5. Duomenų bazės sukūrimas ir užpildymas duomenimis

5.1 Duomenų bazės kūrimas

- Naujas bendrąsias DB kuria sistemos administratorius.
- Lokalias DB gali kurti darbo stoties vartotojas – administratorius.
- DB kuriama: kompiuterio diske, disko kataloge, atminties srityje ar atskirose bylose - **DB įrenginyje** (angl. *database device*).
- SQL standarte nėra sakinių DB kurti.

Daugelyje DBVS yra tarnybinė programa

CREATE DATABASE <DB vardas> [<įrenginys>]

CREATE DATABASE *Darbai*

CREATE DATABASE *Darbai* **ON** D:

DROP DATABASE *Darbai*

5.2. Duomenų tipai

Kuriant lentelę reikia išvardyti jos stulpelius, kiekvienam kurių būtina nurodyti galimų reikšmių aibę – **stulpelio tipą**.

Visų pirma stulpelio duomenų tipui priskiriama **duomenų rūšis:**

- Simboliniai duomenys
- Dvejetainiai duomenys
- Skaičiai
- Datos
- Laikas

Simboliniai duomenys (angl. *character datatypes*).

Simbolių eilutė turi simbolių skaičiaus atributą – ilgį. Fiksuoto ilgio simbolių eilutės:

CHAR(n) – iki 254 baitų,

Kintamo ilgio:

VARCHAR(n) – iki 32672 (4000) baitų ilgio,
CLOB(n[K|M|G]) – iki 2 GB (4 GB).

Kai n > 254 užklausose negalima naudoti:
DISTINCT, **GROUP BY**, **ORDER BY**

Skaitiniai duomenys (*number datatypes*).

SMALLINT – sveikieji skaičiai [-32 768; 32 767].
INTEGER – „dideli“ sveikieji skaičiai (4 baitai) [-2 147 483 648; 2 147 483 647].
BIGINT – „ypač dideli“ sveikieji skaičiai (8 b)
REAL – slankiojo kablelio skaičiai, 32 bitai, pvz., -2E5, 5.555E-18, -.655645e8.
FLOAT (**DOUBLE**) – dvigubo tikslumo skaičiai (64 b)
DECIMAL (n, m) (**NUMERIC**) – dešimtainiai supakuoti skaičiai – iki 1000 (31) dešimtainių skaitmenų.

Dvejetainių duomenų tipai (*binary datatypes*).

Fiksuoto ilgio:
BIT(n)

Kintamojo ilgio:
BIT VARYING(n)
BLOB(n[K|M|G])

PostgreSQL: **bytea**

Datos ir laiko duomenys.

DATE – (atmintyje – 4 baitai)
TIME – (3 baitai, PostgreSQL – 8 baitai)
TIMESTAMP –(10 baitų, PostgreSQL – 8 baitai)

DATE ir **TIME** vaizdavimas priklauso nuo terpės (*locale*),
'2005.01.01', '12:00:00',
'2005-01-01-12.15.55.330000'

DBVS visada „supranta“ ISO datą ir laiką:
'2005-01-01', '12:00:00'

Visi SQL duomenų tipai turi ypatingą reikšmę **NULL**.

Parinkus stulpeliui duomenų tipą, reikia parinkti

- ilgio charakteristiką ir
- tikslumą.

5.3. Duomenų tipų derinimas

Atliekant operacijas su keliais argumentais tenka **derinti duomenų tipus**, pvz.,

2 + 3.1E-5

Automatinis tipų suderinamumas ne visada yra tikslingas ir įmanomas.

Vykdytojai.Kategorija || ' kategorija'

– klaidingas reiškinys

Vieno tipo reikšmės transformavimas į kito tipo reikšmę: **CAST** (<reiškinys> **AS** <duomenų tipas>)

```
SELECT Pavardė,
    CAST(Kategorija AS CHAR(1)) || ' kategorija'
        AS Kategorija
FROM Vykdytojai
WHERE Vykdytojai.Kvalifikacija = 'Informatikas'
```

Pavardė	*Kategorija*
Jonaitis	2 kategorija
Antanaitis	3 kategorija

Visų vykdytojų kategorijų vidurkis (2,8):

```
SELECT AVG(Kategorija) AS "Kategorijų vidurkis"
FROM Vykdytojai
```

Kategorijų vidurkis
2

CAST(**AVG**(*Kategorija*) **AS FLOAT**) – 2.00000000

Vidurkis dešimtainiu skaičiumi **DECIMAL**(5,2):

```
SELECT CAST(AVG(CAST(Kategorija AS FLOAT))
    AS DECIMAL(5,2)) AS "Kategorijų vidurkis"
FROM Vykdytojai
```

PostgreSQL: **AVG** rezultatas **NUMERIC** arba **FLOAT**

CAST leidžia ne bet kokį tipų transformavimą.

Tipų reikšmes galima transformuoti ir skaliarinėmis funkcijomis:

```
SELECT DECIMAL(AVG(FLOAT(Kategorija)), 5, 2))
        AS "Kategorijų vidurkis"
FROM  Vykdytojai
```

Kai standartinės galimybės netenkina, galima apibrėžti savas tipų transformavimo funkcijas.

5.4. Lentelių apibrėžimas

Lentelės (struktūros) apibrėžimas – „aktyvus“ veiksmas.

Lentelės kuriamos SQL DDL sakiniu

CREATE TABLE

Kuriant (apibrėžiant) lentelę būtinai nurodoma:

- lentelės **vardas**, galima patikslinti schema
- lentelės **stulpelių vardai** ir jų **tipai**.


```
CREATE TABLE Vykdytojai (
  Nr           INTEGER  NOT NULL,
  Pavardė      CHAR(30)  NOT NULL,
  Kvalifikacija CHAR(16)
               DEFAULT 'Informatikas',
  Kategorija   SMALLINT,
  Išsilavinimas CHAR(10) )
```

NOT NULL - stulpelis negali įgyti **NULL** reikšmės.
DEFAULT - numatytoji reikšmė.


```
CREATE TABLE Projektai (
  Nr           INTEGER        NOT NULL,
  Pavadinimas VARCHAR(254)   NOT NULL,
  Svarba      CHAR (10)  DEFAULT 'Vidutinė',
  Pradžia     DATE,
  Trukmė      SMALLINT)
CREATE TABLE Vykdymas (
  Projektas    INTEGER      NOT NULL,
  Vykdytojas   INTEGER      NOT NULL,
  Statusas     VARCHAR(32) DEFAULT 'Programuotojas',
  Valandos     SMALLINT)
```


Daugumą lentelės savybių galima nurodyti vėliau:

```
ALTER TABLE Vykdytojai ADD Gimtadienis DATE

ALTER TABLE Vykdytojai DROP Gimtadienis

ALTER TABLE Projektai
          ALTER Svarba DROP DEFAULT

ALTER TABLE Projektai
          ALTER Svarba SET DEFAULT ' Didelė'
```

5.5. Naujų duomenų įvedimas

1-oji forma:

INSERT INTO <lentelės vardas>
[(<stulpelio vardas> {, <stulpelio vardas>})]
VALUES (<reikšmė> {,<reikšmė>})
{, (<reikšmė> {,<reikšmė>})**}**

<reikšmė> ::= <reiškinys> | **NULL** | **DEFAULT**

INSERT INTO *Vykdytojai*
(*Nr, Pavardė, Kvalifikacija, Kategorija,Išsilavinimas*)
VALUES (6, 'Baltakis', 'Informatikas', 2, **NULL**),

INSERT INTO *Vykdytojai*
VALUES (6, 'Baltakis', 'Informatikas', 2, **NULL**)

INSERT INTO *Vykdytojai*
VALUES ('Baltakis', 6, 'Informatikas', 2, **NULL**)
– **klaida**

INSERT INTO *Vykdytojai*
VALUES (6, 'Informatikas', 'Baltakis', 2, **NULL**)
– **klaida?**

INSERT INTO *Vykdytojai* (*Nr, Pavardė, Kategorija*)
VALUES (6, 'Baltakis', 2)

2-oji forma:

INSERT INTO <lentelės vardas>
[(<stulpelio vardas> {, <stulpelio vardas>})]
<užklausa>

Eilutės apie seniai pasibaigusius projektus į kt. lentelę:

INSERT INTO *Seni_Projektai*
SELECT * **FROM** *Projektai*
WHERE *Pradžia* +
CAST(*Trukmė* || 'MONTHS' **AS INTERVAL**) <
CURRENT_DATE – **INTERVAL** '1 **YEAR**'

Duomenys į lentelę gali būti įvedami ir tarnybinėmis programomis.

Duomenų importas – duomenų iš failo įvedimas į lentelę.

Duomenų eksportas – užklausos rezultato išsaugojimas faile.

5.6. Duomenų šalinimas

DELETE FROM <lentelės vardas>
[**WHERE** <paieškos sąlyga>]

Visos (!) tenkinančios paieškos sąlygą eilutės šalinamos.

DELETE FROM *Vykdymas*

– šalinamos visos buvusios lentelėje *Vykdymas* eilutės.

Perkeltos į „archyvą“ eilutės pašalinamos iš pradinės lentelės:

DELETE FROM *Projektai*
WHERE *Pradžia* +
CAST(*Trukmė* || 'MONTHS' **AS INTERVAL**) <
CURRENT_DATE – **INTERVAL** '1 **YEAR**'

Baltakiui išėjus iš darbo:

DELETE FROM *Vykdymas*
WHERE *Vykdytojas* = (**SELECT** *Nr* **FROM** *Vykdytojai*
WHERE *Pavardė* = 'Baltakis')

DELETE FROM *Vykdytojai*
WHERE *Pavardė* = 'Baltakis'

Prieš vykdant sakinį

```
DELETE FROM <lentelės vardas>
WHERE <paieškos sąlyga>
```

rekomenduojama įvykdyti

```
SELECT * FROM <lentelės vardas>
WHERE <paieškos sąlyga>
```


5.7. Esamų duomenų atnaujinimas

```
UPDATE  <lentelės vardas>
SET     <stulpelio vardas> = <reiškinys>
      {,<stulpelio vardas> = <reiškinys>}
[WHERE <paieškos sąlyga>]
```

Atnaujinamos visos (!) eilutės, tenkinančios paieškos sąlygą.

Gražulytė baigė VU ir dėl to jai keliama kategorija 1:

```
UPDATE  Vykdytojai
SET     Išsilavinimas = 'VU',
        Kategorija = Kategorija + 1
WHERE   Pavardė = 'Gražulytė'
```


Visų projektų, kuriuose dalyvauja Baltakis, trukmės pratęsimas:

```
UPDATE  Projektai SET Trukmė = Trukmė * 1.1
WHERE Projektai.Nr IN
  (SELECT Projektas FROM Vykdymas, Vykdytojai
   WHERE  Vykdytojas = Vykdytojai.Nr  AND
          Pavardė = 'Baltakis')
```


Padidinti kategoriją visiems, dalyvaujantiems bent 2 projektuose:

```
UPDATE  Vykdytojai SET Kategorija = Kategorija+1
WHERE  (SELECT COUNT(*)
     FROM Vykdymas WHERE Vykdytojas = Nr) >= 2
```

Čia vidinė užklausa – priklausomoji.

Atnaujinimas su paieškos sąlyga be parametro

```
UPDATE  Vykdytojai
SET  Kategorija = Kategorija + 1
WHERE Nr IN (SELECT Vykdytojas
             FROM   Vykdymas
             GROUP BY Vykdytojas
             HAVING COUNT(*) >= 2)
```


5.8. Lentelių ir DB šalinimas

```
DROP TABLE <lentelės vardas>
DROP TABLE Vykdymas
DROP DATABASE  Darbai
```