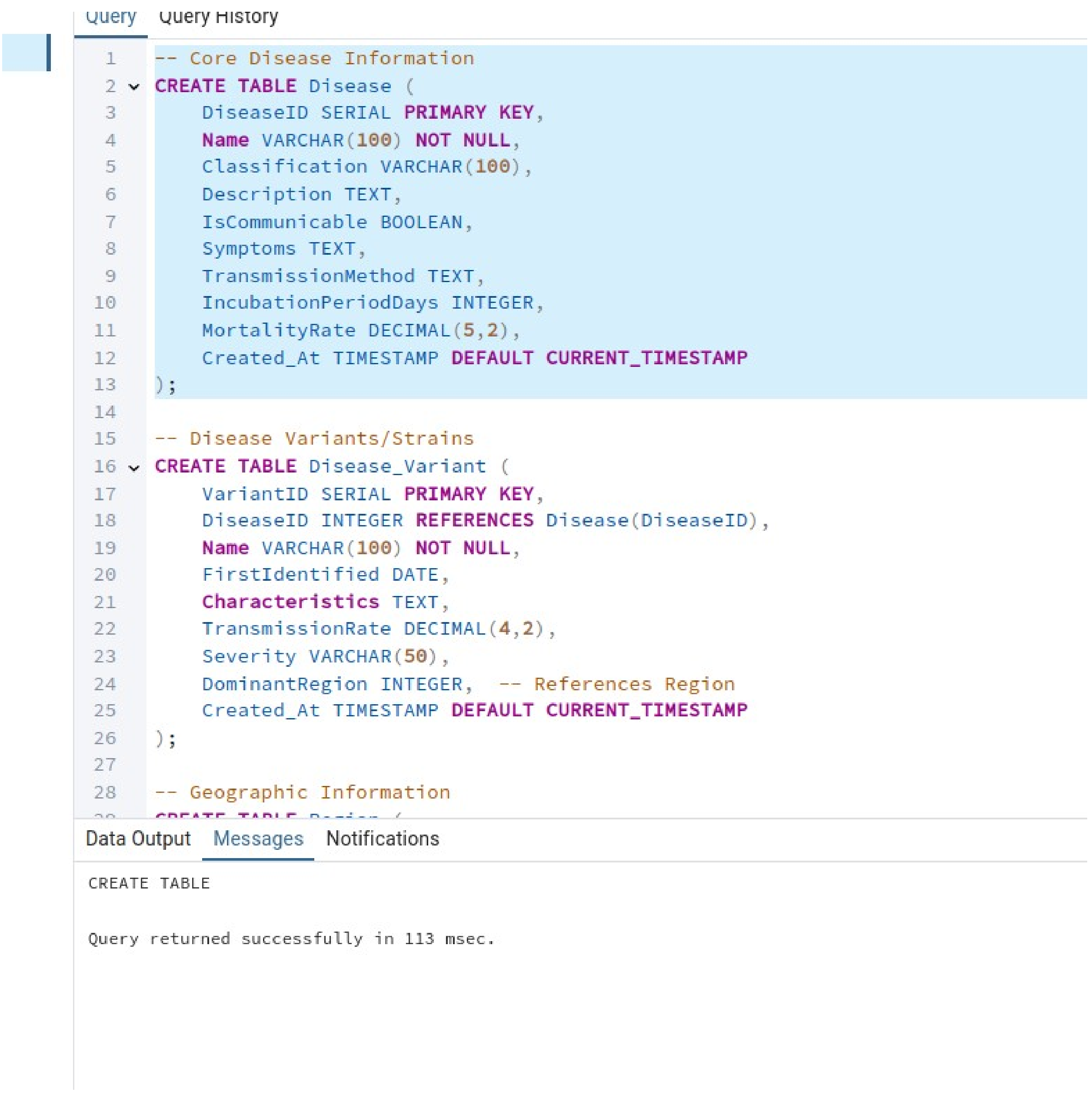
Creating tabels and indexes



Query Query History

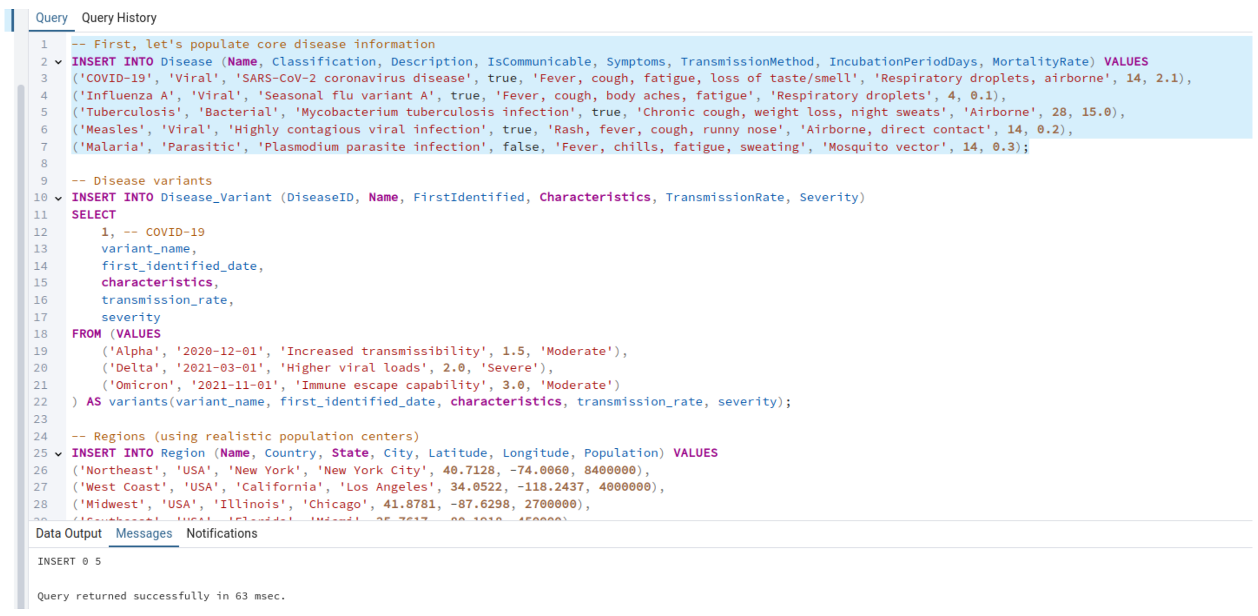
```
-- Core Disease Information
CREATE TABLE Disease (
    DiseaseID SERIAL PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Classification VARCHAR(100),
    Description TEXT,
    IsCommunicable BOOLEAN,
    Symptoms TEXT,
    TransmissionMethod TEXT,
    IncubationPeriodDays INTEGER,
    MortalityRate DECIMAL(5,2),
    Created_At TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Disease Variants/Strains
CREATE TABLE Disease_Variant (
    VariantID SERIAL PRIMARY KEY,
    DiseaseID INTEGER REFERENCES Disease(DiseaseID),
    Name VARCHAR(100) NOT NULL,
    FirstIdentified DATE,
    Characteristics TEXT,
    TransmissionRate DECIMAL(4,2),
    Severity VARCHAR(50),
    DominantRegion INTEGER,  -- References Region
    Created_At TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Geographic Information
```

Data Output Messages Notifications

```
CREATE TABLE

Query returned successfully in 113 msec.
```

....

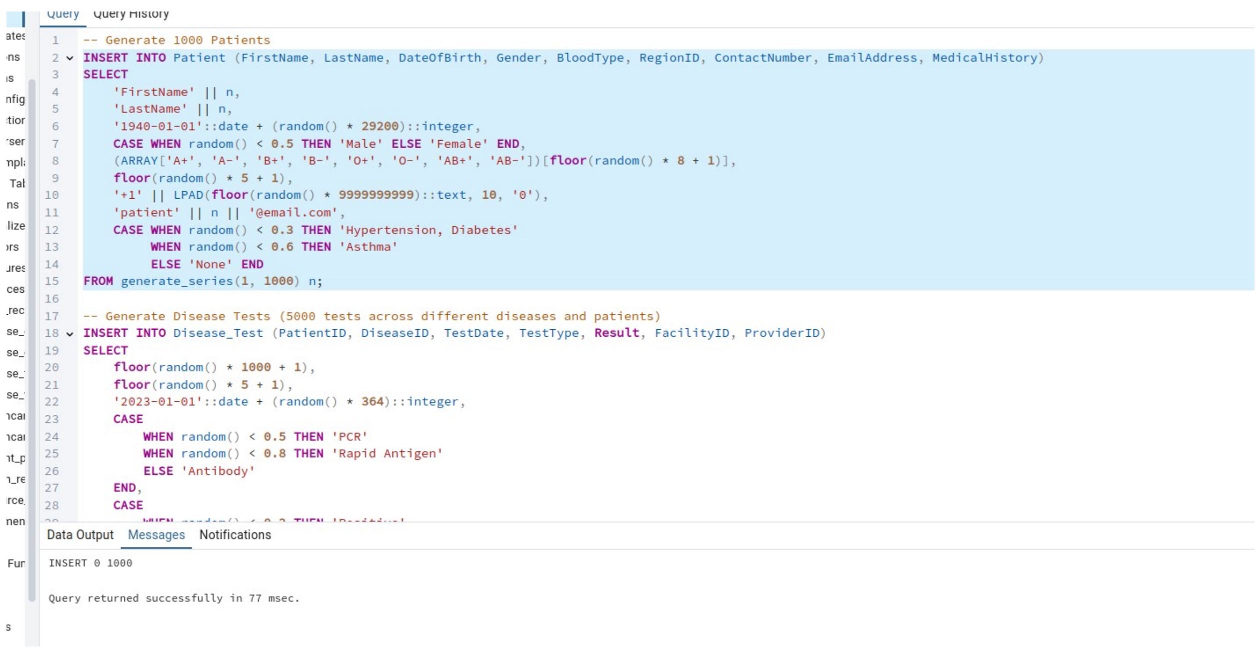
insyertong data



```sql
-- First, let's populate core disease information
INSERT INTO Disease (Name, Classification, Description, IsCommunicable, Symptoms, TransmissionMethod, IncubationPeriodDays, MortalityRate) VALUES
('COVID-19', 'Viral', 'SARS-CoV-2 coronavirus disease', true, 'Fever, cough, fatigue, loss of taste/smell', 'Respiratory droplets, airborne', 14, 2.1),
('Influenza A', 'Viral', 'Seasonal flu variant A', true, 'Fever, cough, body aches, fatigue', 'Respiratory droplets', 4, 0.1),
('Tuberculosis', 'Bacterial', 'Mycobacterium tuberculosis infection', true, 'Chronic cough, weight loss, night sweats', 'Airborne', 28, 15.0),
('Measles', 'Viral', 'Highly contagious viral infection', true, 'Rash, fever, cough, runny nose', 'Airborne, direct contact', 14, 0.2),
('Malaria', 'Parasitic', 'Plasmodium parasite infection', false, 'Fever, chills, fatigue, sweating', 'Mosquito vector', 14, 0.3);

-- Disease variants
INSERT INTO Disease_Variant (DiseaseID, Name, FirstIdentified, Characteristics, TransmissionRate, Severity)
SELECT
    1, -- COVID-19
    variant_name,
    first_identified_date,
    characteristics,
    transmission_rate,
    severity
FROM (VALUES
    ('Alpha', '2020-12-01', 'Increased transmissibility', 1.5, 'Moderate'),
    ('Delta', '2021-03-01', 'Higher viral loads', 2.0, 'Severe'),
    ('Omicron', '2021-11-01', 'Immune escape capability', 3.0, 'Moderate')
) AS variants(variant_name, first_identified_date, characteristics, transmission_rate, severity);

-- Regions (using realistic population centers)
INSERT INTO Region (Name, Country, State, City, Latitude, Longitude, Population) VALUES
('Northeast', 'USA', 'New York', 'New York City', 40.7128, -74.0060, 8400000),
('West Coast', 'USA', 'California', 'Los Angeles', 34.0522, -118.2437, 4000000),
('Midwest', 'USA', 'Illinois', 'Chicago', 41.8781, -87.6298, 2700000),
```

```
INSERT 0 5

Query returned successfully in 63 msec.
```

generating data

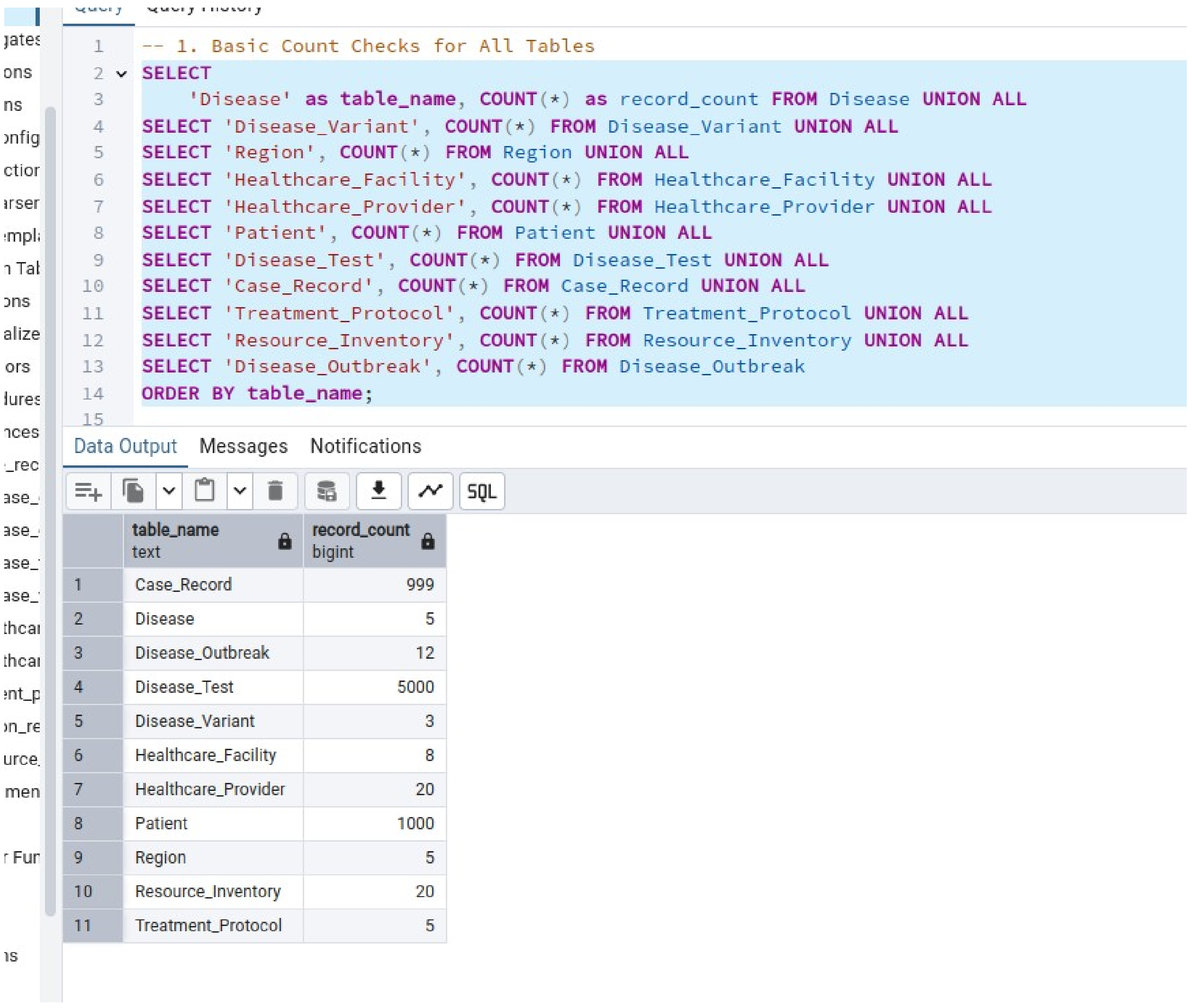


```sql
-- Generate 1000 Patients
INSERT INTO Patient (FirstName, LastName, DateOfBirth, Gender, BloodType, RegionID, ContactNumber, EmailAddress, MedicalHistory)
SELECT
    'FirstName' || n,
    'LastName' || n,
    '1940-01-01'::date + (random() * 29200)::integer,
    CASE WHEN random() < 0.5 THEN 'Male' ELSE 'Female' END,
    (ARRAY['A+', 'A-', 'B+', 'B-', 'O+', 'O-', 'AB+', 'AB-'])[floor(random() * 8 + 1)],
    floor(random() * 5 + 1),
    '+1' || LPAD(floor(random() * 9999999999)::text, 10, '0'),
    'patient' || n || '@email.com',
    CASE WHEN random() < 0.3 THEN 'Hypertension, Diabetes'
         WHEN random() < 0.6 THEN 'Asthma'
         ELSE 'None' END
FROM generate_series(1, 1000) n;

-- Generate Disease Tests (5000 tests across different diseases and patients)
INSERT INTO Disease_Test (PatientID, DiseaseID, TestDate, TestType, Result, FacilityID, ProviderID)
SELECT
    floor(random() * 1000 + 1),
    floor(random() * 5 + 1),
    '2023-01-01'::date + (random() * 364)::integer,
    CASE
        WHEN random() < 0.5 THEN 'PCR'
        WHEN random() < 0.8 THEN 'Rapid Antigen'
        ELSE 'Antibody'
    END,
    CASE
```

```
INSERT 0 1000

Query returned successfully in 77 msec.
```

.....

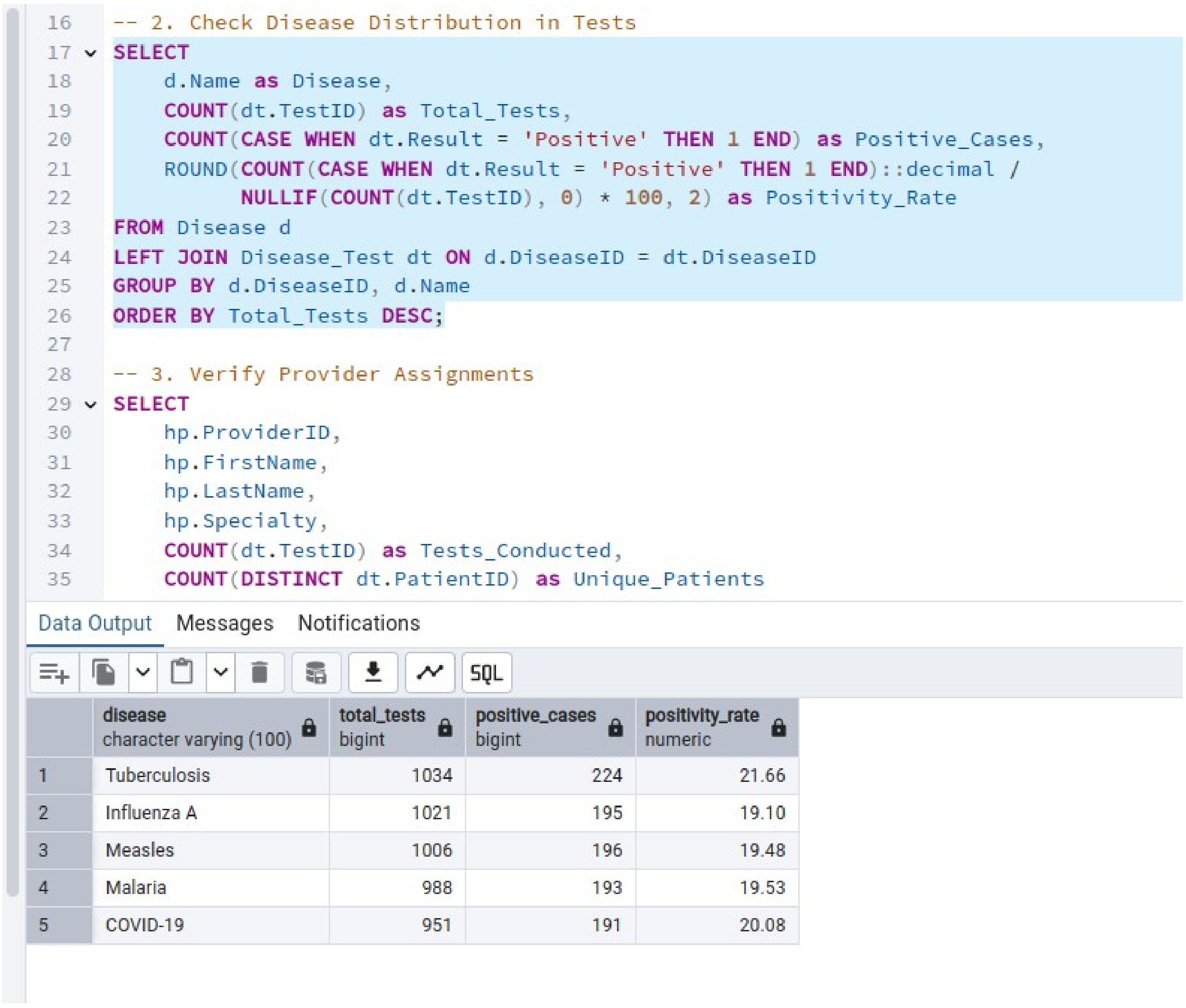
Verification



```sql
-- 1. Basic Count Checks for All Tables
SELECT
    'Disease' as table_name, COUNT(*) as record_count FROM Disease UNION ALL
SELECT 'Disease_Variant', COUNT(*) FROM Disease_Variant UNION ALL
SELECT 'Region', COUNT(*) FROM Region UNION ALL
SELECT 'Healthcare_Facility', COUNT(*) FROM Healthcare_Facility UNION ALL
SELECT 'Healthcare_Provider', COUNT(*) FROM Healthcare_Provider UNION ALL
SELECT 'Patient', COUNT(*) FROM Patient UNION ALL
SELECT 'Disease_Test', COUNT(*) FROM Disease_Test UNION ALL
SELECT 'Case_Record', COUNT(*) FROM Case_Record UNION ALL
SELECT 'Treatment_Protocol', COUNT(*) FROM Treatment_Protocol UNION ALL
SELECT 'Resource_Inventory', COUNT(*) FROM Resource_Inventory UNION ALL
SELECT 'Disease_Outbreak', COUNT(*) FROM Disease_Outbreak
ORDER BY table_name;
```

	table_name text	record_count bigint
1	Case_Record	999
2	Disease	5
3	Disease_Outbreak	12
4	Disease_Test	5000
5	Disease_Variant	3
6	Healthcare_Facility	8
7	Healthcare_Provider	20
8	Patient	1000
9	Region	5
10	Resource_Inventory	20
11	Treatment_Protocol	5

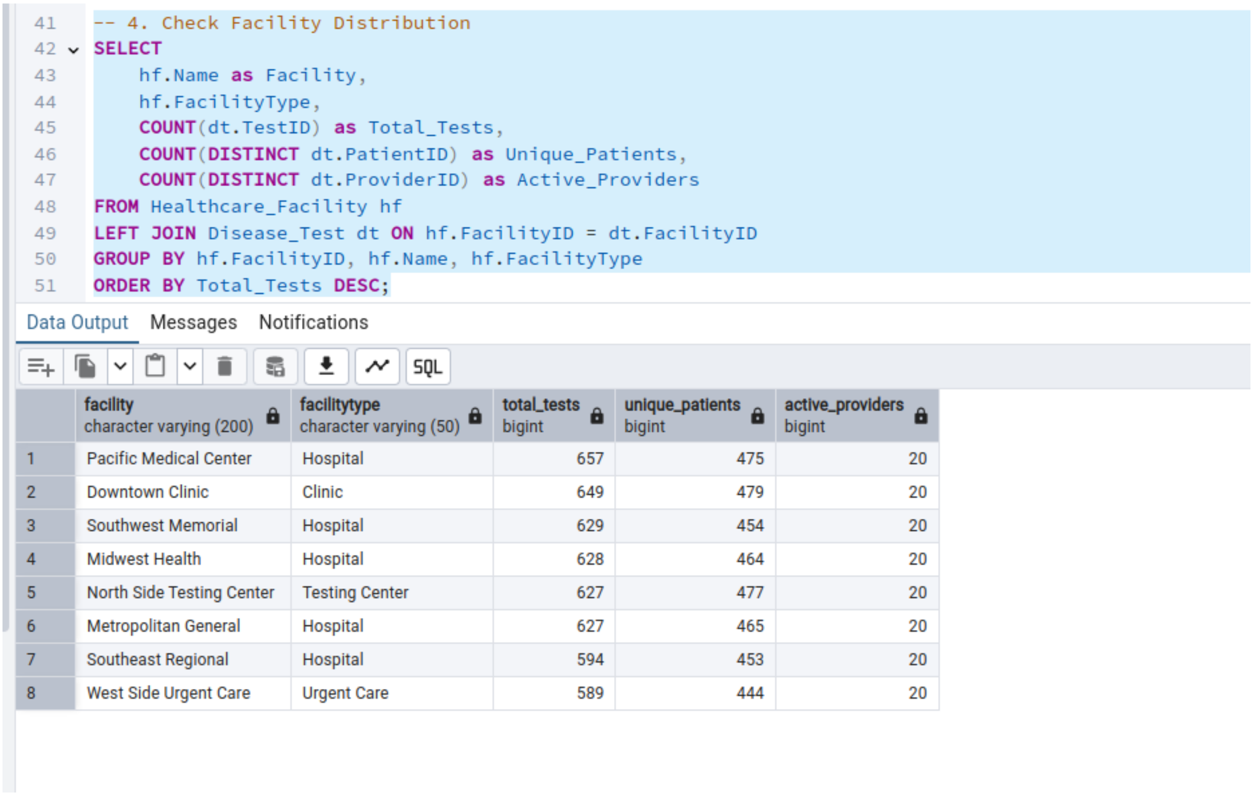
....



```sql
-- 2. Check Disease Distribution in Tests
SELECT
    d.Name as Disease,
    COUNT(dt.TestID) as Total_Tests,
    COUNT(CASE WHEN dt.Result = 'Positive' THEN 1 END) as Positive_Cases,
    ROUND(COUNT(CASE WHEN dt.Result = 'Positive' THEN 1 END)::decimal /
          NULLIF(COUNT(dt.TestID), 0) * 100, 2) as Positivity_Rate
FROM Disease d
LEFT JOIN Disease_Test dt ON d.DiseaseID = dt.DiseaseID
GROUP BY d.DiseaseID, d.Name
ORDER BY Total_Tests DESC;

-- 3. Verify Provider Assignments
SELECT
    hp.ProviderID,
    hp.FirstName,
    hp.LastName,
    hp.Specialty,
    COUNT(dt.TestID) as Tests_Conducted,
    COUNT(DISTINCT dt.PatientID) as Unique_Patients
```

Data Output Messages Notifications

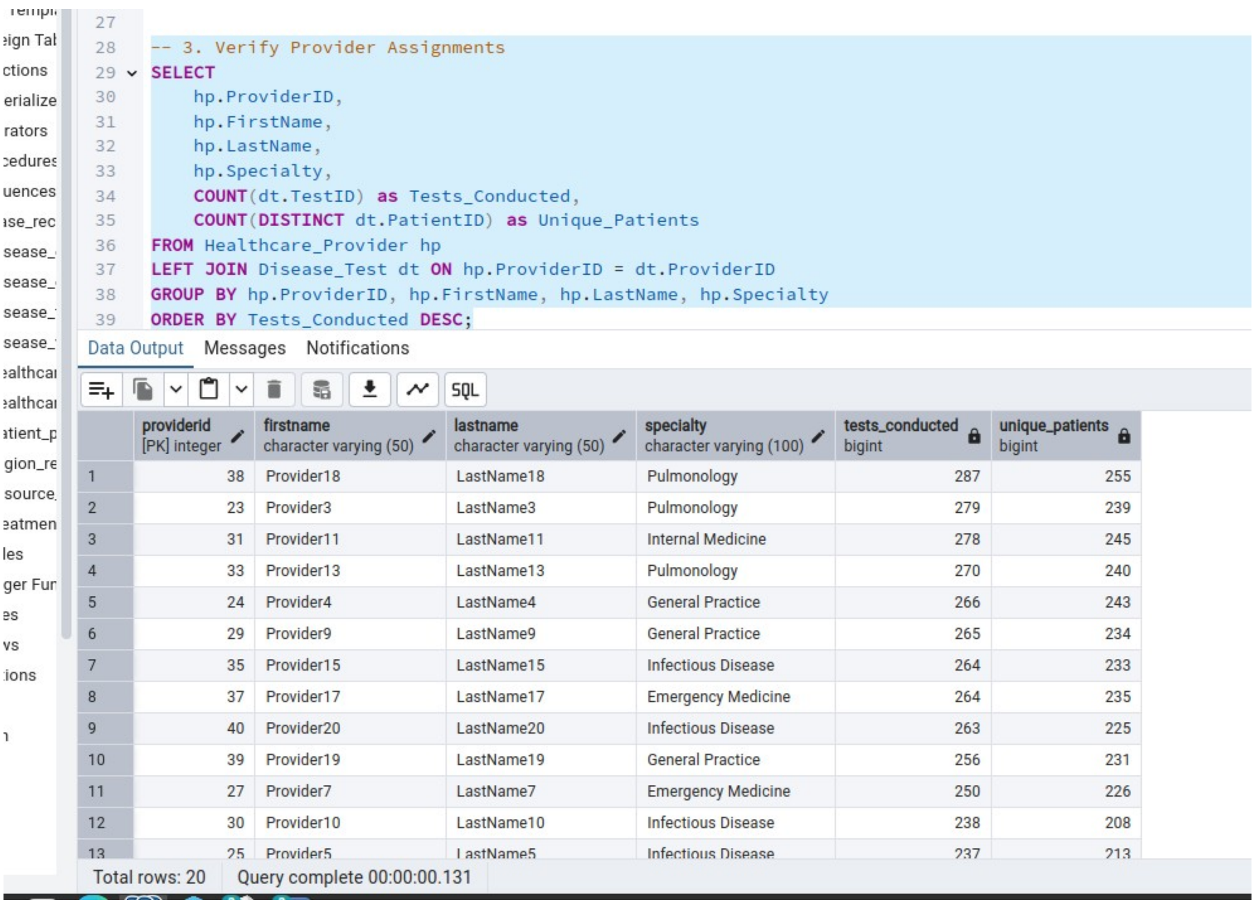
	disease character varying (100)	total_tests bigint	positive_cases bigint	positivity_rate numeric
1	Tuberculosis	1034	224	21.66
2	Influenza A	1021	195	19.10
3	Measles	1006	196	19.48
4	Malaria	988	193	19.53
5	COVID-19	951	191	20.08



```sql
-- 4. Check Facility Distribution
SELECT
    hf.Name as Facility,
    hf.FacilityType,
    COUNT(dt.TestID) as Total_Tests,
    COUNT(DISTINCT dt.PatientID) as Unique_Patients,
    COUNT(DISTINCT dt.ProviderID) as Active_Providers
FROM Healthcare_Facility hf
LEFT JOIN Disease_Test dt ON hf.FacilityID = dt.FacilityID
GROUP BY hf.FacilityID, hf.Name, hf.FacilityType
ORDER BY Total_Tests DESC;
```

Data Output Messages Notifications

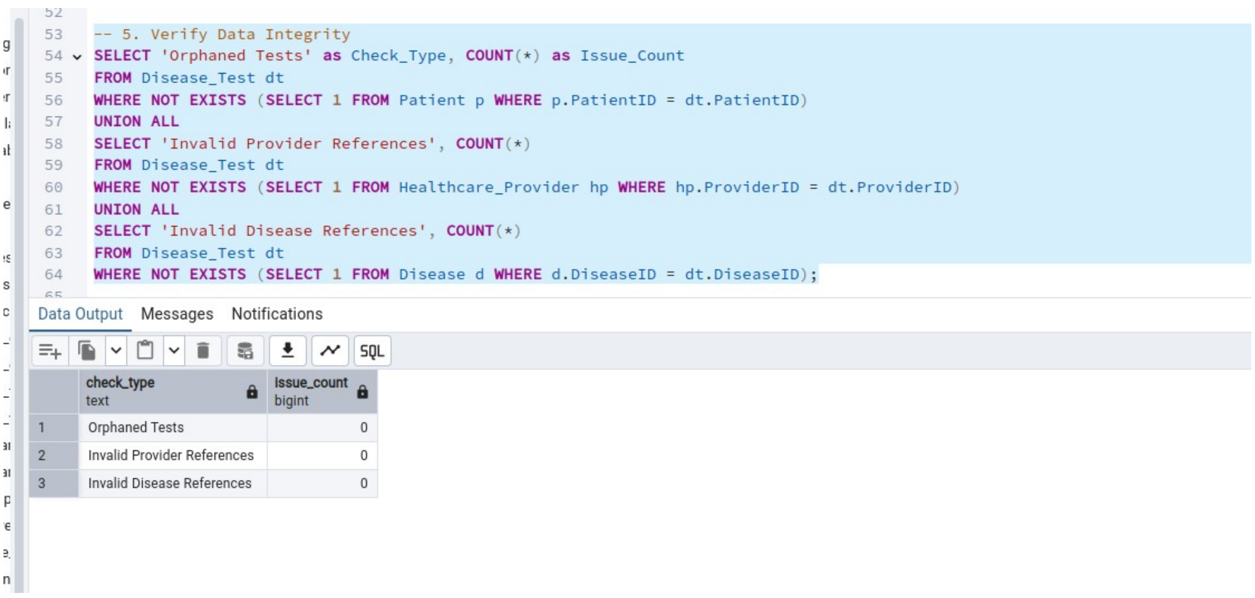
	facility character varying (200)	facilitytype character varying (50)	total_tests bigint	unique_patients bigint	active_providers bigint
1	Pacific Medical Center	Hospital	657	475	20
2	Downtown Clinic	Clinic	649	479	20
3	Southwest Memorial	Hospital	629	454	20
4	Midwest Health	Hospital	628	464	20
5	North Side Testing Center	Testing Center	627	477	20
6	Metropolitan General	Hospital	627	465	20
7	Southeast Regional	Hospital	594	453	20
8	West Side Urgent Care	Urgent Care	589	444	20



```sql
-- 3. Verify Provider Assignments
SELECT
    hp.ProviderID,
    hp.FirstName,
    hp.LastName,
    hp.Specialty,
    COUNT(dt.TestID) as Tests_Conducted,
    COUNT(DISTINCT dt.PatientID) as Unique_Patients
FROM Healthcare_Provider hp
LEFT JOIN Disease_Test dt ON hp.ProviderID = dt.ProviderID
GROUP BY hp.ProviderID, hp.FirstName, hp.LastName, hp.Specialty
ORDER BY Tests_Conducted DESC;
```

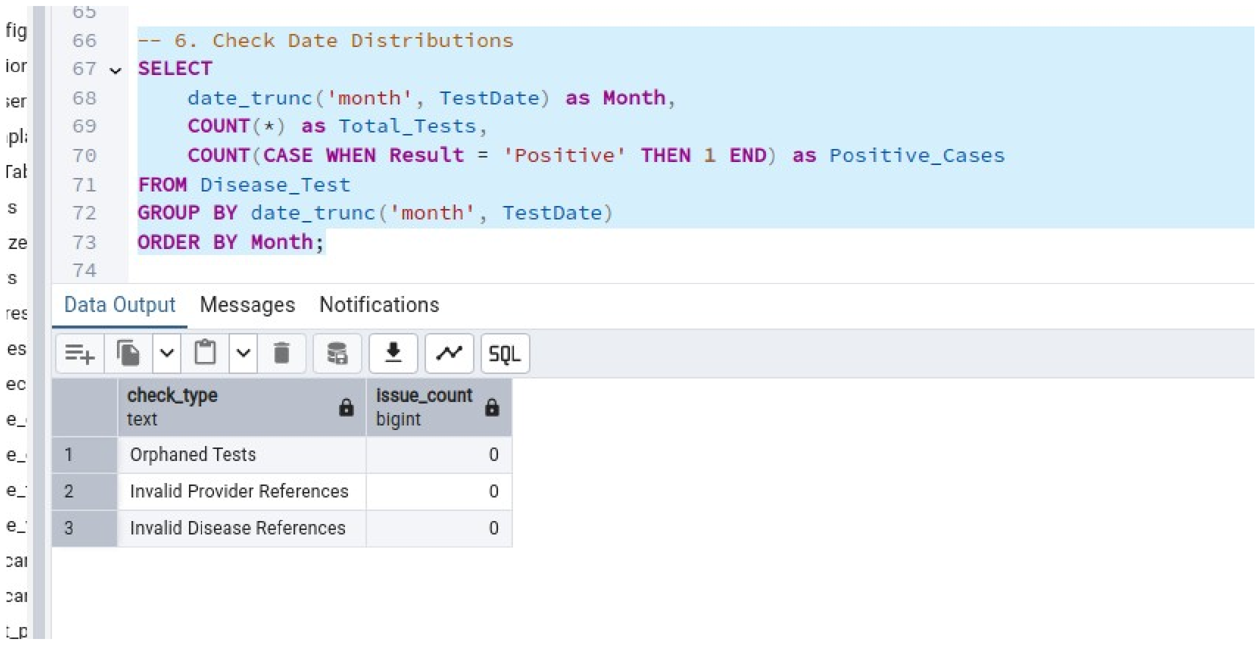
	providerid [PK] integer	firstname character varying (50)	lastname character varying (50)	specialty character varying (100)	tests_conducted bigint	unique_patients bigint
1	38	Provider18	LastName18	Pulmonology	287	255
2	23	Provider3	LastName3	Pulmonology	279	239
3	31	Provider11	LastName11	Internal Medicine	278	245
4	33	Provider13	LastName13	Pulmonology	270	240
5	24	Provider4	LastName4	General Practice	266	243
6	29	Provider9	LastName9	General Practice	265	234
7	35	Provider15	LastName15	Infectious Disease	264	233
8	37	Provider17	LastName17	Emergency Medicine	264	235
9	40	Provider20	LastName20	Infectious Disease	263	225
10	39	Provider19	LastName19	General Practice	256	231
11	27	Provider7	LastName7	Emergency Medicine	250	226
12	30	Provider10	LastName10	Infectious Disease	238	208
13	25	Provider5	LastName5	Infectious Disease	237	213

Total rows: 20   Query complete 00:00:00.131



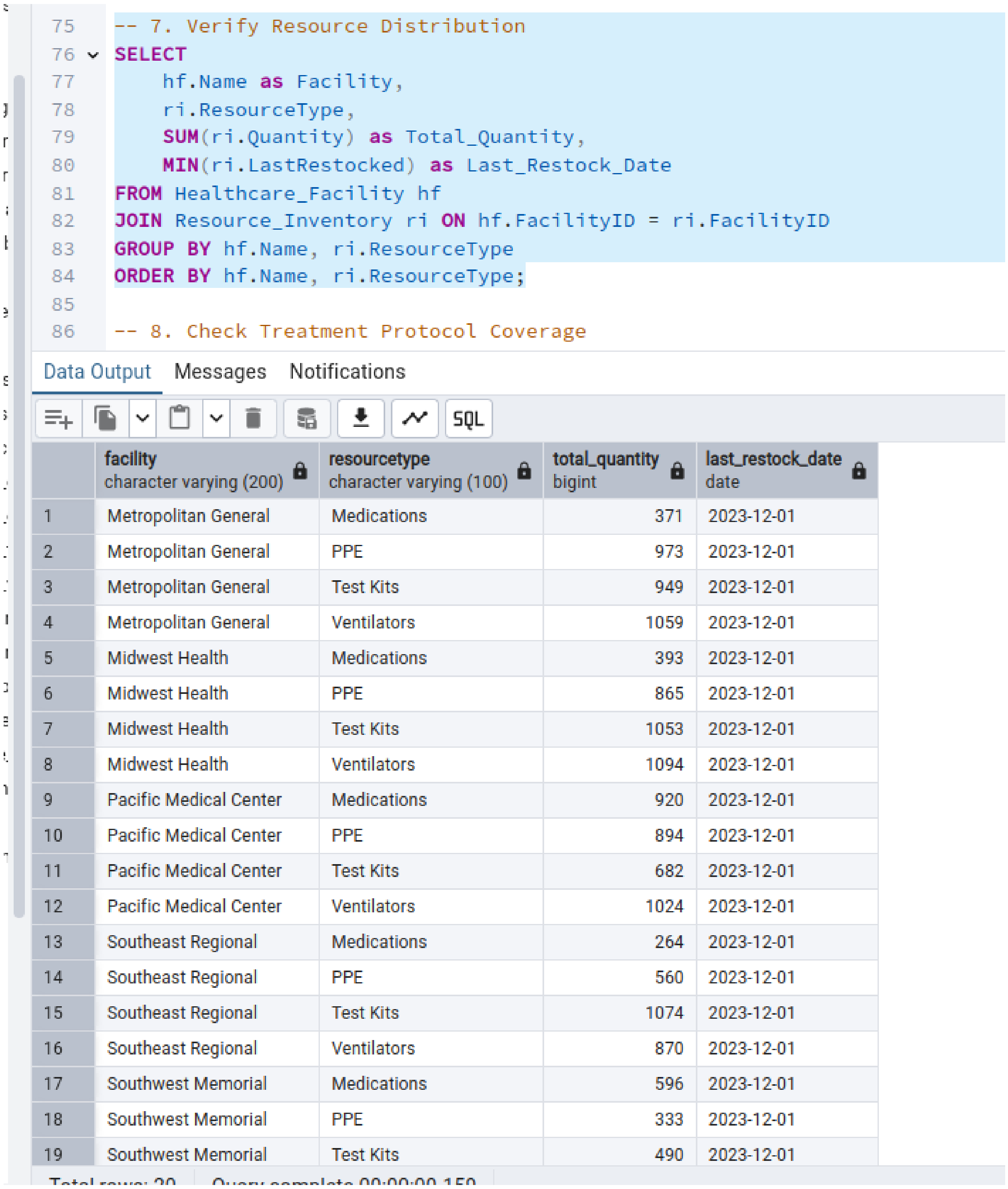
```sql
-- 5. Verify Data Integrity
SELECT 'Orphaned Tests' as Check_Type, COUNT(*) as Issue_Count
FROM Disease_Test dt
WHERE NOT EXISTS (SELECT 1 FROM Patient p WHERE p.PatientID = dt.PatientID)
UNION ALL
SELECT 'Invalid Provider References', COUNT(*)
FROM Disease_Test dt
WHERE NOT EXISTS (SELECT 1 FROM Healthcare_Provider hp WHERE hp.ProviderID = dt.ProviderID)
UNION ALL
SELECT 'Invalid Disease References', COUNT(*)
FROM Disease_Test dt
WHERE NOT EXISTS (SELECT 1 FROM Disease d WHERE d.DiseaseID = dt.DiseaseID);
```

	check_type text	issue_count bigint
1	Orphaned Tests	0
2	Invalid Provider References	0
3	Invalid Disease References	0



```
-- 6. Check Date Distributions
SELECT
    date_trunc('month', TestDate) as Month,
    COUNT(*) as Total_Tests,
    COUNT(CASE WHEN Result = 'Positive' THEN 1 END) as Positive_Cases
FROM Disease_Test
GROUP BY date_trunc('month', TestDate)
ORDER BY Month;
```

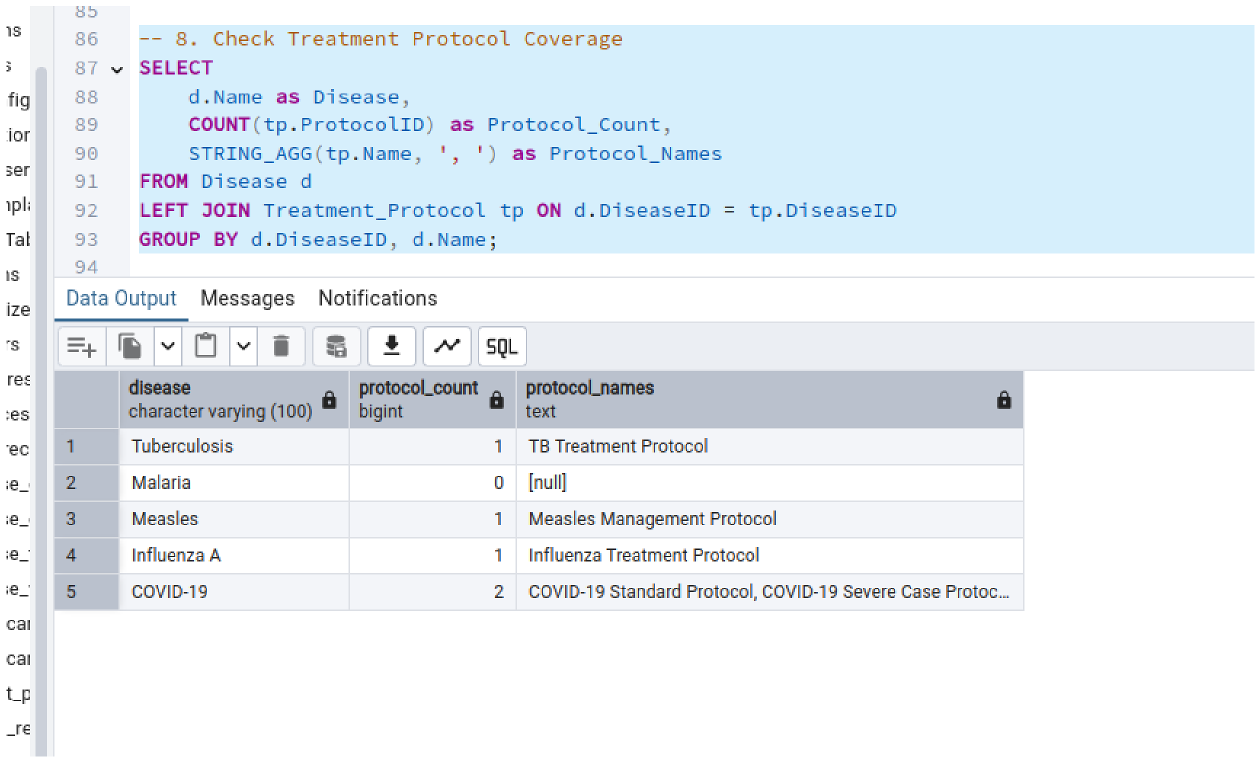
	check_type text	issue_count bigint
1	Orphaned Tests	0
2	Invalid Provider References	0
3	Invalid Disease References	0



```sql
-- 7. Verify Resource Distribution
SELECT
    hf.Name as Facility,
    ri.ResourceType,
    SUM(ri.Quantity) as Total_Quantity,
    MIN(ri.LastRestocked) as Last_Restock_Date
FROM Healthcare_Facility hf
JOIN Resource_Inventory ri ON hf.FacilityID = ri.FacilityID
GROUP BY hf.Name, ri.ResourceType
ORDER BY hf.Name, ri.ResourceType;

-- 8. Check Treatment Protocol Coverage
```

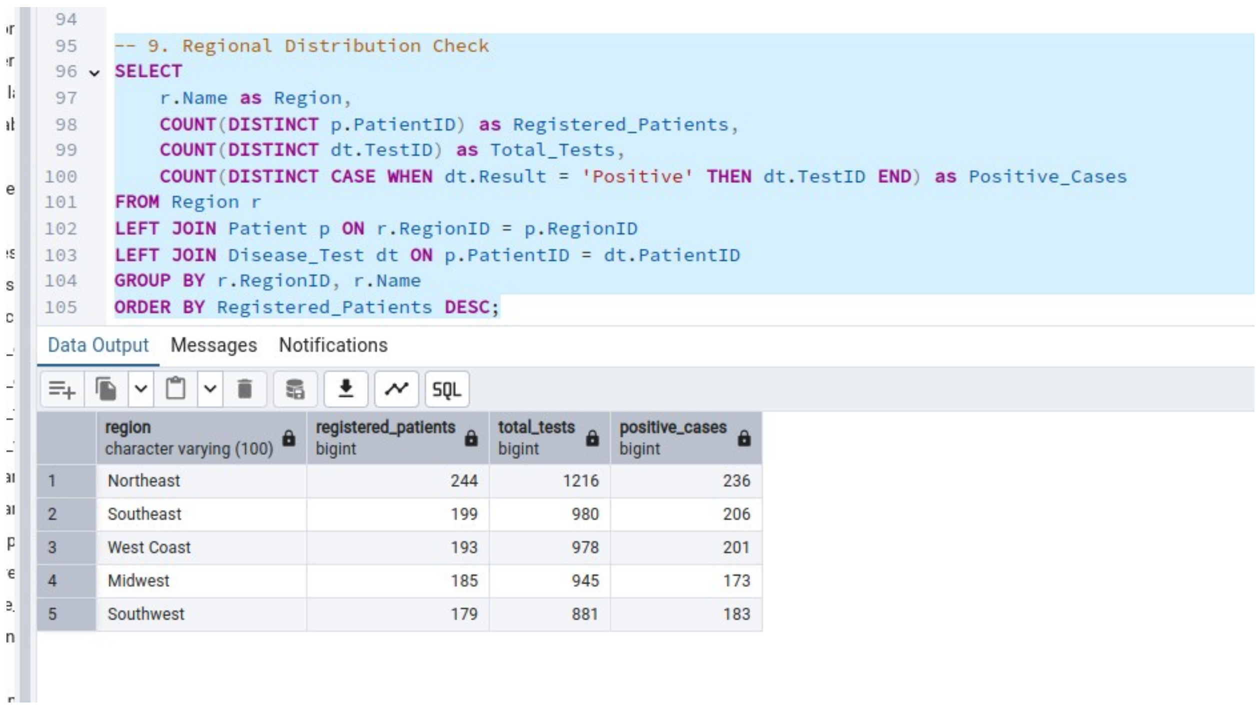
	facility character varying (200)	resourcetype character varying (100)	total_quantity bigint	last_restock_date date
1	Metropolitan General	Medications	371	2023-12-01
2	Metropolitan General	PPE	973	2023-12-01
3	Metropolitan General	Test Kits	949	2023-12-01
4	Metropolitan General	Ventilators	1059	2023-12-01
5	Midwest Health	Medications	393	2023-12-01
6	Midwest Health	PPE	865	2023-12-01
7	Midwest Health	Test Kits	1053	2023-12-01
8	Midwest Health	Ventilators	1094	2023-12-01
9	Pacific Medical Center	Medications	920	2023-12-01
10	Pacific Medical Center	PPE	894	2023-12-01
11	Pacific Medical Center	Test Kits	682	2023-12-01
12	Pacific Medical Center	Ventilators	1024	2023-12-01
13	Southeast Regional	Medications	264	2023-12-01
14	Southeast Regional	PPE	560	2023-12-01
15	Southeast Regional	Test Kits	1074	2023-12-01
16	Southeast Regional	Ventilators	870	2023-12-01
17	Southwest Memorial	Medications	596	2023-12-01
18	Southwest Memorial	PPE	333	2023-12-01
19	Southwest Memorial	Test Kits	490	2023-12-01

Total rows: 20   Query complete 00:00:00.150



```sql
-- 8. Check Treatment Protocol Coverage
SELECT
    d.Name as Disease,
    COUNT(tp.ProtocolID) as Protocol_Count,
    STRING_AGG(tp.Name, ', ') as Protocol_Names
FROM Disease d
LEFT JOIN Treatment_Protocol tp ON d.DiseaseID = tp.DiseaseID
GROUP BY d.DiseaseID, d.Name;
```

	disease character varying (100)	protocol_count bigint	protocol_names text
1	Tuberculosis	1	TB Treatment Protocol
2	Malaria	0	[null]
3	Measles	1	Measles Management Protocol
4	Influenza A	1	Influenza Treatment Protocol
5	COVID-19	2	COVID-19 Standard Protocol, COVID-19 Severe Case Protoc...

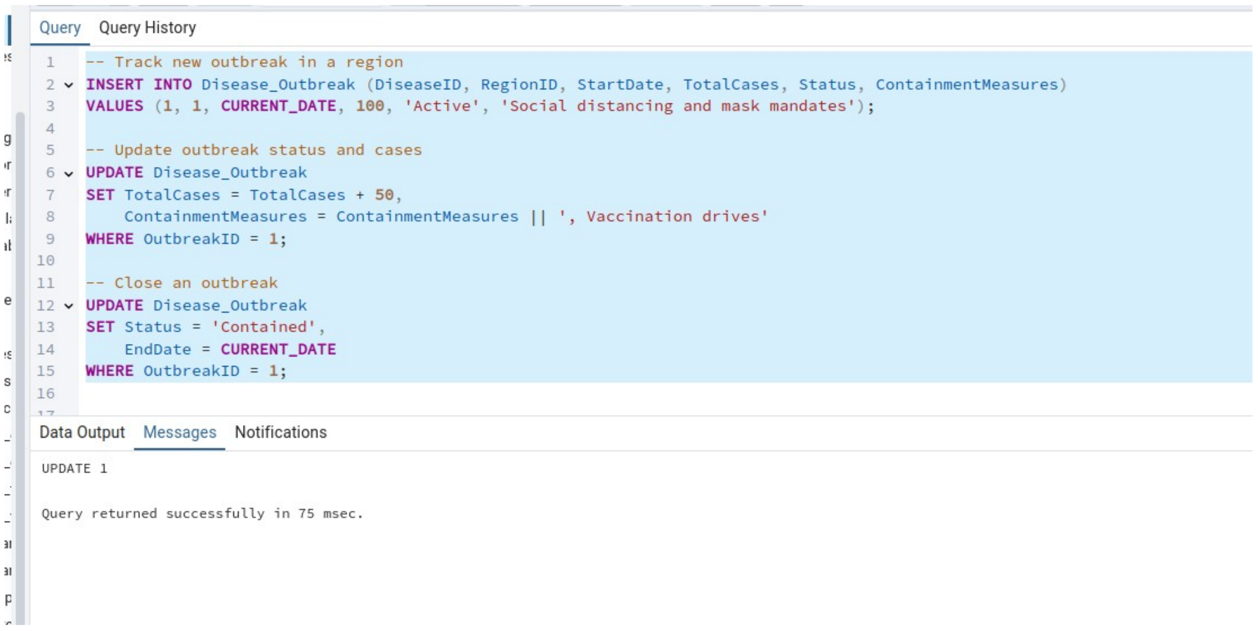


```sql
-- 9. Regional Distribution Check
SELECT
    r.Name as Region,
    COUNT(DISTINCT p.PatientID) as Registered_Patients,
    COUNT(DISTINCT dt.TestID) as Total_Tests,
    COUNT(DISTINCT CASE WHEN dt.Result = 'Positive' THEN dt.TestID END) as Positive_Cases
FROM Region r
LEFT JOIN Patient p ON r.RegionID = p.RegionID
LEFT JOIN Disease_Test dt ON p.PatientID = dt.PatientID
GROUP BY r.RegionID, r.Name
ORDER BY Registered_Patients DESC;
```

	region character varying (100)	registered_patients bigint	total_tests bigint	positive_cases bigint
1	Northeast	244	1216	236
2	Southeast	199	980	206
3	West Coast	193	978	201
4	Midwest	185	945	173
5	Southwest	179	881	183

Creating Operational Queries and DML Operations.

Disease Outbreak Tracking:



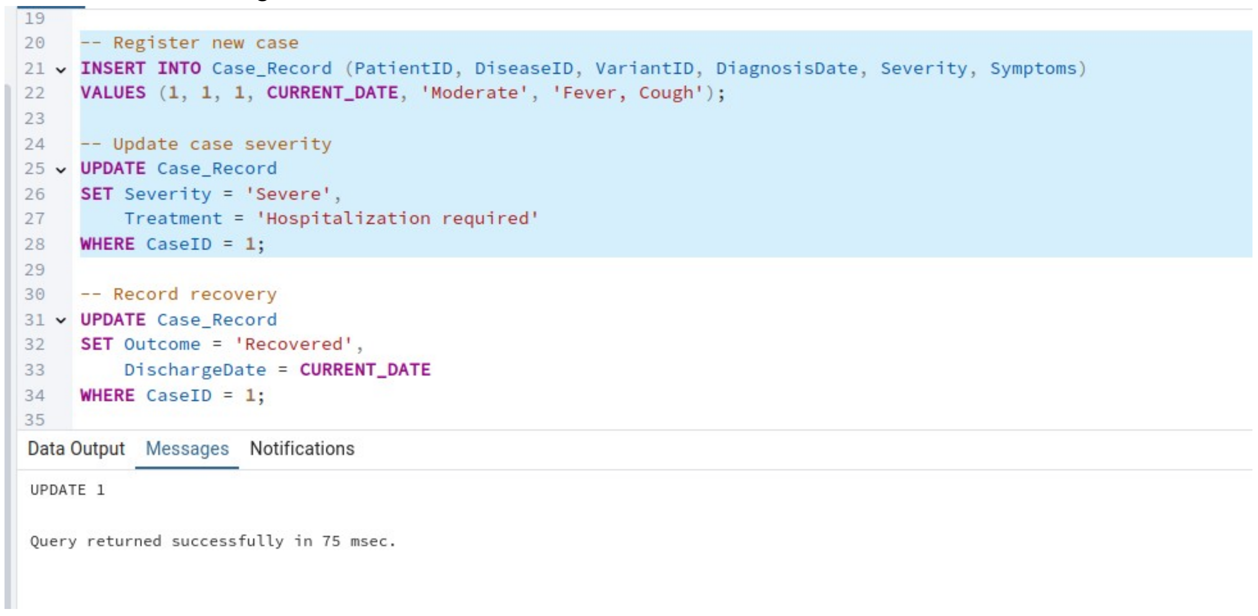
Query Query History

```sql
-- Track new outbreak in a region
INSERT INTO Disease_Outbreak (DiseaseID, RegionID, StartDate, TotalCases, Status, ContainmentMeasures)
VALUES (1, 1, CURRENT_DATE, 100, 'Active', 'Social distancing and mask mandates');

-- Update outbreak status and cases
UPDATE Disease_Outbreak
SET TotalCases = TotalCases + 50,
    ContainmentMeasures = ContainmentMeasures || ', Vaccination drives'
WHERE OutbreakID = 1;

-- Close an outbreak
UPDATE Disease_Outbreak
SET Status = 'Contained',
    EndDate = CURRENT_DATE
WHERE OutbreakID = 1;
```

Data Output Messages Notifications

```
UPDATE 1

Query returned successfully in 75 msec.
```

Patient Case Management:

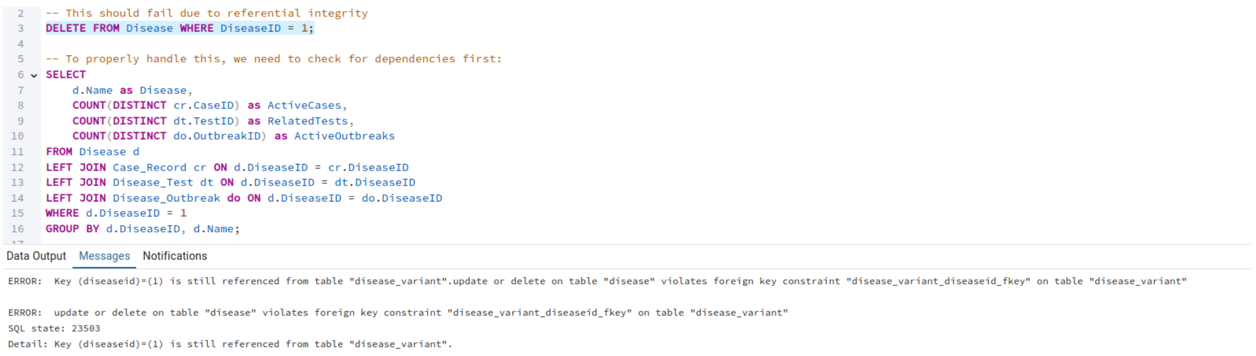


```sql
-- Register new case
INSERT INTO Case_Record (PatientID, DiseaseID, VariantID, DiagnosisDate, Severity, Symptoms)
VALUES (1, 1, 1, CURRENT_DATE, 'Moderate', 'Fever, Cough');

-- Update case severity
UPDATE Case_Record
SET Severity = 'Severe',
    Treatment = 'Hospitalization required'
WHERE CaseID = 1;

-- Record recovery
UPDATE Case_Record
SET Outcome = 'Recovered',
    DischargeDate = CURRENT_DATE
WHERE CaseID = 1;
```

Data Output Messages Notifications

```
UPDATE 1

Query returned successfully in 75 msec.
```

demonstrate referential integrity scenarios that show how our database handles related records across tables, including cascade effects and constraint enforcement.

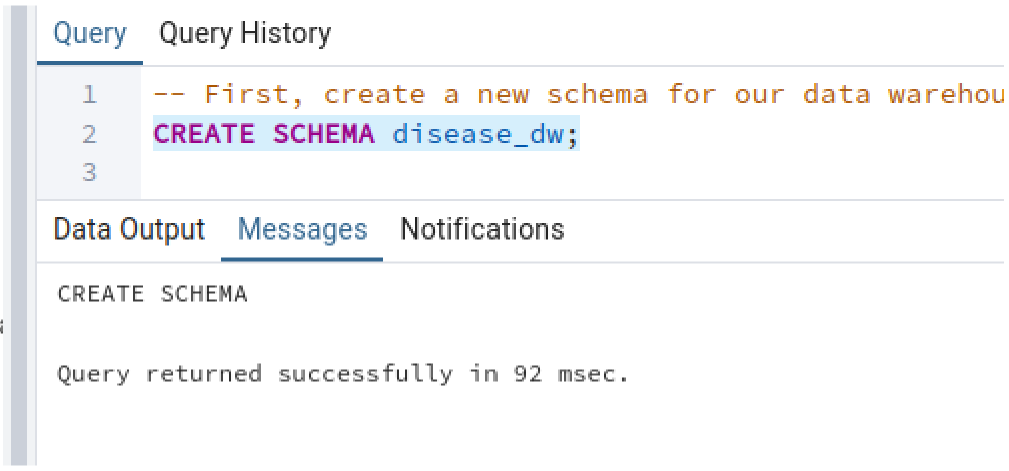


```sql
-- This should fail due to referential integrity
DELETE FROM Disease WHERE DiseaseID = 1;

-- To properly handle this, we need to check for dependencies first:
SELECT
    d.Name as Disease,
    COUNT(DISTINCT cr.CaseID) as ActiveCases,
    COUNT(DISTINCT dt.TestID) as RelatedTests,
    COUNT(DISTINCT do.OutbreakID) as ActiveOutbreaks
FROM Disease d
LEFT JOIN Case_Record cr ON d.DiseaseID = cr.DiseaseID
LEFT JOIN Disease_Test dt ON d.DiseaseID = dt.DiseaseID
LEFT JOIN Disease_Outbreak do ON d.DiseaseID = do.DiseaseID
WHERE d.DiseaseID = 1
GROUP BY d.DiseaseID, d.Name;
```

```
ERROR:  Key (diseaseid)=(1) is still referenced from table "disease_variant".update or delete on table "disease" violates foreign key constraint "disease_variant_diseaseid_fkey" on table "disease_variant"

ERROR:  update or delete on table "disease" violates foreign key constraint "disease_variant_diseaseid_fkey" on table "disease_variant"
SQL state: 23503
Detail: Key (diseaseid)=(1) is still referenced from table "disease_variant".
```

.....

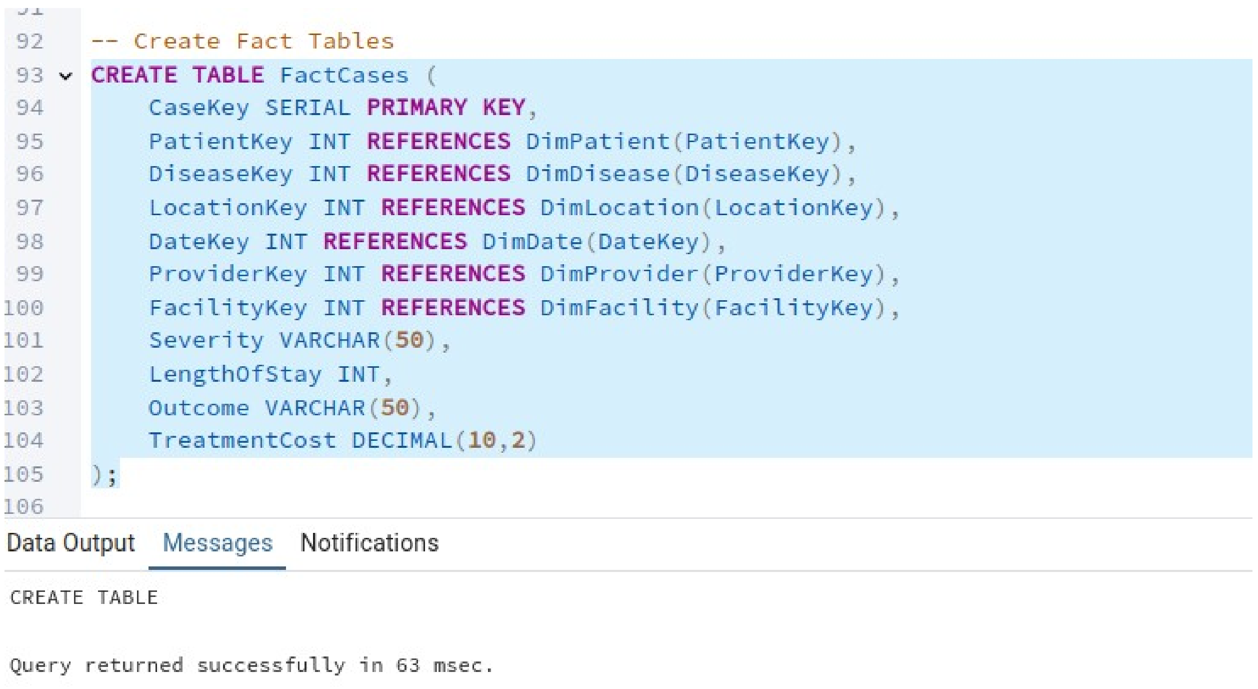
creating the dimensional model (data warehouse) for analytical purposes



```sql
-- First, create a new schema for our data warehou
CREATE SCHEMA disease_dw;
```

```
CREATE SCHEMA

Query returned successfully in 92 msec.
```

.....s

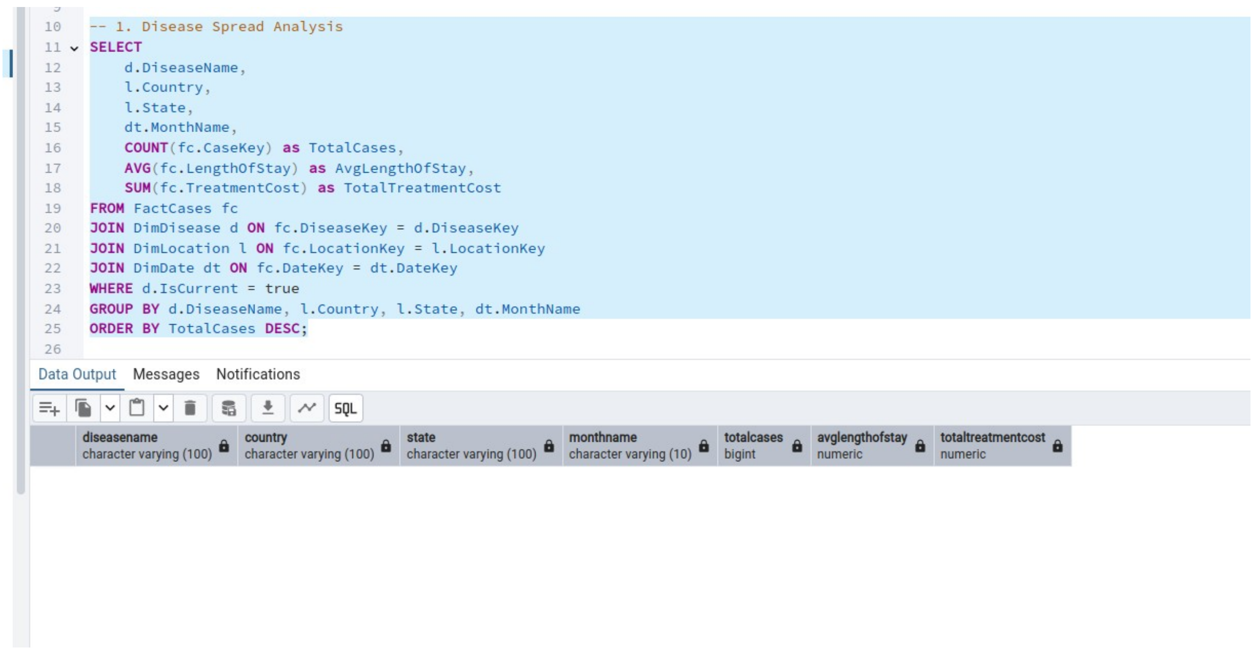


```
-- Create Fact Tables
CREATE TABLE FactCases (
    CaseKey SERIAL PRIMARY KEY,
    PatientKey INT REFERENCES DimPatient(PatientKey),
    DiseaseKey INT REFERENCES DimDisease(DiseaseKey),
    LocationKey INT REFERENCES DimLocation(LocationKey),
    DateKey INT REFERENCES DimDate(DateKey),
    ProviderKey INT REFERENCES DimProvider(ProviderKey),
    FacilityKey INT REFERENCES DimFacility(FacilityKey),
    Severity VARCHAR(50),
    LengthOfStay INT,
    Outcome VARCHAR(50),
    TreatmentCost DECIMAL(10,2)
);
```

```
CREATE TABLE

Query returned successfully in 63 msec.
```

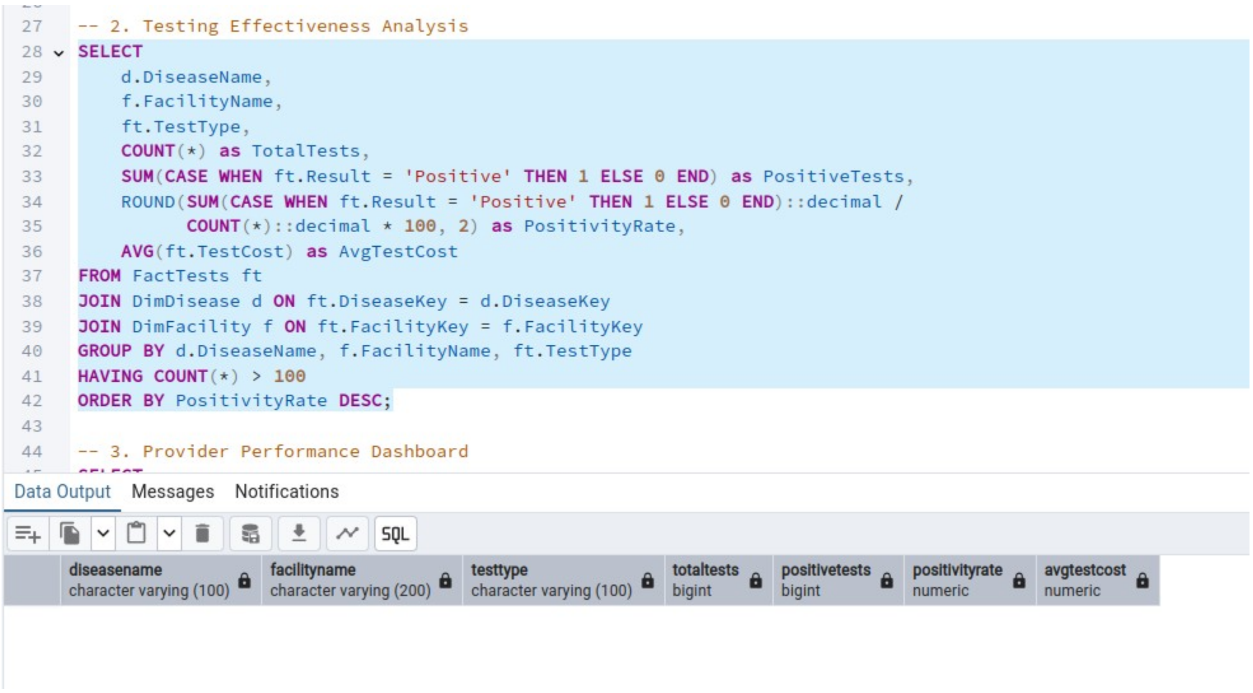
...

Some analytical queries that demonstrate the power of our dimensional model.



```
-- 1. Disease Spread Analysis
SELECT
    d.DiseaseName,
    l.Country,
    l.State,
    dt.MonthName,
    COUNT(fc.CaseKey) as TotalCases,
    AVG(fc.LengthOfStay) as AvgLengthOfStay,
    SUM(fc.TreatmentCost) as TotalTreatmentCost
FROM FactCases fc
JOIN DimDisease d ON fc.DiseaseKey = d.DiseaseKey
JOIN DimLocation l ON fc.LocationKey = l.LocationKey
JOIN DimDate dt ON fc.DateKey = dt.DateKey
WHERE d.IsCurrent = true
GROUP BY d.DiseaseName, l.Country, l.State, dt.MonthName
ORDER BY TotalCases DESC;
```

diseasename character varying (100)	country character varying (100)	state character varying (100)	monthname character varying (10)	totalcases bigint	avglengthofstay numeric	totaltreatmentcost numeric



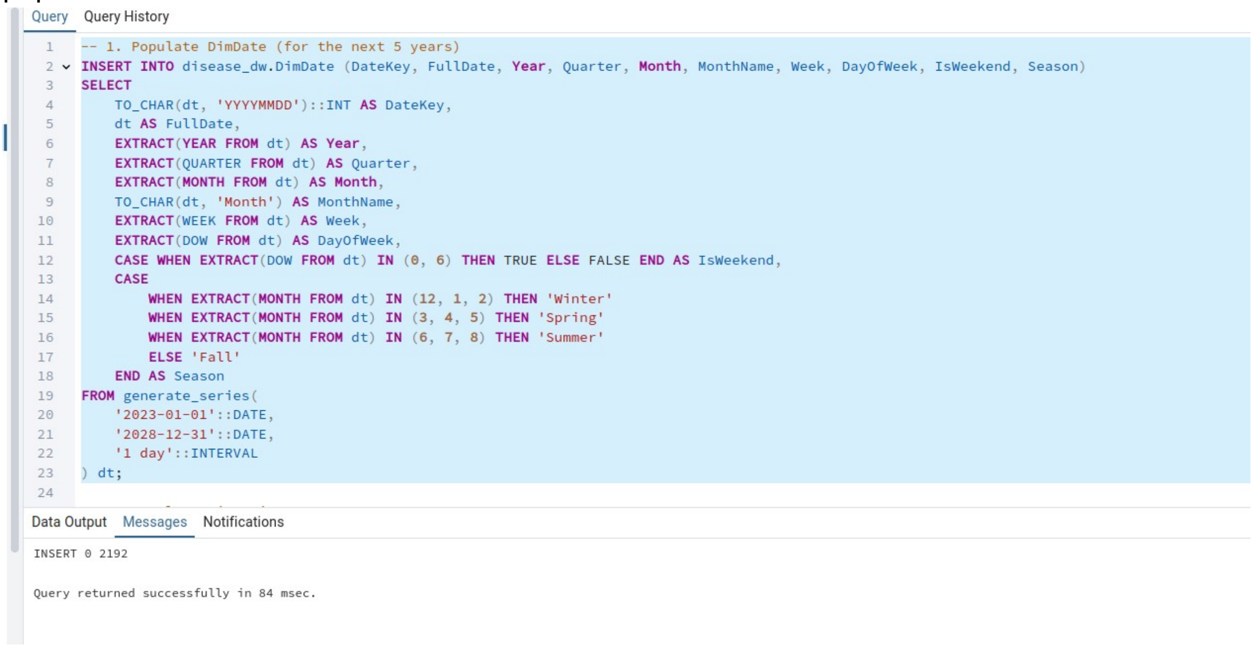
```sql
-- 2. Testing Effectiveness Analysis
SELECT
    d.DiseaseName,
    f.FacilityName,
    ft.TestType,
    COUNT(*) as TotalTests,
    SUM(CASE WHEN ft.Result = 'Positive' THEN 1 ELSE 0 END) as PositiveTests,
    ROUND(SUM(CASE WHEN ft.Result = 'Positive' THEN 1 ELSE 0 END)::decimal /
          COUNT(*)::decimal * 100, 2) as PositivityRate,
    AVG(ft.TestCost) as AvgTestCost
FROM FactTests ft
JOIN DimDisease d ON ft.DiseaseKey = d.DiseaseKey
JOIN DimFacility f ON ft.FacilityKey = f.FacilityKey
GROUP BY d.DiseaseName, f.FacilityName, ft.TestType
HAVING COUNT(*) > 100
ORDER BY PositivityRate DESC;

-- 3. Provider Performance Dashboard
```

Data Output   Messages   Notifications

diseasename character varying (100)	facilityname character varying (200)	testtype character varying (100)	totaltests bigint	positivetests bigint	positivityrate numeric	avgtestcost numeric

...

populate the dimension tables



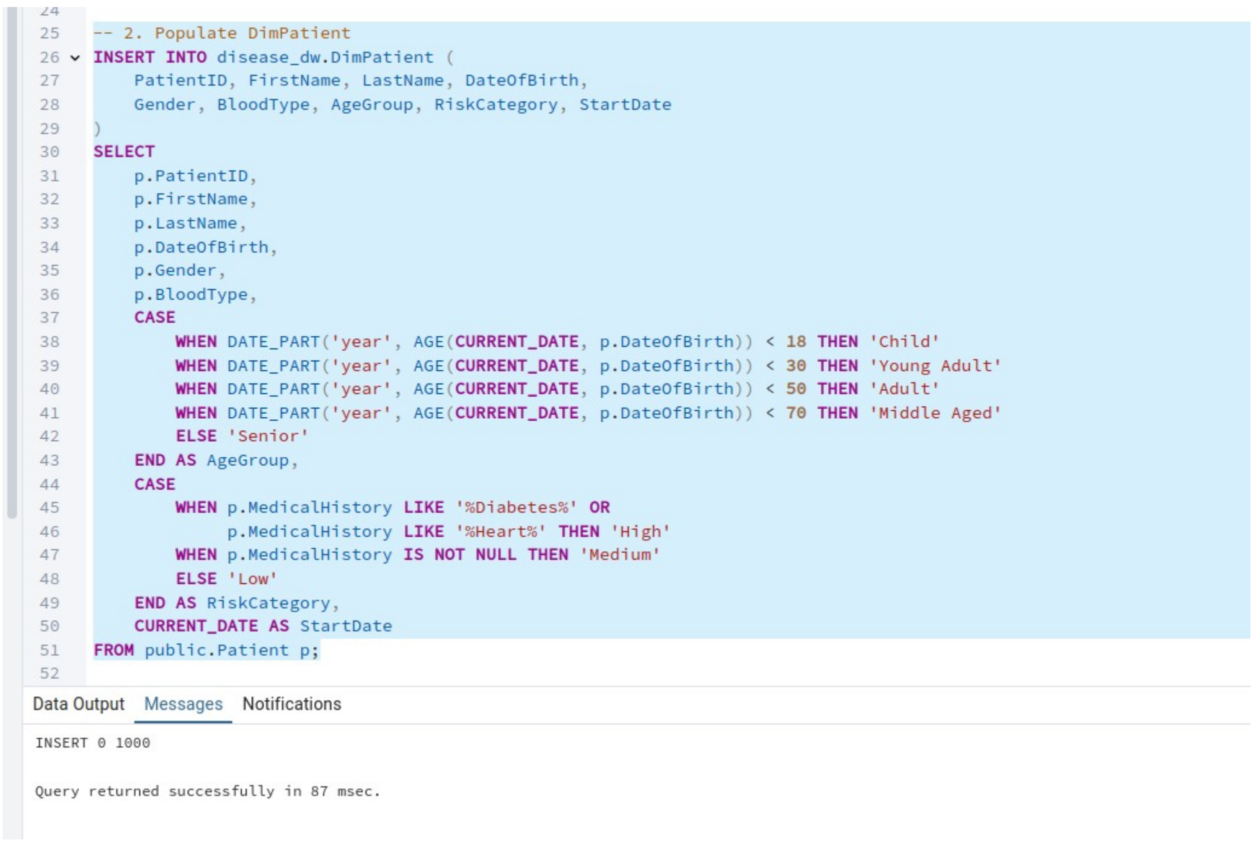
Query   Query History

```sql
-- 1. Populate DimDate (for the next 5 years)
INSERT INTO disease_dw.DimDate (DateKey, FullDate, Year, Quarter, Month, MonthName, Week, DayOfWeek, IsWeekend, Season)
SELECT
    TO_CHAR(dt, 'YYYYMMDD')::INT AS DateKey,
    dt AS FullDate,
    EXTRACT(YEAR FROM dt) AS Year,
    EXTRACT(QUARTER FROM dt) AS Quarter,
    EXTRACT(MONTH FROM dt) AS Month,
    TO_CHAR(dt, 'Month') AS MonthName,
    EXTRACT(WEEK FROM dt) AS Week,
    EXTRACT(DOW FROM dt) AS DayOfWeek,
    CASE WHEN EXTRACT(DOW FROM dt) IN (0, 6) THEN TRUE ELSE FALSE END AS IsWeekend,
    CASE
        WHEN EXTRACT(MONTH FROM dt) IN (12, 1, 2) THEN 'Winter'
        WHEN EXTRACT(MONTH FROM dt) IN (3, 4, 5) THEN 'Spring'
        WHEN EXTRACT(MONTH FROM dt) IN (6, 7, 8) THEN 'Summer'
        ELSE 'Fall'
    END AS Season
FROM generate_series(
    '2023-01-01'::DATE,
    '2028-12-31'::DATE,
    '1 day'::INTERVAL
) dt;
```

Data Output   Messages   Notifications

```
INSERT 0 2192

Query returned successfully in 84 msec.
```

..



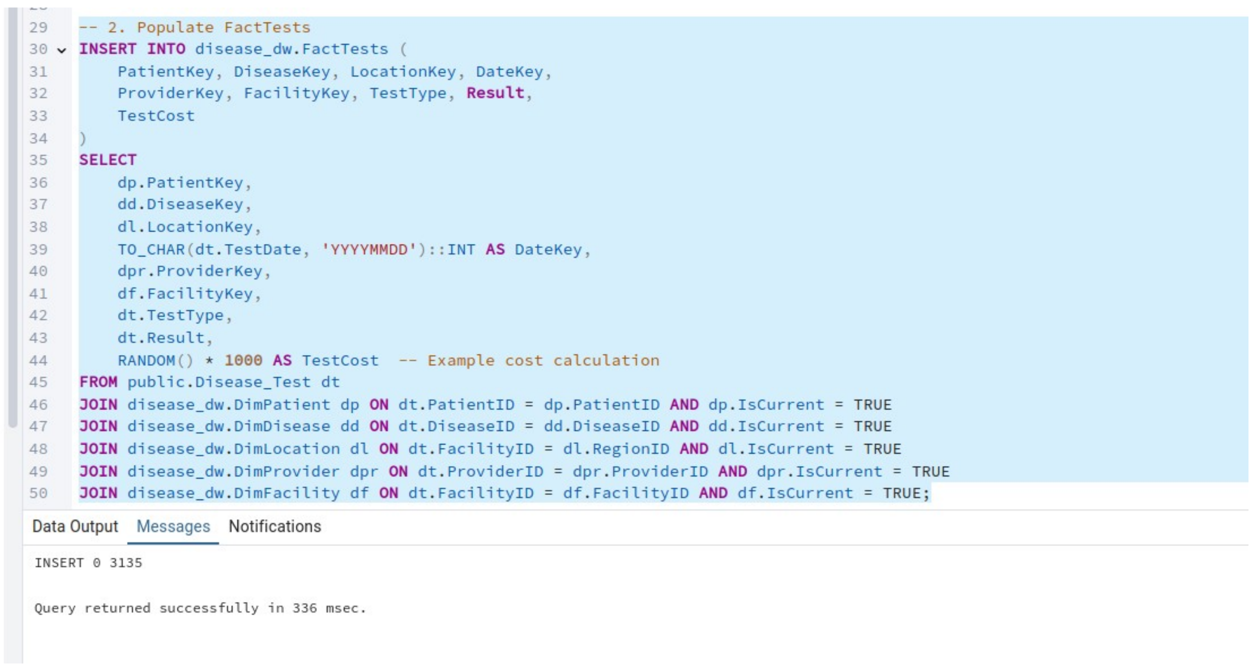
```
-- 2. Populate DimPatient
INSERT INTO disease_dw.DimPatient (
    PatientID, FirstName, LastName, DateOfBirth,
    Gender, BloodType, AgeGroup, RiskCategory, StartDate
)
SELECT
    p.PatientID,
    p.FirstName,
    p.LastName,
    p.DateOfBirth,
    p.Gender,
    p.BloodType,
    CASE
        WHEN DATE_PART('year', AGE(CURRENT_DATE, p.DateOfBirth)) < 18 THEN 'Child'
        WHEN DATE_PART('year', AGE(CURRENT_DATE, p.DateOfBirth)) < 30 THEN 'Young Adult'
        WHEN DATE_PART('year', AGE(CURRENT_DATE, p.DateOfBirth)) < 50 THEN 'Adult'
        WHEN DATE_PART('year', AGE(CURRENT_DATE, p.DateOfBirth)) < 70 THEN 'Middle Aged'
        ELSE 'Senior'
    END AS AgeGroup,
    CASE
        WHEN p.MedicalHistory LIKE '%Diabetes%' OR
             p.MedicalHistory LIKE '%Heart%' THEN 'High'
        WHEN p.MedicalHistory IS NOT NULL THEN 'Medium'
        ELSE 'Low'
    END AS RiskCategory,
    CURRENT_DATE AS StartDate
FROM public.Patient p;
```

Data Output   Messages   Notifications

```
INSERT 0 1000

Query returned successfully in 87 msec.
```

\..

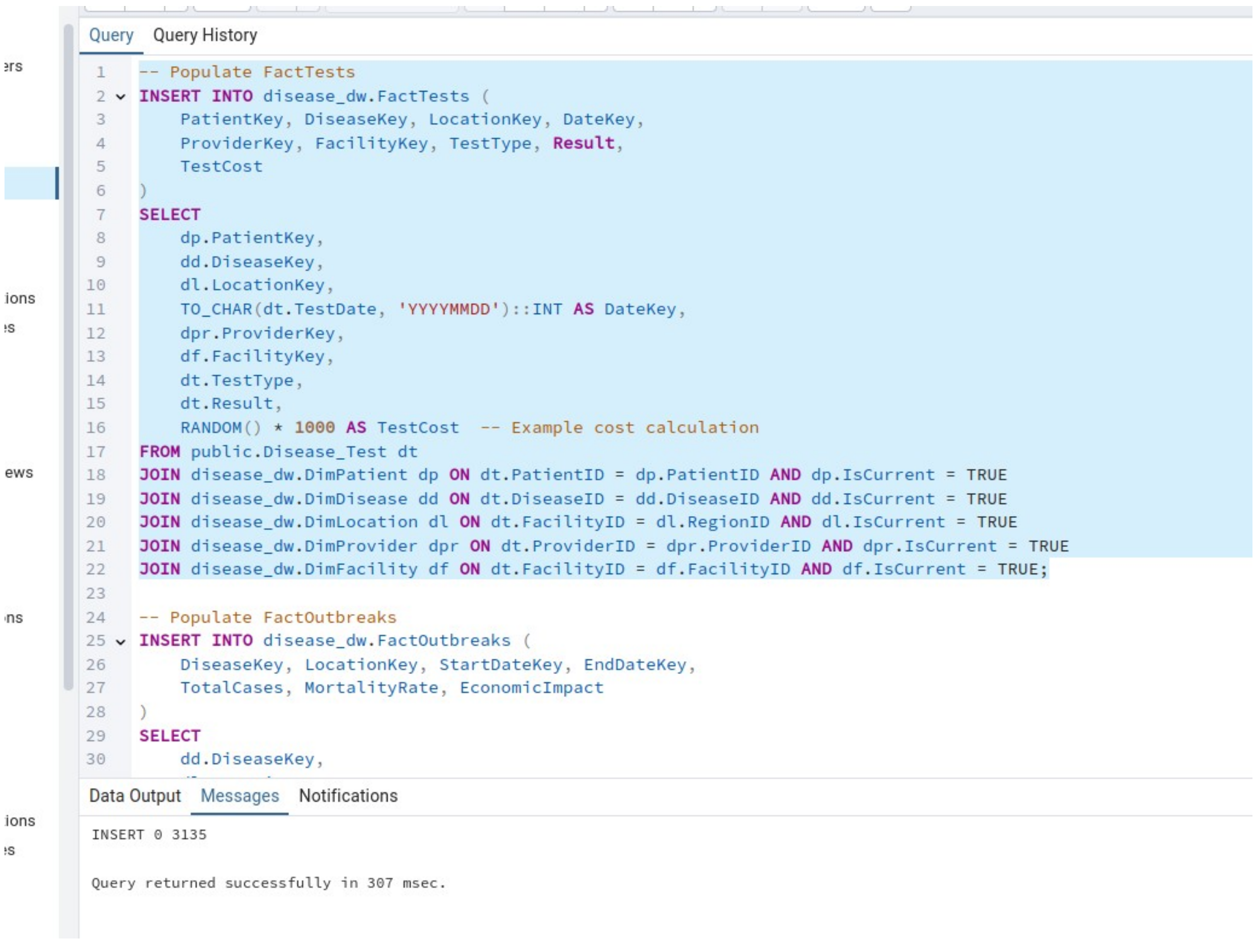
populate the fact tables



```
-- 2. Populate FactTests
INSERT INTO disease_dw.FactTests (
    PatientKey, DiseaseKey, LocationKey, DateKey,
    ProviderKey, FacilityKey, TestType, Result,
    TestCost
)
SELECT
    dp.PatientKey,
    dd.DiseaseKey,
    dl.LocationKey,
    TO_CHAR(dt.TestDate, 'YYYYMMDD')::INT AS DateKey,
    dpr.ProviderKey,
    df.FacilityKey,
    dt.TestType,
    dt.Result,
    RANDOM() * 1000 AS TestCost  -- Example cost calculation
FROM public.Disease_Test dt
JOIN disease_dw.DimPatient dp ON dt.PatientID = dp.PatientID AND dp.IsCurrent = TRUE
JOIN disease_dw.DimDisease dd ON dt.DiseaseID = dd.DiseaseID AND dd.IsCurrent = TRUE
JOIN disease_dw.DimLocation dl ON dt.FacilityID = dl.RegionID AND dl.IsCurrent = TRUE
JOIN disease_dw.DimProvider dpr ON dt.ProviderID = dpr.ProviderID AND dpr.IsCurrent = TRUE
JOIN disease_dw.DimFacility df ON dt.FacilityID = df.FacilityID AND df.IsCurrent = TRUE;
```

Data Output   Messages   Notifications

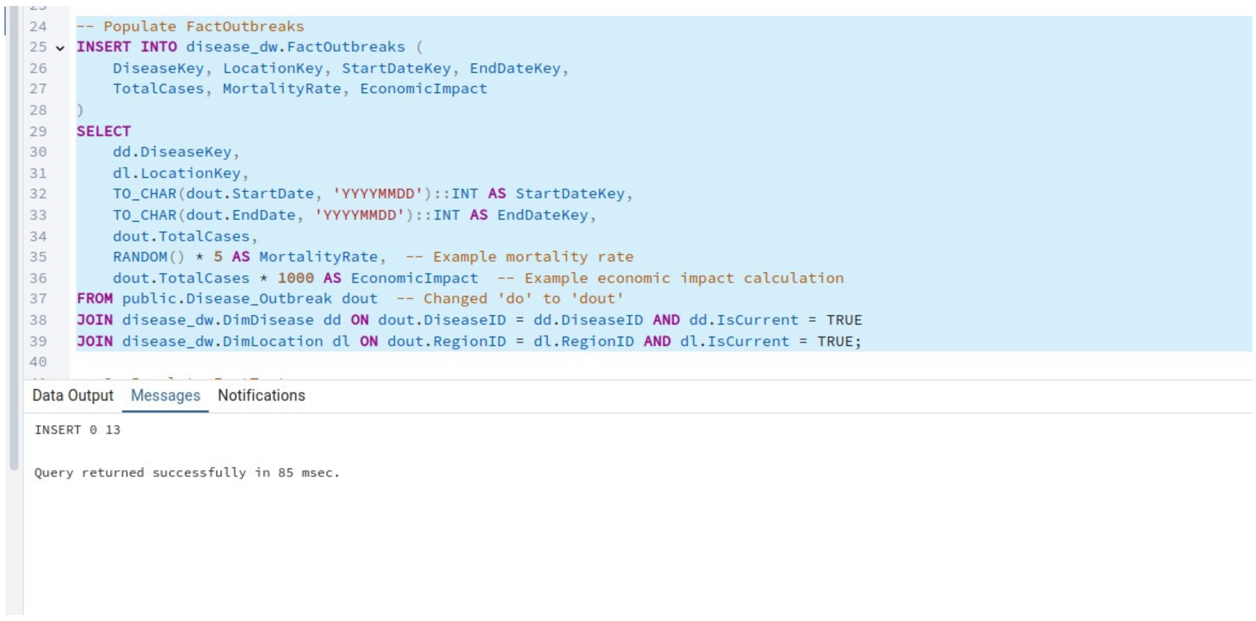
```
INSERT 0 3135

Query returned successfully in 336 msec.
```



```sql
-- Populate FactTests
INSERT INTO disease_dw.FactTests (
    PatientKey, DiseaseKey, LocationKey, DateKey,
    ProviderKey, FacilityKey, TestType, Result,
    TestCost
)
SELECT
    dp.PatientKey,
    dd.DiseaseKey,
    dl.LocationKey,
    TO_CHAR(dt.TestDate, 'YYYYMMDD')::INT AS DateKey,
    dpr.ProviderKey,
    df.FacilityKey,
    dt.TestType,
    dt.Result,
    RANDOM() * 1000 AS TestCost  -- Example cost calculation
FROM public.Disease_Test dt
JOIN disease_dw.DimPatient dp ON dt.PatientID = dp.PatientID AND dp.IsCurrent = TRUE
JOIN disease_dw.DimDisease dd ON dt.DiseaseID = dd.DiseaseID AND dd.IsCurrent = TRUE
JOIN disease_dw.DimLocation dl ON dt.FacilityID = dl.RegionID AND dl.IsCurrent = TRUE
JOIN disease_dw.DimProvider dpr ON dt.ProviderID = dpr.ProviderID AND dpr.IsCurrent = TRUE
JOIN disease_dw.DimFacility df ON dt.FacilityID = df.FacilityID AND df.IsCurrent = TRUE;

-- Populate FactOutbreaks
INSERT INTO disease_dw.FactOutbreaks (
    DiseaseKey, LocationKey, StartDateKey, EndDateKey,
    TotalCases, MortalityRate, EconomicImpact
)
SELECT
    dd.DiseaseKey,
```

```
INSERT 0 3135

Query returned successfully in 307 msec.
```

..



```sql
-- Populate FactOutbreaks
INSERT INTO disease_dw.FactOutbreaks (
    DiseaseKey, LocationKey, StartDateKey, EndDateKey,
    TotalCases, MortalityRate, EconomicImpact
)
SELECT
    dd.DiseaseKey,
    dl.LocationKey,
    TO_CHAR(dout.StartDate, 'YYYYMMDD')::INT AS StartDateKey,
    TO_CHAR(dout.EndDate, 'YYYYMMDD')::INT AS EndDateKey,
    dout.TotalCases,
    RANDOM() * 5 AS MortalityRate,  -- Example mortality rate
    dout.TotalCases * 1000 AS EconomicImpact  -- Example economic impact calculation
FROM public.Disease_Outbreak dout  -- Changed 'do' to 'dout'
JOIN disease_dw.DimDisease dd ON dout.DiseaseID = dd.DiseaseID AND dd.IsCurrent = TRUE
JOIN disease_dw.DimLocation dl ON dout.RegionID = dl.RegionID AND dl.IsCurrent = TRUE;
```

```
INSERT 0 13

Query returned successfully in 85 msec.
```

..

a series of analytical queries to test our data warehouse and reveal meaningful insights:

s

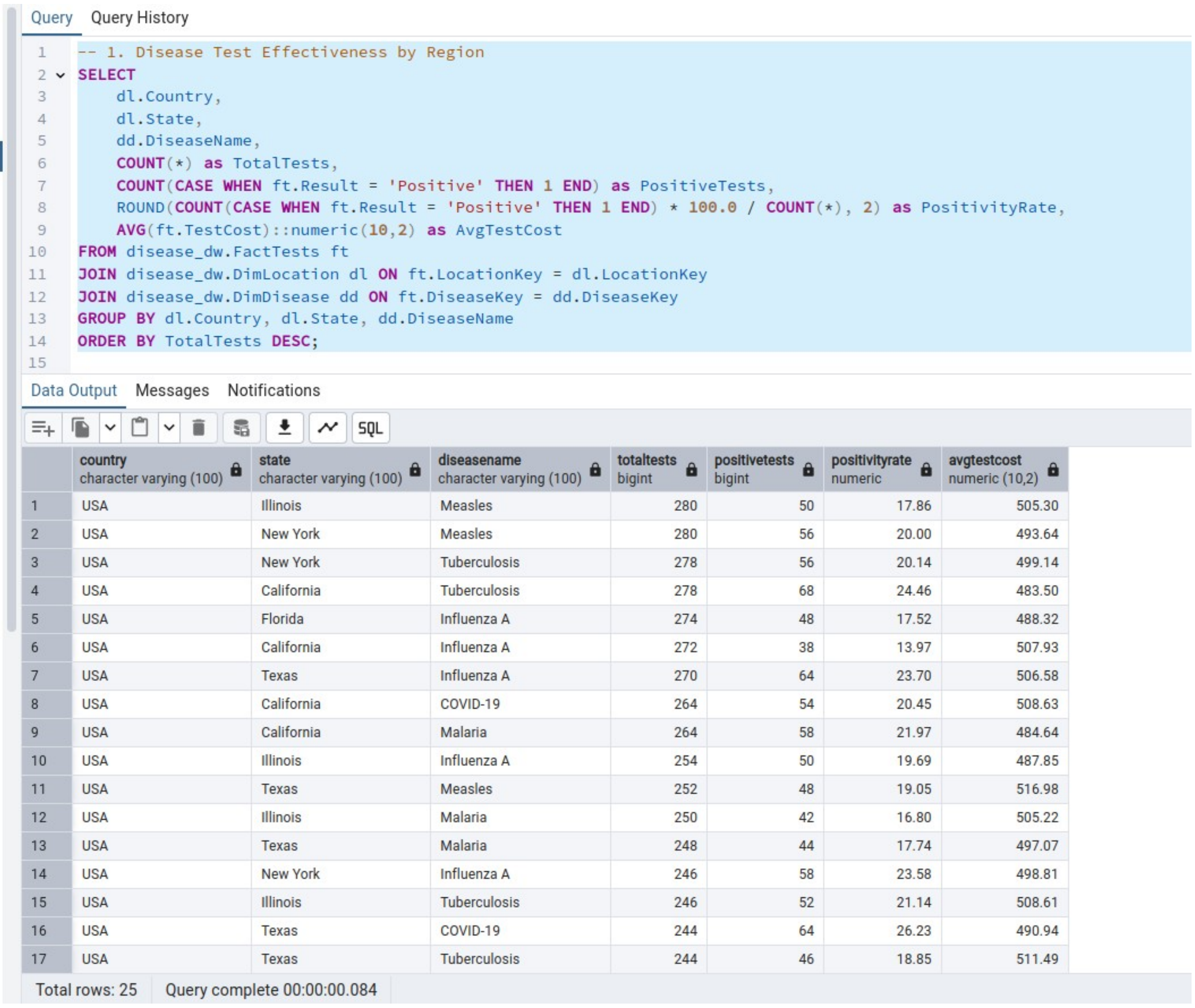
ns

vs

s

ns

vs


Query Query History

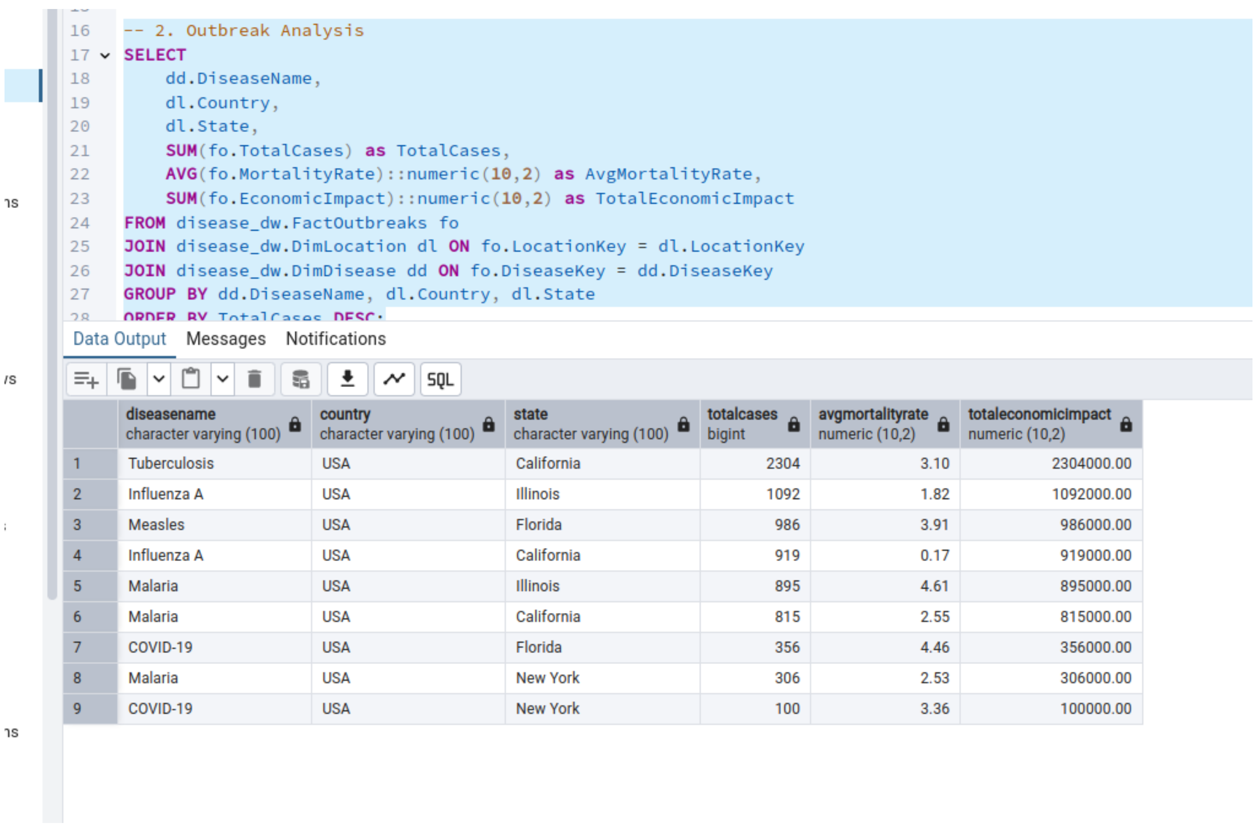
```
-- 1. Disease Test Effectiveness by Region
SELECT
    dl.Country,
    dl.State,
    dd.DiseaseName,
    COUNT(*) as TotalTests,
    COUNT(CASE WHEN ft.Result = 'Positive' THEN 1 END) as PositiveTests,
    ROUND(COUNT(CASE WHEN ft.Result = 'Positive' THEN 1 END) * 100.0 / COUNT(*), 2) as PositivityRate,
    AVG(ft.TestCost)::numeric(10,2) as AvgTestCost
FROM disease_dw.FactTests ft
JOIN disease_dw.DimLocation dl ON ft.LocationKey = dl.LocationKey
JOIN disease_dw.DimDisease dd ON ft.DiseaseKey = dd.DiseaseKey
GROUP BY dl.Country, dl.State, dd.DiseaseName
ORDER BY TotalTests DESC;
```

Data Output Messages Notifications

	country character varying (100)	state character varying (100)	diseasename character varying (100)	totaltests bigint	positivetests bigint	positivityrate numeric	avgtestcost numeric (10,2)
1	USA	Illinois	Measles	280	50	17.86	505.30
2	USA	New York	Measles	280	56	20.00	493.64
3	USA	New York	Tuberculosis	278	56	20.14	499.14
4	USA	California	Tuberculosis	278	68	24.46	483.50
5	USA	Florida	Influenza A	274	48	17.52	488.32
6	USA	California	Influenza A	272	38	13.97	507.93
7	USA	Texas	Influenza A	270	64	23.70	506.58
8	USA	California	COVID-19	264	54	20.45	508.63
9	USA	California	Malaria	264	58	21.97	484.64
10	USA	Illinois	Influenza A	254	50	19.69	487.85
11	USA	Texas	Measles	252	48	19.05	516.98
12	USA	Illinois	Malaria	250	42	16.80	505.22
13	USA	Texas	Malaria	248	44	17.74	497.07
14	USA	New York	Influenza A	246	58	23.58	498.81
15	USA	Illinois	Tuberculosis	246	52	21.14	508.61
16	USA	Texas	COVID-19	244	64	26.23	490.94
17	USA	Texas	Tuberculosis	244	46	18.85	511.49

Total rows: 25 Query complete 00:00:00.084

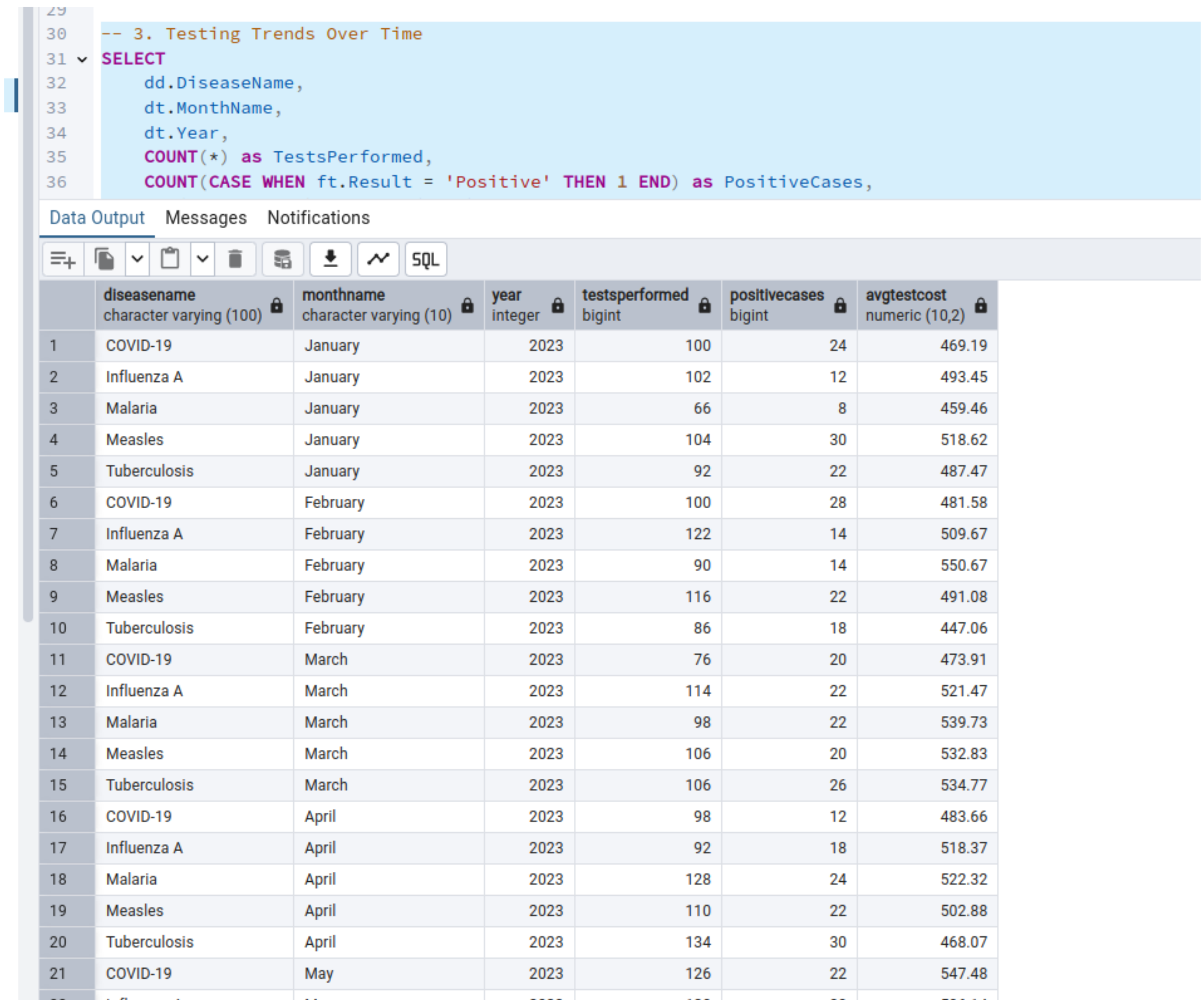
..



```sql
-- 2. Outbreak Analysis
SELECT
    dd.DiseaseName,
    dl.Country,
    dl.State,
    SUM(fo.TotalCases) as TotalCases,
    AVG(fo.MortalityRate)::numeric(10,2) as AvgMortalityRate,
    SUM(fo.EconomicImpact)::numeric(10,2) as TotalEconomicImpact
FROM disease_dw.FactOutbreaks fo
JOIN disease_dw.DimLocation dl ON fo.LocationKey = dl.LocationKey
JOIN disease_dw.DimDisease dd ON fo.DiseaseKey = dd.DiseaseKey
GROUP BY dd.DiseaseName, dl.Country, dl.State
ORDER BY TotalCases DESC;
```

	diseasename character varying (100)	country character varying (100)	state character varying (100)	totalcases bigint	avgmortalityrate numeric (10,2)	totaleconomicimpact numeric (10,2)
1	Tuberculosis	USA	California	2304	3.10	2304000.00
2	Influenza A	USA	Illinois	1092	1.82	1092000.00
3	Measles	USA	Florida	986	3.91	986000.00
4	Influenza A	USA	California	919	0.17	919000.00
5	Malaria	USA	Illinois	895	4.61	895000.00
6	Malaria	USA	California	815	2.55	815000.00
7	COVID-19	USA	Florida	356	4.46	356000.00
8	Malaria	USA	New York	306	2.53	306000.00
9	COVID-19	USA	New York	100	3.36	100000.00

..

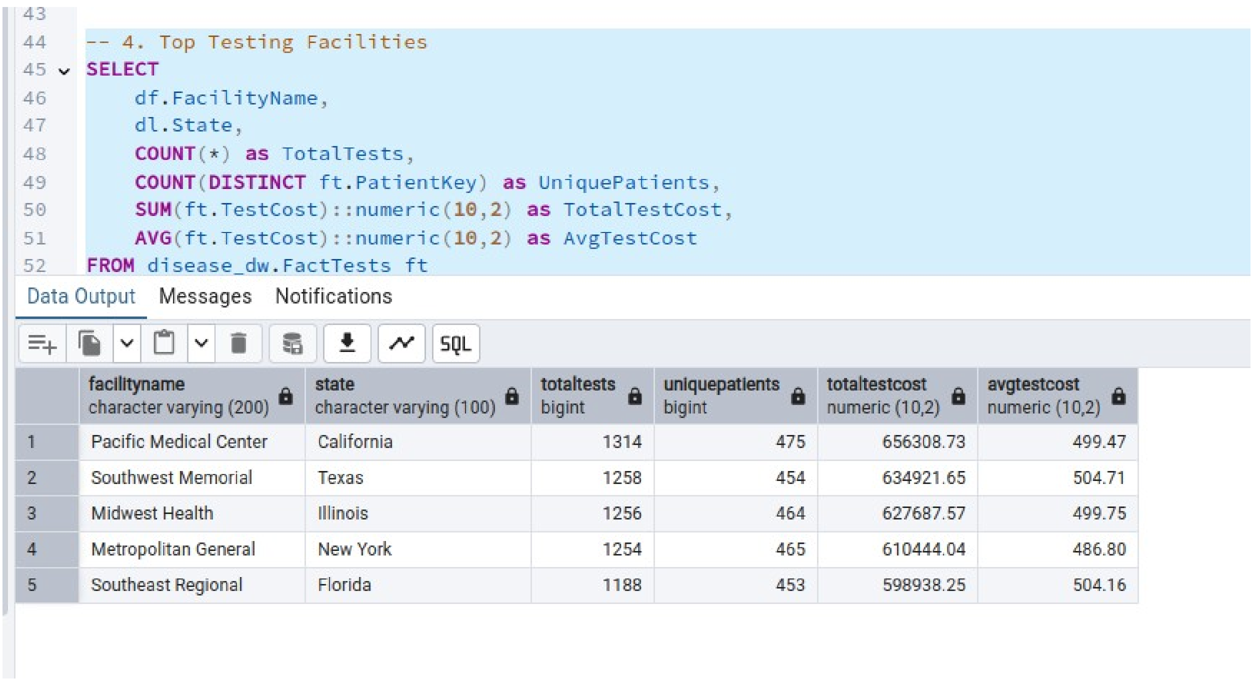


```
-- 3. Testing Trends Over Time
SELECT
    dd.DiseaseName,
    dt.MonthName,
    dt.Year,
    COUNT(*) as TestsPerformed,
    COUNT(CASE WHEN ft.Result = 'Positive' THEN 1 END) as PositiveCases,
```

Data Output Messages Notifications

	diseasename character varying (100)	monthname character varying (10)	year integer	testsperformed bigint	positivecases bigint	avgtestcost numeric (10,2)
1	COVID-19	January	2023	100	24	469.19
2	Influenza A	January	2023	102	12	493.45
3	Malaria	January	2023	66	8	459.46
4	Measles	January	2023	104	30	518.62
5	Tuberculosis	January	2023	92	22	487.47
6	COVID-19	February	2023	100	28	481.58
7	Influenza A	February	2023	122	14	509.67
8	Malaria	February	2023	90	14	550.67
9	Measles	February	2023	116	22	491.08
10	Tuberculosis	February	2023	86	18	447.06
11	COVID-19	March	2023	76	20	473.91
12	Influenza A	March	2023	114	22	521.47
13	Malaria	March	2023	98	22	539.73
14	Measles	March	2023	106	20	532.83
15	Tuberculosis	March	2023	106	26	534.77
16	COVID-19	April	2023	98	12	483.66
17	Influenza A	April	2023	92	18	518.37
18	Malaria	April	2023	128	24	522.32
19	Measles	April	2023	110	22	502.88
20	Tuberculosis	April	2023	134	30	468.07
21	COVID-19	May	2023	126	22	547.48

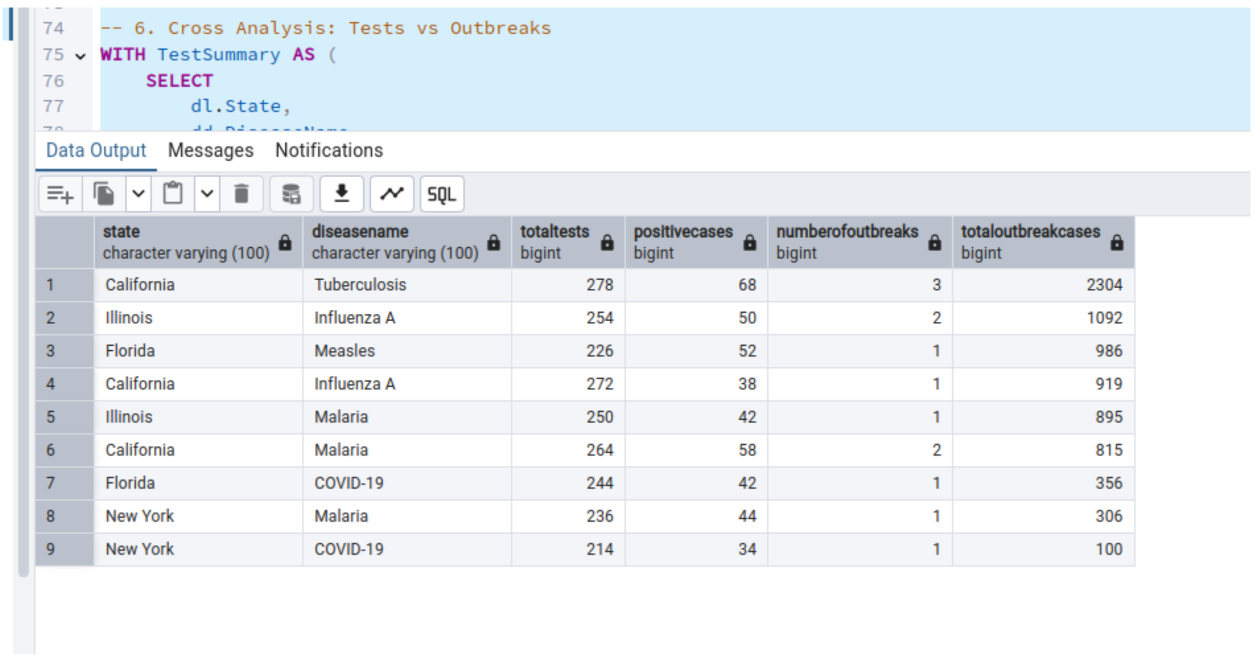
..



```sql
-- 4. Top Testing Facilities
SELECT
    df.FacilityName,
    dl.State,
    COUNT(*) as TotalTests,
    COUNT(DISTINCT ft.PatientKey) as UniquePatients,
    SUM(ft.TestCost)::numeric(10,2) as TotalTestCost,
    AVG(ft.TestCost)::numeric(10,2) as AvgTestCost
FROM disease_dw.FactTests ft
```

	facilityname character varying (200)	state character varying (100)	totaltests bigint	uniquepatients bigint	totaltestcost numeric (10,2)	avgtestcost numeric (10,2)
1	Pacific Medical Center	California	1314	475	656308.73	499.47
2	Southwest Memorial	Texas	1258	454	634921.65	504.71
3	Midwest Health	Illinois	1256	464	627687.57	499.75
4	Metropolitan General	New York	1254	465	610444.04	486.80
5	Southeast Regional	Florida	1188	453	598938.25	504.16

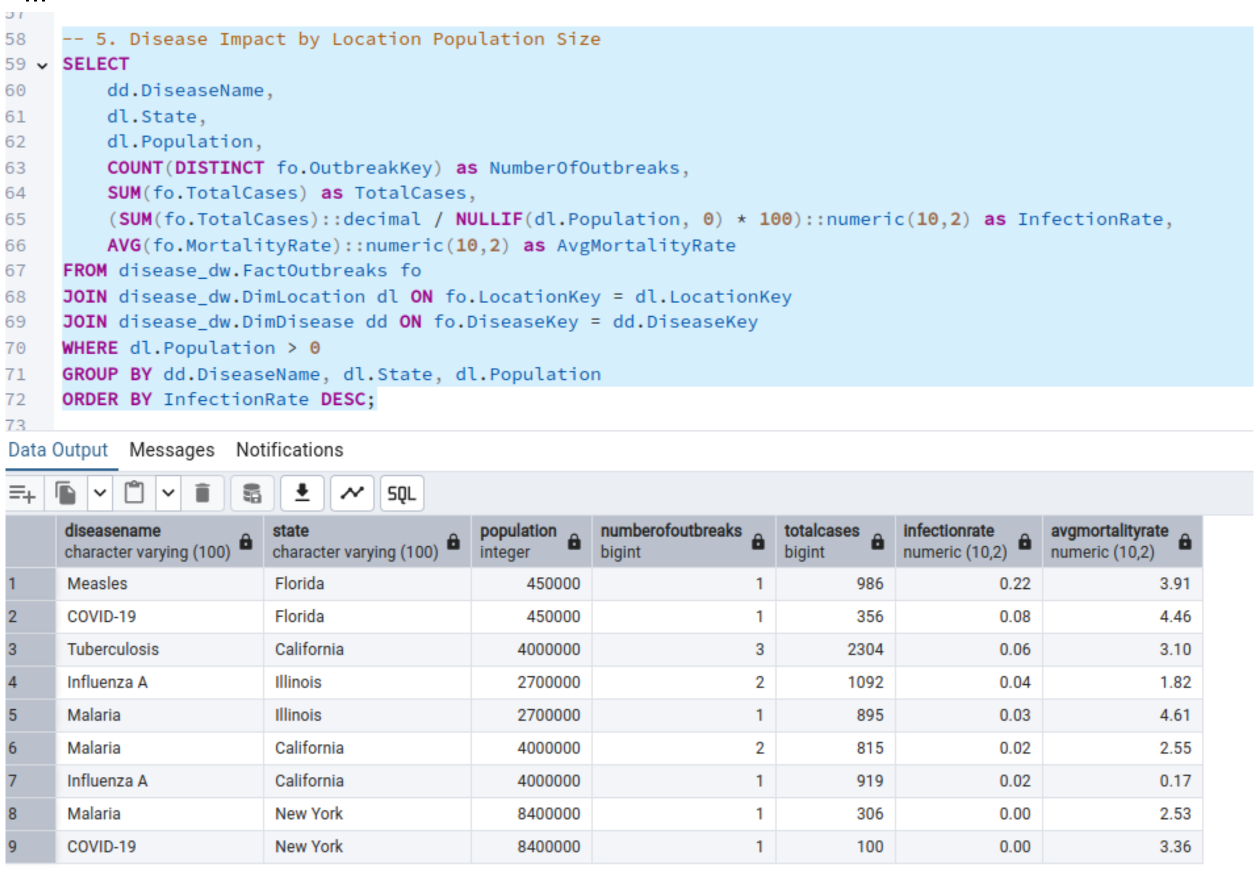
..



```sql
-- 6. Cross Analysis: Tests vs Outbreaks
WITH TestSummary AS (
    SELECT
        dl.State,
```

	state character varying (100)	diseasename character varying (100)	totaltests bigint	positivecases bigint	numberofoutbreaks bigint	totaloutbreakcases bigint
1	California	Tuberculosis	278	68	3	2304
2	Illinois	Influenza A	254	50	2	1092
3	Florida	Measles	226	52	1	986
4	California	Influenza A	272	38	1	919
5	Illinois	Malaria	250	42	1	895
6	California	Malaria	264	58	2	815
7	Florida	COVID-19	244	42	1	356
8	New York	Malaria	236	44	1	306
9	New York	COVID-19	214	34	1	100

>...



```
-- 5. Disease Impact by Location Population Size
SELECT
    dd.DiseaseName,
    dl.State,
    dl.Population,
    COUNT(DISTINCT fo.OutbreakKey) as NumberOfOutbreaks,
    SUM(fo.TotalCases) as TotalCases,
    (SUM(fo.TotalCases)::decimal / NULLIF(dl.Population, 0) * 100)::numeric(10,2) as InfectionRate,
    AVG(fo.MortalityRate)::numeric(10,2) as AvgMortalityRate
FROM disease_dw.FactOutbreaks fo
JOIN disease_dw.DimLocation dl ON fo.LocationKey = dl.LocationKey
JOIN disease_dw.DimDisease dd ON fo.DiseaseKey = dd.DiseaseKey
WHERE dl.Population > 0
GROUP BY dd.DiseaseName, dl.State, dl.Population
ORDER BY InfectionRate DESC;
```

Data Output Messages Notifications

	diseasename character varying (100)	state character varying (100)	population integer	numberofoutbreaks bigint	totalcases bigint	infectionrate numeric (10,2)	avgmortalityrate numeric (10,2)
1	Measles	Florida	450000	1	986	0.22	3.91
2	COVID-19	Florida	450000	1	356	0.08	4.46
3	Tuberculosis	California	4000000	3	2304	0.06	3.10
4	Influenza A	Illinois	2700000	2	1092	0.04	1.82
5	Malaria	Illinois	2700000	1	895	0.03	4.61
6	Malaria	California	4000000	2	815	0.02	2.55
7	Influenza A	California	4000000	1	919	0.02	0.17
8	Malaria	New York	8400000	1	306	0.00	2.53
9	COVID-19	New York	8400000	1	100	0.00	3.36

...

.....