Rakamin Academy

id/x partners

Project Based Internship

Data Warehouse Partitioning

Horizontal Partitioning

Daftar Isi

Exploration Source

[Article]

- Data Partitioning — System Design Concept
- Data Partitioning Techniques in System Design
- SQL Server Tutorial 23: Partitioning Tables and Indexes

[Video]

- Microsoft SQL Server Table Partitioning Demonstration

A. *Introduction*

Berbicara mengenai *partitioning*, terdapat dua teknik partisi yang cukup banyak digunakan yakni *horizontal partitioning* (biasa dikenal *sharding*), dan *vertical partitioning*.

Pada artikel ini, kita akan lebih fokus ke *horizontal partitioning*. Sederhananya *horizontal partitioning* ini lebih menekankan ke pembagian (partisi) tabel berdasarkan **baris**, seringkali dengan **menggunakan rentang** atau kondisi tertentu. Ini berguna ketika menangani tabel yang mengandung jumlah baris yang besar, dan data dapat dikelompokkan secara logis berdasarkan **kriteria** tertentu.

B. Implementasi Proses *Horizontal Partitioning*

Dalam implementasi *real*-nya didunia kerja, melakukan *horizontal partitioning* disarankan dalam beberapa situasi tertentu di mana strategi ini dapat memberikan manfaat yang signifikan. Berikut adalah beberapa situasi di mana *horizontal partitioning* dapat menjadi pilihan yang tepat :

1. Query Rentang Waktu Sering Dilakukan

   Jika seringkali terdapat query atau analisis yang melibatkan rentang waktu tertentu, *horizontal partitioning* dapat membantu mempercepat eksekusi query dengan membatasi pemindaian hanya pada partisi yang relevan berdasarkan tanggal atau waktu.

2. Segmentasi Berdasarkan Kategori atau Jenis Transaksi

   Jika data dapat secara logis dibagi berdasarkan kategori atau jenis transaksi yang berbeda, *horizontal partitioning* dapat memberikan manfaat dengan memfasilitasi akses cepat ke informasi spesifik.

3. Meningkatkan Paralelisme

   Horizontal partitioning dapat meningkatkan paralelisme dalam pemrosesan query karena beberapa partisi dapat diproses secara bersamaan, mempercepat waktu eksekusi.

Penting untuk mempertimbangkan karakteristik unik dari data dan kebutuhan akses data dalam konteks bisnis tertentu sebelum memutuskan untuk menerapkan *horizontal partitioning*. Evaluasi cermat terhadap pola query dan struktur data dapat membantu menentukan apakah horizontal partitioning akan memberikan manfaat yang signifikan.

C. Jenis-Jenis *Horizontal Partitioning*

1. Range Partitioning

Memisahkan data berdasarkan rentang nilai pada suatu kolom.

Contoh Query :

```
CREATE TABLE Transaksi (
    TransactionID INT,
    Amount DECIMAL(10, 2),
    TransactionDate DATE
)
PARTITION BY RANGE (YEAR(TransactionDate)) (
    PARTITION p0 VALUES LESS THAN (1990),
    PARTITION p1 VALUES LESS THAN (2000),
    PARTITION p2 VALUES LESS THAN (2010),
    PARTITION p3 VALUES LESS THAN MAXVALUE
```

);

Penjelasan Query :

- Tabel Transaksi di partisi berdasarkan rentang tahun dari **TransactionDate**.
- Partisi **p0** mencakup data sebelum tahun 1990, **p1** untuk data tahun 1990-1999, dan seterusnya.
- Partisi **p3** mengandung data yang tidak termasuk dalam rentang partisi sebelumnya.

2. *List Partitioning*

Memisahkan data berdasarkan nilai tertentu dari suatu kolom.

Contoh Query :

```
CREATE TABLE Karyawan (
  EmployeeID INT,
  DepartmentID INT
)
PARTITION BY LIST (DepartmentID) (
  PARTITION pEng VALUES (1),
  PARTITION pSales VALUES (2),
  PARTITION pIT VALUES (3),
  PARTITION pOthers VALUES (DEFAULT)
);
```

Penjelasan Query :

- Tabel Karyawan di partisi berdasarkan nilai **DepartmentID**.
- Setiap partisi mencakup karyawan yang terkait dengan departemen tertentu.

- Partisi **pOthers** menangani nilai **DepartmentID** yang tidak termasuk dalam partisi lainnya.

3. *Hash Partitioning*

Memisahkan data berdasarkan fungsi hash terhadap nilai kolom.

Contoh Query :

```
CREATE TABLE Produk (
  ProductID INT,
  ProductName VARCHAR(50)
)
PARTITION BY HASH (ProductID)
PARTITIONS 4;
```

Penjelasan Query :

- Tabel Produk dipartisi dengan metode hash berdasarkan nilai **ProductID**.
- Ada empat partisi yang akan menangani distribusi data secara merata berdasarkan hash dari nilai **ProductID**.

4. *Composite Partitioning*

Kombinasi dua atau lebih jenis *partitioning*.

Contoh Query :

```
CREATE TABLE Penjualan (
  TransactionID INT,
  Amount DECIMAL(10, 2),
  TransactionDate DATE,
  RegionID INT
```

```
)
PARTITION BY RANGE (YEAR(TransactionDate)) SUBPARTITION BY HASH
(RegionID) (
  PARTITION p0 VALUES LESS THAN (2000) (
    SUBPARTITION s0,
    SUBPARTITION s1
  ),
  PARTITION p1 VALUES LESS THAN (2010) (
    SUBPARTITION s2,
    SUBPARTITION s3
  ),
  PARTITION p2 VALUES LESS THAN MAXVALUE (
    SUBPARTITION s4,
    SUBPARTITION s5
  )
);
```

Penjelasan Query :

- Tabel Penjualan di partisi berdasarkan rentang tahun **TransactionDate**.
- Setiap partisi memiliki sub partisi berdasarkan fungsi *hash* dari nilai **RegionID**, menciptakan struktur partisi yang kompleks.

Studi Kasus: *Horizontal Partitioning* dalam Bank Digital

Client Anda adalah sebuah bank digital menghadapi peningkatan pesat dalam volume transaksi perbankan harian. Untuk meningkatkan kinerja query dan

manajemen data historis, *bank* ini memutuskan untuk menerapkan *horizontal partitioning* pada tabel transaksi mereka.

SOLUSI

1. **Query Pembentukan Tabel Transaksi**

```
CREATE TABLE Transactions (
    TransactionID INT PRIMARY KEY,
    AccountNumber INT,
    TransactionDate DATE,
    Amount DECIMAL(15, 2),
    TransactionType VARCHAR(20)
);
```

Penjelasan :

- Tabel **Transactions** dibuat untuk menyimpan data transaksi perbankan.
- Tabel ini belum mengalami *horizontal partitioning* pada tahap ini.

2. **Query Pembentukan Tabel Partisi Horizontal Berdasarkan Tanggal**

```
CREATE TABLE PartitionedTransactions (
    TransactionID INT PRIMARY KEY,
    AccountNumber INT,
    TransactionDate DATE,
    Amount DECIMAL(15, 2),
    TransactionType VARCHAR(20),
    PARTITION BY RANGE (YEAR(TransactionDate)) (
        PARTITION p2020 VALUES LESS THAN (2021),
```

```
    PARTITION p2021 VALUES LESS THAN (2022),
    PARTITION p2022 VALUES LESS THAN (2023),
    PARTITION p2023 VALUES LESS THAN (2024),
    PARTITION pMax VALUES LESS THAN MAXVALUE
  )
);
```

Penjelasan :

- Tabel **PartitionedTransactions** dibuat dengan partisi horizontal berdasarkan tahun transaksi.
- Ini memungkinkan data transaksi untuk dipecah menjadi partisi yang terpisah berdasarkan tahun.

3. Query Contoh Pemilihan Data dari Partisi Tertentu

```
SELECT *
FROM PartitionedTransactions PARTITION (p2021)
WHERE TransactionType = 'Deposit';
```

Penjelasan :

- Query ini memilih semua transaksi deposit dari partisi tahun 2021.
- *Horizontal partitioning* memastikan hanya partisi yang relevan yang akan dievaluasi, sehingga dapat meningkatkan kinerja query.

4. Query Pembaruan Jenis Transaksi

```
UPDATE PartitionedTransactions
SET TransactionType = 'Withdrawal'
WHERE TransactionType = 'Withdraw' AND TransactionDate < '2022-01-01';
```

Penjelasan :

- Dengan *horizontal partitioning*, pembaruan jenis transaksi hanya mempengaruhi partisi yang sesuai, mengurangi dampak pada seluruh dataset.

Melalui penerapan *horizontal partitioning*, *bank* digital ini berhasil meningkatkan efisiensi akses data, mengoptimalkan manajemen data historis, dan merespons lebih cepat terhadap query tertentu.

REFERENCE

Vertical partitioning vs horizontal partitioning

Horizontal Partitioning in System Design | by thebytestream | Medium . Database

Horizontal Partitioning With Example | by Sujoy Nath | Medium