

## Obsah

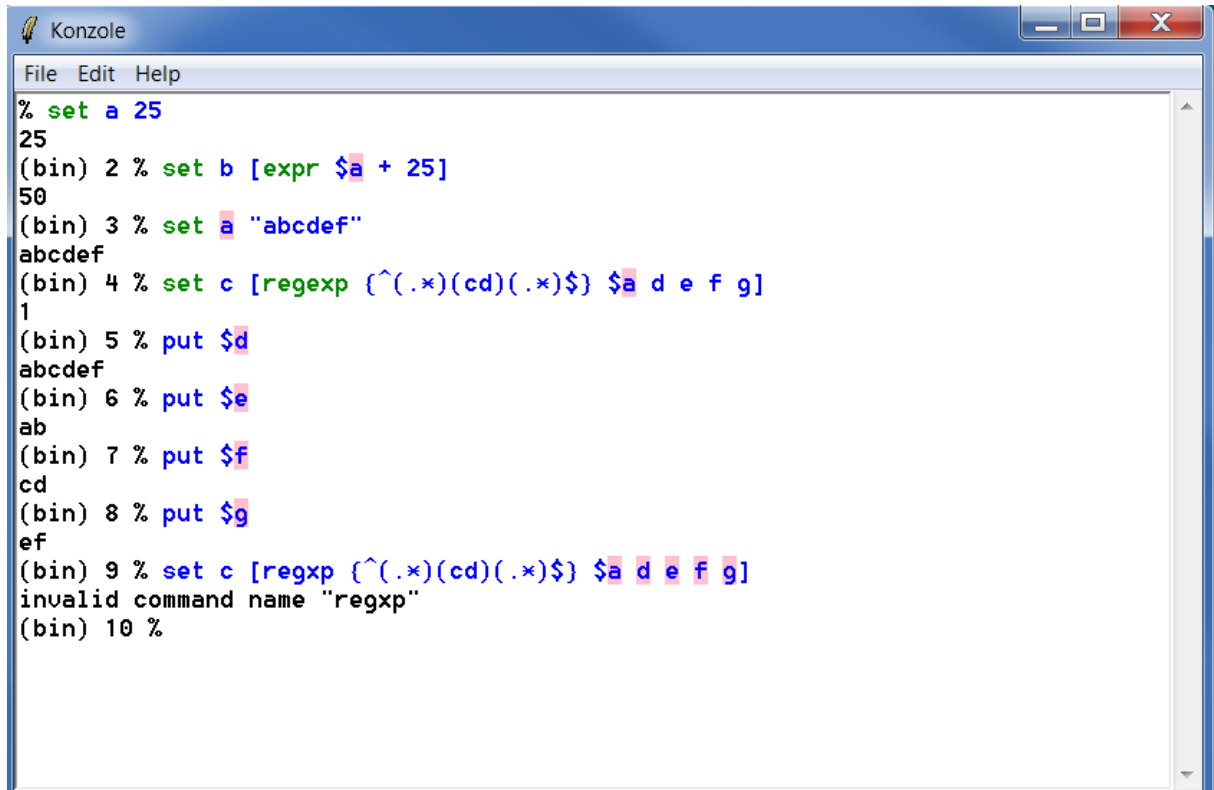
Vývojová prostředí - od pazourku k sofistikovaným nástrojům .....	2
WishTCL .....	2
Prostředí pro práci v databázích.....	3
Informix - dbaccess.....	3
Oracle SQL developer .....	4

## Vývojová prostředí - od pazourku k sofistikovaným nástrojům

Na následujících nástrojích si ukážeme postupných přechod od nejjednodušších nástrojů ovládaných pouze klávesnicí až k nástrojům uplatňujícím všechny vymoženosti grafického prostředí - ovládání myši, kontextové menu apod.

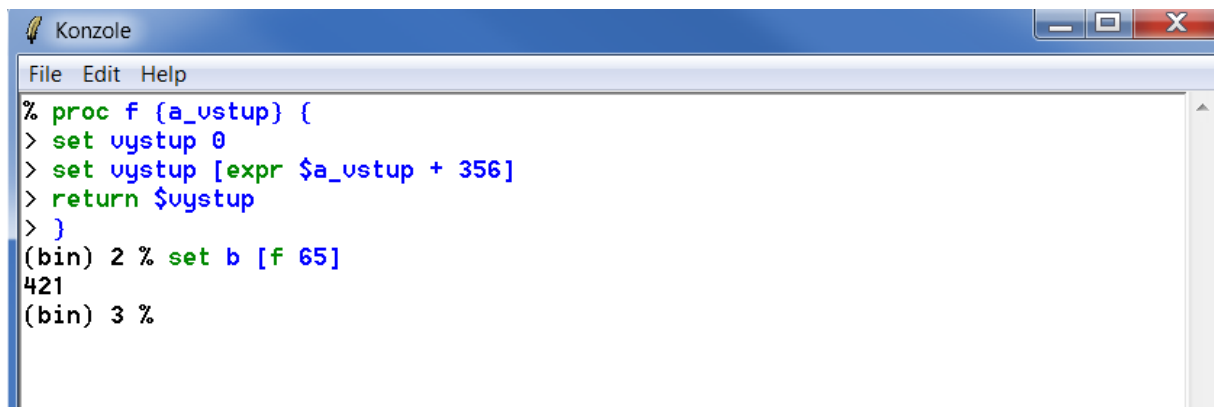
### WishTCL

Jednoduchá pracovní konzole skriptovací jazyka TCL. Bez prostředků pro kompilaci a ladění kódu. Je v ní prakticky možné pouze ověřit správnost syntaxe a vyhodnocení jednoduchých výrazů, příkazů a sekvencí příkazů a funkcí.



```
Konzole
File Edit Help
% set a 25
25
(bin) 2 % set b [expr $a + 25]
50
(bin) 3 % set a "abcdef"
abcdef
(bin) 4 % set c [regexp {^(.*) (cd) (.*)$} $a d e f g]
1
(bin) 5 % put $d
abcdef
(bin) 6 % put $e
ab
(bin) 7 % put $f
cd
(bin) 8 % put $g
ef
(bin) 9 % set c [regexp {^(.*) (cd) (.*)$} $a d e f g]
invalid command name "regexp"
(bin) 10 %
```

použití funkce



```
Konzole
File Edit Help
% proc f {a_vstup} {
> set vystup 0
> set vystup [expr $a_vstup + 356]
> return $vystup
> }
(bin) 2 % set b [f 65]
421
(bin) 3 %
```

## Prostředí pro práci v databázích

### Informix - dbaccess

Dostupné z příkazové řádky operačního systému UNIX, z prostředí MS Windows přes " černé" okno - např. putty. Ovládání přes klávesnici.

```
DBACCESS: █ Query-language Connection Database Table Session Exit
Use SQL query language.

----- Press CTRL-W for Help -----
```

Po výběru databáze je možno provádět konkrétní sql příkazy. Mimo interního editoru se dá použít i externí editor vi, Je nutné znát jeho příkazy. Seznam příkazů lze načíst i uložit do souboru. Dále je možno pracovat s tabulkami - zakládat nové, upravovat stávající nebo je dropnout. Současně je možné získat a upravit další informace jako indexy, přístupová práva, závislosti a omezení (constraints), triggeru, rozmístění tabulek a indexů ve fragmentech. V rámci editoru je možné přepnout na jinou konexi nebo server. Ovládání tohoto nástroje je možné pomocí šipek nebo horkými klávesami (velká písmena v jednotlivých příkazech).

```
NEW:  ESC    = Done editing      CTRL-A = Typeover/Insert      CTRL-R = Redraw
      CTRL-X = Delete character  CTRL-D = Delete rest of line

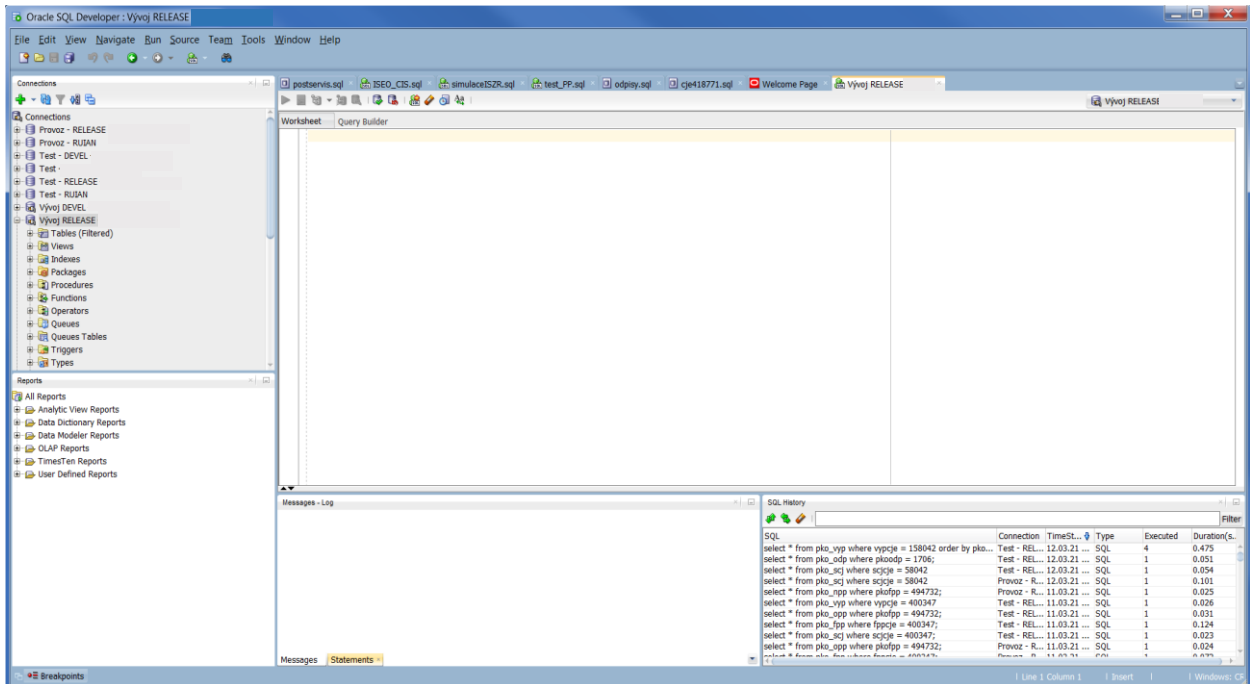
----- db@db_server --- Press CTRL-W for Help -----

select * from .... █
```

Při editaci sql příkazů je zobrazena krátká nápověda, příp. lze pomocí klávesové zkratky Ctrl+W vyvolat nápovědu kdekoli v dbaccessu.

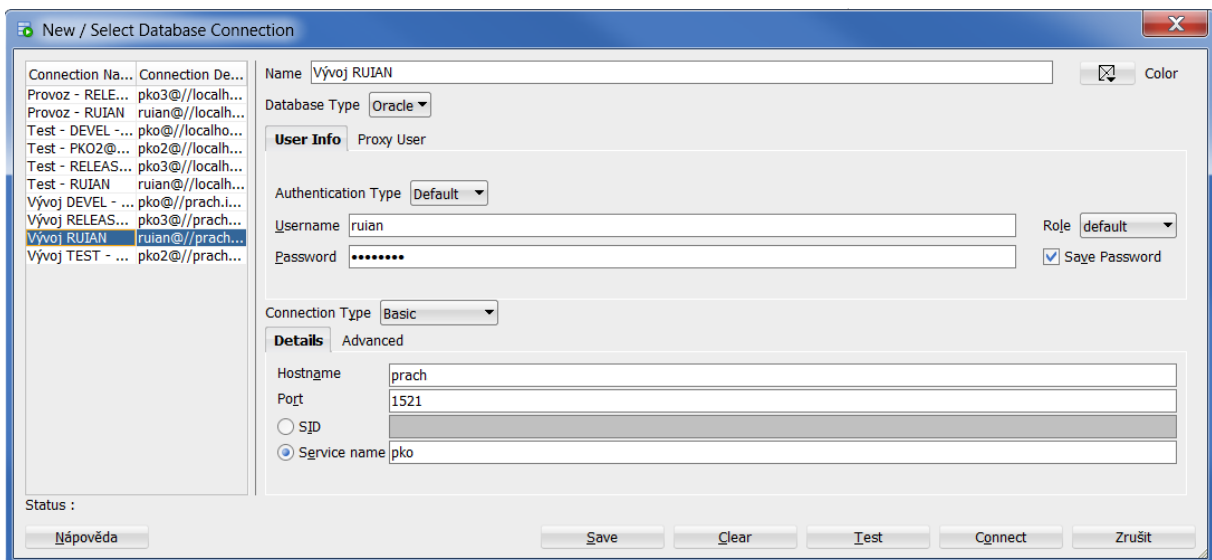
## Oracle SQL developer

Klasická MS Windows aplikace s klasickým ovládáním přes klávesnici či myš.

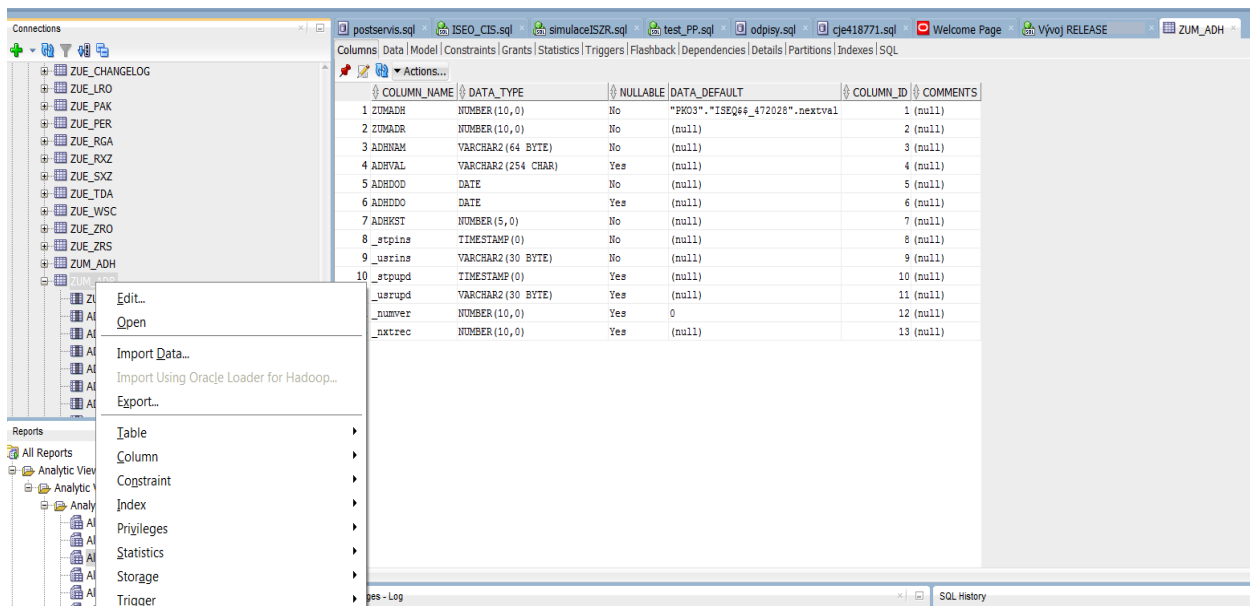


Aplikace je rozdělena na oblast pro připojení k jednotlivým databázím, nabídku předpřipravených reportů, vlastní pracovní plochu, oblast pro zobrazení zpráv a logů a oblast pro historii sql příkazů.

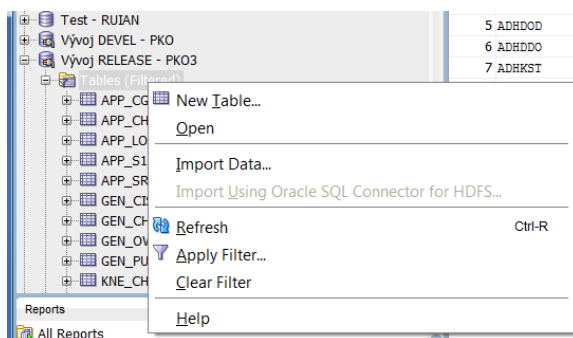
Jednotlivá připojení databází lze přidávat měnit nebo kopírovat.



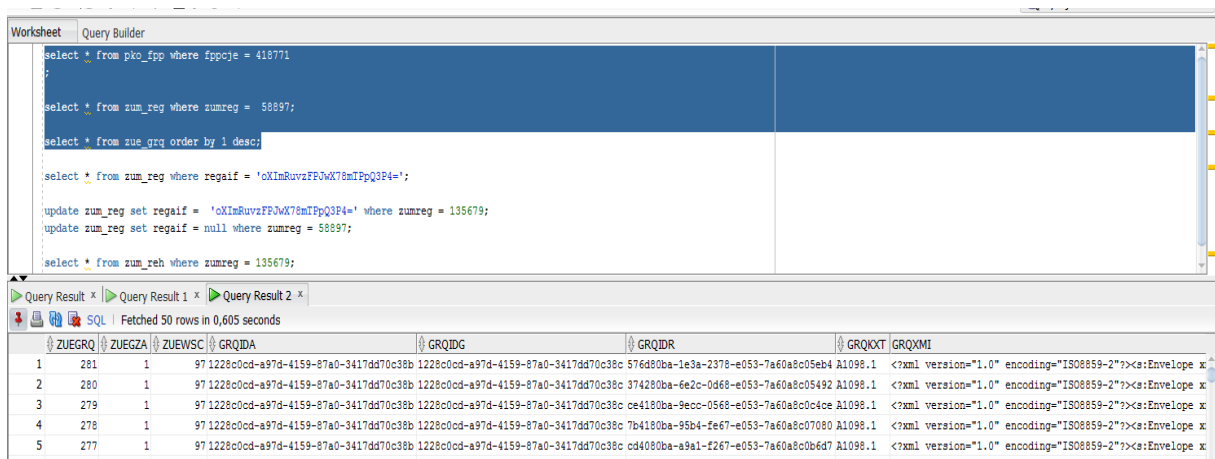
Pro jednotlivé připojené databáze lze vypsat seznam definovaných tabulek, zobrazení a indexů. Dále pak i definované procedury a funkce, operátory, triggerry, materializovaná zobrazení a jejich logy a další možnosti, které Oracle umožňuje. V detailu vybrané možnosti jsou pak umožněny další akce, např. u tabulky lze vypsat její jednotlivé sloupce, datový obsah, omezení, přístupová práva, statistiku, triggerry, závislosti, indexy a další. Definice nových položek či úprava stávajících je dostupná buď přes kontextové menu nebo přes volbu Actions.



Definice nové tabulky je dostupná z kontextového menu v rámci seznamu tabulek.



V rámci pracovní oblasti pak můžeme mít otevřeno více oken/záložek. Každé z těchto oken můžeme mít připojeno k jiné otevřené databázi. V jednotlivých oknech můžeme provádět jednotlivé sql příkazy nebo sekvence příkazů. Vždy je aplikován buď příkaz z aktuální řádky nebo vybraný blok příkazů. V případě volání sekvence příkazů je jejich výstup směřován do samostatné záložky.



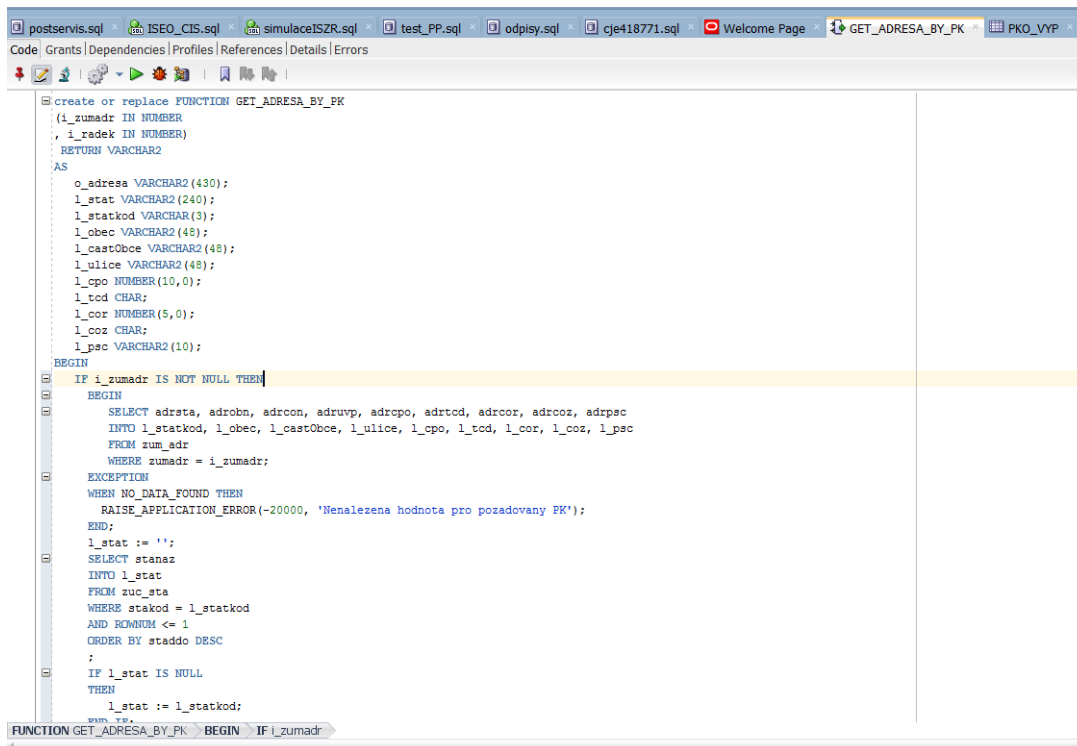
Při psaní jednotlivých sql příkazů nám aplikace nabízí možnosti odpovídající sql příkazům

```
select
SELECT
SELECT * FROM
SELECT * FROM p
SELECT 3 FROM DUAL
SELECT 3f FROM DUAL
...
SELECT * FROM table
```

nebo rovnou sloupce z tabulky, kterou v rámci příkazu prohledáváme.

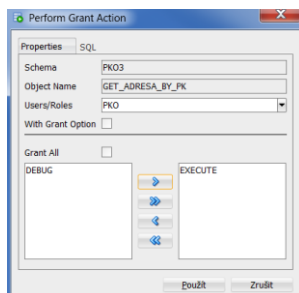
```
SELECT * FROM zum_adr where adrc|
adrobk
adrobn
adrokk
```

Oracle SQL developer umožňuje nejen pracovat s databází, jak jsme zvyklí. Umožní nám také programování v interním jazyce Oracle. V rámci prostředí pak máme prostředky pro kompilaci a ladění kódu.



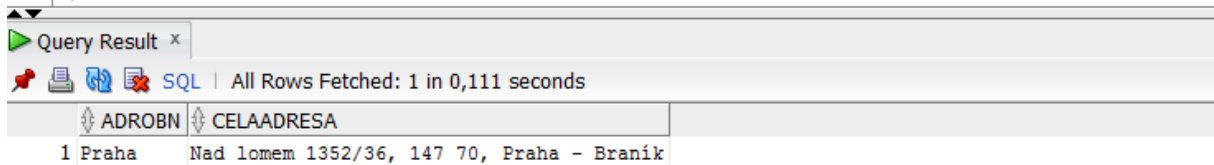
```
create or replace FUNCTION GET_ADRESA_BY_PK
(i_zumadr IN NUMBER
, i_radek IN NUMBER)
RETURN VARCHAR2
AS
o_adresa VARCHAR2(430);
l_stat VARCHAR2(240);
l_statkod VARCHAR(3);
l_obec VARCHAR2(48);
l_castObce VARCHAR2(48);
l_ulice VARCHAR2(48);
l_cpo NUMBER(10,0);
l_tcd CHAR;
l_cor NUMBER(5,0);
l_coz CHAR;
l_psc VARCHAR2(10);
BEGIN
IF i_zumadr IS NOT NULL THEN
BEGIN
SELECT adrsta, adrobn, adrcon, adrupv, adrcpo, adrtcd, adrcor, adrcoz, adrspc
INTO l_statkod, l_obec, l_castObce, l_ulice, l_cpo, l_tcd, l_cor, l_coz, l_psc
FROM zum_adr
WHERE zumadr = i_zumadr;
EXCEPTION
WHEN NO_DATA_FOUND THEN
RAISE_APPLICATION_ERROR(-20000, 'Nenalezena hodnota pro pozadovany FK');
END;
l_stat := '';
SELECT stanaz
INTO l_stat
FROM zuc_sta
WHERE stakod = l_statkod
AND ROWNUM <= 1
ORDER BY staddo DESC
;
IF l_stat IS NULL
THEN
l_stat := l_statkod;
END;
END;
```

Po odladění a kompilaci funkce je nejprve nutné přidělit práva jednotlivým uživatelům nebo rolím, které tuto funkci budou využívat.



Takto definovanou a zkompilevanou funkci pak můžeme použít v rámci sql příkazu

```
select a.adrobn, get_adresa_by_pk(zumadr,0) as celaAdresa from zum_adr a where zumadr = 1000;
```

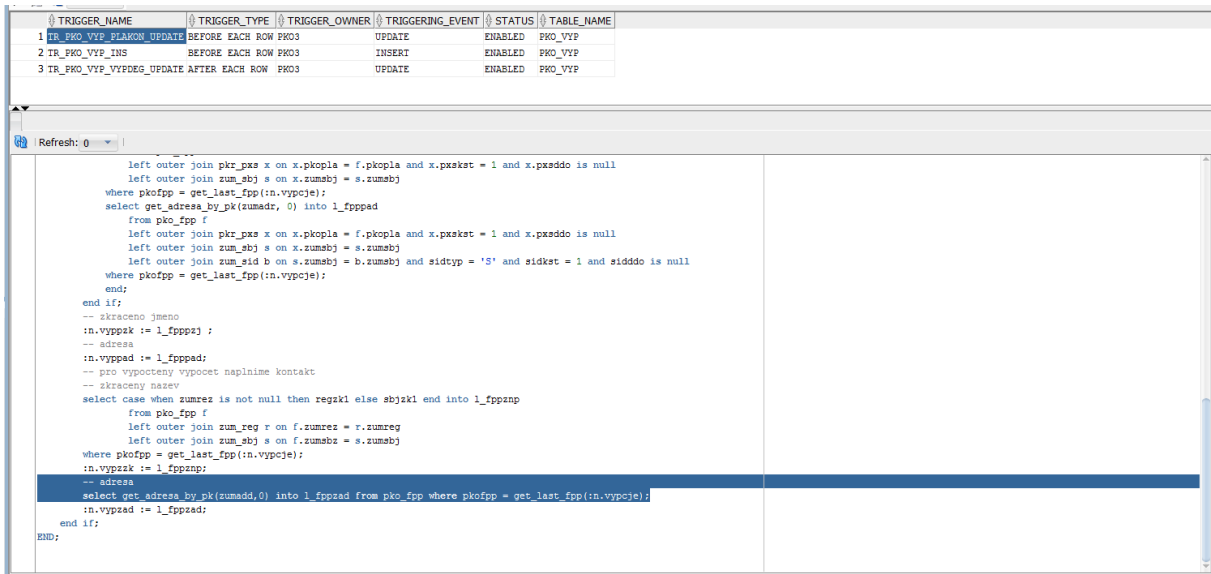


Query Result x

SQL | All Rows Fetched: 1 in 0,111 seconds

ADROBN	CELAADRESA
1 Praha	Nad lomem 1352/36, 147 70, Praha - Braník

Nebo v rámci některého triggeru na základě, kterého se nám doplní data tabulky při změně určitého sloupce.



TRIGGER_NAME	TRIGGER_TYPE	TRIGGER_OWNER	TRIGGERING_EVENT	STATUS	TABLE_NAME
1 TR_FKO_VYP_SIAKOV_UPDATE	BEFORE EACH ROW FKO3		UPDATE	ENABLED	FKO_VYP
2 TR_FKO_VYP_INS	BEFORE EACH ROW FKO3		INSERT	ENABLED	FKO_VYP
3 TR_FKO_VYP_VYPDEG_UPDATE	AFTER EACH ROW FKO3		UPDATE	ENABLED	FKO_VYP

```
Refresh: 0 |  
  
left outer join pkr_pxs x on x.pkopla = f.pkopla and x.pxsakt = 1 and x.pxsddo is null  
left outer join zum_sbj s on x.zumsbj = s.zumsbj  
where pkofpp = get_last_fpp(:n.vypcje);  
select get_adresa_by_pk(zumadr, 0) into l_fpppad  
from pko_fpp f  
left outer join pkr_pxs x on x.pkopla = f.pkopla and x.pxsakt = 1 and x.pxsddo is null  
left outer join zum_sbj s on x.zumsbj = s.zumsbj  
left outer join zum_sjd b on s.zumsbj = b.zumsbj and sidtyp = 'S' and sidkt = 1 and sidddo is null  
where pkofpp = get_last_fpp(:n.vypcje);  
end;  
end if;  
-- zkraceno jmeno  
:n.vyppzk := l_fpppz1 ;  
-- adresa  
:n.vyppad := l_fpppad;  
-- pro vypocteny vypocet naplnime kontakt  
-- zkraceny nazev  
select case when zumrez is not null then rezk1 else sbjz1 end into l_fppznp  
from pko_fpp f  
left outer join zum_reg r on f.zumrez = r.zumreg  
left outer join zum_sbj s on f.zumsbz = s.zumsbj  
where pkofpp = get_last_fpp(:n.vypcje);  
:n.vyppzk := l_fppznp;  
-- adresa  
select get_adresa_by_pk(zumadd,0) into l_fppzad from pko_fpp where pkofpp = get_last_fpp(:n.vypcje);  
:n.vyppzad := l_fppzad;  
end if;  
END;
```

Vlastní aplikace nabízí další sadu funkcí a nástrojů nejen pro definici a správu jednotlivých databází, ale také pro monitorování a správu provozu na jednotlivých databázích. Ale také nástroje pro týmovou práci jako je připojení verzovacího systému GIT a práci s jednotlivými verzemi databázového schématu.

