOIN penz ON penz.id_aruhaz = raktar.id_aruhaz
        WHERE raktar.id_aruhaz = aruhaz_id AND raktar.id_termek = termek_id AND
penz.datum = CURDATE();
    ELSE
        SELECT "NEM TORTENT SEMMI SEM" AS "HIBA";
    END IF;

END//
DELIMITER ;
```

beszallitas

Ezzel az eljárással, az adott áruház raktárába termékeket adhatunk hozzá a már meglévő termékhez vagy pedig új terméket adunk hozzá a raktárhoz, illetve az aznapi kiadáshoz hozzáadja a termék árát.

```
DELIMITER //
CREATE PROCEDURE beszallitas (IN aruhaz_id INT(11), IN termek_id INT(11), IN
mennyiseg INT(3))
BEGIN
    DECLARE id_a, id_t, d BOOLEAN;
    SELECT EXISTS(SELECT id_aruhaz FROM aruhazak WHERE id_aruhaz = aruhaz_id)
    INTO id_a;
    SELECT EXISTS(SELECT id_termek FROM termekek WHERE id_termek = termek_id)
    INTO id_t;
    IF id_a AND id_t THEN
        SELECT EXISTS(SELECT id_termek FROM raktar
        WHERE id_aruhaz = aruhaz_id AND id_termek = termek_id)
        INTO id_t;
        IF id_t THEN
            UPDATE raktar
            SET termek_mennyiseg = termek_mennyiseg + mennyiseg,
                utolso_szallitas = CURDATE()
            WHERE id_aruhaz = aruhaz_id AND id_termek = termek_id;
        ELSE
            INSERT INTO raktar (id_aruhaz, id_termek, termek_mennyiseg,
utolso_szallitas)
            VALUES (aruhaz_id, termek_id, mennyiseg, CURDATE());
        END IF;
        SELECT EXISTS(SELECT datum FROM penz
        WHERE id_aruhaz = aruhaz_id AND datum = CURDATE())
        INTO d;
        IF d THEN
            UPDATE penz
            SET kiadas = kiadas + beszallitas_ara(termek_id, mennyiseg)
            WHERE id_aruhaz = aruhaz_id AND datum = CURDATE();
        ELSE
            INSERT INTO penz (id_aruhaz, bevetel, kiadas, datum)
            VALUES (aruhaz_id, 0, beszallitas_ara(termek_id, mennyiseg),
CURDATE());
        END IF;
        SELECT raktar.id_aruhaz, raktar.termek_mennyiseg, termekek.termek_nev,
termekek.termek_ar, penz.kiadas
        FROM raktar
        JOIN termekek ON termekek.id_termek = raktar.id_termek
        JOIN penz ON penz.id_aruhaz = raktar.id_aruhaz
        WHERE raktar.id_aruhaz = aruhaz_id AND raktar.id_termek = termek_id AND
penz.datum = CURDATE();
    ELSE
        SELECT "NEM TORTENT SEMMI SEM" AS "HIBA";
    END IF;
END//
DELIMITER ;
```

uj_aruhaz

Ezzel az eljárással új árúházat adhatunk hozzá az adatbázishoz.

```
DELIMITER //
CREATE PROCEDURE uj_aruhaz (IN telepules VARCHAR(20), IN cim VARCHAR(27), IN
hazszam INT(3))
BEGIN
    INSERT INTO aruhazak (aruhaz_telepules, aruhaz_cim)
    VALUES (telepules, CONCAT(cim, ', ', hazszam));
    SELECT * FROM aruhazak ORDER BY id_aruhaz DESC LIMIT 1;
END//
DELIMITER ;
```

uj_termek

Ezzel az eljárással új terméket adhatunk hozzá az adatbázishoz.

```
DELIMITER //
CREATE PROCEDURE uj_termek (IN nev VARCHAR(20), IN tipus VARCHAR(20), IN leiras
VARCHAR(512), IN ar INT(5))
BEGIN
    INSERT INTO termekek (termek_nev, termek_tipus, termek_leiras, termek_ar)
    VALUES (nev, tipus, leiras, ar);
    SELECT * FROM termekek ORDER BY id_termek DESC LIMIT 1;
END//
DELIMITER ;
```

uj_munkas

Ezzel az eljárással új munkást adhatunk hozzá az adatbázishoz, ha létezik a munka vagy az árúház, illetve megfelelő hosszúságú JMBG-t vittünk be.

```
DELIMITER //
CREATE PROCEDURE uj_munkas (IN vezeteknev VARCHAR(20), IN keresztnev VARCHAR(20),
IN JMBG BIGINT(13), IN telepules VARCHAR(20), IN cim VARCHAR(27), IN hazszam
INT(3), IN aruhazid INT(11), IN munkaid INT(11))
BEGIN
    DECLARE id_a, id_m BOOLEAN;
    SELECT EXISTS(SELECT id_aruhaz FROM aruhazak WHERE id_aruhaz = aruhazid)
    INTO id_a;
    SELECT EXISTS(SELECT id_munka FROM munkak WHERE id_munka = munkaid)
    INTO id_m;
    IF id_a AND id_M AND LENGTH(JMBG) = 13 THEN
        INSERT INTO munkasok (id_aruhaz, id_munka, jmbg, munkas_vezeteknev,
munkas_keresztnev, munkas_telepules, munkas_cim)
        VALUES (aruhazid, munkaid, JMBG, vezeteknev, keresztnev, telepules,
CONCAT(cim, ', ', hazszam));
        SELECT * FROM munkasok
        ORDER BY id_munkas DESC LIMIT 1;
    ELSE
        SELECT "NEM TORTENT SEMMI SEM" AS "HIBA";
    END IF;
END//
DELIMITER ;
```

aruhaz_info_where

Ezzel az eljárással szűrhetjük az aruhaz_info nézettáblát

```
DELIMITER //
CREATE PROCEDURE munkas_info_where (IN aruhaz_id INT(11), IN vezeteknev
VARCHAR(20), IN keresztnev VARCHAR(20), IN telepules VARCHAR(20),
IN munka VARCHAR(20))
BEGIN
    SELECT * FROM munkas_info
    WHERE
    (aruhaz_id IS NULL OR id_aruhaz = aruhaz_id) AND
    (vezeteknev IS NULL OR munkas_vezeteknev LIKE CONCAT('%', vezeteknev, '%'))
AND (keresztnev IS NULL OR munkas_keresztnev LIKE CONCAT('%', keresztnev, '%'))
AND (telepules IS NULL OR munkas_telepules LIKE CONCAT('%', telepules, '%'))
AND (munka IS NULL OR munka_nev LIKE CONCAT('%', munka, '%'));
END//
DELIMITER ;
```

aruhazraktar_info_where

Ezzel az eljárással szűrhetjük az aruhaz_raktarinfo nézettáblát

```
DELIMITER //
CREATE PROCEDURE aruhazraktar_info_where (IN aruhaz_id INT(11),
IN tnev VARCHAR(20), IN tipus VARCHAR(20))
BEGIN
    SELECT * FROM aruhaz_raktarinfo
    WHERE
        (aruhaz_id IS NULL OR id_aruhaz = aruhaz_id) AND
        (tnev IS NULL OR termek_nev LIKE CONCAT('%', tnev, '%')) AND
        (tipus IS NULL OR termek_tipus LIKE CONCAT('%', tipus, '%'));

END//
DELIMITER ;
```

Függvények

vasarlas_ara

Ez a függvény visszadja a vásárolt termék és a mennyiség szorzatát.

```
DELIMITER //
CREATE FUNCTION vasarlas_ara (termek_id INT(11), mennyiseg INT(3))
RETURNS INT(8) DETERMINISTIC
BEGIN
    DECLARE ar INT(6);
    SELECT termek_ar INTO ar
    FROM termekek
    WHERE id_termek = termek_id;
    SET ar = ar*mennyiseg;
    RETURN ar;
END //
DELIMITER ;
```

beszallitas_ara

Ez a függvény visszadja a beszállítási árát a termék és a mennyiség szorzata alapján.

```
DELIMITER //
CREATE FUNCTION beszallitas_ara (termek_id INT(11), mennyiseg INT(3))
RETURNS INT(8) DETERMINISTIC
BEGIN
    DECLARE ar INT(6);
    SELECT termek_ar INTO ar
    FROM termekek
    WHERE id_termek = termek_id;
    SET ar = (ar*mennyiseg)/2;
    RETURN ar;
END //
DELIMITER ;
```

Eseményindítók

4 + 4 esemény indító van az adatbázisban, négy szolgál arra, hogy minden táblába insert előtt minden szöveget nagybetűsre cseréljen és négy szolgál arra, hogy update esetén is nagybetűsre cserélje a neveket.

insert_name_upper

```
CREATE TRIGGER insert_name_upper
BEFORE INSERT ON munkasok
FOR EACH ROW
SET NEW.munkas_vezeteknev = UPPER(NEW.munkas_vezeteknev),
    NEW.munkas_keresztnev = UPPER(NEW.munkas_keresztnev),
    NEW.munkas_cim = UPPER(NEW.munkas_cim),
    NEW.munkas_telepules = UPPER(NEW.munkas_telepules);
```

update_name_upper

```
CREATE TRIGGER update_name_upper
BEFORE UPDATE ON munkasok
FOR EACH ROW
SET NEW.munkas_vezeteknev = UPPER(NEW.munkas_vezeteknev),
    NEW.munkas_keresztnev = UPPER(NEW.munkas_keresztnev),
    NEW.munkas_cim = UPPER(NEW.munkas_cim),
    NEW.munkas_telepules = UPPER(NEW.munkas_telepules);
```

insert_munkanev_upper

```
CREATE TRIGGER insert_munkanev_upper
BEFORE INSERT ON munkak
FOR EACH ROW
SET NEW.munka_nev = UPPER(NEW.munka_nev);
```

insert_termeknev_upper

```
CREATE TRIGGER insert_termekeknev_upper
BEFORE INSERT ON termekek
FOR EACH ROW
SET NEW.termek_nev = UPPER(NEW.termek_nev),
    NEW.termek_tipus = UPPER(NEW.termek_tipus);
```

insert_aruhaz_upper

```
CREATE TRIGGER insert_aruhaz_upper
BEFORE INSERT ON aruhazak
FOR EACH ROW
SET NEW.aruhaz_telepules = UPPER(NEW.aruhaz_telepules),
    NEW.aruhaz_cim = UPPER(NEW.aruhaz_cim);
```

Nézettáblák

A nézettáblák arra szolgálnak az adatbázisban, hogy kiírassák a fontos információkat az árúházakkal kapcsolatban.

aruhaz_raktarinfo

```
CREATE VIEW aruhaz_raktarinfo AS
SELECT raktar.id_aruhaz, termekek.id_termek, termekek.termek_nev,
termekek.termek_tipus, termekek.termek_ar, raktar.termek_mennyiseg,
raktar.utolso_szallitas
FROM raktar
JOIN termekek ON termekek.id_termek = raktar.id_termek
```

aruhaz_info

```
CREATE VIEW aruhaz_info AS
SELECT aruhazak.id_aruhaz, aruhazak.aruhaz_telepules, aruhazak.aruhaz_cim,
SUM(DISTINCT raktar.termek_mennyiseg) AS "termek_mennyiseg", SUM(DISTINCT
penz.bevetel) AS "bevetel", SUM(DISTINCT penz.kiadas) AS "kiadas",
((SUM(DISTINCT penz.bevetel)) - (SUM(DISTINCT penz.kiadas))) AS "jovedelem"
FROM aruhazak
JOIN raktar ON raktar.id_aruhaz = aruhazak.id_aruhaz
JOIN penz ON penz.id_aruhaz = aruhazak.id_aruhaz
GROUP BY aruhazak.id_aruhaz
```

munkas_info

```
CREATE VIEW munkas_info AS
SELECT munkasok.id_aruhaz, munkasok.munkas_vezeteknev, munkasok.munkas_keresztnev,
munkasok.munkas_telepules, munkak.munka_nev, munkak.munka_fizetes
FROM munkasok
JOIN munkak ON munkak.id_munka = munkasok.id_munka
ORDER BY munkasok.id_aruhaz
```

Táblák vizuális elrendezése