Step 7: Performance Testing

1. Create Test Queries
2. Analyze Query Plans
3. Performance Benchmarks

Query:

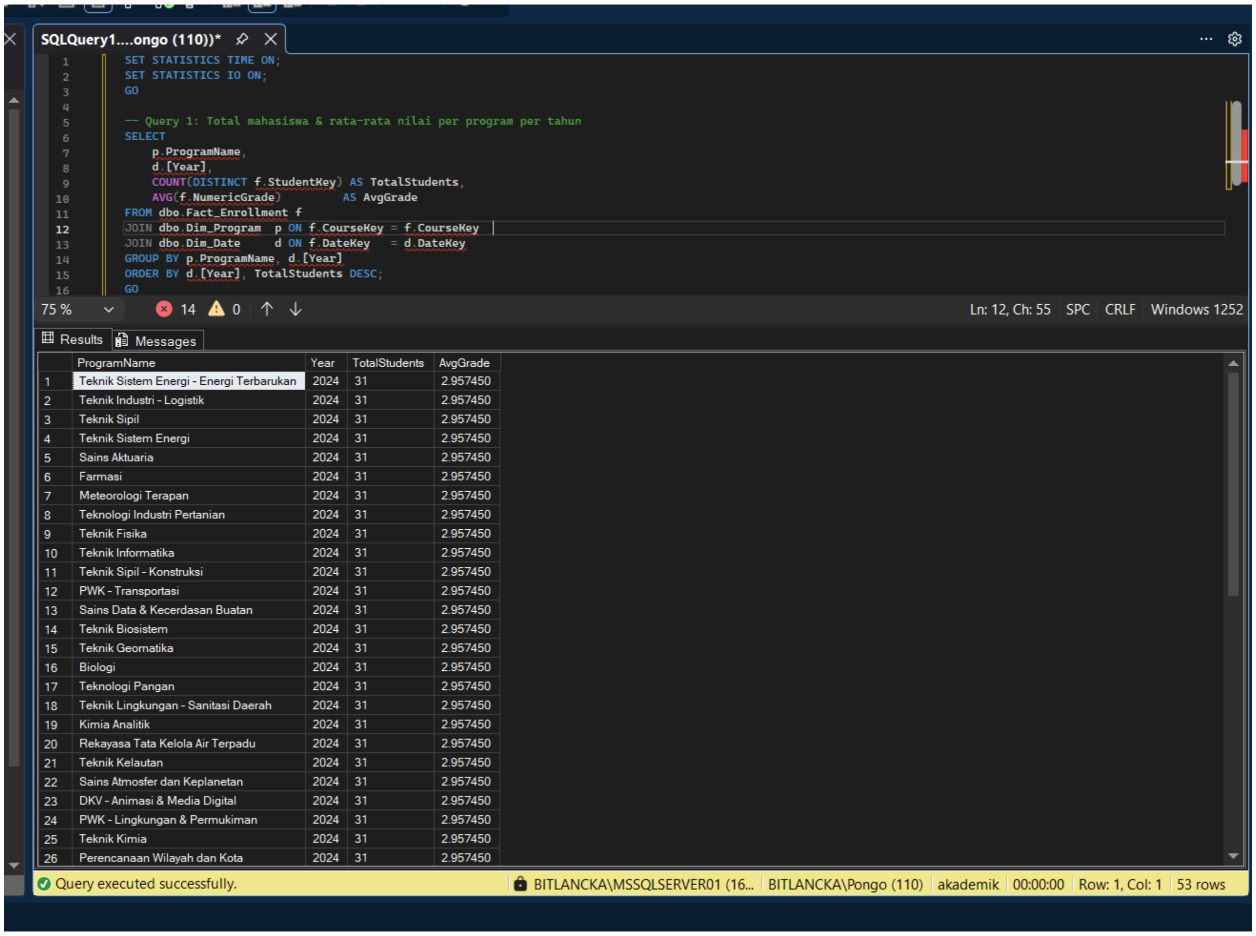
```
----------------------------------
-- Performance Testing
----------------------------------
-- 1. Average grade per program per year
SELECT
   p.ProgramName,
   d.[Year],
   AVG(f.NumericGrade) AS AvgGrade
FROM dbo.Fact_Enrollment f
JOIN dbo.Dim_Date d ON f.DateKey = d.DateKey
JOIN dbo.Dim_Course c ON f.CourseKey = c.CourseKey
JOIN dbo.Dim_Program p ON c.ProgramNaturalID = p.ProgramCode
GROUP BY p.ProgramName, d.[Year]
ORDER BY d.[Year], p.ProgramName;

-- 2. Monthly enrollment trend
SELECT
   d.[Year], d.[Month],
   COUNT(*) AS TotalEnrollments
FROM dbo.Fact_Enrollment f
JOIN dbo.Dim_Date d ON f.DateKey = d.DateKey
GROUP BY d.[Year], d.[Month]
ORDER BY d.[Year], d.[Month];
GO

-- 3. – Funnel Admission → Enrollment → Graduation per Program per Tahun
;WITH Adm AS (
   SELECT
      a.ProgramKey,
      d.[Year] AS AdmissionYear,
      COUNT(DISTINCT a.StudentKey) AS TotalApplicants
   FROM dbo.Fact_Admission a
   JOIN dbo.Dim_Date d ON a.AdmissionDateKey = d.DateKey
   GROUP BY a.ProgramKey, d.[Year]
),
Enr AS (
    SELECT
        c.ProgramNaturalID,
        d.[Year] AS EnrollmentYear,
        COUNT(DISTINCT f.StudentKey) AS TotalEnrolled
    FROM dbo.Fact_Enrollment f
    JOIN dbo.Dim_Course c ON f.CourseKey = c.CourseKey
    JOIN dbo.Dim_Date d   ON f.DateKey   = d.DateKey
    GROUP BY c.ProgramNaturalID, d.[Year]
),
Grad AS (
    SELECT
        g.ProgramKey,
        d.[Year] AS GraduationYear,
        COUNT(DISTINCT g.StudentKey) AS TotalGraduated
    FROM dbo.Fact_Graduation g
    JOIN dbo.Dim_Date d ON g.GraduationDateKey = d.DateKey
    GROUP BY g.ProgramKey, d.[Year]
)
SELECT
    p.ProgramName,
    a.AdmissionYear,
    a.TotalApplicants,
    ISNULL(e.TotalEnrolled, 0)  AS TotalEnrolled,
    ISNULL(g.TotalGraduated, 0) AS TotalGraduated,
    CASE WHEN a.TotalApplicants = 0
         THEN 0 ELSE 1.0 * ISNULL(e.TotalEnrolled, 0) / a.TotalApplicants END AS
Conv_Adm_to_Enroll,
    CASE WHEN a.TotalApplicants = 0
         THEN 0 ELSE 1.0 * ISNULL(g.TotalGraduated, 0) / a.TotalApplicants END AS
Conv_Adm_to_Grad
FROM Adm a
JOIN dbo.Dim_Program p ON a.ProgramKey = p.ProgramKey
LEFT JOIN Enr e
    ON p.ProgramCode = e.ProgramNaturalID
   AND a.AdmissionYear = e.EnrollmentYear
LEFT JOIN Grad g
    ON a.ProgramKey    = g.ProgramKey
   AND a.AdmissionYear = g.GraduationYear
ORDER BY p.ProgramName, a.AdmissionYear;
GO

-- 4: Distribusi lama studi dan GPA per program
SELECT
```

```
    p.ProgramName,
    COUNT(*)                    AS TotalGraduates,
    AVG(g.GPA)                  AS AvgGPA,
    MIN(g.GPA)                  AS MinGPA,
    MAX(g.GPA)                  AS MaxGPA,
    AVG(CAST(g.StudyDuration AS FLOAT)) AS AvgStudyMonths,
    MIN(g.StudyDuration)        AS MinStudyMonths,
    MAX(g.StudyDuration)        AS MaxStudyMonths
FROM dbo.Fact_Graduation g
JOIN dbo.Dim_Program p ON g.ProgramKey = p.ProgramKey
GROUP BY p.ProgramName
ORDER BY p.ProgramName;
GO

-- 5: Top 10 mahasiswa per program (GPA + SKS)
;WITH GradDetail AS (
    SELECT
        g.StudentKey,
        s.StudentNaturalID,
        s.FullName,
        p.ProgramName,
        g.GPA,
        g.TotalCredits,
        ROW_NUMBER() OVER (
            PARTITION BY p.ProgramName
            ORDER BY g.GPA DESC, g.TotalCredits DESC
        ) AS rn
    FROM dbo.Fact_Graduation g
    JOIN dbo.Dim_Student s ON g.StudentKey = s.StudentKey
    JOIN dbo.Dim_Program p ON g.ProgramKey = p.ProgramKey
)
SELECT
    ProgramName,
    rn AS RankInProgram,
    StudentNaturalID,
    FullName,
    GPA,
    TotalCredits
FROM GradDetail
WHERE rn <= 10
ORDER BY ProgramName, rn;
GO
```

```
-- 6: Pengaruh Attendance terhadap NumericGrade
;WITH EnrBucket AS (
   SELECT
      CASE
         WHEN AttendanceRate < 60 THEN '< 60%'
         WHEN AttendanceRate < 75 THEN '60–74%'
         WHEN AttendanceRate < 90 THEN '75–89%'
         ELSE '>= 90%'
      END AS AttendanceBucket,
      NumericGrade
   FROM dbo.Fact_Enrollment
   WHERE NumericGrade IS NOT NULL
)
SELECT
   AttendanceBucket,
   COUNT(*)             AS TotalEnrollments,
   AVG(NumericGrade)       AS AvgGrade,
   MIN(NumericGrade)       AS MinGrade,
   MAX(NumericGrade)       AS MaxGrade
FROM EnrBucket
GROUP BY AttendanceBucket
ORDER BY
   CASE AttendanceBucket
      WHEN '< 60%'  THEN 1
      WHEN '60–74%' THEN 2
      WHEN '75–89%' THEN 3
      WHEN '>= 90%' THEN 4
   END;
GO

-- 7: Trend enrollment per semester dan program
SELECT
   sem.SemesterCode,
   d.[Year],
   p.ProgramName,
   COUNT(*)                   AS TotalEnrollments,
   COUNT(DISTINCT f.StudentKey)   AS DistinctStudents,
   AVG(f.NumericGrade)          AS AvgGrade
FROM dbo.Fact_Enrollment f
JOIN dbo.Dim_Semester sem ON f.SemesterKey = sem.SemesterKey
JOIN dbo.Dim_Date d       ON f.DateKey     = d.DateKey
JOIN dbo.Dim_Course c     ON f.CourseKey   = c.CourseKey
JOIN dbo.Dim_Program p    ON c.ProgramNaturalID = p.ProgramCode
```

```
GROUP BY sem.SemesterCode, d.[Year], p.ProgramName
ORDER BY d.[Year], sem.SemesterCode, p.ProgramName;
GO
```

Output:



SQLQuery1....ongo (110))*

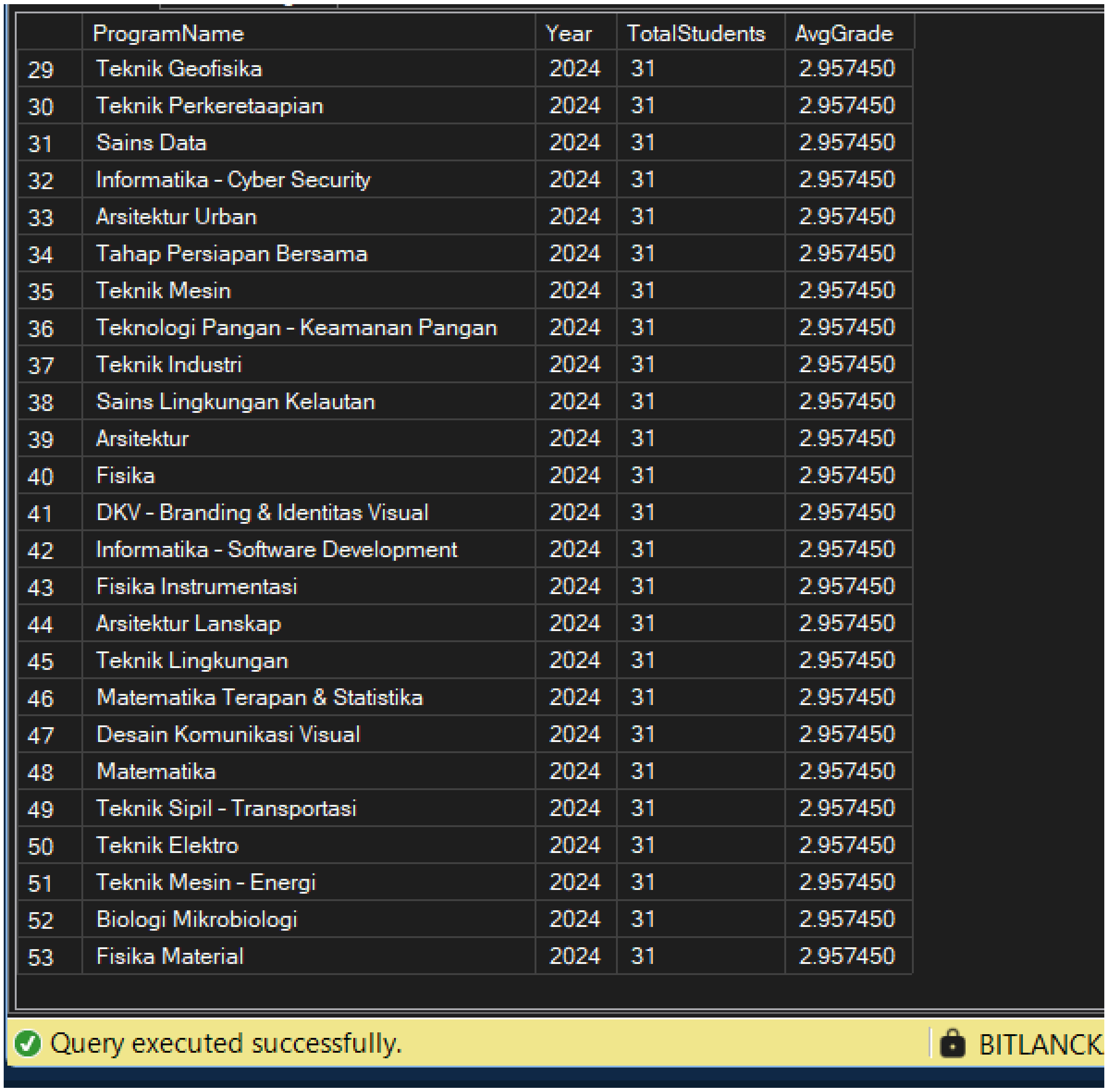
```
SET STATISTICS TIME ON;
SET STATISTICS IO ON;
GO

-- Query 1: Total mahasiswa & rata-rata nilai per program per tahun
SELECT
    p.ProgramName,
    d.[Year],
    COUNT(DISTINCT f.StudentKey) AS TotalStudents,
    AVG(f.NumericGrade)          AS AvgGrade
FROM dbo.Fact_Enrollment f
JOIN dbo.Dim_Program  p ON f.CourseKey = f.CourseKey
JOIN dbo.Dim_Date     d ON f.DateKey   = d.DateKey
GROUP BY p.ProgramName, d.[Year]
ORDER BY d.[Year], TotalStudents DESC;
GO
```

75 % ✖ 14 ⚠ 0   Ln: 12, Ch: 55 SPC CRLF Windows 1252

Results | Messages

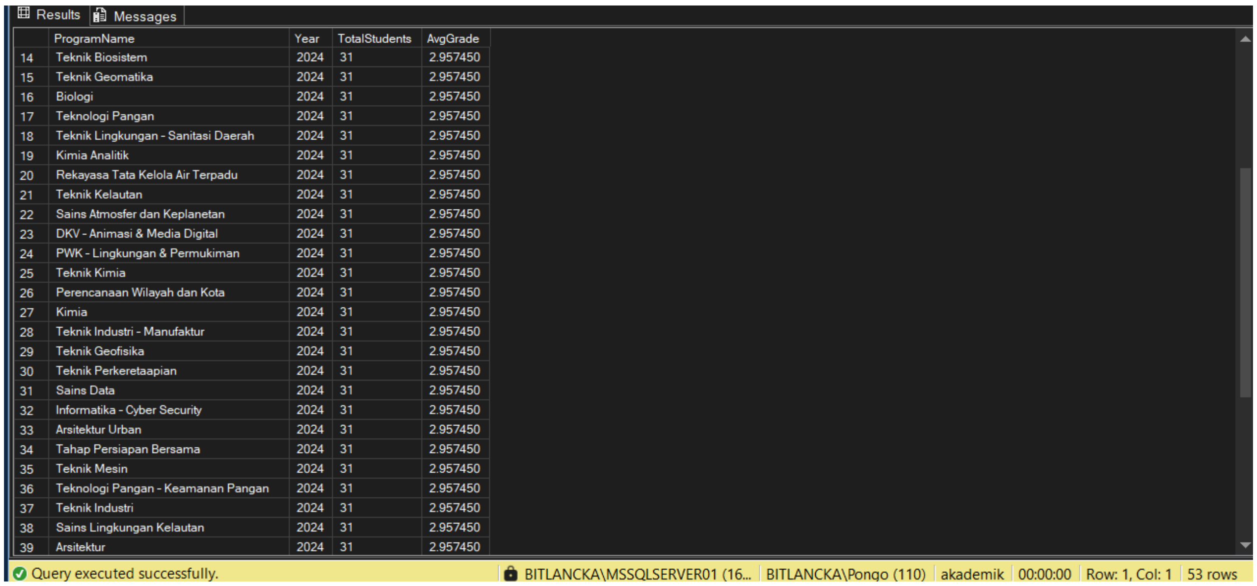
	ProgramName	Year	TotalStudents	AvgGrade
1	Teknik Sistem Energi - Energi Terbarukan	2024	31	2.957450
2	Teknik Industri - Logistik	2024	31	2.957450
3	Teknik Sipil	2024	31	2.957450
4	Teknik Sistem Energi	2024	31	2.957450
5	Sains Aktuaria	2024	31	2.957450
6	Farmasi	2024	31	2.957450
7	Meteorologi Terapan	2024	31	2.957450
8	Teknologi Industri Pertanian	2024	31	2.957450
9	Teknik Fisika	2024	31	2.957450
10	Teknik Informatika	2024	31	2.957450
11	Teknik Sipil - Konstruksi	2024	31	2.957450
12	PWK - Transportasi	2024	31	2.957450
13	Sains Data & Kecerdasan Buatan	2024	31	2.957450
14	Teknik Biosistem	2024	31	2.957450
15	Teknik Geomatika	2024	31	2.957450
16	Biologi	2024	31	2.957450
17	Teknologi Pangan	2024	31	2.957450
18	Teknik Lingkungan - Sanitasi Daerah	2024	31	2.957450
19	Kimia Analitik	2024	31	2.957450
20	Rekayasa Tata Kelola Air Terpadu	2024	31	2.957450
21	Teknik Kelautan	2024	31	2.957450
22	Sains Atmosfer dan Keplanetan	2024	31	2.957450
23	DKV - Animasi & Media Digital	2024	31	2.957450
24	PWK - Lingkungan & Permukiman	2024	31	2.957450
25	Teknik Kimia	2024	31	2.957450
26	Perencanaan Wilayah dan Kota	2024	31	2.957450

✔ Query executed successfully. BITLANCKA\MSSQLSERVER01 (16... | BITLANCKA\Pongo (110) | akademik | 00:00:00 | Row: 1, Col: 1 | 53 rows



	ProgramName	Year	TotalStudents	AvgGrade
29	Teknik Geofisika	2024	31	2.957450
30	Teknik Perkeretaapian	2024	31	2.957450
31	Sains Data	2024	31	2.957450
32	Informatika - Cyber Security	2024	31	2.957450
33	Arsitektur Urban	2024	31	2.957450
34	Tahap Persiapan Bersama	2024	31	2.957450
35	Teknik Mesin	2024	31	2.957450
36	Teknologi Pangan - Keamanan Pangan	2024	31	2.957450
37	Teknik Industri	2024	31	2.957450
38	Sains Lingkungan Kelautan	2024	31	2.957450
39	Arsitektur	2024	31	2.957450
40	Fisika	2024	31	2.957450
41	DKV - Branding & Identitas Visual	2024	31	2.957450
42	Informatika - Software Development	2024	31	2.957450
43	Fisika Instrumentasi	2024	31	2.957450
44	Arsitektur Lanskap	2024	31	2.957450
45	Teknik Lingkungan	2024	31	2.957450
46	Matematika Terapan & Statistika	2024	31	2.957450
47	Desain Komunikasi Visual	2024	31	2.957450
48	Matematika	2024	31	2.957450
49	Teknik Sipil - Transportasi	2024	31	2.957450
50	Teknik Elektro	2024	31	2.957450
51	Teknik Mesin - Energi	2024	31	2.957450
52	Biologi Mikrobiologi	2024	31	2.957450
53	Fisika Material	2024	31	2.957450

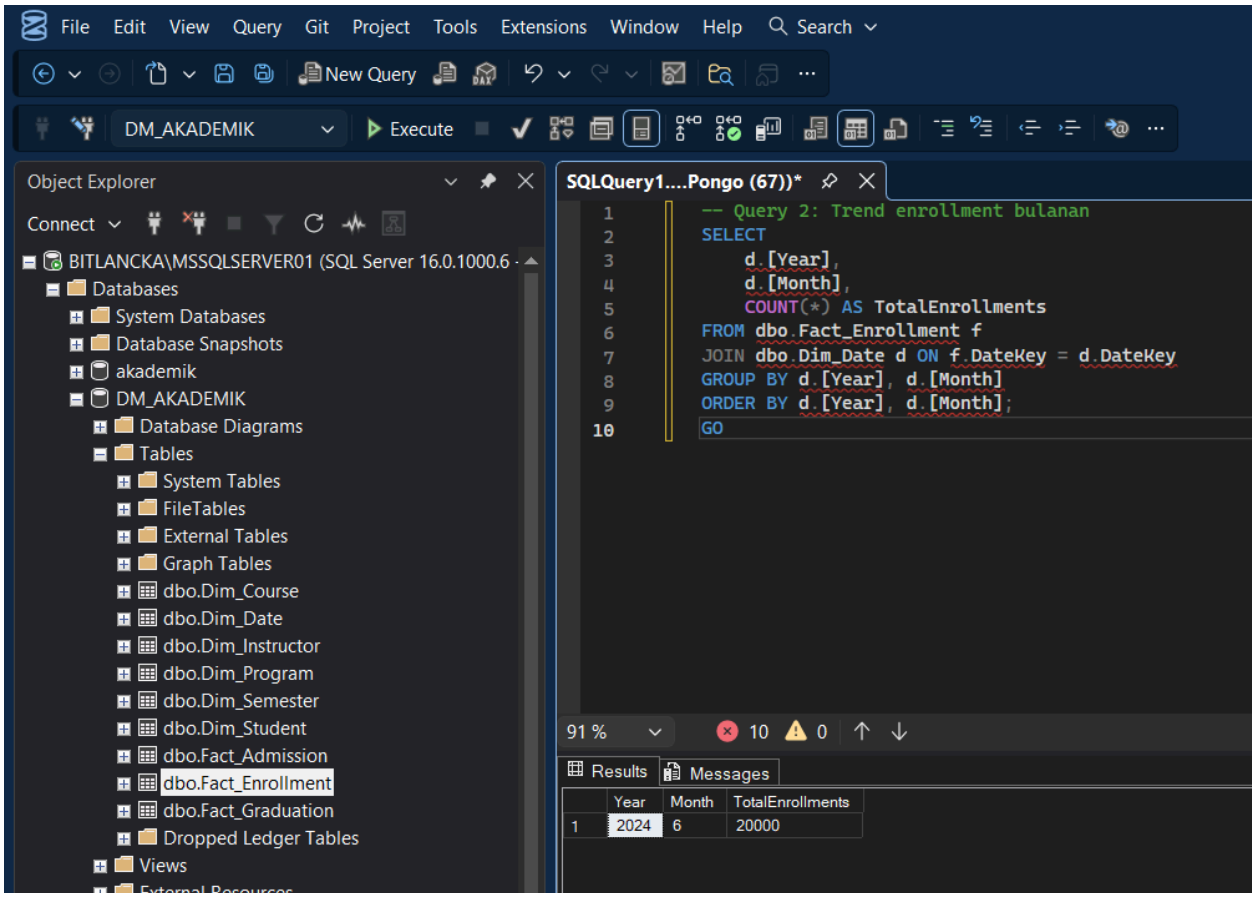
```
Query executed successfully.                BITLANCK
```


Results | Messages

	ProgramName	Year	TotalStudents	AvgGrade
14	Teknik Biosistem	2024	31	2.957450
15	Teknik Geomatika	2024	31	2.957450
16	Biologi	2024	31	2.957450
17	Teknologi Pangan	2024	31	2.957450
18	Teknik Lingkungan - Sanitasi Daerah	2024	31	2.957450
19	Kimia Analitik	2024	31	2.957450
20	Rekayasa Tata Kelola Air Terpadu	2024	31	2.957450
21	Teknik Kelautan	2024	31	2.957450
22	Sains Atmosfer dan Keplanetan	2024	31	2.957450
23	DKV - Animasi & Media Digital	2024	31	2.957450
24	PWK - Lingkungan & Permukiman	2024	31	2.957450
25	Teknik Kimia	2024	31	2.957450
26	Perencanaan Wilayah dan Kota	2024	31	2.957450
27	Kimia	2024	31	2.957450
28	Teknik Industri - Manufaktur	2024	31	2.957450
29	Teknik Geofisika	2024	31	2.957450
30	Teknik Perkeretaapian	2024	31	2.957450
31	Sains Data	2024	31	2.957450
32	Informatika - Cyber Security	2024	31	2.957450
33	Arsitektur Urban	2024	31	2.957450
34	Tahap Persiapan Bersama	2024	31	2.957450
35	Teknik Mesin	2024	31	2.957450
36	Teknologi Pangan - Keamanan Pangan	2024	31	2.957450
37	Teknik Industri	2024	31	2.957450
38	Sains Lingkungan Kelautan	2024	31	2.957450
39	Arsitektur	2024	31	2.957450

```
Query executed successfully.   BITLANCKA\MSSQLSERVER01 (16...   BITLANCKA\Pongo (110)   akademik   00:00:00   Row: 1, Col: 1   53 rows
```

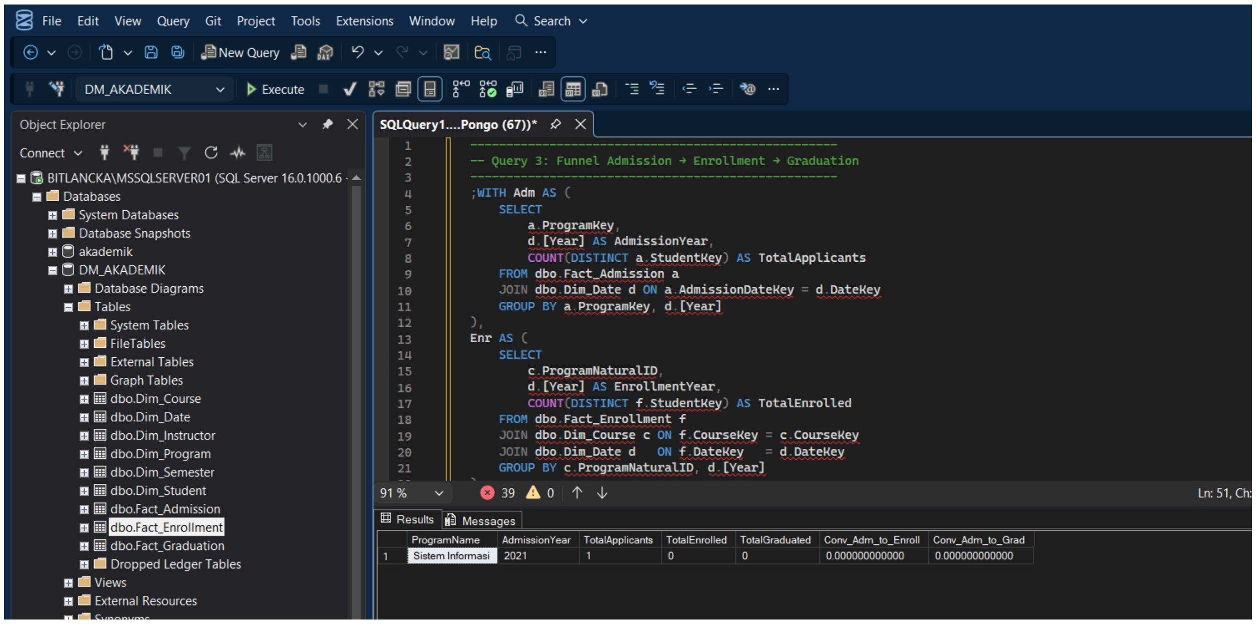
Query pertama digunakan untuk menghitung rata-rata nilai (NumericGrade) mahasiswa pada setiap program studi untuk setiap tahun akademik. Dengan menggabungkan tabel fact enrollment, dimensi tanggal, dan dimensi program, query ini membantu melihat *kualitas akademik rata-rata* tiap program dari tahun ke tahun. Hasilnya dapat digunakan untuk analisis performa program, tren peningkatan atau penurunan IP mahasiswa, serta menjadi indikator mutu pendidikan



```sql
-- Query 2: Trend enrollment bulanan
SELECT
    d.[Year],
    d.[Month],
    COUNT(*) AS TotalEnrollments
FROM dbo.Fact_Enrollment f
JOIN dbo.Dim_Date d ON f.DateKey = d.DateKey
GROUP BY d.[Year], d.[Month]
ORDER BY d.[Year], d.[Month];
GO
```

	Year	Month	TotalEnrollments
1	2024	6	20000

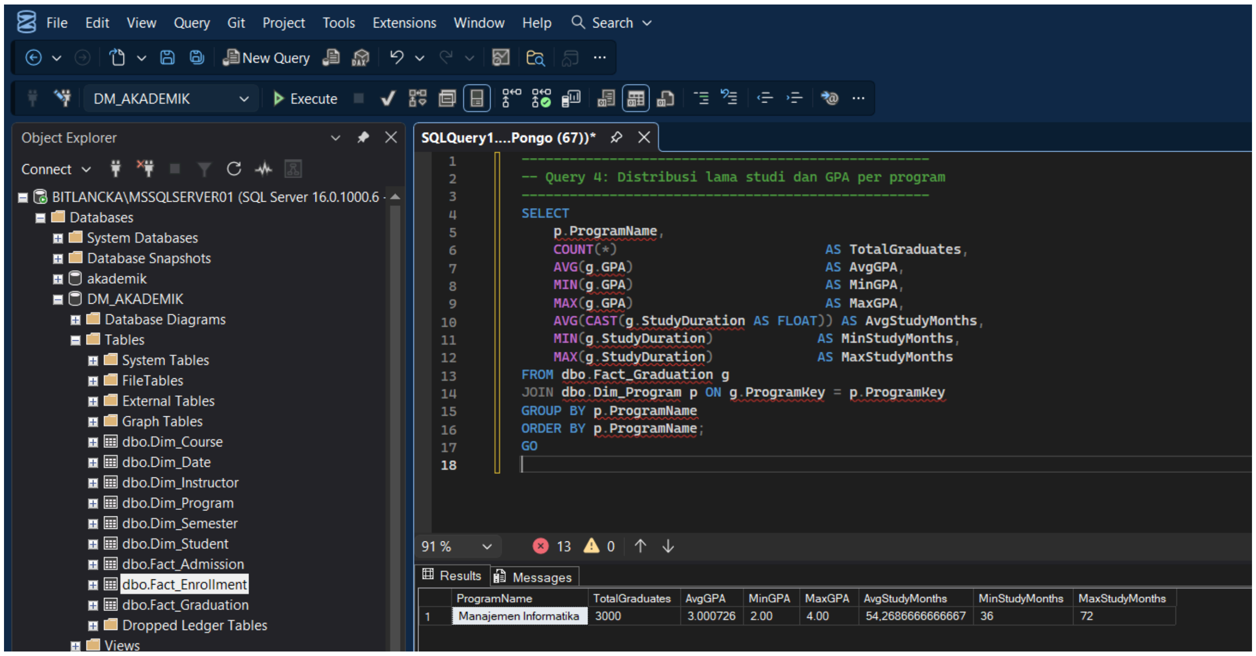
Query ini digunakan untuk melihat jumlah pendaftaran/enrollment per bulan pada setiap tahun. Data ini bermanfaat untuk menganalisis *pola musiman* pendaftaran mahasiswa, misalnya apakah ada lonjakan pada awal semester atau penurunan pada bulan tertentu. Dengan tren bulanan ini, institusi dapat melakukan perencanaan kapasitas kelas, evaluasi jadwal akademik, atau kebutuhan sumber daya.



```sql
--------------------------------------------------
-- Query 3: Funnel Admission → Enrollment → Graduation
--------------------------------------------------
;WITH Adm AS (
    SELECT
        a.ProgramKey,
        d.[Year] AS AdmissionYear,
        COUNT(DISTINCT a.StudentKey) AS TotalApplicants
    FROM dbo.Fact_Admission a
    JOIN dbo.Dim_Date d ON a.AdmissionDateKey = d.DateKey
    GROUP BY a.ProgramKey, d.[Year]
),
Enr AS (
    SELECT
        c.ProgramNaturalID,
        d.[Year] AS EnrollmentYear,
        COUNT(DISTINCT f.StudentKey) AS TotalEnrolled
    FROM dbo.Fact_Enrollment f
    JOIN dbo.Dim_Course c ON f.CourseKey = c.CourseKey
    JOIN dbo.Dim_Date d   ON f.DateKey   = d.DateKey
    GROUP BY c.ProgramNaturalID, d.[Year]
```

	ProgramName	AdmissionYear	TotalApplicants	TotalEnrolled	TotalGraduated	Conv_Adm_to_Enroll	Conv_Adm_to_Grad
1	Sistem Informasi	2021	1	0	0	0.000000000000	0.000000000000

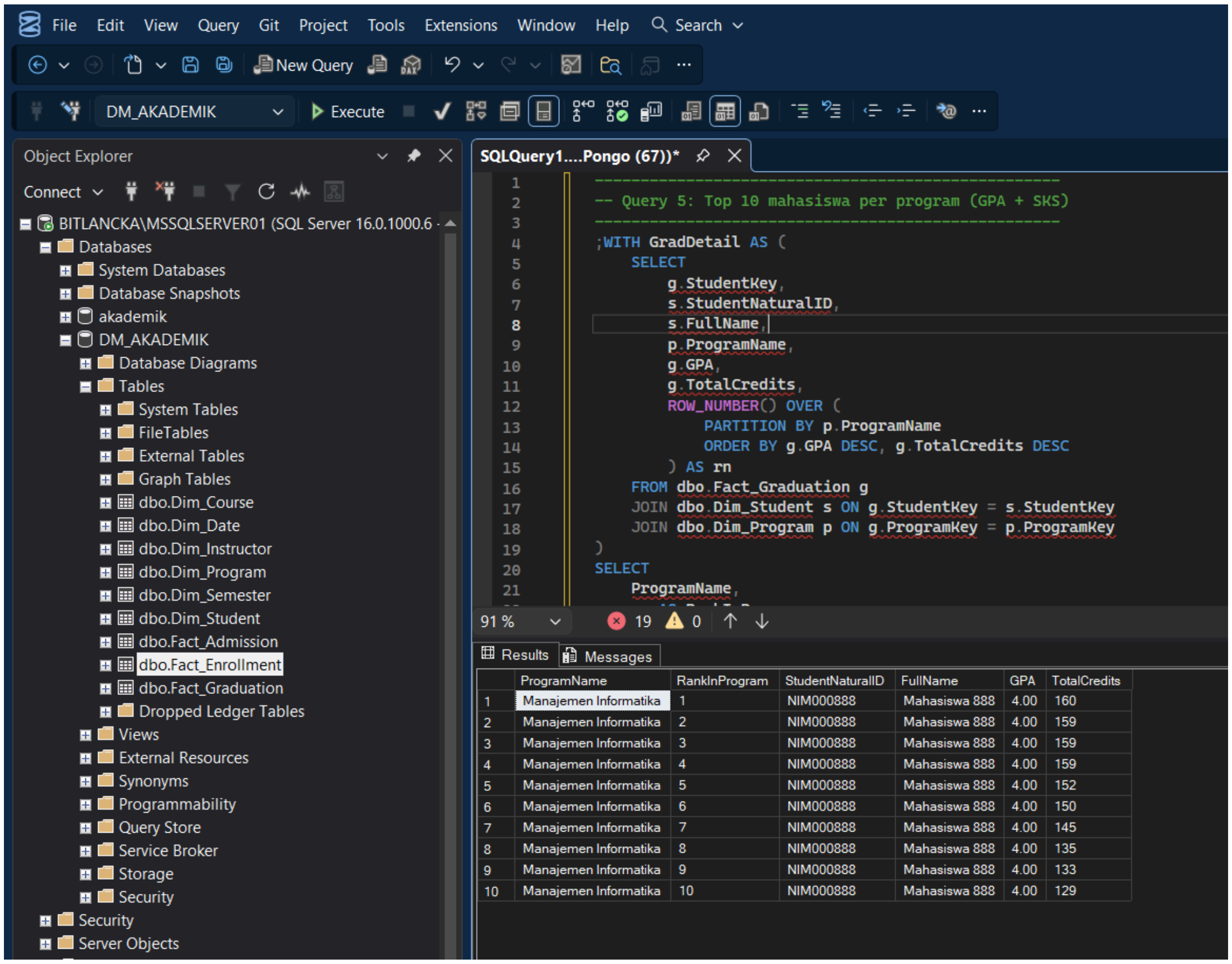
Query ketiga membangun *funnel analisis*, yaitu alur dari pendaftaran (admission) ke keterdaftaran dalam mata kuliah (enrollment) hingga kelulusan (graduation). Analisis funnel seperti ini sangat penting untuk melihat *rasio konversi* pada setiap tahapan pendidikan. Misalnya, berapa banyak dari pendaftar yang benar-benar masuk kuliah, dan berapa banyak dari mereka yang berhasil lulus. Data ini membantu mengukur efektivitas program studi serta menemukan potensi masalah seperti dropout.



```sql
--------------------------------------------------
-- Query 4: Distribusi lama studi dan GPA per program
--------------------------------------------------
SELECT
    p.ProgramName,
    COUNT(*)                              AS TotalGraduates,
    AVG(g.GPA)                            AS AvgGPA,
    MIN(g.GPA)                            AS MinGPA,
    MAX(g.GPA)                            AS MaxGPA,
    AVG(CAST(g.StudyDuration AS FLOAT)) AS AvgStudyMonths,
    MIN(g.StudyDuration)                 AS MinStudyMonths,
    MAX(g.StudyDuration)                 AS MaxStudyMonths
FROM dbo.Fact_Graduation g
JOIN dbo.Dim_Program p ON g.ProgramKey = p.ProgramKey
GROUP BY p.ProgramName
ORDER BY p.ProgramName;
GO
```

	ProgramName	TotalGraduates	AvgGPA	MinGPA	MaxGPA	AvgStudyMonths	MinStudyMonths	MaxStudyMonths
1	Manajemen Informatika	3000	3.000726	2.00	4.00	54,2686666666667	36	72

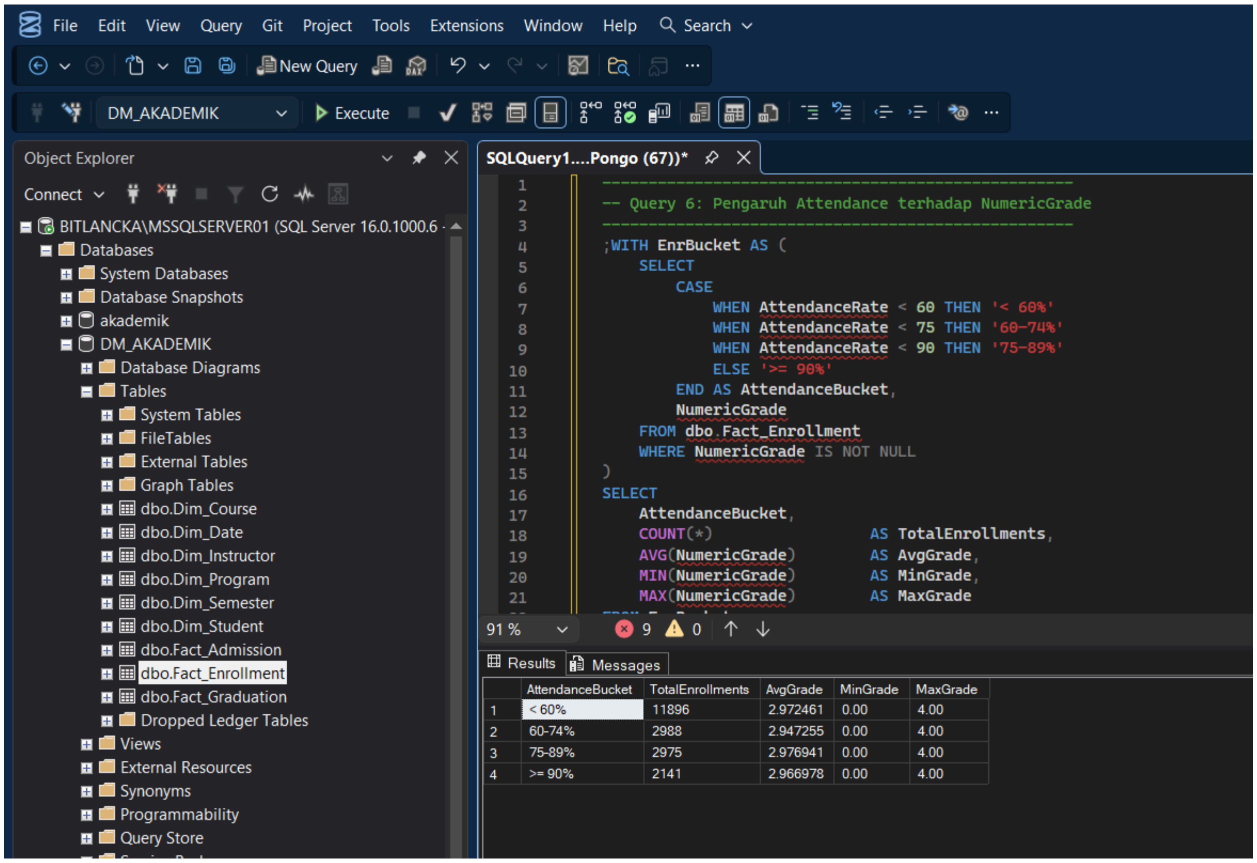
Query ini menghitung distribusi lama studi dan GPA (IPK) bagi mahasiswa yang telah lulus pada setiap program. Hasilnya mencakup jumlah lulusan, rata-rata IPK, IPK minimum-maksimum, serta lama studi rata-rata hingga lulus. Output ini penting untuk menganalisis *mutu lulusan*, konsistensi performa akademik, serta efektivitas kurikulum suatu program studi.



```sql
----------------------------------------------------------
-- Query 5: Top 10 mahasiswa per program (GPA + SKS)
----------------------------------------------------------
;WITH GradDetail AS (
    SELECT
        g.StudentKey,
        s.StudentNaturalID,
        s.FullName,
        p.ProgramName,
        g.GPA,
        g.TotalCredits,
        ROW_NUMBER() OVER (
            PARTITION BY p.ProgramName
            ORDER BY g.GPA DESC, g.TotalCredits DESC
        ) AS rn
    FROM dbo.Fact_Graduation g
    JOIN dbo.Dim_Student s ON g.StudentKey = s.StudentKey
    JOIN dbo.Dim_Program p ON g.ProgramKey = p.ProgramKey
)
SELECT
    ProgramName,
```

	ProgramName	RankInProgram	StudentNaturalID	FullName	GPA	TotalCredits
1	Manajemen Informatika	1	NIM000888	Mahasiswa 888	4.00	160
2	Manajemen Informatika	2	NIM000888	Mahasiswa 888	4.00	159
3	Manajemen Informatika	3	NIM000888	Mahasiswa 888	4.00	159
4	Manajemen Informatika	4	NIM000888	Mahasiswa 888	4.00	159
5	Manajemen Informatika	5	NIM000888	Mahasiswa 888	4.00	152
6	Manajemen Informatika	6	NIM000888	Mahasiswa 888	4.00	150
7	Manajemen Informatika	7	NIM000888	Mahasiswa 888	4.00	145
8	Manajemen Informatika	8	NIM000888	Mahasiswa 888	4.00	135
9	Manajemen Informatika	9	NIM000888	Mahasiswa 888	4.00	133
10	Manajemen Informatika	10	NIM000888	Mahasiswa 888	4.00	129

Query ini menyusun daftar 10 mahasiswa terbaik pada setiap program berdasarkan GPA dan jumlah SKS. Dengan menggunakan ranking per program, institusi dapat dengan mudah mengidentifikasi mahasiswa berprestasi, memberikan penghargaan, atau memanfaatkan data ini untuk laporan akreditasi. Query ini juga menunjukkan kualitas lulusan unggulan di masing-masing program.

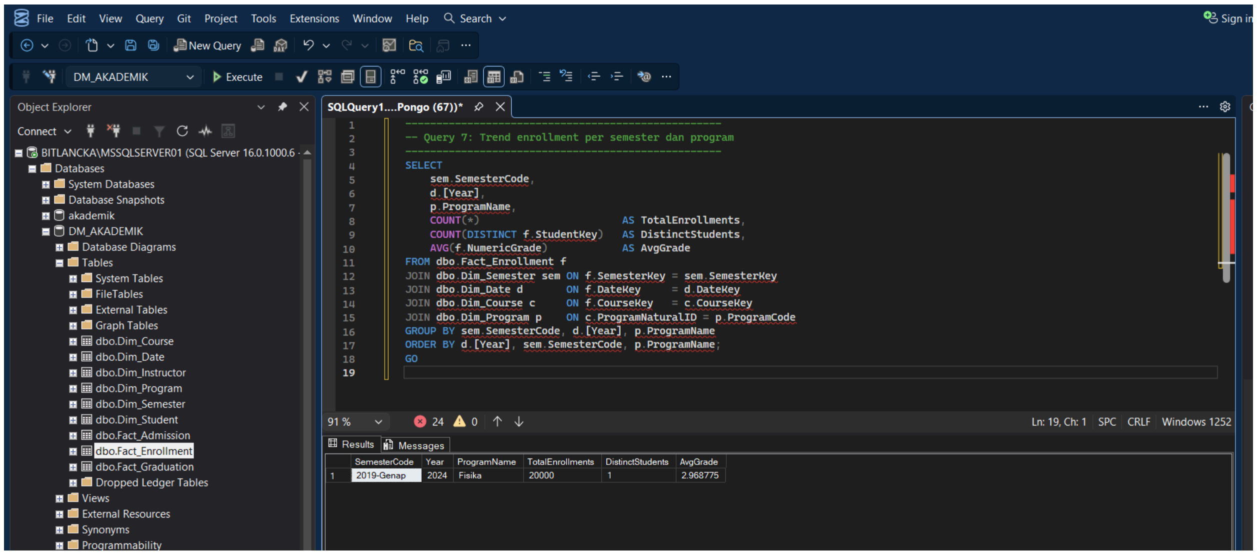


```sql
----------------------------------------------------
-- Query 6: Pengaruh Attendance terhadap NumericGrade
----------------------------------------------------
;WITH EnrBucket AS (
    SELECT
        CASE
            WHEN AttendanceRate < 60 THEN '< 60%'
            WHEN AttendanceRate < 75 THEN '60-74%'
            WHEN AttendanceRate < 90 THEN '75-89%'
            ELSE '>= 90%'
        END AS AttendanceBucket,
        NumericGrade
    FROM dbo.Fact_Enrollment
    WHERE NumericGrade IS NOT NULL
)
SELECT
    AttendanceBucket,
    COUNT(*)                AS TotalEnrollments,
    AVG(NumericGrade)       AS AvgGrade,
    MIN(NumericGrade)       AS MinGrade,
    MAX(NumericGrade)       AS MaxGrade
```

```
   AttendanceBucket  TotalEnrollments  AvgGrade  MinGrade  MaxGrade
1  < 60%             11896             2.972461  0.00      4.00
2  60-74%            2988              2.947255  0.00      4.00
3  75-89%            2975              2.976941  0.00      4.00
4  >= 90%            2141              2.966978  0.00      4.00
```

Query keenam mengelompokkan mahasiswa berdasarkan persentase kehadiran (AttendanceRate) ke dalam bucket (misalnya <60%, 60–74%, 75–89%, ≥90%). Kemudian dihitung rata-rata nilai, nilai minimum, dan maksimum pada tiap kelompok. Analisis ini bertujuan melihat apakah *tingkat kehadiran berpengaruh terhadap nilai*, dan biasanya menunjukkan korelasi positif antara kehadiran tinggi dan nilai lebih baik.

Query keenam mengelompokkan mahasiswa berdasarkan persentase kehadiran (AttendanceRate) ke dalam bucket (misalnya <60%, 60–74%, 75–89%, ≥90%). Kemudian dihitung rata-rata nilai, nilai minimum, dan maksimum pada tiap kelompok. Analisis ini bertujuan melihat apakah *tingkat kehadiran berpengaruh terhadap nilai*, dan biasanya menunjukkan korelasi positif antara kehadiran tinggi dan nilai lebih baik.



```sql
---------------------------------------------------
-- Query 7: Trend enrollment per semester dan program
---------------------------------------------------
SELECT
    sem.SemesterCode,
    d.[Year],
    p.ProgramName,
    COUNT(*)                          AS TotalEnrollments,
    COUNT(DISTINCT f.StudentKey)      AS DistinctStudents,
    AVG(f.NumericGrade)               AS AvgGrade
FROM dbo.Fact_Enrollment f
JOIN dbo.Dim_Semester sem ON f.SemesterKey = sem.SemesterKey
JOIN dbo.Dim_Date d       ON f.DateKey     = d.DateKey
JOIN dbo.Dim_Course c     ON f.CourseKey   = c.CourseKey
JOIN dbo.Dim_Program p    ON c.ProgramNaturalID = p.ProgramCode
GROUP BY sem.SemesterCode, d.[Year], p.ProgramName
ORDER BY d.[Year], sem.SemesterCode, p.ProgramName;
GO
```

	SemesterCode	Year	ProgramName	TotalEnrollments	DistinctStudents	AvgGrade
1	2019-Genap	2024	Fisika	20000	1	2.968775

Query terakhir memberikan gambaran jumlah enrollment, jumlah mahasiswa unik, dan rata-rata nilai pada setiap semester untuk tiap program studi. Data ini membantu memantau perkembangan jumlah peserta kelas per semester, mengidentifikasi perubahan jumlah mahasiswa, dan menganalisis performa akademik berdasarkan semester. Informasi ini bermanfaat untuk perencanaan akademik dan monitoring kapasitas program.