Particioniranje velikih baz podatkov v MySQL je tehnika, ki omogoča razdelitev velikih tabel v manjše, bolj obvladljive dele, imenovane particije. Particije omogočajo hitrejše iskanje podatkov, izboljšano zmogljivost in enostavnejše upravljanje podatkov. Spodaj so navodila za uporabo particioniranja v MySQL:

**1. Razumevanje particioniranja**

Particioniranje v MySQL razdeli tabelo na osnovi določenega stolpca in pravila particioniranja. MySQL podpira dva načina particioniranja:

* **Horizontalno particioniranje**: Razdelitev vrstic na več particij na podlagi vrednosti.
* **Vertikalno particioniranje** (ni podprto neposredno): Razdelitev stolpcev, kar pa običajno zahteva ločene tabele.

Podprte so naslednje metode:

* **RANGE**: Particije se ustvarijo na podlagi obsega vrednosti.
* **LIST**: Particije na osnovi določenih vrednosti.
* **HASH**: Razdelitev z uporabo hash funkcije.
* **KEY**: Razdelitev temelji na MySQL-ovem notranjem algoritmu hash.

**2. Ustvarjanje particionirane tabele**

Particioniranje je mogoče uporabiti samo pri **InnoDB** ali **NDB** tabelah. Tukaj je primer:

**RANGE particioniranje**

Tabela se razdeli na podlagi obsega vrednosti v stolpcu.

CREATE TABLE sales (

 id INT NOT NULL,

 sale\_date DATE NOT NULL,

 amount DECIMAL(10, 2) NOT NULL

)

PARTITION BY RANGE (YEAR(sale\_date)) (

 PARTITION p1 VALUES LESS THAN (2000),

 PARTITION p2 VALUES LESS THAN (2010),

 PARTITION p3 VALUES LESS THAN (2020),

 PARTITION p4 VALUES LESS THAN MAXVALUE

);

**LIST particioniranje**

Tabela se razdeli glede na natančne vrednosti v stolpcu.

CREATE TABLE orders (

 id INT NOT NULL,

 region VARCHAR(50) NOT NULL,

 order\_date DATE NOT NULL

)

PARTITION BY LIST COLUMNS(region) (

 PARTITION north VALUES IN ('North', 'Northeast'),

 PARTITION south VALUES IN ('South', 'Southeast'),

 PARTITION west VALUES IN ('West', 'Southwest'),

 PARTITION east VALUES IN ('East', 'Midwest')

);

**HASH particioniranje**

Tabela se razdeli z uporabo hash funkcije.

CREATE TABLE logs (

 id INT NOT NULL,

 log\_date DATE NOT NULL

)

PARTITION BY HASH(YEAR(log\_date)) PARTITIONS 4;

**KEY particioniranje**

Samodejna razdelitev z uporabo MySQL-ovega hash algoritma.

CREATE TABLE users (

 id INT NOT NULL,

 username VARCHAR(50) NOT NULL

)

PARTITION BY KEY(id) PARTITIONS 4;

**3. Upravljanje particij**

**Dodajanje particij**

Določene vrste particij (RANGE in LIST) omogočajo dodajanje novih particij:

ALTER TABLE sales ADD PARTITION (

 PARTITION p5 VALUES LESS THAN (2030)

);

**Odstranjevanje particij**

Podatki v particiji bodo izbrisani:

ALTER TABLE sales DROP PARTITION p1;

**4. Prednosti particioniranja**

* **Povečana zmogljivost**: Poizvedbe obdelujejo le ustrezno particijo, kar zmanjša čas iskanja.
* **Lažje upravljanje**: Velike tabele postanejo bolj obvladljive.
* **Arhiviranje**: Stare particije lahko preprosto odstranite ali premaknete.

**5. Omejitve particioniranja v MySQL**

* **Indeksi**: Indekse je mogoče ustvariti le na particijskih stolpcih.
* **Particije in tuji ključi**: Tuji ključi niso podprti za particionirane tabele.
* **Stolpci za particioniranje**: Stolpec za particioniranje mora biti del primarnega ključa.

**6. Nasveti za particioniranje**

1. **Analiza podatkov**: Izberite metodo particioniranja glede na porazdelitev podatkov in vrsto poizvedb.
2. **Testiranje**: Pred uvajanjem v produkcijo testirajte delovanje particij na vzorčnih podatkih.
3. **Vzdrževanje**: Redno preverjajte in upravljajte particije, da preprečite, da bi postale prevelike.