

Kelompok 05 Lab A2

1. Bagus Athallah (24060123120002)
2. Akbar Maulana P (24060123140199)
3. Axel Nataksuma (24060123130058)
4. Agathan Khairy Bowo L (24060123140129)
5. Ahmad Rafi Alhammam (24060123140138)

Praktikum 4

1. Query untuk membuat index pada tabel accounts untuk mempercepat pencarian. Pilih salah satu kolom pada table tersebut dan berikan alasan pemilihan kolom tersebut sebagai index!

```
-- 1
CREATE NONCLUSTERED INDEX idx_account_number
ON accounts (account_number);

SELECT
    name AS index_name,
    type_desc,
    is_primary_key,
    is_unique
FROM sys.indexes
WHERE object_id = OBJECT_ID('accounts');
```

98 %

Results Messages

	index_name	type_desc	is_primary_key	is_unique
1	PK_accounts_46A222CD6D0F4BC5	CLUSTERED	1	1
2	idx_account_number	NONCLUSTERED	0	0

Kolom account_number dipilih karena merupakan kolom pencarian yang umum digunakan dalam transaksi perbankan. Selain itu, nilainya unik (mirip nomor rekening) dan Sering digunakan dalam WHERE clause saat mencari akun tertentu.

- Sebelum

Results Messages

SQL Server parse and compile time:
CPU time = 27 ms, elapsed time = 27 ms.

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.

(2 rows affected)
Table 'syspalvalues'. Scan count 0, logical reads 4, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0
Table 'sysidstats'. Scan count 1, logical reads 2, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0
Table 'syschobjs'. Scan count 0, logical reads 2, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0

- Sesudah

Results Messages

SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.

(3 rows affected)
Table 'syspalvalues'. Scan count 0, logical reads 6, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0
Table 'sysidstats'. Scan count 1, logical reads 2, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0
Table 'syschobjs'. Scan count 0, logical reads 2, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0

2. Query untuk membuat index pada tabel transactions untuk mempercepat pencarian. Pilih salah satu kolom pada table tersebut dan berikan alasan pemilihan kolom tersebut sebagai index!

```
-- 2
CREATE NONCLUSTERED INDEX idx_transactions_account_id ON transactions (account_id);

EXEC sp_helpindex 'transactions';
```

98 %

Results Messages

	index_name	index_description	index_keys
1	idx_transactions_account_id	nonclustered located on PRIMARY	account_id
2	PK_transact__85C600AFF854A2C6	clustered, unique, primary key located on PRIMARY	transaction_id

Index dibuat pada kolom `account_id` dalam tabel `transactions` karena kolom ini merupakan foreign key yang sering digunakan dalam pencarian data transaksi berdasarkan akun tertentu. Penggunaan `account_id` dalam klausa `WHERE` dan operasi `JOIN` dengan tabel `accounts` sangat umum dalam sistem perbankan, terutama saat menelusuri riwayat transaksi per akun. Dengan demikian, pembuatan index pada kolom ini dapat secara signifikan mempercepat proses pencarian dan meningkatkan kinerja query yang terkait.

- Sebelum

```
-- 2
CREATE NONCLUSTERED INDEX idx_transactions_account_id ON transactions (account_id);

SET STATISTICS TIME ON;
SET STATISTICS IO ON;

SELECT *
FROM transactions
WHERE account_id = '9cd97ecb-58c9-4610-b4b1-d9f72bcae7f7';

SET STATISTICS TIME OFF;
SET STATISTICS IO OFF;
```

98 %

Results Messages

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.

(4 rows affected)
Table 'transactions'. Scan count 1, logical reads 6, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.

Completion time: 2025-05-23T09:43:49.3487100+07:00

- Sesudah

```
-- 2
CREATE NONCLUSTERED INDEX idx_transactions_account_id ON transactions (account_id);

SET STATISTICS TIME ON;
SET STATISTICS IO ON;

SELECT *
FROM transactions
WHERE account_id = '9cd97ecb-58c9-4610-b4b1-d9f72bcae7f7';

SET STATISTICS TIME OFF;
SET STATISTICS IO OFF;
```

98 %

Results Messages

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.

(4 rows affected)
Table 'transactions'. Scan count 1, logical reads 6, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.

Completion time: 2025-05-23T09:45:19.5685143+07:00

Hasilnya terlihat sama karena query yang dijalankan terlalu ringan. Jika waktu < 1 ms, maka akan ditampilkan sebagai 0 ms.

3. Query untuk membuat composite index pada tabel accounts. Pilih lebih dari satu kolom pada tabel tersebut dan uji performa query sebelum dan setelah menggunakan index! (query pengujian berdasarkan kolom yang dipilih sebagai index)

```
-- 3
CREATE NONCLUSTERED INDEX idx_customer_accounttype
ON accounts (customer_id, account_type);

EXEC sp_helpindex 'accounts';
```

98 %

	index_name	index_description	index_keys
1	idx_account_number	nonclustered located on PRIMARY	account_number
2	idx_customer_accounttype	nonclustered located on PRIMARY	customer_id, account_type
3	PK_accounts_46A222CD6D0F4BC5	clustered, unique, primary key located on PRIMARY	account_id

- Sebelum

```
-- 3
CREATE NONCLUSTERED INDEX idx_customer_accounttype
ON accounts (customer_id, account_type);

SET STATISTICS IO ON;
SET STATISTICS TIME ON;

SELECT *
FROM accounts
WHERE customer_id = '5b3fb023-fd43-4cae-8783-ef1c98d11b38'
AND account_type = 'credit';

SET STATISTICS IO OFF;
SET STATISTICS TIME OFF;
```

98 %

Results	Messages
SQL Server parse and compile time: CPU time = 0 ms, elapsed time = 0 ms.	
(1 row affected) Table 'accounts'. Scan count 1, logical reads 4, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, 1	
SQL Server Execution Times: CPU time = 0 ms, elapsed time = 0 ms.	
SQL Server Execution Times: CPU time = 0 ms, elapsed time = 0 ms.	
Completion time: 2025-05-23T09:26:58.4619578+07:00	

- Sesudah

```
-- 3
CREATE NONCLUSTERED INDEX idx_customer_accounttype
ON accounts (customer_id, account_type);

SET STATISTICS IO ON;
SET STATISTICS TIME ON;

SELECT *
FROM accounts
WHERE customer_id = '5b3fb023-fd43-4cae-8783-ef1c98d11b38'
AND account_type = 'credit';

SET STATISTICS IO OFF;
SET STATISTICS TIME OFF;
```

98 %

Results	Messages
SQL Server parse and compile time: CPU time = 0 ms, elapsed time = 0 ms.	
(1 row affected) Table 'accounts'. Scan count 1, logical reads 4, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, 1	
SQL Server Execution Times: CPU time = 0 ms, elapsed time = 0 ms.	
SQL Server Execution Times: CPU time = 0 ms, elapsed time = 0 ms.	
Completion time: 2025-05-23T09:28:40.0023649+07:00	

Hasilnya terlihat sama karena query yang dijalankan terlalu ringan. Jika waktu < 1 ms, maka akan ditampilkan sebagai 0 ms.

4. Query untuk membuat cursor untuk membaca semua akun dari tabel accounts, lalu periksa balance masing-masing. Jika balance < 9.000, tampilkan pesan: "Saldo rendah untuk account_id: [account_id]".

```
-- 4
DECLARE @account_id CHAR(36);
DECLARE @balance DECIMAL(18, 2);

DECLARE cur_accounts CURSOR FOR
SELECT account_id, balance
FROM accounts;
OPEN cur_accounts;
FETCH NEXT FROM cur_accounts INTO @account_id, @balance;
WHILE @@FETCH_STATUS = 0
BEGIN
    IF @balance < 9000
    BEGIN
        PRINT 'Saldo rendah untuk account_id: ' + @account_id;
    END
    FETCH NEXT FROM cur_accounts INTO @account_id, @balance;
END;
CLOSE cur_accounts;
DEALLOCATE cur_accounts;
```

98 %

Messages

Saldo rendah untuk account_id: 0345E97A-8DEF-42FE-AB36-5124366D027A
Saldo rendah untuk account_id: 07a36a4b-7943-4634-b17a-7c7eb4407cd3
Saldo rendah untuk account_id: 0937445f-d5eb-4ef9-887f-4173fe662dd4
Saldo rendah untuk account_id: 1d97bce1-5c94-49e4-8e2a-6ad56b99afb0
Saldo rendah untuk account_id: 2fdf706a-7187-470c-a1c3-f3e747b72f20
Saldo rendah untuk account_id: 318c43bf-9511-4030-bd13-81a47a4b8cac
Saldo rendah untuk account_id: 34e9be76-4674-4d0f-b7c0-a39e43548747
Saldo rendah untuk account_id: 36f3d280-2886-4e51-adcd-dfc644e59d80
Saldo rendah untuk account_id: 3B6B0001-52C3-4FE1-B930-70DAA90C9F31
Saldo rendah untuk account_id: 418a133e-5e1f-4377-8d48-63de24b5b08e
Saldo rendah untuk account_id: 418fedb1-da8f-4c67-8073-4a147e2f5ded
Saldo rendah untuk account_id: 441aa9d2-bdf2-4b65-a79f-c18e66de0e17

5. Query untuk membuat cursor untuk membaca semua pelanggan (customers) dan gabungkan first_name dan last_name untuk ditampilkan dengan format: "Customer: [Nama Lengkap]".

```
-- 5
DECLARE @first_name VARCHAR(50);
DECLARE @last_name VARCHAR(50);
DECLARE @full_name VARCHAR(101);

DECLARE CustomerCursor CURSOR FOR
SELECT first_name, last_name
FROM customers;
OPEN CustomerCursor;
FETCH NEXT FROM CustomerCursor INTO @first_name, @last_name;
WHILE @@FETCH_STATUS = 0
BEGIN
    SET @full_name = CONCAT(@first_name, ' ', @last_name);
    PRINT 'Customer: ' + @full_name;
    FETCH NEXT FROM CustomerCursor INTO @first_name, @last_name;
END;
CLOSE CustomerCursor;
DEALLOCATE CustomerCursor;
```

98 %

Messages

Customer: Sergeant Maith
Customer: Shaine Matic
Customer: Buddie Sandison
Customer: Stephanus Frear
Customer: Corette Vicent
Customer: Devon McQuode
Customer: Jacquenette Hilldrup
Customer: Bobette Shambrooke
Customer: Donnajean Amerighi
Customer: Barbabra Swires
Customer: Sandro Alten
Customer: Nester Niccolls
Customer: Rebekah Conkie
Customer: Sidonie Morke