



DEBUGOLÁS PL/SQL-BEN

BEMUTATÓT KÉSZÍTETTE: TASNÁDI ISTVÁN-NORBERT

JÁNOSI RANCZ KATALIN TÜNDE VEZETÉSÉVEL

MELY PROGRAMOT FOGJUK VIZSGÁLNI?

- Számos programban van lehetőségünk debugolni PL/SQL programjainkat, mint például Visual Studio, IntelliJ Idea, DataGrip stb.
- Ebben a bemutatóban viszont az Oracle SQL Developer fogjuk megvizsgálni

ELSŐ LÉPÉSEK

- Még mielőtt debugolni tudnánk, meg kell győződjünk először, hogy eleget teszünk a következő feltételeknek:
 - 1) van-e debug jogunk
 - 2) rendesen be van-e konfigurálva az ACL, Network tulajdonságok, port stb.
 - 3) SQL Developer beállításainál Debugger résznél kivan-e pipálva a “Use DBMS_DEBUG” opció

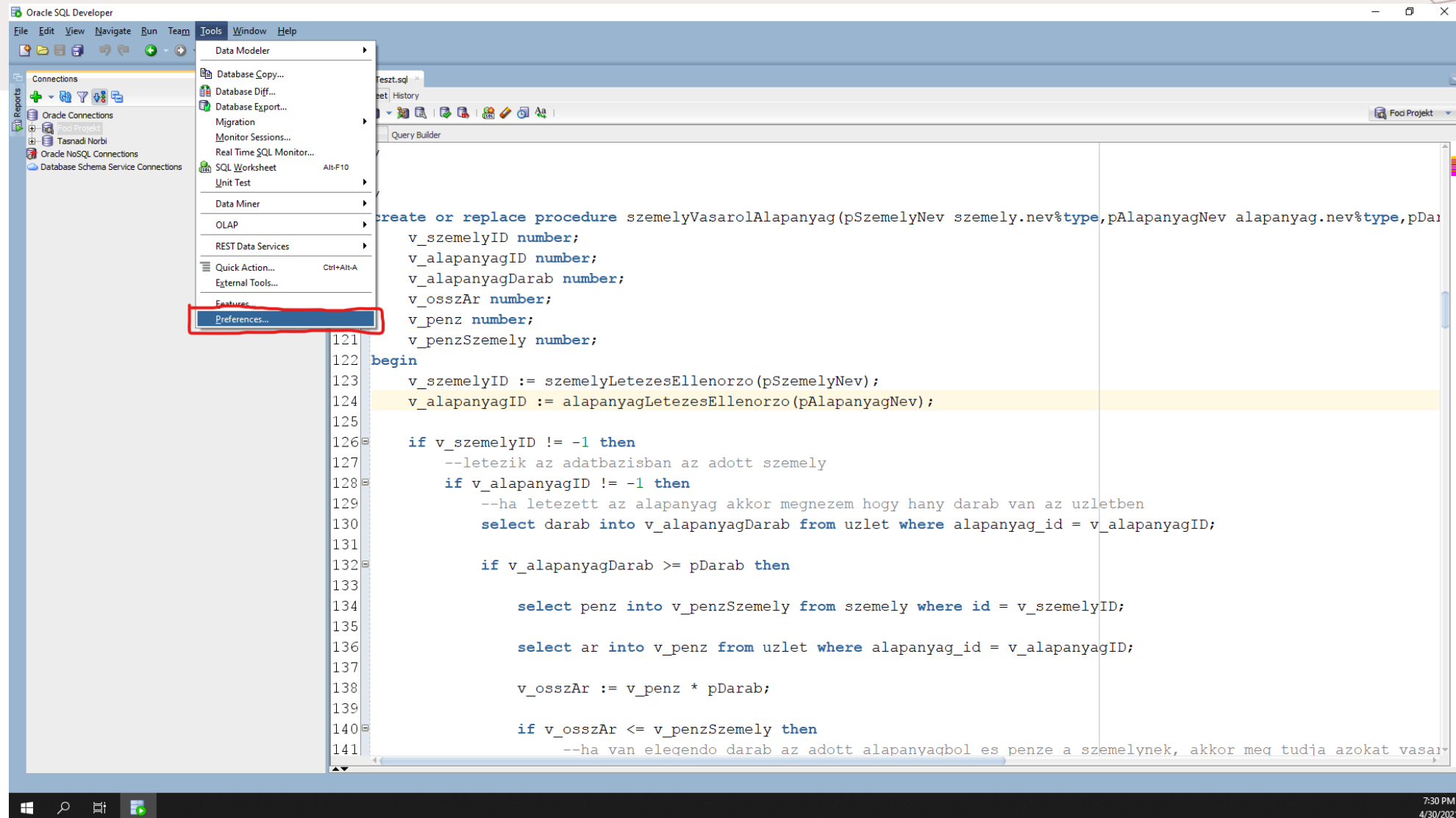
DEBUG JOG MEGADÁS

- Ahhoz, hogy debugolhassuk a PL/SQL programjainkat szükségünk van debug jogokra, nevezetesen a következőkre: ***DEBUG CONNECT SESSION*** és ***DEBUG ANY PROCEDURE***
- Ezt a két jogot az adminisztrátor kell megadja a felhasználóknak
- Az adminisztrátor a következőképpen adhatja meg a szükséges jogokat a usereknek:
 - `grant DEBUG CONNECT SESSION to <user_name> ;`
 - `grant DEBUG ANY PROCEDURE to <user_name> ;`

DEBUGOLÁS MÁSODIK KRITÉRIUM OPCIÓK

- A másik feltétele annak, hogy debugolhassunk az az, hogy rendesen be legyen állítva az ACL, Network tulajdonságok
- Meglehetősen macerás ezeket mind beállítani, sok munkát vesz igénybe
- Ezért be lesz mutatva egy másik módszer, nevezetesen , hogy hogyan lehet beállítani az Oracle SQL Developerben a “**Use DBMS_DEBUG**” opció, ami szignifikánsan megkönnyíti majd dolgunkat

USE DBMS_DEBUG BEÁLLÍTÁSA



The screenshot shows the Oracle SQL Developer interface. The 'Tools' menu is open, and 'Preferences...' is highlighted with a red rectangle. The main editor window contains the following PL/SQL code:

```
create or replace procedure személyVasarolAlapanyag(pSzemelyNev személy.nev%type,pAlapanyagNev alapanyag.nev%type,pDarab
v_szemelyID number;
v_alapanyagID number;
v_alapanyagDarab number;
v_osszAr number;
v_penz number;
v_penzSzemely number;
begin
v_szemelyID := személyLetezesEllenorzo(pSzemelyNev);
v_alapanyagID := alapanyagLetezesEllenorzo(pAlapanyagNev);

if v_szemelyID != -1 then
--letezik az adatbazisban az adott személy
if v_alapanyagID != -1 then
--ha letezett az alapanyag akkor megnezem hogy hany darab van az uzletben
select darab into v_alapanyagDarab from uzlet where alapanyag_id = v_alapanyagID;

if v_alapanyagDarab >= pDarab then

select penz into v_penzSzemely from személy where id = v_szemelyID;

select ar into v_penz from uzlet where alapanyag_id = v_alapanyagID;

v_osszAr := v_penz * pDarab;

if v_osszAr <= v_penzSzemely then
--ha van elegendo darab az adott alapanyagbol es penze a személynek, akkor meg tudja azokat vasar
```

USE DBMS_DEBUG BEÁLLÍTÁSA

The screenshot displays the Oracle SQL Developer interface. A 'Preferences' dialog box is open, with the 'Debugger' category selected in the left-hand tree. The 'Debugger' settings on the right include:

- Show Tool Tip in Code Editor While Debugging
- Show Action Buttons in Log Window While Debugging
- Connection Retry Setting: 90
- Database Debug Protocol:
 - Use DBMS_DEBUG_IDWP
 - Prompt for Debugger Host for Database Debugging
 - Attempt to Break Method Evaluation Deadlocks
 - Debugging Port Range (Minimum: -4000, Maximum: 4999)
 - Use DBMS_DEBUG
- Start Debugging Option:
 - Run Until a Breakpoint Occurs
 - Step Over
 - Step Into
- Enable Change Tracking
 - Changed item Background Color
 - Changed item Foreground Color

The background SQL script in the Worksheet is as follows:

```
112 /
113
114 /
115 create or re
116 v_szeme
117 v_alapar
118 v_alapar
119 v_osszAr
120 v_penz r
121 v_penzSz
122 begin
123 v_szemel
124 v_alapar
125
126 if v_sze
127 --le
128 if v
129
130
131
132
133
134 select penz into v_penzszemely from szemely where id = v_szemelyID;
135
136 select ar into v_penz from uzlet where alapanyag_id = v_alapanyagID;
137
138 v_osszAr := v_penz * pDarab;
139
140 if v_osszAr <= v_penzSzemely then
141 --ha van elegendo darab az adott alapanyagbol es penze a szemelynek, akkor meg tudja azokat vasar
```

USE DBMS_DEBUG BEÁLLÍTÁSA

The screenshot displays the Oracle SQL Developer interface with a SQL worksheet open. A 'Preferences' dialog box is overlaid, showing the 'Debugger' settings. The 'Use DBMS_DEBUG' option is selected and highlighted with a red circle. The background SQL code is partially visible, showing a PL/SQL procedure with various variables and conditional logic.

```
112 /
113
114 /
115 create or re
116 v_szemel
117 v_alapar
118 v_alapar
119 v_osszAr
120 v_penz
121 v_penzSz
122 begin
123 v_szemel
124 v_alapar
125
126 if v_sze
127 --le
128 if v
129
130
131
132
133
134 select penz into v_penzszemely from szemely where id = v_szemelyID;
135
136 select ar into v_penz from uzlet where alapanyag_id = v_alapanyagID;
137
138 v_osszAr := v_penz * pDarab;
139
140 if v_osszAr <= v_penzSzemely then
141 --ha van elegendo darab az adott alapanyagbol es penze a személynek, akkor meg tudja azokat vasar
```

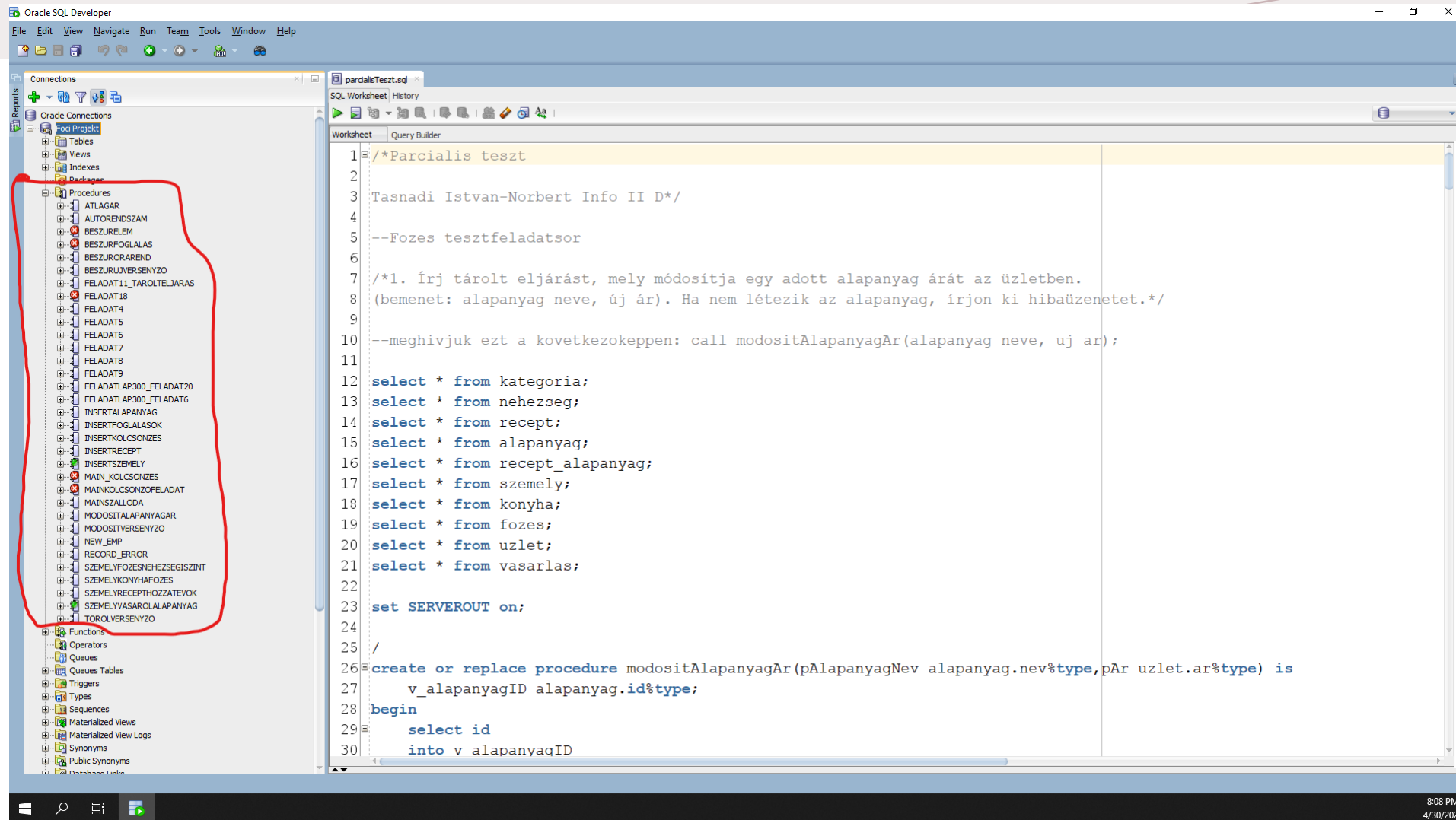
Debugger Preferences:

- Show Tool Tip in Code Editor While Debugging
- Show Action Buttons in Log Window While Debugging
- Connection Retry Setting: 90
- Database Debug Protocol:
 - Use DBMS_DEBUG_IDWP
 - Use DBMS_DEBUG
- Start Debugging Option:
 - Run Until a Breakpoint Occurs
 - Step Over
 - Step Into
- Enable Change Tracking
 - Changed Item Background Color
 - Changed Item Foreground Color

MIKET DEBUGOLHATUNK AZ ORACLE SQL DEVELOPERBEN?

- Az Oracle SQL Developerben debugolhatunk:
 - Procedurákat (procedure)
 - Függvényeket (function)
 - Triggereket (trigger)

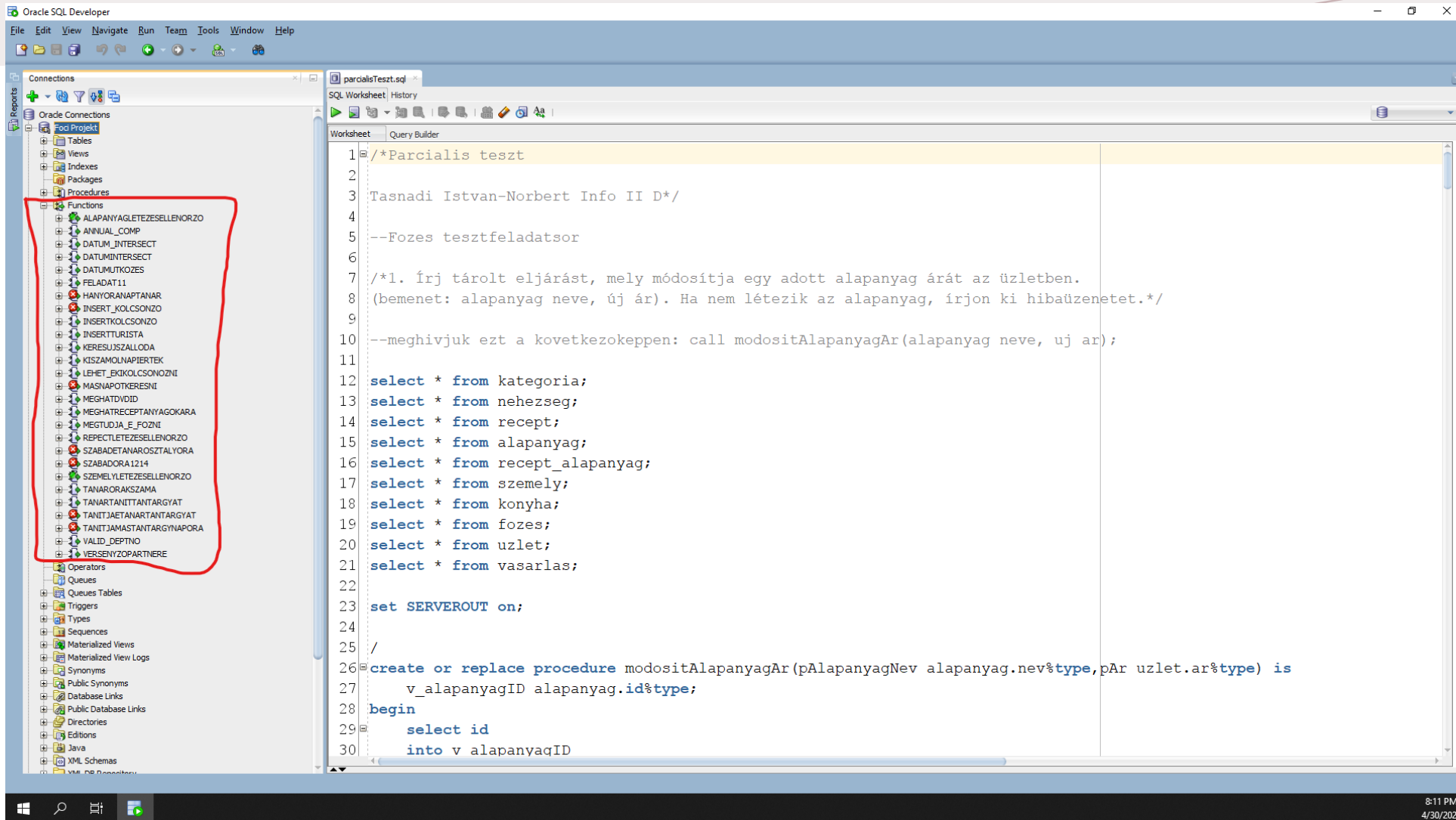
HOL KAPHATOM MEG AZ ÁLTALAM MEGIRT PROCEDÚRÁKAT?



The screenshot displays the Oracle SQL Developer interface. On the left, the 'Connections' pane shows a tree view of the database schema. The 'Procedures' folder is highlighted with a red circle, indicating the location of stored procedures. The main window shows a SQL worksheet with the following code:

```
1 /*Parcialis teszt
2
3 Tasnadi Istvan-Norbert Info II D*/
4
5 --Fozes tesztfeladatsor
6
7 /*1. Írj tárolt eljárást, mely módosítja egy adott alapanyag árát az üzletben.
8 (bemenet: alapanyag neve, új ár). Ha nem létezik az alapanyag, írjon ki hibaüzenetet.*/
9
10 --meghívjuk ezt a következőképpen: call modositAlapanyagAr(alapanyag neve, uj ar);
11
12 select * from kategoria;
13 select * from nehezseg;
14 select * from recept;
15 select * from alapanyag;
16 select * from recept_alapanyag;
17 select * from szemely;
18 select * from konyha;
19 select * from fozes;
20 select * from uzlet;
21 select * from vasarlas;
22
23 set SERVEROUT on;
24
25 /
26 create or replace procedure modositAlapanyagAr(pAlapanyagNev alapanyag.nev%type,pAr uzlet.ar%type) is
27     v_alapanyagID alapanyag.id%type;
28 begin
29     select id
30     into v_alapanyagID
```

HOL KAPHATOM MEG AZ ÁLTALAM MEGIRT FÜGGVÉNYEKET?

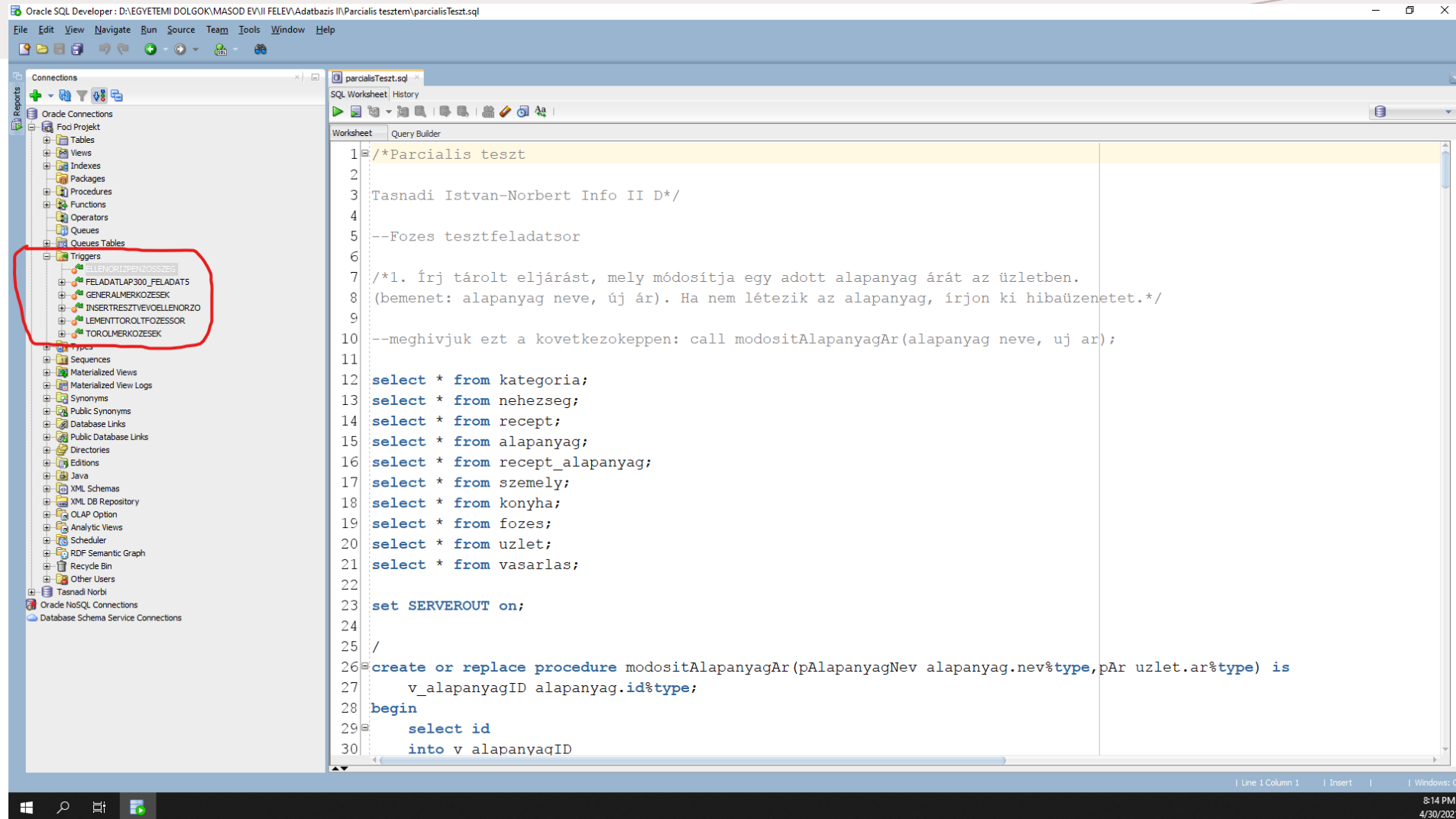


The screenshot shows the Oracle SQL Developer interface. On the left, the 'Connections' pane displays a tree view of database objects. The 'Functions' folder is expanded, and a red rectangle highlights a list of functions including: ALAPANYAGLETEZESELENORZO, ANNJAL_COMP, DATUM_INTERSECT, DATUMINTERSECT, DATUMUTKOZES, FELADAT11, HANYORANAPTANAR, INSERT_KOLCSONZO, INSERTKOLCSONZO, INSERTTURISTA, KERESUJSZALLODA, KISZAMOLNAPIERTEK, LEHET_EKIKOLCSONOZNI, MASNAPOTKERESNI, MEGHATVOID, MEGHATRECEPTANYAGOKARA, MEGTUDJA_E_FOZNI, REPECTLETEZESELENORZO, SZABADETANAROSZTALYORA, SZABADORA1214, SZEMELYLETEZESELENORZO, TANARORAKSZAMA, TANARTANITTANTARGYAT, TANITJAEKTANARTANTARGYAT, TANITJAMASTANTARGYNAPORA, VALID_DEPTNO, and VERSENYZOPARTNERE.

The main window shows a SQL script in a worksheet titled 'parcialsTeszt.sql'. The script contains the following SQL code:

```
1 /*Parcialis teszt
2
3 Tasnadi Istvan-Norbert Info II D*/
4
5 --Fozes tesztfeladatsor
6
7 /*1. Írj tárolt eljárást, mely módosítja egy adott alapanyag árát az üzletben.
8 (bemenet: alapanyag neve, új ár). Ha nem létezik az alapanyag, írjon ki hibaüzenetet.*/
9
10 --meghívjuk ezt a következőképpen: call modositAlapanyagAr(alapanyag neve, uj ar);
11
12 select * from kategoria;
13 select * from nehezseg;
14 select * from recept;
15 select * from alapanyag;
16 select * from recept_alapanyag;
17 select * from személy;
18 select * from konyha;
19 select * from fozes;
20 select * from uzlet;
21 select * from vasarlas;
22
23 set SERVEROUT on;
24
25 /
26 create or replace procedure modositAlapanyagAr(pAlapanyagNev alapanyag.nev%type,pAr uzlet.ar%type) is
27     v_alapanyagID alapanyag.id%type;
28 begin
29     select id
30     into v_alapanyagID
```

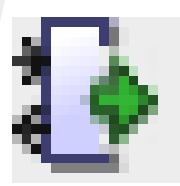
HOL KAPHATOM MEG AZ ÁLTALAM MEGIRT TRIGGEREKET?



The screenshot shows the Oracle SQL Developer interface. On the left, the 'Reports' tree is expanded to 'Triggers', where several triggers are listed and circled in red: ELLENORIZIPENZOSSZEG, FELADATLAP300_FELADATS, GENERALMERKOZESEK, INSERTRESZTVEIOELLENORZO, LEMENTTOROLTOFOZEISSOR, and TOROLMERKOZESEK. The main editor displays a SQL script for a procedure named 'modositAlapanyagAr'.

```
1 /*Parcialis teszt
2
3 Tasnadi Istvan-Norbert Info II D*/
4
5 --Fozes tesztfeladatsor
6
7 /*1. Írj tárolt eljárást, mely módosítja egy adott alapanyag árát az üzletben.
8 (bemenet: alapanyag neve, új ár). Ha nem létezik az alapanyag, írjon ki hibüzenetet.*/
9
10 --meghívjuk ezt a következőképpen: call modositAlapanyagAr(alapanyag neve, uj ar);
11
12 select * from kategoria;
13 select * from nehezseg;
14 select * from recept;
15 select * from alapanyag;
16 select * from recept_alapanyag;
17 select * from személy;
18 select * from konyha;
19 select * from fozes;
20 select * from uzlet;
21 select * from vasarlas;
22
23 set SERVEROUT on;
24
25 /
26 create or replace procedure modositAlapanyagAr(pAlapanyagNev alapanyag.nev%type, pAr uzlet.ar%type) is
27     v_alapanyagID alapanyag.id%type;
28 begin
29     select id
30     into v_alapanyagID
```

A SÉMÁN LÉVŐ IKONOK JELENTÉSE



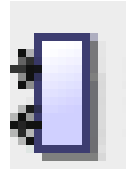
Függvény, ami futásra kész állapotú (Ezen nem lehet debugolni)



Függvény, ami nem futtatható (hibák vannak benne)



Függvény, ami debug módban is futtatható



Procedúra, ami futásra kész állapotú (Ezen nem lehet debugolni)



Procedúra, ami nem futtatható (hibák vannak benne)



Procedúra, ami debug módban is futtatható




Trigger, ami futásra kész állapotú (Ezen nem lehet debugolni)

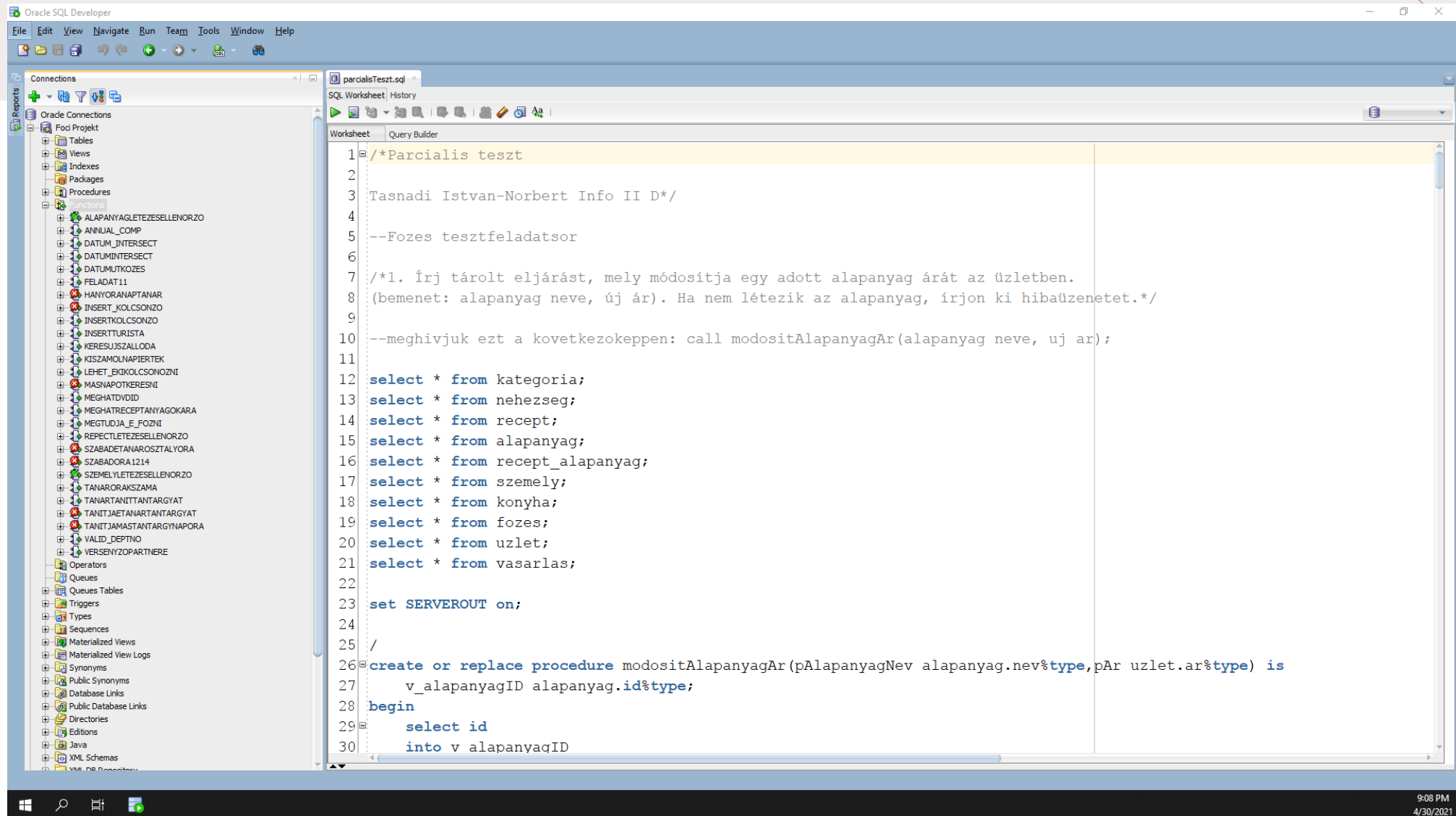


Trigger, ami debug módban is futtatható

HOGYAN DEBUGOLUNK EGY EGYSZERŰ PROCEDÚRÁT, FÜGGVÉNYT, TRIGGERET?

- 1. Kikeressük a sémából az adott procedúrát, függvényt, triggeret
- 2. Megnyitjuk az adott procedúrát, függvényt, triggeret (**egyszer rákattintunk**)
- 3. **Begin-End** közötti sorokban teszünk egy **Breakpoint** -ot
- 4. Az adott procedúrát, függvényt, triggeret **kompiláljuk debugra** (**Compile for Debug**)
- 5. A katicabogár ikonra  rákattintva megadjuk az adott procedúra, függvény, trigger **paramétereit** (ha szükséges azokat megadni), majd az **OK** gombra rányomva elindítjuk a debugolást.

1. KIKERESSÜK A SÉMÁBÓL AZ ADOTT PROCEDÚRÁT, FÜGGVÉNYT, TRIGGERT



The screenshot shows the Oracle SQL Developer interface. On the left, the 'Connections' pane shows a tree view of the database schema, including tables, views, indexes, packages, procedures, and functions. The main window displays a SQL worksheet with the following code:

```
1 /*Parcialis teszt
2
3 Tasnadi Istvan-Norbert Info II D*/
4
5 --Fozes tesztfeladatsor
6
7 /*1. Írj tárolt eljárást, mely módosítja egy adott alapanyag árát az üzletben.
8 (bemenet: alapanyag neve, új ár). Ha nem létezik az alapanyag, írjon ki hibaüzenetet.*/
9
10 --meghívjuk ezt a következőképpen: call modositAlapanyagAr(alapanyag neve, uj ar);
11
12 select * from kategoria;
13 select * from nehezseg;
14 select * from recept;
15 select * from alapanyag;
16 select * from recept_alapanyag;
17 select * from személy;
18 select * from konyha;
19 select * from fozes;
20 select * from uzlet;
21 select * from vasarlas;
22
23 set SERVEROUT on;
24
25 /
26 create or replace procedure modositAlapanyagAr(pAlapanyagNev alapanyag.nev%type, pAr uzlet.ar%type) is
27     v_alapanyagID alapanyag.id%type;
28 begin
29     select id
30     into v_alapanyagID
```

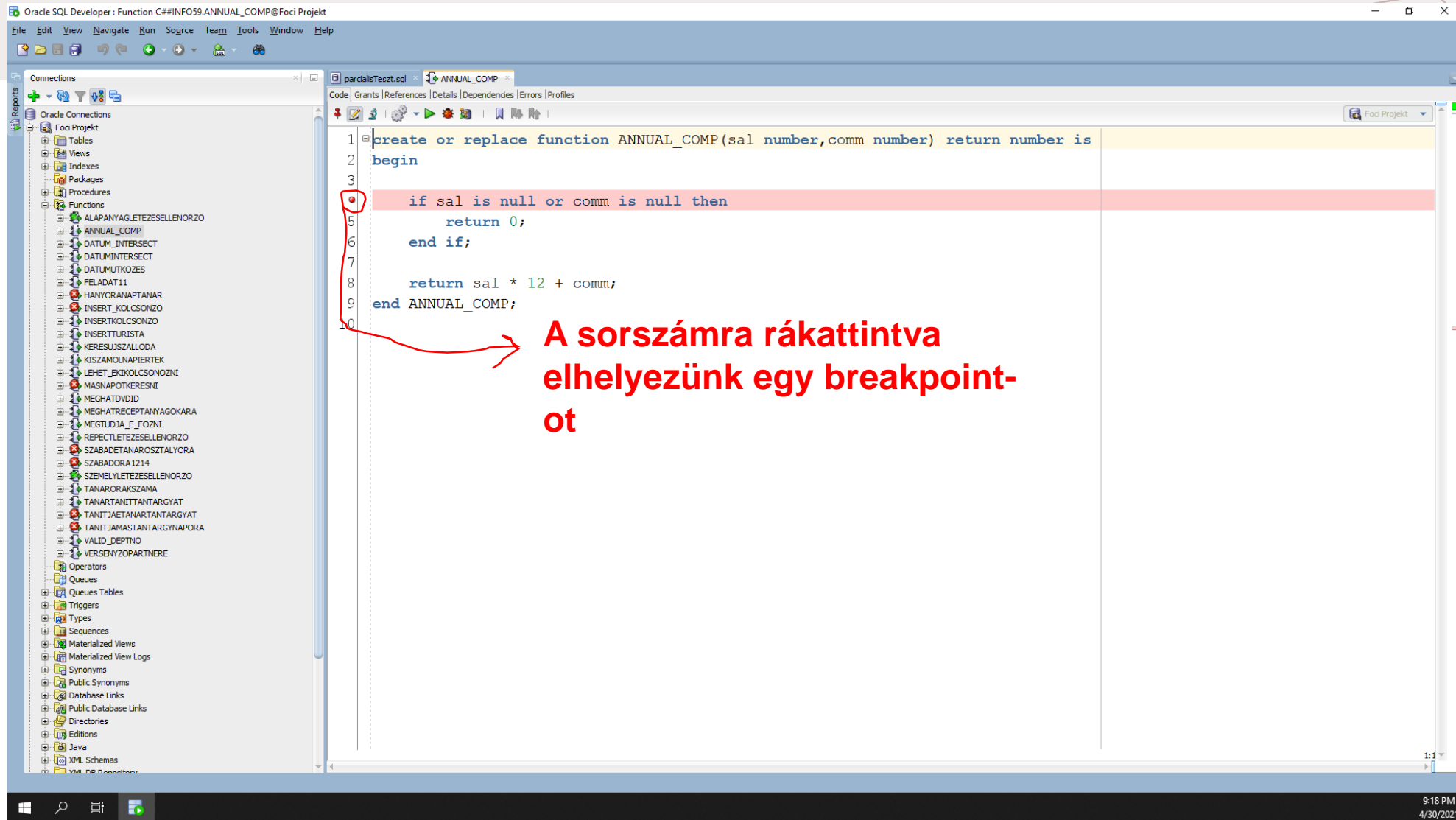
2. MEGNYITJUK AZ ADOTT PROCEDÚRÁT, FÜGGVÉNYT, TRIGGERT (EGYSZER RÁKATTINTUNK)

The screenshot shows the Oracle SQL Developer interface. On the left, the 'Connections' pane displays a tree view of database objects. The 'ANNUAL_COMP' object is highlighted with a red circle, and a red arrow points from it to the text 'Csakis egyszer rákattintunk'. The main window shows a SQL worksheet with the following code:

```
1 /*Parcialis teszt
2
3 Tasnadi Istvan-Norbert Info II D*/
4
5
6
7 /*1. Írj tárolt eljárást, mely módosítja az alapanyag árát az üzletben.
8 (bemenet: alapanyag neve, új ár). Ha nem létezik az alapanyag, írj ki hibaüzenetet.*/
9
10 --meghívjuk ezt a következőképpen: call modositAlapanyagAr(alapanyag neve, uj ar);
11
12 select * from categoria;
13 select * from nehezseg;
14 select * from recept;
15 select * from alapanyag;
16 select * from recept_alapanyag;
17 select * from személy;
18 select * from konyha;
19 select * from fozes;
20 select * from uzlet;
21 select * from vasarlas;
22
23 set SERVEROUT on;
24
25 /
26 create or replace procedure modositAlapanyagAr (pAlapanyagNev alapanyag.nev%type, pAr uzlet.ar%type) is
27     v_alapanyagID alapanyag.id%type;
28 begin
29     select id
30     into v_alapanyagID
```

Csakis egyszer
rákattintunk

3. BEGIN-END KÖZÖTTI SOROKBAN TESZÜNK EGY BREAKPOINT -OT



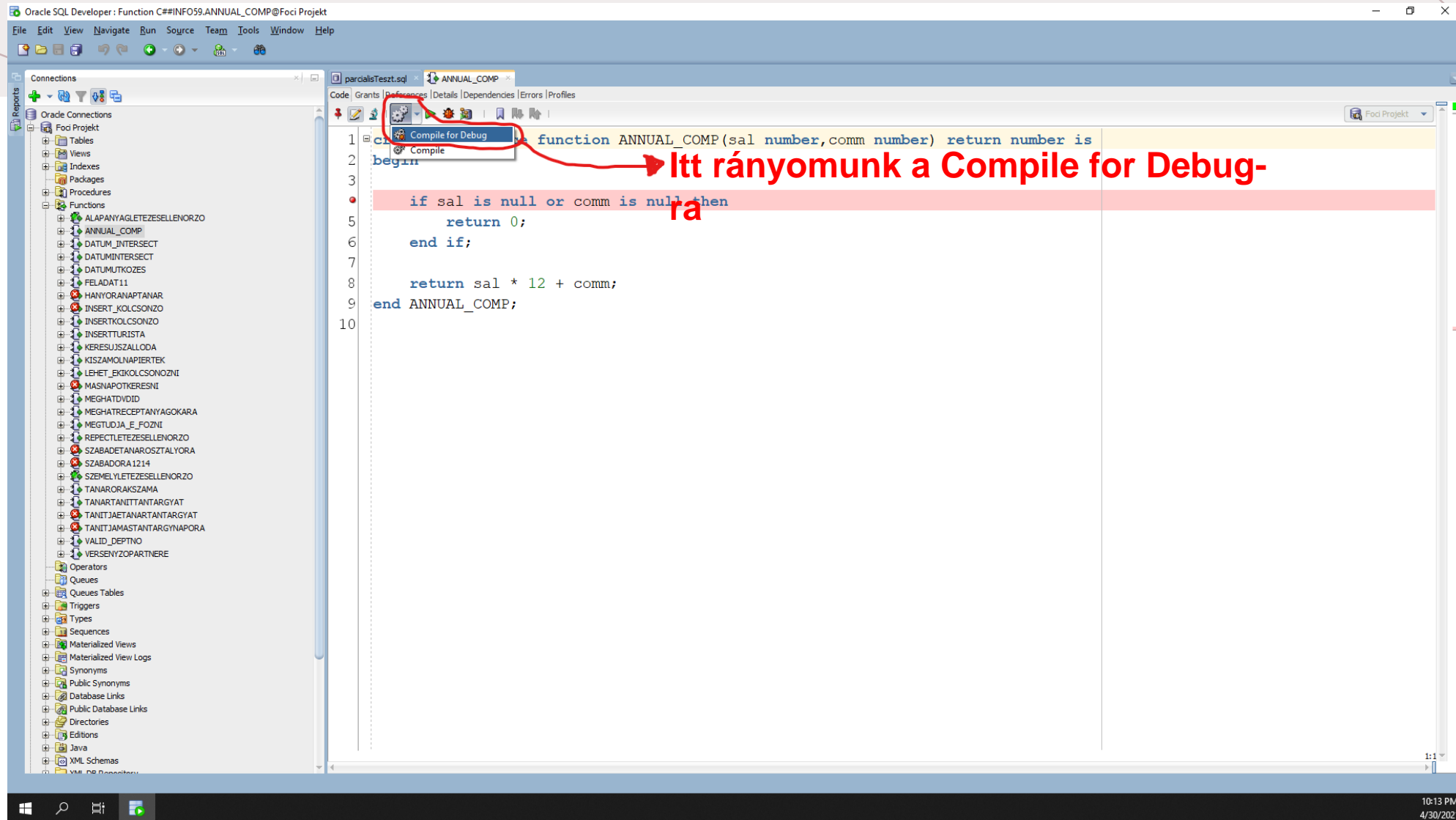
The screenshot shows the Oracle SQL Developer interface. The main window displays the following PL/SQL code:

```
1 create or replace function ANNUAL_COMP (sal number, comm number) return number is
2 begin
3
4     if sal is null or comm is null then
5         return 0;
6     end if;
7
8     return sal * 12 + comm;
9 end ANNUAL_COMP;
10
```

A red circle highlights the number '4' on the left margin, and a red arrow points from this circle to the text: **A sorszámra rákattintva elhelyezünk egy breakpoint -ot**

The interface also shows a 'Connections' pane on the left with a tree view of database objects, and a status bar at the bottom right indicating the time as 9:18 PM on 4/30/2021.

4. AZ ADOTT PROCEDÚRÁT, FÜGGVÉNYT, TRIGGERET KOMPILÁLJUK DEBUGRA (COMPILE FOR DEBUG)



The screenshot displays the Oracle SQL Developer interface. On the left, the 'Connections' pane shows a tree view of database objects, including a list of functions. The main editor window shows the SQL code for a function named 'ANNUAL_COMP'. A red circle highlights the 'Compile for Debug' menu option, with a red arrow pointing to it from the text 'Itt rányomunk a Compile for Debug-ra'. The function code is as follows:

```
1 create or replace function ANNUAL_COMP (sal number, comm number) return number is
2 begin
3
4     if sal is null or comm is null then
5         return 0;
6     end if;
7
8     return sal * 12 + comm;
9 end ANNUAL_COMP;
10
```

The system tray at the bottom right shows the time as 10:13 PM and the date as 4/30/2021.

Oracle SQL Developer : Function C##INFO59.ANNUAL_COMP@Foci Projekt

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections Foci Projekt Tables Views Indexes Packages Procedures Functions ANNUAL_COMP DATUM_INTERSECT DATUMINTERSECT DATUMUTKOZES FELADAT11 HANYORANAPTANAR INSERT_KOLCSONZO INSERTKOLCSONZO INSERTTURISTA KERESUJSZALLODA KISZAMOLNAPIERTEK LEHET_EKIKOLCSONOZNI MASNAPOTKERESNI MEGHATDVDDID MEGHATRECEPTANYAGOKARA MEGTUDJA_E_FOZNI REPECTLETEZESELLENORZO SZABADETANAROSZTALYORA SZABADORA1214 SZEMELYLETEZESELLENORZO TANARORAKSZAMA TANARTANITTANTARGYAT TANITJAEANARTANTARGYAT TANITJAMASTANTARGYNAPORA VALID_DEPTNO VERSENYZOPARTNERE Operators Queues Queues Tables Triggers Types Sequences Materialized Views Materialized View Logs Synonyms Public Synonyms Database Links Public Database Links Directories Editions Java YML Schemac

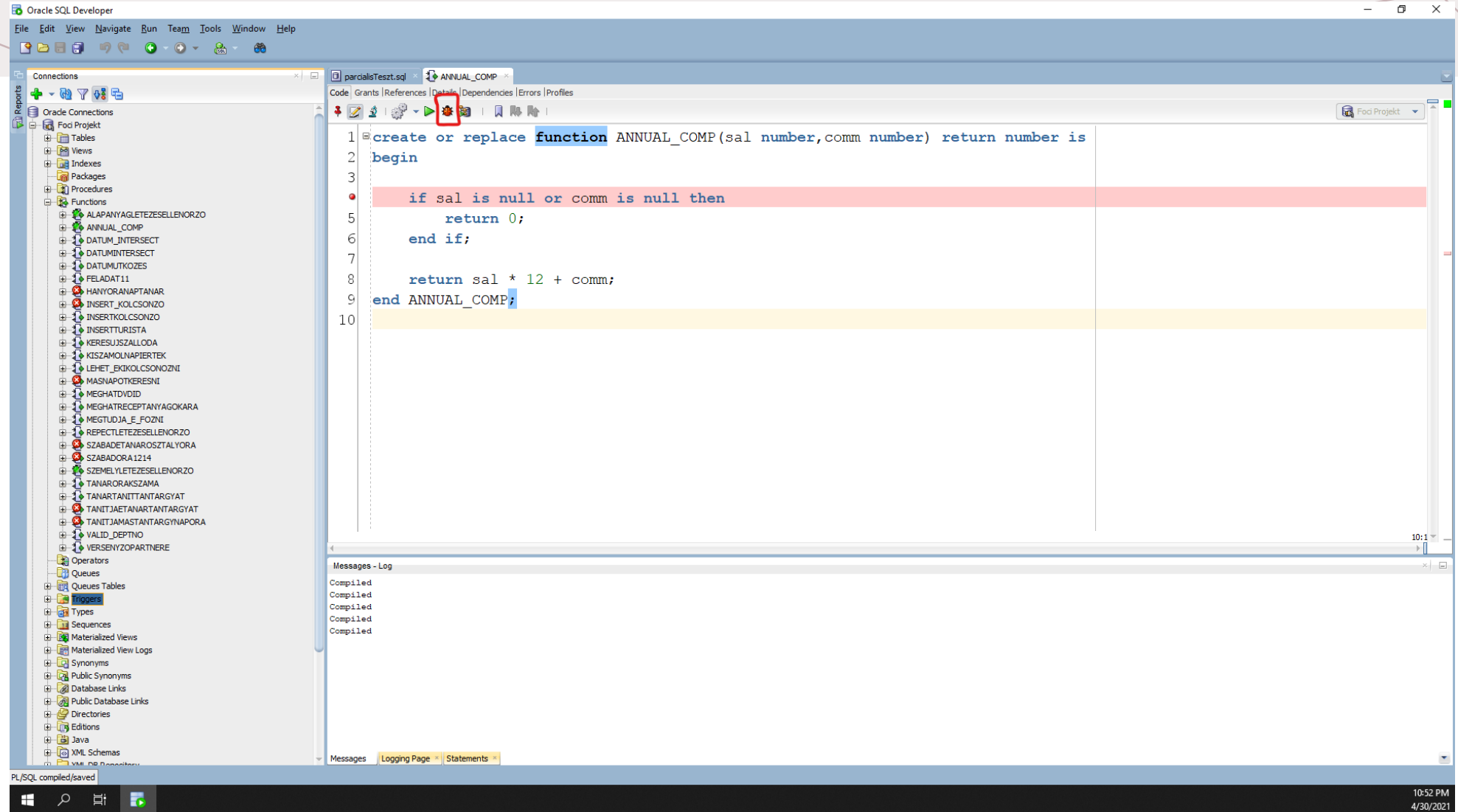
Code Grants References Details Dependencies Errors Profiles Foci Projekt

```
1 create or replace function ANNUAL_COMP (sal number, comm number) return number is
2 begin
3
4     if sal is null or comm is null then
5         return 0;
6     end if;
7
8     return sal * 12 + comm;
9 end ANNUAL_COMP;
10
```

Messages - Log
Compiled
Compiled
Compiled
Compiled

Ha így fog megjelenni, akkor az azt jelenti, hogy kompilálási módban is futtatható

5. RÁKATTINTUNK A IKONRA, HOGY EZZEL TUDJUNK MAJD DEBUGOLNI



The screenshot displays the Oracle SQL Developer interface. On the left, the 'Connections' pane shows a tree view of database objects, including a list of functions. The main editor window shows the following PL/SQL code:

```
1 create or replace function ANNUAL_COMP (sal number, comm number) return number is
2 begin
3
4     if sal is null or comm is null then
5         return 0;
6     end if;
7
8     return sal * 12 + comm;
9 end ANNUAL_COMP;
```

The code is highlighted in yellow, and a red box highlights the 'Run' (bug) icon in the toolbar. Below the code editor, the 'Messages - Log' pane shows the following output:

```
Compiled
Compiled
Compiled
Compiled
Compiled
```

The status bar at the bottom indicates 'PL/SQL compiled/saved' and the system clock shows 10:52 PM on 4/30/2021.

AZ INPUT VALUE OSZLOPNÁL MEGADJUK A PROCEDÚRA / FÜGGVÉNY PARAMÉTEREIT

The screenshot shows the Oracle SQL Developer interface. The main window displays the following PL/SQL code:

```
1 create or replace function ANNUAL_COMP (sal number, comm number) return number is
2 begin
3
4     if sal is null or comm is null then
5         return
6     end if;
7
8     return sal;
9 end ANNUAL_COMP;
```

The 'Debug PL/SQL' dialog box is open, showing the following table:

Parameter	Data Type	Mode	Input Value
<Return Value>	NUMBER	OUT	N/A
SAL	NUMBER	IN	NULL
COMM	NUMBER	IN	NULL

The 'Input Value' column is highlighted with a red box, and a red arrow points to it from the text on the right.

Dupla klikk a NULL értékekre megadjuk a procedura / függvény paramétereit (Fontos, hogy a Data Type-nak megfelelő adatot és formázást adjunk meg!)

- A **Debug PL/SQL** ablakban látható egy névtelen **PL/SQL blokk**, amelyben az **SQL Developer** a háttérben kb. így fogja végrehajtani az adott procedúrát.

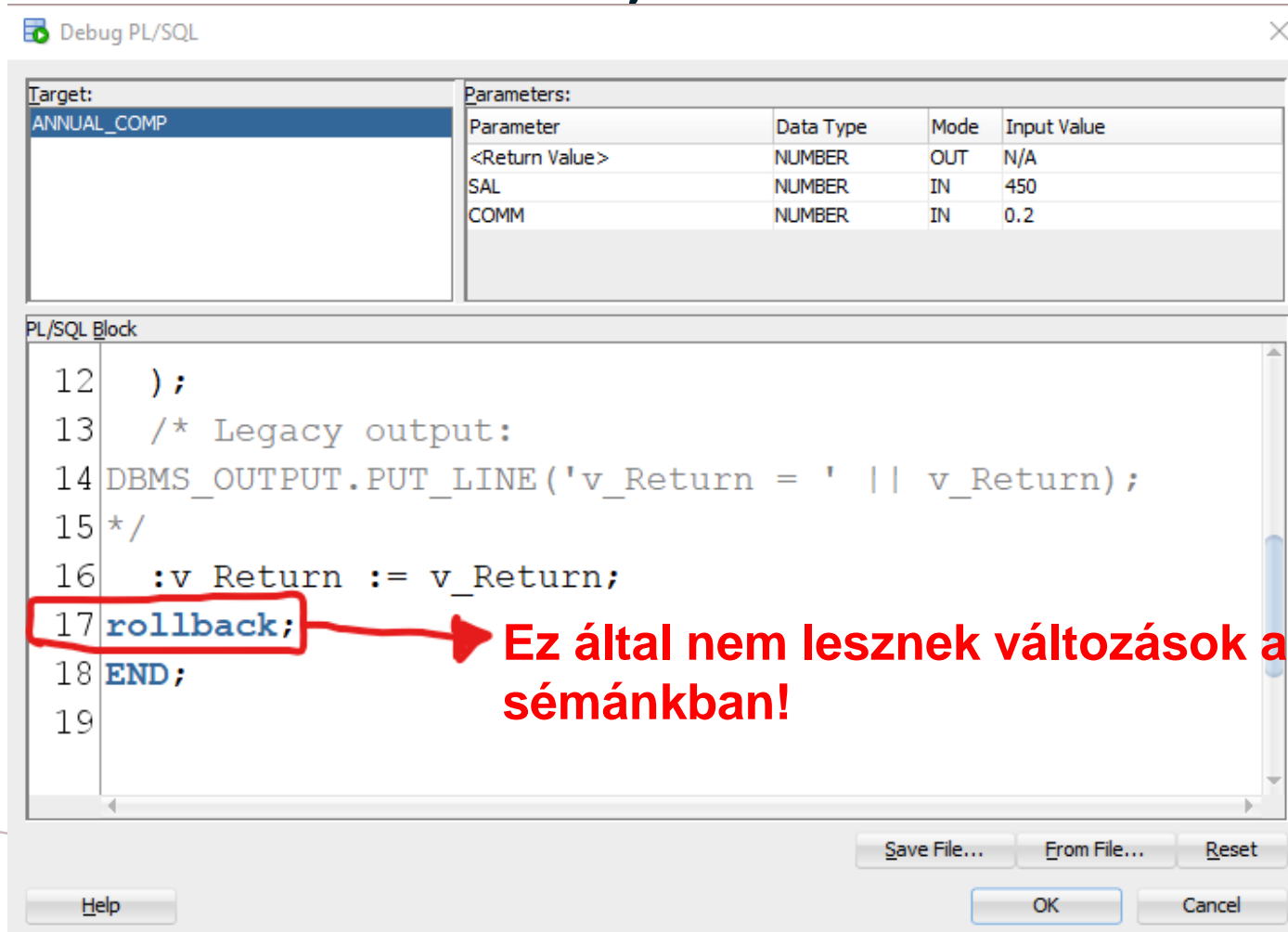
Target: ANNUAL_COMP

Parameter	Data Type	Mode	Input Value
<Return Value>	NUMBER	OUT	N/A
SAL	NUMBER	IN	450
COMM	NUMBER	IN	0.2

```
1 DECLARE
2   SAL NUMBER;
3   COMM NUMBER;
4   v_Return NUMBER;
5 BEGIN
6   SAL := 450;
7   COMM := 0.2;
8
9   v_Return := ANNUAL_COMP (
10    SAL => SAL,
11    COMM => COMM
12  );
13  /* Legacy output:
14  DBMS_OUTPUT.PUT_LINE('v_Return = ' || v_Return);
15  */
16  :v_Return := v_Return;
17  rollback;
18  END;
```

Buttons: Save File..., From File..., Reset, Help, OK, Cancel

- **Fontos megjegyzés:** A debug során a **sémánk nem fog módosulni!** A megjelenített **Debug PL/SQL** ablakban ha lejjebb görgetünk akkor látni fogjuk, hogy a PL/SQL blokk végén van egy **rollback** utasítás, ami gondoskodni fog arról, hogy a sémánkban valóban ne történjék változás.



The screenshot shows the Oracle Debug PL/SQL window. The 'Target' tab is selected, showing 'ANNUAL_COMP'. The 'Parameters' tab is also visible, showing a table with columns: Parameter, Data Type, Mode, and Input Value. The table contains three rows: '<Return Value>' (NUMBER, OUT, N/A), 'SAL' (NUMBER, IN, 450), and 'COMM' (NUMBER, IN, 0.2). The 'PL/SQL Block' tab is active, showing the following code:

```
12 );  
13 /* Legacy output:  
14 DBMS_OUTPUT.PUT_LINE('v_Return = ' || v_Return);  
15 */  
16 :v_Return := v_Return;  
17 rollback;  
18 END;  
19
```

The 'rollback;' statement on line 17 is highlighted with a red box. A red arrow points from this box to the text: **Ez által nem lesznek változások a sémánkban!**

Buttons at the bottom: Save File..., From File..., Reset, Help, OK, Cancel.

A VÉGÉN PEDIG AZ “OK” GOMBRA RÁNYOMVA ELINDITJUK DEBUGOLÁSRA

The screenshot displays the Oracle SQL Developer interface. The main window shows a PL/SQL function definition for `ANNUAL_COMP`. The function signature is `create or replace function ANNUAL_COMP (sal number, comm number) return number is`. The function body includes a conditional check: `if sal is null or comm is null then return`, followed by `end if;` and `return sal`. The function ends with `end ANNUAL_COM`. A dialog box titled "Debug PL/SQL" is open, showing the parameters for the function: `ANNUAL_COMP`. The parameters table is as follows:

Parameter	Data Type	Mode	Input Value
<Return Value>	NUMBER	OUT	N/A
SAL	NUMBER	IN	415
COMM	NUMBER	IN	0.2

The dialog box also shows a PL/SQL block with the following code:

```
1 DECLARE
2   SAL NUMBER;
3   COMM NUMBER;
4   v_Return NUMBER;
5 BEGIN
6   SAL := 415;
7   COMM := 0.2;
8
```

The "OK" button in the dialog box is highlighted with a red rectangle. The background window shows the function code with line numbers 1 through 10. The "Messages - Log" window at the bottom shows several "Compiled" messages. The system tray at the bottom right indicates the time is 11:02 PM on 4/30/2021.

EZ A FELÜLET FOGJA MAJD FOGADNI A FELHASZNÁLÓT

The screenshot displays the Oracle SQL Developer interface. The main editor window shows the following PL/SQL code:

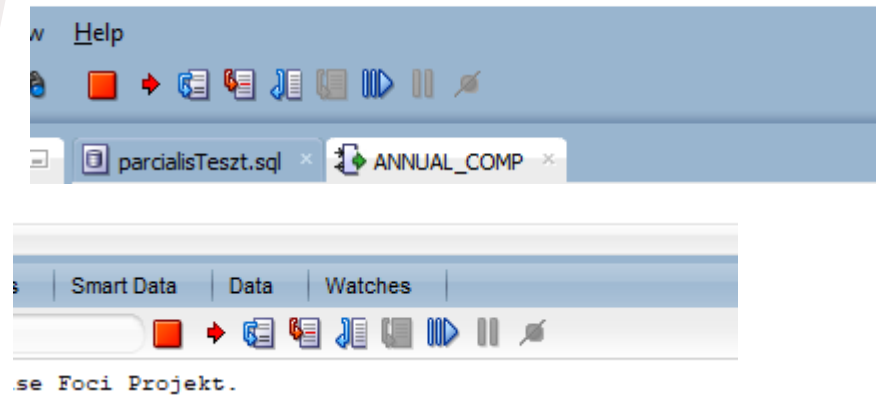
```
1 create or replace function ANNUAL_COMP (sal number, comm number) return number is
2 begin
3
4     if sal is null or comm is null then
5         return 0;
6     end if;
7
8     return sal * 12 + comm;
9 end ANNUAL_COMP;
```

The code is being executed in the debugger. The bottom pane shows the following output:






```
Statements - Log | Breakpoints | Smart Data | Data | Watches
Connecting to the database Foci Projekt.
Debugger attempting to connect to database.
Executing PL/SQL: DECLARE
    id VARCHAR2( 30 );
BEGIN
    id := DBMS_DEBUG.initialize( '192.168.0.106:1619950691305', 0 );
    DBMS_DEBUG.debug_on( TRUE );
END;
Debugger connected to database.
Source breakpoint: ANNUAL_COMP.pls:4
```

The bottom right corner of the window shows the system time and date: 2:01 PM, 5/2/2021.

NAVIGÁLÁSI MENÜ + IKON MAGYARÁZAT



Ezen a két navigálási menün keresztül tudunk majd lépegetni a debug során, kb. hasonlóan, mint más megszokott programokban (pl: Visual Studio)

-  **Stop** ikon, amivel leállíthatjuk a debugolási folyamatot
-  **Step Over** ikon, amivel ugrunk a következő utasítás sorhoz (Gyorsbillentyű : **F8**)
-  **Step Into** ikon, ami által **beléphetünk más procedúrába, függvénybe** ; ennek feltétele, hogy az adott **procedúra / függvény**, amibe be szeretnénk lépni **azok legyenek kompilálva debugolásra** (lásd [itt](#)) (Gyorsbillentyű : **F7**)
-  **Step Out** ikon, ami által **kiléphetünk a jelenlegi procedúrából / függvényből** (Gyorsbillentyű : **Shift + F7**)
-  **Resume** ikon, ami által az adott folyamat tovább fog futni és a legközelebbi **Breakpoint**-nál fog **blokkolodni** (Gyorsbillentyű : **F9**)

Description	Type
Orade exception, Persistent	Exception Breakpoint
\$Orade.Procedure.C##INFO59 FELADAT 14 12	Source Breakpoint
\$Orade.Procedure.C##INFO59 MAINKOLCSONZOFELADAT 2	Source Breakpoint
\$Orade.Trigger.C##INFO59 ELLENORIZPENZOSSZEG 1	Source Breakpoint
\$Orade.Procedure.C##INFO59 AUTORENSZAM 2	Source Breakpoint
\$Orade.Procedure.C##INFO59 SZEMELYVASAROLALAPANYAG 9	Source Breakpoint
\$Orade.Function.C##INFO59 ANNUAL_COMP 4	Source Breakpoint

- **Breakpoints**-nál tudjuk megnézni, hogy adott felhasználóknak melyik nevű procedúrájában / függvényében / triggerjében hányadik során vannak elhelyezve breakpointok , illetve itt látható, hogy mely breakpointok lettek aktiválva és melyek nem.

• Például:  \$Orade.Procedure.C##INFO59 SZEMELYVASAROLALAPANYAG 9

Hányadik sorban található az adott breakpoint

Adott breakpoint aktiválva van

Metodus típusa

User neve, amely birtokolja az adott metodust (jelen esetben procedúrát)

Metodus neve, amelyben található breakpoint (jelen esetben procedúra neve)

VÁLTOZÓK ÉRTÉKEINEK VIZSGÁLATA

- A debug során a változók jelenlegi értékeit kétféleképpen tudjuk megnézni:

1. **Data** ablaknál tudjuk megnézni az adott procedúrában, függvényben definiált összes változó jelenlegi értékét, illetve a paraméterben megadott változók értékeit is
például:



The screenshot shows a debugger interface with the 'Data' window active. The window displays a table of variables and their values. The table has three columns: 'Name', 'Value', and 'Type'. The variables shown are 'sal' with a value of 450 and 'comm' with a value of .2. Both are of type 'number'. The debugger is currently debugging a function named 'ANNUAL_COMP' at the 'begin' location. The status bar at the bottom indicates the current time is 9:1.

Name	Value	Type
sal	450	number
comm	.2	number

- **Data** ablak másik példa:

The screenshot shows an IDE with a PL/SQL procedure definition in the main editor and its execution results in the Data window below. The procedure is named `szemelyVasarolAlapanyag` and takes three parameters: `pSzemelyNev` (string), `pAlapanyagNev` (string), and `pDarab` (number). The procedure declares several local variables and performs a series of checks and queries to determine if a person has bought a certain amount of a material.

```
1 create or replace procedure szemelyVasarolAlapanyag(pSzemelyNev személy.nev%type,pAlapanyagNev alapanyag.nev%type,pDarab nu
2     v_szemelyID number;
3     v_alapanyagID number;
4     v_alapanyagDarab number;
5     v_osszAr number;
6     v_penz number;
7     v_penzSzemely number;
8 begin
9     v_szemelyID := személyLetezesEllenorzo(pSzemelyNev);
10    v_alapanyagID := alapanyagLetezesEllenorzo(pAlapanyagNev);
11
12    if v_szemelyID != -1 then
13        --letezik az adatbazisban az adott személy
14        if v_alapanyagID != -1 then
15            --ha letezett az alapanyag akkor megnezem hogy hany darab van az uzletben
16            select darab into v_alapanyagDarab from uzlet where alapanyag_id = v_alapanyagID;
17
18            if v_alapanyagDarab >= pDarab then
19
20                select penz into v_penzSzemely from személy where id = v_szemelyID;
```

The Data window shows the following variables and their values:

Name	Value	Type
pSzemelyNev	Kiss Hanna	nev%type
pAlapanyagNev	hagyma	nev%type
pDarab	10	number
v_szemelyID	NULL	number
v_alapanyagID	NULL	number
v_alapanyagDarab	NULL	number
v_osszAr	NULL	number
v_penz	NULL	number
v_penzSzemely	NULL	number

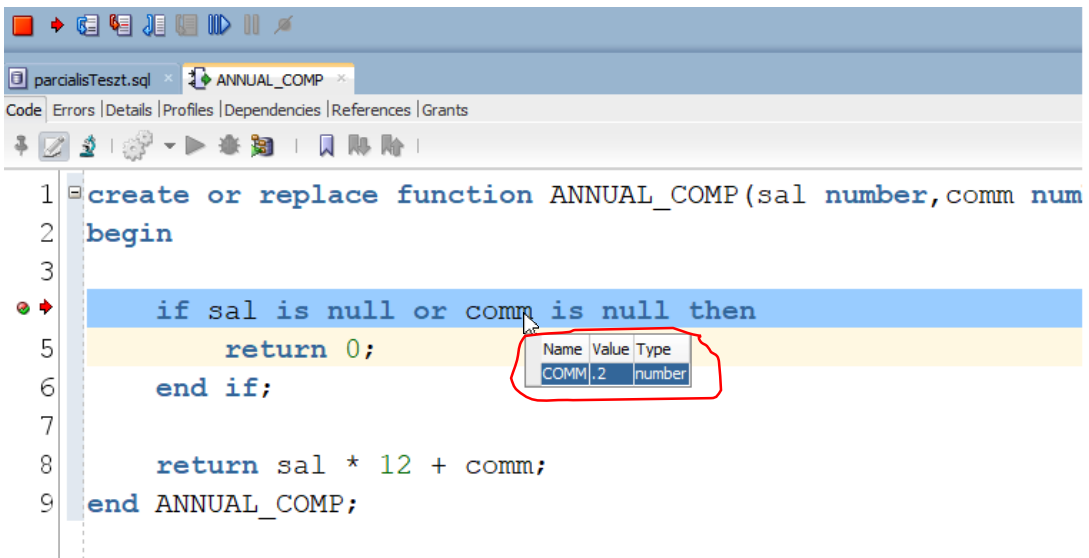
Annotations in the image point to the parameter variables (`pSzemelyNev`, `pAlapanyagNev`, `pDarab`) and the locally declared variables (`v_szemelyID`, `v_alapanyagID`, `v_alapanyagDarab`, `v_osszAr`, `v_penz`, `v_penzSzemely`) in the Data window, with the following text:

- Paraméterben lévő változók adatai
- Procedúrában deklarált változók adatai

VÁLTOZÓK ÉRTÉKEINEK VIZSGÁLATA

2. Ha az adott **változó nevére rávisszük az egeret**, akkor a debugger meg fogja jeleníteni annak a **változónak az adatait egy kisebb ablakban** (nevezetesen annak **nevét, értékét és típusát**)

például:

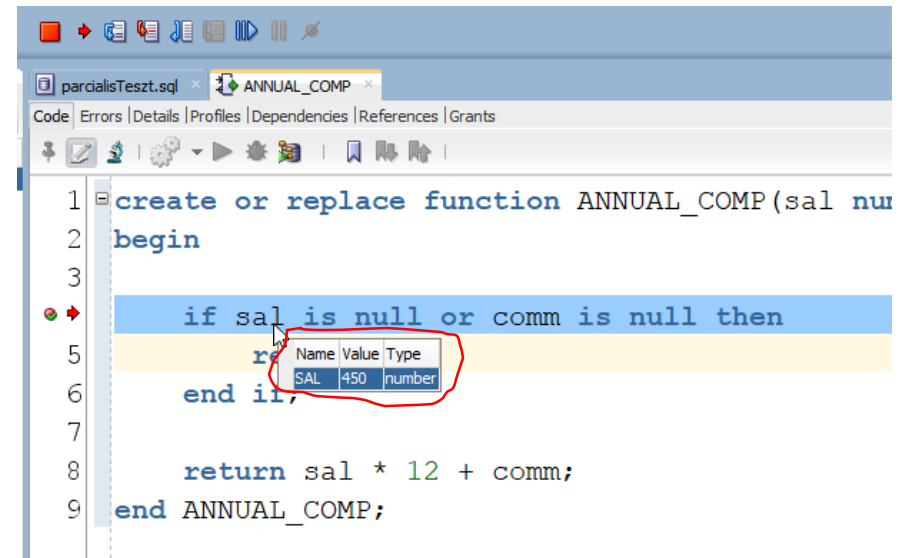


The screenshot shows a debugger window with a SQL script. The script is as follows:

```
1 create or replace function ANNUAL_COMP(sal number,comm num
2 begin
3
4 if sal is null or comm is null then
5     return 0;
6 end if;
7
8 return sal * 12 + comm;
9 end ANNUAL_COMP;
```

The line `if sal is null or comm is null then` is highlighted. A mouse cursor is hovering over the variable `comm`. A small popup window displays the following information:

Name	Value	Type
COMM	2	number



The screenshot shows the same debugger window with the same SQL script. The line `if sal is null or comm is null then` is highlighted. A mouse cursor is hovering over the variable `sal`. A small popup window displays the following information:

Name	Value	Type
SAL	450	number

HOL NÉZHETEM MEG, HOGY MIT TÉRIT VISSZA AZ ADOTT PROCEDÚRA / FÜGGVÉNY?

- A debug végén az eredményt, amit vissza fog téríteni az adott procedúra / függvény, azt az **Output Variables** ablaknál nézhetem meg.
- Például ha a korábbi slideokban bemutatott **ANNUAL_COMP** debug módban lefutatom akkor ezt az eredményt fogom látni a korábban említett **Output Variables**-nél:

function ANNUAL_COMP begin

Output Variables - Log

Variable	Value
<Return Value>	5400.2

Ez lesz az eredmény, amit az ANNUAL_COMP függvény visszatérített

Messages | Logging Page | Statements | Debugging: IdeConnections%23Foci+Projekt.jpr | Output Variables

DBMS_OUTPUT UTASÍTÁS DEBUG MÓDBAN

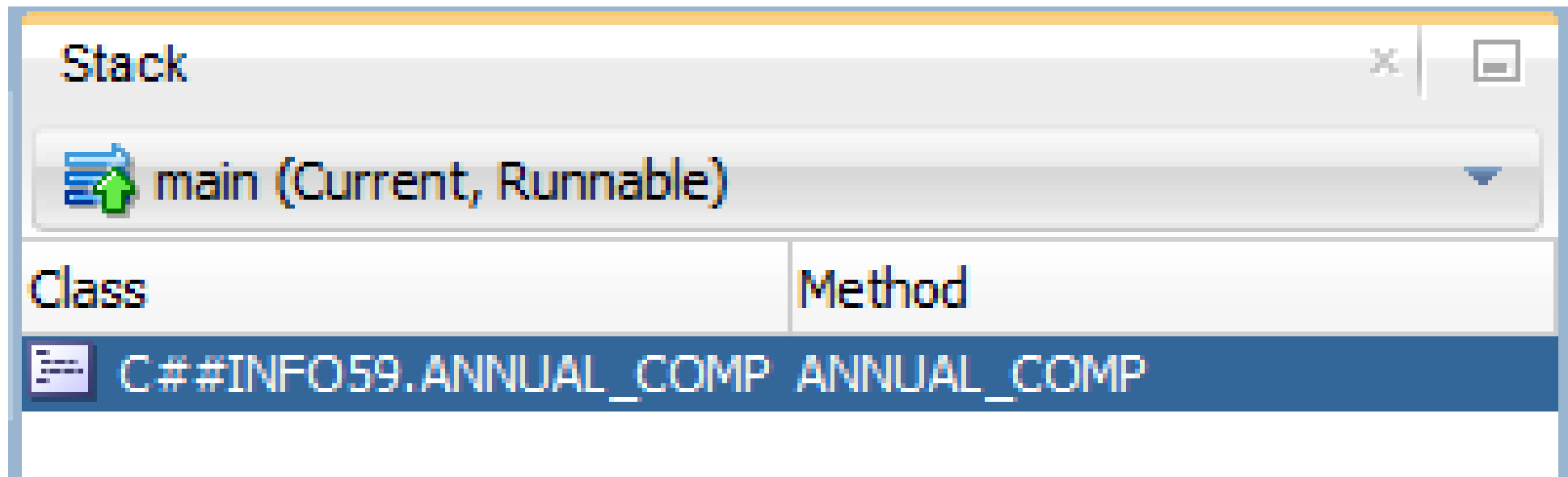
- A debug során a **dbms_output.put_line(...)** eredménye a **Debugging:IdeConnections...** ablakban fog megjelenni (kék színű szöveggel), **de csakis miután a debug folyamat véget ért.**

```
procedure személyVasarolAlapanyag begin
  id := DBMS_DEBUG.initialize( '192.168.0.106:1619971533206', 0 );
  DBMS_DEBUG.debug_on( TRUE );
END;
Debugger connected to database.
Source breakpoint: SZEMELYVASAROLALAPANYAG.pls:9
Executing PL/SQL: BEGIN
  DBMS_DEBUG.debug_off();
END;
hagyma-bol osszesen csak 5 db van az uzletben, ezert Kiss Hanna nem tud belole venni 10 darabot!
Process exited.
Disconnecting from the database Foci Projekt.
Debugger disconnected from database.
```

Dbms_output.put_line(...) eredménye, ami csakis a debug folyamat befejezte után fog megjelenni!

STACK ABLAK


- A **Stack** ablakban fogjuk látni, hogy a debugger jelenleg mely procedúrában / függvényben van , ahol látni fogjuk a **User** nevét és mellette a **procedúra / függvény nevét**



BONYOLULTABB PL/SQL PROGRAM DEBUGOLÁSA

- Az eddigi slideokban megvizsgáltuk, hogy hogyan tudunk debugolni egyszerű procedúrákat / függvényeket, amelyekben nem voltak egyéb procedúra / függvény hívások
- De viszont felvetődhet a kérdés, hogy mégis hogyan debugolhatunk egy olyan PL/SQL programot, amelyben már több különböző procedúra, függvény is meghívódik?
- Az elkövetkező slideokban erre a kérdésre próbálunk választ adni

BONYOLULTABB PL/SQL PROGRAMOK DEBUGOLÁS LÉPÉSEK

1. Kikeressük a sémából azt a PL/SQL programot (legyen az procedúra vagy függvény attól függően, hogy melyikben van több procedúra / függvény hívás)
2. Kompiláljuk azt debugra (**Compile for Debug**)
3. Elhelyezzük a **Begin-End** közötti sorokban **breakpointot**
4. Azokat a **procedúrákat / függvényeket** , amelyekbe be szeretnénk lépni (**Step Into** által) a debug során azokat is **ki kell keresni egyenként a sémából és külön azokat is kompilálni debugolásra (azokra is végrehajtani egy Compile for Debug-ot)** , mert anélkül nem fogunk tudni belépni a debug során azokba a procedúrákba / függvényekbe, függetlenül attól, hogy megnyomtuk a **Step Into** ikont!
5. Miután **külön lekompiláltuk debugra azokat a procedúrákat / függvényeket is, amelyekbe be szeretnénk majd lépni a debug során**, akkor nem mészdt más hátra , mint megnyomjuk a  ikont, megadjuk az adott PL/SQL program **paramétereit** (ha szükséges azokat megadni), majd végül az **OK** gomb által elindítjuk azt debugolásra.

• Példaképp legyen a következő Főzés nevezetű adatbázisunk megadva:

Kategória (id, név)

Nehézség (id, név)

Recept (id, név, kategória_id, nehézség_id, idő, kalória)

Alapanyag (id, név, mértékegység)

Recept_Alapanyag (recept_id, alapanyag_id, darab)

Személy (id, név, pénz, nehézség_id)

Konyha (személy_id, alapanyag_id, darab)

Főzés (személy_id, recept_id, dátum)

Üzlet (alapanyag_id, ár, darab)

Vásárlás (személy_id, alapanyag_id, dátum, darab)

- A következő feladat van megadva:

2. Írj tárolt eljárást, mely segítségével megvalósítható egy alapanyag megvásárlása egy adott személy által.

Bemenet: személy neve, alapanyag neve, darab.

Ellenőrizd, hogy létezik-e az adott személy és alapanyag. Ha az alapanyag nem létezett, akkor szúrja be új adatként az alapanyag táblába (ár: 1, mértékegység: db). Ha létezett, nézze meg hány darab van belőle az üzletben és csak akkor tudja megvenni ha van elég. Ha meg tudja venni, szúrja be a vásárlás táblába és csökkentse a személy pénzét annyival, amennyibe került az alapanyag.

A következő slideokban meg lesznek adva a feladathoz tartozó PL/SQL program, illetve azon procedúrái, függvényei, amiket meghívunk majd.

```
/
create or replace function szemelyLetezesEllenorzo(pSzemelyNev szemely.nev%type) return number is
    v_szemelyID number;
begin
    select id
    into v_szemelyID
    from szemely
    where nev = pSzemelyNev;

    return v_szemelyID;

    exception when no_data_found then return -1;
    when others then dbms_output.put_line('Hiba leptt fel a szemelyLetezesEllenorzo fuggvenyben');
end szemelyLetezesEllenorzo;
/

/
create or replace function alapanyagLetezesEllenorzo(pAlapanyagNev alapanyag.nev%type) return number is
    v_alapanyagID number;
begin
    select id
    into v_alapanyagID
    from alapanyag
    where nev = pAlapanyagNev;

    return v_alapanyagID;

    exception when no_data_found then return -1;
    when others then dbms_output.put_line('Hiba leptt fel az alapanyagLetezesEllenorzo fuggvenyben!');
end alapanyagLetezesEllenorzo;
/
```

```
/
create or replace procedure insertAlapanyag(pAlapanyagNev alapanyag.nev%type, pMertekegyseg alapanyag.mertekegyseg%type default 'db') is
    v_id number;
begin
    insert into alapanyag values((select max(id)+1 from alapanyag),pAlapanyagNev,pMertekegyseg)
    returning id into v_id;
    dbms_output.put_line('Sikerult beszurni a(z) ' || pAlapanyagNev || ' az alapanyag tablaba!');

    insert into uzlet values (v_id,1,10);
    dbms_output.put_line('Sikerult beszurni a(z) ' || pAlapanyagNev || ' az uzlet tablaba!');

    exception when others then dbms_output.put_line('Hiba lepett fel az insertAlapanyag proceduraban!');
end insertAlapanyag;
/
```

```
/
create or replace procedure insertSzemely(pSzemelyNev szemely.nev%type,pPenz szemely.penz%type default 0 , pNehezseg_id szemely.nehezseg_id%type default 1) is
begin
    insert into szemely values((select max(id) + 1 from szemely),pSzemelyNev,pPenz,pNehezseg_id);
    dbms_output.put_line('Sikeresen be lett szurva a szemely tablaba a kovetkezo szemely: ' || pSzemelyNev);

    exception when others then dbms_output.put_line('Hiba lepett az insertSzemely proceduraban!');
end insertSzemely;
/
```

```

/
create or replace procedure személyVasarolAlapanyag(pSzemelyNev személy.nev%type,pAlapanyagNev alapanyag.nev%type,pDarab number) is
  v_szemelyID number;
  v_alapanyagID number;
  v_alapanyagDarab number;
  v_osszAr number;
  v_penz number;
  v_penzSzemely number;
begin
  v_szemelyID := személyLetezesEllenorzo(pSzemelyNev);
  v_alapanyagID := alapanyagLetezesEllenorzo(pAlapanyagNev);

  if v_szemelyID != -1 then
    --letezik az adatbazisban az adott személy
    if v_alapanyagID != -1 then
      --ha letezett az alapanyag akkor megnezem hogy hany darab van az uzletben
      select darab into v_alapanyagDarab from uzlet where alapanyag_id = v_alapanyagID;

      if v_alapanyagDarab >= pDarab then

        select penz into v_penzSzemely from személy where id = v_szemelyID;

        select ar into v_penz from uzlet where alapanyag_id = v_alapanyagID;

        v_osszAr := v_penz * pDarab;

        if v_osszAr <= v_penzSzemely then
          --ha van elegendo darab az adott alapanyagbol es penze a személynek, akkor meg tudja azokat vasarolni
          insert into vasarlas values(v_szemelyID,v_alapanyagID,pDarab,sysdate);

          update személy set penz = penz - v_osszAr where id = v_szemelyID;
          dbms_output.put_line('Sikeres vasarlas tortent,es az adott személynek a penze is sikeresen le lett csökkentve!');
        end if;
      end if;
    end if;
  end if;
end;

```



```

--mivelhog az adott személy megvásárolta a recepthez szükséges anyagokat, ezért az üzletből is csökkenni fog az alapanyagok száma(darabszáma)
update uzlet set darab = darab - pDarab where alapanyag_id = v_alapanyagID;
else
  dbms_output.put_line(pSzemelyNev || ' nincs elegendő pénz, hogy vásároljon ' || pDarab || ' darab ' || pAlapanyagNev || '-t!');
end if;
else
  dbms_output.put_line(pAlapanyagNev || '-ből összesen csak ' || v_alapanyagDarab || ' db van az üzletben, ezért ' || pSzemelyNev || ' nem tud belőle venni ' || pDarab || ' darabot!');
end if;
else
  --nem létezik az adott alapanyag az adatbázisban ezért be kell majd ezt szúrjam
  dbms_output.put_line(pAlapanyagNev || ' nem létezik az adatbázisban, ezért ez be lesz szúrva!');
  insertAlapanyag(pAlapanyagNev);
end if;
else
  dbms_output.put_line(pSzemelyNev || ' nem létezik az adatbázisban, ezért ez be lesz szúrva!');
  insertSzemely(pSzemelyNev);
end if;

exception when others then dbms_output.put_line('Hiba lépett fel a személyVasarolAlapanyag procedurában!');
end személyVasarolAlapanyag;
/

```

Most pedig ezt a **szemelyVasarolAlapanyag** procedúrát fogjuk debugolni

1. MEGKERESSÜK EZT A PROCEDÚRÁT A SÉMÁNKBÓL



The screenshot shows the Oracle SQL Developer interface. On the left, the 'Connections' pane displays a tree view of the database schema. The 'Procedures' folder is expanded, and the procedure 'SZEMELYVASAROLALAPANYAG' is highlighted with a red box. An arrow points from this box to a red text box containing the instruction: **Ezt kompiláljuk debugra (Compile for debug)**.

The main editor window shows the PL/SQL code for the procedure 'szemelyVasarolAlapanyag'. The code is as follows:

```
112 /
113 /
114 /
115 create or replace procedure szemelyVasarolAlapanyag (pSzemelyNev személy.nev%type, pAlapanyagNev alapanyag.nev
116     v_szemelyID number;
117     v_alapanyagID number;
118     v_alapanyagDarab number;
119     v_osszAr number;
120     v_penz number;
121     v_penzSzemely number;
122 begin
123     v_szemelyID := szemelyLetezesEllenorzo (pSzemelyNev);
124     v_alapanyagID := alapanyagLetezesEllenorzo (pAlapanyagNev);
125
126     if v_szemelyID != -1 then
127         --letezik az adatbazisban az adott személy
128         if v_alapanyagID != -1 then
129             --ha letezett az alapanyag akkor megnezem hogy hany darab van az uzletben
130             select darab into v_alapanyagDarab from uzlet where alapanyag_id = v_alapanyagID;
131
132             select penz into v_penzSzemely from szemely where id = v_szemelyID;
133
134             select ar into v_penz from uzlet where alapanyag_id = v_alapanyagID;
135
136             v_osszAr := v_penz * pDarab;
```

2. TEGYÜNK BREAKPOINTOT A BEGIN ÉS END SOROK KÖZÖTT

The screenshot displays the Oracle SQL Developer interface. The main window shows a PL/SQL procedure named `szemelyVasarolAlapanyag` with several variables and a `begin` block. A red highlight is placed on the first line of the `begin` block, indicating a breakpoint. The procedure code is as follows:

```
1 create or replace procedure szemelyVasarolAlapanyag(pSzemelyNev személy.nev%type,pAlapanyagNev alapanyag.nev%type)
2   v_szemelyID number;
3   v_alapanyagID number;
4   v_alapanyagDarab number;
5   v_osszAr number;
6   v_penz number;
7   v_penzSzemely number;
8 begin
9   v_szemelyID := szemelyLetezesEllenorzo(pSzemelyNev);
10  v_alapanyagID := alapanyagLetezesEllenorzo(pAlapanyagNev);
11
12  if v_szemelyID != -1 then
13    --letezik az adatbazisban az adott személy
14    if v_alapanyagID != -1 then
15      --ha letezett az alapanyag akkor megnezem hogy hany darab van az uzletben
16      select darab into v_alapanyagDarab from uzlet where alapanyag_id = v_alapanyagID;
17
18  if v_alapanyagDarab >= pDarab then
```

The bottom window shows the execution log, which includes the following messages:

```
id := DBMS_DEBUG.initialize( '192.168.0.106:1619972529075', 0 );
DBMS_DEBUG.debug_on( TRUE );
END;
Debugger connected to database.
ORA-01400: cannot insert NULL into ("C##INFO59"."SZEMELY"."ID")
ORA-06512: at line 3
Executing PL/SQL: BEGIN
      DBMS_DEBUG.debug_off();
END;
Process exited.
Disconnecting from the database Foci Projekt.
Debugger disconnected from database.
```

The system tray at the bottom right shows the date and time: 11:39 PM, 5/2/2021.

- Továbbá csináljuk a következőket:

- kompiláljuk az **alapanyagLetezesEllenorzo** függvényt debugolásra (Compile for Debug)
- És a **szemelyLetezesEllenorzo** függvényt pedig ne kompiláljuk debugolásra

The screenshot shows the Oracle SQL Developer interface. On the left, the 'Functions' folder is expanded, showing a list of functions. Two functions are highlighted with red boxes and arrows: 'ALAPANYAGLETEZESELENORZO' and 'SZEMELYLETEZESELENORZO'. The main editor window displays the PL/SQL code for the 'szemelyVasarolAlapanyag' procedure. The code includes a call to 'szemelyLetezesEllenorzo' on line 9, which is highlighted in red. The debug log at the bottom shows the execution of the procedure, including the initialization of the debug environment and the execution of the PL/SQL code.

```
1 create or replace procedure szemelyVasarolAlapanyag(pSzemelyNev szemely.nev%type,pAlapanyagNev alapanyag.nev%type)
2   v_szemelyID number;
3
4   v_penz number;
5   v_penzSzemely number;
6
7 begin
8   v_szemelyID := szemelyLetezesEllenorzo(pSzemelyNev);
9   v_alapanyagID := alapanyagLetezesEllenorzo(pAlapanyagNev);
10
11   if v_szemelyID != -1 then
12     --letezik az adatbazisban az adott szemely
13     if v_alapanyagID != -1 then
14
15
16
17   if v_alapanyagDarab >= pDarab then
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

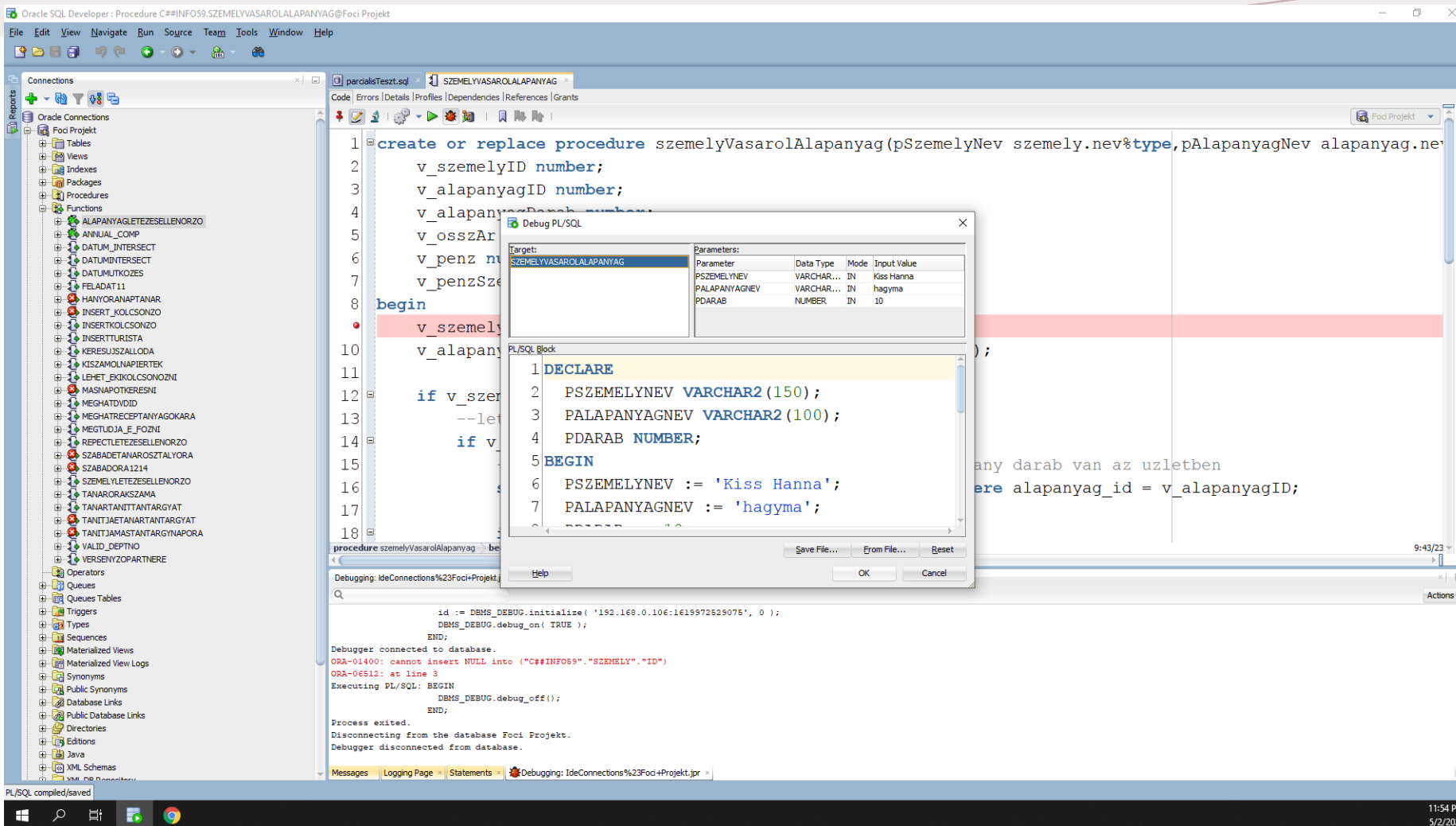
Ezt kompiláljuk debugolásra (Compile for Debug)

Ezt pedig NEM kompiláljuk debugolásra

Debugging: IdeConnections%23Foci-Projekt.pr - Log

```
id := DBMS_DEBUG.initialize( '192.168.0.106:1619972629076', 0 );
DBMS_DEBUG.debug_on( TRUE );
END;
Debugger connected to database.
ORA-01400: cannot insert NULL into ("C##INF065"."SZEMELY"."ID")
ORA-06512: at line 9
Executing PL/SQL: BEGIN
DBMS_DEBUG.debug_off();
END;
Process exited.
Disconnecting from the database Foci Projekt.
Debugger disconnected from database.
```

A IKONRA RÁNYOMVA MEGADJUK A PROCEDÚRA PARAMÉTEREIT, MAJD AZ "OK" GOMBRA RÁKATTINTVA ELINDÍTJUK EZT DEBUGOLÁSRA



Oracle SQL Developer: Procedure C##INFO59.SZEMELYVASAROLALAPANYAG@Foci Projekt

File Edit View Navigate Run Source Team Tools Window Help

Connections

Oracle Connections

Foci Projekt

Tables

Views

Indexes

Packages

Procedures

Functions

ALAPANYAGLETEZESLENFORZO

ANNUAL_COMP

DATUM_INTERSECT

DATUMINTERSECT

DATUMUKOZES

FELADAT11

HANYORANAPTANAR

INSERT_KOLCSONZO

INSERTKOLCSONZO

INSERTTURISTA

KERESUSZALLODA

KISZAMOLNAPIRTEK

LEHET_EKIKOLCSONOZNI

MASNAPOTKERESNI

MEGHATDVID

MEGHATRECEPTANYAGOKARA

MEGTUDJA_E_FOZNI

REPECTLETEZESLENFORZO

SZABADETANAROSZTALYORA

SZABADORA1214

SZEMELYLETEZESLENFORZO

TANARORAKSZAMA

TANARTANITANTARGYAT

TANTAJAETANARTANTARGYAT

TANTAJAMASTANTARGYINAPORA

VALID_DEPTNO

VERSENYZOPARTNERE

Operators

Queues

Queues Tables

Triggers

Types

Sequences

Materialized Views

Materialized View Logs

Synonyms

Public Synonyms

Database Links

Public Database Links

Directories

Editions

Java

XML Schemas

XML NS Dependencies

parcialsTeszt.sql

SZEMELYVASAROLALAPANYAG

Code Errors Details Profiles Dependencies References Grants

Foci Projekt

```
1 create or replace procedure szemelyVasarolAlapanyag(pSzemelyNev szemely.nev%type,pAlapanyagNev alapanyag.nev%type)
2   v_szemelyID number;
3   v_alapanyagID number;
4   v_alapanyagDarab number;
5   v_osszar number;
6   v_penz number;
7   v_penzSzam number;
8 begin
9   v_szemelyID := v_szemelyID;
10  v_alapanyagID := v_alapanyagID;
11  if v_szemelyID is null then
12    --let
13  end if
14  if v_alapanyagID is null then
15  end if
16  any darab van az uzletben
17  are alapanyag_id = v_alapanyagID;
18  end;
```

Debug PL/SQL

Target: SZEMELYVASAROLALAPANYAG

Parameter	Data Type	Mode	Input Value
PSZEMELYNEV	VARCHAR2	IN	Kiss Hanna
PALAPANYAGNEV	VARCHAR2	IN	hagyma
PDARAB	NUMBER	IN	10

PL/SQL Block



```
1 DECLARE
2   PSZEMELYNEV VARCHAR2(150);
3   PALAPANYAGNEV VARCHAR2(100);
4   PDARAB NUMBER;
5 BEGIN
6   PSZEMELYNEV := 'Kiss Hanna';
7   PALAPANYAGNEV := 'hagyma';
```

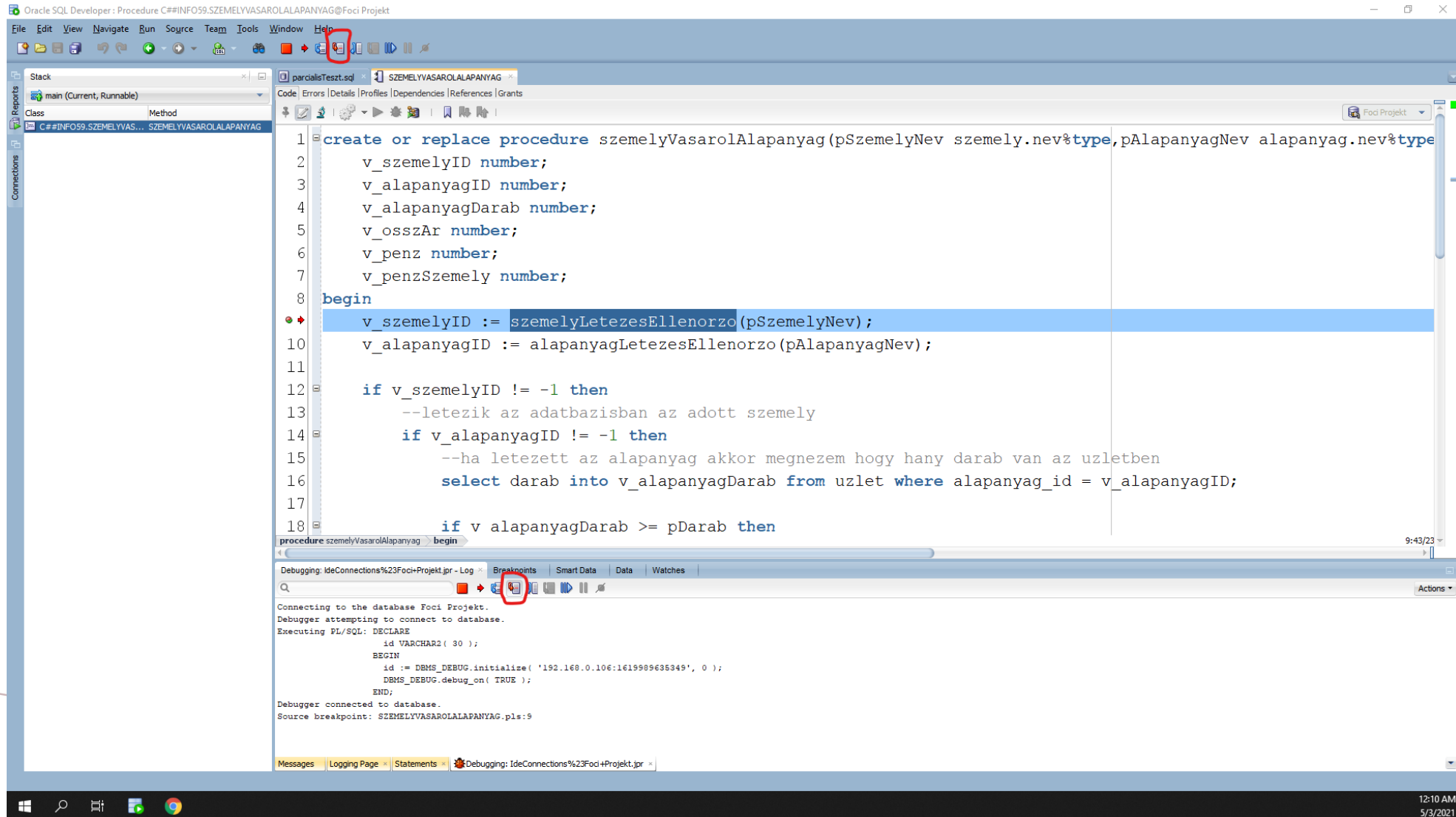
Debugging: IdeConnections%23Foci+Projekt...

Messages Logging Page Statements Debugging: IdeConnections%23Foci+Projekt.jp...

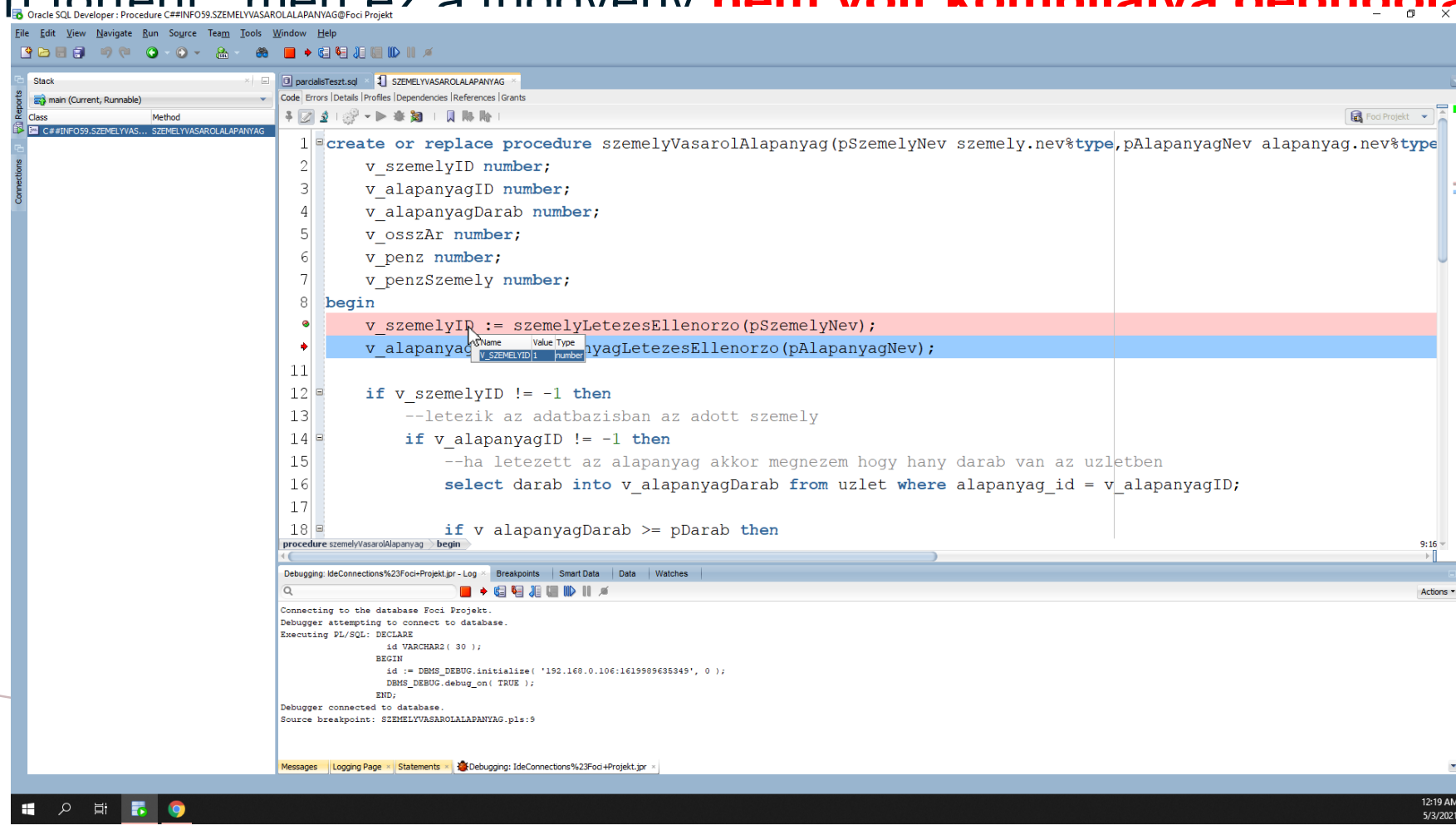
PL/SQL compiled/saved

11:54 PM 5/2/2021

- Amint látható a program megállt a 9-dik sorban.
- Most próbáljunk belépni ebbe a **szemelyLetezesEllenorzo** függvénybe a **Ste**  **Into** gomb által ( ikonra rákattintva vagy F7-et nyomva)

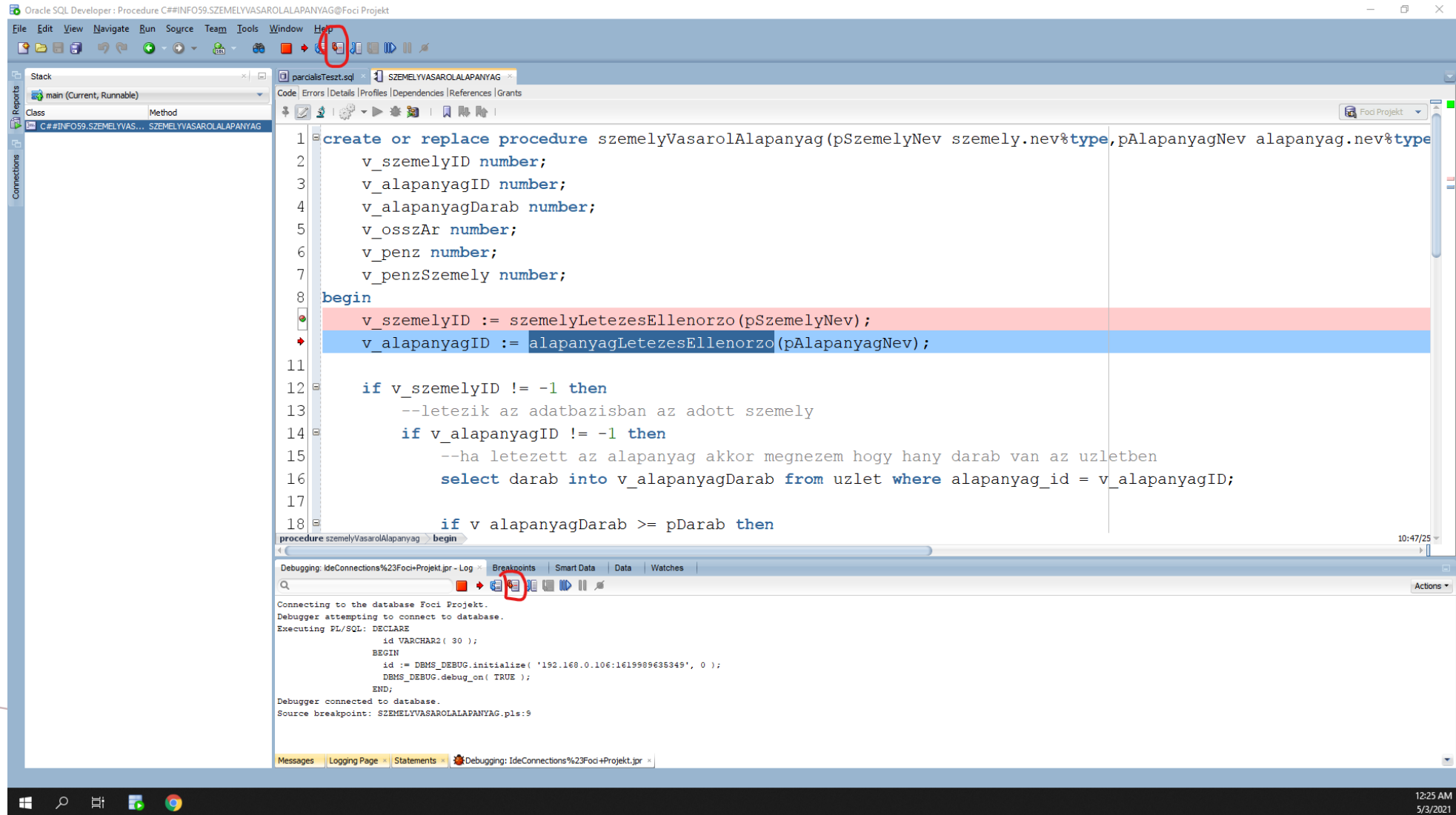


- Amint látható, hiába kattintottunk a **Step Into** gombra, a debugger **nem lépett be** ebbe a **szemelyLetezesEllenorzo** függvénybe, hanem azt csakis végrehajtotta és az eredményt visszatérítette a **v_szemelyID** változóba
- ez azért történt mert ez a függvény **nem volt kompilálva debuggolásra**

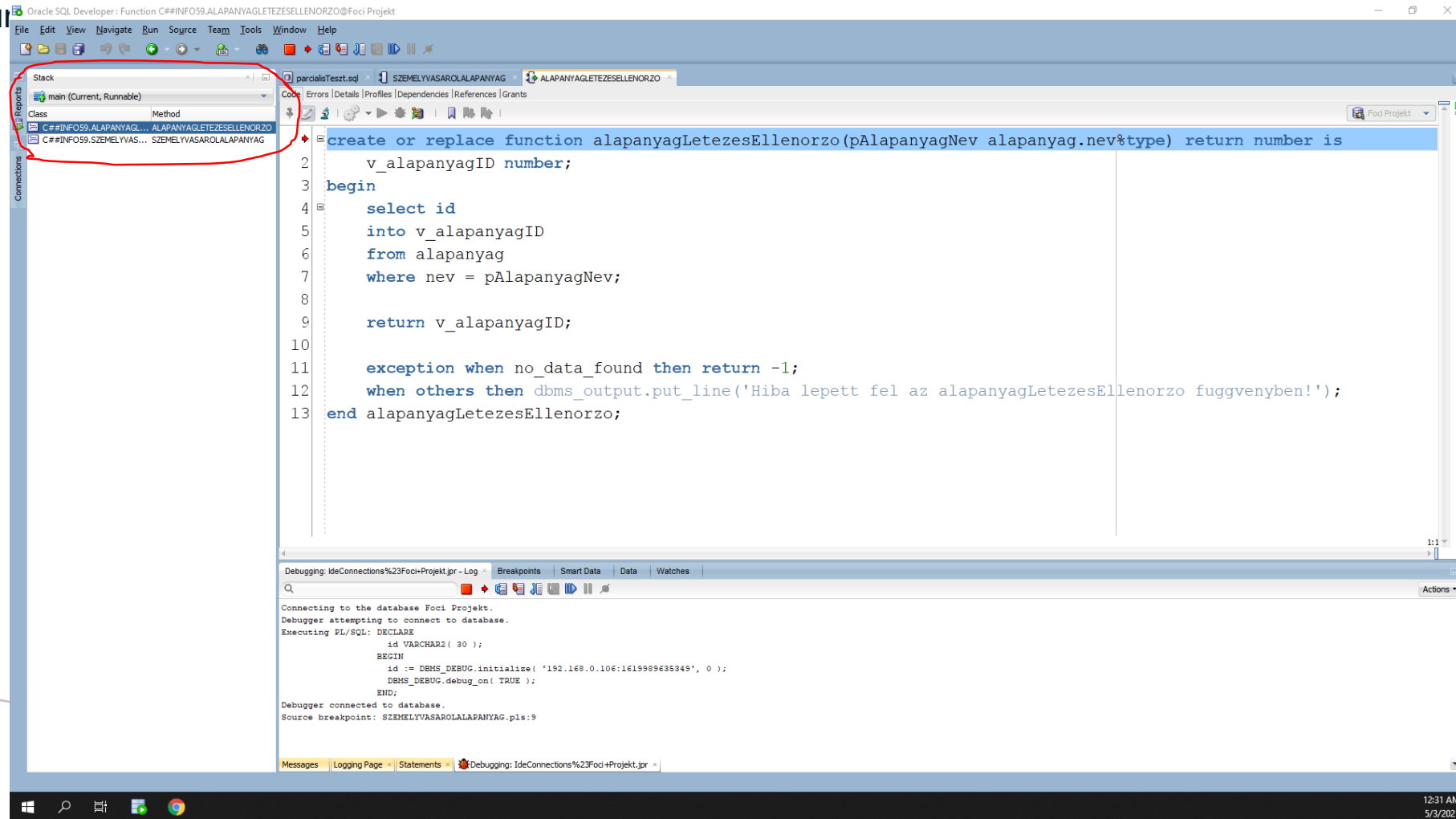


```
Oracle SQL Developer : Procedure C#\INFO59.SZEMELYVASAROLALAPANYAG@Foci Projekt
File Edit View Navigate Run Source Team Tools Window Help
parolisTeszt.sql SZEMELYVASAROLALAPANYAG
Code Errors Details Profiles Dependencies References Grants
Stack
main (Current, Runnable)
Class
C#\INFO59.SZEMELYVAS... SZEMELYVASAROLALAPANYAG
Connections
1 create or replace procedure szemelyVasarolAlapanyag (pSzemelyNev szemely.nev%type, pAlapanyagNev alapanyag.nev%type
2     v_szemelyID number;
3     v_alapanyagID number;
4     v_alapanyagDarab number;
5     v_osszAr number;
6     v_penz number;
7     v_penzSzemely number;
8 begin
9     v_szemelyID := szemelyLetezesEllenorzo (pSzemelyNev);
10    v_alapanyagID := szemelyLetezesEllenorzo (pAlapanyagNev);
11
12    if v_szemelyID != -1 then
13        --letezik az adatbazisban az adott szemely
14        if v_alapanyagID != -1 then
15            --ha letezett az alapanyag akkor megnezem hogy hany darab van az uzletben
16            select darab into v_alapanyagDarab from uzlet where alapanyag_id = v_alapanyagID;
17
18            if v_alapanyagDarab >= pDarab then
19
20            end if;
21        end if;
22    end if;
23    v_osszAr := v_osszAr + v_penz;
24    v_penz := v_penz + v_penzSzemely;
25 end;
procedure szemelyVasarolAlapanyag begin
Debugging: IdeConnections%23Foci+Projekt.pr - Log Breakpoints Smart Data Data Watches
Connecting to the database Foci Projekt.
Debugger attempting to connect to database.
Executing PL/SQL: DECLARE
    id VARCHAR2( 30 );
BEGIN
    id := DBMS_DEBUG.initialize( '192.168.0.106:1615989638345', 0 );
    DBMS_DEBUG.debug_on( TRUE );
END;
Debugger connected to database.
Source breakpointint: SZEMELYVASAROLALAPANYAG.pls:9
Messages Logging Page Statements Debugging: IdeConnections%23Foci+Projekt.pr
```

- Most próbáljunk belépni az **alapanyagLetezesEllenorzo** függvénybe (**Step Into** segítségével)



- Amint látható most bent vagyunk az **alapanyagLetezesEllenorzo** függvényben, ahol tovább tudunk debugolni használva a **Step Over** gombot (F8 gombot nyomva)
- Ez azért történt meg a másik függvénnyel szemben, mert ezt az **alapanyagLetezesEllenorzo** függvényt **kompiláltuk debugolásra** (míg a **szemelyLetezesEllenorzo** függvényt nem)
- **Ez a dolog érvényes procedúrákra is , nemcsak a függvényekre!**
- Ugyanakkor a **Stack** ablakban is megjelenik, hogy per pillanat az **alapanyagLetezesEllenorzo** függvényben vagyunk



- Ha a **Step Over**-t használtuk, hogy kilépünk az **alapanyagLetezesEllenorzo** függvényből, akkor látni fogjuk, hogy visszakerülünk a **szemelyVasarolAlapanyag** procedúrába, ahol a program a következő sorhoz fog ugorni
- Amit visszatért az **alapanyagLetezesEllenorzo** függvény, az el lesz tárolva a **v_alapanyagID** változóba
- Ugyanakkor azt is látjuk, hogy a Stack-nél az **alapanyagLetezesEllenorzo** függvény eltűnt és újra csak a **szemelyVasarolAlapanyag** procedúra van, ez által jelezvén nekünk, hogy újra a **szemelyVasarolAlapanyag** procedúrában vagyunk

The screenshot shows the Oracle SQL Developer interface during a debug session. The Stack window on the left is circled in red, showing 'main (Current, Runnable)' and 'C#INFO59.SZEMELYVASAROLALAPANYAG'. The code editor displays the following PL/SQL code:

```

1 create or replace procedure szemelyVasarolAlapanyag(pSzemelyNev szemely.nev%type,pAlapanyagNev alapanyag.nev%type
2     v_szemelyID number;
3     v_alapanyagID number;
4     v_alapanyagDarab number;
5     v_osszAr number;
6     v_penz number;
7     v_penzSzemely number;
8 begin
9     v_szemelyID := szemelyLetezesEllenorzo(pSzemelyNev);
10    v_alapanyagID := alapanyagLetezesEllenorzo(pAlapanyagNev);
11    if v_szemelyID != -1 then
12        --letezik az adatbazisban az adott szemely
13        if v_alapanyagID != -1 then
14            --ha letezett az alapanyag akkor megnezem hogy hany darab van az uzletben
15            select darab into v_alapanyagDarab from uzlet where alapanyag_id = v_alapanyagID;
16        if v_alapanyagDarab >= pDarab then
17
18

```

The debugger console at the bottom shows the following output:

```

Connecting to the database Foci Projekt.
Debugger attempting to connect to database.
Executing PL/SQL: DECLARE
  id VARCHAR2( 30 );
BEGIN
  id := DBMS_DEBUG.initialize( '192.168.0.106:1619991420523', 0 );
  DBMS_DEBUG.debug_on( TRUE );
END;
Debugger connected to database.
Source breakpoint: SZEMELYVASAROLALAPANYAG.pls:9

```

MI A TANULSÁG?

- Abban az esetben, ha én akarok debugolni egy bonyolultabb PL/SQL programot úgy, hogy belépjek bizonyos procedúrákba, függvényekbe a **Step Into** által, akkor **meg kell győződjek** először, hogy azok a **procedúrák, függvények le vannak-e kompilálva debugolásra vagy sem.**

Köszönöm szépen a
figyelmet! 😊