Základ je tabulka

Lehký úvod do PostgreSQL

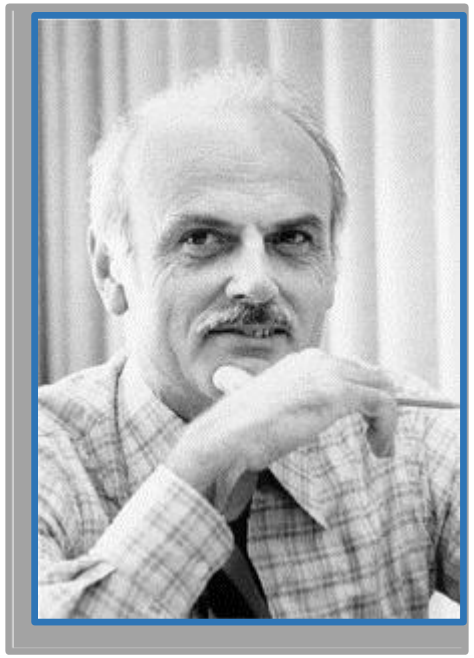
Agenda školení

1. Úvod do databáze PostgreSQL
2. Klientské nástroje pro práci s databází
3. Co je to SQL jazyk
4. Základní datové typy používané v PostgreSQL
5. Cizí klíč (Foreign key)
6. Constraint Check
7. Constraint Unique

1. Úvod do databáze PostgreSQL

RDBMS – Relation Database Management System

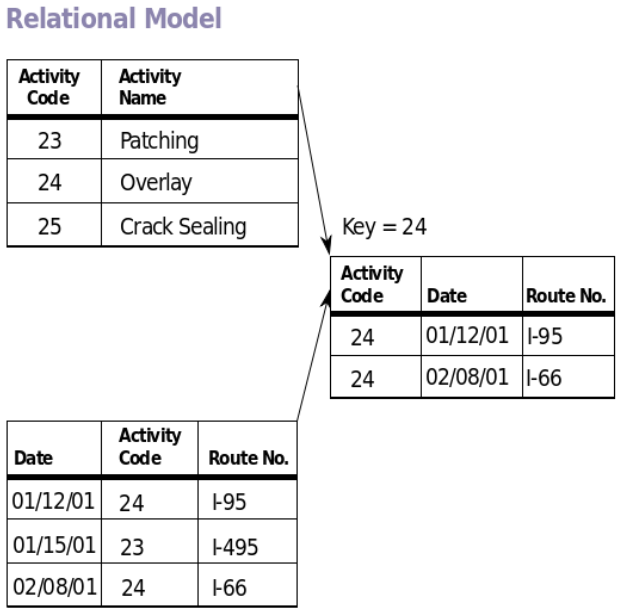
Je databázový řídicí systém založený na relačním modelu vymyšleném E.F.Codd pro IBM [San Jose Research Laboratory]



Edgar Frank "Ted" Codd
(19 August 1923 – 18 April 2003)

Zdroj: https://en.wikipedia.org/wiki/Edgar_F._Codd

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



PostgreSQL

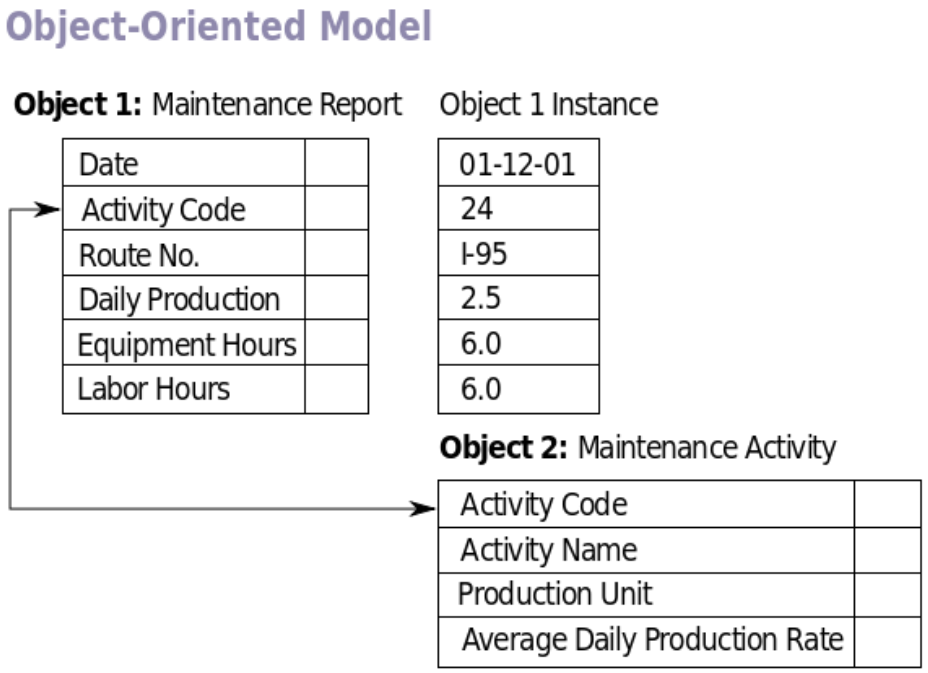
Je objektovo-relační databázový systém (ORDBMS) založený na Postgres, verze 4.2.
Vyvinutý byl University of California at Berkeley Computer Science Department.
POSTGRES je průkopníkem mnohých koncepcí, které později převzali některé komerční databázové systémy.

Zdroj: http://www.postgresql.org/docs/9.3/interactive/intro-whatis.html

Současná verze 9.5.1

ORDMBS –Object Relation Database Management System

Objektově-relační databázový řídicí systém je databázový systém podobný RDBMS ale s objektově-orientovaným databázovým modelem : objekty, třídy a dědění jsou přímo podporované v databázových schématech a dotazovacím jazyku. Oproti čistě relačním systémům obsahují navíc rozšíření (extenzi) datového modelu pomocí **uživatelských datových typů a metod.**



Zdroj: https://en.wikipedia.org/wiki/Object-relational_database

ORDBMS - charakteristiky

1. Komplexní data

Definice modelů pomocí uživatelsky definovaných typů (UDT)

2. Typová dědičnost

Strukturované typy mohou obsahovat subtypy.

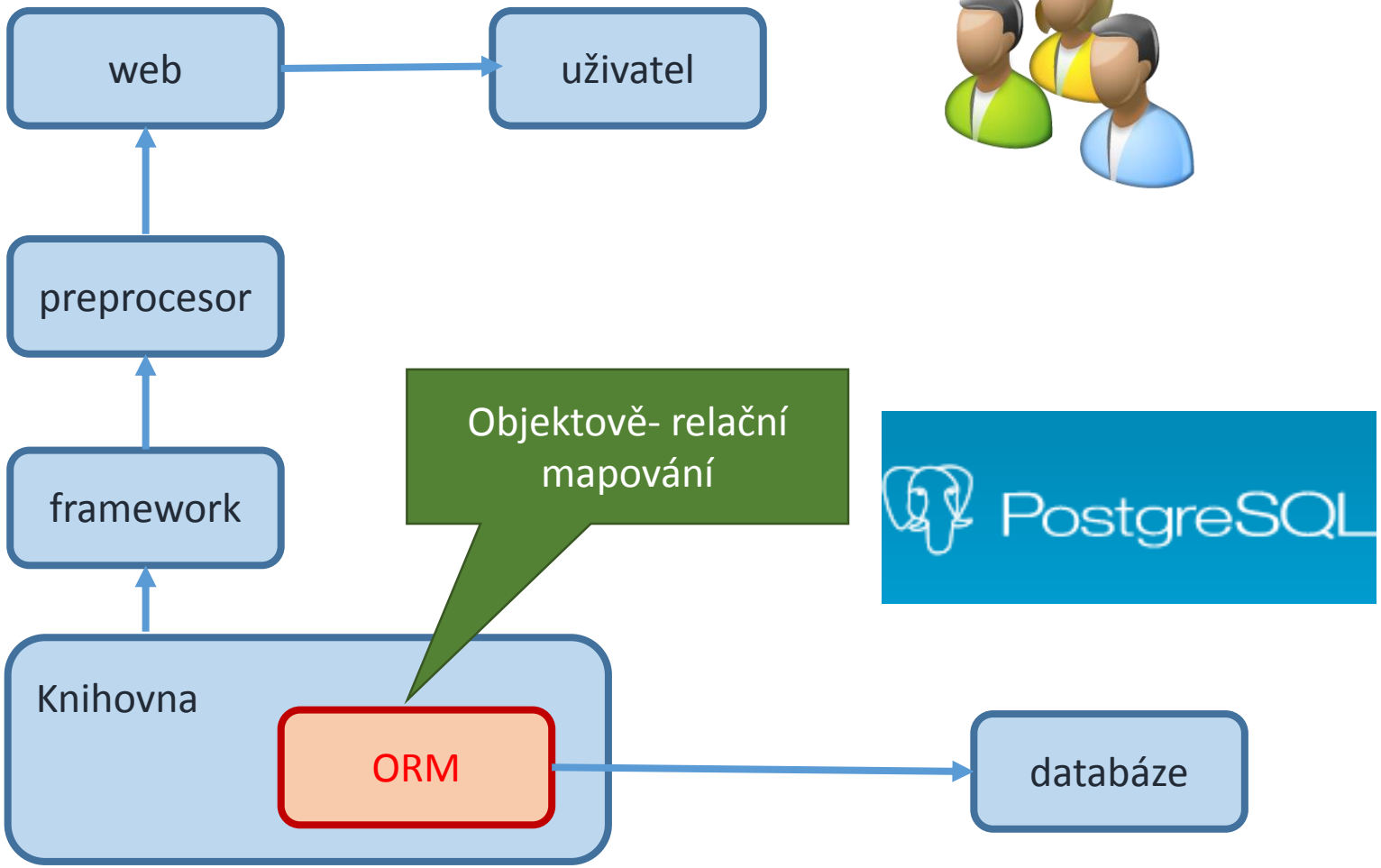
3. Objektové omezení

Jsou relace přes objekty. Objekty jsou uchovávané jako persistentní objekty.
Ty jsou identifikovány pomocí objektového identifikátoru (OID).

Zdroj: https://en.wikipedia.org/wiki/Object-relational_database

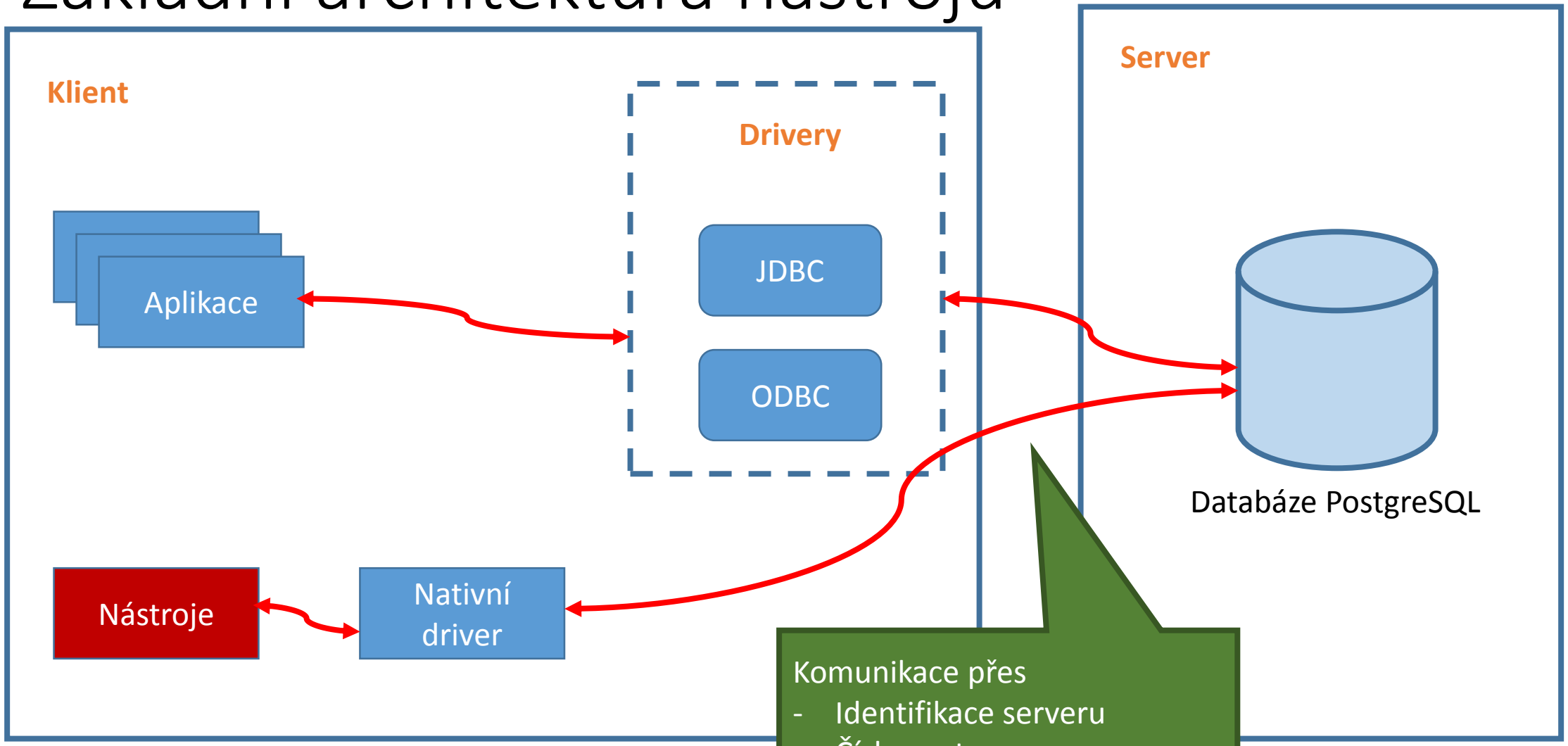
ORDBMS na IBA

Doctrine



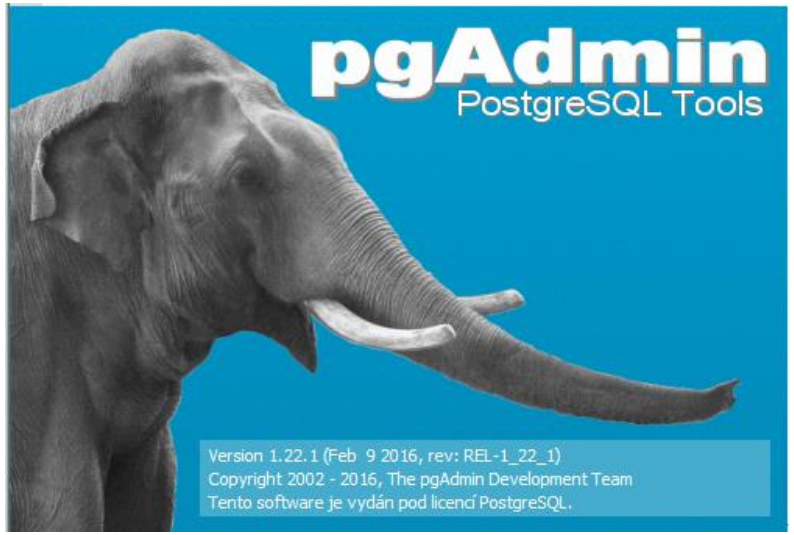
2. Klientské nástroje pro práci s databází

Základní architektúra nástrojů



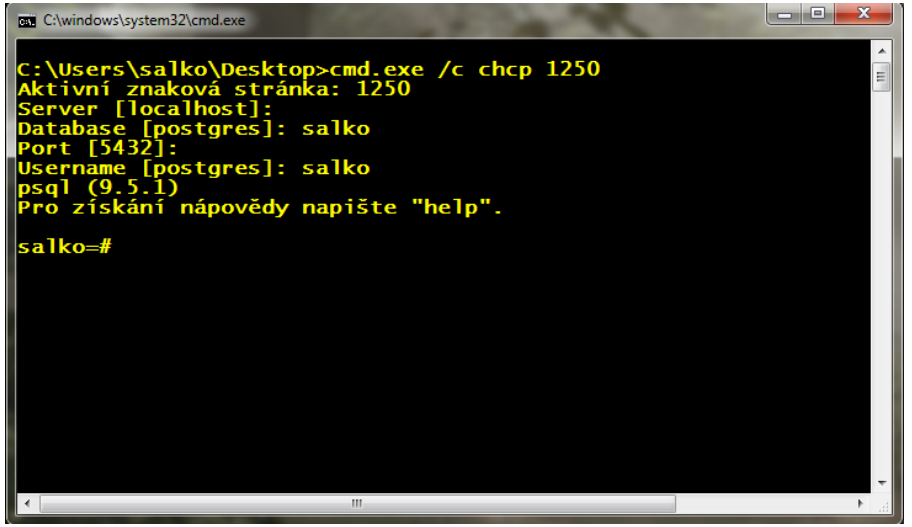
Základní nástroje

pgAdmin



Sada vizuálních nástrojů pro práci s PostgreSQL databázemi.

psql

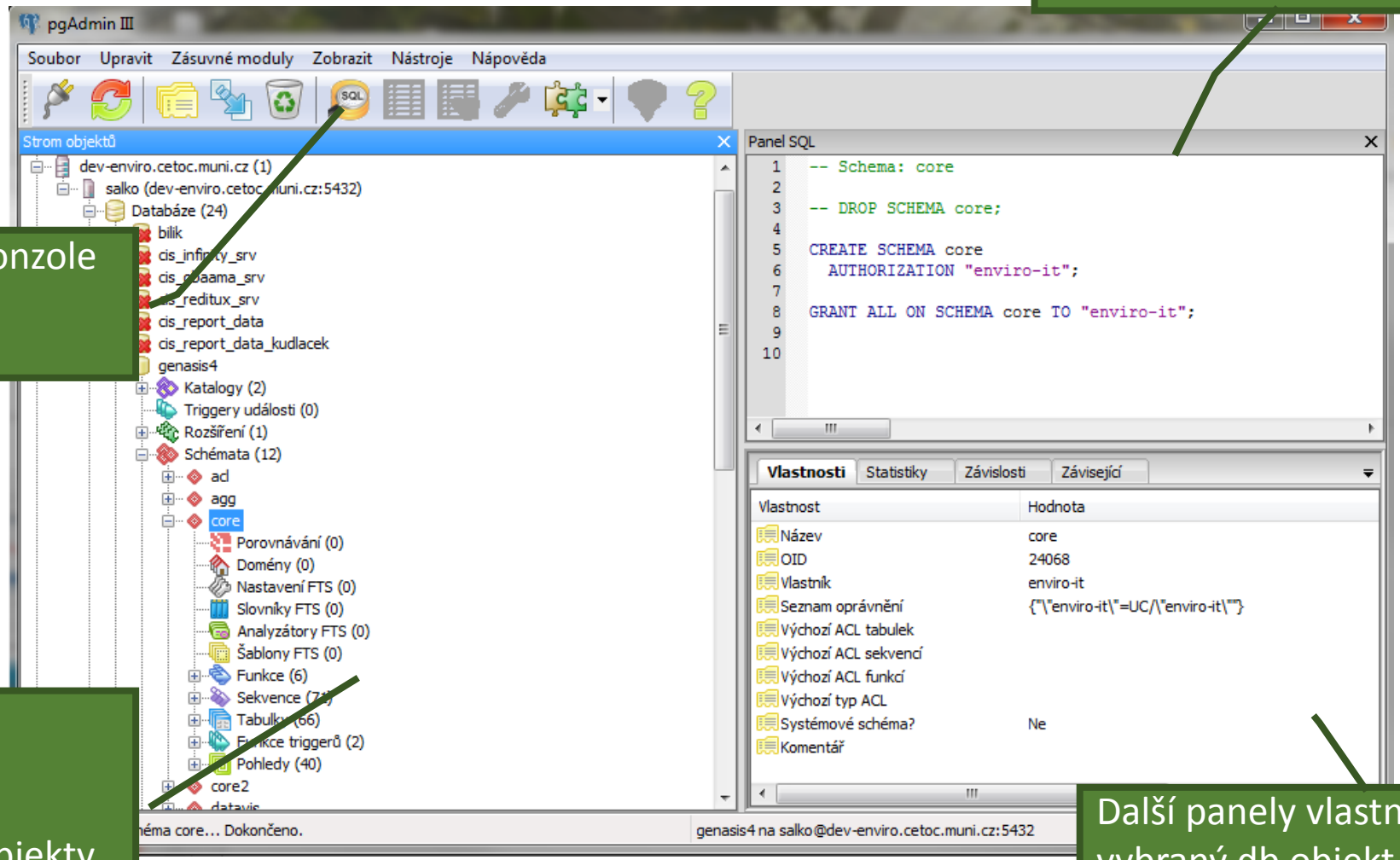


Konzolová aplikace pro práci s PostgreSQL databází.

Upozornění:
Konzole pracuje v kódové stránce 852 (nebo může i Win1250). PostgreSQL pracuje s UTF-8 defaultně.
Ne všechna znaky se správně, nebo vůbec zobrazí na výstupu konzoly.

PgAdmin

SQL Panel
SQL příkazy pro vytvoření vybraného db objektu



Spuštění SQL windows konzole k aktuální databázi

Strom objektů
- Skupiny serverů
- Servery – connect
- Základní Datbázové objekty

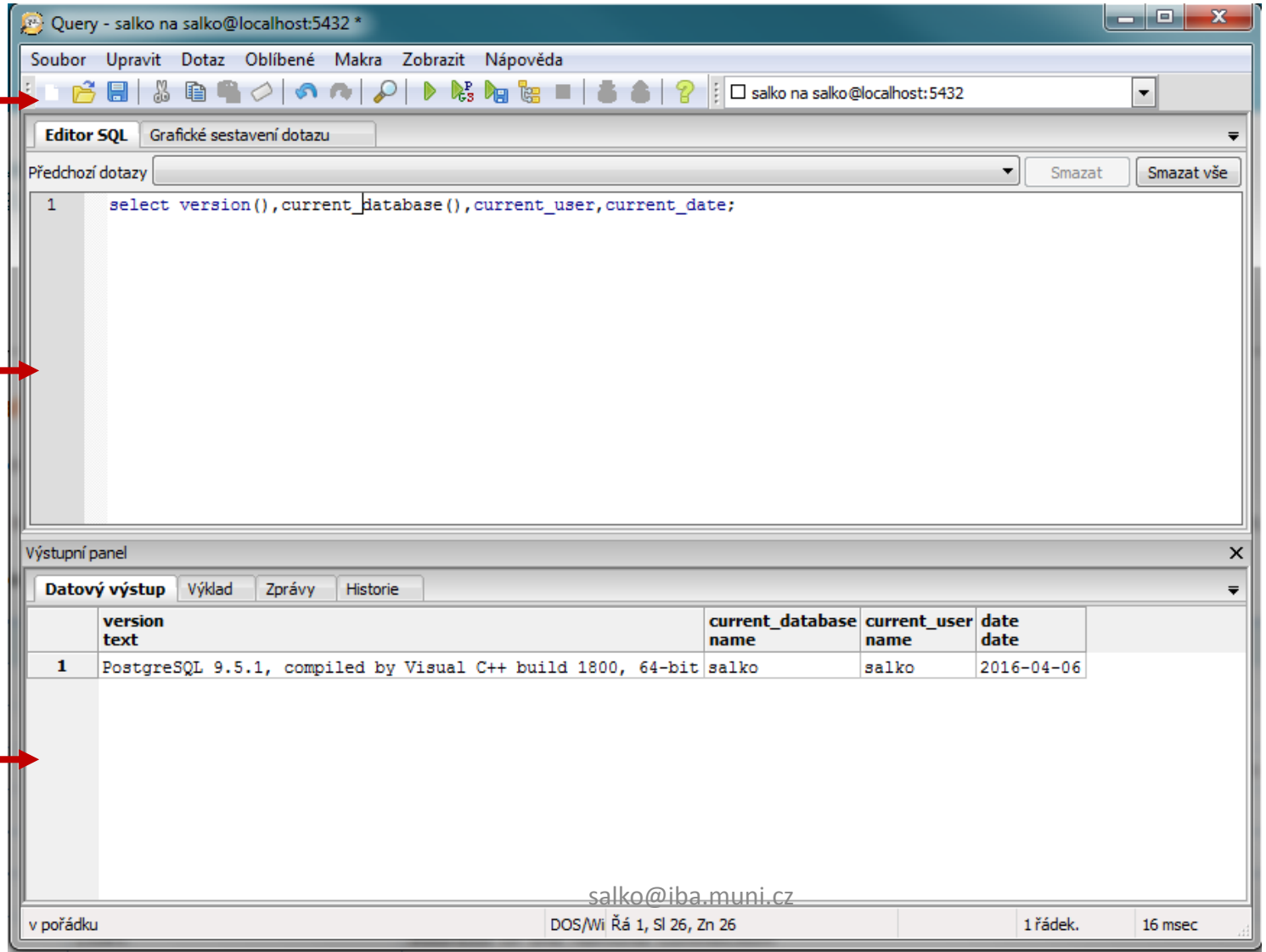
Další panely vlastností pro vybraný db objekt

PgAdmin windows konzole

Panel nástrojů

Editace dotazů
a
Panel graf. dotazů

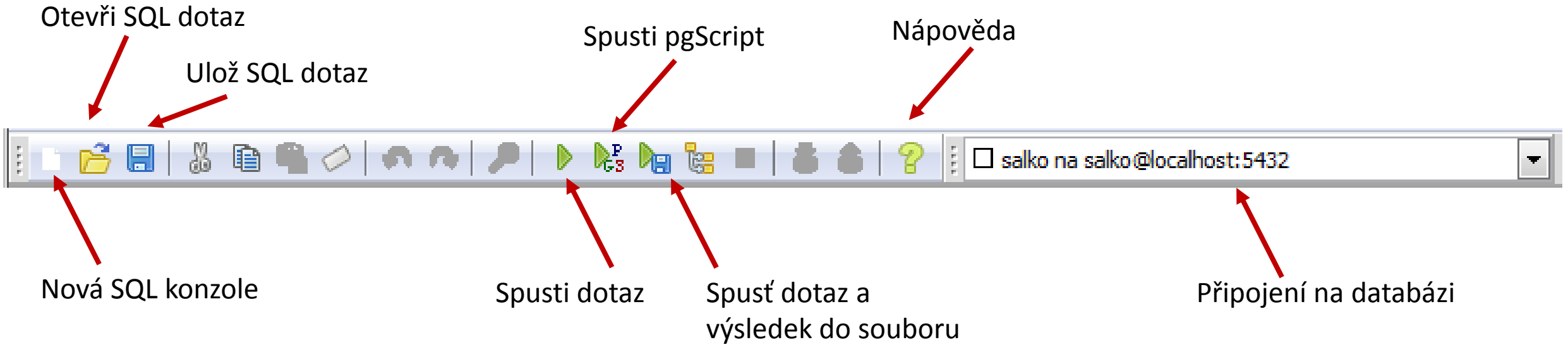
Výstupní panely



Lze :

- Spouštět SQL Dotazy
- Definovat db objekty
- Ukládat dotazy
- Otvírat dotazy
- Ukládat výsledky
- Grafické dotazy

PgAdmin SQL konzole panel nástrojů



3. Co je to SQL jazyk ?

SQL

SQL (vyslovováno anglicky *es-kjů-el* [ɛs kjuː ɛł] IPA, někdy též *síkvl* [siːkwəl] IPA) je zkratka (anglicky *Structured Query Language*) pro standardizovaný strukturovaný dotazovací jazyk, který je používán pro práci s daty v relačních databázích.

Zdroj: https://cs.wikipedia.org/wiki/SQL

- Vznikl standard SQL-86.
- V roce 1992 byl proto přijat nový standard *SQL-92* (někdy se uvádí jen *SQL2*).
- Zatím nejnovějším standardem je *SQL3* (*SQL-99*).

Skupiny příkazů

- Příkazy pro manipulaci s daty (SELECT, INSERT, UPDATE, DELETE, …)
- Příkazy pro definici dat (CREATE, ALTER, DROP, …)
- Příkazy pro řízení přístupových práv (GRANT, REVOKE)
- Příkazy pro řízení transakcí (START TRANSACTION, COMMIT, ROLLBACK)

Ještě existuje další dělení na:
- DML – manipulační příkazy
- DQL – dotazovací příkazy

Příklad: 01_sql_prikazy_obecne.sql

SQL jazyk

1. Vytvoření tabulky

```
create table obchod
(id serial primary key,
 nazev text not null unique,
 hodnoceni smallint not null check (hodnoceni > 0 and hodnoceni < 5),
 datum_vytvoreni date not null default now());
 -- drop table obchod;
```

DDL – jazyk
(create,alter,drop)

2. Vložení dat

```
insert into obchod
(nazev,hodnoceni)
values
('Billa',1),
('Albert',2),
('Globus',1),
('Penny',4),
('Lidl',3);
```

DML – jazyk
(insert,update,delete,truncate)

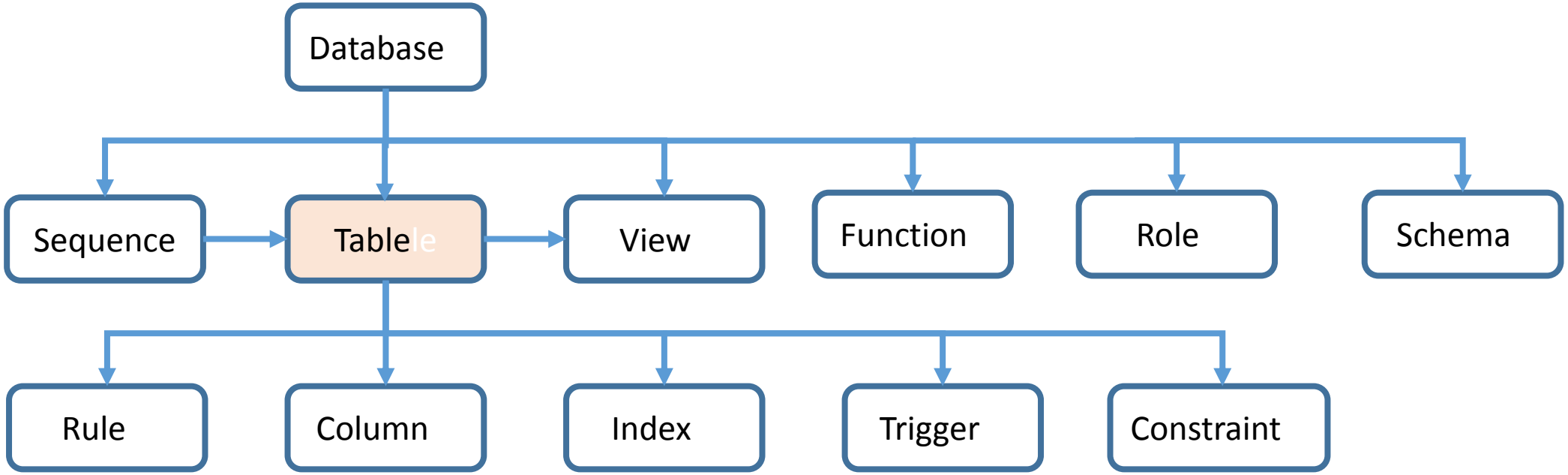
3. Výběr dat

```
select * from obchod;
```

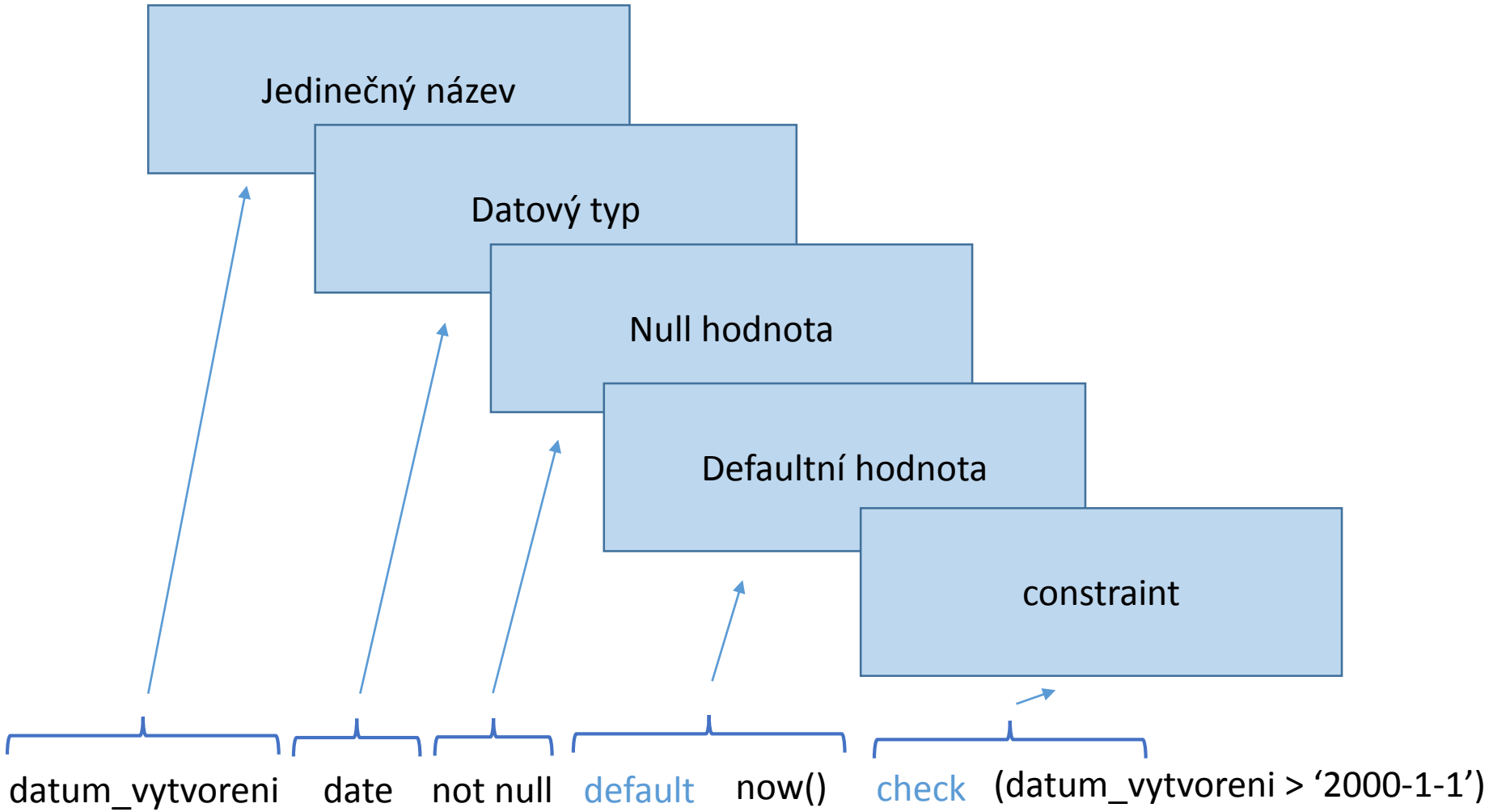
DQL – jazyk
(select)

Id	Nazev	Hodnoceni	Datum_vytvoreni
1	Billa	1	2016-03-21
2	Albert	2	2016-03-21
3	Globus	1	2016-03-21
...			

Základní objekty PostgreSQL



Atributy sloupce



Pojmenování sloupců – dobrá praxe

Charakteristika hodnot uložených pod daným sloupcem

V našem kulturním prostředí se používají mnemo zkratky :

txt – text	dat - datum
id – identifikátor	status - stav

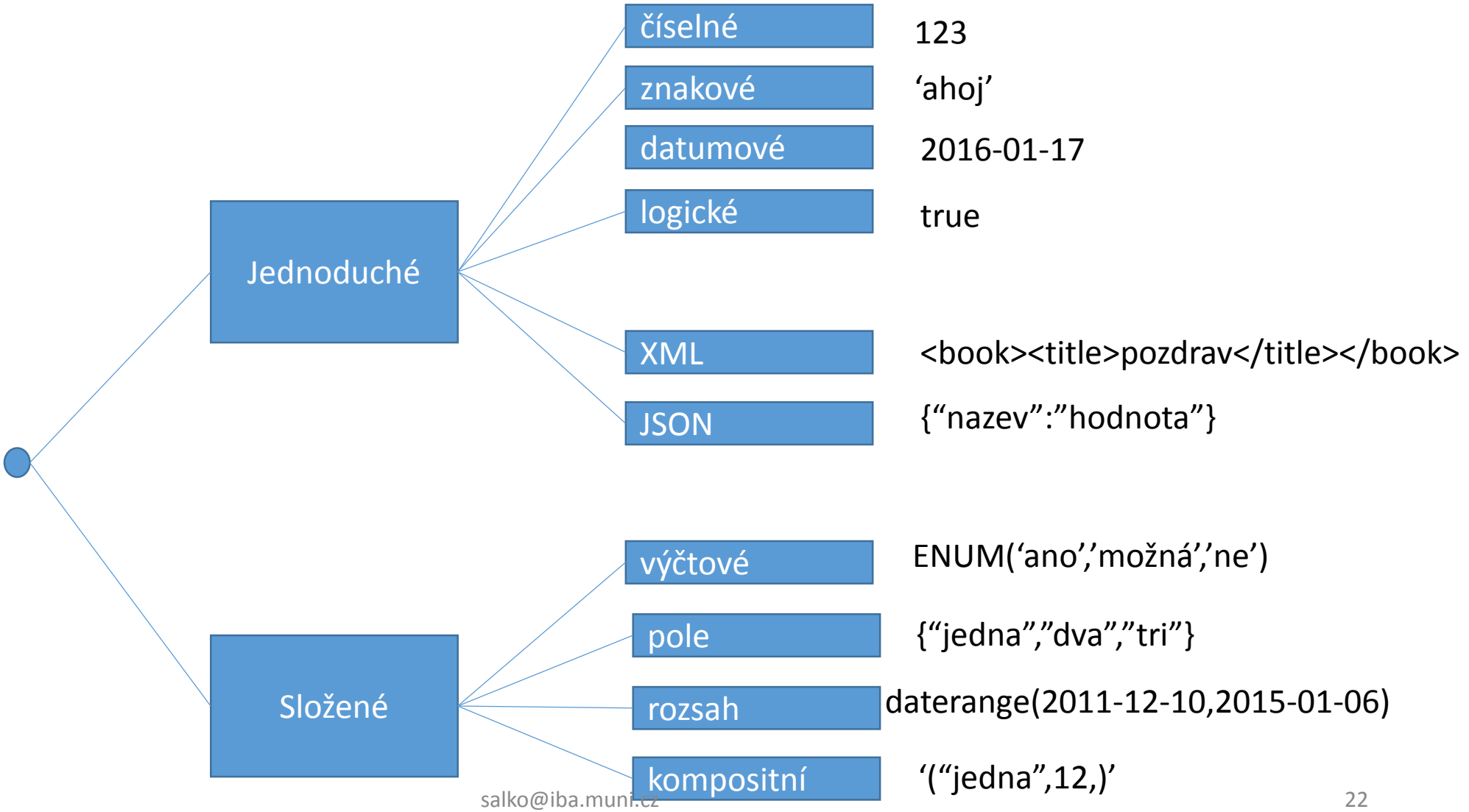
Obvykle jsou to názvy bez diakritiky v jednotném čísle. Slova jsou obvykle odděleny podtržítkem.

Množné číslo používat, když se jedná o pole, nebo Json, XML, ...

4. Základní datové typy používané v PostgreSQL

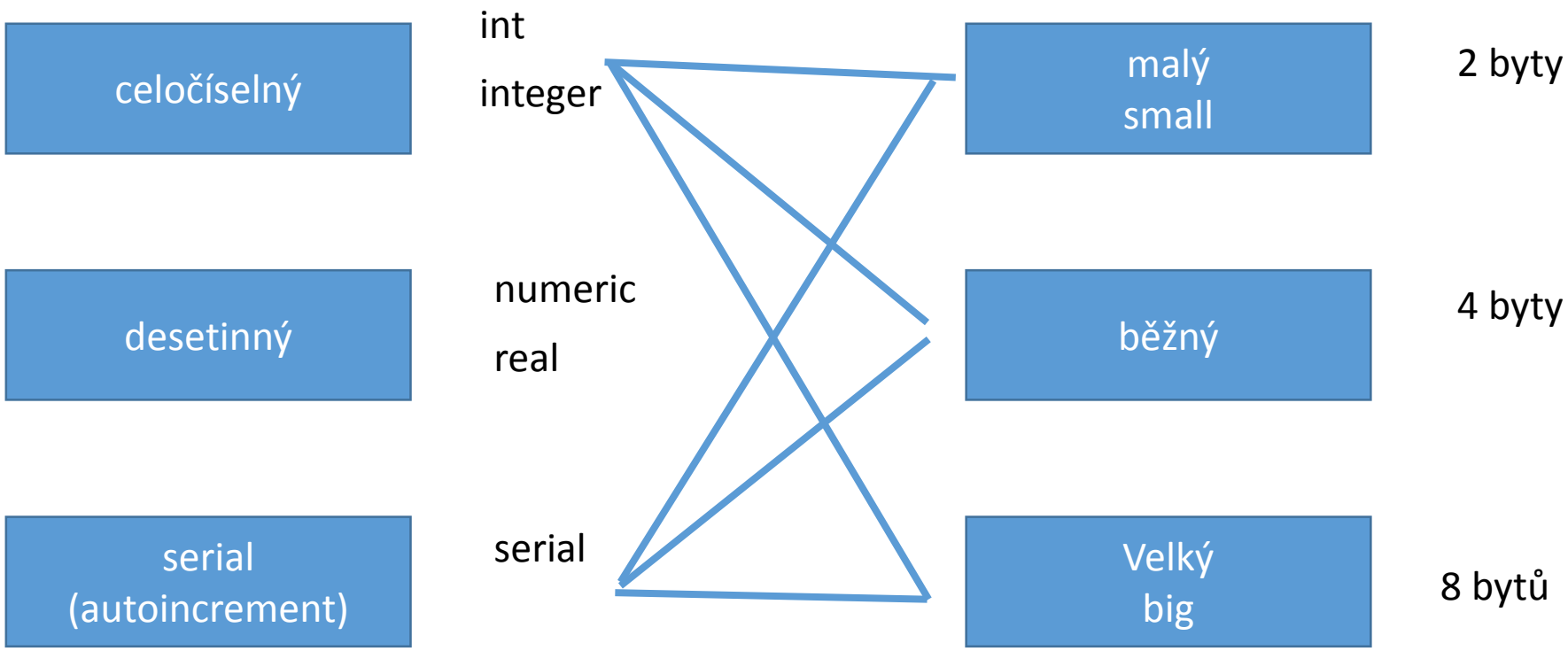
Datový typ sloupců v PostgreSQL

Příklad: 02_sql_datove_typy.sql



Číselné datové typy

03_sql_ciselne_datove_typy.sql



Znakové datové typy

04_sql_znakove_datove_chyby

variabilní délka

varchar(n)

fixní délka

char(n)

Libovolná délka

text

```
CREATE TABLE test1 (a character(4));
INSERT INTO test1 VALUES ('ok');
SELECT a, char_length(a) FROM test1; -- (1)

  a   | char_length
------+-------------
 ok   |           2

CREATE TABLE test2 (b varchar(5));
INSERT INTO test2 VALUES ('ok');
INSERT INTO test2 VALUES ('good      ');
INSERT INTO test2 VALUES ('too long');
ERROR:  value too long for type character varying(5)
INSERT INTO test2 VALUES ('too long'::varchar(5));
-- explicit truncation
SELECT b, char_length(b) FROM test2;

   b    | char_length
-------+-------------
 ok     |           2
 good   |           5
 too l  |           5
```

Datumové datové typy

05_sql_datumove_datove_typy.sql

časové razítko

timestamp [without time zone]

```
select now();
select current_timestamp;
```

	now timestamp with time zone
1	2016-03-09 08:04:52.042354+01

```
select now()::timestamp;
```

	now timestamp without time zone
1	2016-03-09 08:06:29.105724

datum

date

```
select now()::date;
select current_date;
```

	now date
1	2016-03-09

čas

time
time without time zone

```
select now()::time;
select current_time;
```

	now time without time zone
1	08:10:38.972363

interval

```
select now() - '2016-03-13' as days;
```

	days interval
1	-3 days -15:50:53.6294

Datumové datové typy – příklad

Rozdíl mezi různými datumovými typy

```
-- drop table pokus1;

create table pokus1
(id serial primary key,
 rozdil interval day,
 datum date default now()::date,
 cas_lokalni   time without time zone  default now()::time,
 cas_svetovi   time with time zone  default now()::time);

insert into pokus1 (rozdil)
values (now() - '1964-03-07')
returning *;
```

Interval

Na konci časová zóna

	id integer	rozdil interval day	datum date	cas_svetovy time without time zone	cas_lokalni time with time zone
1	1	18995 days	2016-03-09	08:21:00.830056	08:21:00.830056+01

Datumové datové typy

```
-- Nastaveni lokalniho casu pro Japonsko a zase zpatky

SET TIMEZONE TO 'Japan';
select now()::time with time zone, now()::time without time zone;
SET TIME ZONE LOCAL;
```

	now time with time zone	now time without time zone
1	18:02:45.502843+09	18:02:45.502843

Časová zóna je posunuta
o 9 hodin

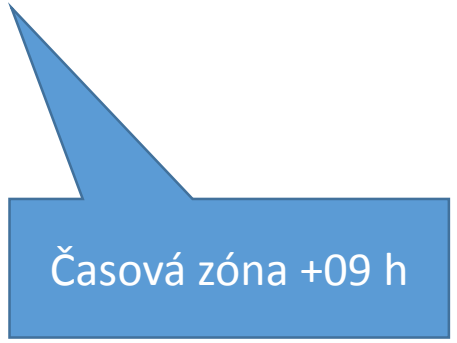
Datumové datové typy

```
-- Orezava se na minuty
select INTERVAL '-08:34:52' HOUR TO MINUTE ;

 interval
----------
 -08:34:00
(1 řádka)
```

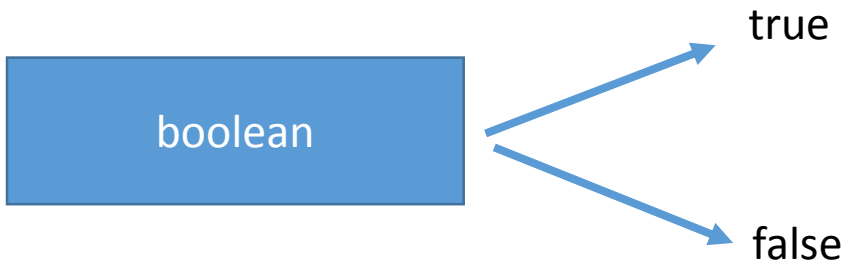
```
-- Kolik hodin je dana hodnota v minutach
select INTERVAL '457898' MINUTE

 interval
------------
 7631:38:00
(1 řádka)
```

```
-- Zobrazeni bezneho casoveho razitka s casovou zonou
select to_char(CURRENT_TIMESTAMP,'dd.mm.yyyy HH24:MI (TZ)')
as datum_nemecky;

 datum_nemecky
-----------------------
 09.03.2016 10:11 (CET)
(1 řádka)
```

```
-- Prevod casu v timezone na cas ve stredni Evrope
select time with time zone '16:31:46.311947+09' AT TIME ZONE 'CET'
as cas_lokalni;

 cas_lokalni
-------------------
 08:31:46.311947+01
(1 řádka)
```



Logický datový typ



V případě použití insert,update,delete je možno použít klíčové slovo returning na vrácení počtu záznamů

06_sql_logicke_datove_typy.sql

```
-- drop table tab_log;

-- Vytvori se tabulka
create table tab_log
(id serial primary key,
 hodnota boolean default false,
 nazev varchar(60));

-- Vlozi se zaznamy
 insert into tab_log
 (hodnota,nazev)
 values
 (true,'platí'),
 (false,'neplatí')
 returning *;

-- Odpoved
 id | hodnota |  nazev
----+---------+---------
  1 | t       | platí
  2 | f       | neplatí
(2 řádky)
```

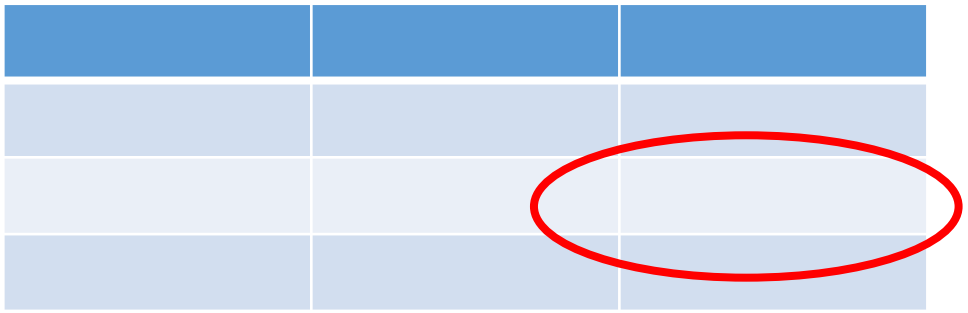
XML datový typ

Extensible Markup Language (zkráceně **XML**, česky *rozšiřitelný značkovací jazyk*)
je obecný značkovací jazyk, který byl vyvinut a standardizován konsorciem W3C

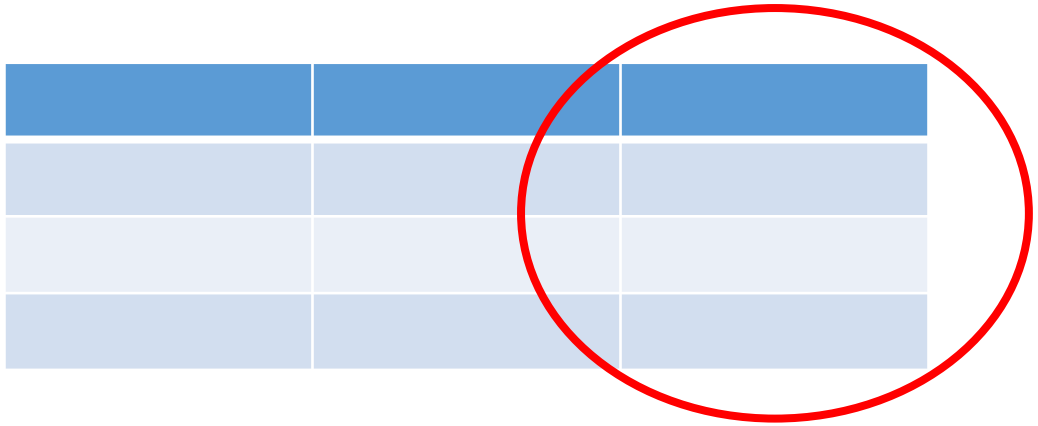
Zdroj : https://cs.wikipedia.org/wiki/Extensible_Markup_Language

Ukládaní v tabulkách

Co řádek tabulky to zvláštní skupina dat.



Jedna tabulka jedna skupina dat.



XML datový typ – řádková skupina dat I

Tabulka

id integer	typ varchar(80)	hodnota XML
1	hmotnost	<hmotnost> …
2	čas	<cas> …

Co řádek to jedna skupina dat

DDL vytvoření tabulky

```
create table jednotka
(id serial primary key,
 typ varchar(80) not null unique,
 hodnota xml not null);
```

Při konstrukci tabulky snažíme se obsloužit datovou integritu a obchodní pravidla

Příklad vloženého xml

```
<hmotnost>
  <jednotka>
        <nazev>kilogram</nazev><zkr>kg</zkr><koeficient>1</koeficient>
  </jednotka>
  <jednotka>
        <nazev>gram</nazev><zkr>gr</zkr><koeficient>0.001</koeficient>
  </jednotka>
  <jednotka>
        <nazev>mikrogram</nazev><zkr>µg</zkr><koeficient>0.000000001</k
oeficient>
  </jednotka>
  <jednotka>
        <nazev>tuna</nazev><zkr>t</zkr><koeficient>1000</koeficient>
  </jednotka>
 </hmotnost>
```

XML datový typ – řádková skupina dat II

```
select
array_to_string(xpath('/jednotka/nazev/text()'::text, radek),',') as nazev,
array_to_string(xpath('/jednotka/zkr/text()'::text, radek),',') as zkr,
array_to_string(xpath('/jednotka/koeficient/text()'::text, radek),',')::numeric as
koeficient
from
(select unnest(xpath('/cas/jednotka'::text,hodnota)) as radek
 from jednotka where typ = 'cas') a;
```

Výsledek výběru hodnot je jedno rozměrné pole. To se převede na string a ještě podle požadovaného vystupního typu.

Nejdříve se vybere řádek.
Pole hodnot se rozdělí do řádků

Nazev Text	Zkratka text	Koeficient numeric
sekunda	s	1
minuta	min	60
den (střední sluneční)	d	86400
sol (střední Mars)";"	sol	88775

XML datový typ – celá tabulka I

Všechny záznamy tvoří data stejného typu

weather_day

id	place	meteo_date	meteo_vals
1	Veveri	21.3.2016	{“meteo”:…
2	Praque	21.3.2016	{“meteo”:…

place,meteo_date jsou jedinečné

```
create table weather_day
(id serial primary key,
 place varchar(80) not null,
 meteo_date date not null,
 meteo_vals xml not null,
 unique(place,meteo_date));
```

```
'<meteo>
<hourly><time>9</time><temp>6</temp><feels>
3</feels><humidity>61</humidity></hourly>
<hourly><time>10</time><temp>7</temp><feels
>5</feels><humidity>55</humidity></hourly>
<hourly><time>11</time><temp>8</temp><feels
>5</feels><humidity>52</humidity></hourly> …
```

JSON datový typ

07_sql_json_datove_typy.sql

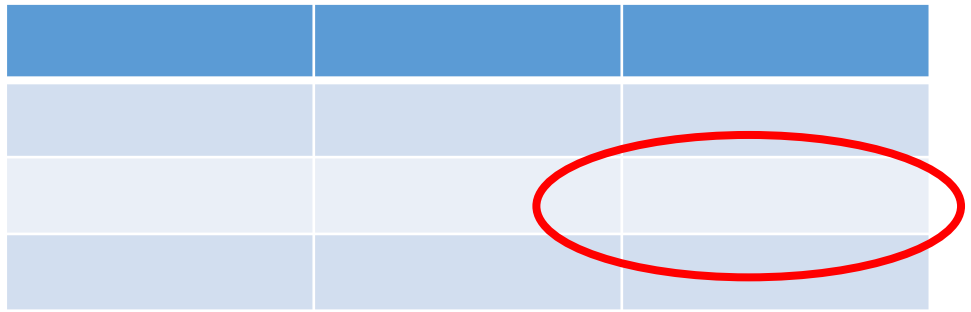
JavaScript Object Notation (*JavaScriptový objektový zápis*, **JSON**) je způsob zápisu dat (datový formát) nezávislý na počítačové platformě, určený pro přenos dat, která mohou být organizována v polích nebo agregována v objektech.

Zdroj : https://cs.wikipedia.org/wiki/JavaScript_Object_Notation

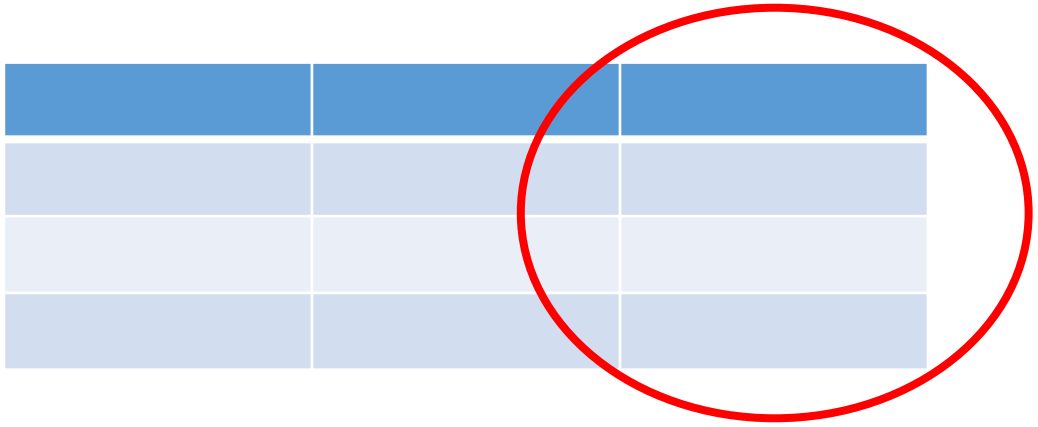
Manipulace a způsob použití je obdobný jako u XML

Ukládaní v tabulkách

Co řádek tabulky to zvláštní skupina dat.



Jedna tabulka jedna skupina dat.



JSON datový typ – řádková skupina dat I

Tabulka

id integer	typ varchar(80)	hodnota JSON
1	hmotnost	<hmotnost> …
2	čas	<cas> …

Co řádek to jedna skupina dat

DDL vytvoření tabulky

```
create table jednotka
(id serial primary key,
 typ varchar(80) not null unique,
 hodnota json not null);
```

Při konstrukci tabulky snažíme se obsloužit datovou integritu a obchodní pravidla

Příklad vloženého json

```
'{"hmotnost":[
        {"nazev":"kilogram","zkr":"kg","koeficient":1},
        {"nazev":"gram","zkr":"gr","koeficient":0.001},
        {"nazev":"mikrogram","zkr":"µg","koeficient":0.000000001},
        {"nazev":"tuna","zkr":"t","koeficient":1000}]}'::json
```

JSON datový typ – celá tabulka II

weather_day

id	place	meteo_date	meteo_vals
1	Veveri	21.3.2016	{“meteo”:…
2	Praque	21.3.2016	{“meteo”:…

Všechny záznamy tvoří data stejného typu

place,meteo_date jsou jedinečné

```
create table weather_day
(id serial primary key,
 place varchar(80) not null,
 meteo_date date not null,
 meteo_vals json not null,
 unique(place,meteo_date));
```

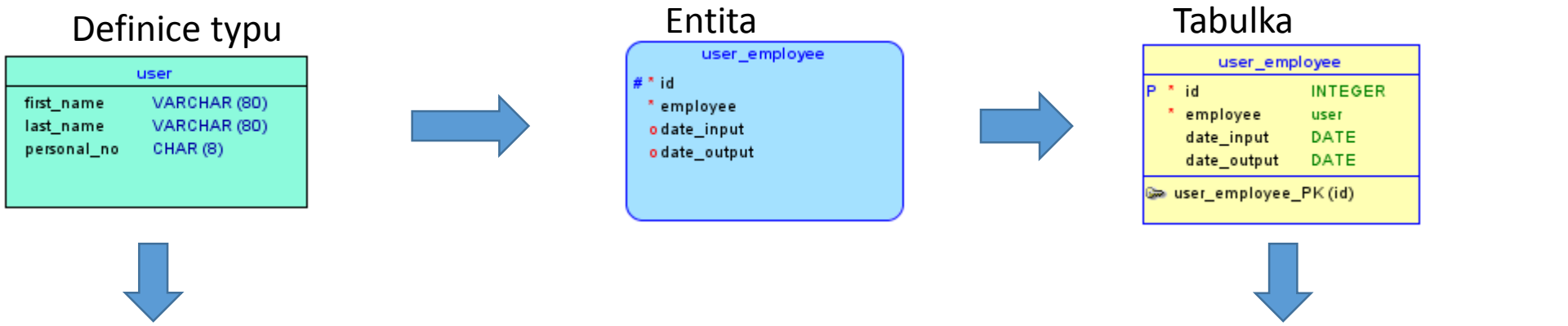
```
'{"meteo":[
{"time":9,"temp":6,"feels":5,"humidity":61},
{"time":10,"temp":7,"feels":5,"humidity":55},
{"time":11,"temp":8,"feels":5,"humidity":52},
{"time":12,"temp":8,"feels":5,"humidity":53},
{"time":13,"temp":8,"feels":6,"humidity":52},
{"time":14,"temp":9,"feels":6,"humidity":52},
{"time":15,"temp":9,"feels":6,"humidity":52},
{"time":16,"temp":8,"feels":6,"humidity":55}]}':
:JSON
```

Kompositní datový typ

08_sql_kompopzitni_datove_typy.sql

Kompositní datový typ reprezentuje strukturu řádku nebo záznamu. Základ tvoří seznam názvu polí a jejich datových typů.

Zdroj : http://www.postgresql.org/docs/9.3/static/rowtypes.html




```
create type t_user as
(first_name varchar(80),
 last_name varchar(80),
 personal_no char(8));
```

```
create table user_employee
(id serial primary key,
 employee t_user,
 date_input date,
 date_outup date);
```

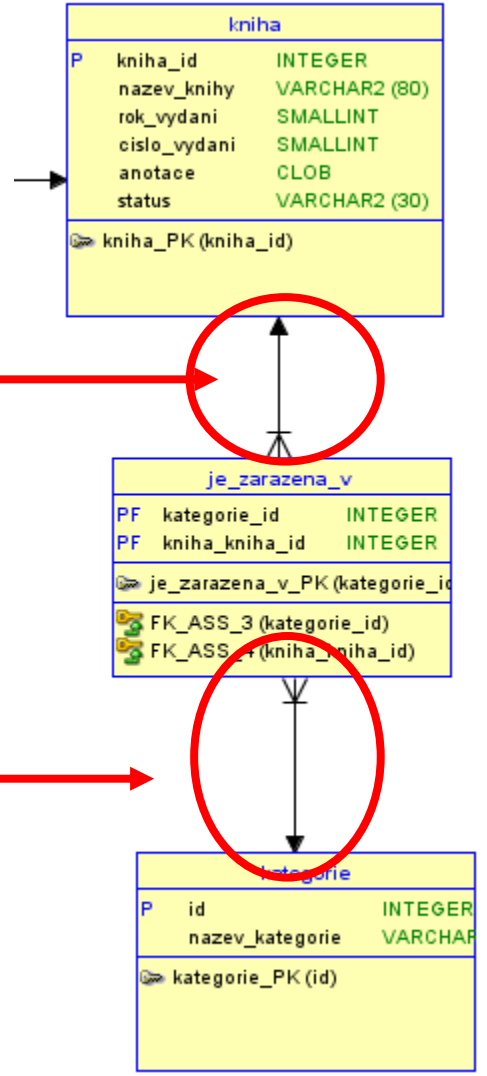
5. Cizí klíč (Foreign key)

09_sql_foreign_key.sql

Cizí klíč (Foreign key)

Je SQL konstrukt pomocí kterého zabezpečíme referenční integritu mezi tabulkami.

```
create table kniha_zarazeni
(kategorie_id integer not null references kategorie(id),
 kniha_id integer not null references kniha(id),
 primary key (kategorie_id,kniha_id)
);
```

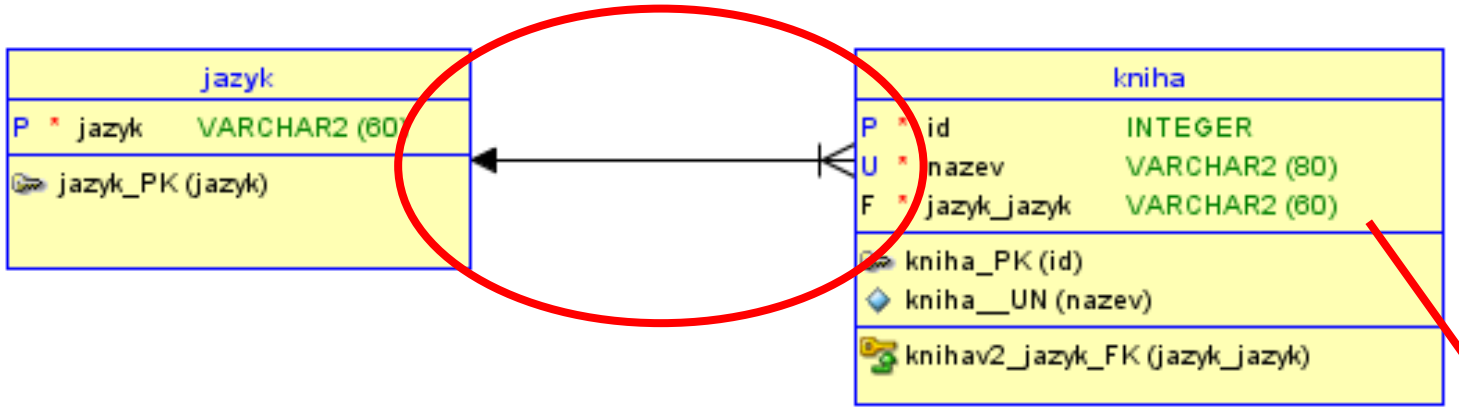


Cizí klíč (Foreign key) – pokročilejší

on update
on delete

- cascade — Kaskádní akce
- set null — Nastavení zrušené, aktualizované hodnoty na null
- set default — Nastavení zrušené, aktualizované hodnoty na default

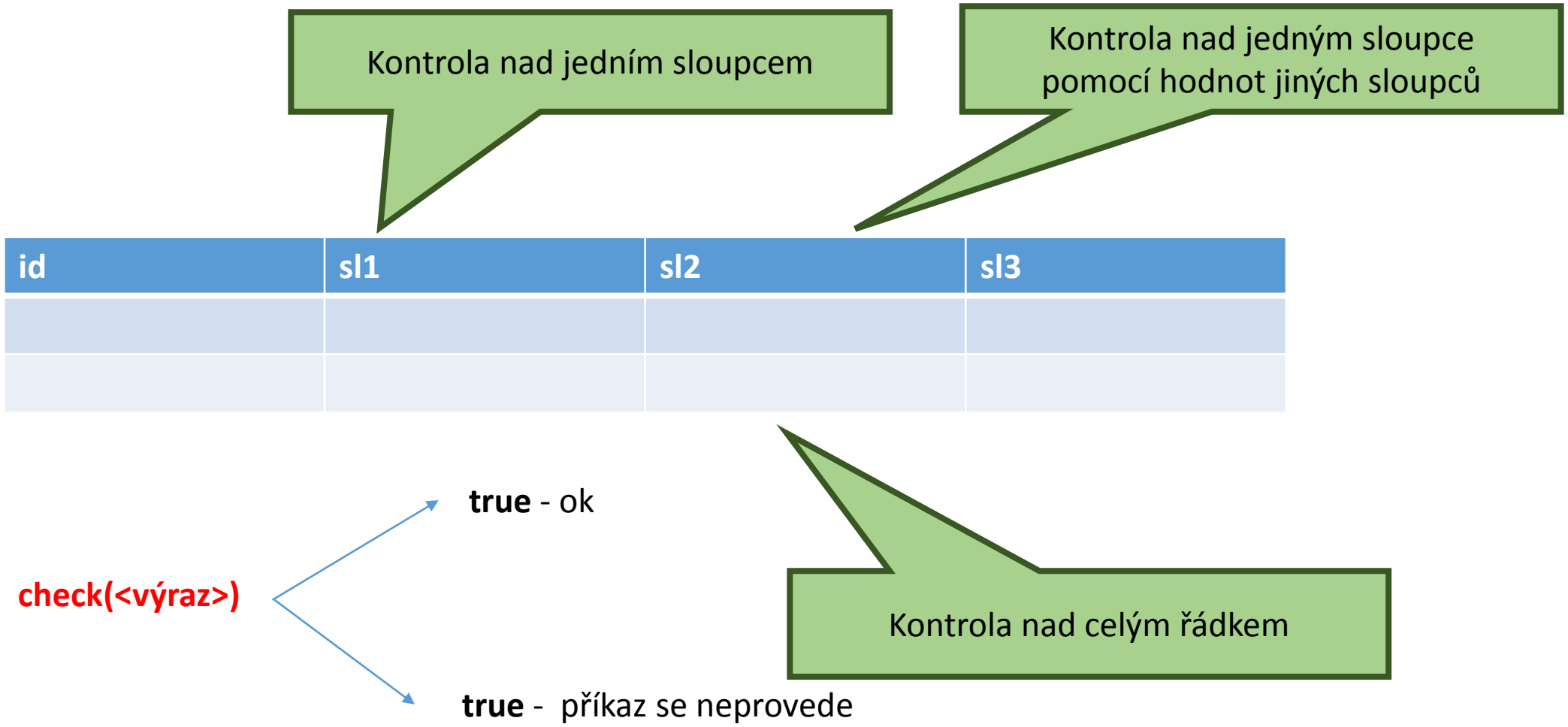
alter table kniha alter column jazyk set default 'Neurčeno';



Null hodnota umožňuje nezadat žádnou hodnotu

6. Constraint Check

Check – kontrola vstupních dat



7. Constraint Unique

unique – zabezpečení jedinečnosti dat

Unique nad jedným sloupcem

Unique nad více sloupci

```
create table color
(id serial primary key,
 color varchar(60) not null unique,
 rgb_1 smallint not null,
 rgb_2 smallint not null,
 rgb_3 smallint not null,
 unique(rgb_1,rgb_2,rgb_3));
```

Konec školení