Darbas su duomenimis

2. Duomenų bazės sukūrimas

Prisijunkite prie MySQL

- Select now(); Ši komanda turi grąžinti sistemos laiką.

MySQL duomenų tipai
Tekstinė informacija

- Char(20) – fiksuotas ilgis atmintyje (0-255 bytes)
- Varchar(20) – fiksuotas naudojamų simbolių skaičius
- Duomenų bazių galimas koduotes galima pažiūrėti su komanda SHOW CHARACTER SET;
- varchar(20) character set latin1 – komanda leidžia pakeisti stulpelio koduotę.
- create database european_sales character set latin1; nustato visos duomenų bazės koduotę.

Text type	Maximum number of bytes
`tinytext`	255
`text`	65,535
`mediumtext`	16,777,215
`longtext`	4,294,967,295

MySQL duomenų tipai
Skaitinė informacija

Float
P – precission – leistinas visų skaitmenų skaičius
S – scale – leistinas skaitmenų skaičius po kablelio

Type	Signed range	Unsigned range
`tinyint`	−128 to 127	0 to 255
`smallint`	−32,768 to 32,767	0 to 65,535
`mediumint`	−8,388,608 to 8,388,607	0 to 16,777,215
`int`	−2,147,483,648 to 2,147,483,647	0 to 4,294,967,295
`bigint`	−2^63 to 2^63 - 1	0 to 2^64 - 1

Type	Numeric range
`float(` *p* `,` *s* `)`	−3.402823466E+38 to −1.175494351E-38 and 1.175494351E-38 to 3.402823466E+38
`double(` *p* `,` *s* `)`	−1.7976931348623157E+308 to −2.2250738585072014E-308 and 2.2250738585072014E-308 to 1.7976931348623157E+308

MySQL duomenų tipai
Temporal (datos ir laikas) informacija

Component	Definition	Range
YYYY	Year, including century	`1000` to `9999`
MM	Month	`01` (January) to `12` (December)
DD	Day	`01` to `31`
HH	Hour	`00` to `23`
HHH	Hours (elapsed)	`-838` to `838`
MI	Minute	`00` to `59`
SS	Second	`00` to `59`

Type	Default format	Allowable values
`date`	YYYY-MM-DD	`1000-01-01` to `9999-12-31`
`datetime`	YYYY-MM-DD HH:MI:SS	`1000-01-01 00:00:00.000000` to `9999-12-31 23:59:59.999999`
`timestamp`	YYYY-MM-DD HH:MI:SS	`1970-01-01 00:00:00.000000` to `2038-01-18 22:14:07.999999`
`year`	YYYY	`1901` to `2155`
`time`	HHH:MI:SS	`-838:59:59.000000` to `838:59:59.000000`

Pavyzdys
Sukurkime SQL schema - Person

```
CREATE TABLE person
 (person_id SMALLINT UNSIGNED,
  fname VARCHAR(20),
  lname VARCHAR(20),
  eye_color ENUM('BR','BL','GR'), -- stulpelio reikšmės turi būti iš nurodyto sąrašo
  birth_date DATE,
  street VARCHAR(30),
  city VARCHAR(20),
  state VARCHAR(20),
  country VARCHAR(20),
  postal_code VARCHAR(20),
  CONSTRAINT pk_person PRIMARY KEY (person_id) -- nurodome primary key stulpelį
 );
```

Pavyzdys
Sukurkime SQL schema – Favorite food

```
CREATE TABLE favorite_food
(person_id SMALLINT UNSIGNED,
food VARCHAR(20),
-- nurodome primary key - sudaro du stulpeliai
CONSTRAINT pk_favorite_food PRIMARY KEY (person_id, food),
-- nurodome, kad person_id yra ir foreign key - kada naudinga ši sąlyga?
CONSTRAINT fk_fav_food_person_id FOREIGN KEY (person_id)
REFERENCES person (person_id)
);
```

Pavyzdys
Patikriname sukurtas lenteles

desc person;

Kas yra null reikšmė?

- Not applicable – neaišku
- Unknown – nežinoma reikšmė
- Empty set – tuščia aibė

Pavyzdys
Generuokime Primary key automatiškai

```
ALTER TABLE person MODIFY person_id SMALLINT UNSIGNED
AUTO_INCREMENT;


set foreign_key_checks=0;
ALTER TABLE person
    MODIFY person_id SMALLINT UNSIGNED AUTO_INCREMENT;
set foreign_key_checks=1;
```

Pavyzdys
Duomenų įkėlimas

```
INSERT INTO person
 (person_id, fname, lname, eye_color, birth_date)
VALUES (null, 'William','Turner', 'BR', '1972-05-27');
```

Pavyzdys
Duomenų peržiūra

```
SELECT person_id, fname, lname, birth_date FROM person;
```

Pavyzdys
Duomenų įkėlimas

```
INSERT INTO favorite_food (person_id, food) VALUES (1, 'pizza');
```

Pavyzdys
Atnaujinkime duomenis

```
UPDATE person
SET street = '1225 Tremont St.',
city = 'Boston',
state = 'MA',
country = 'USA',
postal_code = '02138'
WHERE person_id = 1;
```

Pavyzdys
Ištrinkime duomenis

```
DELETE FROM person WHERE person_id = 2;
```

Užduotis
Lentelės kūrimas susitikimo metu

- Sukurkime duomenų modelį Picerija
  - Kokias lenteles naudosime?
  - Kokius laukus turės lentelės?
  - Kokius stulpelių duomenų tipus naudosime?
  - Kokias reikšmes leisime naudoti stulpeliuose?
- Sukurkime duomenų bazę picerija su suskurtomis lentelėmis
- Įdėkite po 5 eilutes į kiekvieną lentelę

Užduotis
Normalizuokime sukurtas lenteles

Normalizacija užtikrina, kad lentelėje nėra besidubliuojančios informacijos.

Normalization is the process of ensuring that there are no duplicate (other than foreign keys) or compound columns (address fields) in your database design.

Normalizuokime iki

- 1NF
- 2NF
- 3NF

Užduotis

1. Sukurkime duomenų bazę picerija_3NF
2. Sukurkime normalizuotas lenteles
3. Įkelkite po 5 eilutes
4. Palyginkte picerija_3NF ir picerija duomenų bazių schemas

Kas blogo gali nutikti dedant duomenis į lenteles?

- Neunikalus Piramary key
- Neegzistuojantis Foreign key
- Stulpelio duomenų tipo kitoks nei vedama reikšmė
- Neteisingas datos konvertavimas

Mes naudosime Sakila duomenų bazę

- Show tables;
- Desc customer;
- Select * from actor;

Klausimai:

1. Ar Sakila duomenų bazė tenkina normaliąsias formas?
2. Peržiūrėkite lentelių schemas:
   - Ar raktai yra automatiškai inkrementuojami?
   - Ar yra raktų, kuriuos sudaro keli stulpeliai?
   - Kokie duomenų tipai yra naudojami?