Jak dostat data do databáze

neb PostgresSQL technikálie

Agenda školení

1. Představení českého guru na PostgreSQL a kde jsou užitečné informace
2. Základy jazyka a čtení syntaxe SQL
3. Přehled vložení dat do databáze
4. Import dat pomocí SQL příkazu
5. Import dat pomocí PgAdmina
6. Import dat pomocí copy
7. Import dat pomocí pg_restore
8. Import dat pomocí dblinku
9. Import dat pomocí foreign tables

Lidi kteří ovlivnili naše dnešní chápaní software

Historické okénko

Zdroj: https://en.wikipedia.org/wiki/Alan_Turing

Alan Mathison Turing

(23.6.1923 – 7.6.1954)

- Počítačový vědec, matematik, logik, kryptoanalytik
- Jako první použil slova software
- Během II. světové války pracoval úspěšně na dekryptování nastavení německého kódovacího stroje Enigma (Kopií Enigmy sestavil Polský odboj, ale chybělo k němu nastavení.) pomocí elektromechanického počítače „Bombe“.
Důsledek byl ten, že se zkrátila doba války cc o 2 až 4 roky.

Na motivy jeho života a práce byl natočen známí film ‘Kód Enigmy’ (The Imitation game)

Zdroj: http://www.csfd.cz/film/283747-kod-enigmy/prehled/

1. Představení českého guru na PostgreSQL a kde jsou užitečné informace

Český guru v PostgreSQL

Pavel Stěhule

Pavel Stěhule je odborníkem na relační databázový systém PostgreSQL,
který v současnosti pracuje jako vývojář ve společnosti GoodData.

Zdroj: http://www.root.cz/autori/pavel-stehule/

Email: stehule@kix.fsv.cvut.cz

Kde najdete jeho názory :

- http://www.root.cz/autori/pavel-stehule/
- http://okbob.blogspot.cz/
- http://postgres.cz/wiki/Pavel_St%C4%9Bhule
- https://groups.google.com/forum/?hl=cs#!forum/postgresql-cz

2. Základy jazyka a čtení syntaxe SQL

Zápis SQL syntaxe

Abychom mohli používat SQL jazyk musíme poznat metajazyk (jazyk ve kterém je zapsaná syntax příkazů) a ve kterém jsou příkazy (commands) zapsané.

Zdroj : http://www.postgresql.org/docs/9.5/static/sql-syntax-lexical.html

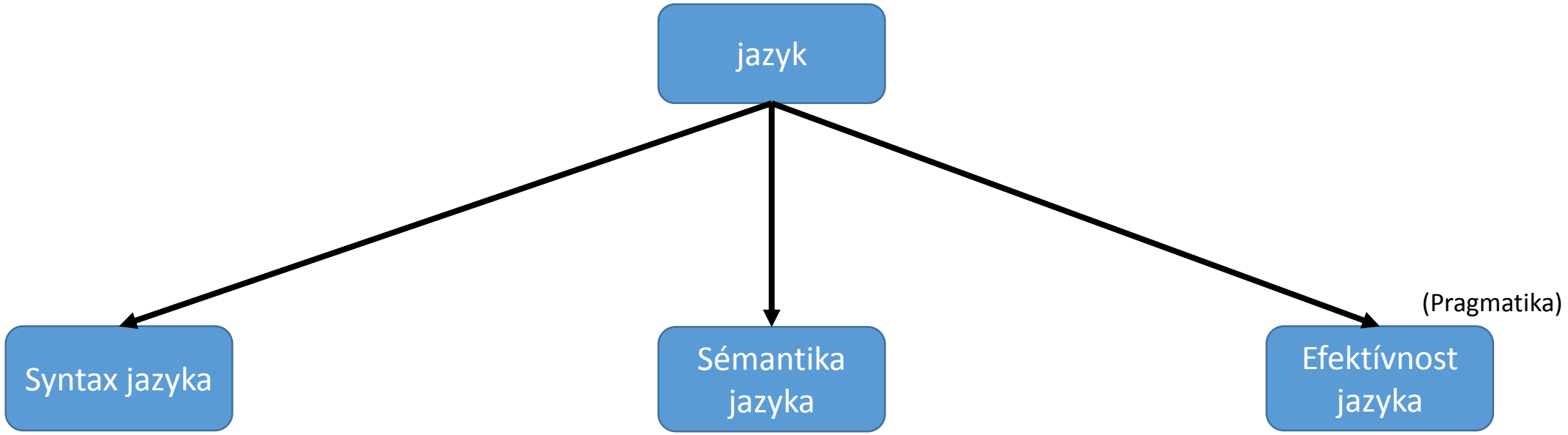
Příklad zápisu příkladu v metajazyku

Synopsis

Zdroj : http://www.postgresql.org/docs/9.1/static/sql-select.html

```
[ WITH [ RECURSIVE ] with_query [, ...] ]
SELECT [ ALL | DISTINCT [ ON ( expression [, ...] ) ] ]
    * | expression [ [ AS ] output_name ] [, ...]
    [ FROM from_item [, ...] ]
    [ WHERE condition ]
    [ GROUP BY expression [, ...] ]
    [ HAVING condition [, ...] ]
    [ WINDOW window_name AS ( window_definition ) [, ...] ]
    [ { UNION | INTERSECT | EXCEPT } [ ALL | DISTINCT ] select ]
    [ ORDER BY expression [ ASC | DESC | USING operator ] [ NULLS { FIRST | LAST } ] [, ...] ]
    [ LIMIT { count | ALL } ]
    [ OFFSET start [ ROW | ROWS ] ]
    [ FETCH { FIRST | NEXT } [ count ] { ROW | ROWS } ONLY ]
    [ FOR { UPDATE | SHARE } [ OF table_name [, ...] ] [ NOWAIT ] [...] ]

where from_item can be one of:

    [ ONLY ] table_name [ * ] [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
    ( select ) [ AS ] alias [ ( column_alias [, ...] ) ]
    with_query_name [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
    function_name ( [ argument [, ...] ] ) [ AS ] alias [ ( column_alias [, ...] | column_definition [, ...] ) ]
    function_name ( [ argument [, ...] ] ) AS ( column_definition [, ...] )
    from_item [ NATURAL ] join_type from_item [ ON join_condition | USING ( join_column [, ...] ) ]

and with_query is:

    with_query_name [ ( column_name [, ...] ) ] AS ( select | insert | update | delete )

TABLE [ ONLY ] table_name [ * ]
```

Složky Jazyka



INSERT INTO table_name [AS alias]
[(column_name [, ...])]
{ DEFAULT VALUES | VALUES
({ expression | DEFAULT } [, ...]) [, ...] | query }

INSERT INTO users
(first_name,last_name)
VALUES
('Rudolf','Carnap');

INSERT INTO users
(first_name,last_name)
VALUES
('Rudolf','Carnap'),
('Charles',' Morris');

Syntax SQL jazyka

Identifkátory identifikují databázové objekty v závislosti na použití. Nestadardní názvy se dávají do složených úvozovek. (“)

Pro zápis syntaxe příkazů se používá **Backusova-Naurova forma (BNF)**. Syntax SQL je **bezkontextová gramatika**.

Konstanty jsou hodnoty, které vstupují do příkazů. Jsou číselné, binární nebo řetězcové. Řetězcové se uzavírají do jednoduchých úvozovek (‘).

Opakování (…) předchozí token se může libovolně krát opakovat

Množina token ({}) Znamená že jeden token s množiny může být použit. Na oddělení se používá (|)

```
[ WITH [ RECURSIVE ] with_query [, ...] ]
INSERT INTO table_name [ AS alias ] [ ( column_name [, ...] ) ]
    { DEFAULT VALUES | VALUES ( { expression | DEFAULT }
[, ...] ) [, ...] | query }
    [ ON CONFLICT [ conflict_target ] conflict_action ]
[       RETURNING * | output_expression [ [ AS ] output_name ]
[, ...] ]
```

Alternatíva (|) používá se v množinách ({}) a odděľuje jednotlivé alternativy

Klíčová slova mají v SQL fixní význam v příkazu.

Token – slova jazyka jsou identifikátory, klíčová slova, konstanty

Volitelné formy ([]) nemusí být zahrnuty do příkazu

4. Přehled postupů vložení dat do databáze

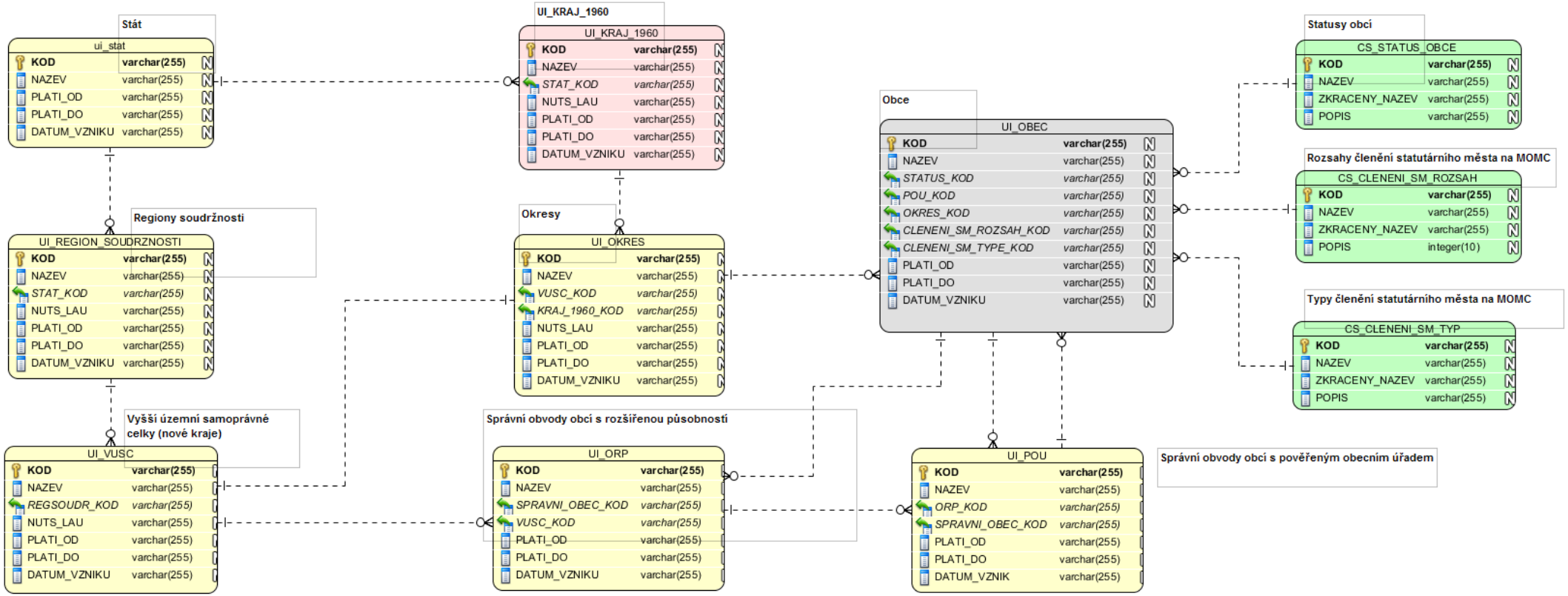
Způsoby vložení dat do databáze

1. SQL příkazy
2. Import přes pgAdmin
3. Příkazem copy
4. Příkazem pg_restore
5. Použitím dblinku
6. Použitím foriegn table

Jaké data budeme importovat ?

Jedná se o veřejně poskytovaná data ve formátu pro import takže struktura nedopovídá normám návrhu datových modelů

Uzemní celky ČR dle stavu z 13.5.2016



4. Import dat pomocí SQL příkazu

Import dat pomocí SQL příkazu

```
[ WITH [ RECURSIVE ] with_query [, ...] ]
INSERT INTO table_name [ AS alias ] [ ( column_name [, ...] ) ]
        { DEFAULT VALUES | VALUES ( { expression | DEFAULT } [, ...] ) [, ...] | query }
        [ ON CONFLICT [ conflict_target ] conflict_action ]
[       RETURNING * | output_expression [ [ AS ] output_name ] [, ...] ]
```

Budeme zde používat tuto formu příkazu Insert into …

```
INSERT INTO table_name [(column_name [, …])]
        VALUES ( expression [, …]) [, …]
        [RETURNING {* | output_expression} [, …] ]
```

Volby, které nebudeme v současnosti potřebovat jsou přeškrtnuté

Jednoduché vložení dat

```
1  -- ================================
2  -- 01 - Jednoduché vložení dat
3  -- ================================
4
5  -- Vytvoříme tabulku statů
6  create table ui_stat
7  (kod integer primary key
8  ,nazev text unique
9  ,plati_od date
10 ,plati_do date
11 ,datum_vzniku date);
12
13 -- Relizujeme insert
14 insert into ui_stat
15 (kod,nazev,plati_od,plati_do,datum_vzniku)
16 values
17 (1,'Česká republika',to_date('6.6.2015','DD.MM.YYYY'),null,to_date('1.1.1993','DD.MM.YYYY'))
18 returning *;
19
20
```

Vytvoříme tabulku

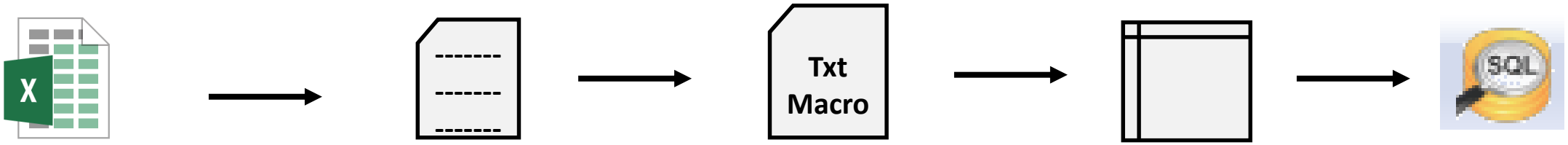
Vložíme data

Výsledek kontrolního výpisu

Datový výstup | Výklad | Zprávy | Historie

	kod integer	nazev text	plati_od date	plati_do date	datum_vzniku date
1	1	Česká republika	2015-06-06		1993-01-01

Import dat pomocí SQL příkazu insert



Zdroj dat
Excelovský soubor

CiselnikPor_Okres.xls

Zdroj: Veřejně přístupná data ČSU

Textový,nebo
CSV soubor

por_okres.csv

Úprava dat
vhodnost pro
insert

v pgSQL spuštění
Insert

(3508,'Most','','CZ0425','Jihočeský kraj'),
(3507,'Louny','','CZ0424','Jihočeský kraj'),
(3506,'Litoměřice','','CZ0423','Jihočeský kraj'),
(3505,'Liberec','','CZ0513','Jihočeský kraj'),
(3504,'Jablonec nad Nisou','','CZ0512','Jihočeský kraj'),
(3503,'Chomutov','','CZ0422','Jihočeský kraj'),

```
create table por_okres
(okres integer primary key,
nazev text,
zkratka text,
nuts4 text,
kraj text);
```

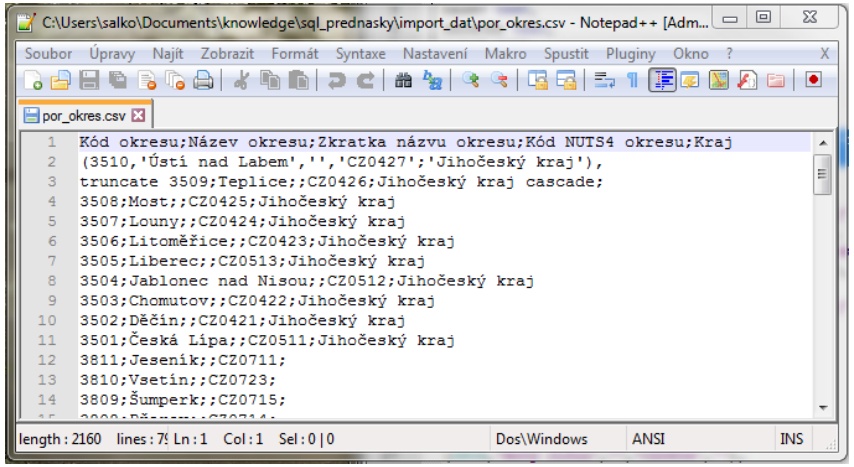
Úprava dat do tvaru vhodného pro import

z tvaru

3510;Ústí nad Labem;;CZ0427;Jihočeský kraj
3509;Teplice;;CZ0426;Jihočeský kraj
3508;Most;;CZ0425;Jihočeský kraj
3507;Louny;;CZ0424;Jihočeský kraj

na tvar

(3510,'Ústí nad Labem','','CZ0427','Jihočeský kraj'),
(3509,'Teplice','','CZ0426','Jihočeský kraj'),
(3508,'Most','','CZ0425','Jihočeský kraj'),
(3507,'Louny','','CZ0424','Jihočeský kraj'),



- Přes Makro – Začít nahrávaní - nahrát makro pro úpravu jednoho řádku.
- Přes Makro – Ukončit nahrávaní – uložit makro
- Přes Makro – Spustit makro vícekrát – Až do konce souboru

Doporučuji free editor Notepad ++
https://notepad-plus-plus.org/download/v6.9.1.html

Jak připravit řádkové makro

Doporučuji free editor

3508;Most;;CZ0425;Jihočeský kraj

1. Vložit (

2. Vyhledat ;
A nahradit jej ,’

3. Vyhledat ;
A nahradit jej ‘,’

4. Vyhledat ;
A nahradit jej ‘,’

5. Vyhledat ;
A nahradit jej ‘,’

6. ‘End’ klávesa
Na konec řádky.
Přidat ‘),

7. ‘Home’ klávesa a
pak šípka dolu

(3508,’Most’,’’,’CZ0425’,’Jihočeský kraj’),

Příklad složeného insertu

```
7
8   -- Vytvoříme tabuku
9   create table ui_region_soudrznosti
10  (kod integer primary key,
11   nazev text not null unique,
12   stat_kod integer not null,
13   nuts_lau char(4) not null unique,
14   plati_od date not null,
15   plati_do date,
16   datum_vzniku date not null);
17
18   -- Zadáme příkaz insert
19   insert into ui_region_soudrznosti
20   (kod,nazev,stat_kod,nuts_lau,plati_od,datum_vzniku)
21   values
22  (60,'Jihovýchod',1,'CZ06',to_date('1.4.2016 0:00','DD.MM.YYYY'),to_date('1.1.2001 0:00','DD.MM.YYYY')),
23  (35,'Jihozápad',1,'CZ03',to_date('24.3.2016 0:00','DD.MM.YYYY'),to_date('1.1.2001
24  (86,'Moravskoslezsko',1,'CZ08',to_date('31.3.2016 0:00','DD.MM.YYYY'),to_date('1.1
25  (19,'Praha',1,'CZ01',to_date('31.3.2016 0:00','DD.MM.YYYY'),to_date('1.1.2001 0:0
26  (51,'Severovýchod',1,'CZ05',to_date('22.3.2016 0:00','DD.MM.YYYY'),to_date('1.1.2
27  (43,'Severozápad',1,'CZ04',to_date('29.3.2016 0:00','DD.MM.YYYY'),to_date('1.1.200
28  (78,'Střední Morava',1,'CZ07',to_date('22.3.2016 0:00','DD.MM.YYY'),to_date('1.1.2
29  (27,'Střední Čechy',1,'CZ02',te('1.4.2016 0:00','DD.MM.YYYY'),to_date('1.1.20
30  returning *;
31
32
```

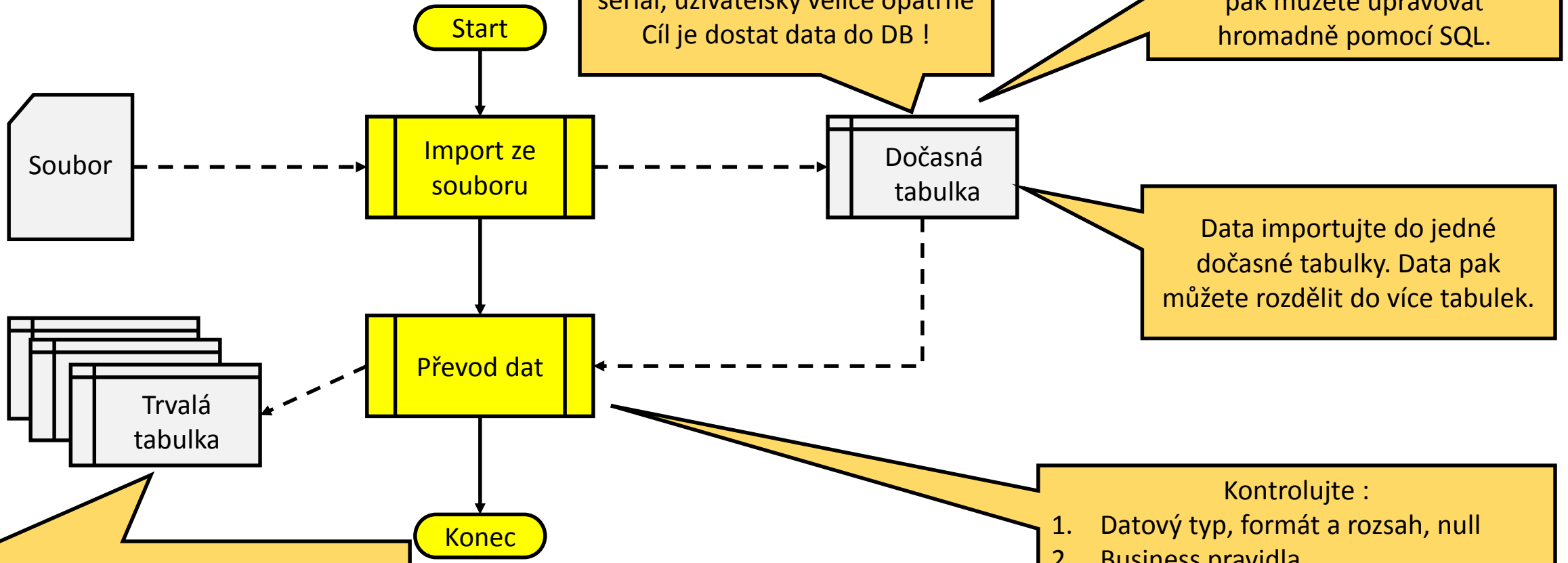
Použita funkce po převod do date
to_date(<text>,<format>)

Výsledek

Datový výstup | Výklad | Zprávy | Historie

	kod integer	nazev text	stat_kod integer	nuts_lau character(4)	plati_od date	plati_do date	datum_vzniku date
1	60	Jihovýchod	1	CZ06	2016-04-01		2001-01-01
2	35	Jihozápad	1	CZ03	2016-03-24		2001-01-01
3	86	Moravskoslezsko	1	CZ08	2016-03-31		2001-01-01
4	19	Praha	1	CZ01	2016-03-31		2001-01-01
5	51	Severovýchod	1	CZ05	2016-03-22		2001-01-01
6	43	Severozápad	1	CZ04	2016-03-29		2001-01-01
7	78	Střední Morava	1	CZ07	2016-03-22		2001-01-01
8	27	Střední Čechy	1	CZ02	2016-04-01		2001-01-01

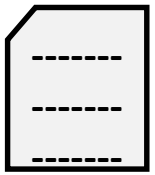
Doporučný způsob importu

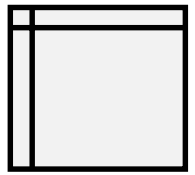


5. Import dat pomocí pgAdmina

Import pomocí pgAdmina

Příklad 03





Zdroj dat
Excelovský soubor

Textový,nebo
CSV soubor

v pgSQL
vytvoření tabullky

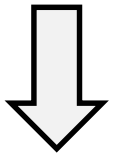
v pgAdminovi rutina
pro import dat

CiselnikPor_Kraj.xls

Zdroj: Veřejně přístupná data ČSU

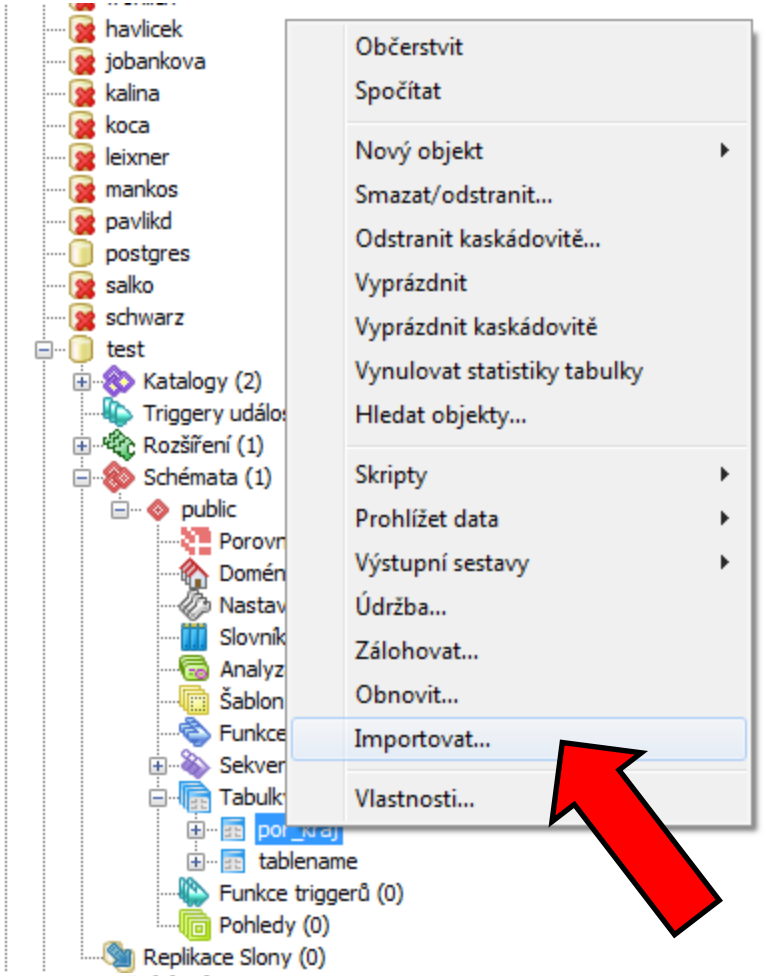
por_kraj.csv

```
create table por_kraj
(kod integer primary key,
 nazev text,
 zkratka text,
 nuts3 text,
 oblast text);
```

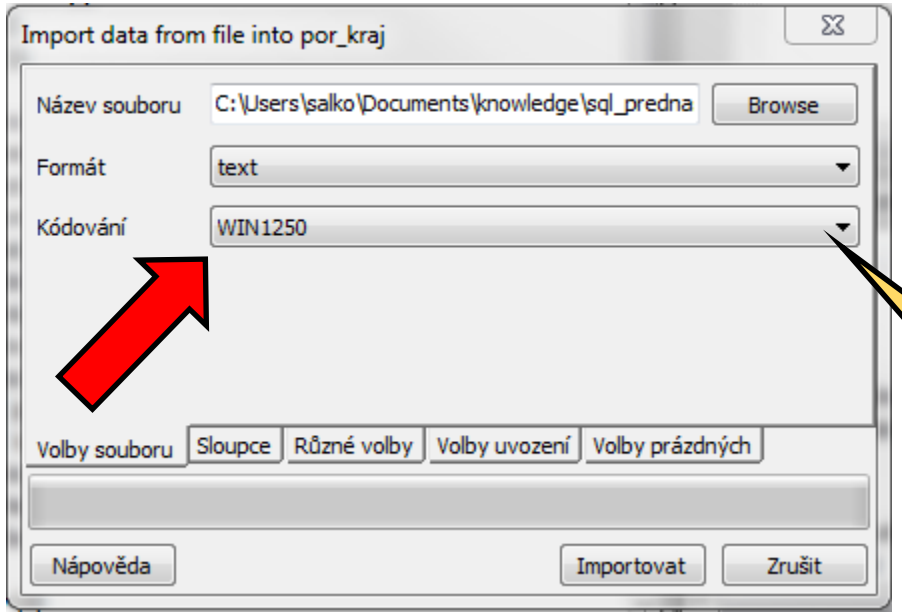


Import dat pomocí pgAdmina

pgAdmin importuje data přes příkaz copy ze standarního vstupního zařízení (volba STDIN).

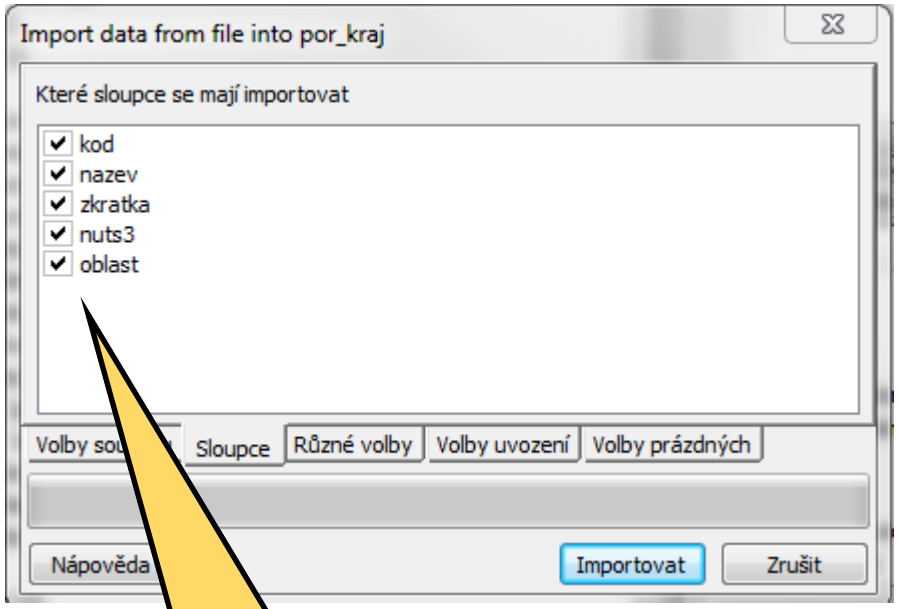


Dialogové okno
Import data from file into …, záložka Volby souboru

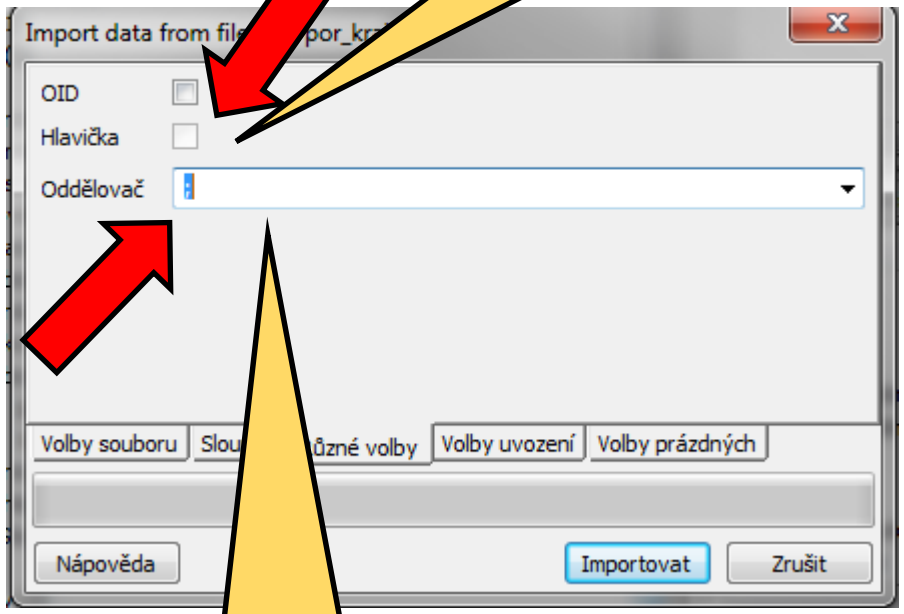


Pozor db je defaultně v kódové stránce UTF-8
Zdroj dat v WIN1250

Import dat pomocí pgAdmina



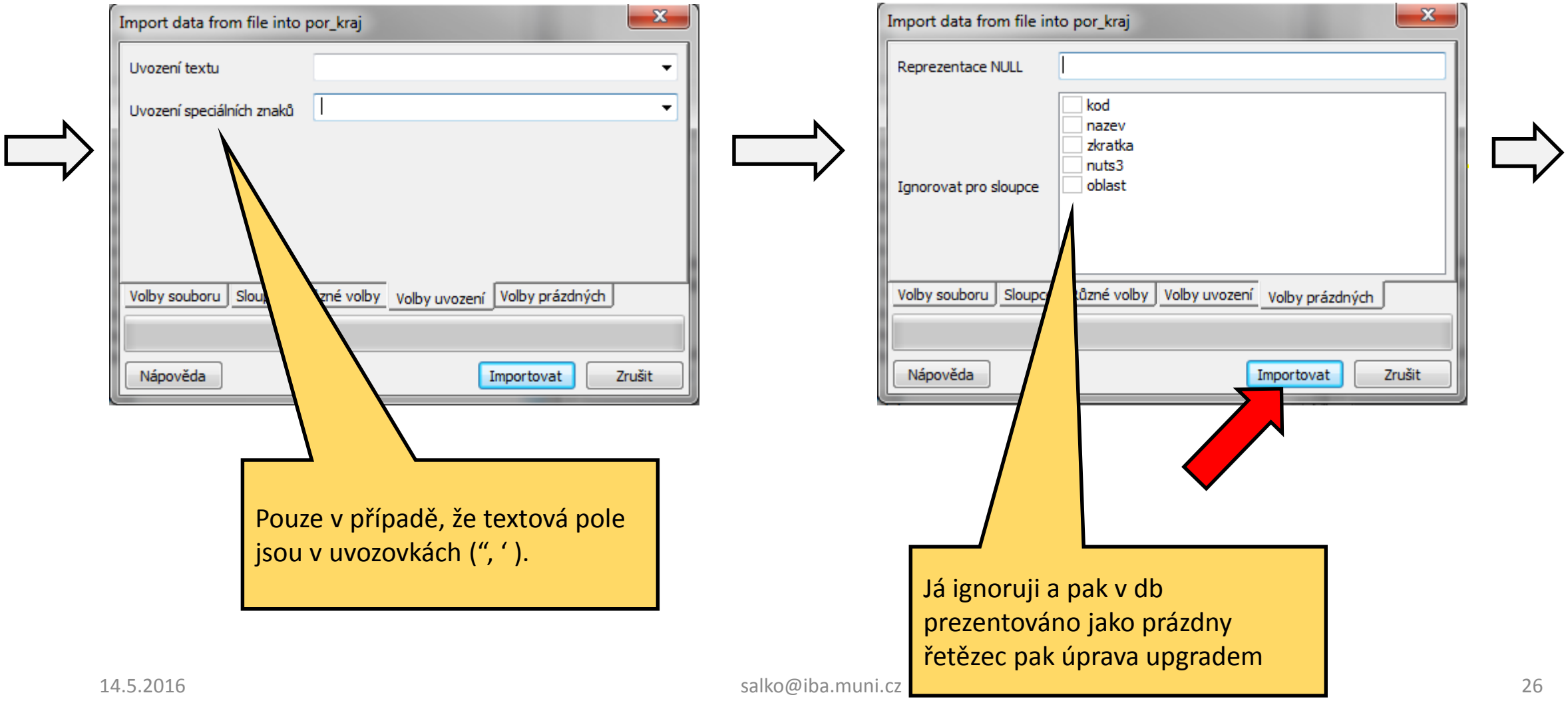
Pozor !
Záleží na pořadí sloupců. Když nesouhlasí volte příkaz COPY.



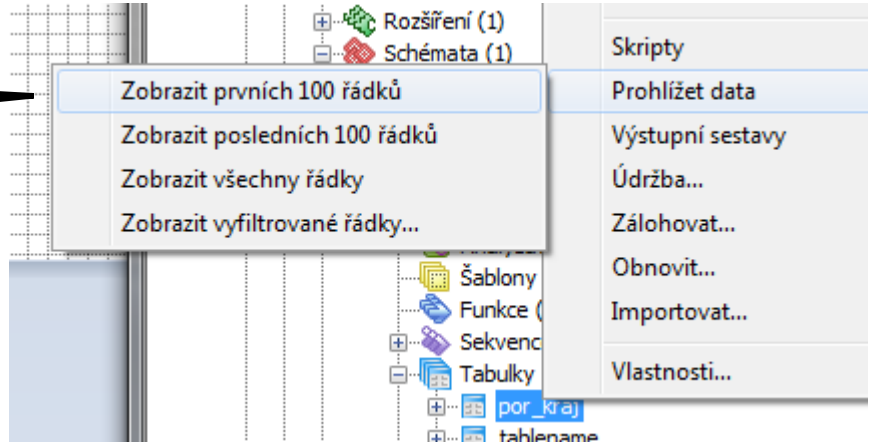
Jestli je csv soubor obsahuje hlavičku (názvy sloupců) tak zaškrtnout.

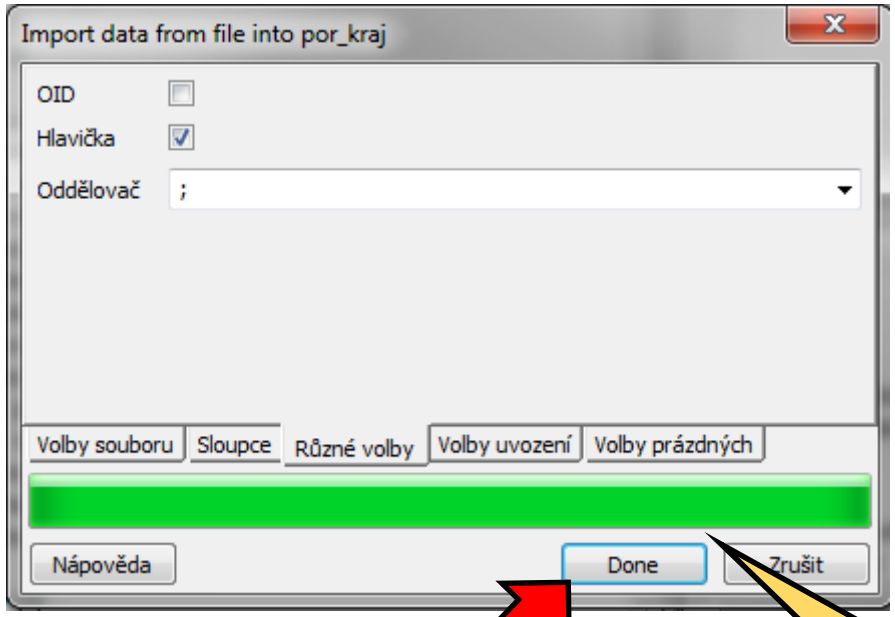
Odělovač sloupců
Pro csv soubor obvykle „;“ (středník)

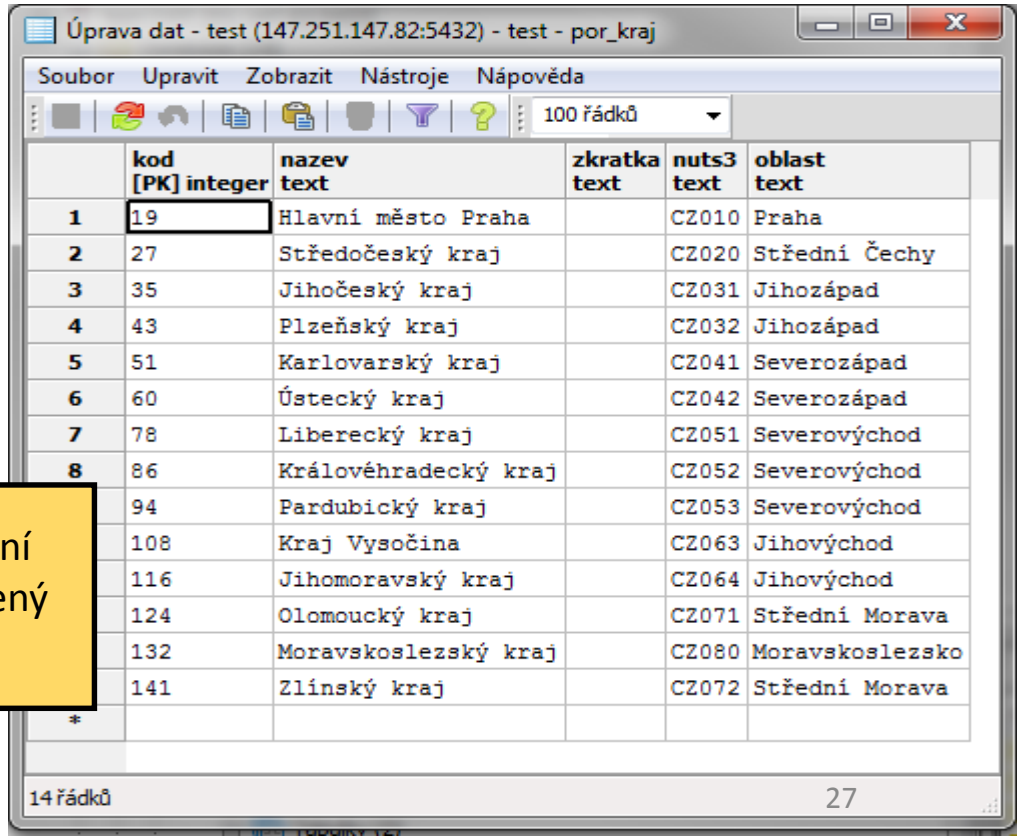
Import dat pomocí pgAdmina



Po importu možno ověřit zobrazením části dat.







	kod [PK] integer	nazev text	zkratka text	nuts3 text	oblast text
1	19	Hlavní město Praha		CZ010	Praha
2	27	Středočeský kraj		CZ020	Střední Čechy
3	35	Jihočeský kraj		CZ031	Jihozápad
4	43	Plzeňský kraj		CZ032	Jihozápad
5	51	Karlovarský kraj		CZ041	Severozápad
6	60	Ústecký kraj		CZ042	Severozápad
7	78	Liberecký kraj		CZ051	Severovýchod
8	86	Královéhradecký kraj		CZ052	Severovýchod
	94	Pardubický kraj		CZ053	Severovýchod
	108	Kraj Vysočina		CZ063	Jihovýchod
	116	Jihomoravský kraj		CZ064	Jihovýchod
	124	Olomoucký kraj		CZ071	Střední Morava
	132	Moravskoslezský kraj		CZ080	Moravskoslezsko
	141	Zlínský kraj		CZ072	Střední Morava
*					

Po úspěšném importu se změní tlačítko Import na Done a zelený pruh.

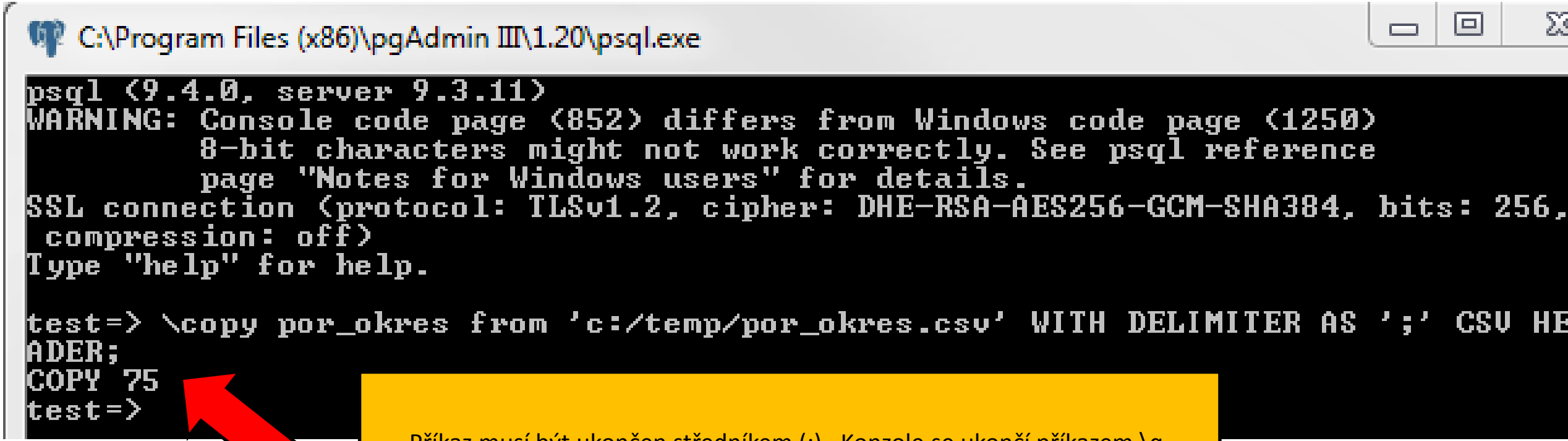
8. Import dat pomocí COPY

Import dat pomocí copy

Příklad 04

Copy lze použít ve windows z psql konzoly. Musí se použít příkaz **\copy.** Nejdříve je nutno vymazat data z tabulky por_okres příkazem **delete from por_okres;**

\copy por_okres from 'c:/temp/por_okres.csv' WITH DELIMITER AS ';' CSV HEADER



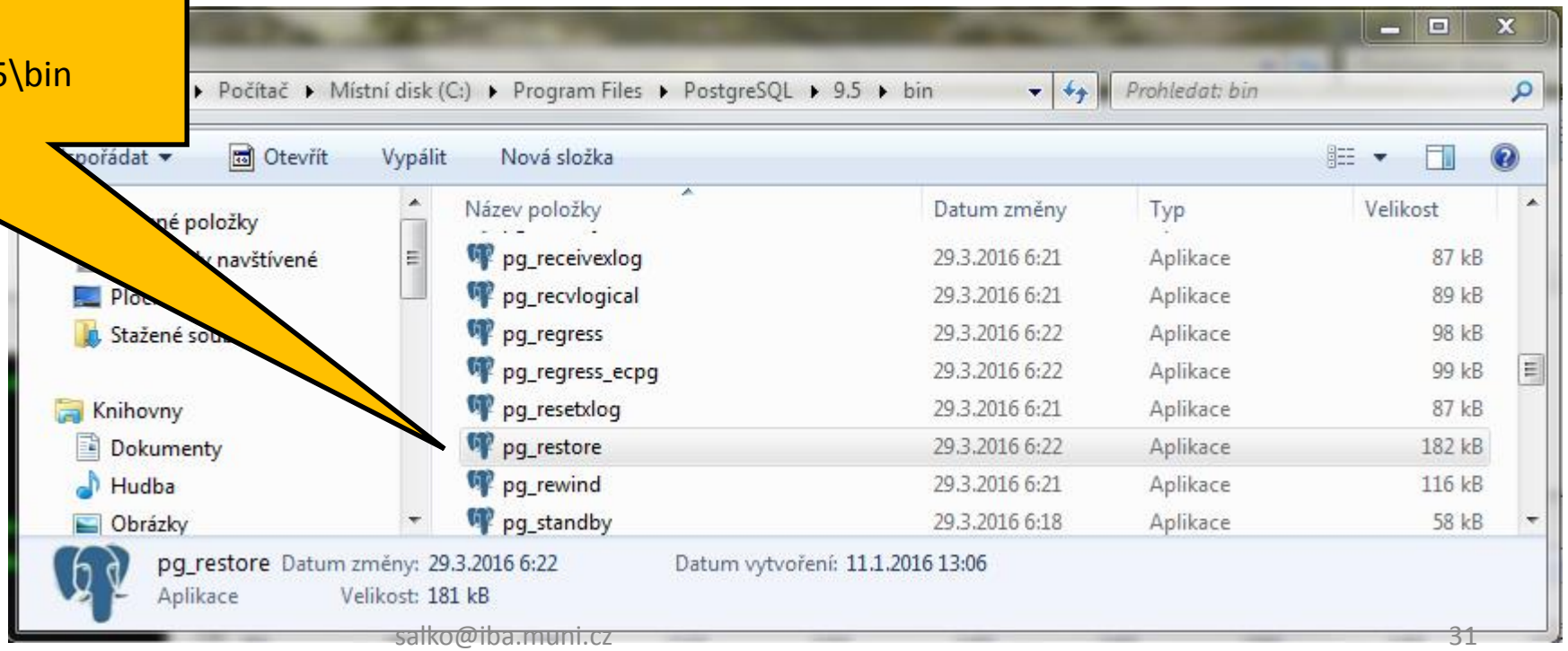
Příkaz musí být ukončen středníkem (;). Konzole se ukončí příkazem \q.

7. Import dat pomocí pg_restore

pg_restore utilita

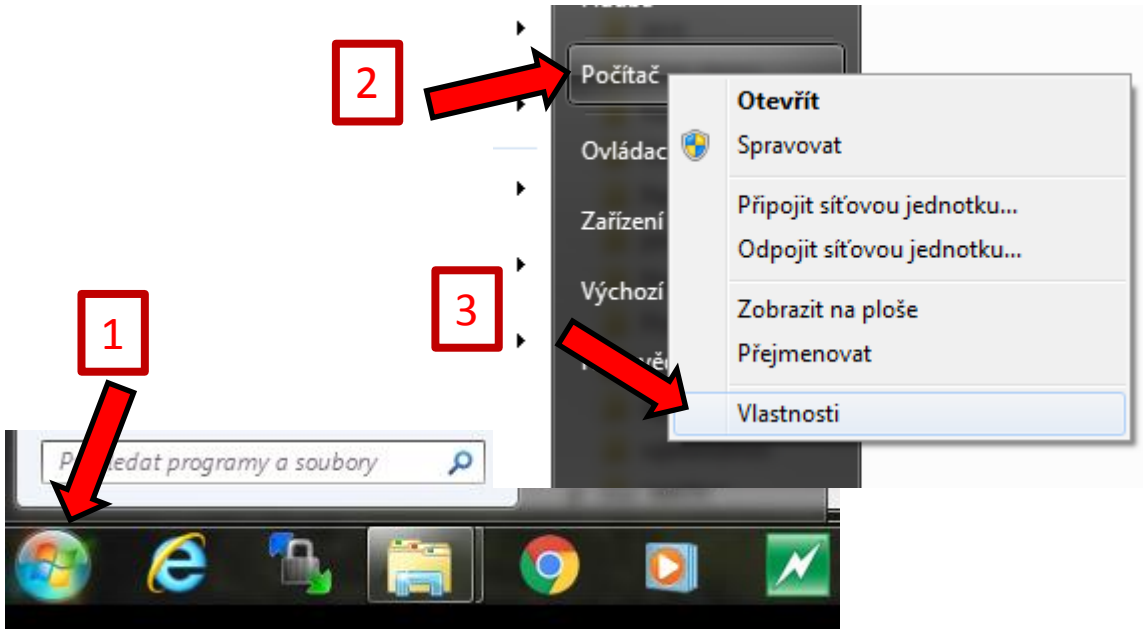
pg_restore je utilita, pomoci které importujeme data a datové a řídící struktury a funkce do databáze ve formátu jaký poskytuje exportní utilita pg_dump.

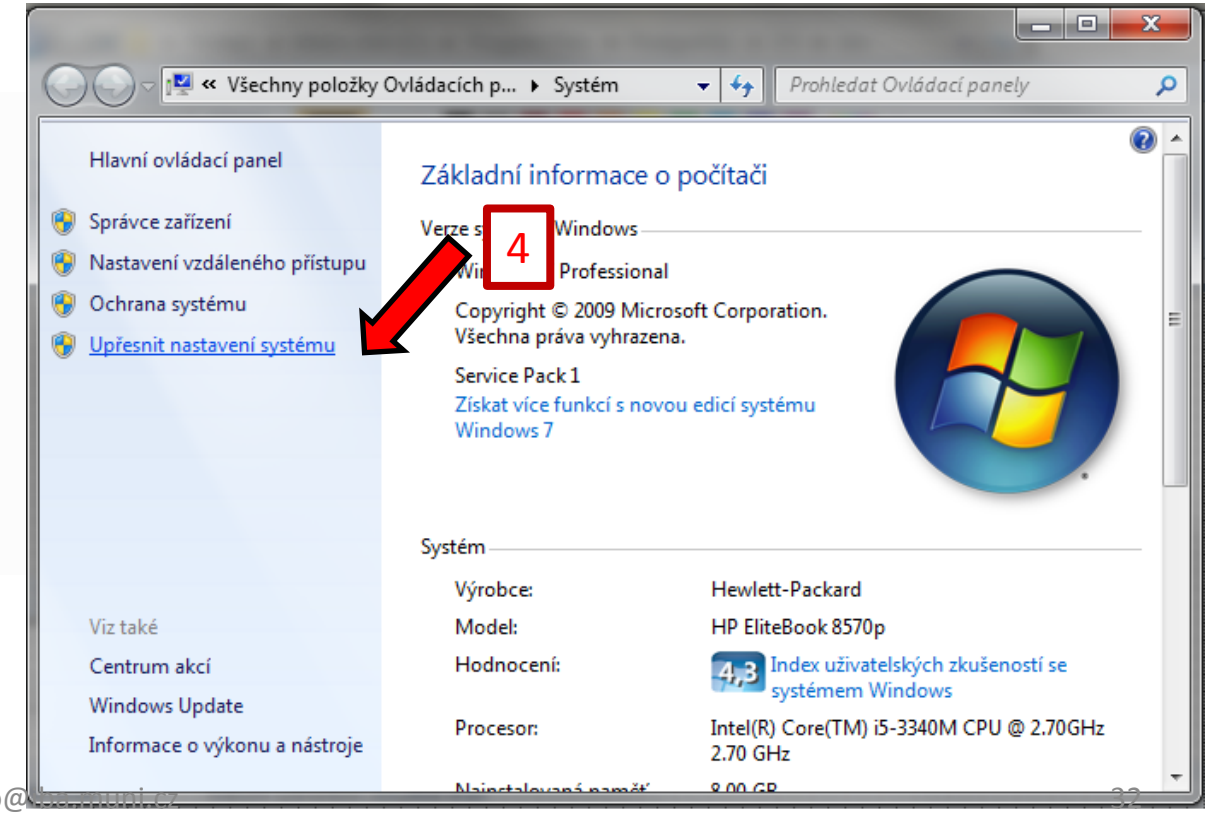
Utilitu je možno spouštět z příkazového řádku.
Je umístěna v
C:\Program Files\PostgreSQL\9.5\bin



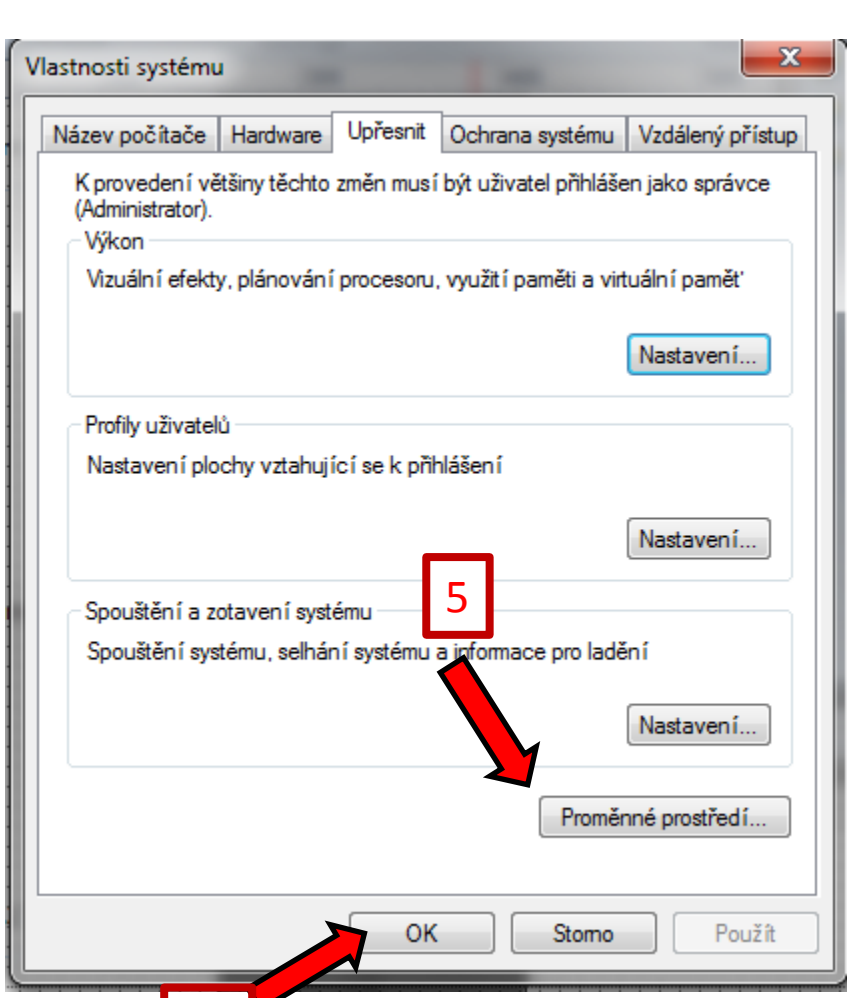
Nastavení cesty do systemové proměnné path 1.

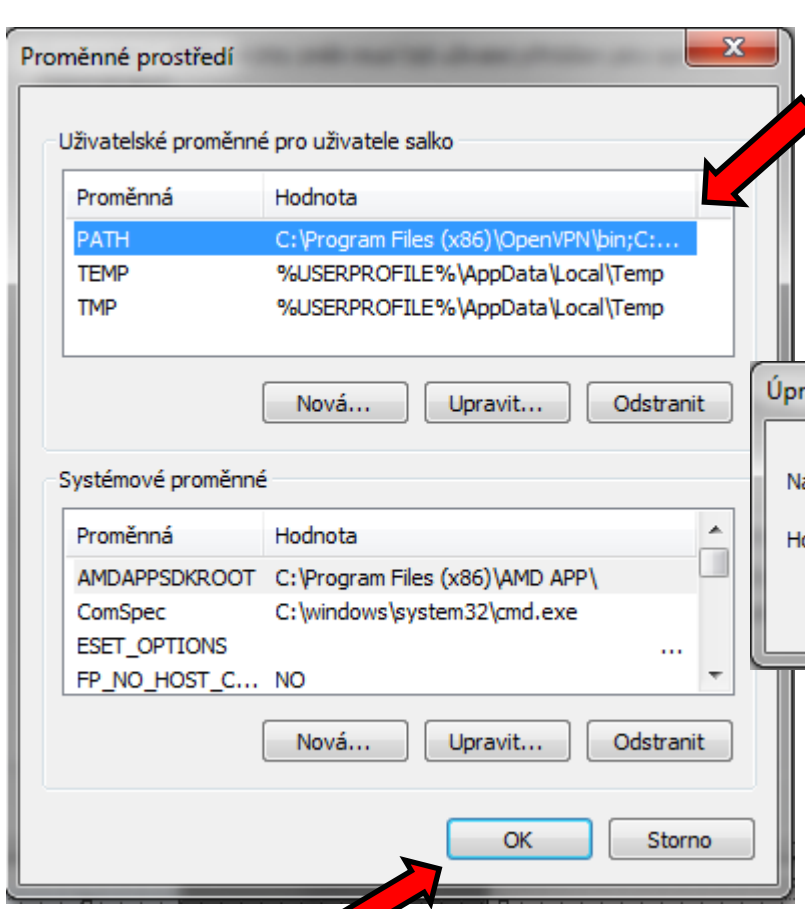
Aby Jste mohli pg_restore spouštět z libovolného adresáře na počítači, musíte cestu k pg_restore nastavit do systemové proměnné path.

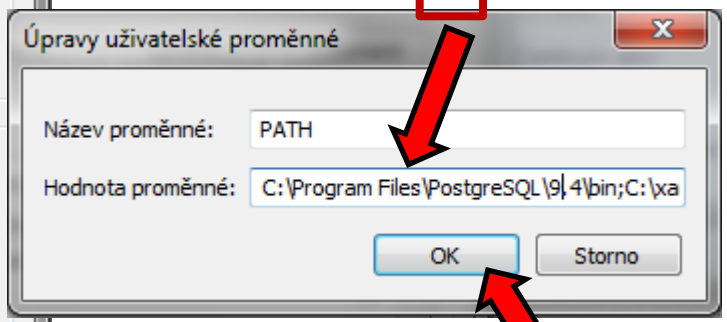




Nastavení cesty do systemové proměnné path 2.







5

6

7

8

9

10

Musí být uvedena v hodnotě. Předchozí zachovat !

C:\Program Files (x86)\OpenVPN\bin;C:\Program Files\PostgreSQL\9.4\bin;C:\xampp\php;

Kam si ukládá postgre_sql hesla

Hesla jsou uložena v souboru c:\Users\<uzivatel>\AppData\Romaing\postgresql\pgpass.conf .
Přístup je tam jako administrátor !
Nejsou vidět, doporučuji zpřístupnit přes Total Commander.

Pozor na ochranu hesel před zneužitím !

Formát:

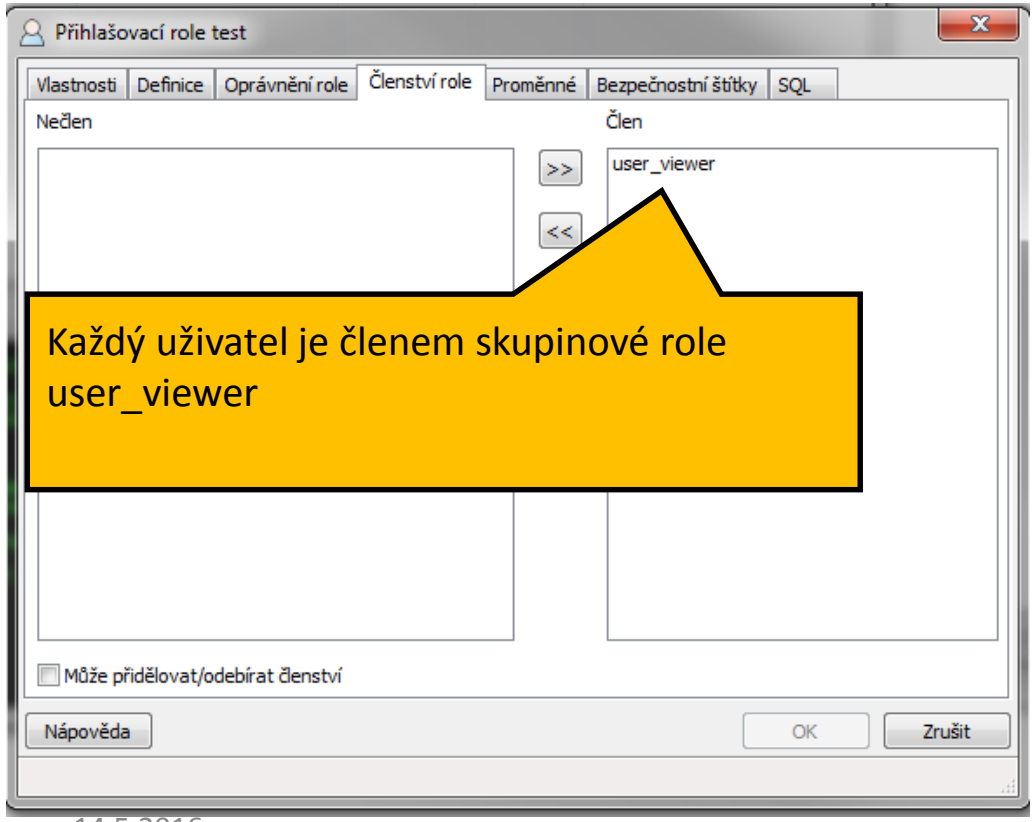
hostname:port:database:username:password

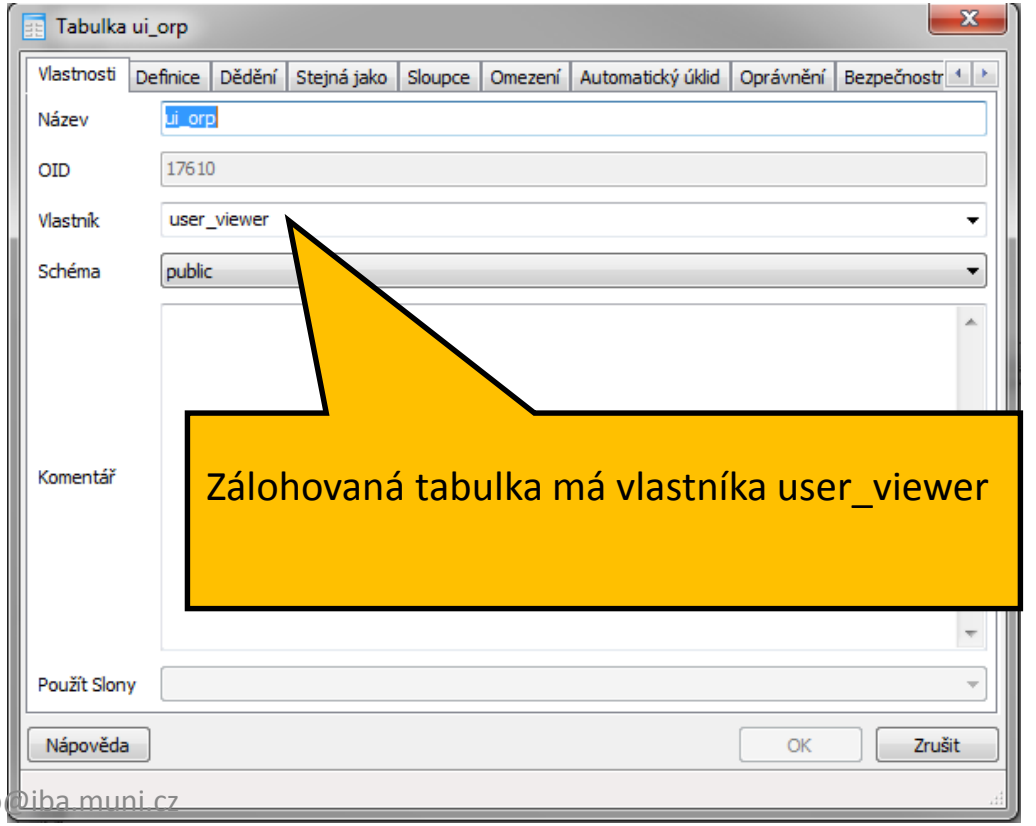
pg_admin zde uchovává hesla a
Sem se odkazují i všechny ostatní
PostgreSQL utlity

dev-enviro.cetoc.muni.cz:5432:*:salko: <heslo>
dev-enviro.cetoc.muni.cz:5432:*:genasis_datavis: <heslo>
enviro.cetoc.muni.cz:5432:*:genasis_genadmin: <heslo>
enviro.cetoc.muni.cz:5432:*:genasis_datavis_r: <heslo>
localhost:5432:*:salko: <heslo>
localhost:5432:postgres:postgres:<heslo>

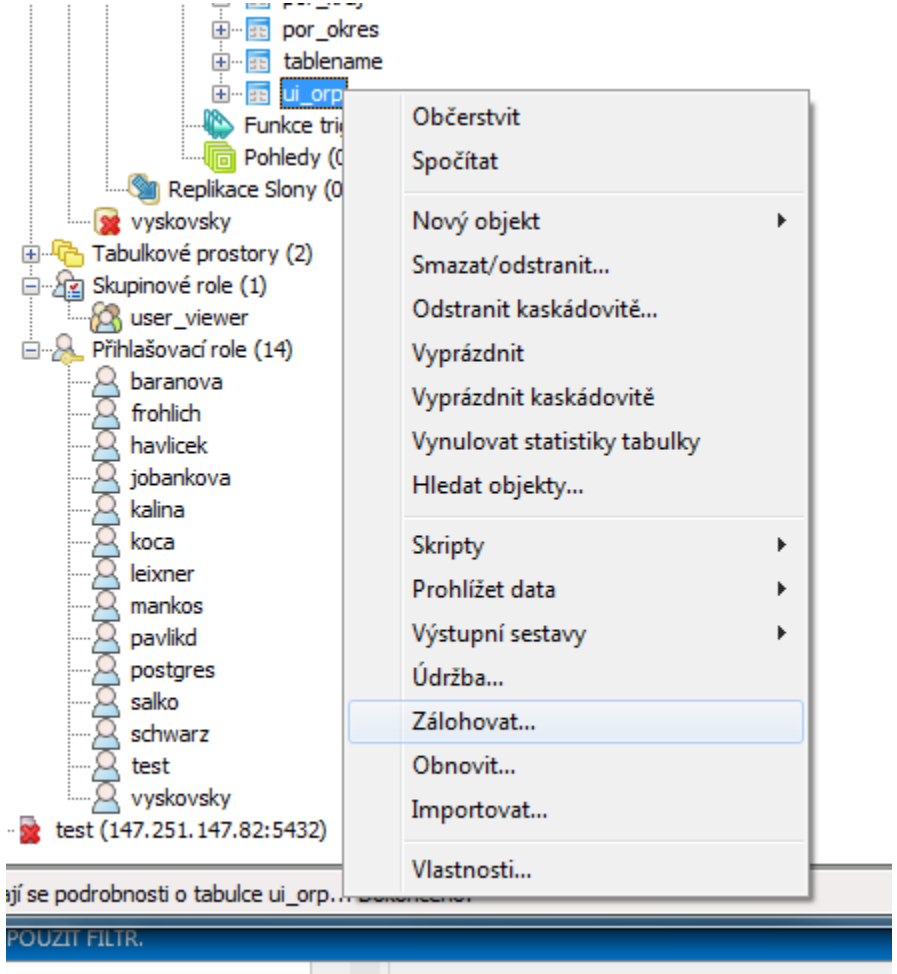
Jak vytvořit backup tabulky (tabulek)

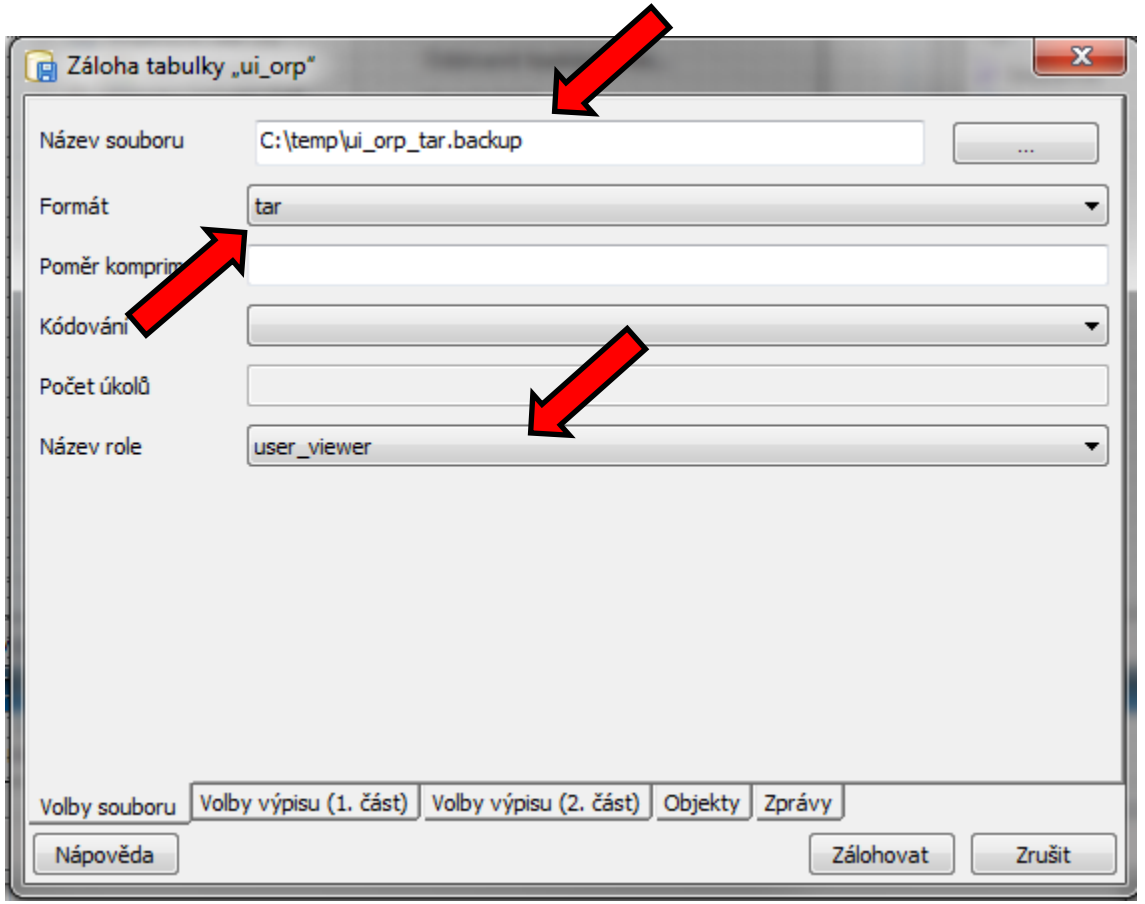
V případě, že importujeme data z jedné tabulky v jedné db do druhé musíme to dělat pod stejnou identitou, nebo musíme být členem společné skupinové roly, která musí mít práva nad db objekty.



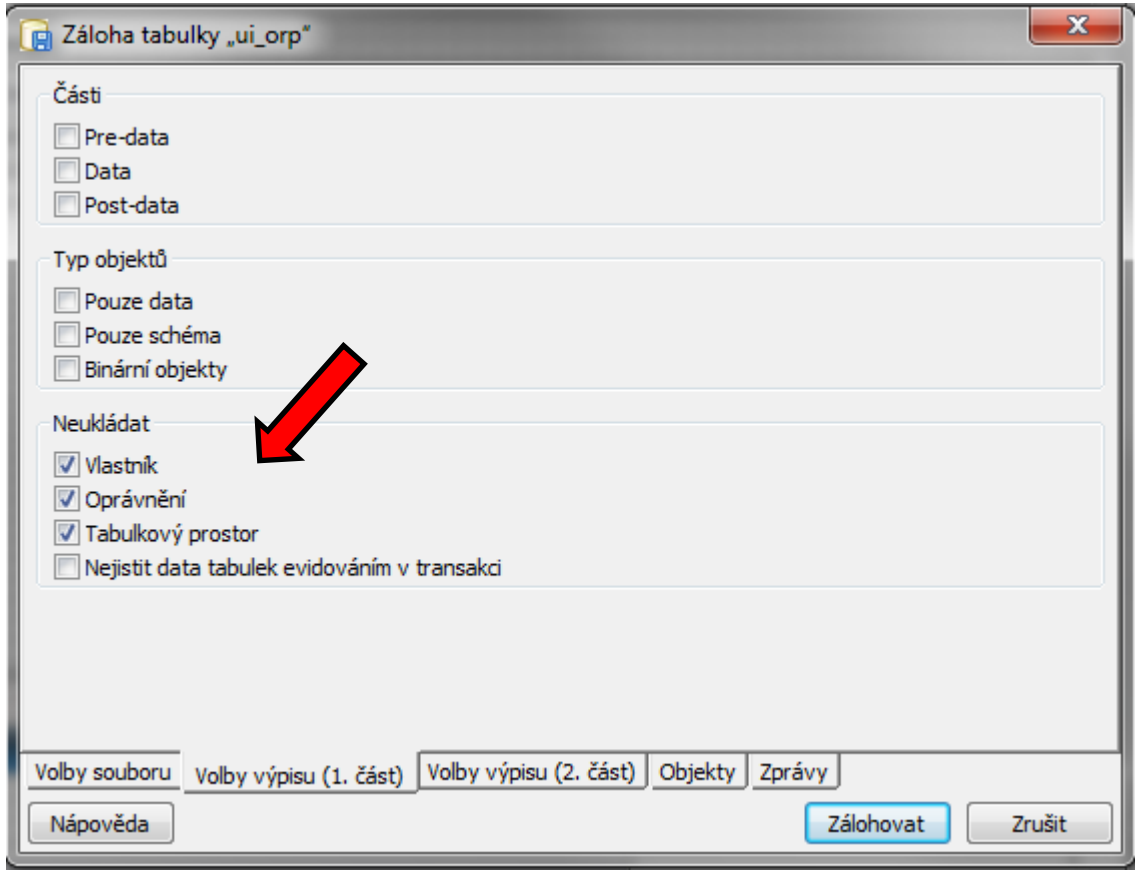


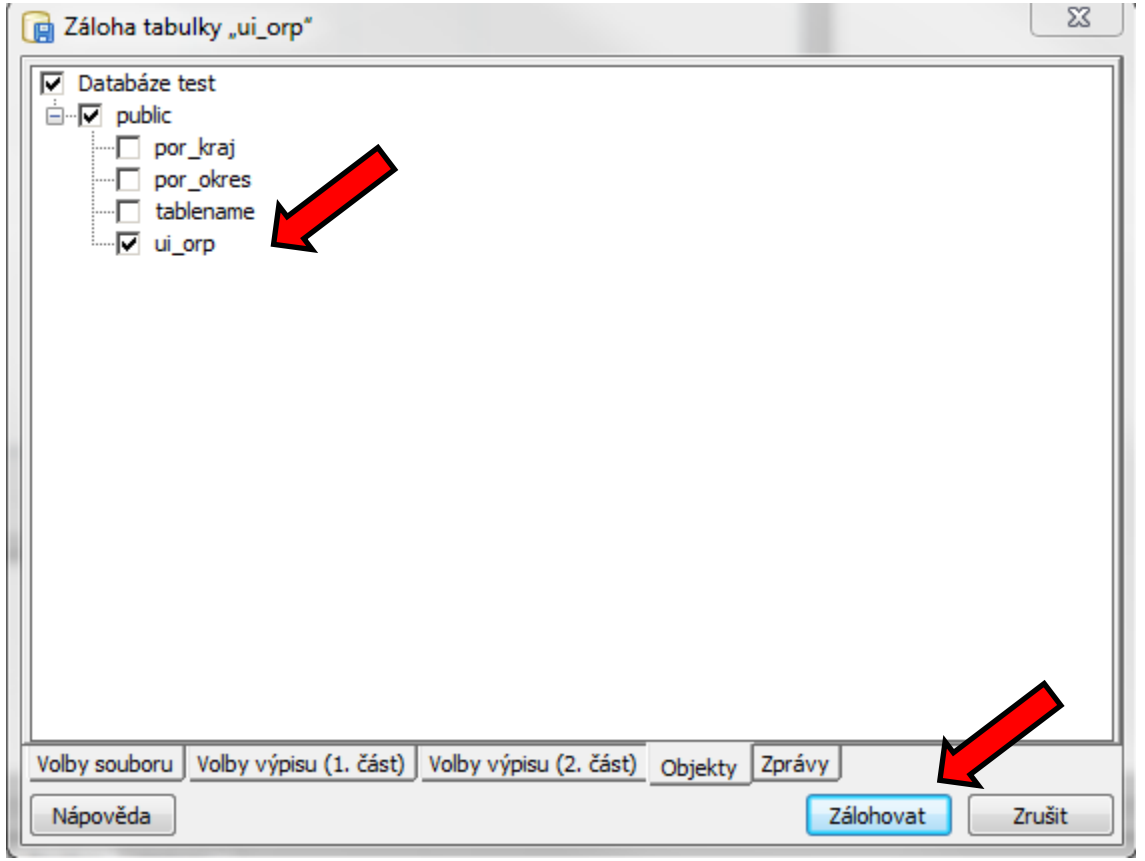
Jak vytvořit backup tabulky (tabulek)



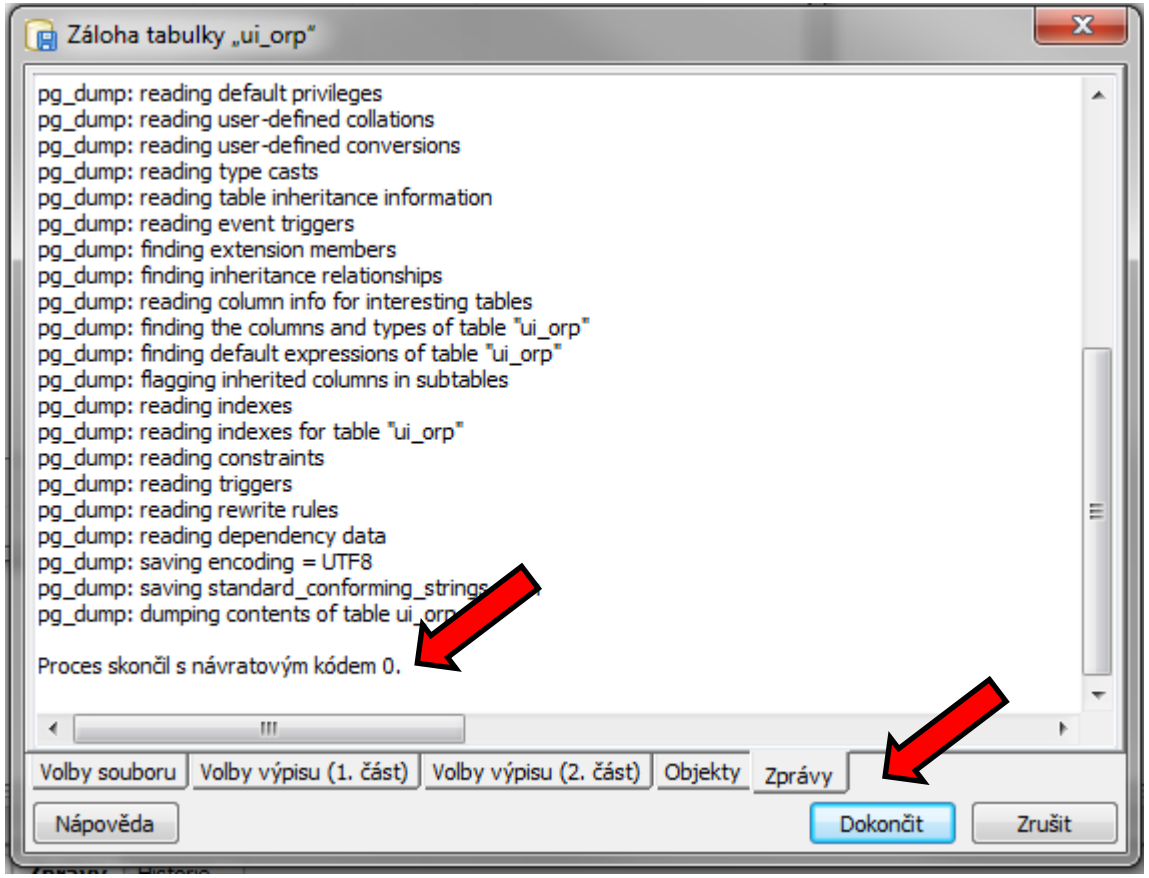


Jak vytvořit backup tabulky (tabulek)





Jak vytvořit backup tabulky (tabulek)

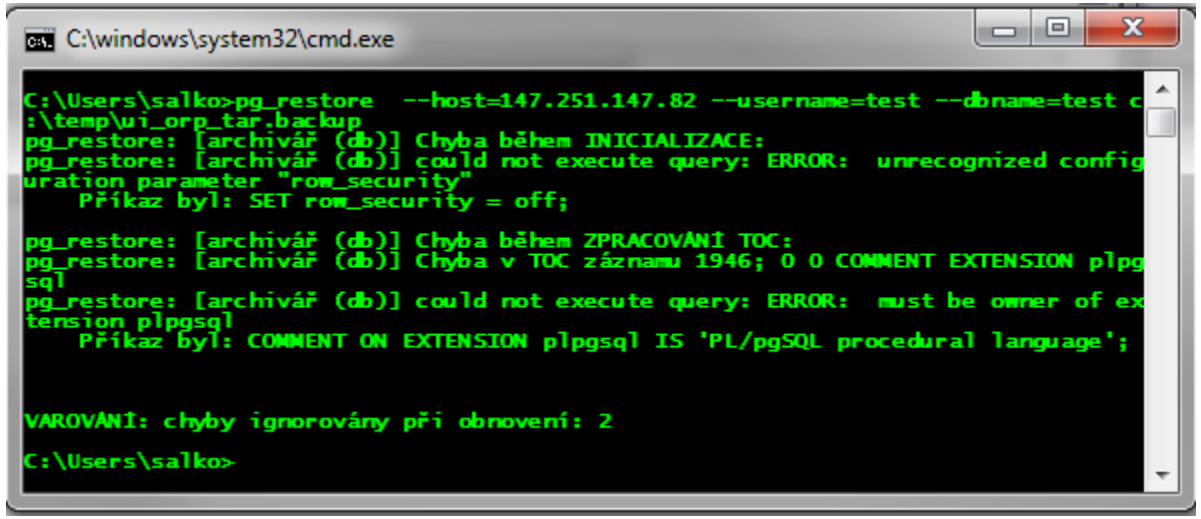


Import kompresovaného backupu

```
C:\temp>pg_restore --host=147.251.147.82 --username=test --dbname=test c:\temp\
ui_orp_tar.backup
```

Doporučuji číst hlášení utility

- Zde row_security náleží do verze 9.5. My používáme verzi 9.3. Můžeme ignorovat.
- Extenze plpgsql je vlastníkem Administrátor a tak jej nemůžeme ovlivnit. Můžeme ignorovat.



Příklad 06

Import nekompresovaného backupu

```
C:\temp>pg_restore --host=147.251.147.82 --username=test --dbna
ui_orp.backup
pg_restore: [archivář] vstupní soubor se zdá být dump v textové
te prosím psql.
```

pg_restore doporučuje použít psql

C:\temp>psql --host=147.251.147.82 --username=test --dbname=test --file=c:\temp\ui_orp.backup

Spuštěním v cmd se tabulka s daty obnoví.

Můžete ignorovat.

Tip
Je možnost otevřít SQL dotazovací nástroj a otevřít takto vytvořený dump spustit přímo z něj.

```
CREATE EXTENSION
psql:c:/temp/ui_orp.backup:30: ERROR:  must be owner of extension plpgsql
```

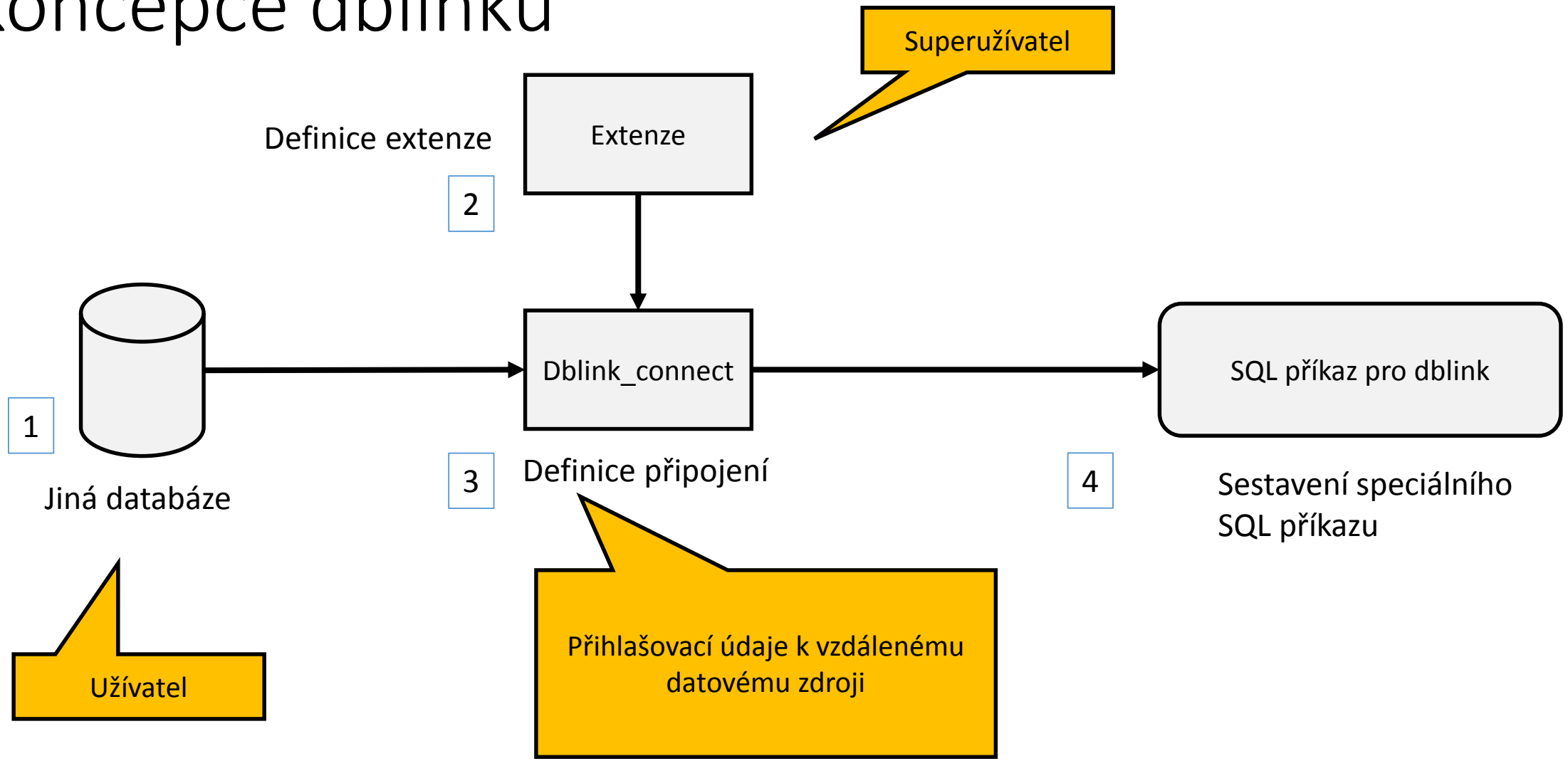
8. Import dat pomocí dblinku

Co je to dblink ?

Dblink je preresistentní připojení k vzdálené databázi. Je možné v jedné databází používat objekty v jiné databázi.

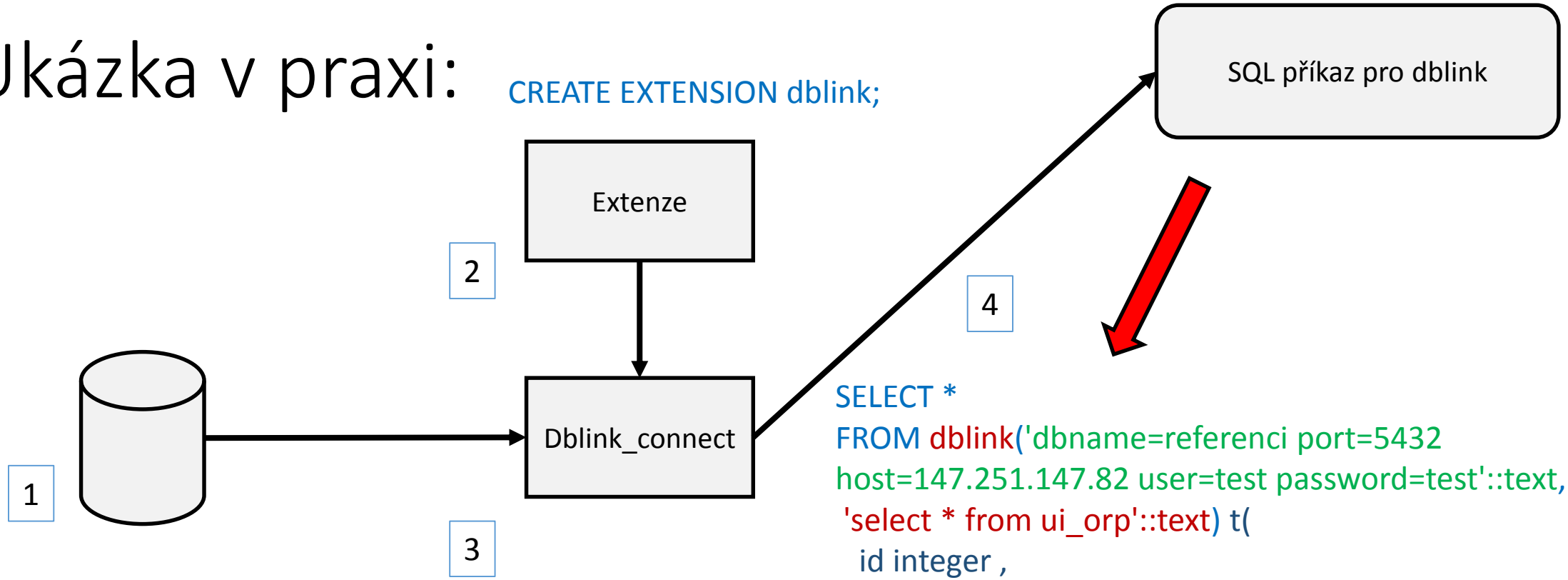
Pro extrení servery se doporučuje použít foreign table.

Koncepce dblinku



Ukázka v praxi:

CREATE EXTENSION dblink;



Jiná databáze:
Host: 147.251.147.82
Název: referencni
Port: 5432
User: test
Password: test

'dbname=**referenci**
port=**5432**
host=**147.251.147.82**
user=**test**
password=**test**' :: text

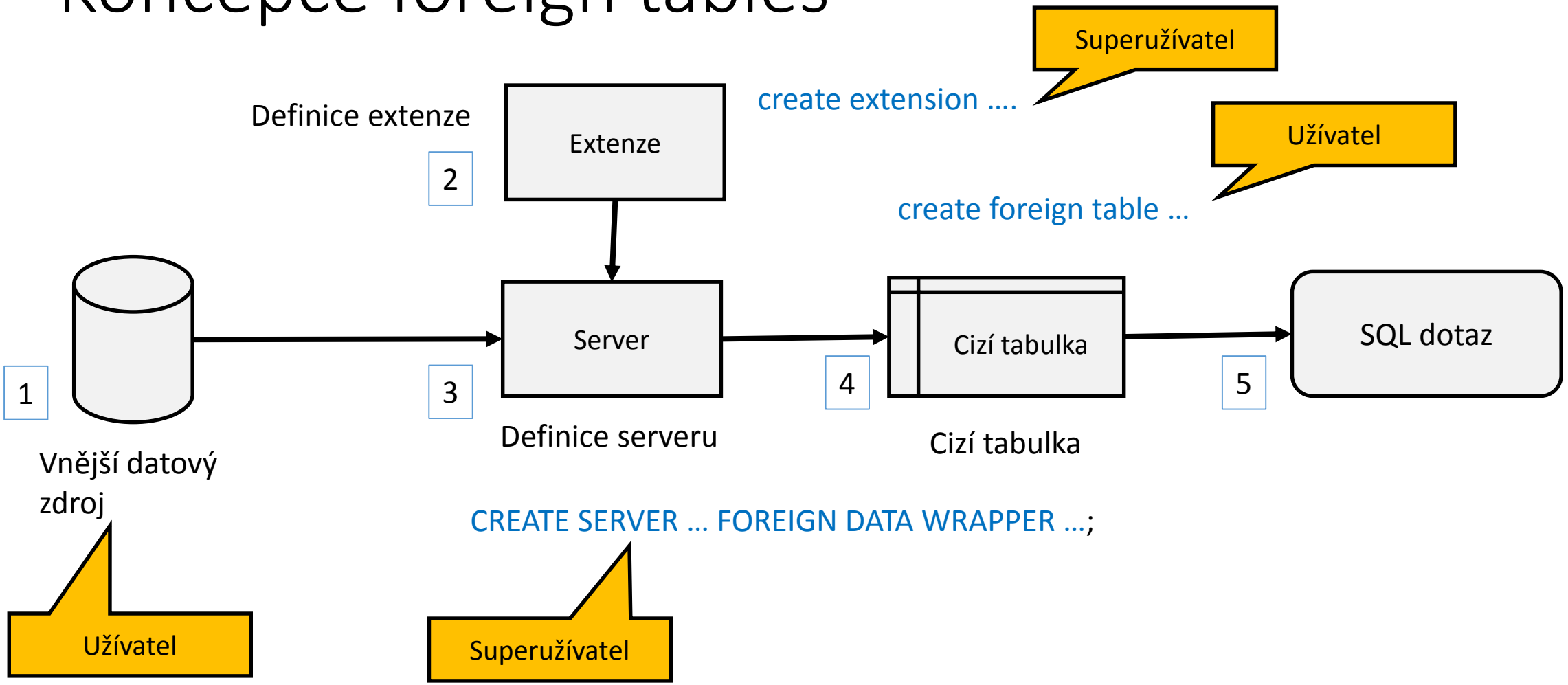
```
SELECT *
FROM dblink('dbname=referenci port=5432
host=147.251.147.82 user=test password=test'::text,
 'select * from ui_orp'::text) t(
  id integer ,
 kod integer,
 nazev text,
 spravni_obec_kod integer,
 vusc_kod integer,
 plati_od date,
 plati_do date,
 datum_vzniku date);
```

9. Import dat pomocí foregin tables

Co je to foreign tables ?

Foreign table (nebo cizí tabulka) je způsob jak vnější datové zdroje použít v běžné databáze.

Koncepce foreign tables



Jaké mohou být vnější datové zdroje

Zdroj: Foreign data wrappers

https://wiki.postgresql.org/wiki/Foreign_data_wrappers

DB rozhraní:

- ODBC
- JDBC
- JDBC2
- …

Databáze relační:

- PostgreSQL
- Oracle
- MySQL
- Informix
- SQLLite
- MonetDB
- …

NoSQL databáze:

- MongoDB
- Casandra
- CoucheDB
- …

File wrapper:

- CSV
- XML
- JSON
- pg_dump
- …

A další …
(Je možné psát i vlastní)

Interní moduly pro foreign table

file_fdw

Je extenze, která umožňuje řízení datových souborů na serveru. Ekvivalent příkazu copy.

Zdroj: http://www.postgresql.org/docs/9.3/static/file-fdw.html

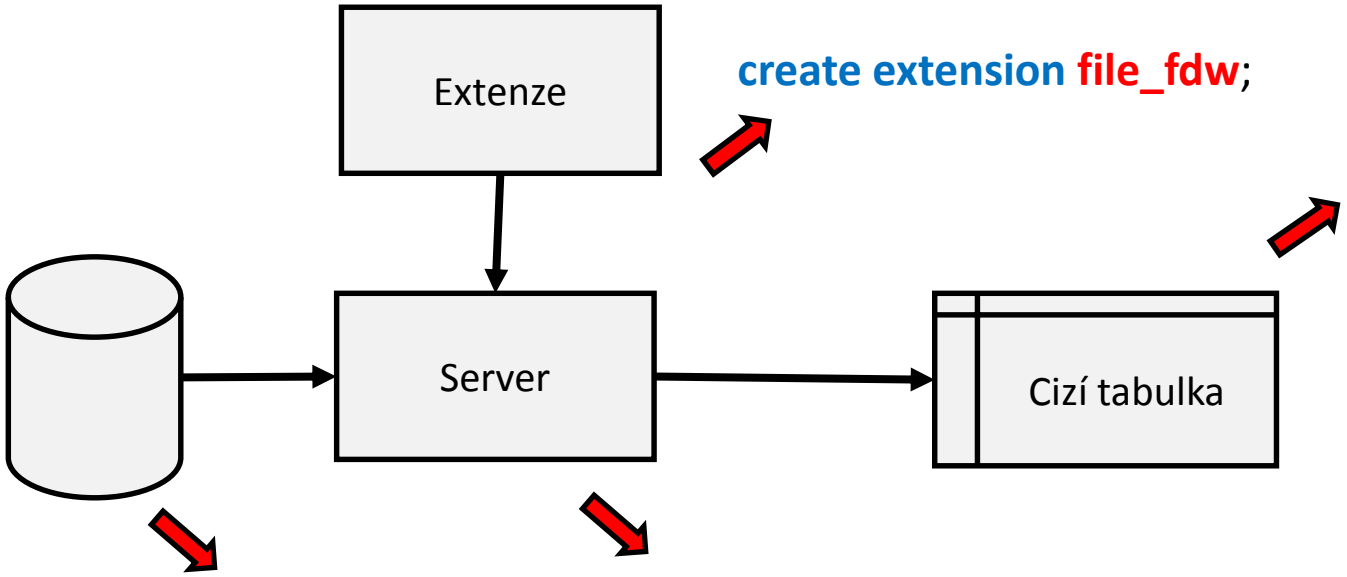
postgre_fdw

Umožňuje řízení dat na extreních PostgreSQL serverech.

Zdroj:http://www.postgresql.org/docs/9.3/static/postgres-fdw.html

Ukázka v praxi: Zpřístupnění CSV

Příklad 08




```
create extension file_fdw;
```

```
create foreign table ui_obec (
kod text,
nazev text,
status_kod text,
pou_kod text,
okres_kod text,
cleneni_sm_rozsah_kod text,
cleneni_sm_typ_kod text,
plati_od text,
plati_do text,
datum_vzniku text
) SERVER file_server
OPTIONS (format 'csv',header 'true',filename
'/home/salko/import/ui_obec.csv', delimiter ';', null '');
```

Soubor: /home/salko/import/ui_obec.csv

```
CREATE SERVER file_server FOREIGN DATA WRAPPER file_fdw;
```

Příklad SQL dotazu : select * from ui_obec;

Vkládaní velkého počtu dat

1. Zakažte autocommit
2. Použijte COPY
3. Zrušte indexy
4. Zrušte cizí klíče
5. Zvyšte max_val_size
6. Zakažte WAL archivaci a streamovou replikaci
7. Spustěte později ANALYZE

Zdroj: http://www.postgresql.org/docs/9.5/static/populate.html

Konec školení