# Constraints in SQL

Databázové praktikum

ZS 2015/2016

# Constraints in SQL

SQL umožňuje:

- obmedziť hodnoty v danom stĺpci
- pridať podmienku pre niekoľko stĺpcov zároveň
- zakázať NULL
- obmedziť duplicitu
- udržiavať referenčnú integritu medzi tabuľkami (cudzie kľúče)
- http://www.postgresql.org/docs/9.4/interactive/ddl-constraints.html

# CHECK

- umožňuje zúžiť dátové typy (napr. pridať rozsah)
  - `price numeric CHECK (price > 0)`
  - `price numeric CONSTRAINT positive_price CHECK (price > 0)`
- umožňuje pridať väzby medzi stĺpcami v jednej tabuľke
  - `CONSTRAINT valid_discount CHECK (price > discounted_price)`
- pomenovanie uľahčuje debugovanie a hlásenie chýb

# CHECK

```
CREATE TABLE products (
    product_no integer,
    name text,
    price numeric CHECK (price > 0),
    discounted_price numeric CHECK (discounted_price > 0),
    CHECK (price > discounted_price)
);
```

# NOT NULL

```
name text NOT NULL
```

je ekvivalentné špeciálnemu prípadu CHECK:

```
CHECK (name IS NOT NULL)
```

prvá verzia je rýchlejšia, ale constraint sa nedá pomenovať

# UNIQUE

Unikátne kľúče: v tabuľke nemôžu byť dva riadky s rovnakou hodnotou (resp. n-ticou v daných stĺpcoch)

- `product_no integer UNIQUE,`
- `product_no integer CONSTRAINT must_be_different UNIQUE,`
- 
```
CREATE TABLE products (
    product_no integer,
    name text,
    price numeric,
    UNIQUE (product_no)
);
```

# UNIQUE

Adding a unique constraint will automatically create a unique btree index on the column or group of columns used in the constraint.

A uniqueness constraint on only some rows can be enforced by creating a partial index.

`http://www.postgresql.org/docs/9.4/interactive/indexes-partial.html`

# PRIMARY KEY

- jediný primárny kľúč pre jednu tabuľku
- užitočné pre klientske aplikácie (napr. pri UPDATE treba jednoznačne identifikovať, ktorý riadok sa mení)
- príklady:

```
product_no integer PRIMARY KEY
product_no integer UNIQUE NOT NULL
PRIMARY KEY (a, c)
```

# FOREIGN KEY

- špecifikuje, že hodnota v danom stĺpci (prípadne skupine stĺpcov) je odkazom na existujúcu hodnotu v nejakej inej tabuľke
- inak povedané, zabezpečuje referenčnú integritu
- vhodné použiť pri dekompozícii veľkej relácie na malé
- príklad:

```
CREATE TABLE t1 (
    a integer PRIMARY KEY, b integer, c integer,
    FOREIGN KEY (b, c) REFERENCES other_table (c1, c2)
);
```

# FOREIGN KEY

```
CREATE TABLE products (
    product_no integer PRIMARY KEY,
    name text, price numeric);
CREATE TABLE orders (
    order_id integer PRIMARY KEY);
CREATE TABLE order_items (
    product_no integer REFERENCES products ON DELETE RESTRICT,
    order_id integer REFERENCES orders ON DELETE CASCADE,
    quantity integer, PRIMARY KEY (product_no, order_id));
```

# FOREIGN KEY

ON DELETE

CASCADE -- zmaže riadok, ktorý sa odkazoval na mazaný riadok

RESTRICT -- nezmaže nič, operácia DELETE skončí s chybou

SET NULL -- nastaví odkaz na NULL (zlyhá pre NOT NULL stĺpce)

SET DEFAULT -- nastaví hodnotu odkazu na defaultnú
(zlyhá, ak takto vznikne nekorektný odkaz)

NO ACTION -- toto je default, DELETE skončí s chybou

# FOREIGN KEY

```
ON UPDATE
    CASCADE -- zmení odkazujúcu hodnotu na novú hodnotu odkazovanej
               (zachová väzbu medzi riadkami)
    RESTRICT -- zabráni zmene v odkazovanej tabuľke, UPDATE zlyhá
    SET NULL -- nastaví odkaz na NULL (väzba zanikne)
    SET DEFAULT -- nastaví hodnotu odkazu na defaultnú
                   (zlyhá, ak takto vznikne nekorektný odkaz)
    NO ACTION -- toto je default, väzba sa pokazí
```

# FOREIGN KEY

- kedy sa vyhodnocuje platnosť odkazu?

```
DEFERRED -- až pri commite transakcie
IMMEDIATE -- okamžite po vykonaní operácie
```

- možnosti pri definícii cudzieho kľúča:

```
DEFERRABLE INITIALLY IMMEDIATE -- v rámci transakcie
     možno zvoliť, kedy vyhodnocovať, default je ihneď
DEFERRABLE INITIALLY DEFERRED -- v rámci transakcie
     možno zvoliť, kedy vyhodnocovať, default je commit
NOT DEFERRABLE -- vždy okamžité vyhodnocovanie
```

http://www.postgresql.org/docs/9.4/static/sql-set-constraints.html

# GRANT a REVOKE

- GRANT slúži na prideľovanie oprávnení, REVOKE na odoberanie
- http://www.postgresql.org/docs/9.4/static/sql-grant.html
- http://www.postgresql.org/docs/9.4/static/sql-revoke.html
- užitočné príkazy v psql:
  - `\du`
  - `\z <tablename>`
  - `SELECT grantee, privilege_type FROM information_schema.role_table_grants WHERE table_name='test';`