

teknologi
informasi
UNIVERSITAS MUHAMMADIYAH YOGYAKARTA

Modul Praktikum

Pengembangan Aplikasi Basisdata

Apriliya Kurnianti S.T., M.Eng



Daftar Isi

Optimasi Query dalam Aplikasi.....	1
Tujuan Praktikum	1
Alat dan Bahan	1
Langkah-Langkah Praktikum	1

Optimasi Query dalam Aplikasi

Tujuan Praktikum

Memahami Optimasi Query dalam Aplikasi

Alat dan Bahan

- SQL Server (Minimal SQL Server Express)
- SQL Server Management Studio (SSMS)
- Visual Studio (Minimal 2019 dengan .NET Framework)
- C# Windows Forms (WinForms)

Langkah-Langkah Praktikum

1. Gunakan database OrganisasiMahasiswa
2. Buat query index pada sql server

```
-- 1. Indeks pada kolom Nama (untuk mempercepat pencarian berdasarkan nama)
IF NOT EXISTS (
    SELECT 1
    FROM sys.indexes
    WHERE name = 'idx_Mahasiswa_Nama'
    AND object_id = OBJECT_ID('dbo.Mahasiswa')
)
BEGIN
    CREATE NONCLUSTERED INDEX idx_Mahasiswa_Nama
    ON dbo.Mahasiswa(Nama);
    PRINT 'Created idx_Mahasiswa_Nama';
END
ELSE
    PRINT 'idx_Mahasiswa_Nama sudah ada.';

-- 2. Indeks pada kolom Email (jika Anda sering mencari atau meng-JOIN berdasarkan email)
IF NOT EXISTS (
    SELECT 1
    FROM sys.indexes
    WHERE name = 'idx_Mahasiswa_Email'
    AND object_id = OBJECT_ID('dbo.Mahasiswa')
)
BEGIN
    CREATE NONCLUSTERED INDEX idx_Mahasiswa_Email
    ON dbo.Mahasiswa(Email);
    PRINT 'Created idx_Mahasiswa_Email';
END
ELSE
    PRINT 'idx_Mahasiswa_Email sudah ada.';

-- 3. Indeks pada kolom Telepon (untuk lookup cepat berdasarkan nomor telepon)
IF NOT EXISTS (
    SELECT 1
    FROM sys.indexes
    WHERE name = 'idx_Mahasiswa_Telepon'
    AND object_id = OBJECT_ID('dbo.Mahasiswa')
)
BEGIN
    CREATE NONCLUSTERED INDEX idx_Mahasiswa_Telepon
    ON dbo.Mahasiswa(Telepon);
    PRINT 'Created idx_Mahasiswa_Telepon';
END
ELSE
    PRINT 'idx_Mahasiswa_Telepon sudah ada.';
GO
```

3. Revisi code anda pada form 1

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.IO;
using System.Runtime.Caching;
using System.Windows.Forms;
using NPOI.SS.UserModel;
using NPOI.XSSF.UserModel;

namespace CRUD24
{
    3 references
    public partial class Form1 : Form
    {
        private readonly string connectionString =
            "Data Source=DESKTOP-RAM20FI\\APRILIYA;Initial Catalog=OrganisasiMahasiswa;Integrated Security=True";

        // Inisialisasi cache dan kebijakan kadaluarsa
        private readonly MemoryCache _cache = MemoryCache.Default;
        private readonly CacheItemPolicy _policy = new CacheItemPolicy
        {
            AbsoluteExpiration = DateTimeOffset.Now.AddMinutes(5)
        };
        private const string CacheKey = "MahasiswaData";

        1 reference
        public Form1()
        {
            InitializeComponent();
        }

        1 reference
        private void Form1_Load(object sender, EventArgs e)
        {
            EnsureIndexes();
            LoadData();
        }

        1 reference
        private void EnsureIndexes()
        {
            using (var conn = new SqlConnection(connectionString))
            {
                conn.Open();
                var indexScript = @"
                IF OBJECT_ID('dbo.Mahasiswa', 'U') IS NOT NULL
                BEGIN
                    IF NOT EXISTS (SELECT 1 FROM sys.indexes WHERE name='idx_Mahasiswa_Nama')
                        CREATE NONCLUSTERED INDEX idx_Mahasiswa_Nama ON dbo.Mahasiswa(Nama);
                    IF NOT EXISTS (SELECT 1 FROM sys.indexes WHERE name='idx_Mahasiswa_Email')
                        CREATE NONCLUSTERED INDEX idx_Mahasiswa_Email ON dbo.Mahasiswa(Email);
                    IF NOT EXISTS (SELECT 1 FROM sys.indexes WHERE name='idx_Mahasiswa_Telepon')
                        CREATE NONCLUSTERED INDEX idx_Mahasiswa_Telepon ON dbo.Mahasiswa(Telepon);
                END";
                using (var cmd = new SqlCommand(indexScript, conn))
                {
                    cmd.ExecuteNonQuery();
                }
            }
        }
    }
}
```

5 references

```
private void LoadData()
{
    DataTable dt;
    if (_cache.Contains(CacheKey))
    {
        dt = _cache.Get(CacheKey) as DataTable;
    }
    else
    {
        dt = new DataTable();
        using (var conn = new SqlConnection(connectionString))
        {
            conn.Open();
            var query = "SELECT NIM AS [NIM], Nama, Email, Telepon, Alamat FROM dbo.Mahasiswa";
            var da = new SqlDataAdapter(query, conn);
            da.Fill(dt);
        }
        _cache.Add(CacheKey, dt, _policy);
    }

    dgvMahasiswa.AutoGenerateColumns = true;
    dgvMahasiswa.DataSource = dt;
}
```

1 reference

```
private void AnalyzeQuery(string sqlQuery)
{
    using (var conn = new SqlConnection(connectionString))
    {
        conn.InfoMessage += (s, e) => MessageBox.Show(e.Message, "STATISTICS INFO");
        conn.Open();
        var wrapped = $"
        SET STATISTICS IO ON;
        SET STATISTICS TIME ON;
        {sqlQuery};
        SET STATISTICS IO OFF;
        SET STATISTICS TIME OFF;";
        using (var cmd = new SqlCommand(wrapped, conn))
        {
            cmd.ExecuteNonQuery();
        }
    }
}
```

1 reference

```
private void BtnTambah(object sender, EventArgs e)
{
    if (string.IsNullOrEmpty(txtNIM.Text) || string.IsNullOrEmpty(txtNama.Text))
    {
        MessageBox.Show("Harap isi semua data!", "Peringatan");
        return;
    }
    try
    {
        using (var conn = new SqlConnection(connectionString))
        {
            conn.Open();
            using (var cmd = new SqlCommand("AddMahasiswa", conn))
            {
                cmd.CommandType = CommandType.StoredProcedure;
                cmd.Parameters.AddWithValue("@NIM", txtNIM.Text.Trim());
                cmd.Parameters.AddWithValue("@Nama", txtNama.Text.Trim());
                cmd.Parameters.AddWithValue("@Email", txtEmail.Text.Trim());
                cmd.Parameters.AddWithValue("@Telepon", txtTelepon.Text.Trim());
                cmd.Parameters.AddWithValue("@Alamat", txtAlamat.Text.Trim());
                cmd.ExecuteNonQuery();
            }
        }
        _cache.Remove(CacheKey);
        MessageBox.Show("Data berhasil ditambahkan!", "Sukses");
        LoadData();
        ClearForm();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message, "Kesalahan");
    }
}
```

```

1 reference
private void BtnHapus(object sender, EventArgs e)
{
    if (dgvMahasiswa.SelectedRows.Count == 0) return;
    if (MessageBox.Show("Yakin ingin menghapus data ini?", "Konfirmasi", MessageBoxButtons.YesNo) != DialogResult.Yes)
        return;
    try
    {
        var nim = dgvMahasiswa.SelectedRows[0].Cells["NIM"].Value.ToString();
        using (var conn = new SqlConnection(connectionString))
        {
            conn.Open();
            using (var cmd = new SqlCommand("DeleteMahasiswa", conn))
            {
                cmd.CommandType = CommandType.StoredProcedure;
                cmd.Parameters.AddWithValue("@NIM", nim);
                cmd.ExecuteNonQuery();
            }
        }
        _cache.Remove(CacheKey);
        MessageBox.Show("Data berhasil dihapus!", "Sukses");
        LoadData();
        ClearForm();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message, "Kesalahan");
    }
}

```

```

1 reference
private void BtnUbah(object sender, EventArgs e)
{
    if (dgvMahasiswa.SelectedRows.Count == 0)
    {
        MessageBox.Show("Pilih data yang akan diubah!", "Peringatan");
        return;
    }
    try
    {
        using (var conn = new SqlConnection(connectionString))
        {
            conn.Open();
            using (var cmd = new SqlCommand("UpdateMahasiswa", conn))
            {
                cmd.CommandType = CommandType.StoredProcedure;
                cmd.Parameters.AddWithValue("@NIM", txtNIM.Text.Trim());
                cmd.Parameters.AddWithValue("@Nama", txtNama.Text.Trim());
                cmd.Parameters.AddWithValue("@Email", txtEmail.Text.Trim());
                cmd.Parameters.AddWithValue("@Telepon", txtTelepon.Text.Trim());
                cmd.Parameters.AddWithValue("@Alamat", txtAlamat.Text.Trim());
                cmd.ExecuteNonQuery();
            }
        }
        _cache.Remove(CacheKey);
        MessageBox.Show("Data berhasil diperbarui!", "Sukses");
        LoadData();
        ClearForm();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message, "Kesalahan");
    }
}

```

```

1 reference
private void BtnRefresh_Click(object sender, EventArgs e)
{
    _cache.Remove(CacheKey);
    LoadData();
}

```

```

1 reference
private void button1_Click(object sender, EventArgs e)
{
    using (var openFileDialog = new OpenFileDialog())
    {
        openFileDialog.Filter = "Excel Files|*.xlsx;*.xlsm";
        if (openFileDialog.ShowDialog() == DialogResult.OK)
            PreviewData(openFileDialog.FileName);
    }
}

```

```

/// <summary>
/// Menampilkan pratinjau data dari file Excel sebelum impor
/// </summary>
1 reference
private void PreviewData(string filePath)
{
    try
    {
        using (var fs = new FileStream(filePath, FileMode.Open, FileAccess.Read))
        {
            IWorkbook workbook = new XSSFWorkbook(fs);
            ISheet sheet = workbook.GetSheetAt(0);
            DataTable dt = new DataTable();

            // Header kolom
            IRow headerRow = sheet.GetRow(0);
            foreach (var cell in headerRow.Cells)
            {
                dt.Columns.Add(cell.ToString());
            }

            // Baris data
            for (int i = 1; i <= sheet.LastRowNum; i++)
            {
                IRow dataRow = sheet.GetRow(i);
                DataRow newRow = dt.NewRow();
                int cellIndex = 0;
                foreach (var cell in dataRow.Cells)
                {
                    newRow[cellIndex++] = cell.ToString();
                }
                dt.Rows.Add(newRow);
            }

            PreviewForm previewForm = new PreviewForm(dt);
            previewForm.ShowDialog();
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error reading the Excel file: " + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

```

1 reference
private void BtnAnalyze_Click(object sender, EventArgs e)
{
    var heavyQuery = "SELECT Nama, Email, Telepon FROM dbo.Mahasiswa WHERE Nama LIKE 'A%'";
    AnalyzeQuery(heavyQuery);
}

```

```

1 reference
private void dgvMahasiswa_CellClick(object sender, DataGridViewCellEventArgs e)
{
    if (e.RowIndex < 0) return;
    var row = dgvMahasiswa.Rows[e.RowIndex];
    txtNIM.Text = row.Cells[0].Value?.ToString() ?? string.Empty;
    txtNama.Text = row.Cells[1].Value?.ToString() ?? string.Empty;
    txtEmail.Text = row.Cells[2].Value?.ToString() ?? string.Empty;
    txtTelepon.Text = row.Cells[3].Value?.ToString() ?? string.Empty;
    txtAlamat.Text = row.Cells[4].Value?.ToString() ?? string.Empty;
}

```

```

3 references
private void ClearForm()
{
    txtNIM.Clear();
    txtNama.Clear();
    txtEmail.Clear();
    txtTelepon.Clear();
    txtAlamat.Clear();
    dgvMahasiswa.ClearSelection();
}

```

Penjelasan :

A. Inisialisasi & Konfigurasi Umum

- connectionString menyimpan info koneksi ke SQL Server.
- Menggunakan Integrated Security, sehingga autentikasi Windows.
- MemoryCache: cache in-memory kebanyakan .NET untuk menyimpan hasil query.

- `CacheItemPolicy`: menetapkan TTL (time-to-live) 5 menit.
- `CacheKey`: kunci unik untuk menyimpan/ambil `DataTable` mahasiswa di cache.

B. Lifecycle Form

- `InitializeComponent()`: auto-generated, membangun UI dari Designer.
- `Form1_Load` dipicu saat form tampil pertama kali:
 - o `EnsureIndexes()`: membuat indeks di database jika belum ada.
 - o `LoadData()`: memuat data mahasiswa, dari cache atau database.

C. Membuat Indeks Otomatis

- Mengeksekusi T-SQL kondisional:
 - o Cek apakah tabel Mahasiswa ada.
 - o Cek eksistensi indeks `idx_Mahasiswa_Nama` (dan Email, Telepon).
 - o Buat nonclustered index bila belum ada.
- Mempercepat operasi `WHERE Nama=...`, `WHERE Email=...`, dan `WHERE Telepon=...`

D. Caching dan Load Data

- Cek Cache: jika `MahasiswaData` ada di cache, ambil dari memori—tanpa hit database.
- Query Database: jika cache kosong, jalankan `SELECT ... FROM Mahasiswa`, isi `DataTable`, kemudian simpan di cache selama 5 menit.
- Bind ke Grid: tampilkan `DataTable` pada `DataGridView` (`dgvMahasiswa`).

E. Analisis Performa Query

- Menyisipkan perintah `STATISTICS IO/TIME` untuk melihat:
 - o I/O reads per tabel.
 - o Waktu CPU dan Total waktu sejak eksekusi query dimulai hingga berakhir, termasuk semua latensi, sedangkan CPU time hanya menghitung waktu prosesor (elapsed time).
- Hasilnya di-handle lewat event `InfoMessage`, ditampilkan ke pengguna.

F. Operasi CRUD (Tambah, Hapus, Ubah)

- Menggunakan SP (`AddMahasiswa`, `DeleteMahasiswa`, `UpdateMahasiswa`) menjaga logika database terpusat.
- Cache invalidation: setiap perubahan data menghapus entry cache, sehingga data grid selalu up-to-date.

G. Refresh & Clear Form

- Refresh: tombol manual untuk reload data (menghapus cache).
- ClearForm: reset semua field input dan hilangkan highlight di grid.

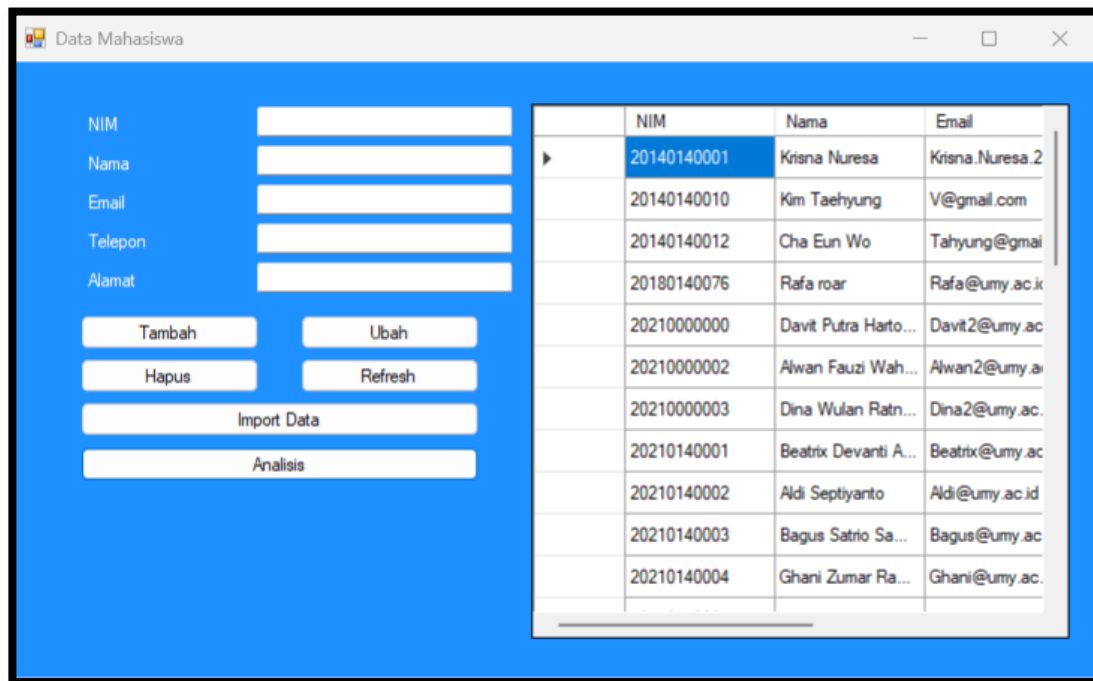
H. Impor Data dari Excel

- NPOI: library .NET untuk membaca .xlsx.
- Data ditampilkan di form kedua (PreviewForm) untuk validasi sebelum di-push ke database.

Dengan struktur ini, aplikasi sudah menerapkan:

1. Optimal: indexing + caching mengurangi beban database.
2. Terukur: STATISTICS IO/TIME memudahkan profiling query.
3. Tahan kesalahan: validasi input dan error handling via try/catch.
4. Modular: logika CRUD di SP, preview impor terpisah di PreviewForm.

Tampilan akhir Form



NIM	Nama	Email
20140140001	Krisna Nuresa	Krisna.Nuresa.2
20140140010	Kim Taehyung	V@gmail.com
20140140012	Cha Eun Wo	Tahyung@gmail
20180140076	Rafa roar	Rafa@umy.ac.id
20210000000	Davit Putra Harto...	Davit2@umy.ac
20210000002	Alwan Fauzi Wah...	Alwan2@umy.ac
20210000003	Dina Wulan Ratn...	Dina2@umy.ac
20210140001	Beatrix Devanti A...	Beatrix@umy.ac
20210140002	Aldi Septiyanto	Aldi@umy.ac.id
20210140003	Bagus Satrio Sa...	Bagus@umy.ac
20210140004	Ghani Zumar Ra...	Ghani@umy.ac