MySQL duomenų bazė

MySQL DB parsisiuntimas ir instaliavimas

Parsisiųskime MySQL DB (naujausią versiją)

- Pasirinkime ir parsisiųskime MySQL Community Server:
- https://dev.mysql.com/downloads/

Pasirenkame MySQL Installer (su visais priedais (MySQL Workbench)

MySQL connector‘iaus parsisiuntimas

Parsisiųskime MySQL connectorių

https://dev.mysql.com/downloads/connector/j/

Jei jis nebuvo instaliuotas kartu su MySQL duomenų baze

Sukurkime papildomą vartotoją

Įsijunkime MySQL Workbench ir sukurkime naują vartotoją.

Naują vartotoją galime padaryti žemiau pateiktomis komandomis

```
CREATE USER 'vartotojas'@'localhost' IDENTIFIED BY 'slaptazodis';
GRANT ALL PRIVILEGES ON * . * TO 'vartotojas'@'localhost';
```

Užduotis

Sukurkime naują DB (schemą).

Tuomet sukurkime naują lentelę.

Lentelės kūrimas

Lentelės gali būti sukurtos naudojant SQL užklausą CREATE TABLE.

Pvz.:

```
CREATE TABLE IF NOT EXISTS `pirkejai` (
    `pirkejo_nr` int(10) unsigned NOT NULL AUTO_INCREMENT,
    `vardas` varchar(40) COLLATE utf8_bin NOT NULL,
    `pavarde` varchar(40) COLLATE utf8_bin NOT NULL,
    `nuolaida` tinyint(4) NOT NULL DEFAULT '0',
    PRIMARY KEY (`pirkejo_nr`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin AUTO_INCREMENT=1 ;
```

Stulpelių tipai

MySQL turi keturis pagrindinius stulpelių tipus: skaitmeninius, teksto eilutės, datos ir specialusis.

Nors yra daugiau specializuotų tipų, tačiau kiekvieną iš jų galima priskirti vienam iš keturių pagrindinių tipų.

Turėtumėte pasirinkti mažiausią galimą stulpelio tipą, kadangi taip bus sutaupomas laikas bei priėjimas prie lentelės ir jos papildymas vyks žymiai sparčiau

Tačiau jei bus pasirinktas per mažas laukelio dydis, įvedimo metu duomenys gali būti iš viso neįvesti arba įvesta tik jų dalis.

Skaitmeninių stulpelių tipai

Skaitmeninis tipas skirtas bet kurio tipo skaitmeniniams duomenims įvesti, pvz., kainai, amžiui ar kiekiui.

Yra du pagrindiniai skaitmeninių duomenų tipai: sveikieji (ne dešimtainiai ir netrupmeniniai skaičiai) ir slankiojo kablelio.

Visiems skaitmeniniams tipams galimos dvi parinktys: UNSIGNED ir ZEROFILL.

UNSIGNED (teigiamas skaičius) neleidžia ivesti neigiamų skaičių.

ZEROFILL užpildo reikšmę nuliais vietoj įprastinių tarpų ir kartu suteikia jai tipą UNSIGNED.

Skaitmeninių stulpelių tipai

Pasirinkite patį mažiausią tinkamą tipą (TINYINT vietoj INT. jei reikšmė nebus didesnė kaip 127).

Sveikiesiems skaičiams parinkite sveiką tipą (prisiminkite, kad pinigai gali būti sveiki skaičiai, pvz., daug dažniau jie gali būti išreiškiami ne litais, o centais, t.y. sveikaisiais skaičiais). Tačiau jei racionalu, galima naudoti **DECIMAL** tipą.

Jei reikalingas ypatingas tikslumas, vietoj slankiojo kablelio tipo naudokite sveikąjį (apvalinimo klaidos pakeičia slankiojo kablelio skaičių reikšmes).

Skaitmeninių stulpelių tipai

TINYINT - mažas sveikas skaičius -128 iki 127 (SIGNED), 0 iki 255 (UNSIGNED), užima 1 baitą.

BOOL - TINYINT(1) sinonimas

SMALLINT - Mažas svekasis skaičius -32,768 iki 32,767 (SIGNED); 0 iki 65,535 (UNSIGNED), užima 2 baitus.

MEDIUMINT - Vidutinio dydžio sveikas skaičius -8,388,608 iki 8,388,607 (SIGNED); 0 iki 16,777,215 (UNSIGNED), užima 3 baitus.

INT - Sveikas skaičius -2,147,483,648 iki 2,147,483,647 (SIGNED); 0 iki 4,294,967,295 (UNSIGNED), užima 4 baitus.

BIGINT - Didelis sveikas skaičius, -9,223,372,036,854,775,808 iki 9,223,372,036,854,775,807 (SIGNED); 0 iki 18,446,744,073,709,551,615 (UNSIGNED); užima 8 baitus.

Skaitmeninių stulpelių tipai

FLOAT [(precision)] - Slankiojo kablelio skaičius. Precision (tikslumas) < =24, M,D)] - Dvigubo tikslumo slankiojo kablelio skaičius. Nuo 1skirtas paprasto tikslumo slankiojo kablelio skaičiui. precision tarp 25 iki 53 skirtas dvigubo tikslumo slankiojo kablelio skaičiui. FLOAT (X) turi tokią pačią reikšmių sritį kaip FLOAT ir DOUBLE tipai, bet bendras išvedamų pozicijų ir dešimtainių pozicijų skaičius neapibrėžtas.

DOUBLE[(.7976931348623157E+308 iki 2.2250738585072014E-308. M nurodo bendrą išvedamų pozicijų skaičių, D nurodo dešimtainių pozicijų skaičių.

DECIMAL[(M[, D])] - Mažas dešimtainis skaičius saugomas kaip eilutė, vienas baitas vienam ženklui (tai vadinama *nesuglaudintu tipu* - visi kiti skaitmeniniai tipai suglaudinti). M nurodo bendrą skaitmenų skaičių (įskaitant ženklą ir kableli dešimtainei daliai). D nurodo dešimtainių pozicijų skaičių ir turi visuomet būti mažesnis už M.

Eilučių stulpelių tipai

CHAR (M) - simbolis. Fiksuoto ilgio eilutė, iš dešinės pusės užpildyta tarpų seka, atitinkančia eilutės ilgi. Galimas ilgis nuo 0 iki 255 simbolių.

VARCHAR(M) - Kintamo ilgio simbolių eilutė, kurioje tarpų seka panaikinama, kai reikšmė įrašoma saugojimui. Nuo 0 iki 255 simbolių.

TINYTEXT - Ne daugiau 255 simbolių (2^8 - 1) + 1 baitas laukelio dydžio užrašymui. Dažniausiai patartina naudoti VARCHAR, nes darbas vyks sparčiau.

TEXT - Ne daugiau 65,535 simbolių (2^16 - 1) + 2 baitai laukelio dydžio užrašymui.

MEDIUMTEXT - ne daugiau 16,777,215 simbolių (2^24 -1) + 3 baitai laukelio dydžio užrašymui.

LONGTEXT- ne daugiau 4,294,967,295 simbolių (2^32 -1)+ 4 baitai laukelio dydžio užrašymui. Nepriklausomai nuo sistemos išteklių, ne didesnis kaip 16 MB vienoje lentelės eilutėje.

Char duomenų tipas

I	A	N							
V	I	N	C	E	N	T			
M	I	R	I	A	M				

Paveiksle atvaizduoti simboliai, įrašyti mažos lentelės viename lauke. Pirmojo lauko dydis yra CHAR (10).

Kiekvienas įrašas užima 10 baitą Jei laukas mažesnis nei 10 baitų, likusios pozicijos užpildomos atitinkamu tarpų skaičiumi.

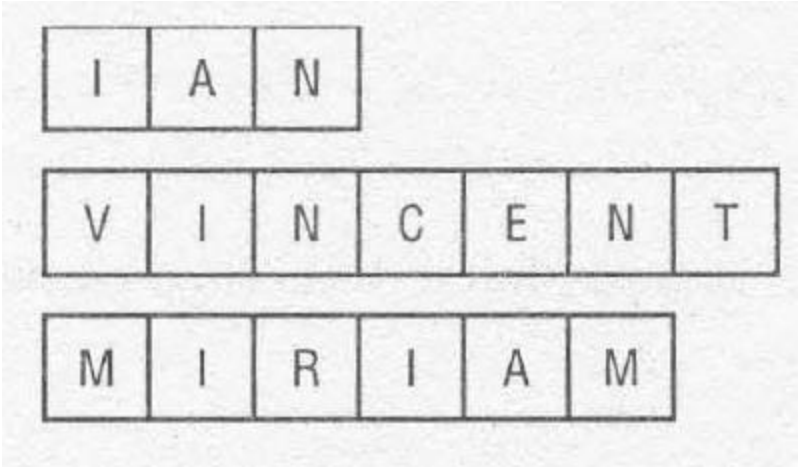
Char duomenų tipas

Char duomenų tipas pasižymi šiais bruožais:

- Yra labai sparčios (kadangi MySQL "žino", kad tolesnis laukas prasideda nuo 11 pozicijos) ;
- Lengvai įrašomos į operatyviąją atmintinę;
- Lentelės lengvai atkuriamos avarijos atveju (kadangi įrašų pozicijų skaičius fiksuotas, MySQL žino, kur yra kiekvienas įrašas, todėl bus prarastas tik avarijos metu įvedamas įrašas.
- Užima daugiau vietos diske (trys įrašai užims 30 simbolių erdvę, nors tie trys vardai, kartu sudėjus, turi tik 16 simbolių);

Varchar duomenų tipas

Laukai VarChar duomenų tipe yra skirtingo ilgio. Jie bus saugomi taip:



Nors šis duomenų formatas užima mažiau vietos, jis sudėtingesnis. Kiekvienas įrašas turi antraštę, nurodančią jo ilgį.

Varchar duomenų tipas

Varchar duomenų tipas pasižymi šiais bruožais:

- Visi eilutės laukai yra dinaminiai (nebent jie mažesni kaip 4 baitai. Bet šiuo atveju nebus sutaupoma labai daug vietos, o dėl papildomos painiavos gali būti prarasti duomenys).
- Paprastai užima mažiau vietos diske nei fiksuotos lentelės.
- Lentelėms reikalinga nuolatinė priežiūra, kad būtų išvengta fragmentacijos. (Pavyzdžiui, jei pakeisite *Jan* į *Jane, "e"* ne iš karto atsiras po *"n",* kadangi ši vieta iš pradžių bus užimta kito langelio ar įrašo.)

Varchar duomenų tipas

Varchar duomenų tipas pasižymi šiais bruožais:

- Avarijos atveju lentelės atkūrimas nėra paprastas, ypač jei ji labai fragmentuota.
- Kiekvienas įrašas turi antraštę, nurodančią, kurie eilutės laukai tušti ir kuriuose skaitmeninio formato laukuose įvestas nulis (bet ne NULL tipo įrašas). Šiuo atveju jie nesaugomi diske. Netuščios eilutės ilgis lygus baitui ir eilutės turiniui.
- Jei laukai fragmentuoti, kiekvienas naujas ryšys papildomai sunaudos 6 baitus ir jo dydis bus ne mažesnis kaip 20 baitų (be to, šie laukai gali turėti savų ryšių, kurie užims dar daugiau vietos).

Eilučių stulpelių tipai

ENUM('vale1 , 'value2' , ...) - sąrašas. Gali turėti vieną iš išvardytų reikšmių, NULL ar "". Sąraše gali būti ne daugiau kaip 65 535 reikšmių.

SET('vale1 , 'value2' , ...) - rinkinys. Gali turėti nuo 0 iki 64 reikšmių. kurios išvardytos sąraše

Eilučių stulpelių tipai

Vadovaukitės šiais nurodymais, kai reikia nustatyti eilutės tipą:

Nepriskirkite skaičiams eilutės tipo laukų. Tam skirtas skaitmeninis tipas. Kiekvienas skaitmuo eilutės lauke užima visą baitą, priešingai skaitmeniniam laukui, kur skaitmenims saugoti naudojami bitai. Rūšiuojant skaičius, kuriems priskirtas eilutės tipas, rezultatai gaunami nenuoseklūs.

Spartesniam darbui naudokite fiksuoto ilgio laukus, tokius kaip CHAR.

Tam, kad būtų sutaupyta vieta, naudokite kintamo ilgio laukus, tokius kaip VARCHAR.

Eilučių stulpelių tipai

Jei laukui bus sutekiama ne daugiau kaip viena reikšmė, naudokite ENUM.

Jei laukui bus suteikiama daugiau kaip viena reikšmė, naudokite SET.

Teksto paieškai, kuriame nebus skiriamos didžiosios ir mažosios raidės, naudokite TEXT.

Teksto paieškai, kuriame turi būti skiriamos didžiosios ir mažosios raidės, naudokite BLOB.

Nesaugokite paveikslėlių ar kitų dvejetainių objektų duomenų bazėje, o naudokite OS failų sistemą.

Datos ir laiko laukų tipai

DATETIME YYYY-MM-DD HH:MM:SS nuo 1000-01-01 00:00:00iki 9999-12-31 23:59:59

DATE YYYY-MM-DD nuo 1000-01-01 iki 9999-12-31

TIMESTAMP YYYYMMDDHHMMSS

TIME HH:MM:SS

YEAR YYYY

Datos ir laiko laukų tipai

Time stamp tipai:

- TIMESTAMP (14) - YYYYMMDDHHMMSS
- TIMESTAMP(12) - YYMMDDHHMMSS
- TIMESTAMP(10) - YYMMDDHHMM
- TIMESTAMP(8) - YYYYMMDD
- TIMESTAMP(6) - YYMMDD
- TIMESTAMP(4) - YYMM
- TIMESTAMP(2) – YY

Taip pat datai ir laikui saugoti galima naudoti Int tipą ir saugoti UNIX TIMESTAMP reikšmes, tačiau reiktų prisiminti jog 0 atitiks datą: 1970.01.01 00:00:00

Lentelių tipai

Lentelės tipas nurodomas sukūrimo užklausoje **ENGINE.**

Dažniausiai naudojami tipai: InnoDB, BDB, ISAM, MyISAM, MERGE, HEAP, MEMORY, Archive, CSV .....

Yra du transakcinių lentelių tipai (InnoDB ir BDB). Tačiau likusios lentelės (ISAM, MyISAM, MERGE (sujungtos), HEAP (sukauptos), Memory, CSV) nėra transakcinės.

Tinkamo lentelės tipo pasirinkimas gali turėti didžiulę įtaką darbo spartai ir DB struktūrai.

MyISAM lentelės

MyISAM tipo lentelės pakeitė ISAM lentelių tipą. MyISAM turi mažesnės apimties

indeksus nei ISAM, dėl to sistema naudoja mažiau išteklių, vykdydama komandą SELECT. Tačiau MyISAM naudoja daugiau procesoriaus energijos,

prireikus įterpti įrašą, kai indeksai suglaudinti.

MyISAM duomenų failai turi plėtinį. MYO, indeksų failai ttiri plėtinį. MY I. MyISAM duomenų

bazės saugomos kataloge

Duomenų failai visuomet didesni nei indeksų failai. 4 skyriuje pamatysite, kaip teisingai

naudoti indeksus ir kokie jie turi būti.

MyISAM lentelės

MyISAM l entelės skirstomos į tris tipus: statines, dinamines ir suglaudintas.

Kai lentelė jau sukurta, MySQL nustato, kokią lentelę - dinaminę ar statinę panaudoti.

Pagal numatytąjį nustatymą naudojamos statinės lentelės, kuriose nėra VARCHAR, BLOB, ar TEXT laukų. Jei yra bent vienas iš šių laukų, lentelė tampa dinaminė.

MyISAM Suglaudintos lentelės

Suglaudintos lentelės tik skaitomos ir užima žymiai mažiau vietos diske. Jos puikiai tinka saugoti archyviniams duomenims, kurie nesikeičia (yra tik skaitomi), ir kurių saugojimui nėra skirta daug Vietos.

MyISAM Suglaudintos lentelės

Suglaudintos lentelės pasižymi šiais bruožais:

- Sukuriamos myisampack priemone.
- Lentelės žymiai mažesnės.
- Kadangi kiekvienas įrašas suglaudintas atskirai, papildomos prieigos lieka mažai.
- Kiekvienas laukelis gali būti suglaudintas skirtingai, naudojant skirtingus suglaudinimo algoritmus.
- Galima suglaudinti fiksuoto ir dinaminio formato lenteles .

MyISAM Sujungtos lentelės

Sujungtos (MERGE) lentelės yra MyISAM lentelių sujungimas. Jas turėtumėte naudoti tuomet, kai MyISAM lentelės tampa pernelyg didelės.

Sujungtų (MERGE) lentelių pranašumai:

- Tam tikrais atvejais darbas vyksta sparčiau (galite padalyti lenteles į kelias dalis ir saugoti jas atskiruose diskuose, sujungta lentelė leidžia dirbti su jomis kaip su viena lentele).
- Mažesnis lentelės dydis. Kai kuriuose operacinėse sistemose failų dydis ribotas. Padalijus lenteles ir sukūrus sujungtą lentelę galima išvengti šios problemos. Failų apdorojimas tampa daug paprastesnis.
- Galite nurodyti, kad pagrindinė lentelė bus tik skaitoma ir komanda INSERT leisti vykdyti tik esamoje lentelėje. Tai reiškia, kad jei vykdomos komandos UPDATE ar INSERT, bus rizikuojama sugadinti tik vieną mažą lentelę. Šiai lentelei pataisyti neprireiks daug laiko.

MyISAM Sujungtos lentelės

Sujungtų lentelių trūkumai:

- Atliekant paiešką eq_ref šios lentelės gana lėtos.
- Turite būti atsargūs, keisdami vieną iš pagrindinių lentelių, nes taip galite sugadinti sujungtą lentelę (paprasčiausiai sujungta lentelė gali tapti neprieinama).
- Neveikia komanda REPLACE.
- Šios lentelės naudoja ilgesnius failų aprašus.

MyISAM Sujungtos lentelės

Sujungtos lentelės sukūrimas:

```
CREATE TABLE klient_1 (
    id INT AUTO INCREMENT PRIMARY KEY,
    vardas VARCHAR(40),
    pavarde VARCHAR(30),
    tipas TINYINT(4),
    gimimo_diena DATE

) TYPE=MyISAM;


CREATE TABLE klient_2 (
    id INT AUTO INCREMENT PRIMARY KEY,
    vardas VARCHAR(40),
    pavarde VARCHAR(30),
    tipas TINYINT(4),
    gimimo_diena DATE

) TYPE=MyISAM;
```

MyISAM Sujungtos lentelės

Sujungtos lentelės sukūrimas:

```
CREATE TABLE klientai_1_2  (
    id INT AUTO INCREMENT PRIMARY KEY,
    vardas VARCHAR(40),
    pavarde VARCHAR(30),
    tipas TINYINT(4),
    gimimo_diena DATE
 ) TYPE=MERGE
UNION=(klientai_1, klientai_2)
INSERT_METHOD = LAST;
```

HEAP lentelės

Sukauptos (HEAP) lentelės yra pačios sparčiausios, kadangi saugomos atmintyje ir naudoja atsitiktinius indeksus.

Jų trūkumas tas, kad jei įvyktų avarija, duomenys būtų prarasti, nes jie saugomi atmintyje. Taip pat šios lentelės negali saugoti didelio skaičiaus duomenų.

Dažniausiai sukauptos lentelės naudojamos tuomet, kai norima greičiau gauti rezultatus iš jau egzistuojančių lentelių. Pradinėse lentelėse įterpiami ir keičiami įrašai, o naujos sukurtos sukauptos (HEAP) lentelės skirtos sparčiam skaitymui.

HEAP lentelės

Heap lentelės sukūrimas:

```
CREATE TABLE klient (
    id INT AUTO INCREMENT PRIMARY KEY,
    vardas VARCHAR(40),
    pavarde VARCHAR(30),
    tipas TINYINT(4),
    gimimo_diena DATE
) TYPE=MyISAM;
```

```
CREATE TABLE heap_test TYPE=HEAP SELECT id, vardas, pavarde  FROM klient ;
```

HEAP lentelės

Sukauptos (HEAP) lentelės pasižymi šiomis savybėmis:

- Sukauptos lentelės saugomos operatyviojoje atmintyje.
- Lenteles apriboja mysqld reikšmė max_heap_table_size.
- Raktai naudojami kitaip nei MyISAM tipo lentelėse. Jie negali būti naudojami komandoje ORDER BY.
- Eilutės paieškai naudojamas visas raktas, o ne jo dalys.
- Indeksų paieškai naudojami tik operatoriai: = ir <=>.
- MySQL srities tvarkyklė negali nustatyti, kiek eilučių yra tarp dviejų irašų.

HEAP lentelės

Sukauptos (HEAP) lentelės pasižymi šiomis savybėmis:

- Lentelės sparčiai dirba tik tuomet, kai raktai tinkamai panaudojami.
- Sukauptos lentelės, skirtingai nei šifruotos lentelės, leidžia naudoti ne tik unikalius raktus.
- Laukas NULL neturi indekso.
- Šios lentelės neturi AUTO _ INCREMENT laukų.
- Nėra BLOB ar TEXT laukų.
- Sukauptos lentelės bus lėtesnės, jei įvesite indeksą, kurio lentelė nenaudoja.

InnoDB lentelės

InnoDB lentelės - tai transakcinio tipo lentelės (tai reiškia, kad galite vykdyti komandas COMMIT ir ROLLBACK). Kai vyksta iterpimo procesas, visa MyISAM yra blokuota ir joks kitas veiksmas, kad ir koks trumpalaikis jis būtų, su ta lentele negali būti atliekamas.

InnoDB lentelės naudoja eilutės lygio užblokavimą, tai reiškia, kad tik eilutė, o ne visa lentelė blokuojama ir, kad su kitomis eilutėmis gali būti atliekami veiksmai.

Greičio atžvilgiu, InnoDB lenteles turėtumėte naudoti, jei numato te, kad komandos INSERT ar UPOATE bus dažniau reikalingos nei komandos SELECT. MyISAM lenteles geriau pasirinkti tuo atveju, kai komandų SELECT bus atliekama daugiau nei UPDATE ar INSERT.

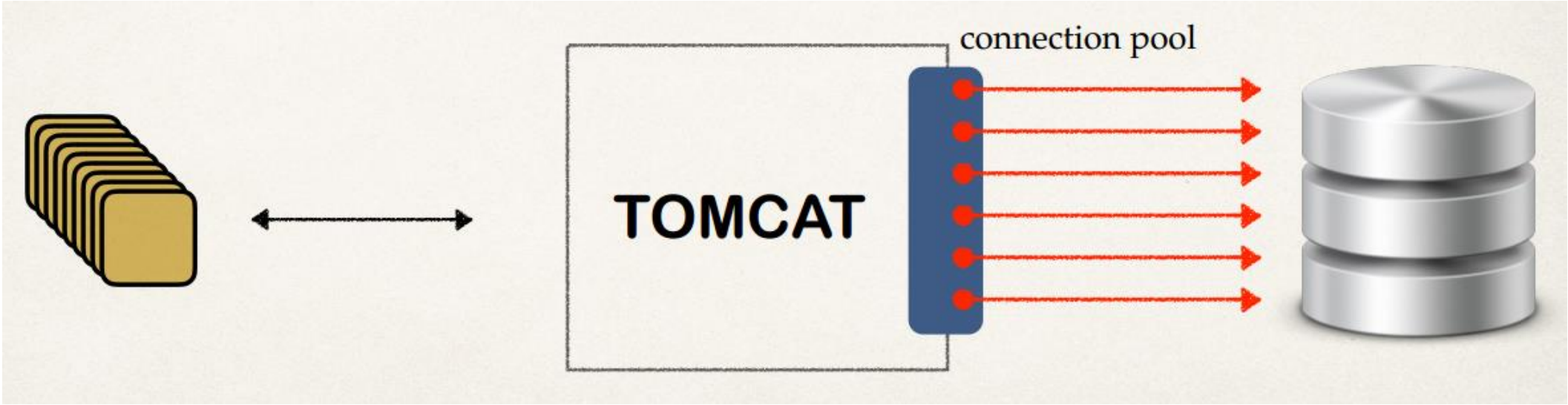
Įrašų įterpimas

```
INSERT INTO pirkejai  (pirkejo_nr,vardas, pavarde, nuolaida)
    VALUES(1, 'Jonas', 'Jonaitis', '10');
```

Įrašų įterpimas

```
INSERT INTO pirkejai
        (pirkejo_nr,vardas, pavarde, nuolaida)
VALUES
        (2, 'Petras', 'Petraitis', '7'),
        (3, 'Kestas', 'Kazys', '8'),
        (4, 'Mindaugas', 'Mindaugenas', '8');
```

MySQL prisijungimų valdymas



MySQL JDBC bibliotekos įdiegimas

Parsisiųstą JDBC biblioteką įdiekime į mūsų projektą.

Nukopijuokite failą mysql-connector-java-8.0.12.jar į

WebContent/WEB-INF/lib/mysql-connector-java-8.0.12.jar

JDBC pool sukūrimas

Norėdami sukurkti TomCat poolą prisijungimams prie DB turėtumėme sukurti failą: WebContent/META-INF/context.xml

```
<Context>
<Resource name="jdbc/lentele" auth="Container"
type="javax.sql.DataSource" maxActive="20" maxIdle="5" maxWait="10000"
username="vartotojas" password="slaptazodis"
driverClassName="com.mysql.jdbc.Driver"
url="jdbc:mysql://localhost:3306/lentele" />
</Context>
```

Duomenų atvaizdavimas

```
Connection connection=null;
Statement statement = null;
ResultSet rs = null;

try {
//1. Iš dataSrource gauti prisijungimą
connection=dataSource.getConnection();

//2. Susikurti Statement (pagal gautąjį prisijungimą)
statement = connection.createStatement();

//3. Gauti rezultatus įvykdžius SQL
String sql="SELECT * FROM naujas.lentele;";
rs=statement.executeQuery(sql);

 // Atvaizduoti gautus rezultatus
 while (rs.next()) {
  String name=rs.getString("name");
  out.println(name);
 }
} catch (SQLException e) {
e.printStackTrace();
}
```

Įprastinė JDBC užklausų vykdymo seka

Norint įvykdyti SQL užklausą iš Java, reiktų vykdyti toką seką:

1. Iš dataSrource gauti prisijungimą
2. Susikurti Statement (pagal gautąjį prisijungimą)
3. Gauti rezultatus įvykdžius SQL (ant statement) ir juos pasitalpinti į rezultato lauką
4. Atvaizduoti gautus rezultatus