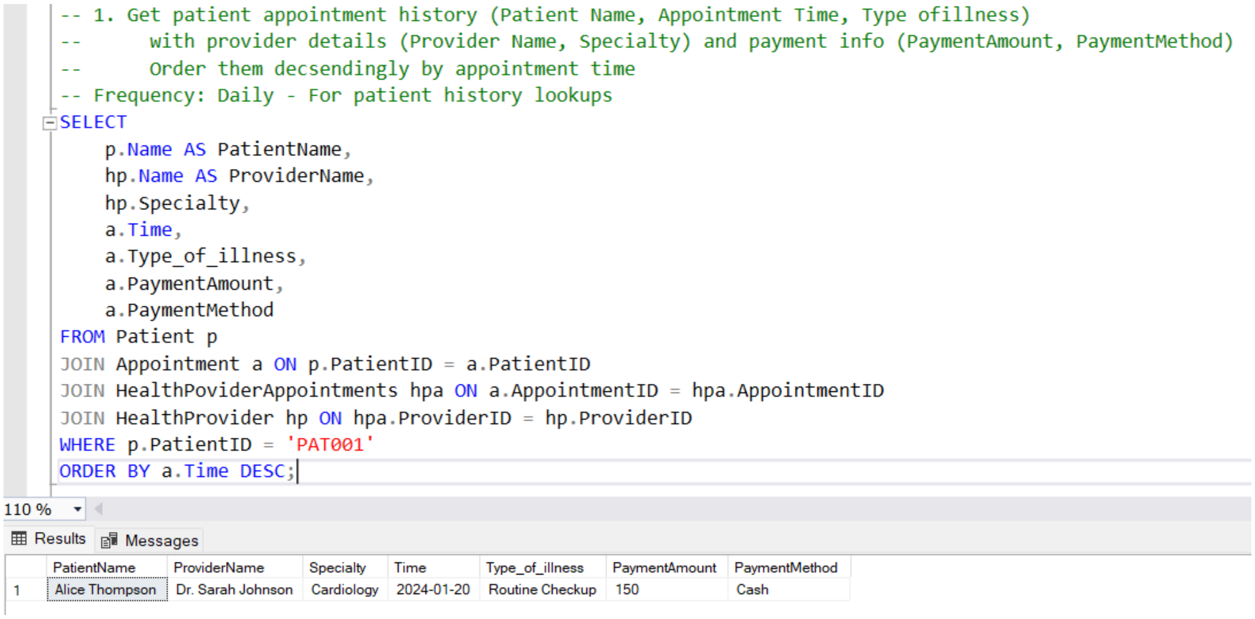
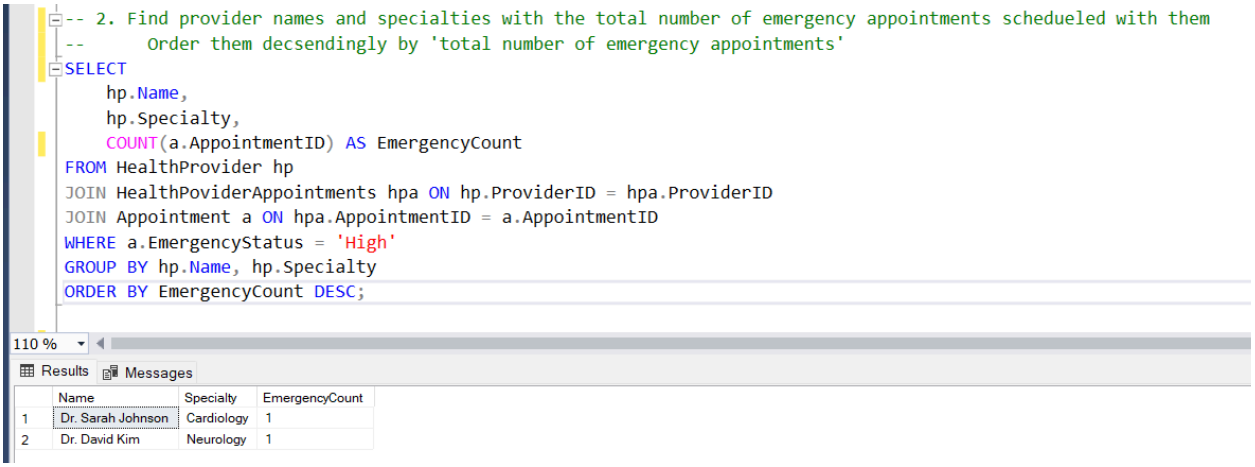
MOHAMED MOSTAFA | 22-101203



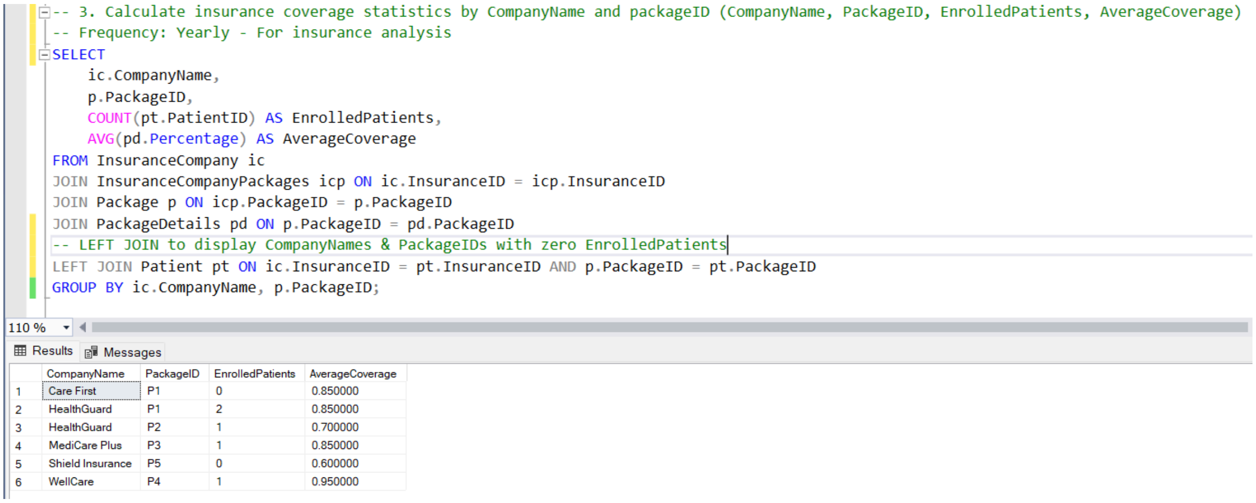
```sql
-- 1. Get patient appointment history (Patient Name, Appointment Time, Type ofillness)
--       with provider details (Provider Name, Specialty) and payment info (PaymentAmount, PaymentMethod)
--       Order them decsendingly by appointment time
-- Frequency: Daily - For patient history lookups
SELECT
    p.Name AS PatientName,
    hp.Name AS ProviderName,
    hp.Specialty,
    a.Time,
    a.Type_of_illness,
    a.PaymentAmount,
    a.PaymentMethod
FROM Patient p
JOIN Appointment a ON p.PatientID = a.PatientID
JOIN HealthPoviderAppointments hpa ON a.AppointmentID = hpa.AppointmentID
JOIN HealthProvider hp ON hpa.ProviderID = hp.ProviderID
WHERE p.PatientID = 'PAT001'
ORDER BY a.Time DESC;
```

	PatientName	ProviderName	Specialty	Time	Type_of_illness	PaymentAmount	PaymentMethod
1	Alice Thompson	Dr. Sarah Johnson	Cardiology	2024-01-20	Routine Checkup	150	Cash



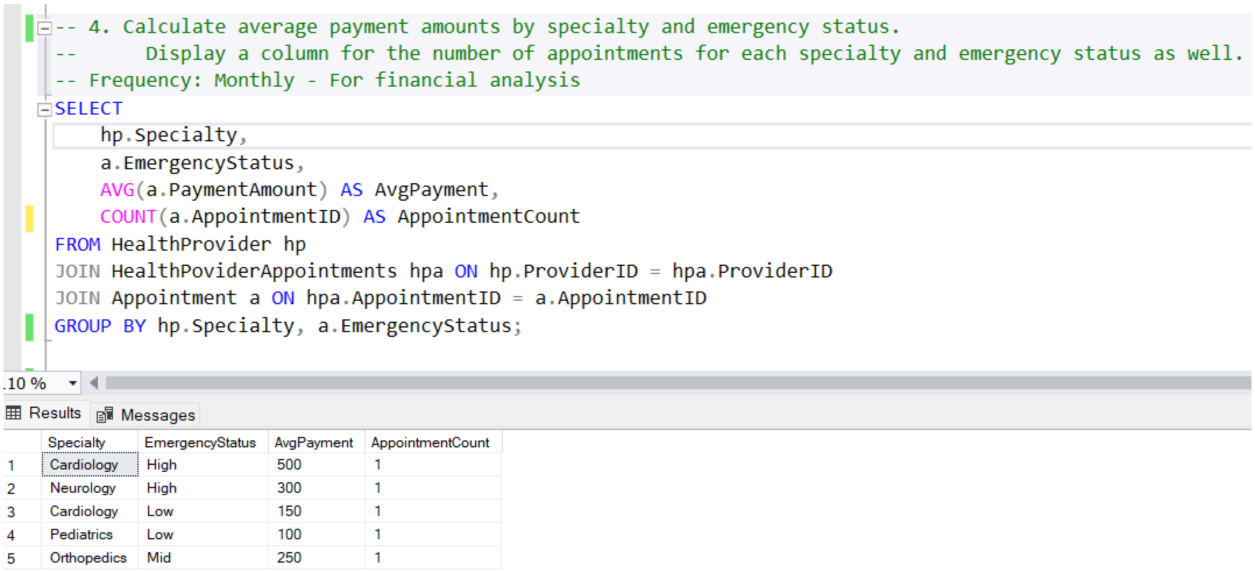
```sql
-- 2. Find provider names and specialties with the total number of emergency appointments schedueled with them
--       Order them decsendingly by 'total number of emergency appointments'
SELECT
    hp.Name,
    hp.Specialty,
    COUNT(a.AppointmentID) AS EmergencyCount
FROM HealthProvider hp
JOIN HealthPoviderAppointments hpa ON hp.ProviderID = hpa.ProviderID
JOIN Appointment a ON hpa.AppointmentID = a.AppointmentID
WHERE a.EmergencyStatus = 'High'
GROUP BY hp.Name, hp.Specialty
ORDER BY EmergencyCount DESC;
```

	Name	Specialty	EmergencyCount
1	Dr. Sarah Johnson	Cardiology	1
2	Dr. David Kim	Neurology	1



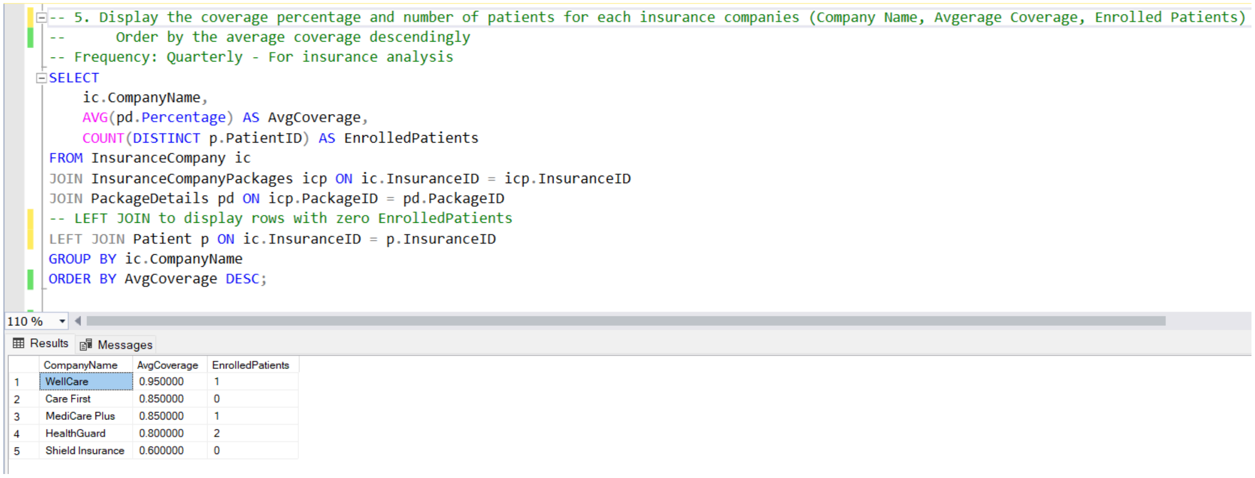
```sql
-- 3. Calculate insurance coverage statistics by CompanyName and packageID (CompanyName, PackageID, EnrolledPatients, AverageCoverage)
-- Frequency: Yearly - For insurance analysis
SELECT
    ic.CompanyName,
    p.PackageID,
    COUNT(pt.PatientID) AS EnrolledPatients,
    AVG(pd.Percentage) AS AverageCoverage
FROM InsuranceCompany ic
JOIN InsuranceCompanyPackages icp ON ic.InsuranceID = icp.InsuranceID
JOIN Package p ON icp.PackageID = p.PackageID
JOIN PackageDetails pd ON p.PackageID = pd.PackageID
-- LEFT JOIN to display CompanyNames & PackageIDs with zero EnrolledPatients
LEFT JOIN Patient pt ON ic.InsuranceID = pt.InsuranceID AND p.PackageID = pt.PackageID
GROUP BY ic.CompanyName, p.PackageID;
```

	CompanyName	PackageID	EnrolledPatients	AverageCoverage
1	Care First	P1	0	0.850000
2	HealthGuard	P1	2	0.850000
3	HealthGuard	P2	1	0.700000
4	MediCare Plus	P3	1	0.850000
5	Shield Insurance	P5	0	0.600000
6	WellCare	P4	1	0.950000



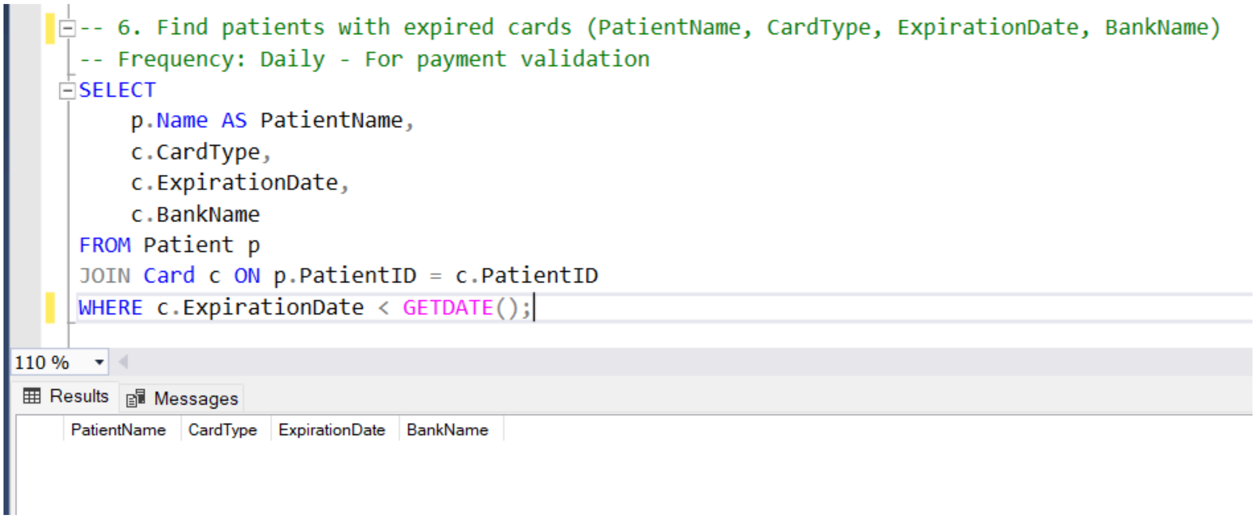
```sql
-- 4. Calculate average payment amounts by specialty and emergency status.
--      Display a column for the number of appointments for each specialty and emergency status as well.
-- Frequency: Monthly - For financial analysis
SELECT
    hp.Specialty,
    a.EmergencyStatus,
    AVG(a.PaymentAmount) AS AvgPayment,
    COUNT(a.AppointmentID) AS AppointmentCount
FROM HealthProvider hp
JOIN HealthPoviderAppointments hpa ON hp.ProviderID = hpa.ProviderID
JOIN Appointment a ON hpa.AppointmentID = a.AppointmentID
GROUP BY hp.Specialty, a.EmergencyStatus;
```

	Specialty	EmergencyStatus	AvgPayment	AppointmentCount
1	Cardiology	High	500	1
2	Neurology	High	300	1
3	Cardiology	Low	150	1
4	Pediatrics	Low	100	1
5	Orthopedics	Mid	250	1



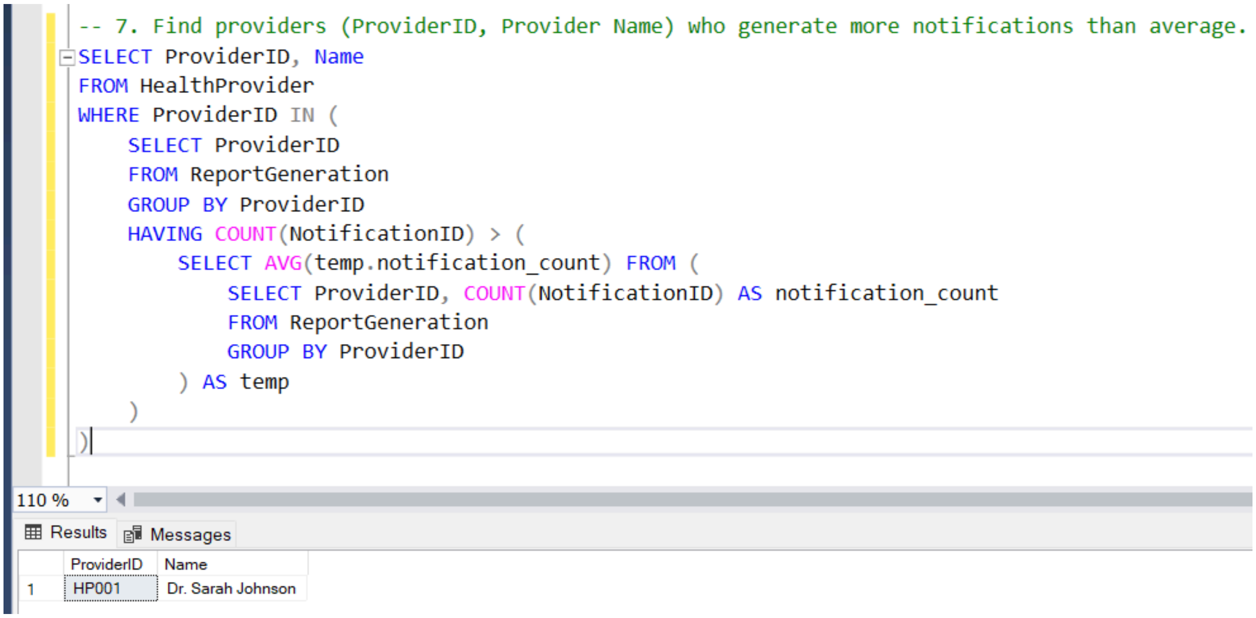
```sql
-- 5. Display the coverage percentage and number of patients for each insurance companies (Company Name, Avgerage Coverage, Enrolled Patients)
--      Order by the average coverage descendingly
-- Frequency: Quarterly - For insurance analysis
SELECT
    ic.CompanyName,
    AVG(pd.Percentage) AS AvgCoverage,
    COUNT(DISTINCT p.PatientID) AS EnrolledPatients
FROM InsuranceCompany ic
JOIN InsuranceCompanyPackages icp ON ic.InsuranceID = icp.InsuranceID
JOIN PackageDetails pd ON icp.PackageID = pd.PackageID
-- LEFT JOIN to display rows with zero EnrolledPatients
LEFT JOIN Patient p ON ic.InsuranceID = p.InsuranceID
GROUP BY ic.CompanyName
ORDER BY AvgCoverage DESC;
```

	CompanyName	AvgCoverage	EnrolledPatients
1	WellCare	0.950000	1
2	Care First	0.850000	0
3	MediCare Plus	0.850000	1
4	HealthGuard	0.800000	2
5	Shield Insurance	0.600000	0



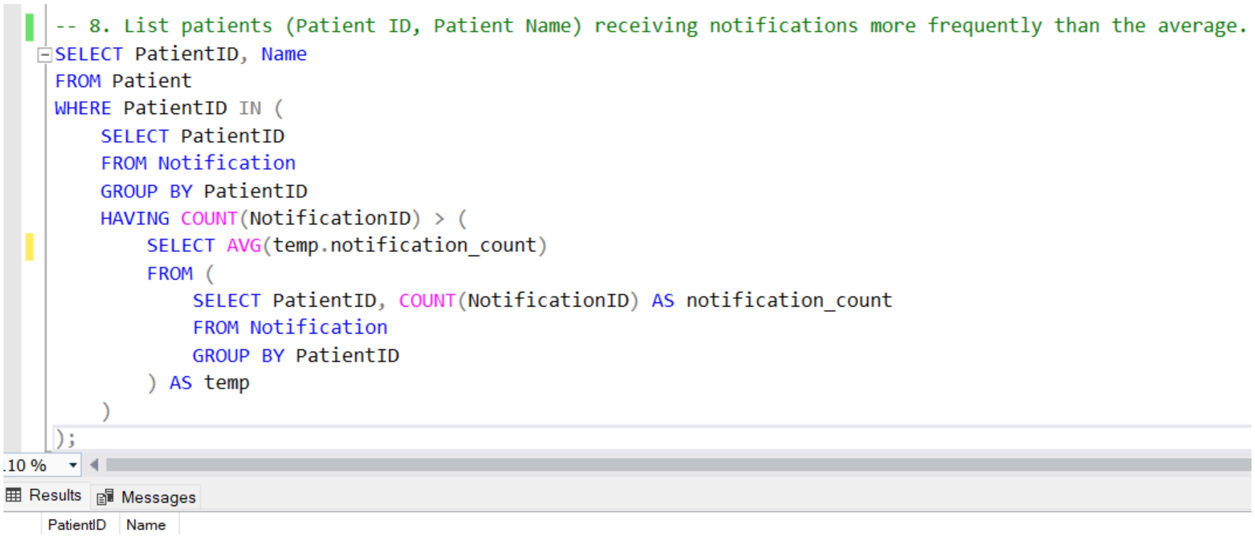
```sql
-- 6. Find patients with expired cards (PatientName, CardType, ExpirationDate, BankName)
-- Frequency: Daily - For payment validation
SELECT
    p.Name AS PatientName,
    c.CardType,
    c.ExpirationDate,
    c.BankName
FROM Patient p
JOIN Card c ON p.PatientID = c.PatientID
WHERE c.ExpirationDate < GETDATE();
```

PatientName	CardType	ExpirationDate	BankName



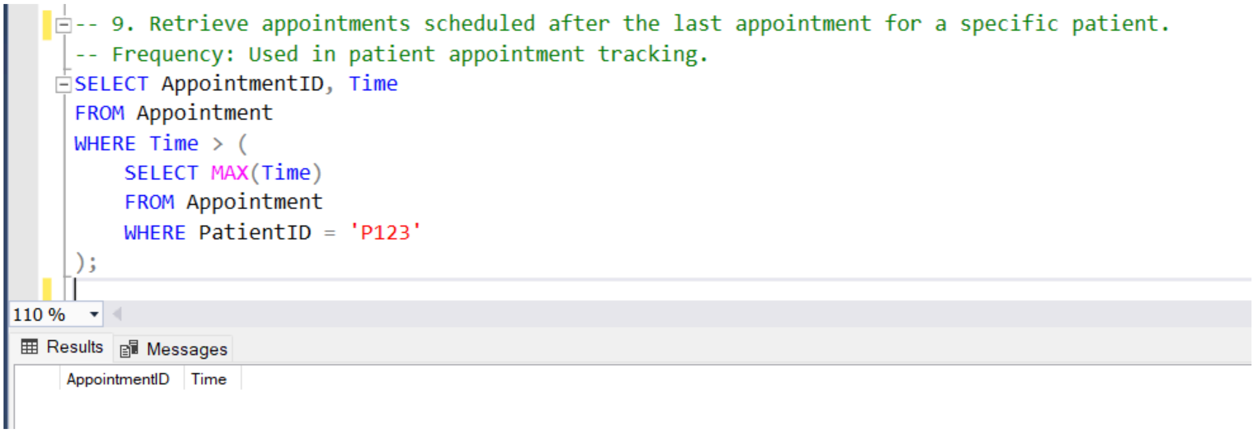
```sql
-- 7. Find providers (ProviderID, Provider Name) who generate more notifications than average.
SELECT ProviderID, Name
FROM HealthProvider
WHERE ProviderID IN (
    SELECT ProviderID
    FROM ReportGeneration
    GROUP BY ProviderID
    HAVING COUNT(NotificationID) > (
        SELECT AVG(temp.notification_count) FROM (
            SELECT ProviderID, COUNT(NotificationID) AS notification_count
            FROM ReportGeneration
            GROUP BY ProviderID
        ) AS temp
    )
)
```

	ProviderID	Name
1	HP001	Dr. Sarah Johnson



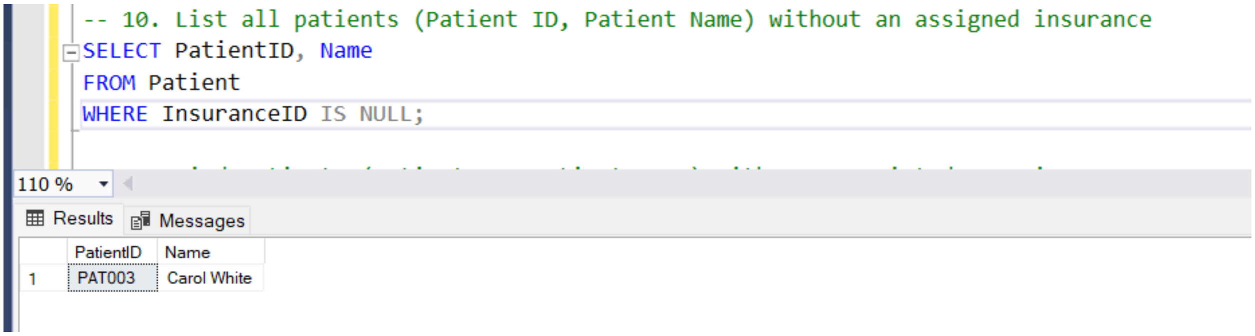
```sql
-- 8. List patients (Patient ID, Patient Name) receiving notifications more frequently than the average.
SELECT PatientID, Name
FROM Patient
WHERE PatientID IN (
    SELECT PatientID
    FROM Notification
    GROUP BY PatientID
    HAVING COUNT(NotificationID) > (
        SELECT AVG(temp.notification_count)
        FROM (
            SELECT PatientID, COUNT(NotificationID) AS notification_count
            FROM Notification
            GROUP BY PatientID
        ) AS temp
    )
);
```

PatientID	Name



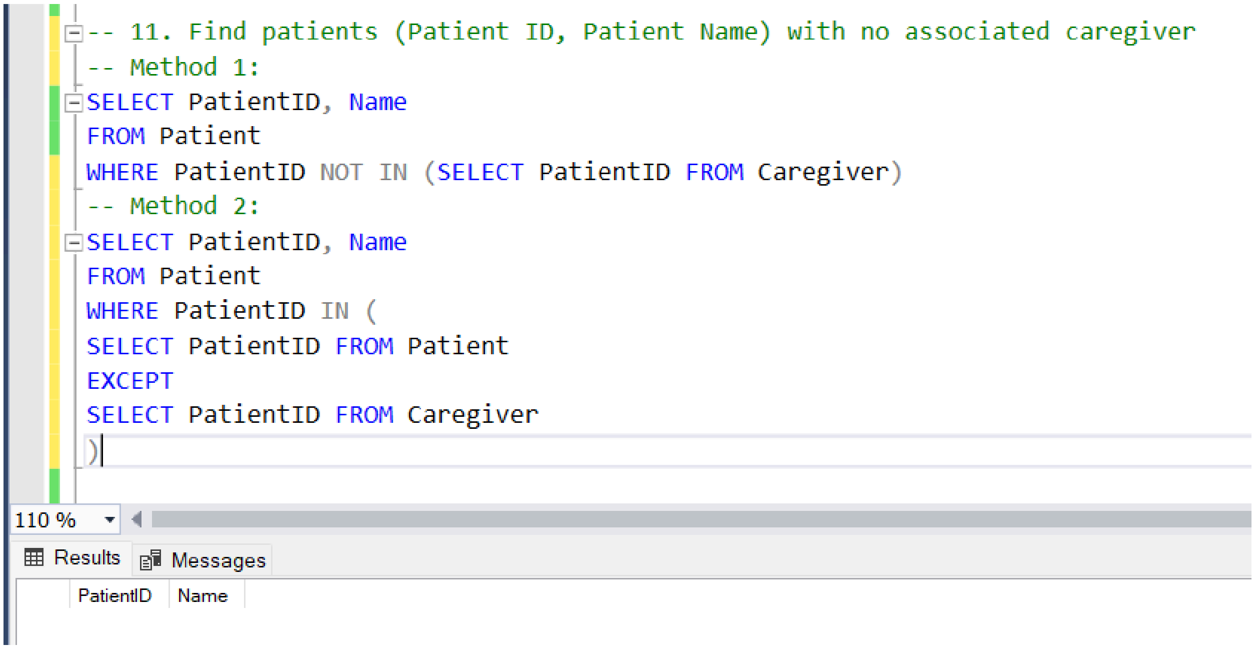
```sql
-- 9. Retrieve appointments scheduled after the last appointment for a specific patient.
-- Frequency: Used in patient appointment tracking.
SELECT AppointmentID, Time
FROM Appointment
WHERE Time > (
    SELECT MAX(Time)
    FROM Appointment
    WHERE PatientID = 'P123'
);
```

AppointmentID	Time



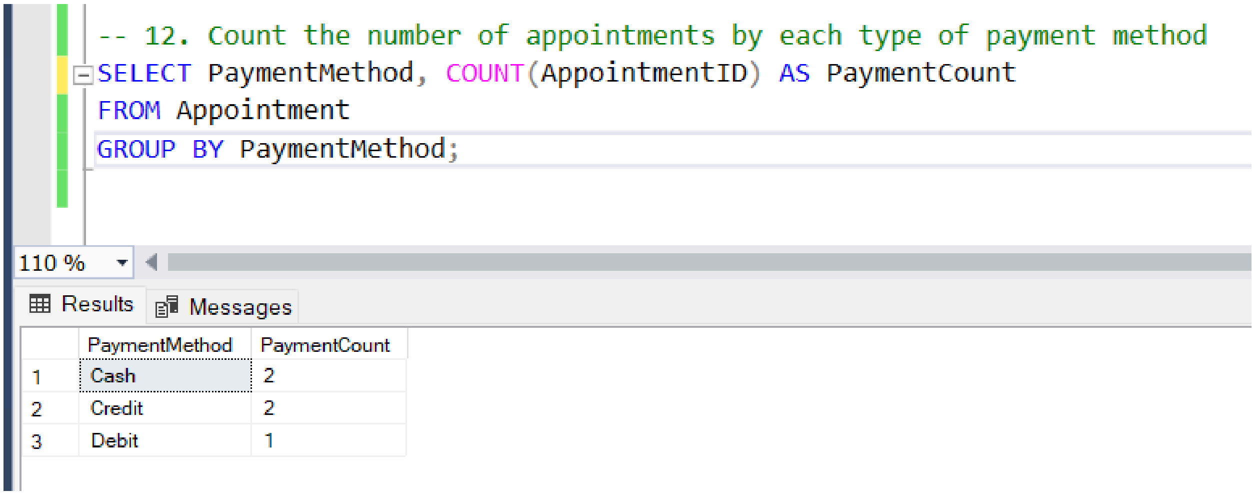
```sql
-- 10. List all patients (Patient ID, Patient Name) without an assigned insurance
SELECT PatientID, Name
FROM Patient
WHERE InsuranceID IS NULL;
```

	PatientID	Name
1	PAT003	Carol White



```sql
-- 11. Find patients (Patient ID, Patient Name) with no associated caregiver
-- Method 1:
SELECT PatientID, Name
FROM Patient
WHERE PatientID NOT IN (SELECT PatientID FROM Caregiver)
-- Method 2:
SELECT PatientID, Name
FROM Patient
WHERE PatientID IN (
SELECT PatientID FROM Patient
EXCEPT
SELECT PatientID FROM Caregiver
)
```

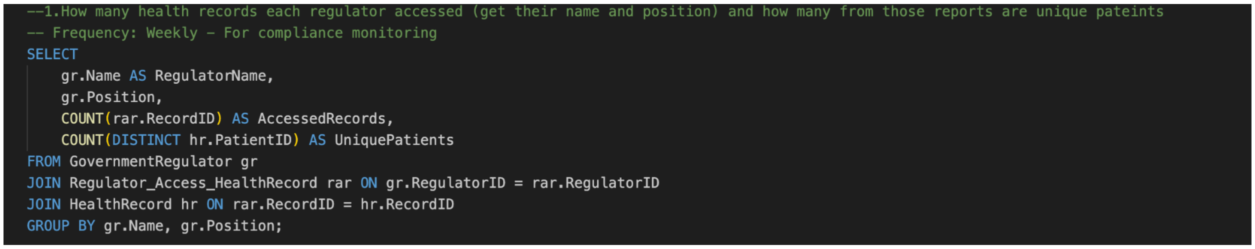
	PatientID	Name



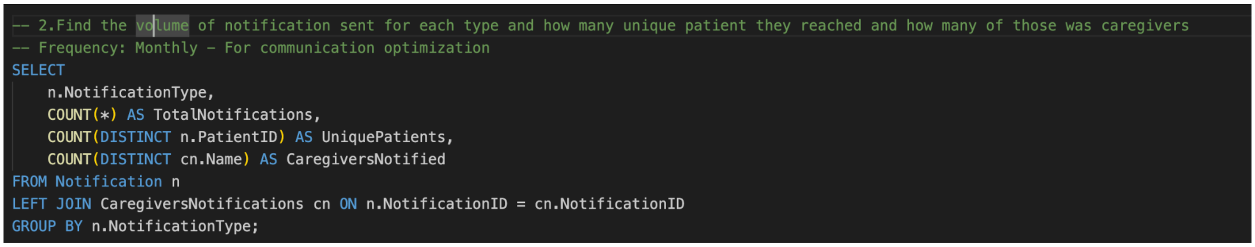
```sql
-- 12. Count the number of appointments by each type of payment method
SELECT PaymentMethod, COUNT(AppointmentID) AS PaymentCount
FROM Appointment
GROUP BY PaymentMethod;
```

	PaymentMethod	PaymentCount
1	Cash	2
2	Credit	2
3	Debit	1

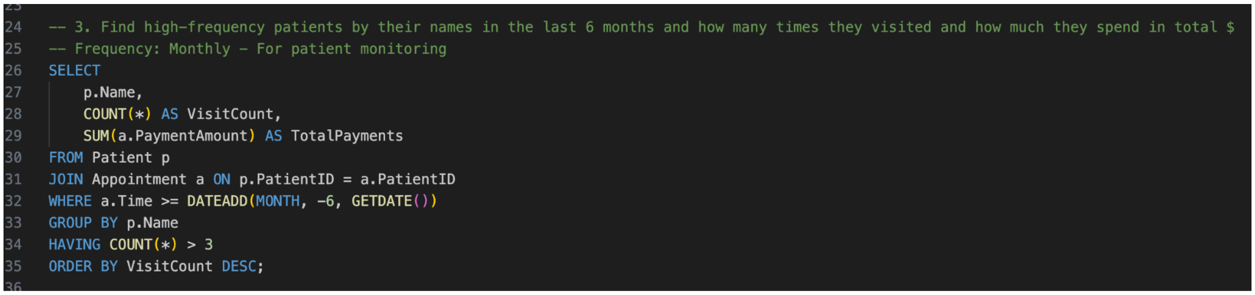
YOUSSEF WALID | 22-101048



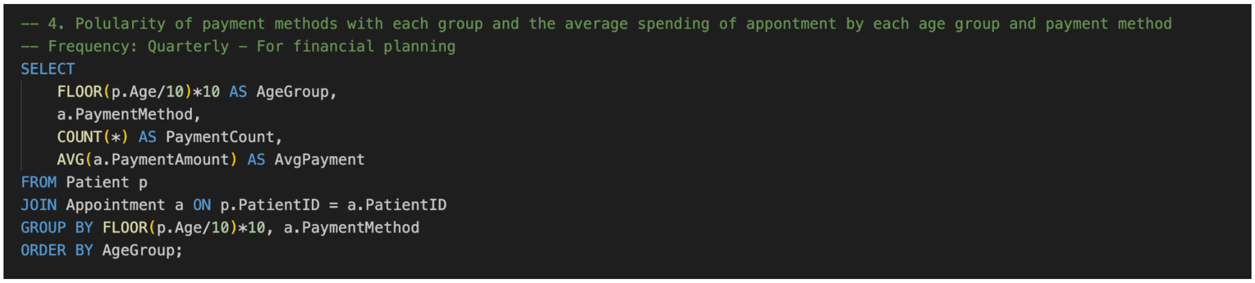
```sql
--1.How many health records each regulator accessed (get their name and position) and how many from those reports are unique pateints
-- Frequency: Weekly - For compliance monitoring
SELECT
    gr.Name AS RegulatorName,
    gr.Position,
    COUNT(rar.RecordID) AS AccessedRecords,
    COUNT(DISTINCT hr.PatientID) AS UniquePatients
FROM GovernmentRegulator gr
JOIN Regulator_Access_HealthRecord rar ON gr.RegulatorID = rar.RegulatorID
JOIN HealthRecord hr ON rar.RecordID = hr.RecordID
GROUP BY gr.Name, gr.Position;
```



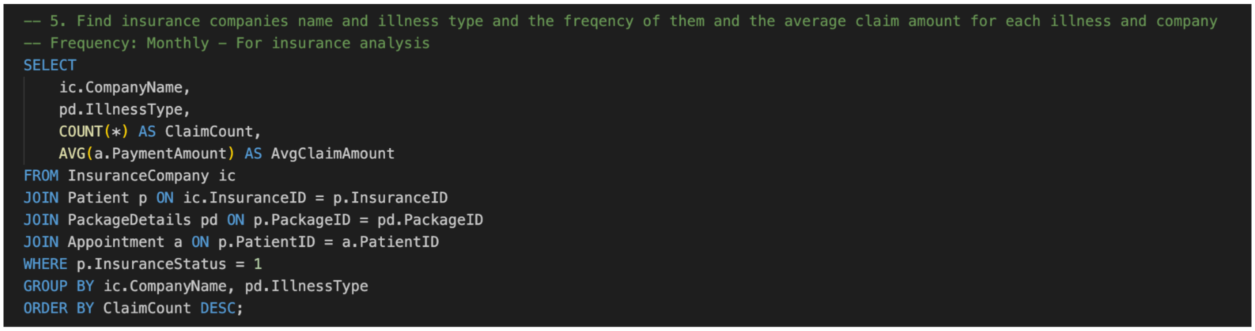
```sql
-- 2.Find the volume of notification sent for each type and how many unique patient they reached and how many of those was caregivers
-- Frequency: Monthly - For communication optimization
SELECT
    n.NotificationType,
    COUNT(*) AS TotalNotifications,
    COUNT(DISTINCT n.PatientID) AS UniquePatients,
    COUNT(DISTINCT cn.Name) AS CaregiversNotified
FROM Notification n
LEFT JOIN CaregiversNotifications cn ON n.NotificationID = cn.NotificationID
GROUP BY n.NotificationType;
```



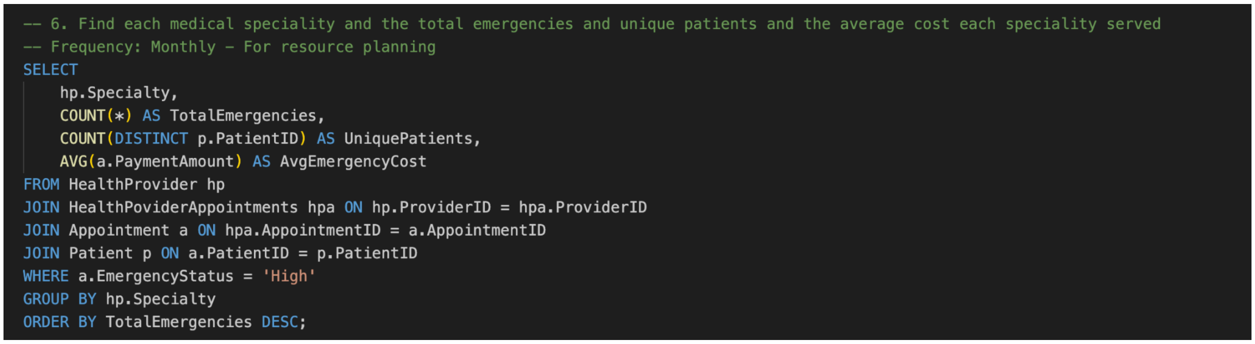
```sql
-- 3. Find high-frequency patients by their names in the last 6 months and how many times they visited and how much they spend in total $
-- Frequency: Monthly - For patient monitoring
SELECT
    p.Name,
    COUNT(*) AS VisitCount,
    SUM(a.PaymentAmount) AS TotalPayments
FROM Patient p
JOIN Appointment a ON p.PatientID = a.PatientID
WHERE a.Time >= DATEADD(MONTH, -6, GETDATE())
GROUP BY p.Name
HAVING COUNT(*) > 3
ORDER BY VisitCount DESC;
```



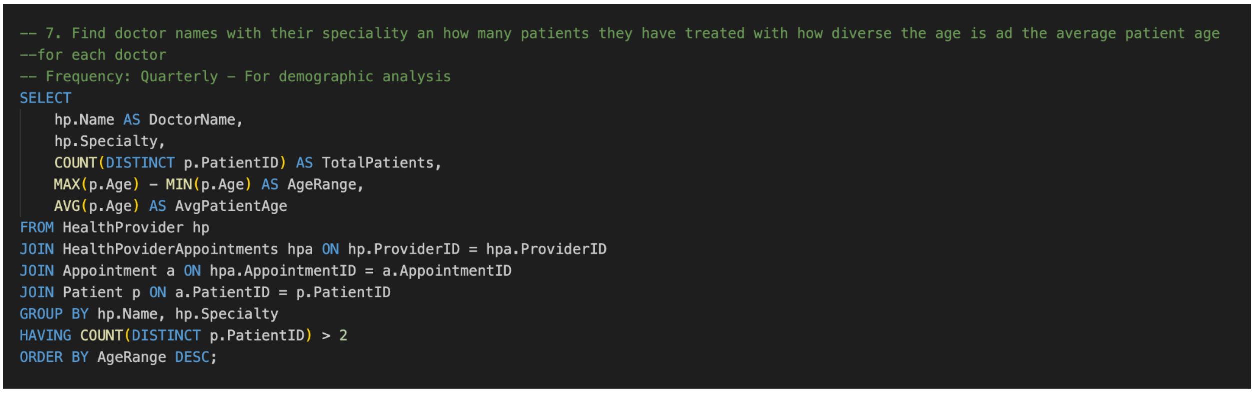
```sql
-- 4. Polularity of payment methods with each group and the average spending of apporntment by each age group and payment method
-- Frequency: Quarterly - For financial planning
SELECT
    FLOOR(p.Age/10)*10 AS AgeGroup,
    a.PaymentMethod,
    COUNT(*) AS PaymentCount,
    AVG(a.PaymentAmount) AS AvgPayment
FROM Patient p
JOIN Appointment a ON p.PatientID = a.PatientID
GROUP BY FLOOR(p.Age/10)*10, a.PaymentMethod
ORDER BY AgeGroup;
```



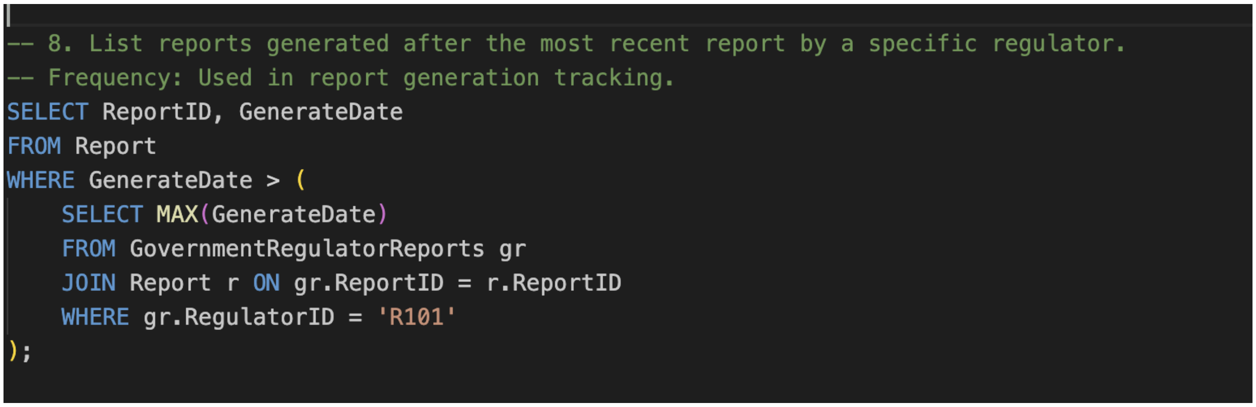
```sql
-- 5. Find insurance companies name and illness type and the freqency of them and the average claim amount for each illness and company
-- Frequency: Monthly - For insurance analysis
SELECT
    ic.CompanyName,
    pd.IllnessType,
    COUNT(*) AS ClaimCount,
    AVG(a.PaymentAmount) AS AvgClaimAmount
FROM InsuranceCompany ic
JOIN Patient p ON ic.InsuranceID = p.InsuranceID
JOIN PackageDetails pd ON p.PackageID = pd.PackageID
JOIN Appointment a ON p.PatientID = a.PatientID
WHERE p.InsuranceStatus = 1
GROUP BY ic.CompanyName, pd.IllnessType
ORDER BY ClaimCount DESC;
```



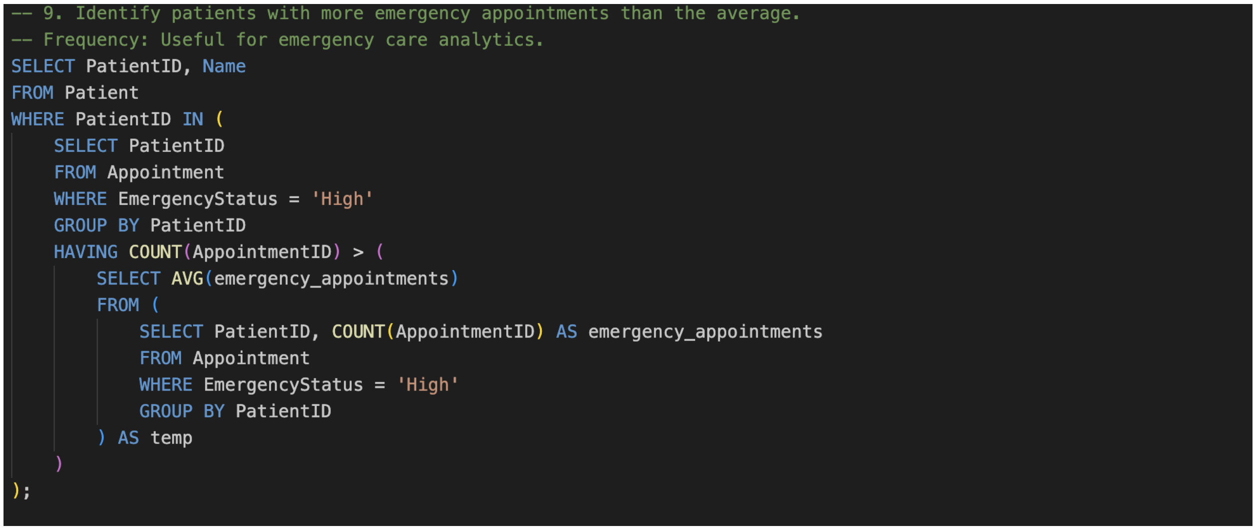
```sql
-- 6. Find each medical speciality and the total emergencies and unique patients and the average cost each speciality served
-- Frequency: Monthly - For resource planning
SELECT
    hp.Specialty,
    COUNT(*) AS TotalEmergencies,
    COUNT(DISTINCT p.PatientID) AS UniquePatients,
    AVG(a.PaymentAmount) AS AvgEmergencyCost
FROM HealthProvider hp
JOIN HealthPoviderAppointments hpa ON hp.ProviderID = hpa.ProviderID
JOIN Appointment a ON hpa.AppointmentID = a.AppointmentID
JOIN Patient p ON a.PatientID = p.PatientID
WHERE a.EmergencyStatus = 'High'
GROUP BY hp.Specialty
ORDER BY TotalEmergencies DESC;
```



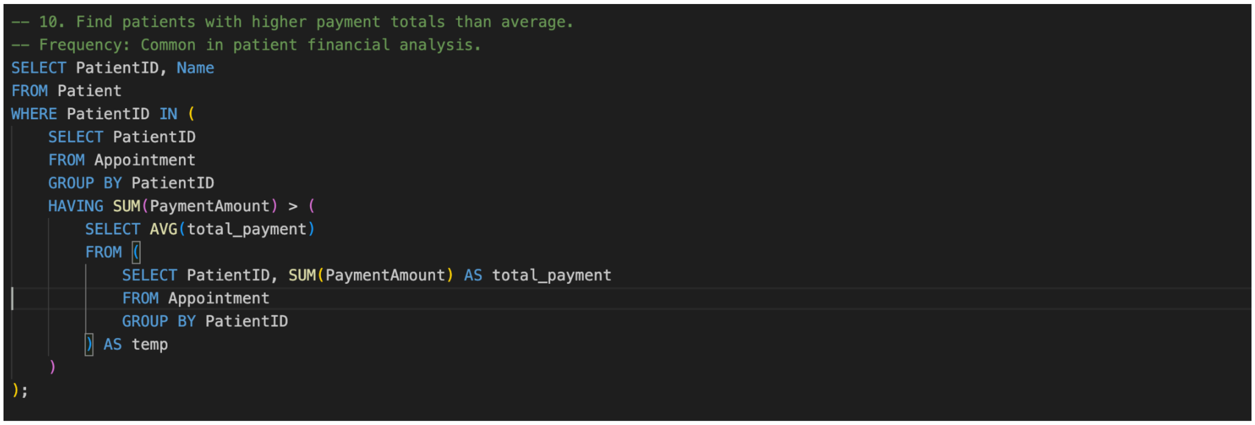
```sql
-- 7. Find doctor names with their speciality an how many patients they have treated with how diverse the age is ad the average patient age
--for each doctor
-- Frequency: Quarterly - For demographic analysis
SELECT
    hp.Name AS DoctorName,
    hp.Specialty,
    COUNT(DISTINCT p.PatientID) AS TotalPatients,
    MAX(p.Age) - MIN(p.Age) AS AgeRange,
    AVG(p.Age) AS AvgPatientAge
FROM HealthProvider hp
JOIN HealthPoviderAppointments hpa ON hp.ProviderID = hpa.ProviderID
JOIN Appointment a ON hpa.AppointmentID = a.AppointmentID
JOIN Patient p ON a.PatientID = p.PatientID
GROUP BY hp.Name, hp.Specialty
HAVING COUNT(DISTINCT p.PatientID) > 2
ORDER BY AgeRange DESC;
```



```sql
-- 8. List reports generated after the most recent report by a specific regulator.
-- Frequency: Used in report generation tracking.
SELECT ReportID, GenerateDate
FROM Report
WHERE GenerateDate > (
    SELECT MAX(GenerateDate)
    FROM GovernmentRegulatorReports gr
    JOIN Report r ON gr.ReportID = r.ReportID
    WHERE gr.RegulatorID = 'R101'
);
```

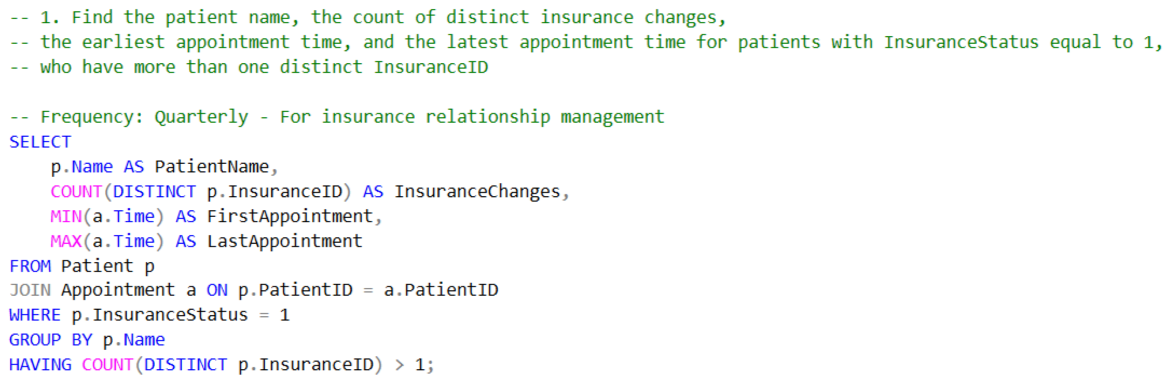


```sql
-- 9. Identify patients with more emergency appointments than the average.
-- Frequency: Useful for emergency care analytics.
SELECT PatientID, Name
FROM Patient
WHERE PatientID IN (
    SELECT PatientID
    FROM Appointment
    WHERE EmergencyStatus = 'High'
    GROUP BY PatientID
    HAVING COUNT(AppointmentID) > (
        SELECT AVG(emergency_appointments)
        FROM (
            SELECT PatientID, COUNT(AppointmentID) AS emergency_appointments
            FROM Appointment
            WHERE EmergencyStatus = 'High'
            GROUP BY PatientID
        ) AS temp
    )
);
```

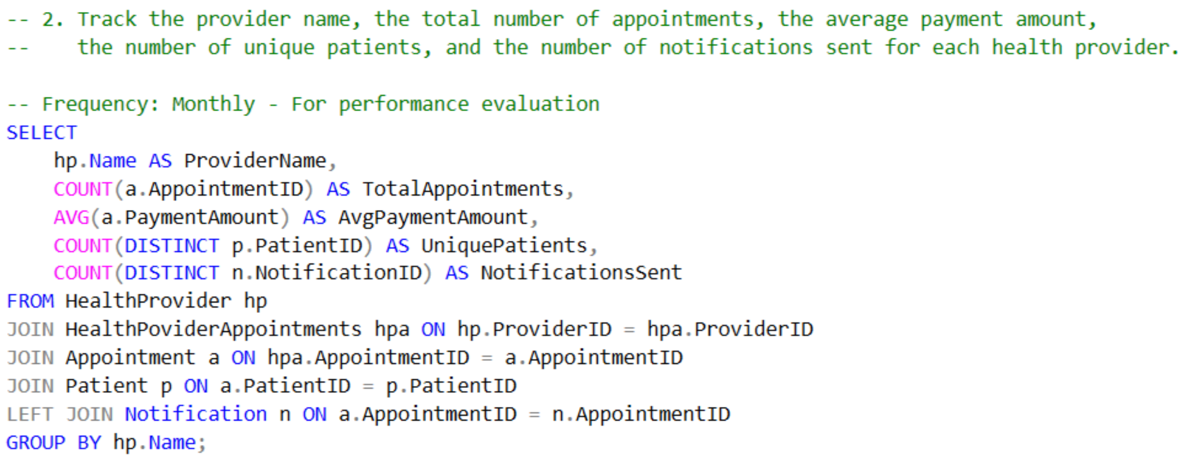


```sql
-- 10. Find patients with higher payment totals than average.
-- Frequency: Common in patient financial analysis.
SELECT PatientID, Name
FROM Patient
WHERE PatientID IN (
    SELECT PatientID
    FROM Appointment
    GROUP BY PatientID
    HAVING SUM(PaymentAmount) > (
        SELECT AVG(total_payment)
        FROM (
            SELECT PatientID, SUM(PaymentAmount) AS total_payment
            FROM Appointment
            GROUP BY PatientID
        ) AS temp
    )
);
```

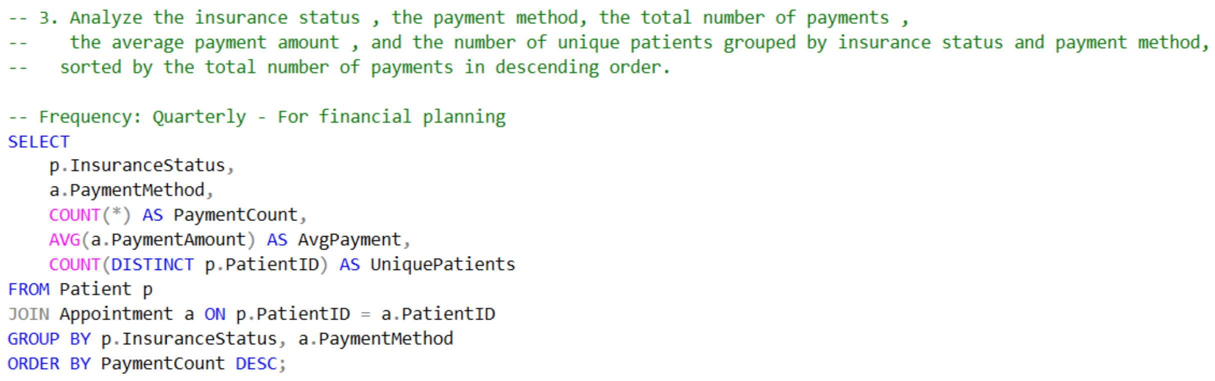
ADHAM SOBHY | 23-101003



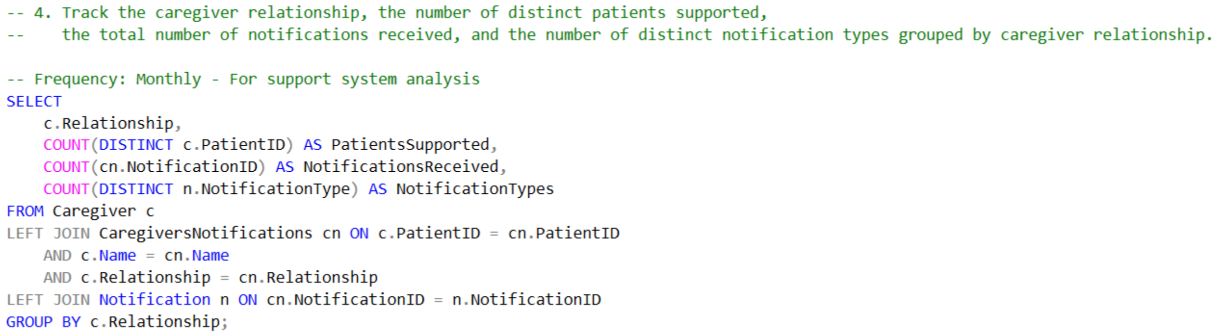
```sql
-- 1. Find the patient name, the count of distinct insurance changes,
-- the earliest appointment time, and the latest appointment time for patients with InsuranceStatus equal to 1,
-- who have more than one distinct InsuranceID

-- Frequency: Quarterly - For insurance relationship management
SELECT
    p.Name AS PatientName,
    COUNT(DISTINCT p.InsuranceID) AS InsuranceChanges,
    MIN(a.Time) AS FirstAppointment,
    MAX(a.Time) AS LastAppointment
FROM Patient p
JOIN Appointment a ON p.PatientID = a.PatientID
WHERE p.InsuranceStatus = 1
GROUP BY p.Name
HAVING COUNT(DISTINCT p.InsuranceID) > 1;
```



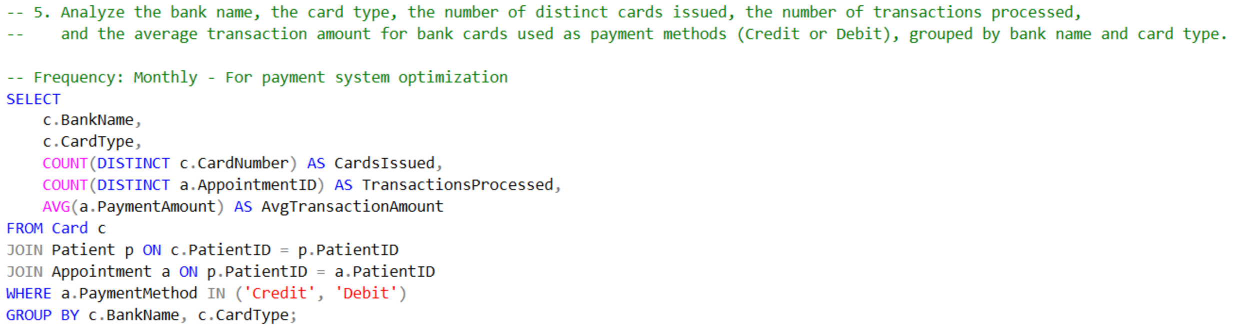
```sql
-- 2. Track the provider name, the total number of appointments, the average payment amount,
--    the number of unique patients, and the number of notifications sent for each health provider.

-- Frequency: Monthly - For performance evaluation
SELECT
    hp.Name AS ProviderName,
    COUNT(a.AppointmentID) AS TotalAppointments,
    AVG(a.PaymentAmount) AS AvgPaymentAmount,
    COUNT(DISTINCT p.PatientID) AS UniquePatients,
    COUNT(DISTINCT n.NotificationID) AS NotificationsSent
FROM HealthProvider hp
JOIN HealthPoviderAppointments hpa ON hp.ProviderID = hpa.ProviderID
JOIN Appointment a ON hpa.AppointmentID = a.AppointmentID
JOIN Patient p ON a.PatientID = p.PatientID
LEFT JOIN Notification n ON a.AppointmentID = n.AppointmentID
GROUP BY hp.Name;
```



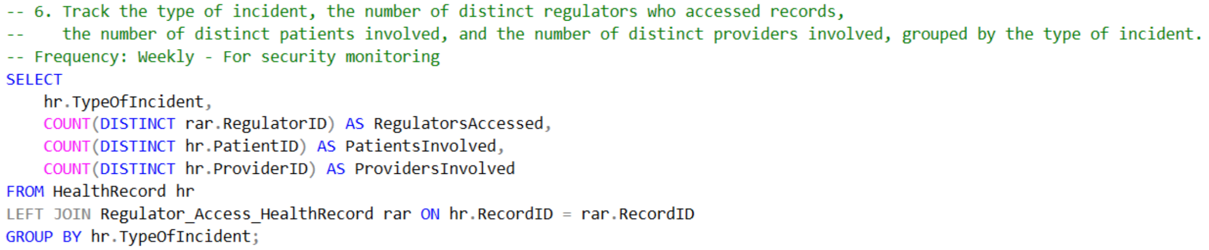
```sql
-- 3. Analyze the insurance status , the payment method, the total number of payments ,
--    the average payment amount , and the number of unique patients grouped by insurance status and payment method,
--   sorted by the total number of payments in descending order.

-- Frequency: Quarterly - For financial planning
SELECT
    p.InsuranceStatus,
    a.PaymentMethod,
    COUNT(*) AS PaymentCount,
    AVG(a.PaymentAmount) AS AvgPayment,
    COUNT(DISTINCT p.PatientID) AS UniquePatients
FROM Patient p
JOIN Appointment a ON p.PatientID = a.PatientID
GROUP BY p.InsuranceStatus, a.PaymentMethod
ORDER BY PaymentCount DESC;
```



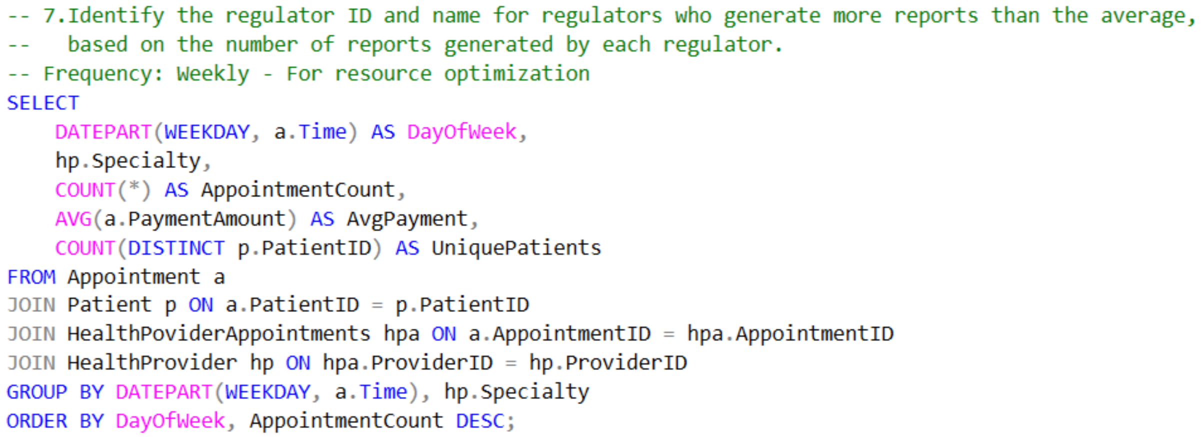
```
-- 4. Track the caregiver relationship, the number of distinct patients supported,
--    the total number of notifications received, and the number of distinct notification types grouped by caregiver relationship.

-- Frequency: Monthly - For support system analysis
SELECT
    c.Relationship,
    COUNT(DISTINCT c.PatientID) AS PatientsSupported,
    COUNT(cn.NotificationID) AS NotificationsReceived,
    COUNT(DISTINCT n.NotificationType) AS NotificationTypes
FROM Caregiver c
LEFT JOIN CaregiversNotifications cn ON c.PatientID = cn.PatientID
    AND c.Name = cn.Name
    AND c.Relationship = cn.Relationship
LEFT JOIN Notification n ON cn.NotificationID = n.NotificationID
GROUP BY c.Relationship;
```



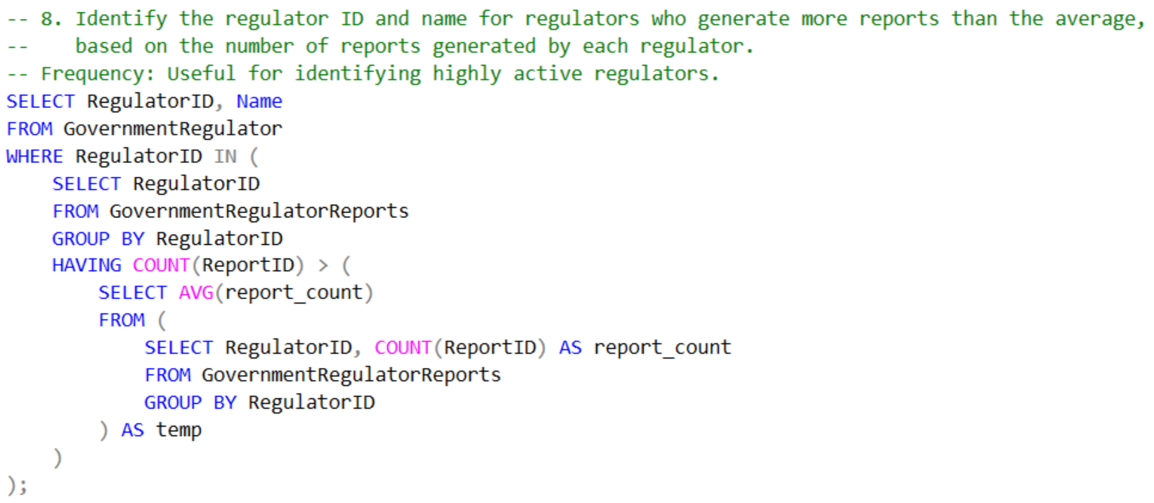
```
-- 5. Analyze the bank name, the card type, the number of distinct cards issued, the number of transactions processed,
--    and the average transaction amount for bank cards used as payment methods (Credit or Debit), grouped by bank name and card type.

-- Frequency: Monthly - For payment system optimization
SELECT
    c.BankName,
    c.CardType,
    COUNT(DISTINCT c.CardNumber) AS CardsIssued,
    COUNT(DISTINCT a.AppointmentID) AS TransactionsProcessed,
    AVG(a.PaymentAmount) AS AvgTransactionAmount
FROM Card c
JOIN Patient p ON c.PatientID = p.PatientID
JOIN Appointment a ON p.PatientID = a.PatientID
WHERE a.PaymentMethod IN ('Credit', 'Debit')
GROUP BY c.BankName, c.CardType;
```



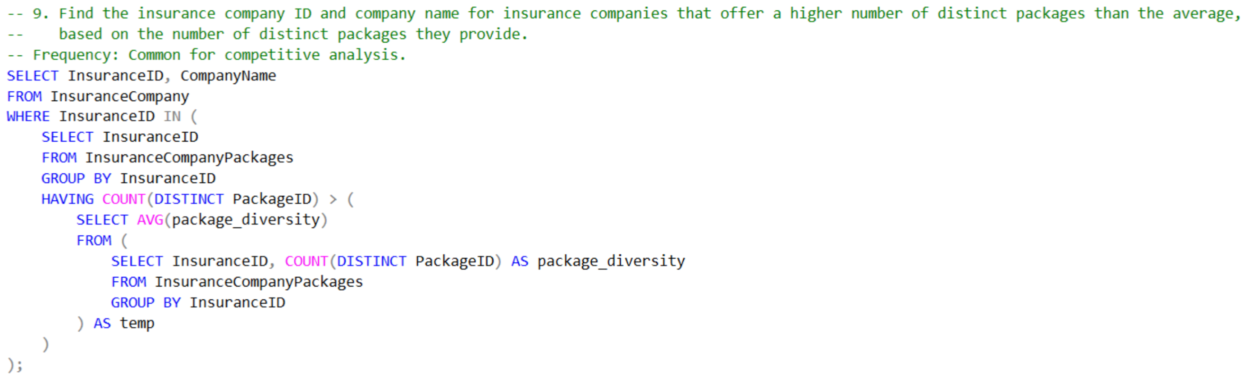
```
-- 6. Track the type of incident, the number of distinct regulators who accessed records,
--    the number of distinct patients involved, and the number of distinct providers involved, grouped by the type of incident.
-- Frequency: Weekly - For security monitoring
SELECT
    hr.TypeOfIncident,
    COUNT(DISTINCT rar.RegulatorID) AS RegulatorsAccessed,
    COUNT(DISTINCT hr.PatientID) AS PatientsInvolved,
    COUNT(DISTINCT hr.ProviderID) AS ProvidersInvolved
FROM HealthRecord hr
LEFT JOIN Regulator_Access_HealthRecord rar ON hr.RecordID = rar.RecordID
GROUP BY hr.TypeOfIncident;
```



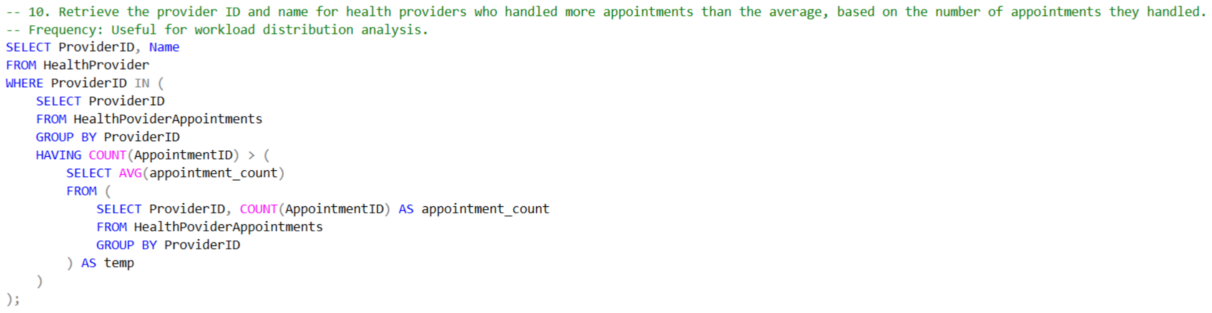
```
-- 7.Identify the regulator ID and name for regulators who generate more reports than the average,
--    based on the number of reports generated by each regulator.
-- Frequency: Weekly - For resource optimization
SELECT
    DATEPART(WEEKDAY, a.Time) AS DayOfWeek,
    hp.Specialty,
    COUNT(*) AS AppointmentCount,
    AVG(a.PaymentAmount) AS AvgPayment,
    COUNT(DISTINCT p.PatientID) AS UniquePatients
FROM Appointment a
JOIN Patient p ON a.PatientID = p.PatientID
JOIN HealthPoviderAppointments hpa ON a.AppointmentID = hpa.AppointmentID
JOIN HealthProvider hp ON hpa.ProviderID = hp.ProviderID
GROUP BY DATEPART(WEEKDAY, a.Time), hp.Specialty
ORDER BY DayOfWeek, AppointmentCount DESC;
```



```sql
-- 8. Identify the regulator ID and name for regulators who generate more reports than the average,
--    based on the number of reports generated by each regulator.
-- Frequency: Useful for identifying highly active regulators.
SELECT RegulatorID, Name
FROM GovernmentRegulator
WHERE RegulatorID IN (
    SELECT RegulatorID
    FROM GovernmentRegulatorReports
    GROUP BY RegulatorID
    HAVING COUNT(ReportID) > (
        SELECT AVG(report_count)
        FROM (
            SELECT RegulatorID, COUNT(ReportID) AS report_count
            FROM GovernmentRegulatorReports
            GROUP BY RegulatorID
        ) AS temp
    )
);
```

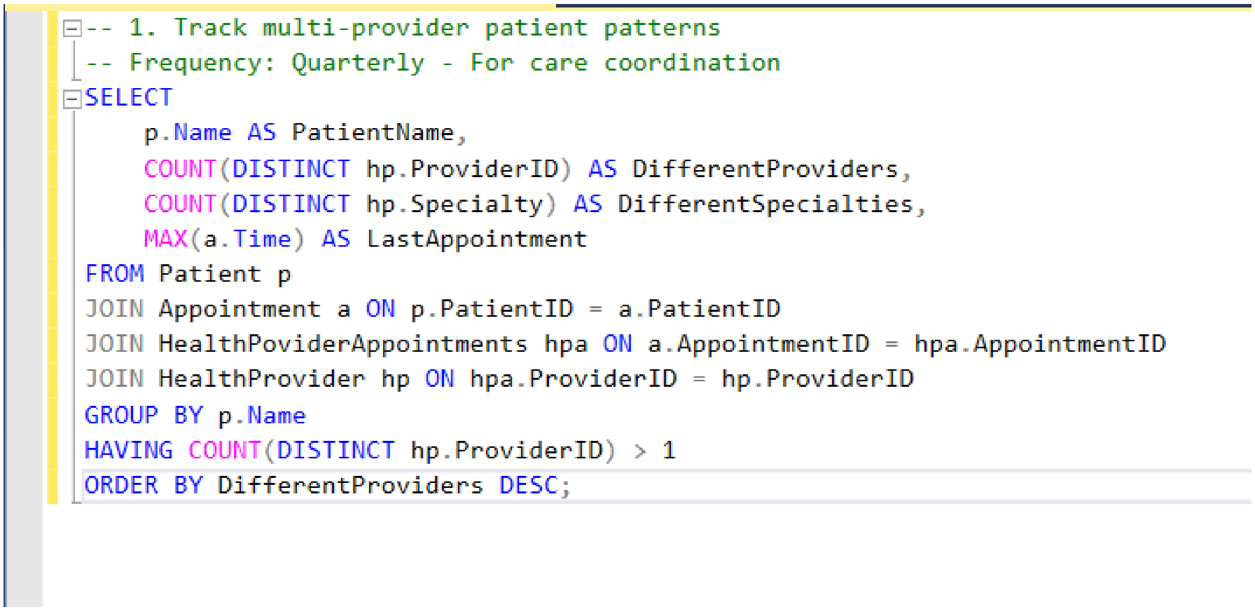


```sql
-- 9. Find the insurance company ID and company name for insurance companies that offer a higher number of distinct packages than the average,
--    based on the number of distinct packages they provide.
-- Frequency: Common for competitive analysis.
SELECT InsuranceID, CompanyName
FROM InsuranceCompany
WHERE InsuranceID IN (
    SELECT InsuranceID
    FROM InsuranceCompanyPackages
    GROUP BY InsuranceID
    HAVING COUNT(DISTINCT PackageID) > (
        SELECT AVG(package_diversity)
        FROM (
            SELECT InsuranceID, COUNT(DISTINCT PackageID) AS package_diversity
            FROM InsuranceCompanyPackages
            GROUP BY InsuranceID
        ) AS temp
    )
);
```

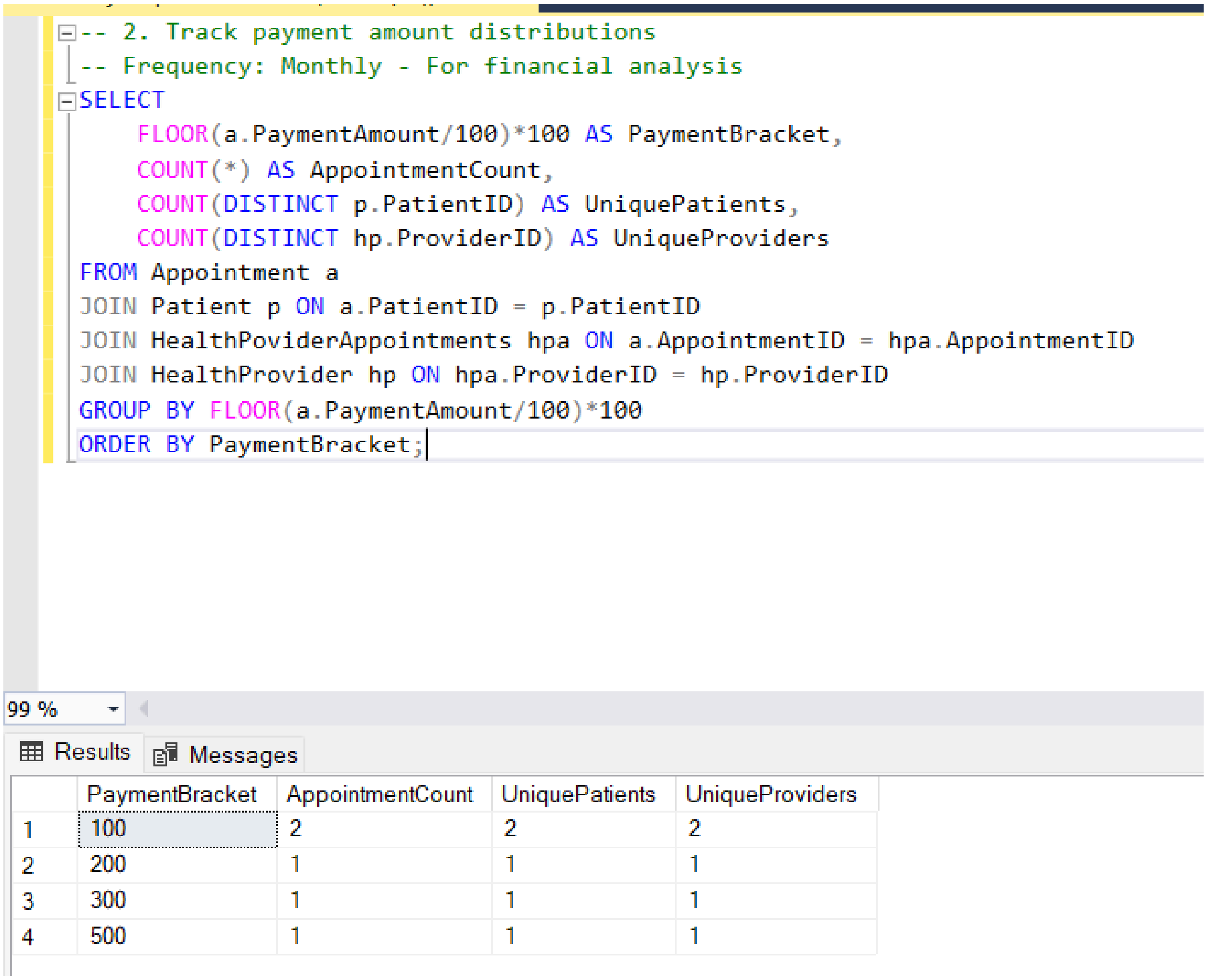


```sql
-- 10. Retrieve the provider ID and name for health providers who handled more appointments than the average, based on the number of appointments they handled.
-- Frequency: Useful for workload distribution analysis.
SELECT ProviderID, Name
FROM HealthProvider
WHERE ProviderID IN (
    SELECT ProviderID
    FROM HealthPoviderAppointments
    GROUP BY ProviderID
    HAVING COUNT(AppointmentID) > (
        SELECT AVG(appointment_count)
        FROM (
            SELECT ProviderID, COUNT(AppointmentID) AS appointment_count
            FROM HealthPoviderAppointments
            GROUP BY ProviderID
        ) AS temp
    )
);
```

MOHAMED IBRAHIM | 22-101058

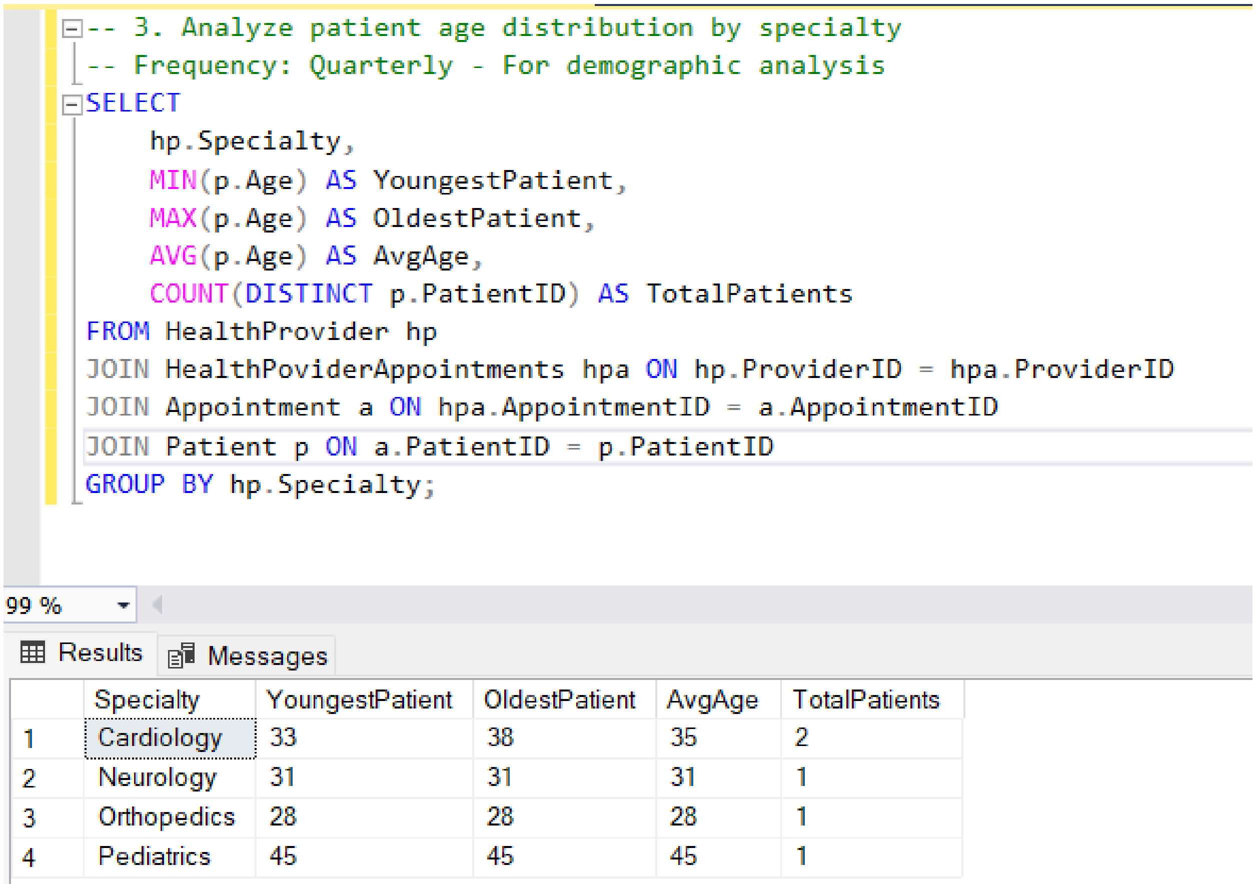


```
-- 1. Track multi-provider patient patterns
-- Frequency: Quarterly - For care coordination
SELECT
    p.Name AS PatientName,
    COUNT(DISTINCT hp.ProviderID) AS DifferentProviders,
    COUNT(DISTINCT hp.Specialty) AS DifferentSpecialties,
    MAX(a.Time) AS LastAppointment
FROM Patient p
JOIN Appointment a ON p.PatientID = a.PatientID
JOIN HealthPoviderAppointments hpa ON a.AppointmentID = hpa.AppointmentID
JOIN HealthProvider hp ON hpa.ProviderID = hp.ProviderID
GROUP BY p.Name
HAVING COUNT(DISTINCT hp.ProviderID) > 1
ORDER BY DifferentProviders DESC;
```



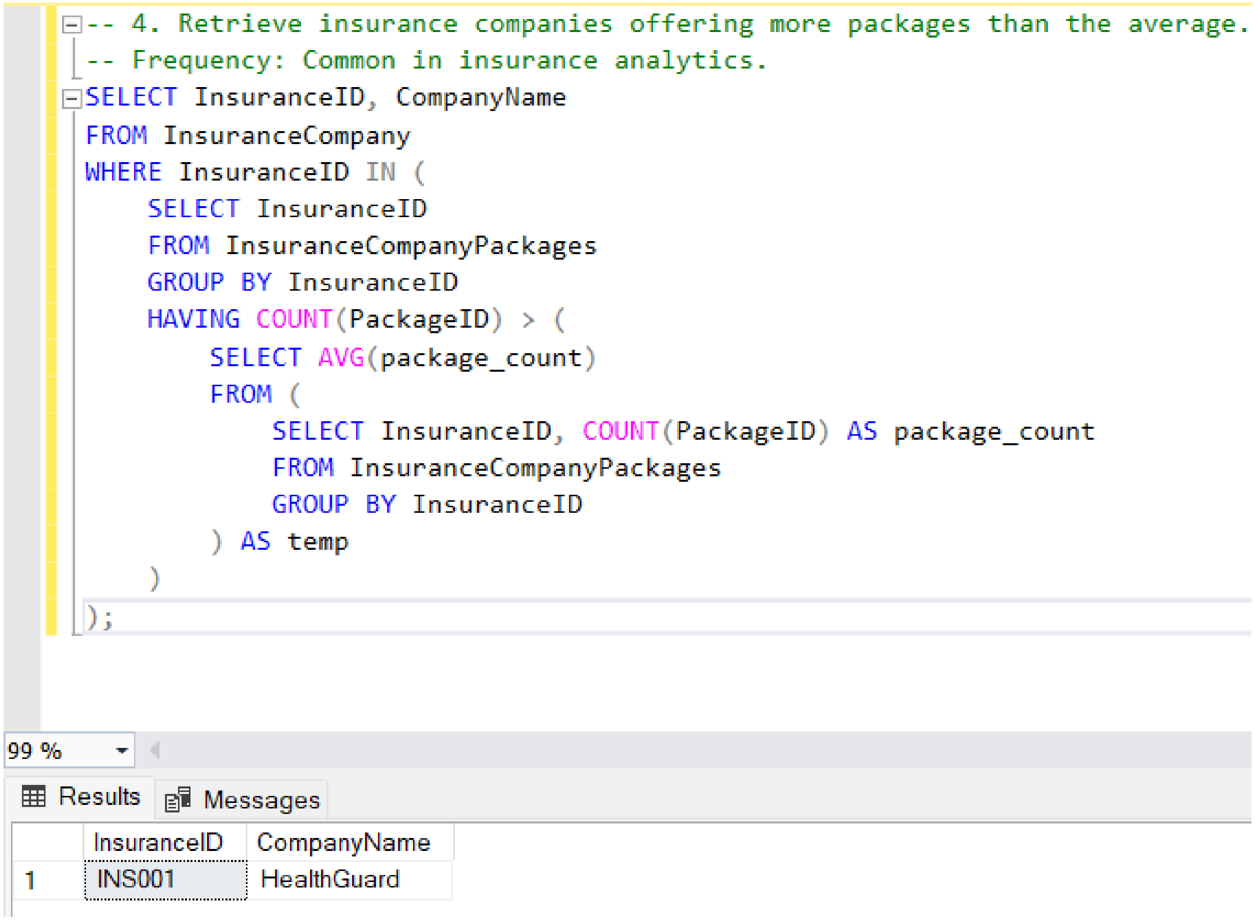
```
-- 2. Track payment amount distributions
-- Frequency: Monthly - For financial analysis
SELECT
    FLOOR(a.PaymentAmount/100)*100 AS PaymentBracket,
    COUNT(*) AS AppointmentCount,
    COUNT(DISTINCT p.PatientID) AS UniquePatients,
    COUNT(DISTINCT hp.ProviderID) AS UniqueProviders
FROM Appointment a
JOIN Patient p ON a.PatientID = p.PatientID
JOIN HealthPoviderAppointments hpa ON a.AppointmentID = hpa.AppointmentID
JOIN HealthProvider hp ON hpa.ProviderID = hp.ProviderID
GROUP BY FLOOR(a.PaymentAmount/100)*100
ORDER BY PaymentBracket;
```

	PaymentBracket	AppointmentCount	UniquePatients	UniqueProviders
1	100	2	2	2
2	200	1	1	1
3	300	1	1	1
4	500	1	1	1



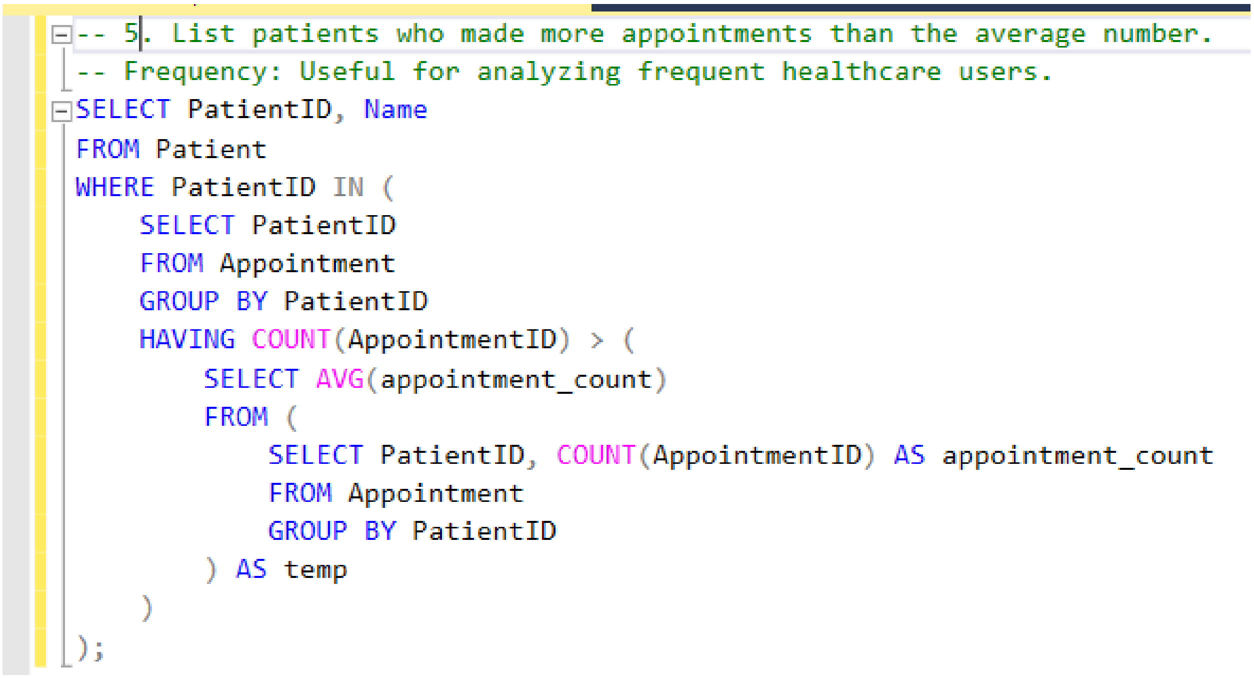
```sql
-- 3. Analyze patient age distribution by specialty
-- Frequency: Quarterly - For demographic analysis
SELECT
    hp.Specialty,
    MIN(p.Age) AS YoungestPatient,
    MAX(p.Age) AS OldestPatient,
    AVG(p.Age) AS AvgAge,
    COUNT(DISTINCT p.PatientID) AS TotalPatients
FROM HealthProvider hp
JOIN HealthPoviderAppointments hpa ON hp.ProviderID = hpa.ProviderID
JOIN Appointment a ON hpa.AppointmentID = a.AppointmentID
JOIN Patient p ON a.PatientID = p.PatientID
GROUP BY hp.Specialty;
```

	Specialty	YoungestPatient	OldestPatient	AvgAge	TotalPatients
1	Cardiology	33	38	35	2
2	Neurology	31	31	31	1
3	Orthopedics	28	28	28	1
4	Pediatrics	45	45	45	1

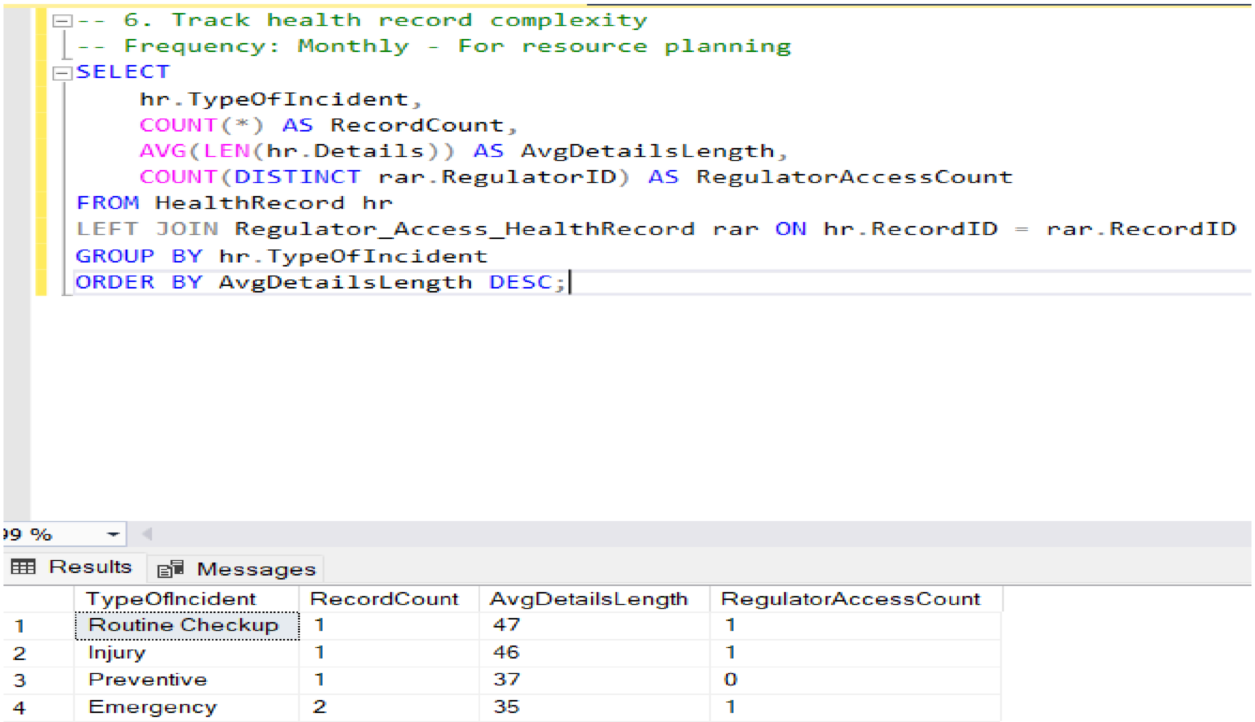


```sql
-- 4. Retrieve insurance companies offering more packages than the average.
-- Frequency: Common in insurance analytics.
SELECT InsuranceID, CompanyName
FROM InsuranceCompany
WHERE InsuranceID IN (
    SELECT InsuranceID
    FROM InsuranceCompanyPackages
    GROUP BY InsuranceID
    HAVING COUNT(PackageID) > (
        SELECT AVG(package_count)
        FROM (
            SELECT InsuranceID, COUNT(PackageID) AS package_count
            FROM InsuranceCompanyPackages
            GROUP BY InsuranceID
        ) AS temp
    )
);
```

	InsuranceID	CompanyName
1	INS001	HealthGuard

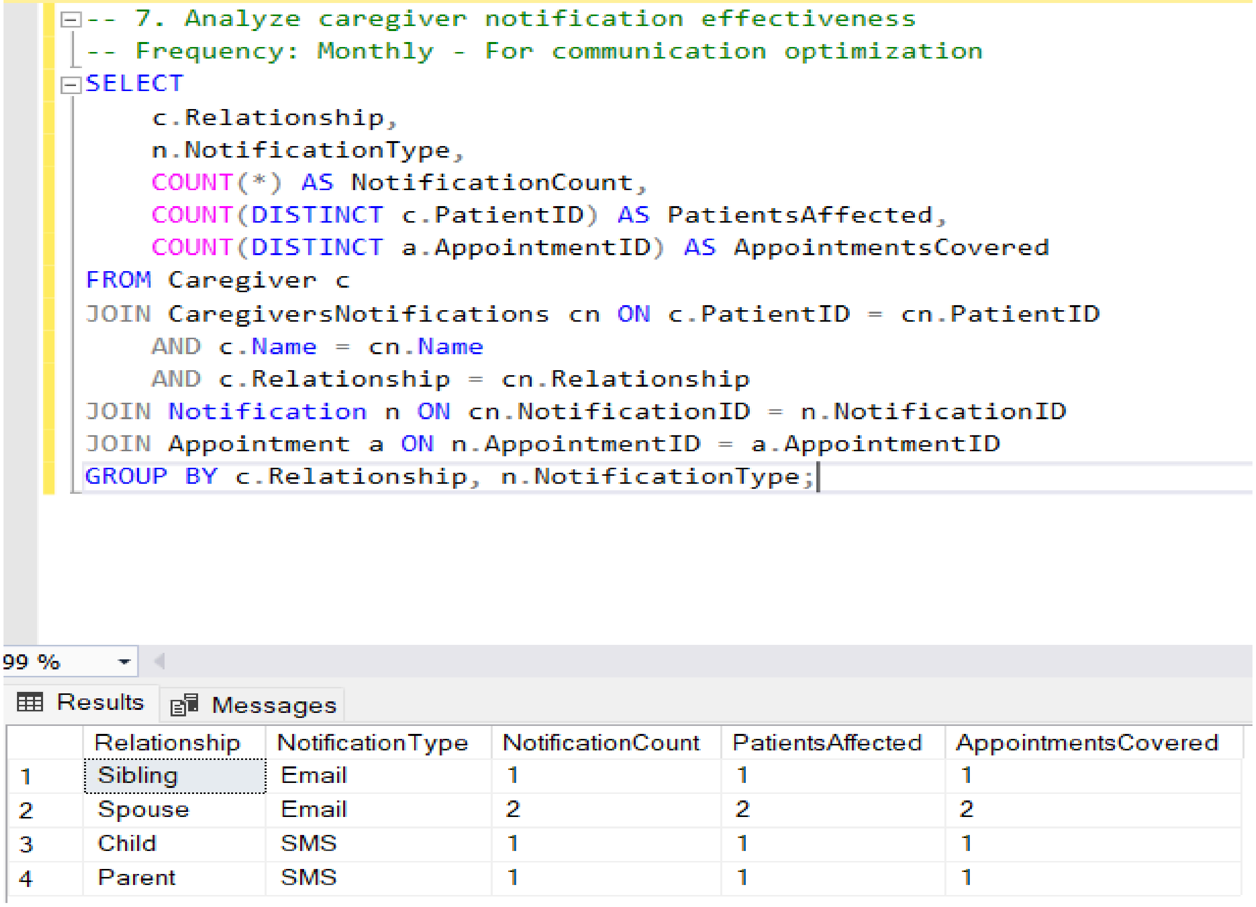


```sql
-- 5. List patients who made more appointments than the average number.
-- Frequency: Useful for analyzing frequent healthcare users.
SELECT PatientID, Name
FROM Patient
WHERE PatientID IN (
    SELECT PatientID
    FROM Appointment
    GROUP BY PatientID
    HAVING COUNT(AppointmentID) > (
        SELECT AVG(appointment_count)
        FROM (
            SELECT PatientID, COUNT(AppointmentID) AS appointment_count
            FROM Appointment
            GROUP BY PatientID
        ) AS temp
    )
);
```



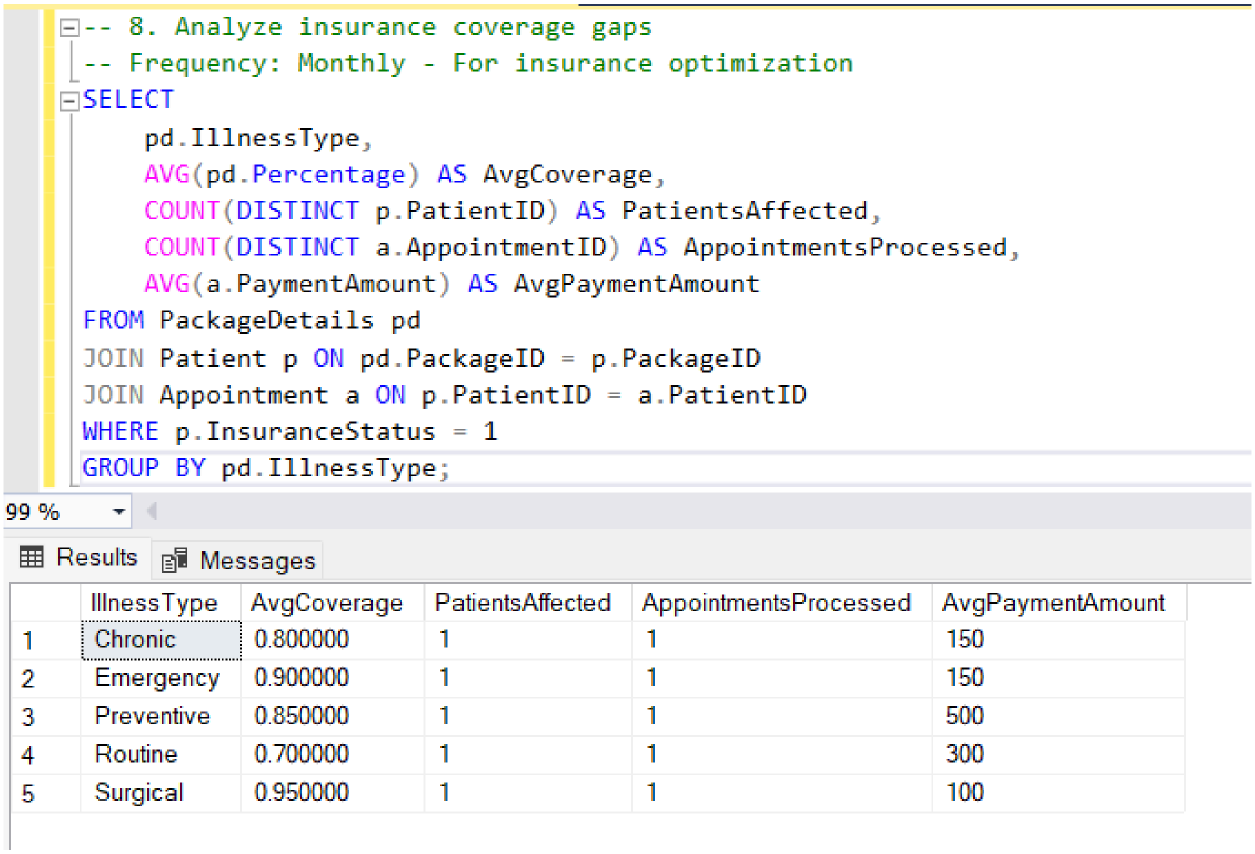
```sql
-- 6. Track health record complexity
-- Frequency: Monthly - For resource planning
SELECT
    hr.TypeOfIncident,
    COUNT(*) AS RecordCount,
    AVG(LEN(hr.Details)) AS AvgDetailsLength,
    COUNT(DISTINCT rar.RegulatorID) AS RegulatorAccessCount
FROM HealthRecord hr
LEFT JOIN Regulator_Access_HealthRecord rar ON hr.RecordID = rar.RecordID
GROUP BY hr.TypeOfIncident
ORDER BY AvgDetailsLength DESC;
```

	TypeOfIncident	RecordCount	AvgDetailsLength	RegulatorAccessCount
1	Routine Checkup	1	47	1
2	Injury	1	46	1
3	Preventive	1	37	0
4	Emergency	2	35	1



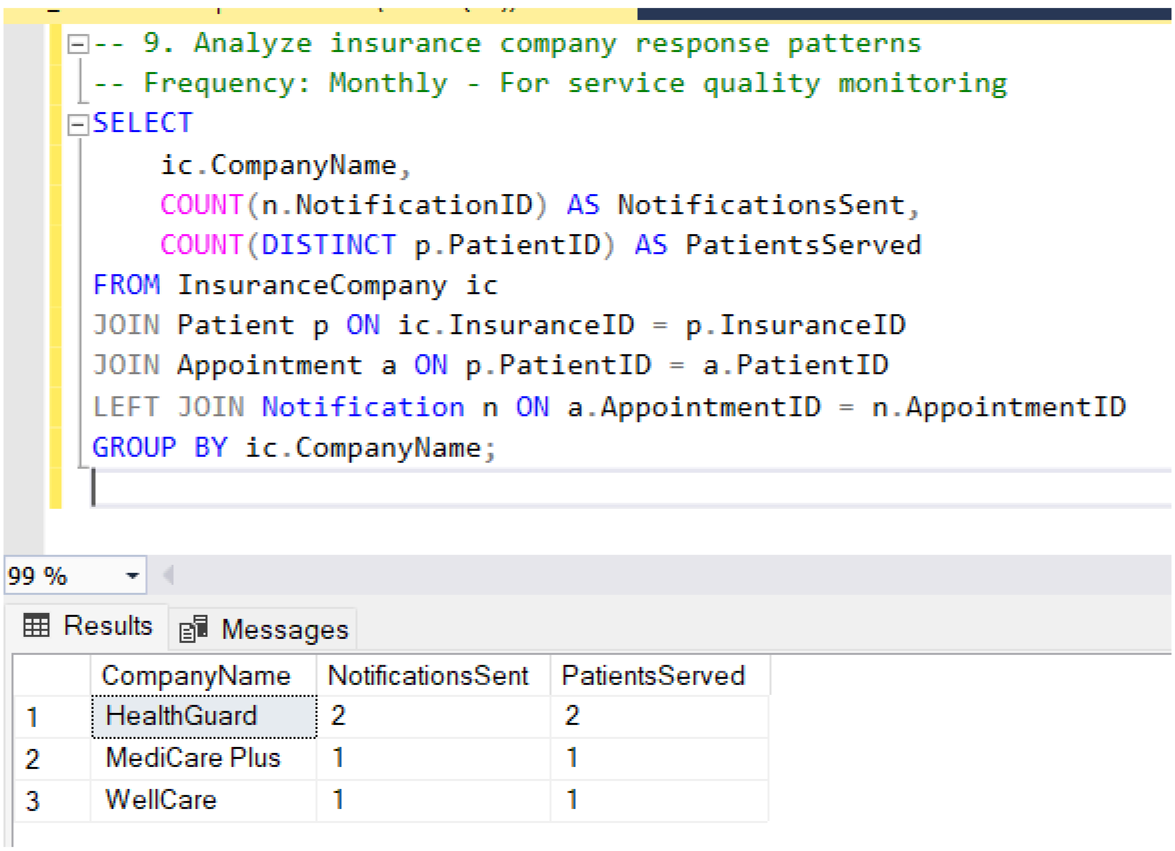
```sql
-- 7. Analyze caregiver notification effectiveness
-- Frequency: Monthly - For communication optimization
SELECT
    c.Relationship,
    n.NotificationType,
    COUNT(*) AS NotificationCount,
    COUNT(DISTINCT c.PatientID) AS PatientsAffected,
    COUNT(DISTINCT a.AppointmentID) AS AppointmentsCovered
FROM Caregiver c
JOIN CaregiversNotifications cn ON c.PatientID = cn.PatientID
    AND c.Name = cn.Name
    AND c.Relationship = cn.Relationship
JOIN Notification n ON cn.NotificationID = n.NotificationID
JOIN Appointment a ON n.AppointmentID = a.AppointmentID
GROUP BY c.Relationship, n.NotificationType;
```

	Relationship	NotificationType	NotificationCount	PatientsAffected	AppointmentsCovered
1	Sibling	Email	1	1	1
2	Spouse	Email	2	2	2
3	Child	SMS	1	1	1
4	Parent	SMS	1	1	1



```
-- 8. Analyze insurance coverage gaps
-- Frequency: Monthly - For insurance optimization
SELECT
    pd.IllnessType,
    AVG(pd.Percentage) AS AvgCoverage,
    COUNT(DISTINCT p.PatientID) AS PatientsAffected,
    COUNT(DISTINCT a.AppointmentID) AS AppointmentsProcessed,
    AVG(a.PaymentAmount) AS AvgPaymentAmount
FROM PackageDetails pd
JOIN Patient p ON pd.PackageID = p.PackageID
JOIN Appointment a ON p.PatientID = a.PatientID
WHERE p.InsuranceStatus = 1
GROUP BY pd.IllnessType;
```

	IllnessType	AvgCoverage	PatientsAffected	AppointmentsProcessed	AvgPaymentAmount
1	Chronic	0.800000	1	1	150
2	Emergency	0.900000	1	1	150
3	Preventive	0.850000	1	1	500
4	Routine	0.700000	1	1	300
5	Surgical	0.950000	1	1	100



```
-- 9. Analyze insurance company response patterns
-- Frequency: Monthly - For service quality monitoring
SELECT
    ic.CompanyName,
    COUNT(n.NotificationID) AS NotificationsSent,
    COUNT(DISTINCT p.PatientID) AS PatientsServed
FROM InsuranceCompany ic
JOIN Patient p ON ic.InsuranceID = p.InsuranceID
JOIN Appointment a ON p.PatientID = a.PatientID
LEFT JOIN Notification n ON a.AppointmentID = n.AppointmentID
GROUP BY ic.CompanyName;
```

	CompanyName	NotificationsSent	PatientsServed
1	HealthGuard	2	2
2	MediCare Plus	1	1
3	WellCare	1	1