Adatbázis rendszerek 2

Féléves egyéni **PL/SQL** feladat  
**Egyéni oktatási cégek**

**Készítette**:

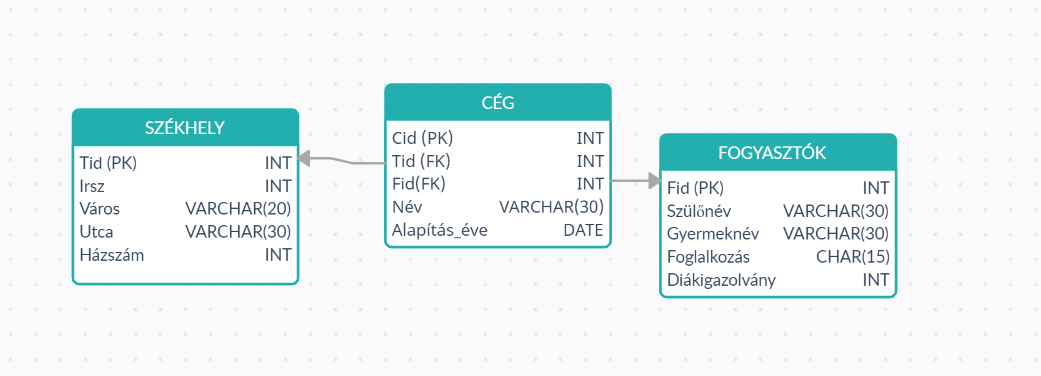
Oravecz Áron D3U3EE

**Gyakorlatvezető**:

Bednarik László

## Témakör ismertetése leírása

## Adatbázis modell:

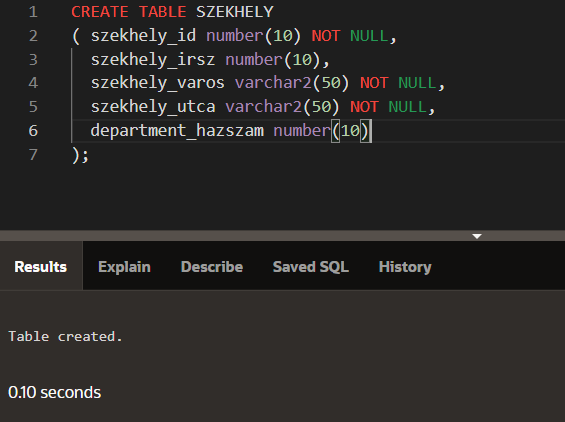


## Adatbázis jellemzése:

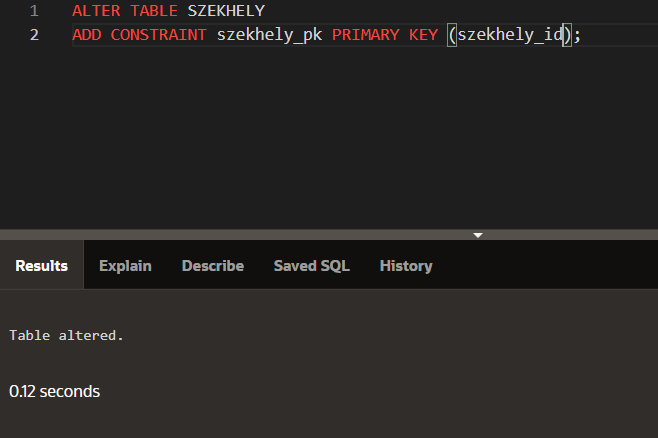
* Székhely jelen esetben 1:1 kapcsolatban áll a Céggel.
* A Fogyasztók a céggel pedig 1:N kapcsolatban állnak egymással.
* A Székhely és a Fogyasztókat kapcsolótáblával köthetjük össze a Cég tábla jelenlétében.
* **Székhely**:
  + A Cég tartózkodási helyét tartalmazza. Elsődleges kulcsként egy Tid-t alkalmazok, ami azonosító ként szolgál.
  + Irányítószám: ahol elhelyezkedik a székhely körzeti szinten, ugyanakkor egész szám típusú rekord.
  + Város és az utca szöveges típusúak.
  + A házszám numerikus számként funkcionál jelen esetben.
* **Fogyasztók**:
  + A Céget igénybe vevő csoportot foglalja magába.
  + Fid: azonosító és egyben elsődleges kulcsként működik. Ő tárolja el az összefüggő rekordok adatait.
  + Gyermeknév és a Szülőnév szöveges adattípusként tárolandó.
  + Diákigazolvány pedig numerikus adattípus.
* **Cég**:
  + A Cég azonosítóval rendelkezik.
  + De ebben az esetben leginkább kapcsolótáblaként szolgál, ami összekapcsolja a Fogyasztókat a Székhelyekkel.
  + A másik két táblának a kulcsait tartalmazza, így azok idegen kulcsként értelmezhetőek.
  + A Cég tartalmaz egy szöveges adattípust, ami a Cég nevét tárolja.
  + Tartalmaz egy Dátum típusú adatot is a tábla, ami az alapítás dátumát tartalmazza.

## Táblák létrehozása:

**Székhely:**

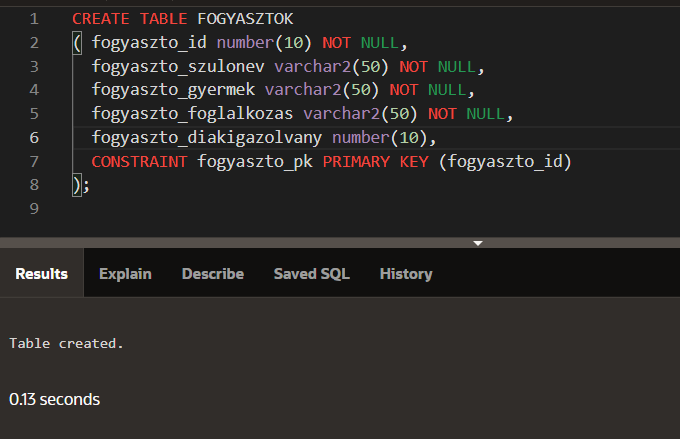


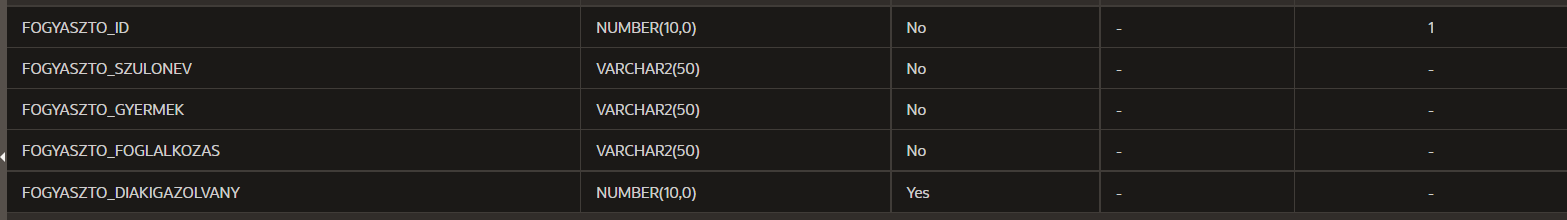
Kiegészítés mert a primary key nem lett definiálva.



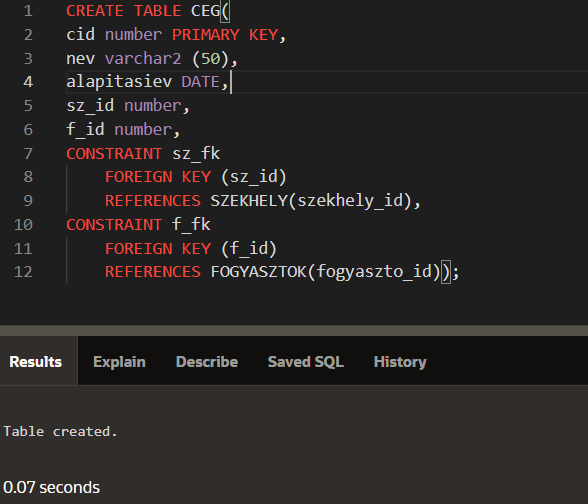


**Fogyasztók:**





**Cég:**





**Táblák feltöltése:**

**Szekhely:**

BEGIN

INSERT INTO SZEKHELY VALUES (1,1027,’Budapest’,’Varga köz’,18);

INSERT INTO SZEKHELY VALUES (2,1025,’Budapest’,’Kossuth utca’,7);

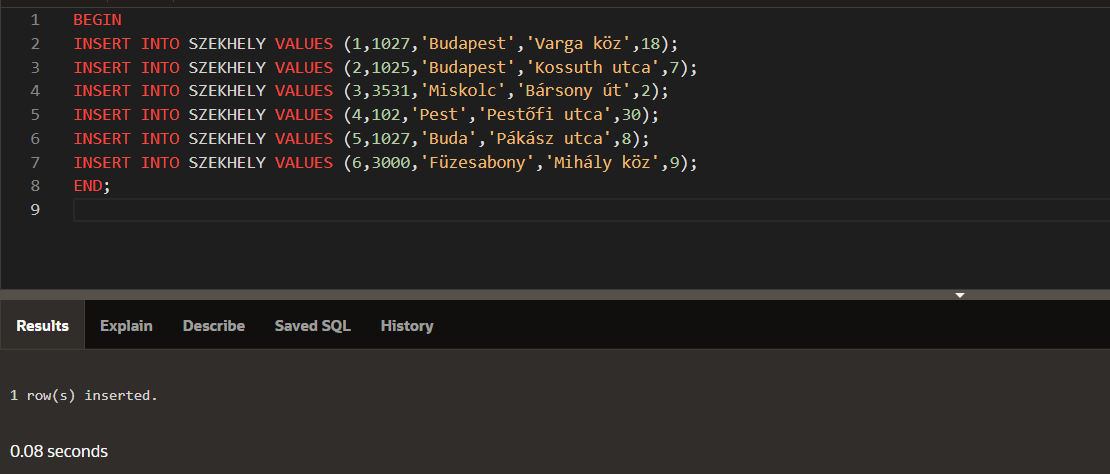
INSERT INTO SZEKHELY VALUES (3,3531,’Miskolc’,’Bársony út’,2);

INSERT INTO SZEKHELY VALUES (4,102,’Pest’,’Pestőfi utca’,30);

INSERT INTO SZEKHELY VALUES (5,1027,’Buda’,’Pákász utca’,8);

INSERT INTO SZEKHELY VALUES (6,3000,’Füzesabony’,’Mihály köz’,9);

END;



**Fogyaszto:**

BEGIN

INSERT INTO FOGYASZTOK VALUES (1,’Bánfi Orsolya’,’Kovács Péter’,’Asztalos’,7417175879);

INSERT INTO FOGYASZTOK VALUES (2, Tóth József’,’Tóth Miklós’,’Programozó’,7417175880);

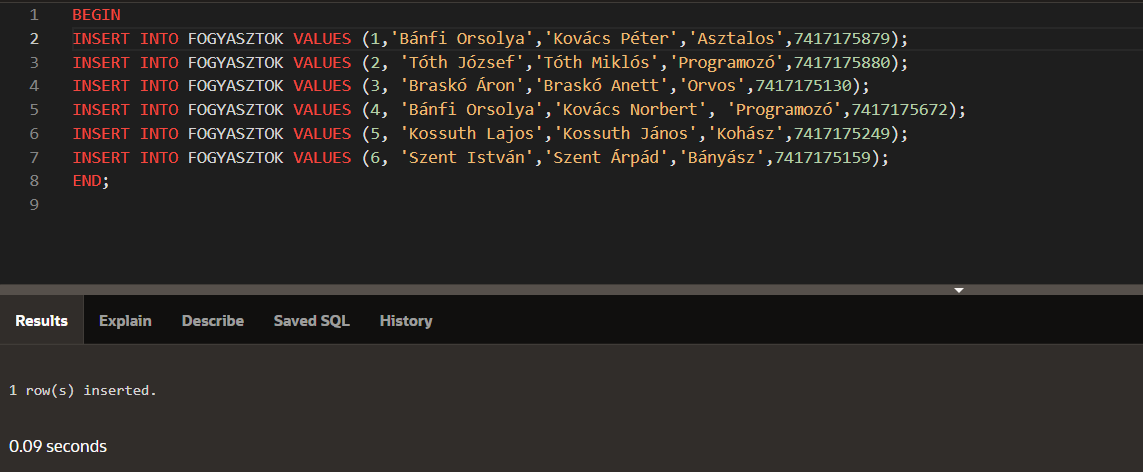
INSERT INTO FOGYASZTOK VALUES (3, ’Braskó Áron’,’Braskó Anett’,’Orvos’,7417175130);

INSERT INTO FOGYASZTOK VALUES (4, ’Bánfi Orsolya’,’Kovács Norbert, ’Programozó’,7417175672);

INSERT INTO FOGYASZTOK VALUES (5, ’Kossuth Lajos’,’Kossuth János’,’Kohász’,7417175249);

INSERT INTO FOGYASZTOK VALUES (6, ’Szent István’,’Szent Árpád’,’Bányász’,7417175159);

END;

****

**Ceg:**

BEGIN

INSERT INTO CEG VALUES (1,’Logiscool’, TO\_DATE (’20080521’, ’YYYYMMDD’),1 ,4);

INSERT INTO CEG VALUES (2,’WoodenSchool’, TO\_DATE (’20200118’, ’YYYYMMDD’),2 ,1);

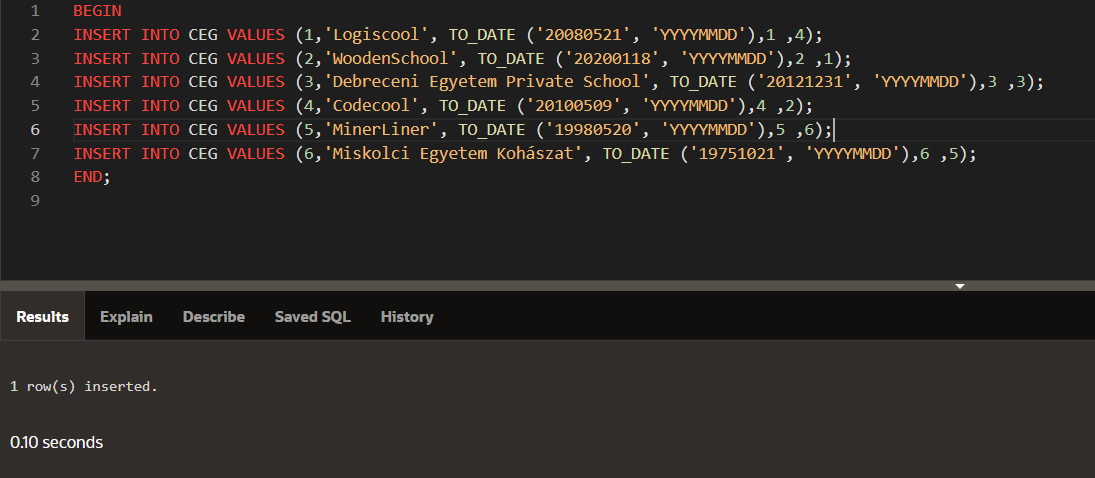
INSERT INTO CEG VALUES (3,’Debreceni Egyetem Private School’, TO\_DATE (’20121231’, ’YYYYMMDD’),3 ,3);

INSERT INTO CEG VALUES (4,’Codecool’, TO\_DATE (’20100509’, ’YYYYMMDD’),4 ,2);

INSERT INTO CEG VALUES (5,’MinerLiner’, TO\_DATE (’19980520’, ’YYYYMMDD’),5 ,6);

INSERT INTO CEG VALUES (6,’Miskolci Egyetem Kohászat’, TO\_DATE (’19751021’, ’YYYYMMDD’),6 ,5);

END;

****

**A Táblázatokkal néhány kapcsolatos függvény és lekérdezés**

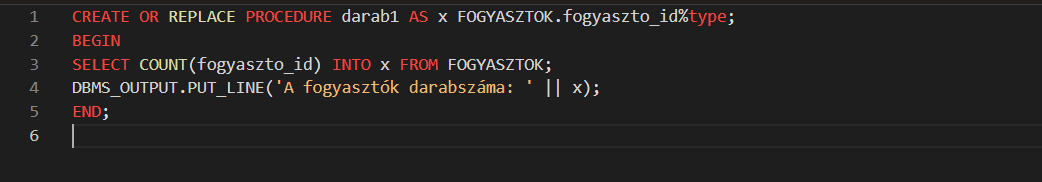
**Darab számláló:**

CREATE OR REPLACE PROCEDURE darab1 AS x FOGYASZTOK.fogyaszto\_id%type;

BEGIN

SELECT COUNT(fogyaszto\_id) INTO x FROM FOGYASZTOK;

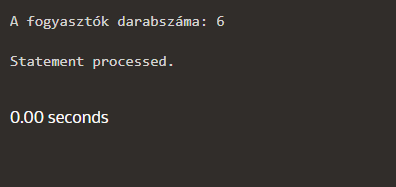
DBMS\_OUTPUT.PUT\_LINE(’A fogyasztók darabszáma: ’ || x);

END; 

BEGIN

darab1;

END;



**PL/SQL függvény, ami egy megadott gyermek alapján megadja milyen szakterületen tanul.**

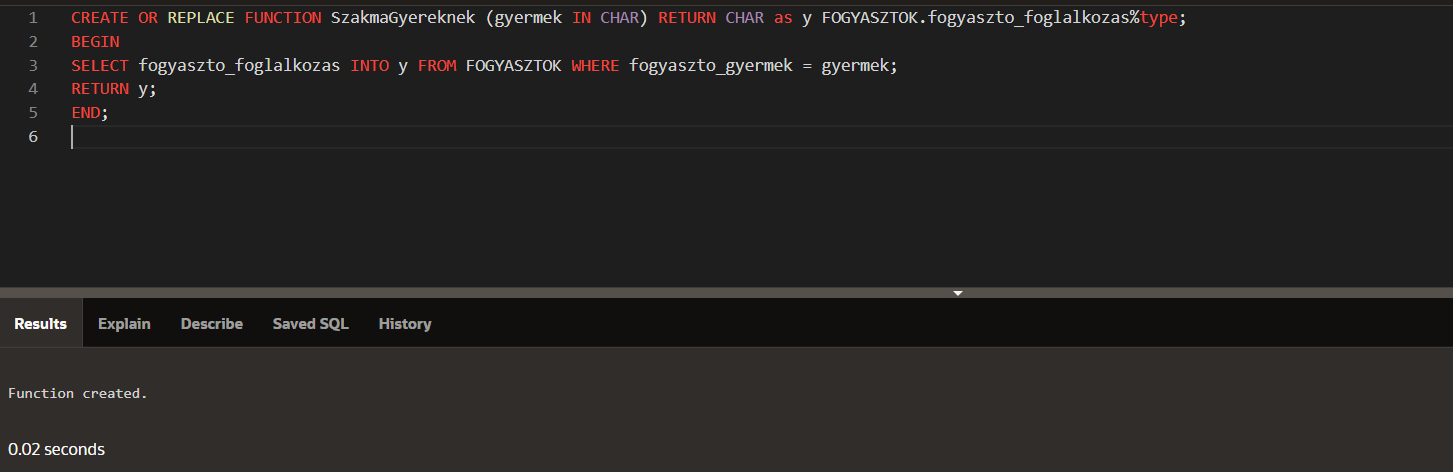
CREATE OR REPLACE FUNCTION SzakmaGyereknek (gyermek IN CHAR) RETURN CHAR AS y FOGYASZTOK.fogyaszto\_foglalkozas%type;

BEGIN

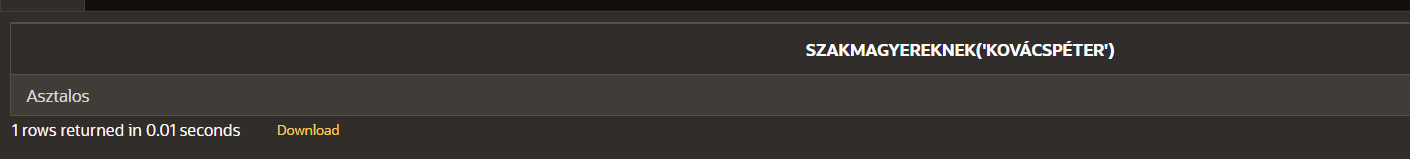
SELECT fogyaszto\_foglalkozas INTO y FROM FOGYASZTOK WHERE fogyaszto\_gyermek = gyermek;

RETURN y;

END;



SELECT SzakmaGyereknek(’Braskó Anett’) FROM FOGYASZTOK;



A **function** olyan, hogy vissza kell adnia egy visszatérési értéket

A **procedure** olyan, mint a C programozási nyelvben a void vagyis végrehajt valamit, de nem ad vissza értéket.

**A feladat kötelező lépései:**

**• tárolt eljárás adatok felvitelére,**

**SZÉKHELY**:

Megvizsgálom, hogy ha létezik már ilyen id-val egy székhely akkor nem sikeres a beszúrás. Ellenkező esetben felveszem.

create or replace procedure BeszurSzekhely(szid in number, irsz in number, szekhely\_varos in char, szekhely\_utca in char, hazszam in number) as

x number;

begin

select count(\*) into x from SZEKHELY where szekhely\_id = szid;

if x > 0 then

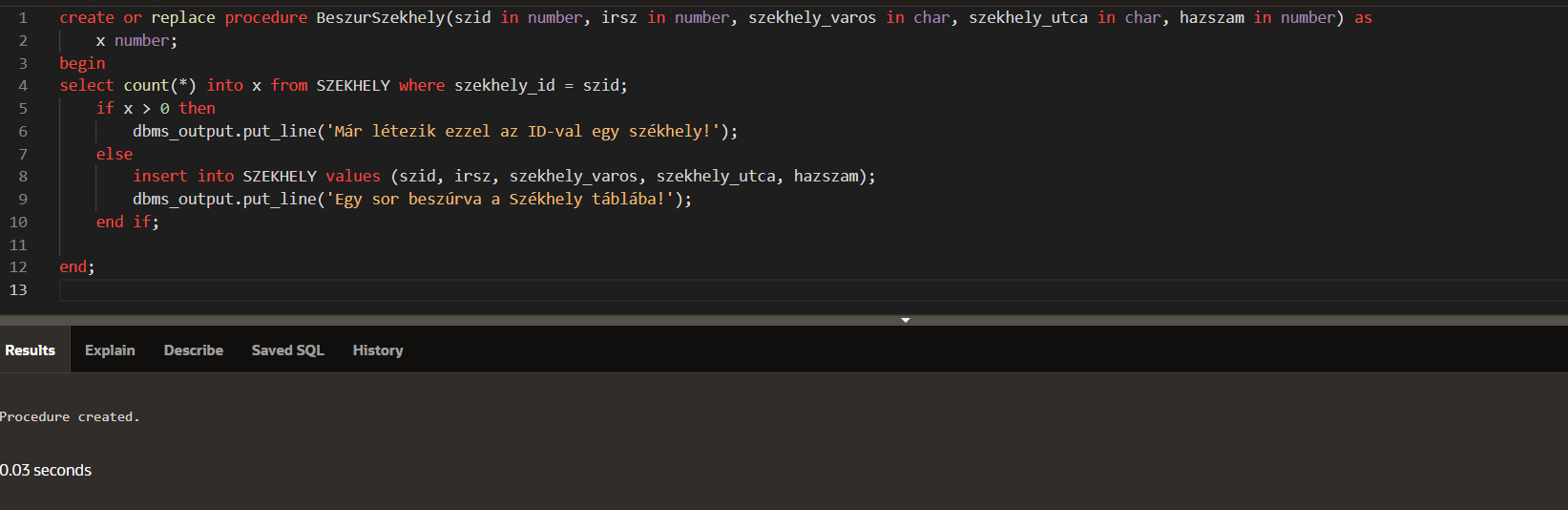
dbms\_output.put\_line('Már létezik ezzel az ID-val egy székhely!');

else

insert into SZEKHELY values (szid, irsz, szekhely\_varos, szekhely\_utca, hazszam);

dbms\_output.put\_line('Egy sor beszúrva a Székhely táblába!');

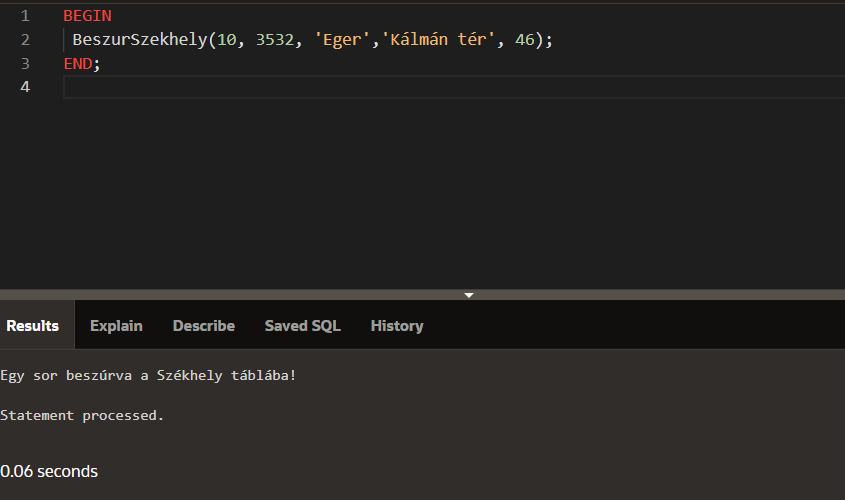
end if;

end;

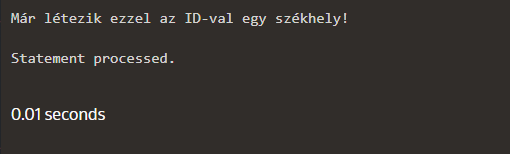
BEGIN

SELECT BeszurSzekhely(10, 3532, 'Eger','Kálmán tér', 46) FROM SZEKHELY;

END;



rossz esetben:



**FOGYASZTÓK:**

Megnézem létezik e már ezzel az id-val a fogyasztó, ha nem akkor megnézem, hogy ugyan az e a diákigazolványszáma, ha az sem hasonlít egyikre se akkor felveszem.

create or replace procedure BeszurFogyasztok(fid in number, sznev in char, gynev in char, ffoglalkozas in char, diakig in number) as

x number;

begin

select count(\*) into x from FOGYASZTOK where fogyaszto\_id = fid;

if x > 0 then

dbms\_output.put\_line('Már létezik ezzel az ID-val egy fogyasztó!');

else

select count(\*) into x from FOGYASZTOK where fogyaszto\_diakigazolvany = diakig;

if x > 0 then

dbms\_output.put\_line('Már létezik ezzel az igazolványszámmal egy fogyasztó!');

else

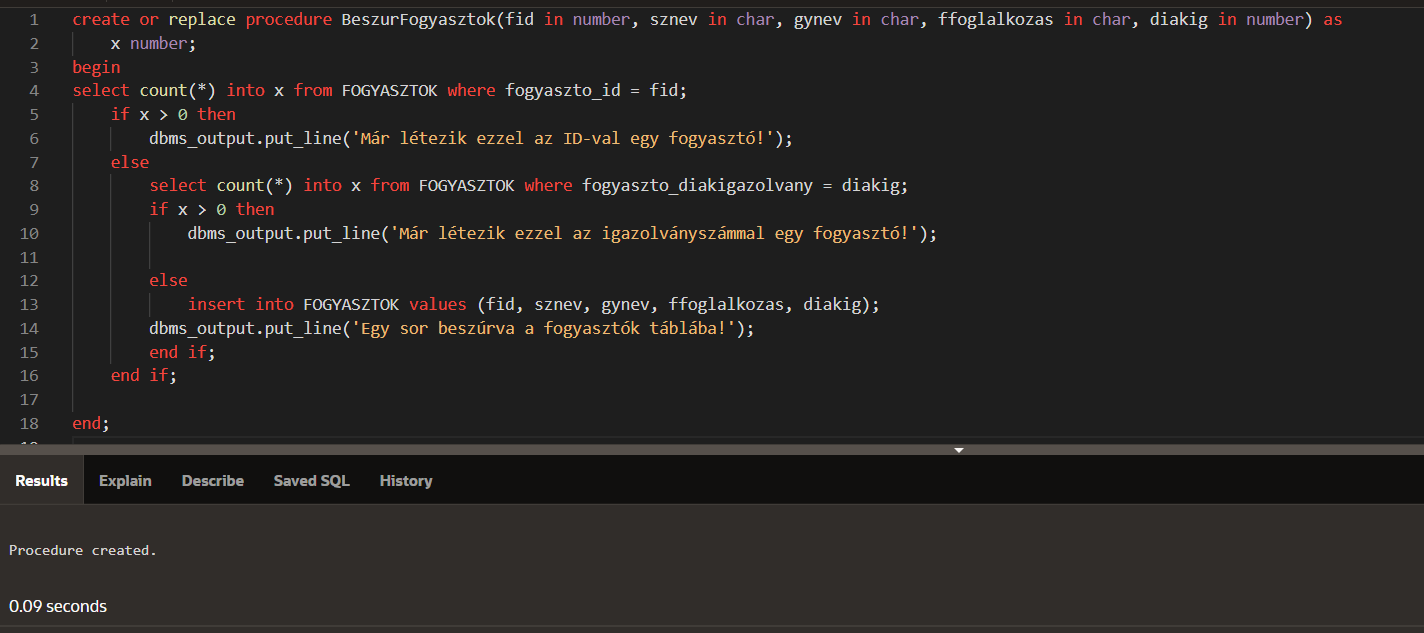
insert into FOGYASZTOK values (fid, sznev, gynev, ffoglalkozas, diakig);

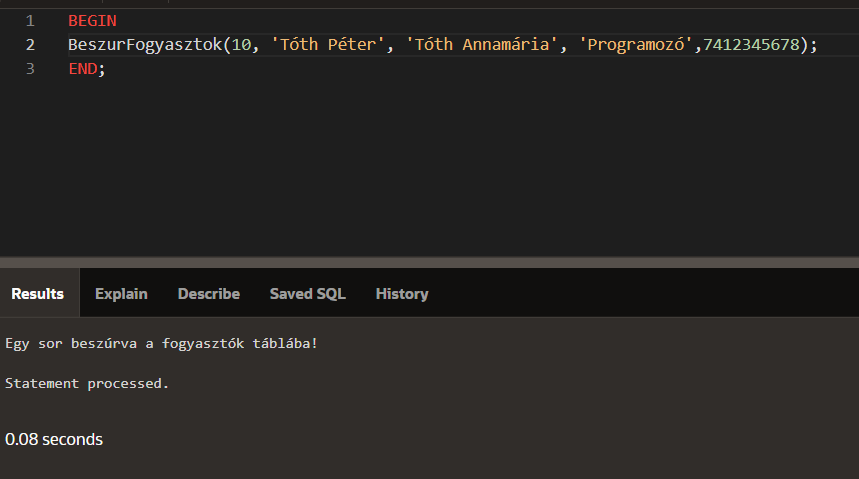
dbms\_output.put\_line('Egy sor beszúrva a fogyasztók táblába!');

end if;

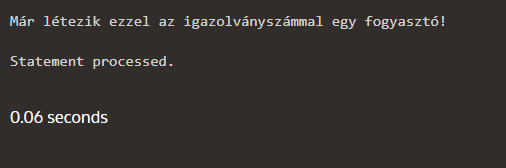
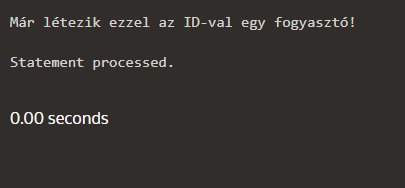
end if;

end;





rossz esetben:



**CÉG:**

Megvizsgáljuk ez a cég létezik e már ezzel az ID, ha nem megnézzük a székhely id-ját jó helyen helyezkedik e el. Illetve megvizsgáljuk, ha az a fogyasztó már létezik e azon a telephelyen.

create or replace procedure BeszurCeg(cegid in number, cnev in char, datum in date, szekid in number, fogyid in number) as

x number;

begin

select count(\*) into x from CEG where cid = cegid;

if x > 0 then

dbms\_output.put\_line('Már létezik ezzel az ID-val egy cég!');

else

select count(\*) into x from CEG where sz\_id = szekid;

if x > 0 then

dbms\_output.put\_line('Már létezik a cég ezzel a székhely id-val');

else

select count(\*) into x from CEG where f\_id = fogyid;

if x < 1 then

dbms\_output.put\_line('Nem létezik ez a fogyasztó id a cégnél!');

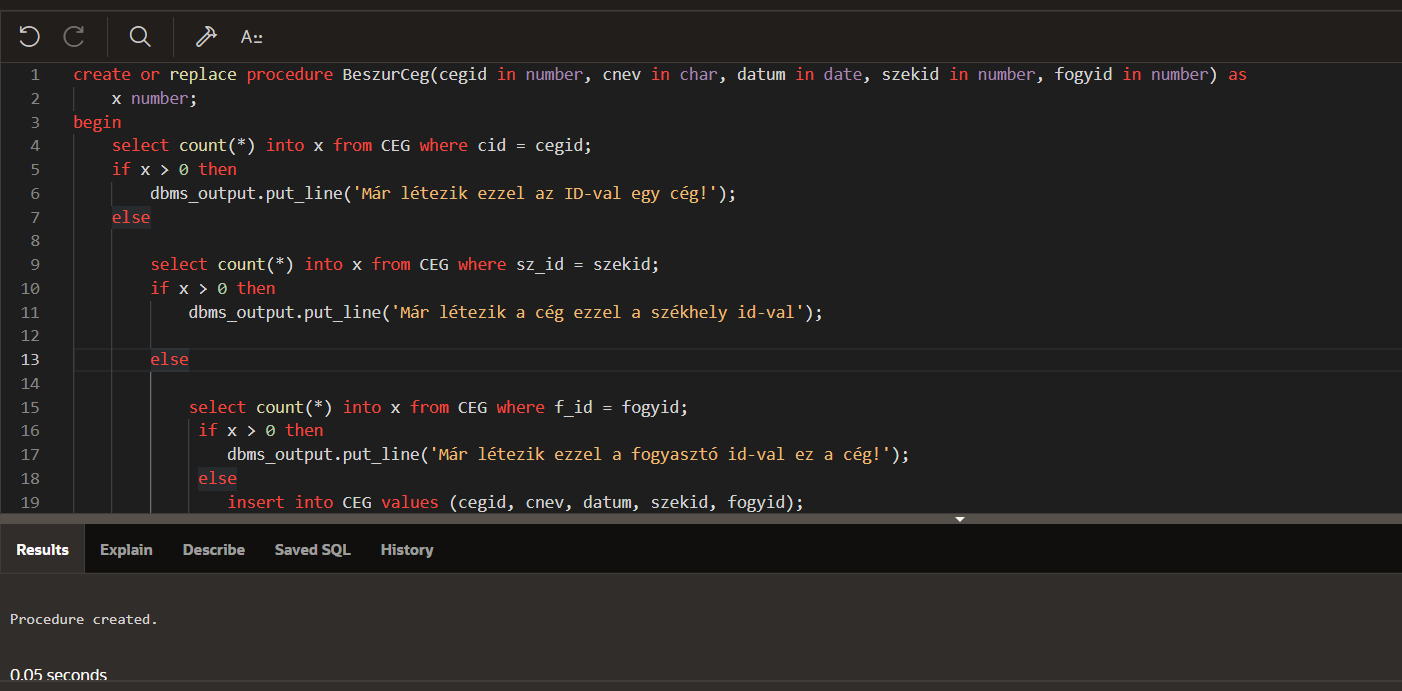
else

insert into CEG values (cegid, cnev, datum, szekid, fogyid);

dbms\_output.put\_line('Egy sor beszúrva a cég táblába!');

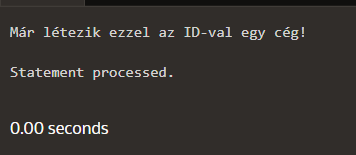
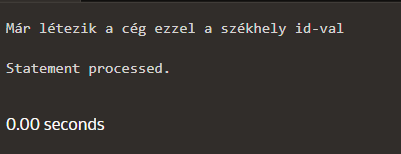
end if;

end if;

end if;  
end;



rossz esetben pl.:

• **tárolt eljárás adatok módosítására,**

FOGYASZTO

Megváltoztatom a megadott fogyaszto id-ja alapján a diákigazolvány számát.

CREATE OR REPLACE PROCEDURE FOGYASZTODiakMod (fogyid in number, diakig in number) as x number;

BEGIN

SELECT COUNT(\*) INTO x FROM FOGYASZTOK WHERE fogyaszto\_id = fogyid;

if x < 1 THEN

DBMS\_OUTPUT.PUT\_LINE('Nem létezik ezzel az ID-val a Fogyasztó!');

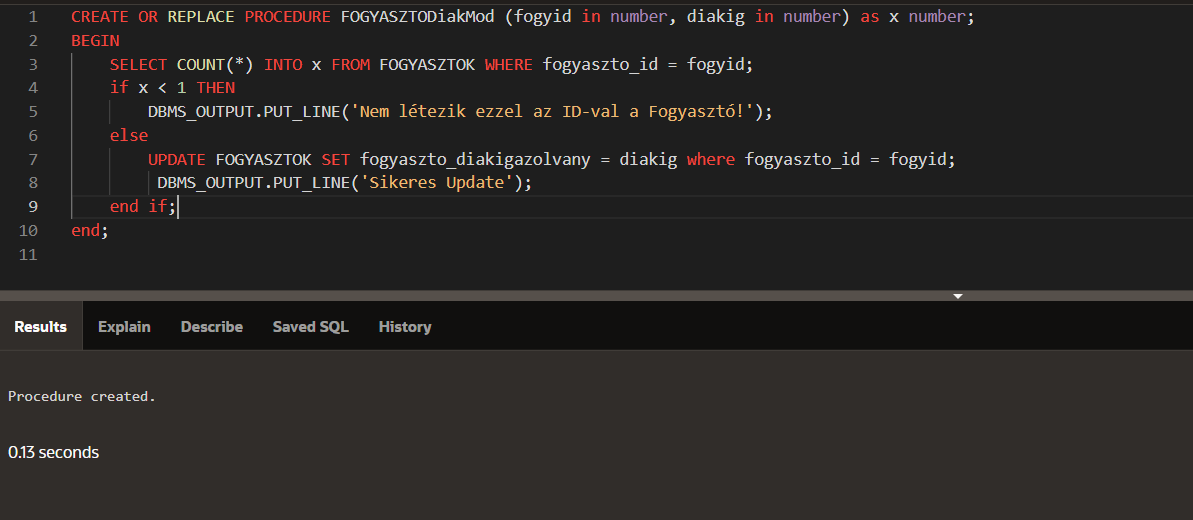
else

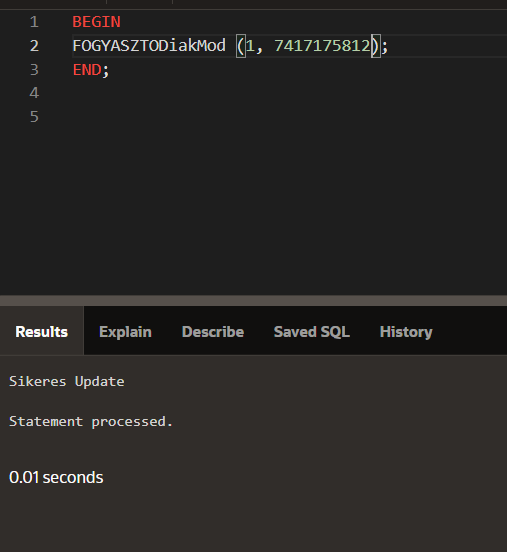
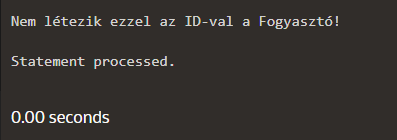
UPDATE FOGYASZTOK SET fogyaszto\_diakigazolvany = diakig where fogyaszto\_id = fogyid;

DBMS\_OUTPUT.PUT\_LINE('Sikeres Update');

end if;

end;



SZEKHELY:

A székhely id segítségével megváltoztatom a székhely irányítószámát, ha az az id nem létezik, nem sikeres a módosítás.

CREATE OR REPLACE PROCEDURE SZEKHELYirszMod (szekid in number, irszmod in number) as x number;

BEGIN

SELECT COUNT(\*) INTO x FROM SZEKHELY WHERE szekhely\_id = szekid;

if x < 1 THEN

DBMS\_OUTPUT.PUT\_LINE('Nem létezik ezzel az ID-val a Székhely!');

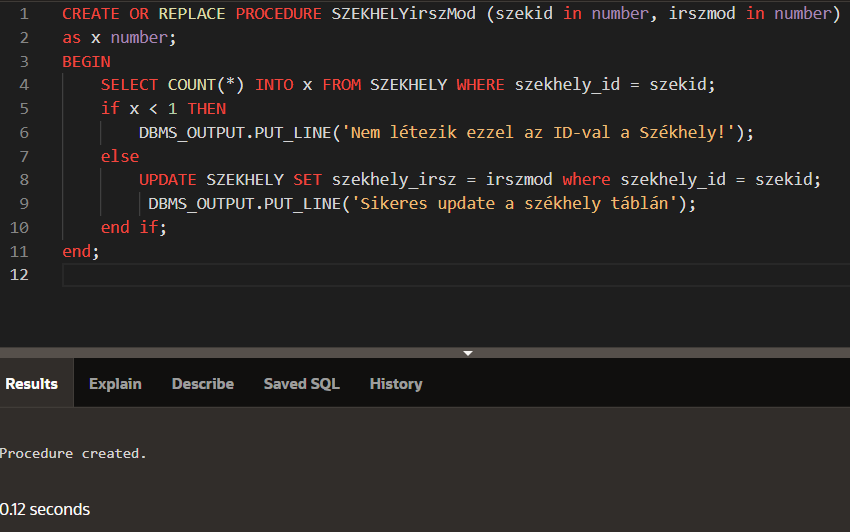
else

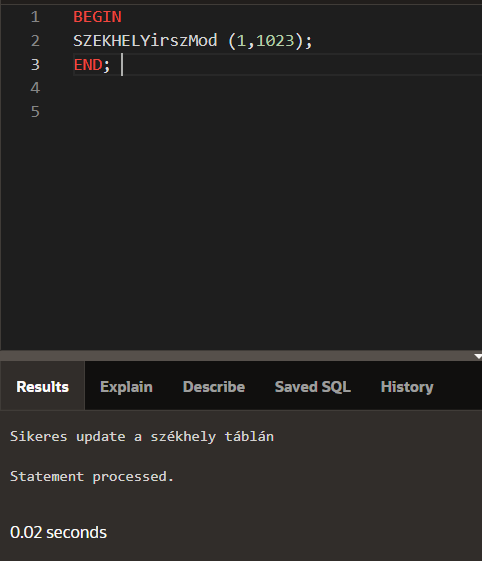
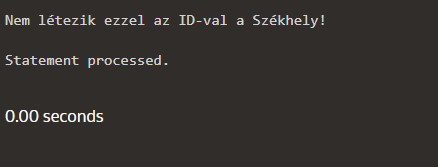
UPDATE SZEKHELY SET szekhely\_irsz = irszmod where szekhely\_id = szekid;

DBMS\_OUTPUT.PUT\_LINE('Sikeres update a székhely táblán');

end if;

end;



 rosz esetben pedig:

CEG:

a helyes cég id megadásával lehet a fogyasztó idját és a székhely idját is változtatni.

CREATE OR REPLACE PROCEDURE CEGszidfidMod (cegid in number, szid in number, fid in number)

as x number;

BEGIN

SELECT COUNT(\*) INTO x FROM CEG WHERE cid = cegid;

if x < 1 THEN

DBMS\_OUTPUT.PUT\_LINE('Nem létezik ezzel az ID-val a CÉG!');

else

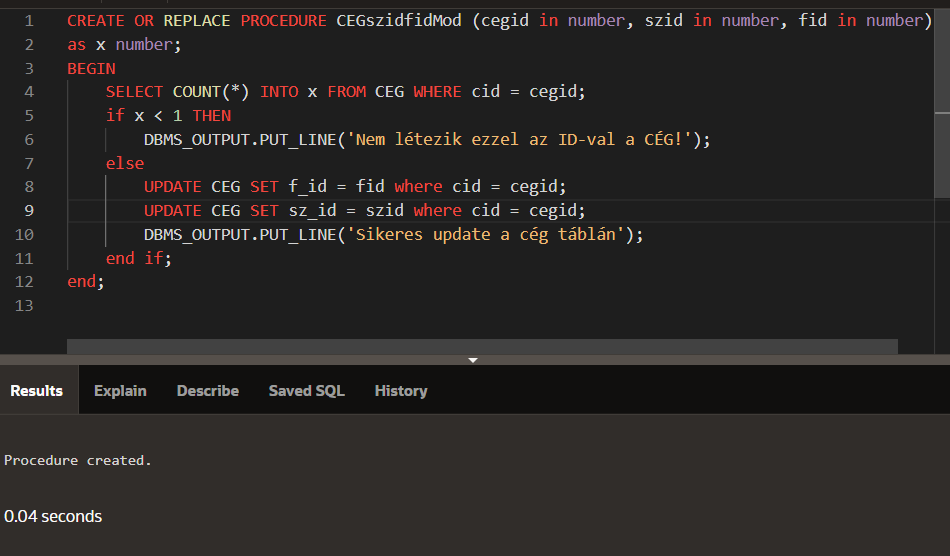
UPDATE CEG SET f\_id = fid where cid = cegid ;

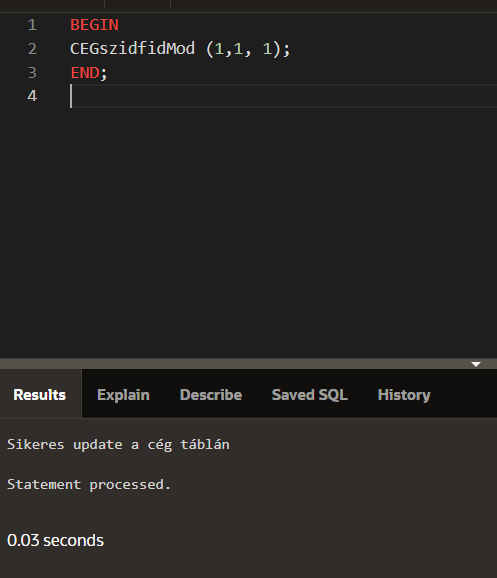
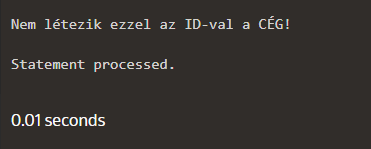
UPDATE CEG SET sz\_id = szid where cid = cegid;

DBMS\_OUTPUT.PUT\_LINE('Sikeres update a cég táblán');

end if;

end;



 Rossz esetben: 

**• tárolt eljárás adatok törlésére,**

FOGYASZTOK:

töröl egy paraméterként kapott fogyasztói név alapján, ha nincs ilyen akkor hibát ír ki a képernyőre de csak azokat tudja törölni amik nincsenek hivatkozva másik táblában

CREATE OR REPLACE PROCEDURE Del\_fogy (fogynev in char) as db number;

h\_nev exception;

begin

select count(\*) into db from FOGYASZTOK where fogyaszto\_gyermek = fogynev;

if db=0 then raise h\_nev;

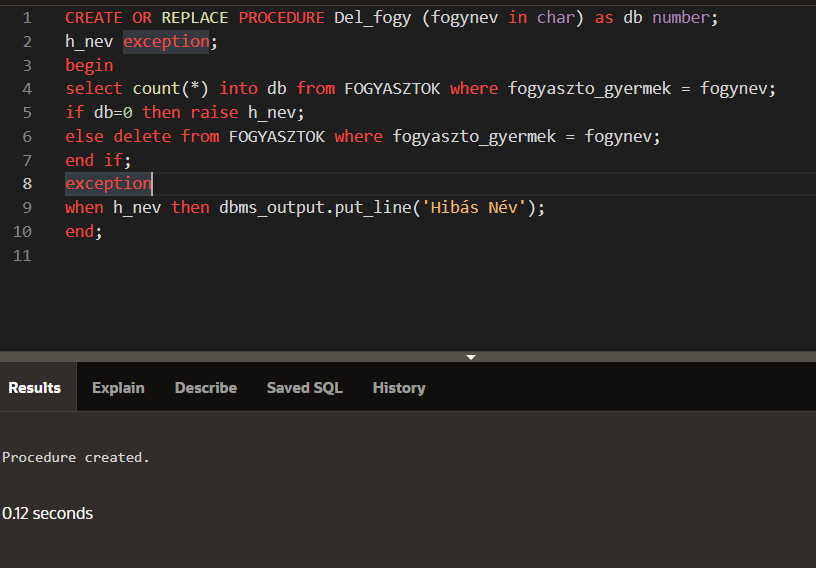
else delete from FOGYASZTOK where fogyaszto\_gyermek = fogynev;

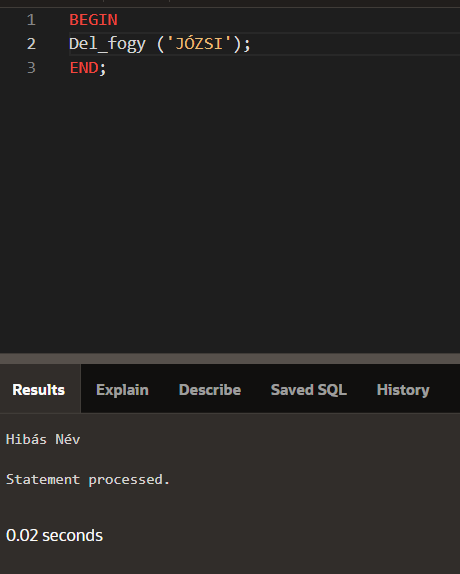
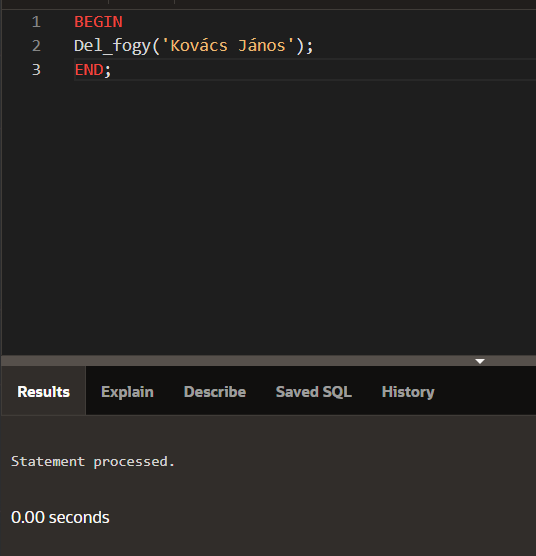
end if;

exception

when h\_nev then dbms\_output.put\_line(’Hibás Név’);

end;



ezért inkább másik módszert alkalmazok és az id alapján törlök rekordot és ha hivatkozok arra akkor nem lehet törölni és egy speciális hibát adok vissza.

CREATE OR REPLACE PROCEDURE Del\_fogy (fogyid in number) as db number;

h\_nev exception;

hiv\_sor exception;

begin

select count(\*) into db from FOGYASZTOK where fogyaszto\_id = fogyid;

if db=0 then raise h\_nev;

else

select count(\*) into db from CEG where f\_id = fogyid;

if db > 0 then raise hiv\_sor;

else delete from FOGYASZTOK where fogyaszto\_id = fogyid;

end if;

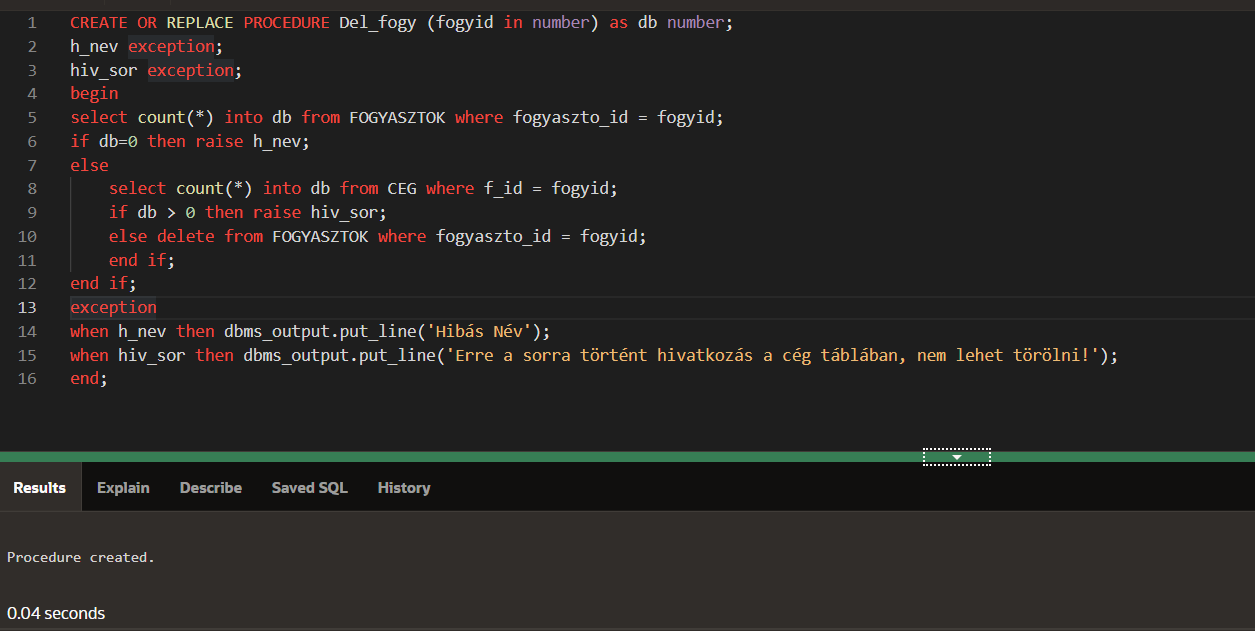
end if;

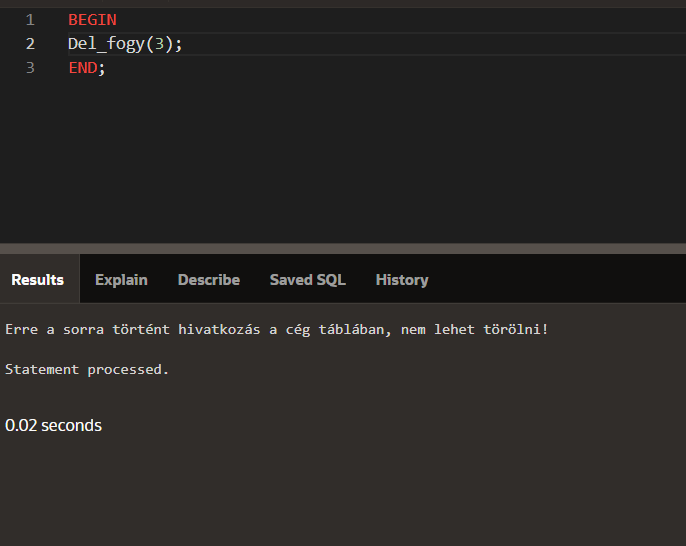
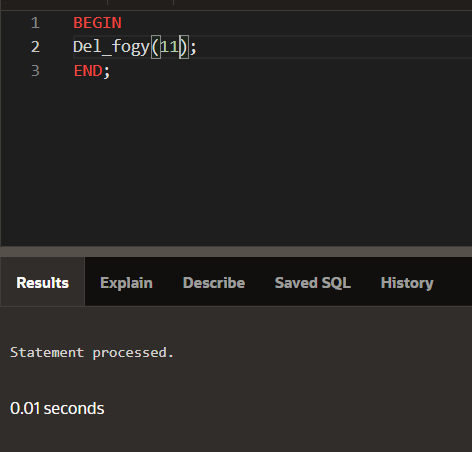
exception

when h\_nev then dbms\_output.put\_line('Hibás Fogyasztó ID');

when hiv\_sor then dbms\_output.put\_line('Erre a sorra történt hivatkozás a cég táblában, nem lehet törölni!');

end;



SZEKHELY:

CREATE OR REPLACE PROCEDURE Del\_szek (szekid in number) as db number;

h\_irsz exception;

hiv\_sor exception;

begin

select count(\*) into db from SZEKHELY where szekhely\_id = szekid;

if db=0 then raise h\_irsz;

else

select count(\*) into db from CEG where sz\_id = szekid;

if db > 0 then raise hiv\_sor;

else delete from SZEKHELY where szekhely\_id = szekid;

end if;

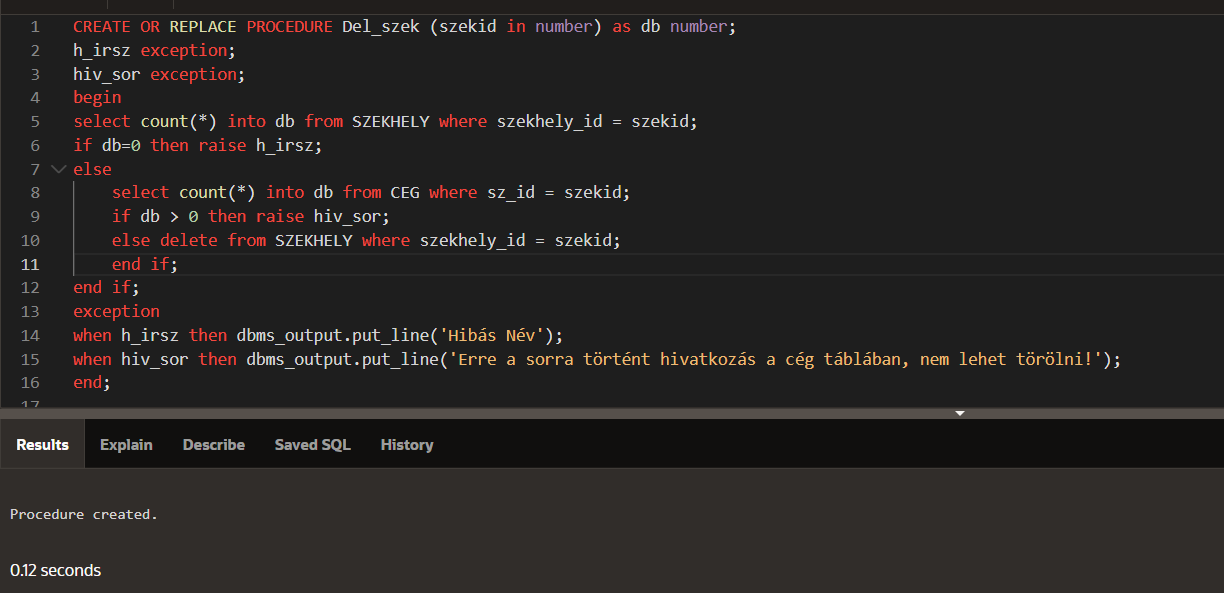
end if;

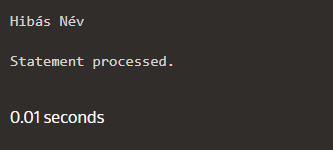
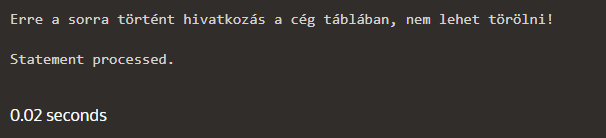
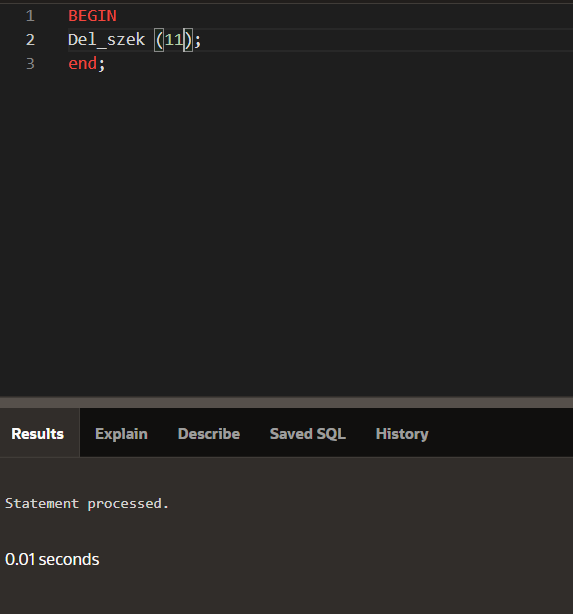
exception

when h\_irsz then dbms\_output.put\_line('Hibás Székhely ID');

when hiv\_sor then dbms\_output.put\_line('Erre a sorra történt hivatkozás a cég táblában, nem lehet törölni!');

end;



CEG:

CREATE OR REPLACE PROCEDURE Del\_ceg (cegid in number) as db number;

h\_datum exception;

begin

select count(\*) into db from CEG where cid = cegid;

if db=0 then raise h\_datum;

else

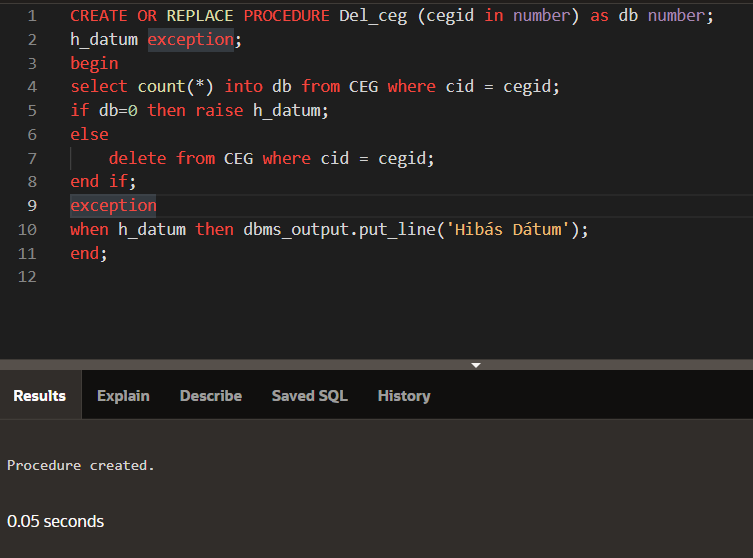
delete from CEG where cid = cegid;

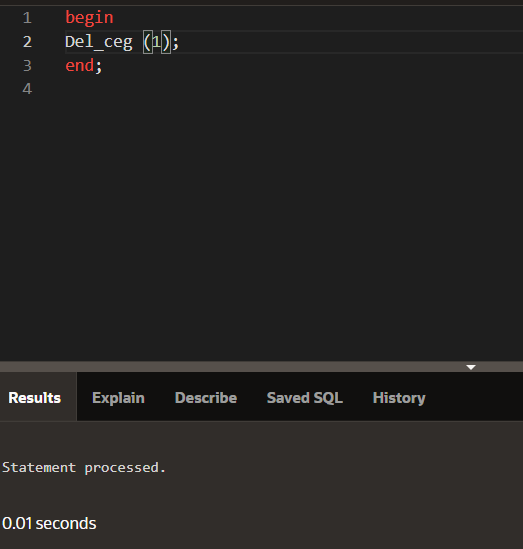
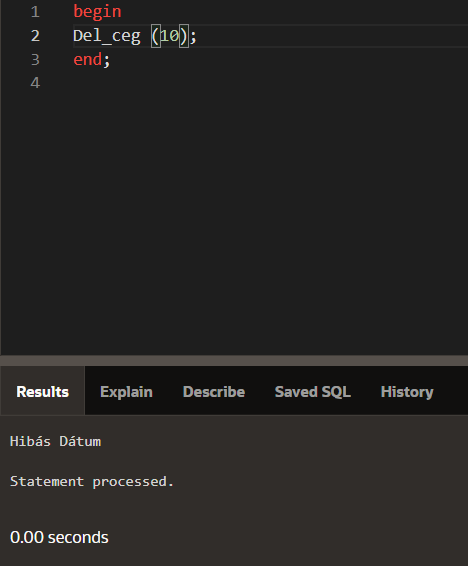
end if;

exception

when h\_datum then dbms\_output.put\_line('Hibás Cég ID');

end;





**• tárolt függvény adott rekord mezőinek lekérdezésére,**

Kurzor attributomok:

%rowcount – értéket ad egy pillanatra beolvasott rekord számára és ennek a száma.

%found – értéke igaz, ha a fetch utasítás után nem értelmezhető. Az első fetch utásítás kiadása előtt mindig az értéke NULL.

%notfound – értéke igaz, ha az adott fetch utasítás már nem talál a táblában rekordot Csak egy már kiadott fetch utasítás után értelmezhető. Az első fetch utasítás kiadása előtt értéke null.

%isopen – ha a kurzor nyitott akkor az értéke igaz.

SZEKHELY:

Adott várossal rendelkező város irányítószámát növelem egy adott százalékkal. for ciklus segítségével oldom meg ezt a feladatot.

create or replace function szekhely\_irsznov (varosbe in char, ert in int) return number as

cursor cur\_a is select \* from SZEKHELY where szekhely\_varos = varosbe

for update of szekhely\_irsz;

db number :=0;

begin

for cv in cur\_a loop

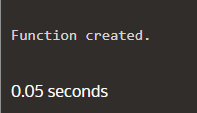
update SZEKHELY set szekhely\_irsz = cv.szekhely\_irsz\*(1+ert/100) where current of cur\_a;

db:= db+1;

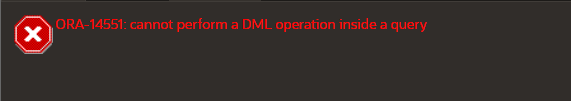
end loop;

return db;

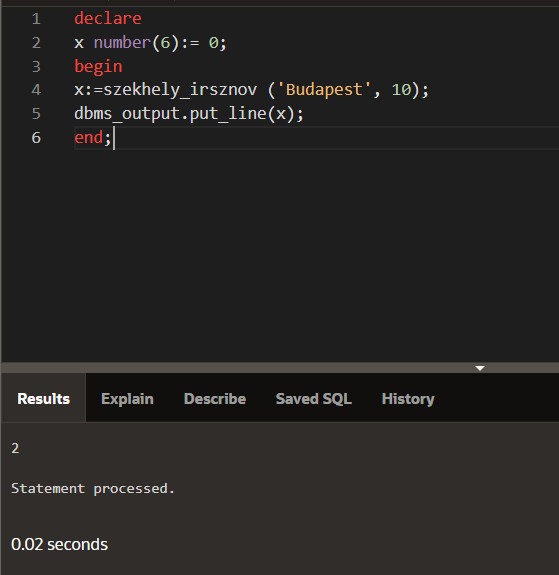
end;



select szekhely\_irsznov ('Budapest', 10) from dual;



nem megfelelő ez a futtatás ezért ezt kell használni.



10%-al növeltem a budapesti irányítószámokat.

Illetve megszámolom hány ilyen rekord volt amit változtattam.

Kiírom az a Székhely elhelyezkedésének a városát.

**implicit kurzor használata**: hasonlóképpen csak implicit verzióban.

create or replace procedure szekhely\_irsznovim (varosbe in char, ert in int) is db number:=0;

begin

for cv in (select \* from SZEKHELY where szekhely\_varos = varosbe) loop

dbms\_output.put\_line('In: '||cv.szekhely\_varos||'-'||cv.szekhely\_irsz||'-'||cv.szekhely\_id);

update SZEKHELY set szekhely\_irsz=cv.szekhely\_irsz\*(1+ert/100) where SZEKHELY.szekhely\_id = cv.szekhely\_id;

select \* into cv from SZEKHELY where szekhely\_id = cv.szekhely\_id;

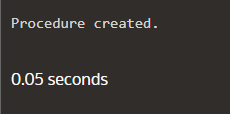
dbms\_output.put\_line('Out: '||cv.szekhely\_varos||'-'||cv.szekhely\_irsz||'-'||cv.szekhely\_id);

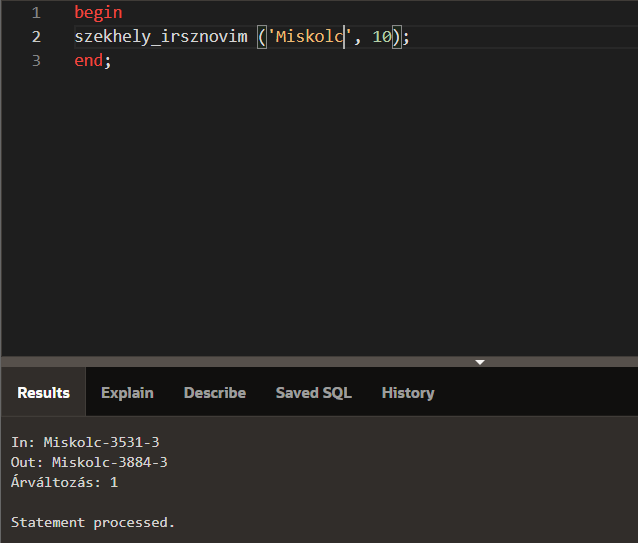
db:=db+1;

end loop;

dbms\_output.put\_line('Árváltozás: '||db);

end;





FOGYASZTOK:

Kurzor segítségével a FOGYASZTÓK táblában lekérdezem azokat az adatokat, amik, egy bizonyos típusú szakma paraméter alapján kilistázom azokat.

**explicit verzióban:**

create procedure FogyInSzakmaLista3 (szakma in char) is cursor cur\_a is select \* from FOGYASZTOK where fogyaszto\_foglalkozas = szakma;

a cur\_a%rowtype;

begin

open cur\_a;

loop

fetch cur\_a into a;

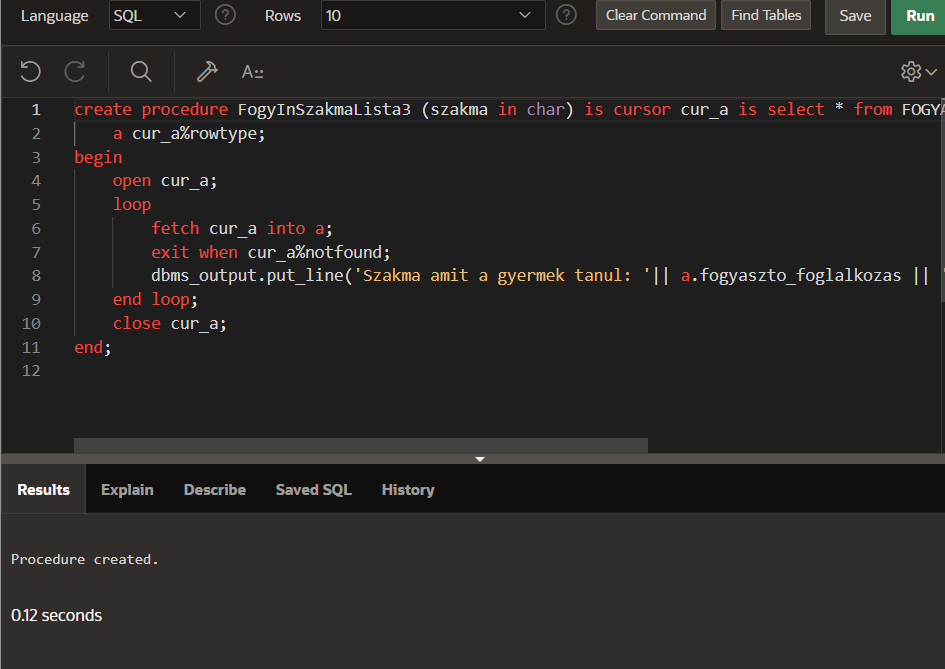
exit when cur\_a%notfound;

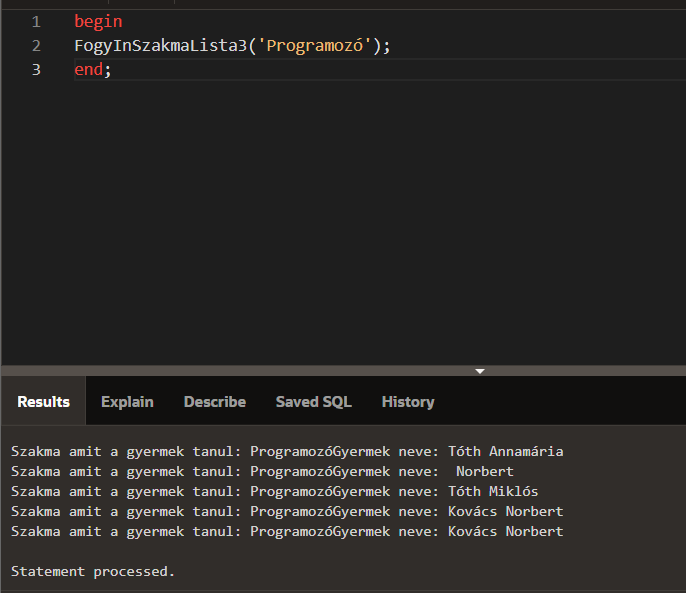
dbms\_output.put\_line('Szakma amit a gyermek tanul: '|| a.fogyaszto\_foglalkozas || 'Gyermek neve: '|| a.fogyaszto\_gyermek);

end loop;

close cur\_a;

end;





CEG:

Kurzor segítségével lekérdezem a CÉG táblában az egy bizonyos index határok között minden adatott.

create or replace procedure IndexAlapjanKiir (index1 in int, index2 in int)as

cursor cur is select \* from CEG where cid >= index1 and cid <= index2;

cv cur%rowtype;

begin

open cur;

fetch cur into cv;

if cur%notfound then

dbms\_output.put\_line('Nincs találat!');

else

dbms\_output.put\_line('Cégneve:' || cv.nev || ' Alapítás dátuma: '|| cv.alapitasiev || ' index :' || cv.cid);

loop

fetch cur into cv;

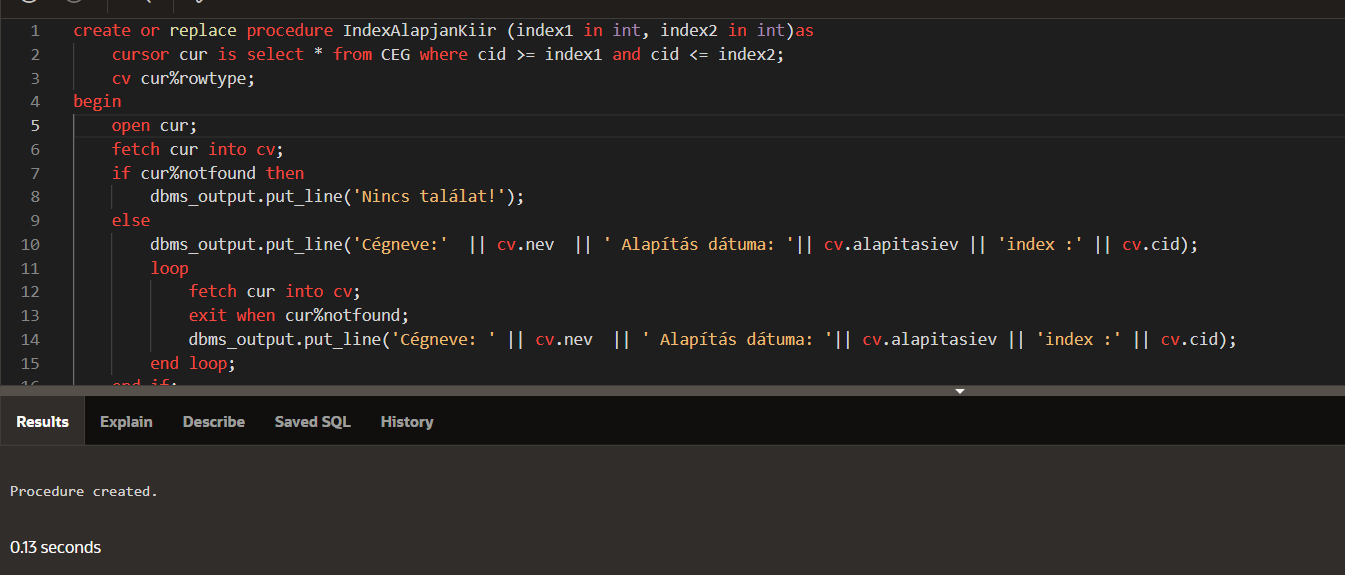
exit when cur%notfound;

dbms\_output.put\_line('Cégneve: ' || cv.nev || ' Alapítás dátuma: '|| cv.alapitasiev || ' index :' || cv.cid);

end loop;

end if;

close cur;

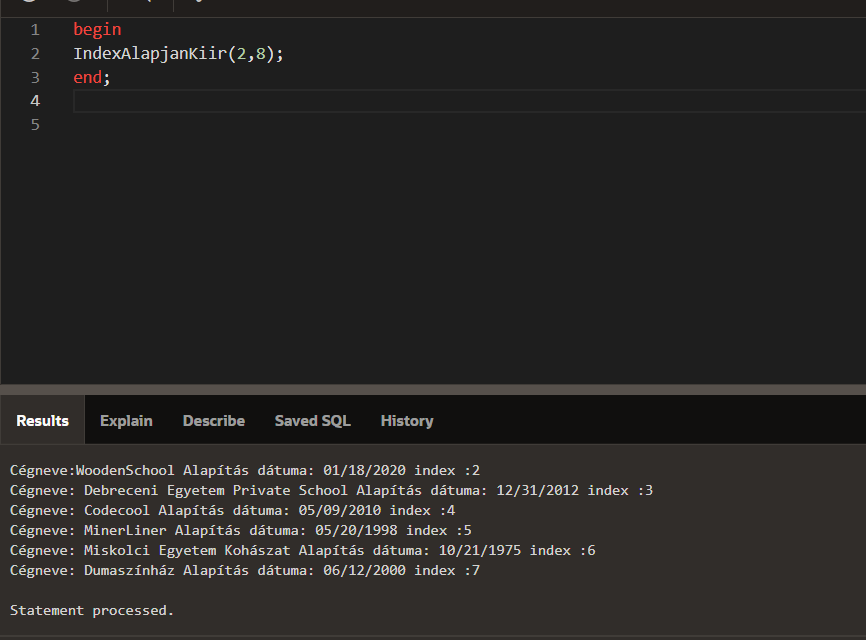
end;

**futtatás:**

begin

IndexAlapjanKiir(2,8);

end;



• **tárolt függvény adott feltételű rekordok aggregált értékének lekérdezésére,**

**Cégre:**

Kiválasztok egy adott azonosítót és megkeresem rá a megfelelő cég nevét:

CREATE OR REPLACE FUNCTION CegIDKeres (cegazonosito IN NUMBER) RETURN CHAR AS x number;

vi char(100) :='Nem létező azonosító!';

y char(40);

BEGIN

SELECT COUNT(\*) INTO x FROM CEG WHERE cid = cegazonosito;

IF x >= 1 THEN

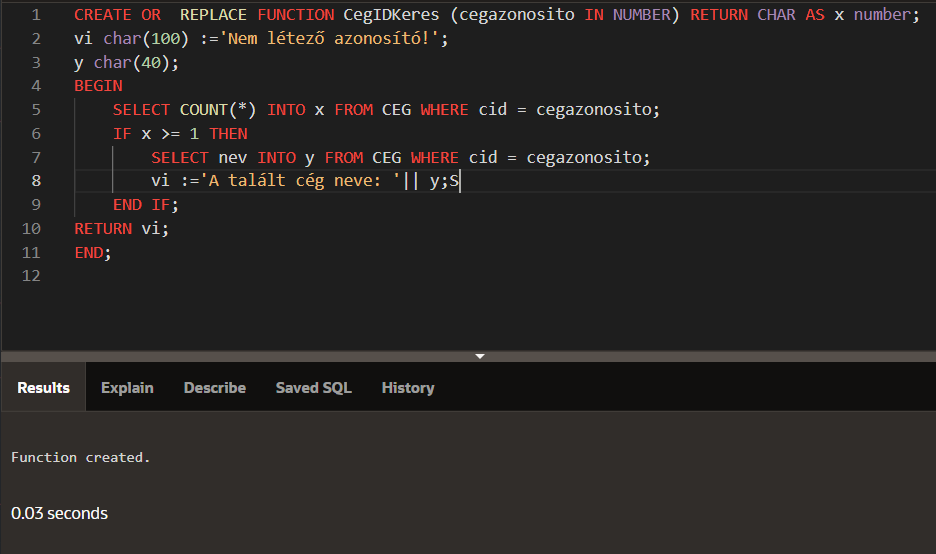
SELECT nev INTO y FROM CEG WHERE cid = cegazonosito;

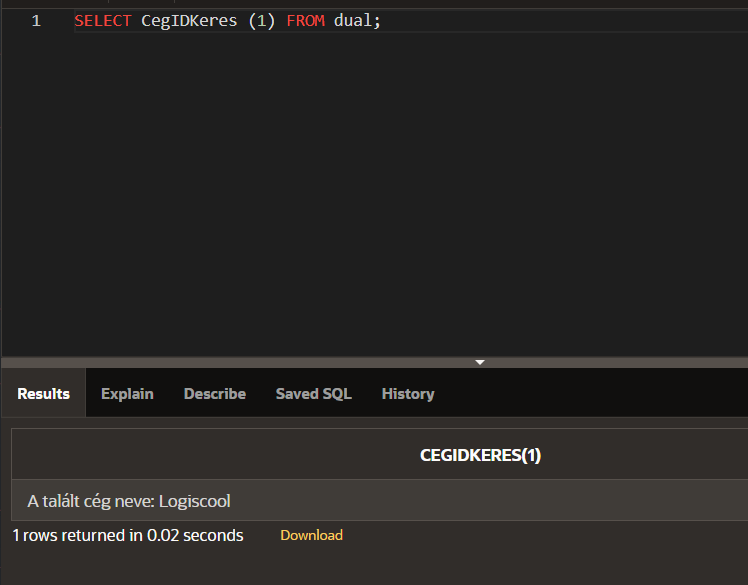
vi :='A talált cég neve: '|| y;

END IF;

RETURN vi;

END;

Az x változó a bekért azonosítóval rendelkező sorok számát tárolja és az y a cég neve tárolására alkalmazandó.  




**Székhely:**

Tárolt program, ami töröl egy paraméterként megadott id alapján az adott székhely rekordot.

Ha nincs ilyen hiba üzenetet kapunk.

create or replace procedure del\_SZid (idbe in number) as db number;

h\_roszid exception;

begin

select count(\*) into db from SZEKHELY where szekhely\_id = idbe;

if db=0 then raise h\_roszid;

else

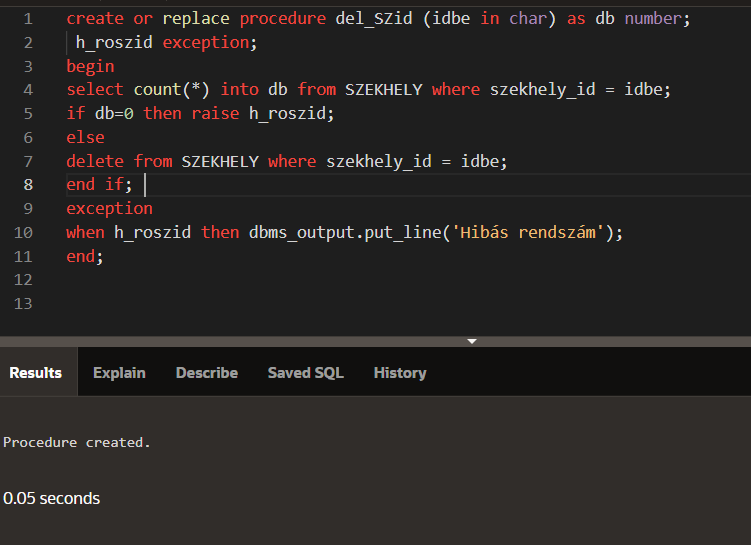
delete from SZEKHELY where szekhely\_id = idbe;

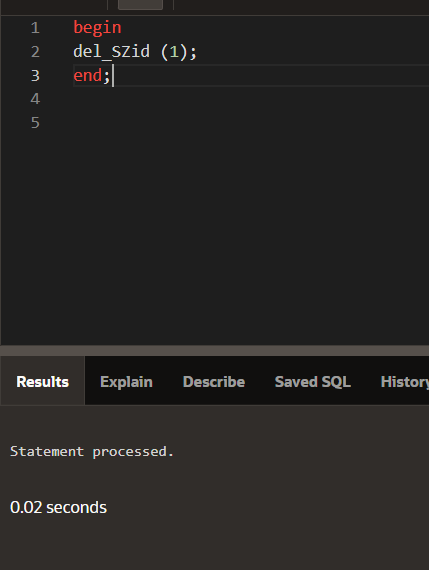
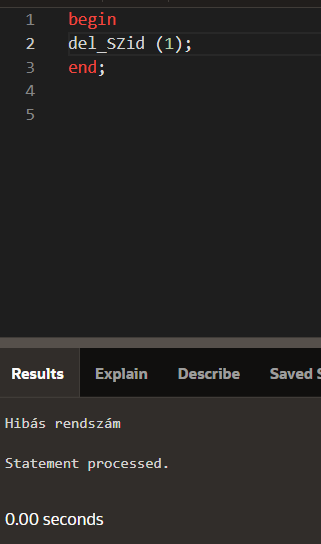
end if;

exception

when h\_roszid then dbms\_output.put\_line('Hibás rendszám');

end;



 másodjára -> 

**Fogyasztók:**

Kiválasztok egy adott azonosítót és megkeresem rá a megfelelő fogyasztó diáákigazolványszámát:

CREATE OR REPLACE FUNCTION FogyDiakKeres (fogyid IN NUMBER) RETURN CHAR AS x number;

idvalasz char(100) :='Nem létező azonosító!';

ig number(10);

BEGIN

SELECT COUNT(\*) INTO x FROM FOGYASZTOK WHERE fogyaszto\_id = fogyid;

IF x >= 1 THEN

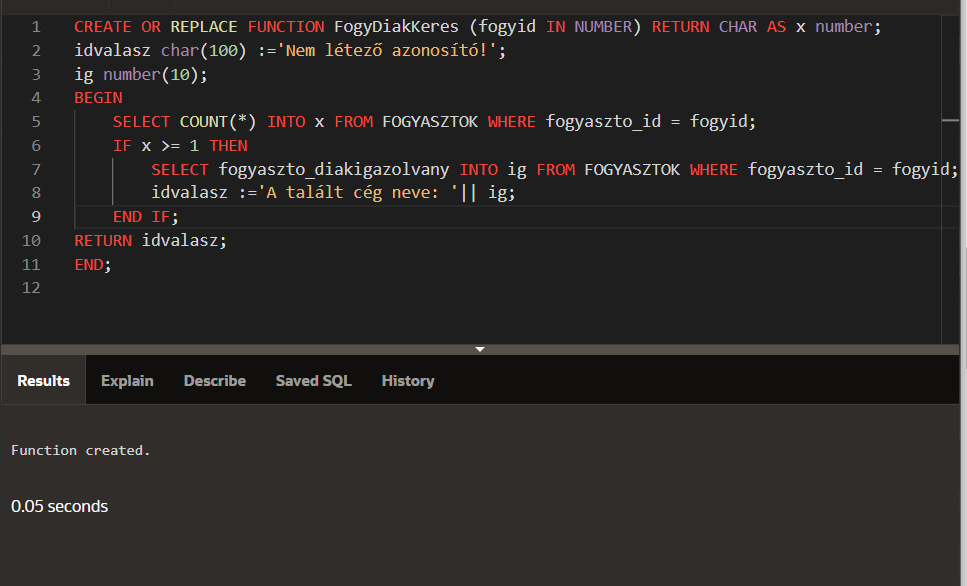
SELECT fogyaszto\_diakigazolvany INTO ig FROM FOGYASZTOK WHERE fogyaszto\_id = fogyid;

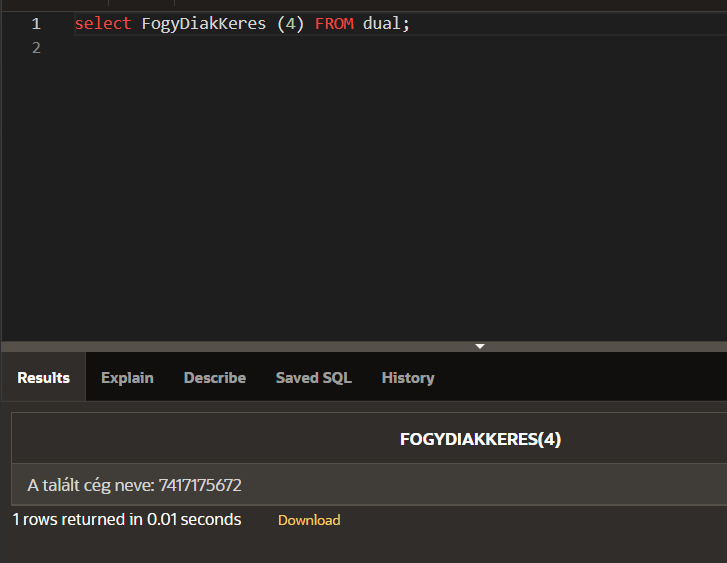
idvalasz :='A talált cég neve: '|| ig;

END IF;

RETURN idvalasz;

END;





**• tárolt csomag készítése egy tábla funkcióinak összefogására (legyen benne a táblához készített összes tárolt alprogram, de legalább két darab),**

**Székhely táblán csomag:**

Készítek egy ModSZ tárolt eljárást, ami módosítja egy paraméterként megadott ID alapján az irányítószámot egy paraméterként megadott értékre.

Majd egy tárolt függvényt hozok létre KerV mely visszaadja paraméterként adott székhelyek városát.

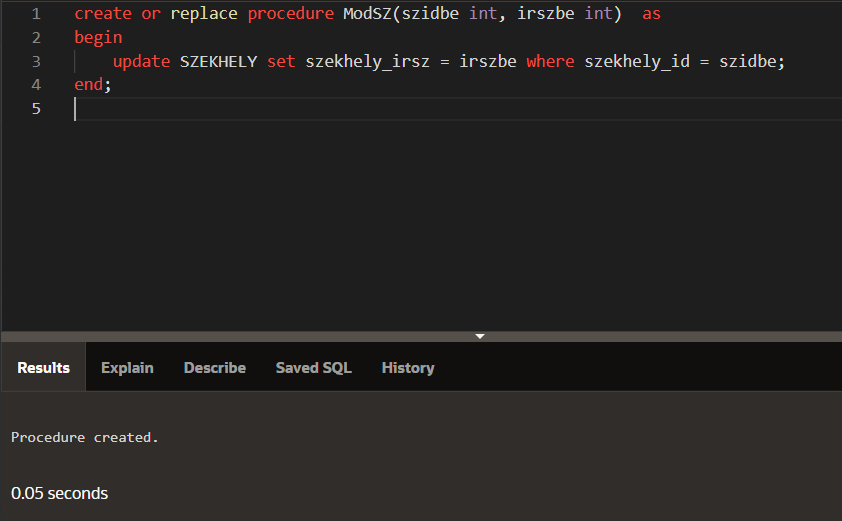
A két alprogramot elhelyezem egy csomagban, aminek a neve SZEKHELY\_cs.

create or replace procedure ModSZ(szidbe int, irszbe int) as

begin

update SZEKHELY set szekhely\_irsz = irszbe where szekhely\_id = szidbe;

end;



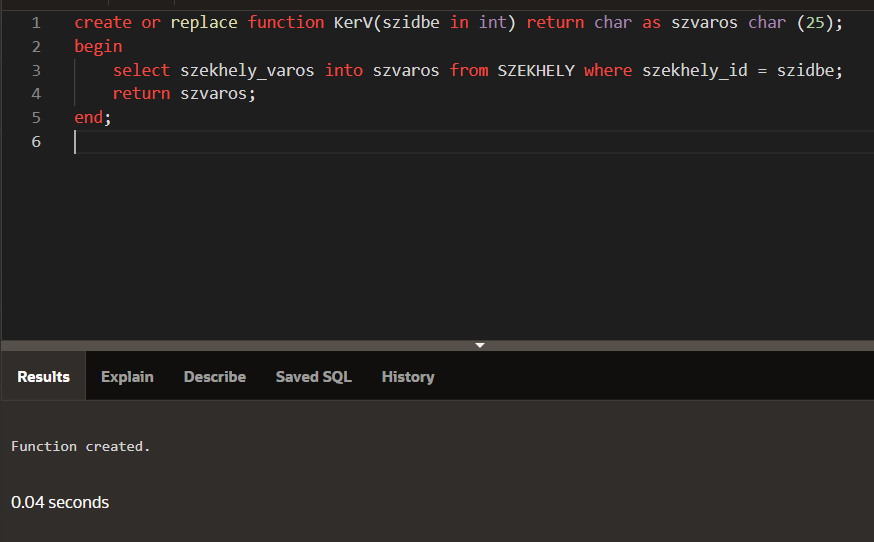
create or replace function KerV(szidbe in int) return char as szvaros char (25);

begin

select szekhely\_varos into szvaros from SZEKHELY where szekhely\_id = szidbe;

return szvaros;

end;



create or replace package SZEKHELY\_cs as

procedure ModSZ(szidbe int, irszbe int);

function KerV(szidbe in int) return char;

end SZEKHELY\_cs;

create or replace package body SZEKHELY\_cs as

procedure ModSZ(szidbe int, irszbe int) as

begin

update SZEKHELY set szekhely\_irsz = irszbe where szekhely\_id = szidbe;

end;

function KerV(szidbe in int) return char as szvaros char(25);

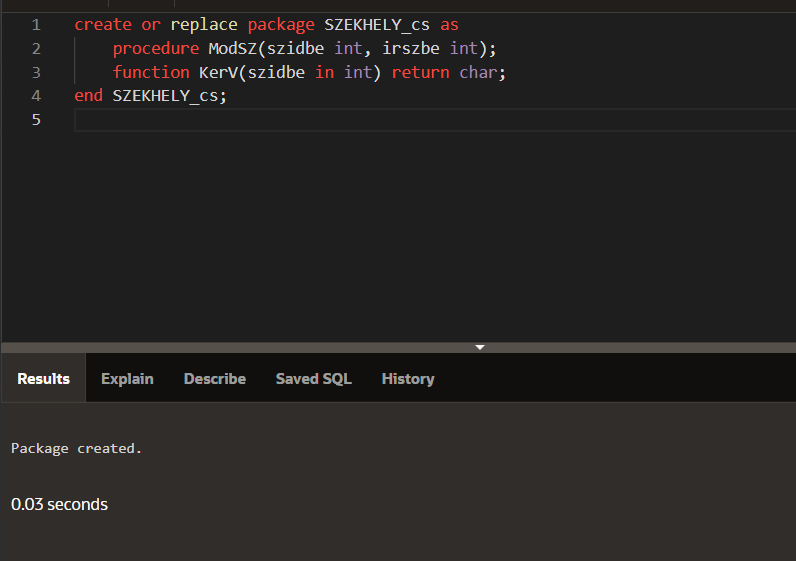
begin

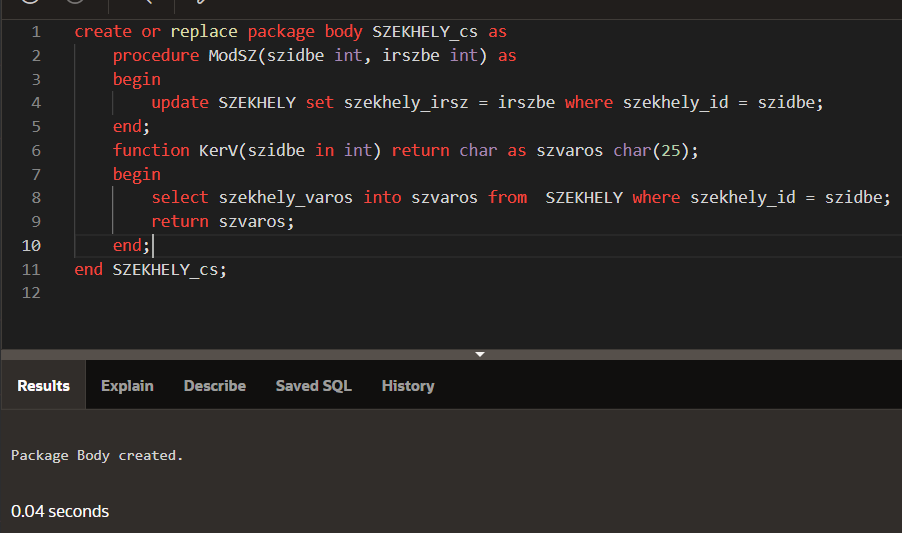
select szekhely\_varos into szvaros from SZEKHELY where szekhely\_id = szidbe;

return szvaros;

end;

end SZEKHELY\_cs;





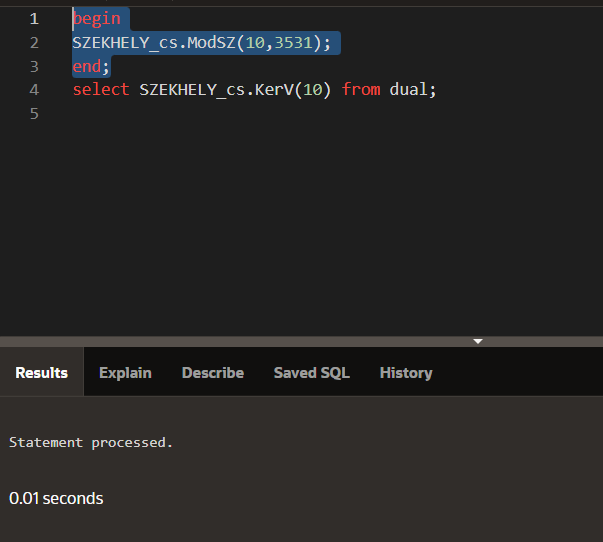
**futtatás**:

begin

SZEKHELY\_cs.ModSZ(10,3531);

end;

select SZEKHELY\_cs.KerV(10) from dual;





**Fogyasztó táblán csomag:**

Készítek egy ModF tárolt eljárást, ami módosítja egy paraméterként megadott ID alapján az diákigazolvány számát egy paraméterként megadott értékre.

Majd egy tárolt függvényt hozok létre KerSZ mely visszaadja paraméterként adott fogyasztónak a szülő nevét.

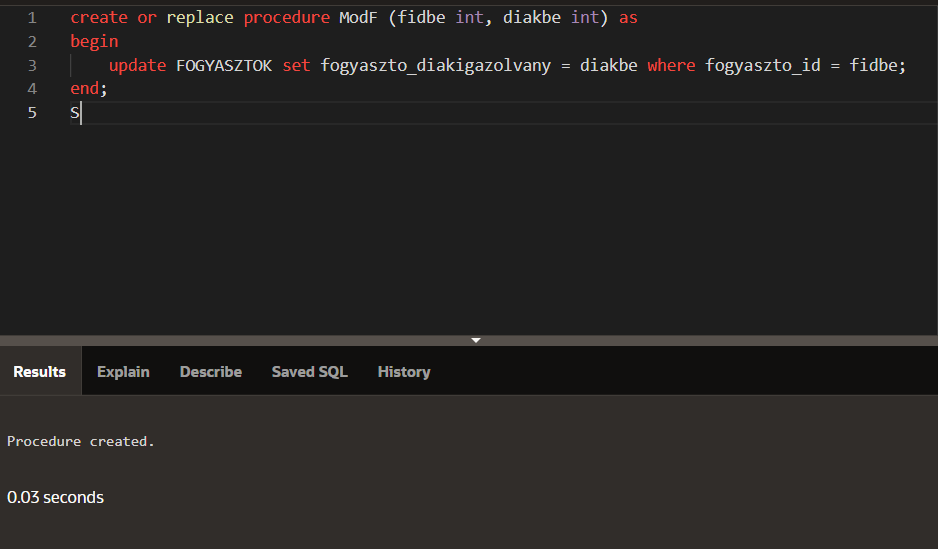
A két alprogramot elhelyezem egy csomagban, aminek a neve FOGYASZTOK\_cs.

create or replace procedure ModF (fidbe int, diakbe int) as

begin

update FOGYASZTOK set fogyaszto\_diakigazolvany = diakbe where fogyaszto\_id = fidbe;

end;



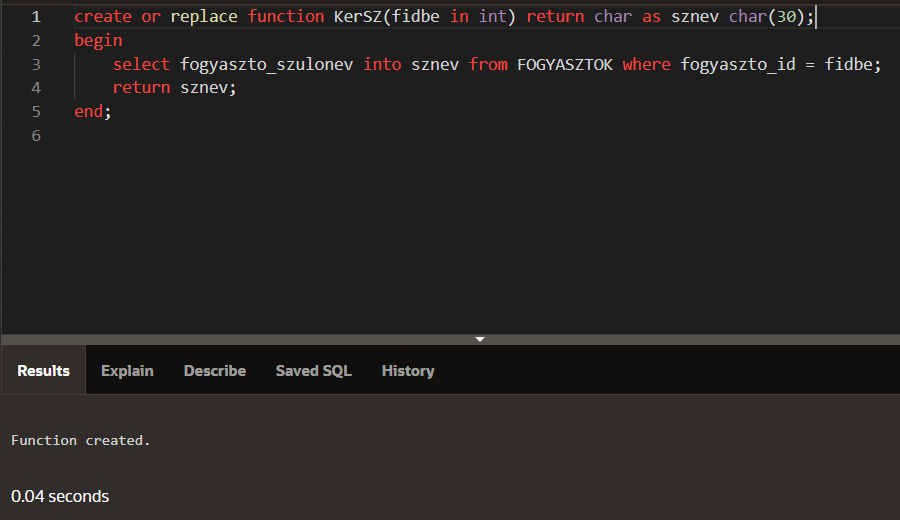
create or replace function KerSZ(fidbe in int) return char as sznev char(30);

begin

select fogyaszto\_szulonev into sznev from FOGYASZTOK where fogyaszto\_id = fidbe;

return sznev;

end;



create or replace package FOGYASZTOK\_cs as

procedure ModF(fidbe int, diakbe int);

function KerSZ(fidbe int) return char;

end FOGYASZTOK\_cs;

create or replace package body FOGYASZTOK\_cs as

procedure ModF(fidbe int, diakbe int) as

begin

update FOGYASZTOK set fogyaszto\_diakigazolvany = diakbe where fogyaszto\_id = fidbe;

end;

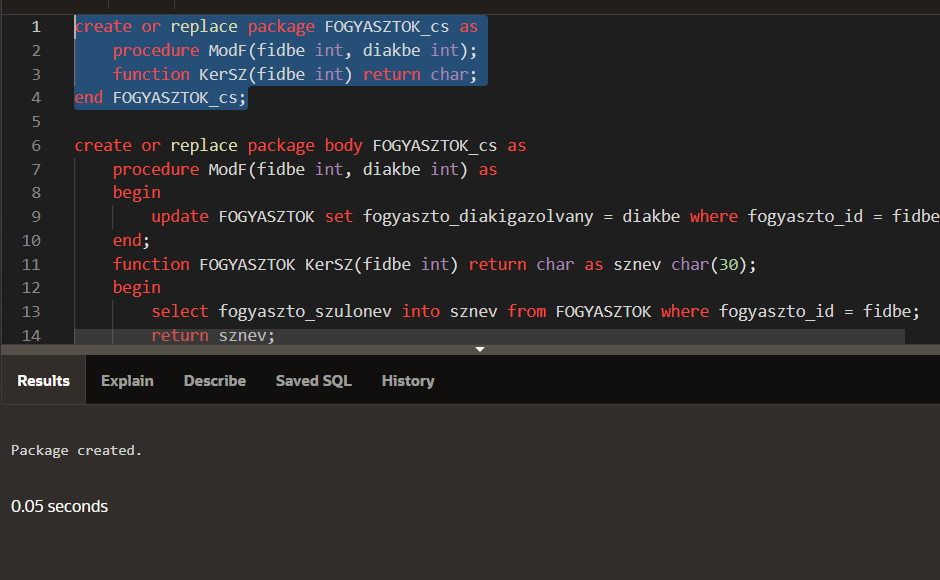
function KerSZ(fidbe int) return char as sznev char(30);

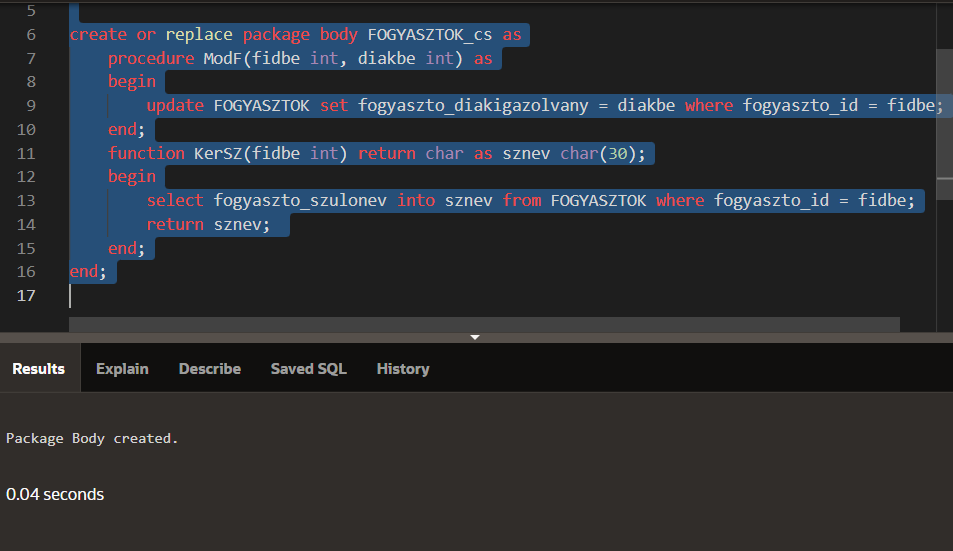
begin

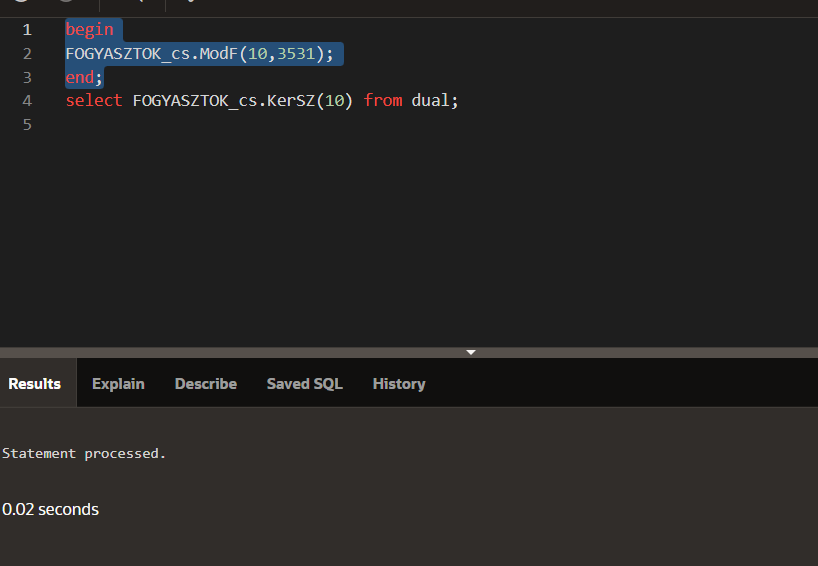
select fogyaszto\_szulonev into sznev from FOGYASZTOK where fogyaszto\_id = fidbe;

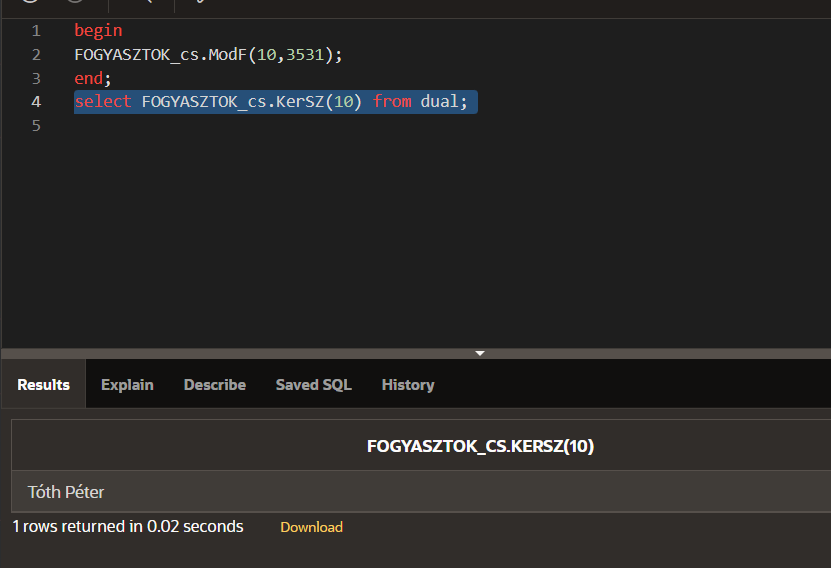
return sznev;

end;



****





**Cég táblán csomag:**

Készítek egy ModD tárolt eljárást, ami módosítja egy paraméterként megadott ID alapján a dátumot egy paraméterként megadott értékre.

Majd egy tárolt függvényt hozok létre KerN, mely visszaadja paraméterként adott cégnek nevét.

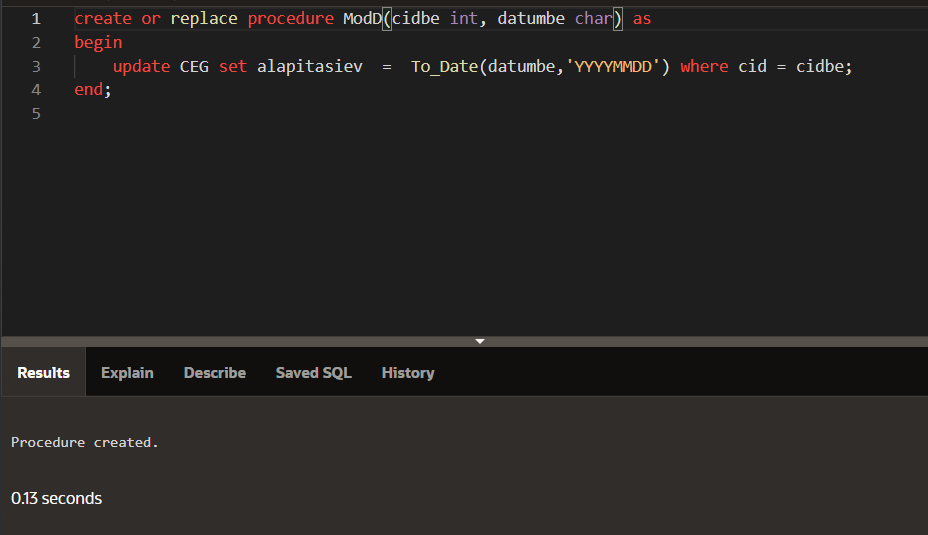
A két alprogramot elhelyezem egy csomagban, aminek a neve CEG\_cs.

create or replace procedure ModD(cidbe int, datumbe char) as

begin

update CEG set alapitasiev = To\_Date(datumbe,’YYYYMMDD’) where cid = cidbe;

end;



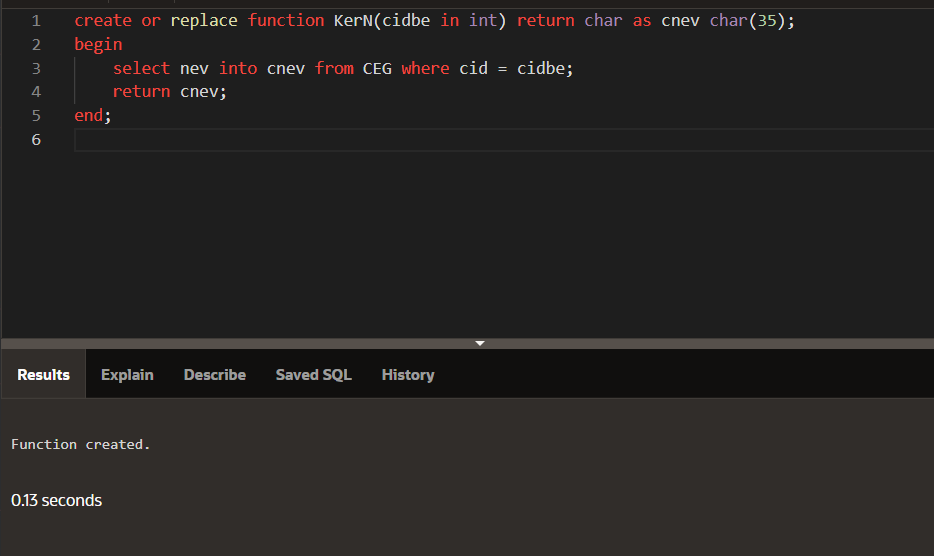
create or replace function KerN(cidbe in int) return char as cnev char(35);

begin

select nev into cnev from CEG where cid = cidbe;

return cnev;

end;

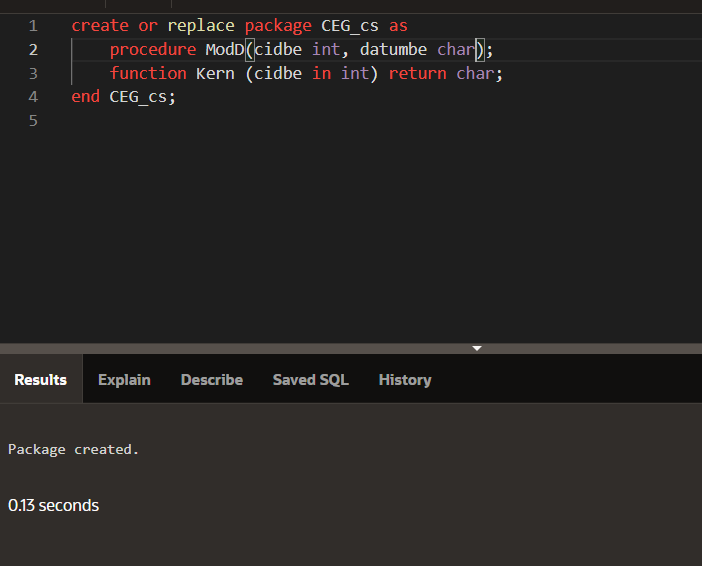


create or replace package CEG\_cs as

procedure ModD(cidbe int, datumbe char);

function Kern (cidbe in int) return char;

end CEG\_cs;



create or replace package body CEG\_cs as

procedure ModD(cidbe int, datumbe char) as

begin

update CEG set alapitasiev = TO\_DATE(datumbe,'YYYYMMDD') where cid = cidbe;

end;

function KerN(cidbe in int) return char as cnev char (35);

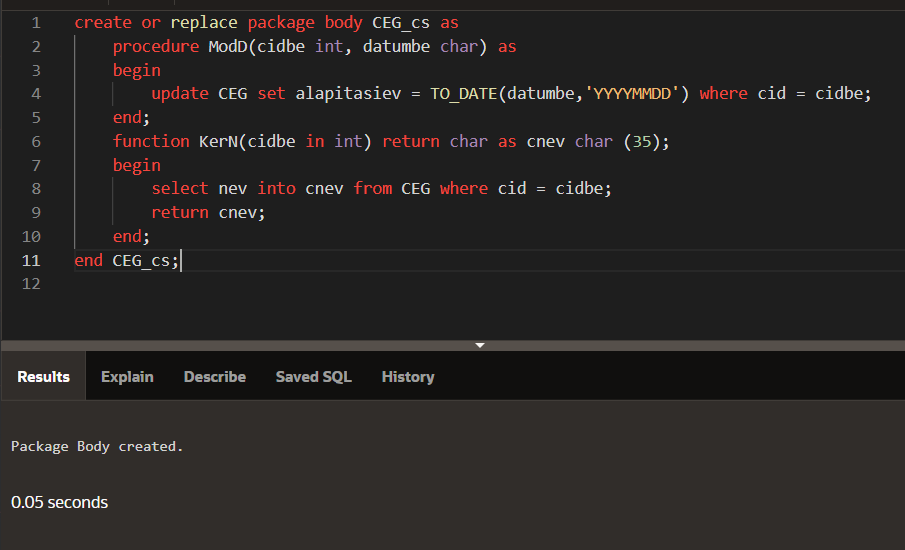
begin

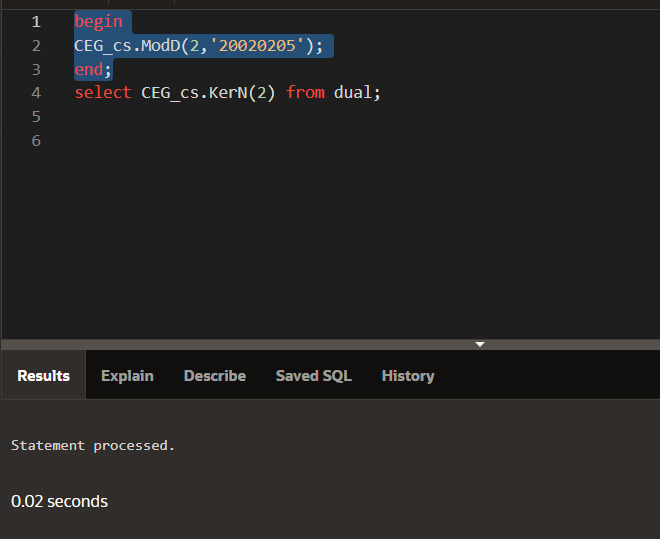
select nev into cnev from CEG where cid = cidbe;

return cnev;

end;

end CEG\_cs;



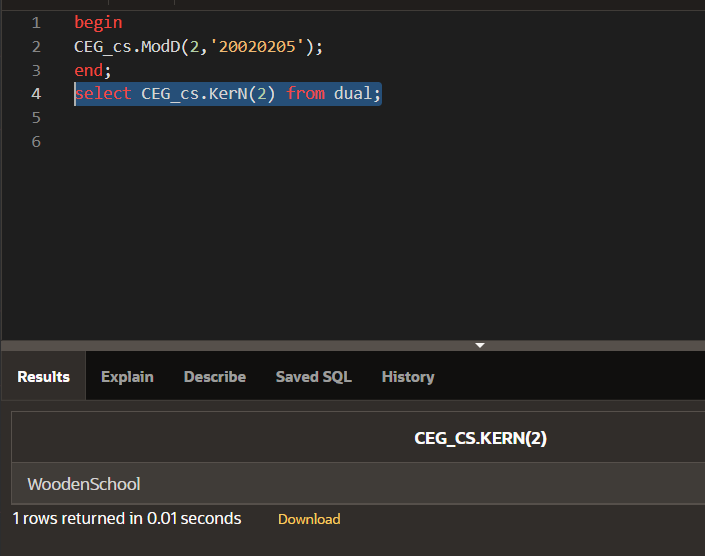


begin

CEG\_cs.ModD(2,'20020205');

end;

select CEG\_cs.KerN(2) from dual;



**• trigger készítése kulcs érték automatikus megadására,**

Trigger: adott esemény bekövetkezésekor automatikusan végrehajtásra kerülő tárolt műveletsor. Nem kell rá hivatkozni!

**SZEKHELY:**

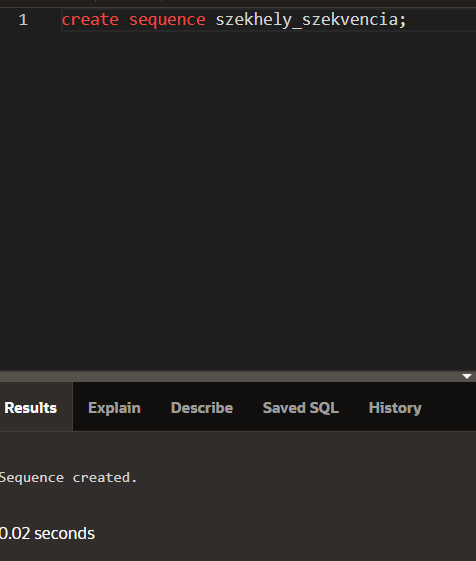
create sequence szekhely\_szekvencia;

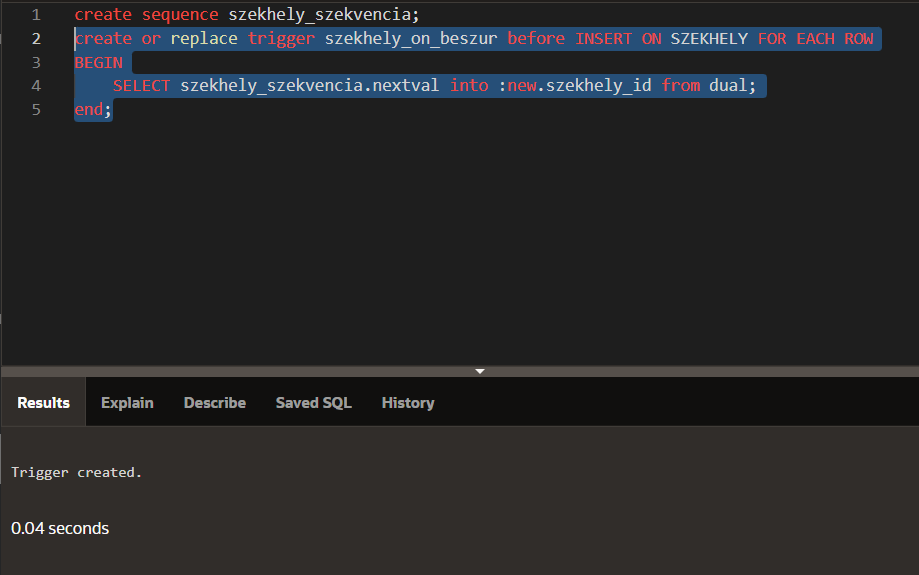
create or replace trigger szekhely\_on\_beszur before INSERT ON SZEKHELY FOR EACH ROW

BEGIN

SELECT szekhely\_szekvencia.nextval into :new.szekhely\_id from dual;

end;





**FOGYASZTOK:**

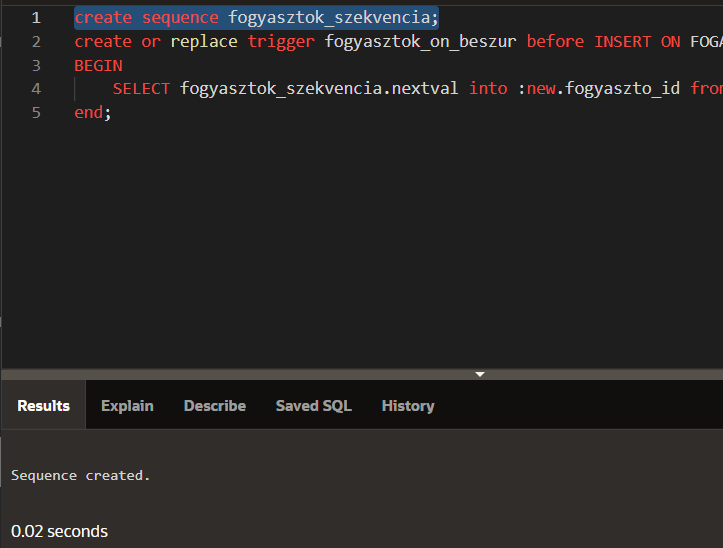
create sequence fogyasztok\_szekvencia;

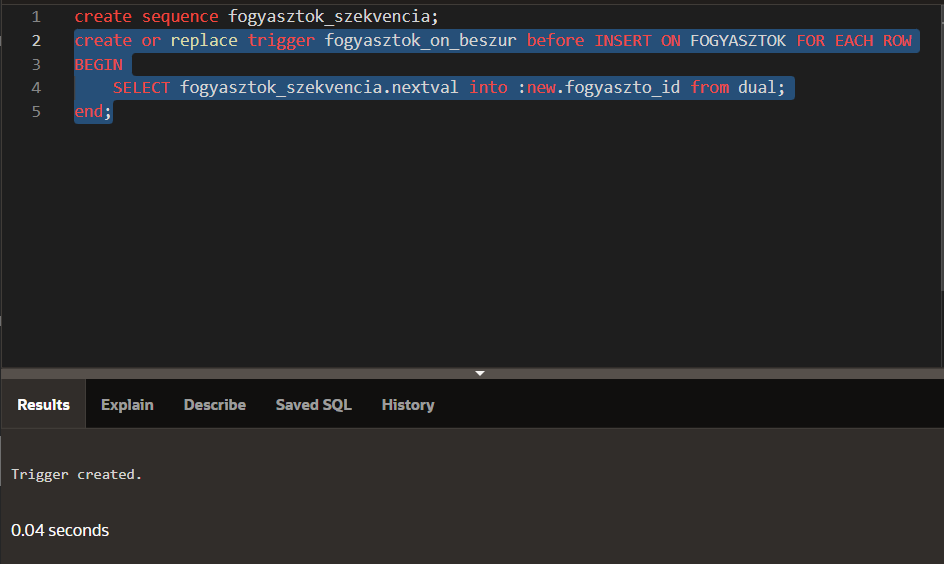
create or replace trigger fogyasztok\_on\_beszur before INSERT ON FOGYASZTOK FOR EACH ROW

BEGIN

SELECT fogyasztok\_szekvencia.nextval into :new.fogyaszto\_id from dual;

end;





**CEG:**

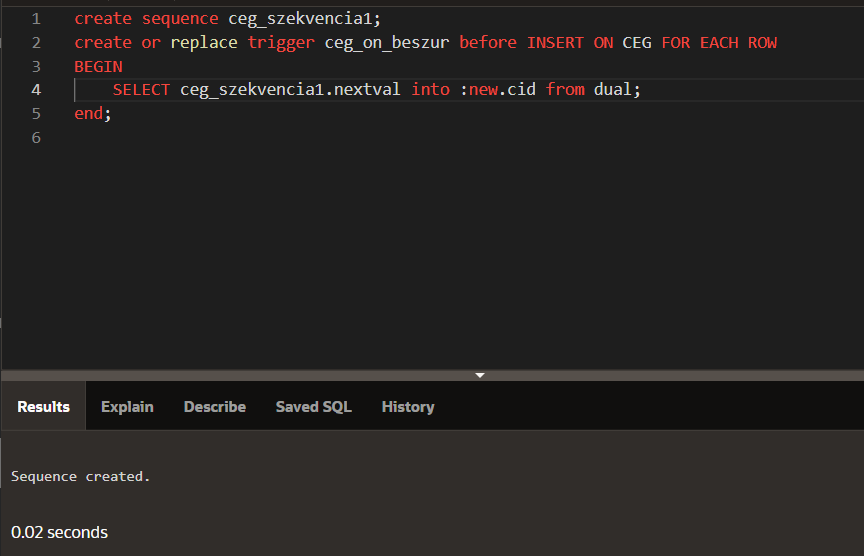
create sequence ceg\_szekvencia1;

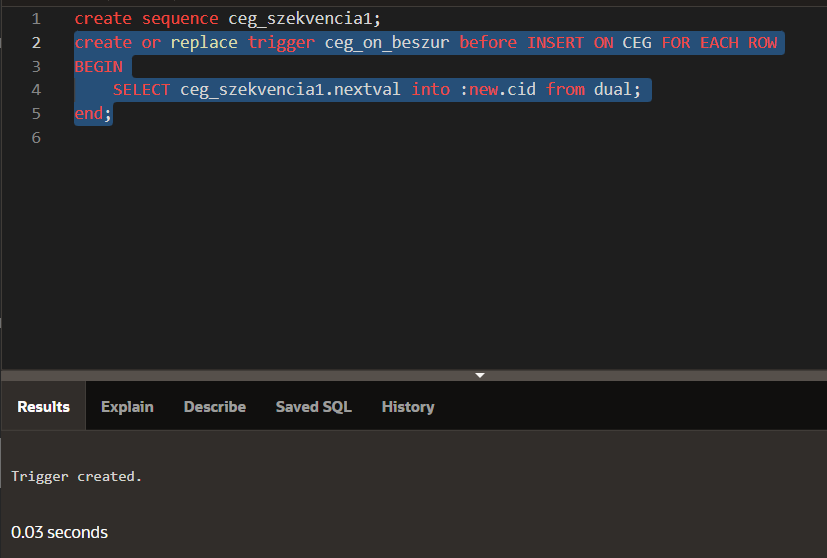
create or replace trigger ceg\_on\_beszur before INSERT ON CEG FOR EACH ROW

BEGIN

SELECT ceg\_szekvencia1.nextval into :new.cid from dual;

end;





**• trigger készítése a módosítások kontrollálására**.

**SZEKHELY:**

Ellenőrizzük, hogy megfelelő e a irányítószámnak a nagysága, ha nem akkor nem módosítja.

create or replace trigger irszell before UPDATE of szekhely\_irsz on SZEKHELY FOR EACH ROW

declare

maxirsz number;

minirsz number;

BEGIN

maxirsz :=10000;

minirsz := 999;

if (:new.szekhely\_irsz not between minirsz and maxirsz ) then

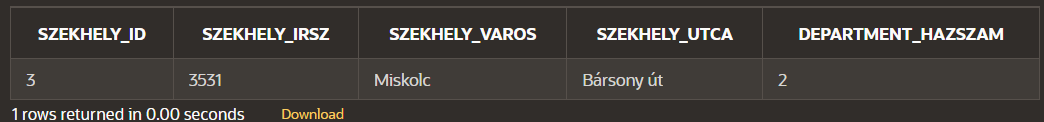
:new.szekhely\_irsz := :old.szekhely\_irsz;

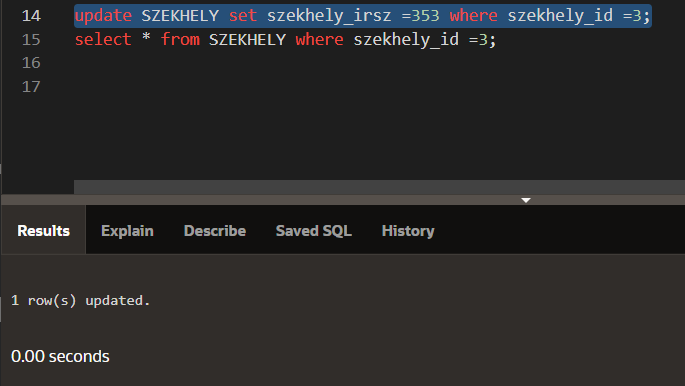
end if;

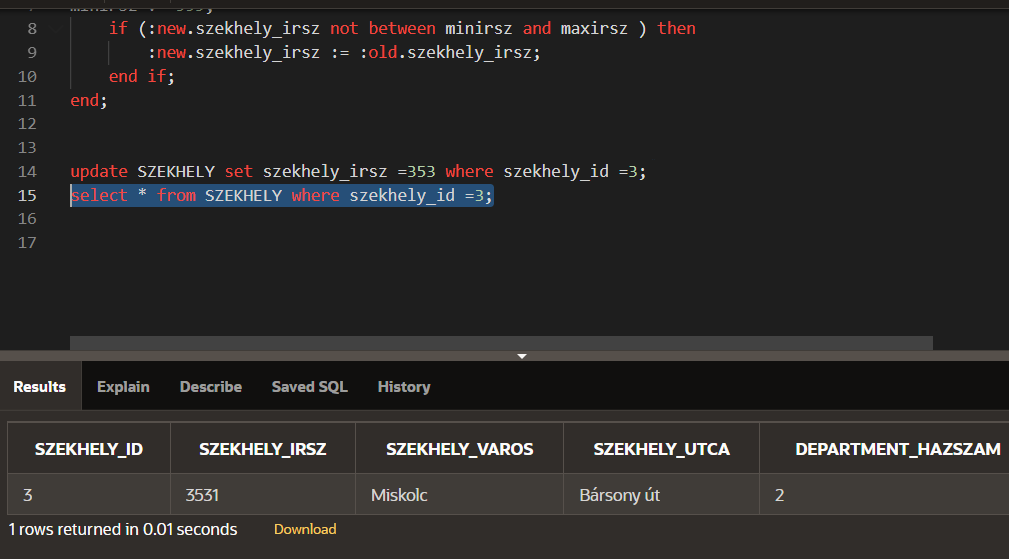
end;

update SZEKHELY set szekhely\_irsz =3531 where szekhely\_id =3;

select \* from SZEKHELY where szekhely\_id =3;







nem változtatta meg az irányítószámot helytelen adatra.

**FOGYASZTOK:**

ellenőrizzük, hogy megfelelő e a diákigazolvány nagysága, ha nem akkor nem módosítja.

create or replace trigger diakigell before UPDATE ON FOGYASZTOK FOR EACH ROW

declare

maxdiak number;

mindiak number;

BEGIN

maxdiak :=10000000000;

mindiak := 999999999;

if (:new.fogyaszto\_diakigazolvany not between mindiak and maxdiak ) then

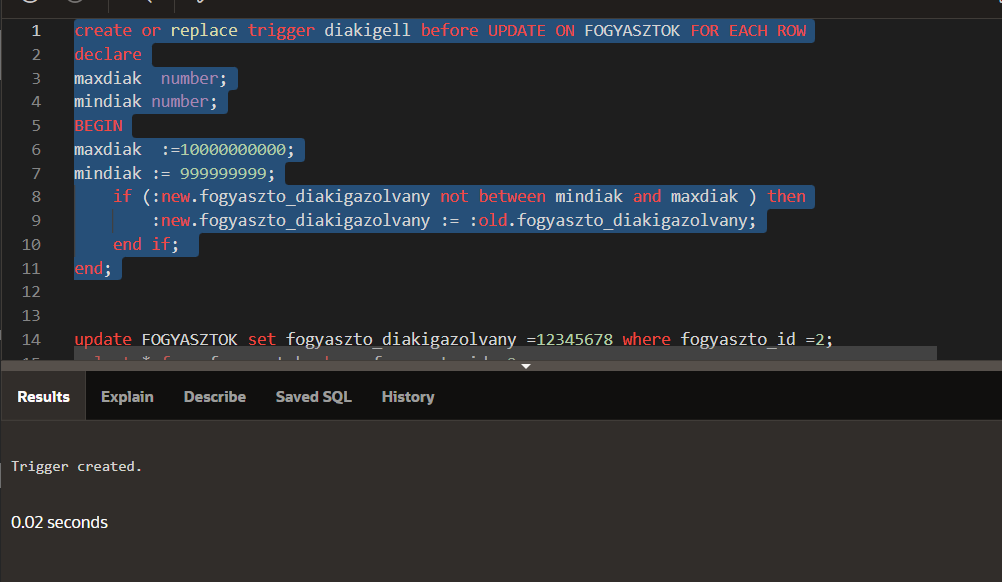
:new.fogyaszto\_diakigazolvany := :old.fogyaszto\_diakigazolvany;

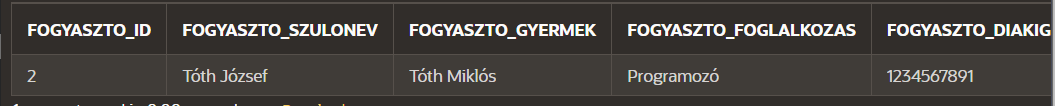
end if;

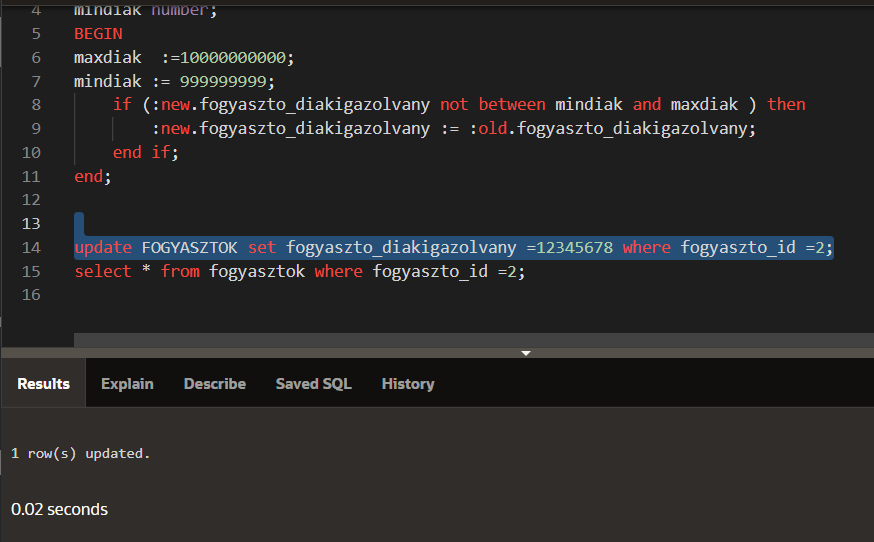
end;

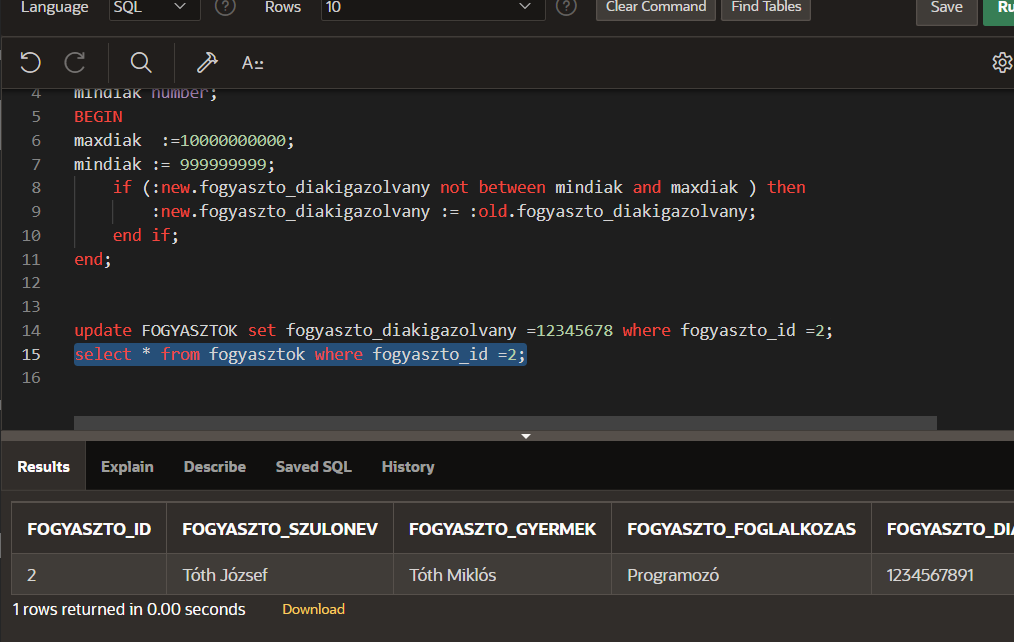
update FOGYASZTOK set fogyaszto\_diakigazolvany =12345678 where fogyaszto\_id =2;

select \* from fogyasztok where fogyaszto\_id =2;









nem változtatta meg a diákigazolványszámát helytelen adatra.

**CEG:**

Cél: A megváltoztatni kívánt dátum, ha nagyobb, mint a jelenlegi sysdate akkor sajnos nem jön létre a cég.

**helyes megoldás:**

create or replace trigger CheckEvCeg before update of alapitasiev on CEG for each row

begin

if :new.alapitasiev > sysdate then

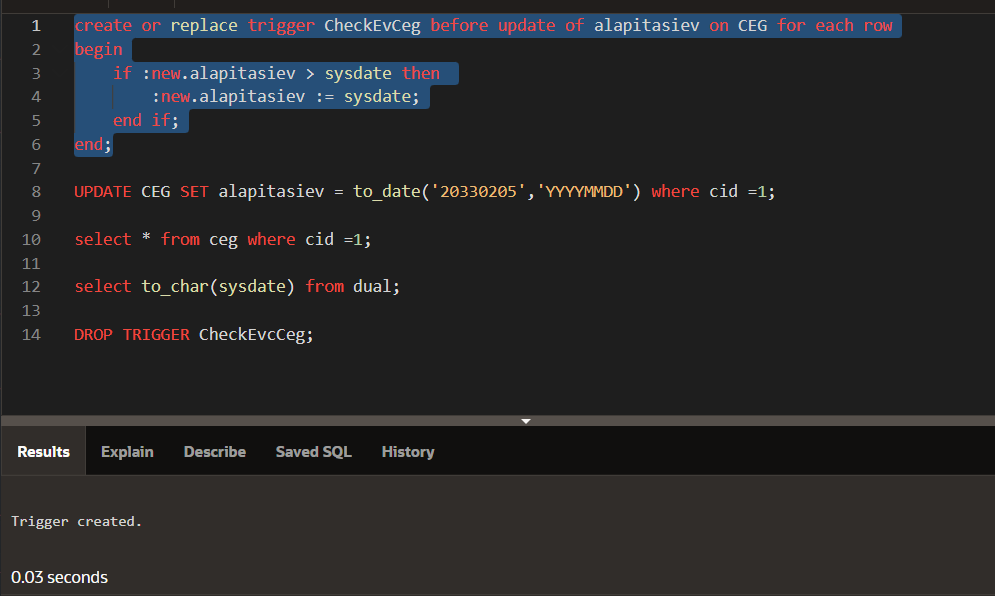
:new.alapitasiev := sysdate;

end if;

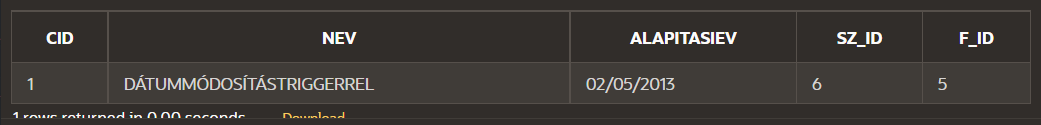
end;

UPDATE CEG SET alapitasiev = to\_date('20330205','YYYYMMDD') where cid =1;

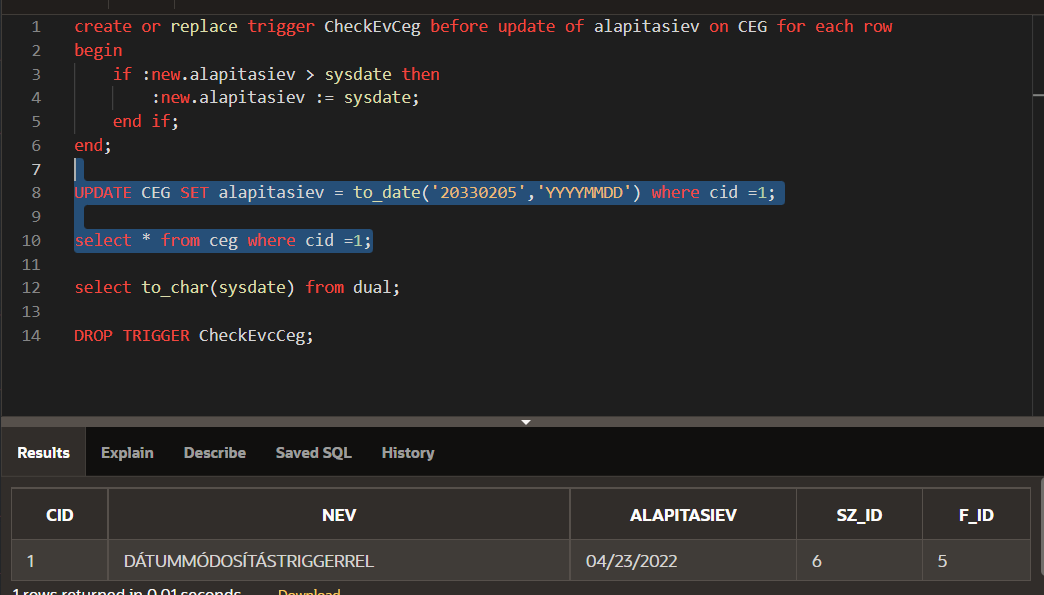
select \* from ceg where cid =1;



eredeti dátum:



majd a rossz dátum használata és kiiratás:



**• trigger készítése módosítási események naplózására (többféle esemény naplózása egyetlen triggerrel plusz pontot ér)**

napló létrehozása:

create table Naplo (muvelet char(50), datum date, felhasznalo char(20));

**szekhely:**

create or replace trigger irszellnaplo before UPDATE of szekhely\_irsz on SZEKHELY FOR EACH ROW

declare

maxirsz number;

minirsz number;

msg char(50) := concat(:old.szekhely\_irsz, ' Módosítva lett a Székhely irányítószáma.');

BEGIN

maxirsz :=10000;

minirsz := 999;

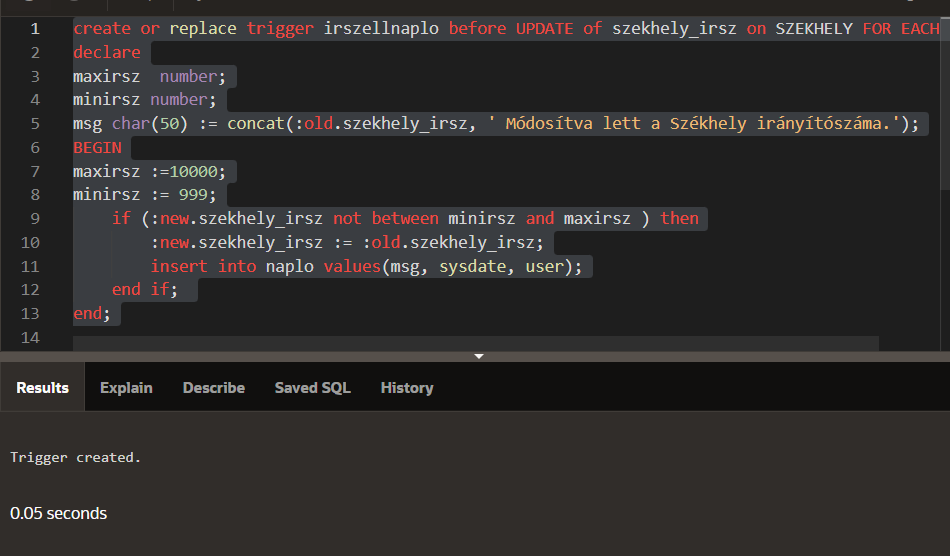
if (:new.szekhely\_irsz not between minirsz and maxirsz ) then

:new.szekhely\_irsz := :old.szekhely\_irsz;

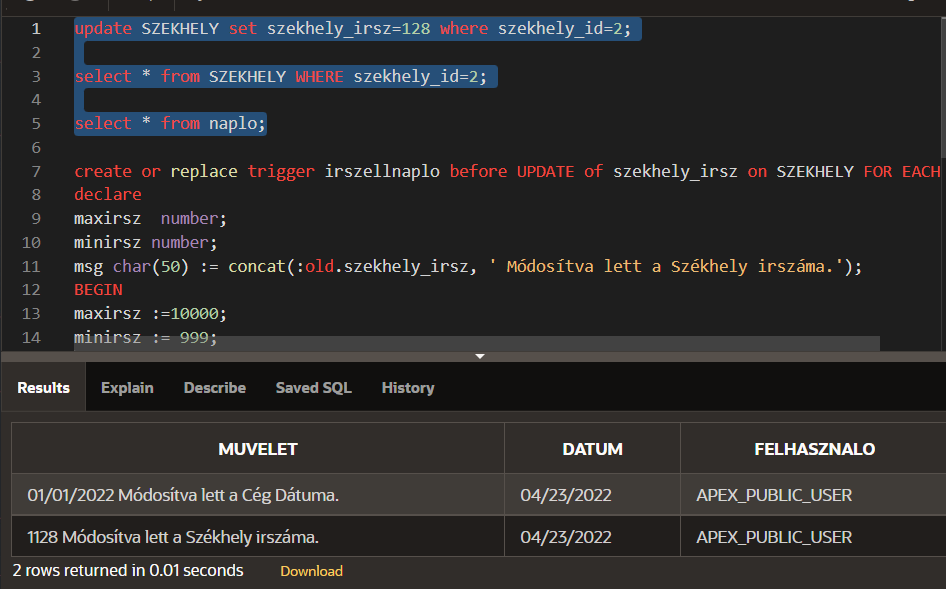
insert into naplo values(msg, sysdate, user);

end if;

end;



naplózás és érték felvétel megpróbálása alacsony szinten:



**fogyasztok:**

create or replace trigger diakigellnaplo before UPDATE ON FOGYASZTOK FOR EACH ROW

declare

maxdiak number;

mindiak number;

msg char(50) := concat(:old.fogyaszto\_diakigazolvany, ' Módosítva lett a fogyasztó diákja.');

BEGIN

maxdiak :=10000000000;

mindiak := 999999999;

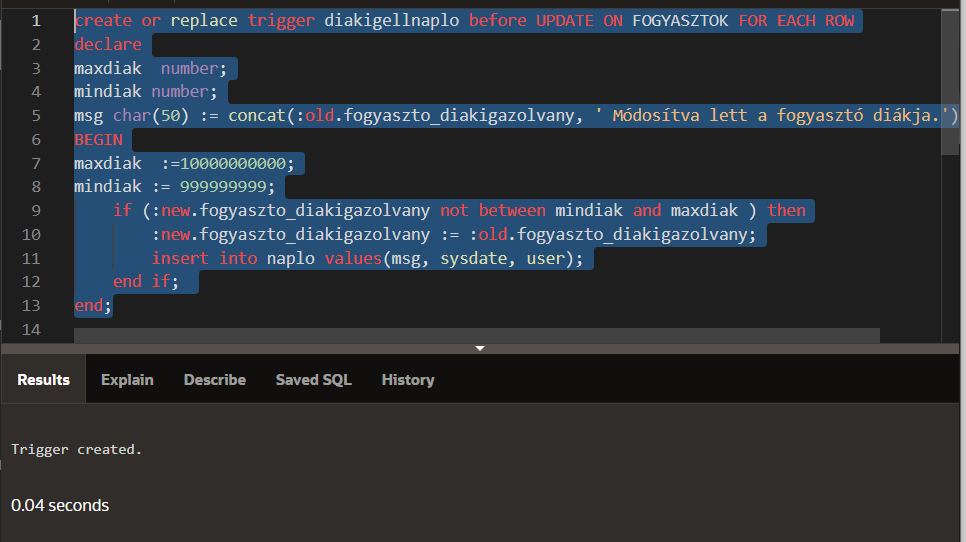
if (:new.fogyaszto\_diakigazolvany not between mindiak and maxdiak ) then

:new.fogyaszto\_diakigazolvany := :old.fogyaszto\_diakigazolvany;

insert into naplo values(msg, sysdate, user);

end if;

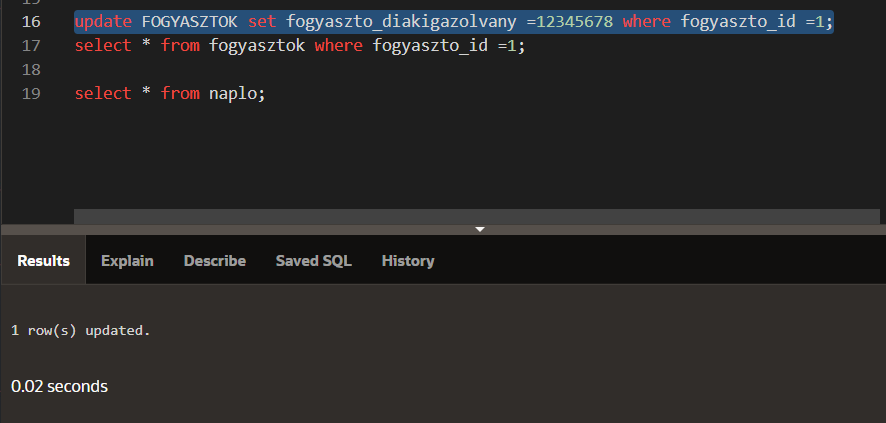
end;



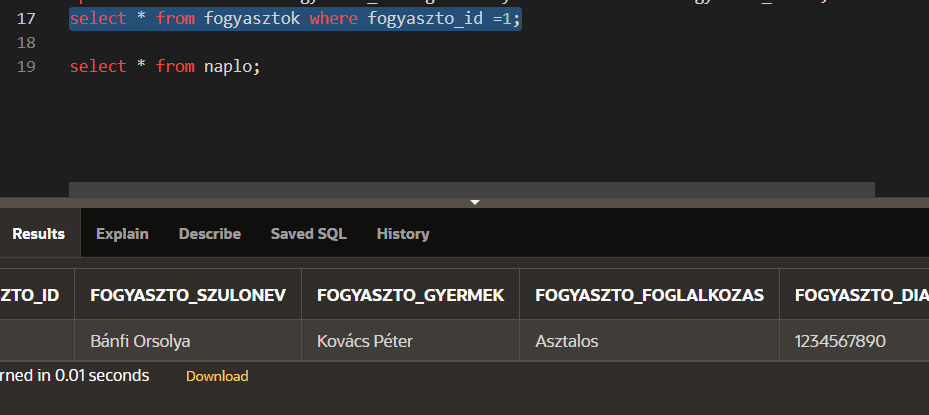
update FOGYASZTOK set fogyaszto\_diakigazolvany =12345678 where fogyaszto\_id =2;

select \* from fogyasztok where fogyaszto\_id =2;

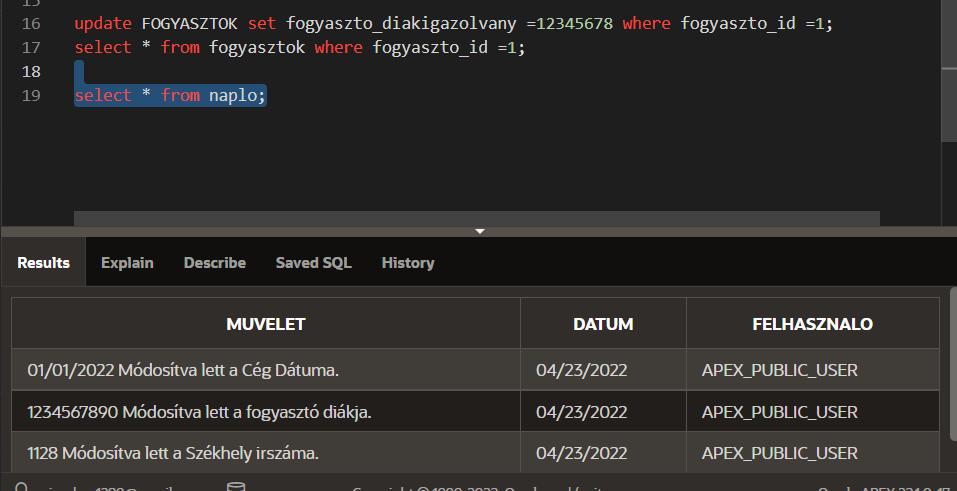
frissítés:



frissítés megnézése, hogy teljes mértékben nem történt meg:



napló megtekintése:



**ceg:**

Cél: A megváltoztatni kívánt dátum, ha nagyobb, mint a jelenlegi sísdate akkor sajnos nem jön létre a cég és ennek naplózása.

create or replace trigger CheckEvCegNaplo before update of alapitasiev on CEG for each row

declare

msg char(50) := concat(:old.alapitasiev, ' Módosítva lett a Cég Dátuma.');

begin

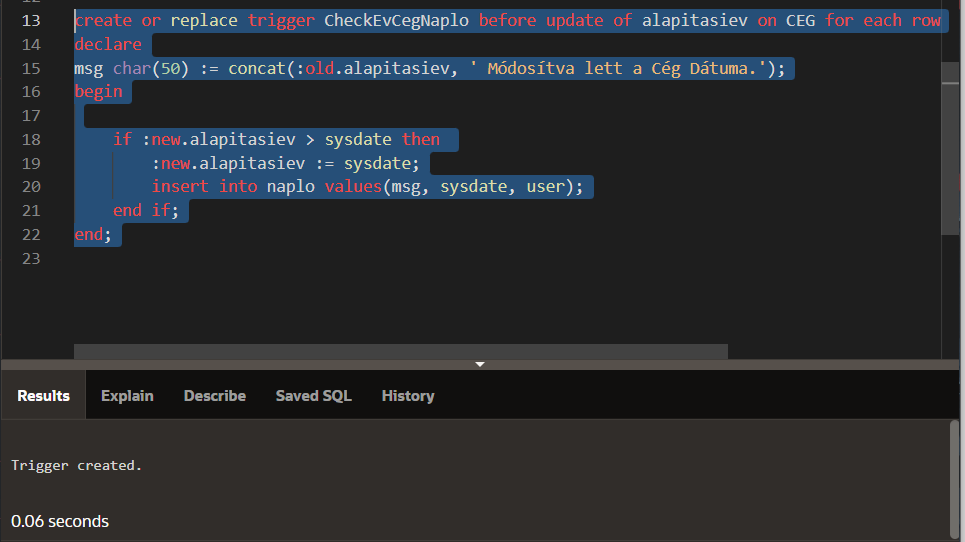
if :new.alapitasiev > sysdate then

:new.alapitasiev := sysdate;

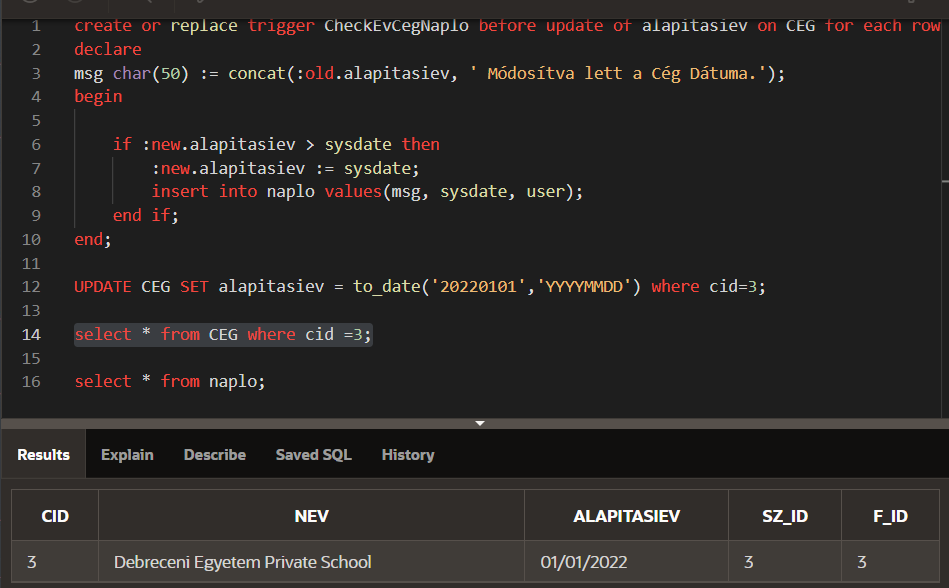
insert into naplo values(msg, sysdate, user);

end if;

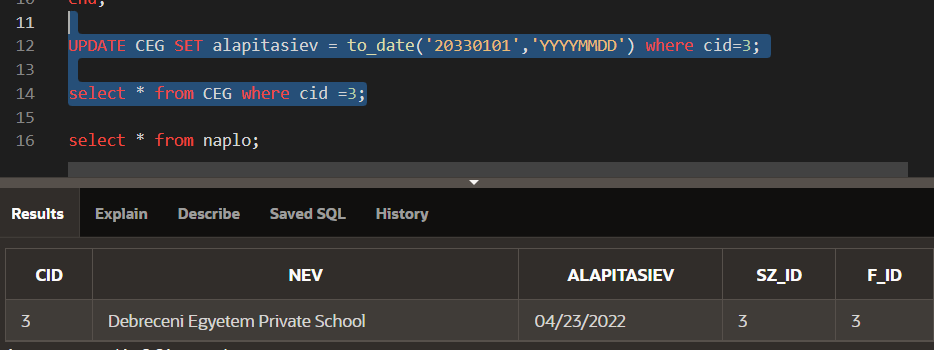
end;

****

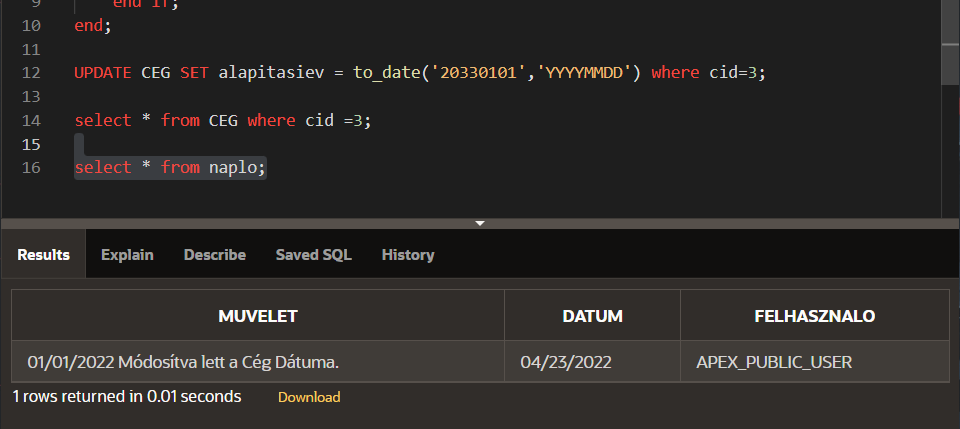
eredeti:



naplózáshoz megváltoztatjuk a dátumot:



naplóba került adat:



A PL/SQL-ben megvalósítandó elemek:

• normál parancsok használata,

• implicit kurzor,

• explicit kurzor,

• kivételkezelés (exception használata).