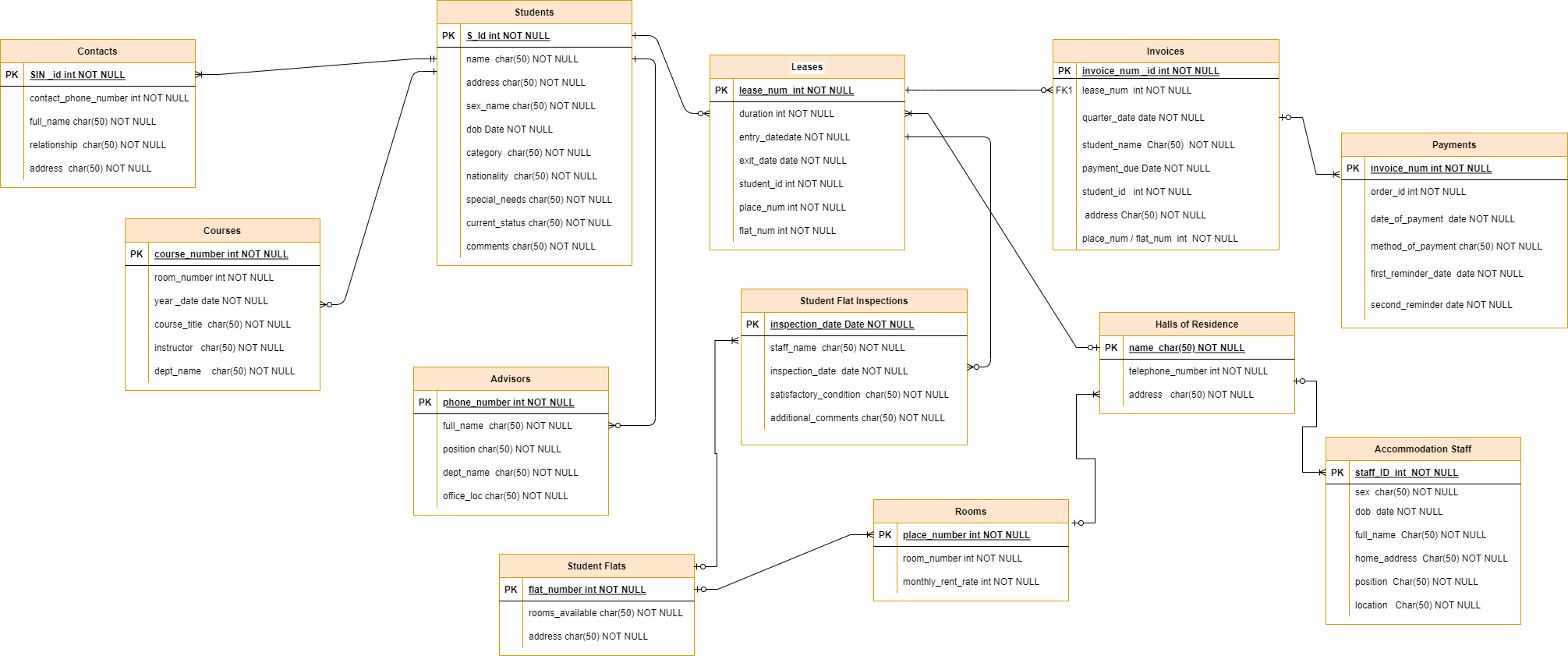
**DBFUND – Final Project: Designing and Implementing a Database**

**Tutor Were Vincent**

**DBFUND – Final Project: Designing and Implementing a Database**

1. **Diagram representing the tables and their relationships**
2. **Diagram**

****

1. **Relationships**
2. ***Students and Leases***

One-to-Many: A student can have multiple leases over time, but each lease belongs to one student.

1. ***Students and Advisors***

One-to-Many: Each student has one advisor, but an advisor may advise multiple students.

1. ***Students and Courses***

Many-to-Many: A student can be registered for multiple courses, and a course can have multiple students.

1. ***Students and Contacts***

One-to-Many: Each student can have multiple contacts, but each contact is associated with one student.

1. ***Leases and Invoices***

One-to-Many: Each lease can have multiple invoices, but each invoice is related to one lease.

1. ***Leases and Student Flat Inspections***

One-to-Many: Each lease can have multiple inspections, but each inspection is related to one lease.

1. ***Leases and Student Flats (or Halls of Residence)***

Many-to-One: Each lease is associated with one student flat or hall, but multiple leases can be associated with the same flat or hall.

1. ***Invoices and Payments***

One-to-Many: Each invoice can have multiple payments, but each payment is associated with one invoice.

1. ***Advisors and Students***

One-to-Many: Each advisor advises multiple students, but each student has only one advisor.

1. ***Halls of Residence and Rooms***

One-to-Many: Each hall has multiple rooms, but each room belongs to one hall.

1. ***Student Flats and Rooms***

One-to-Many: Each student flat has multiple rooms, but each room belongs to one student flat.

1. ***Student Flat Inspections and Student Flats***

One-to-Many: Each inspection is related to one student flat, but a student flat may have multiple inspections over time.

1. ***Accommodation Staff and Halls of Residence***

One-to-Many: Each staff member is associated with one hall, but each hall has multiple staff members.

1. ***Courses and Students***

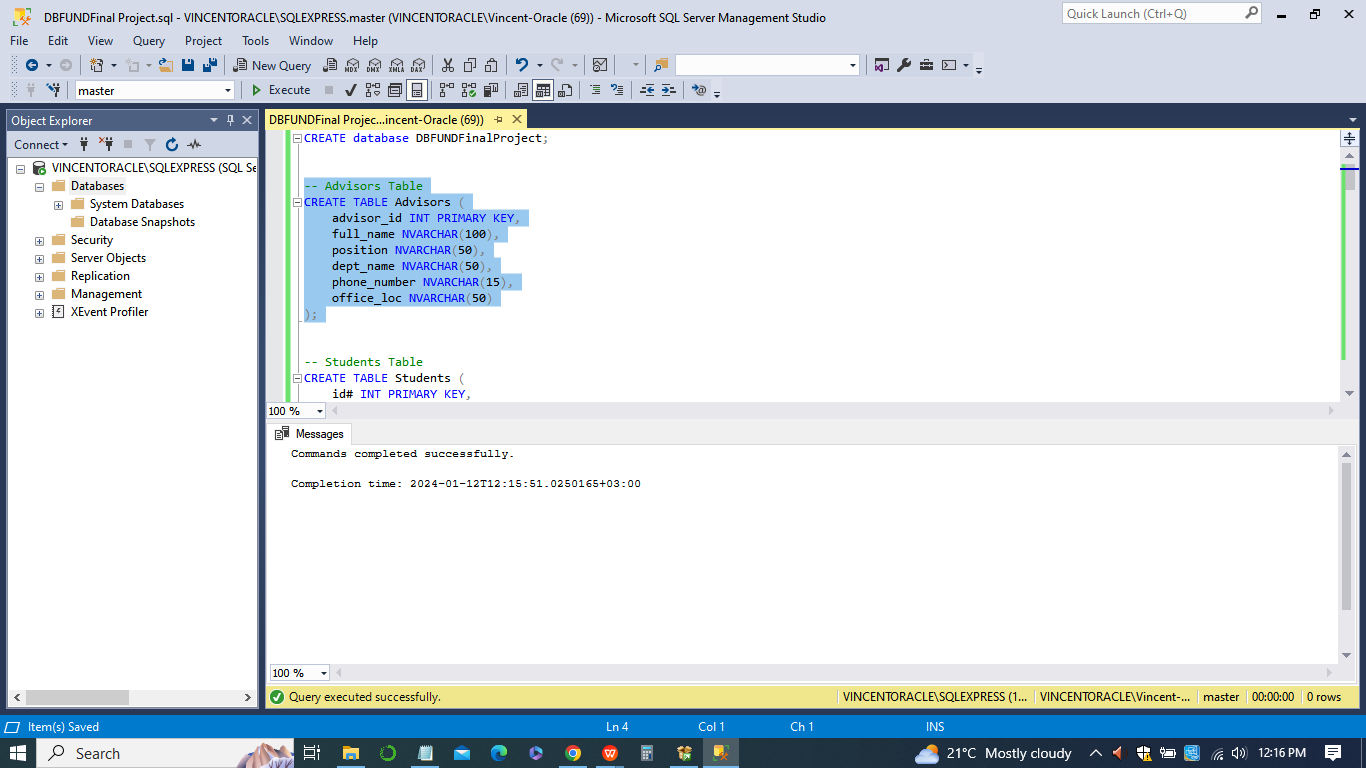
Many-to-Many: A course can have multiple students, and a student can be enrolled in multiple courses.

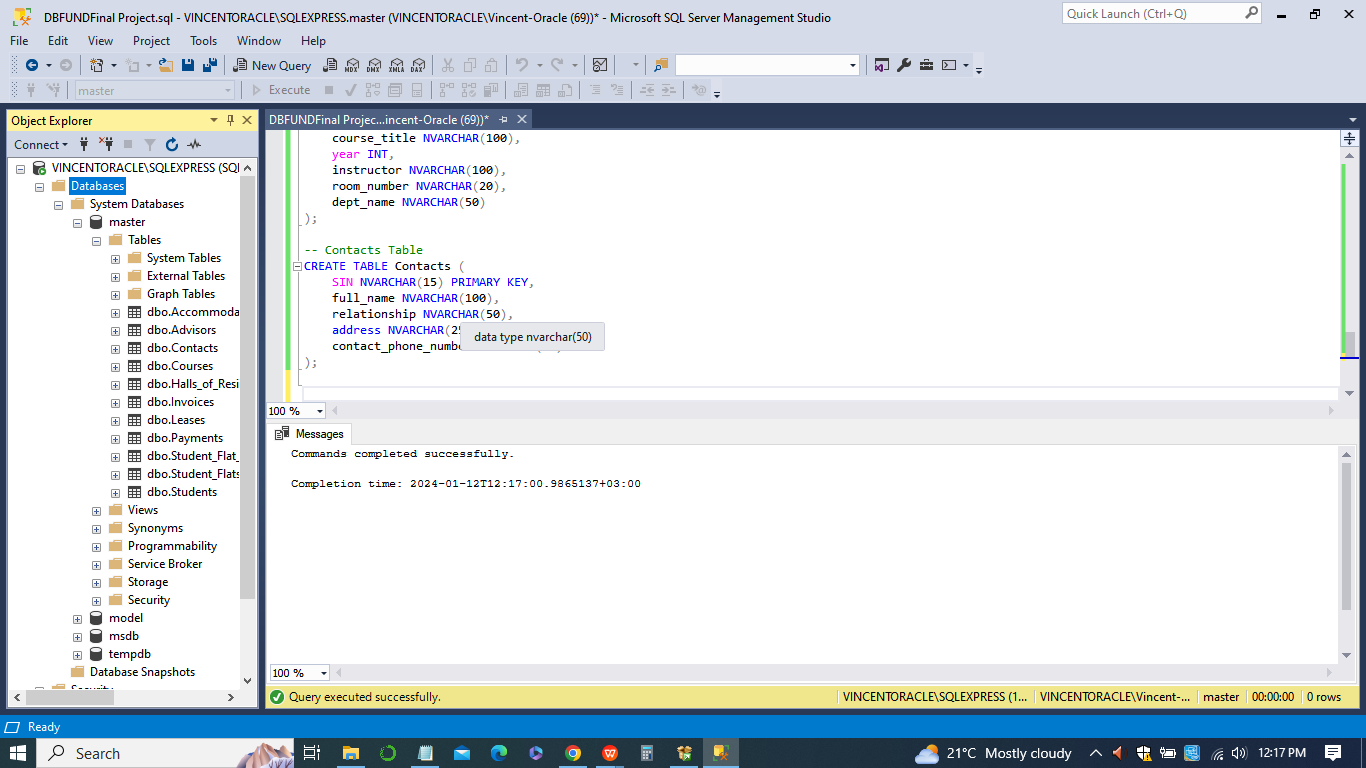
1. ***Contacts and Students***

One-to-Many: Each contact is associated with one student, but each student can have multiple contacts.

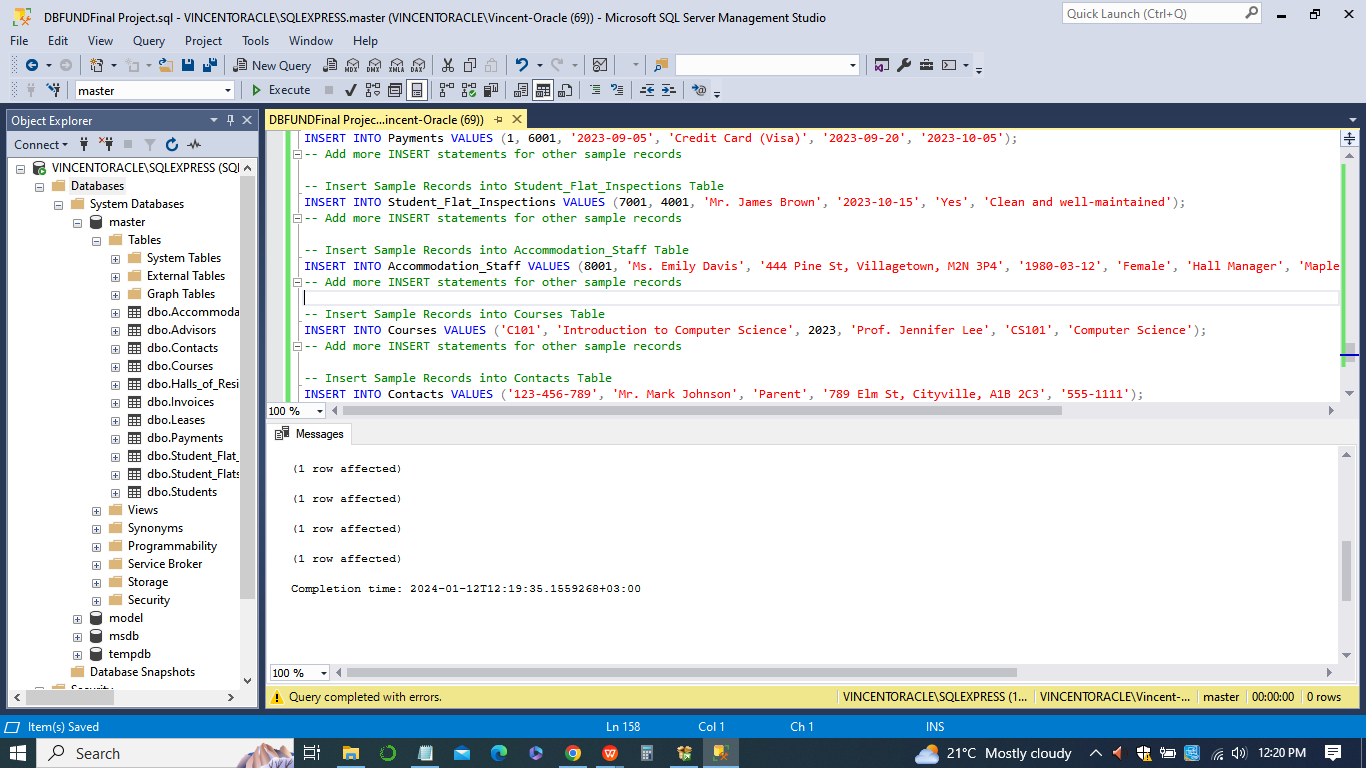
These relationships help define how the different entities in the database are connected and how data flows between them. It's essential to maintain referential integrity and ensure that the relationships reflect the real-world connections between the entities accurately.

1. **Tables in Microsoft SQL Server 2019 (GUI or commands)**



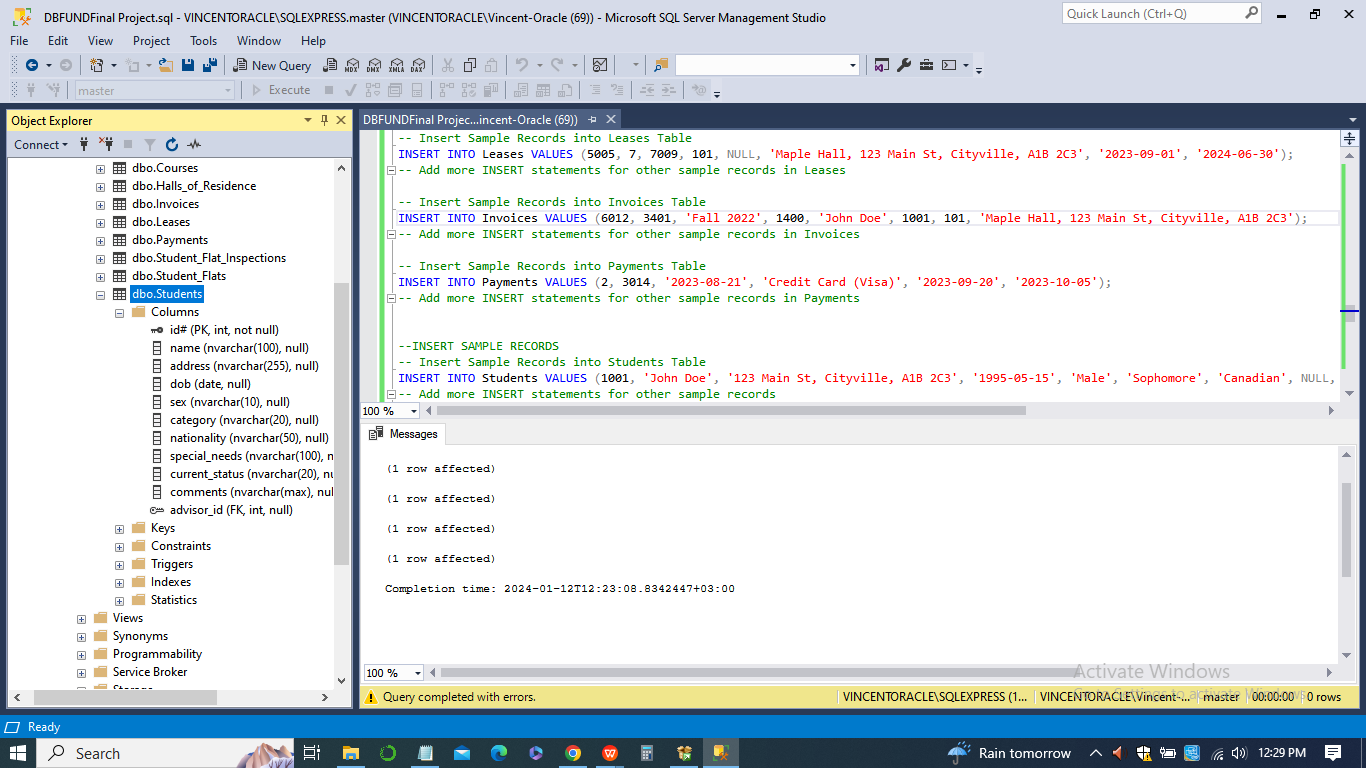


1. **Sample records within each table**



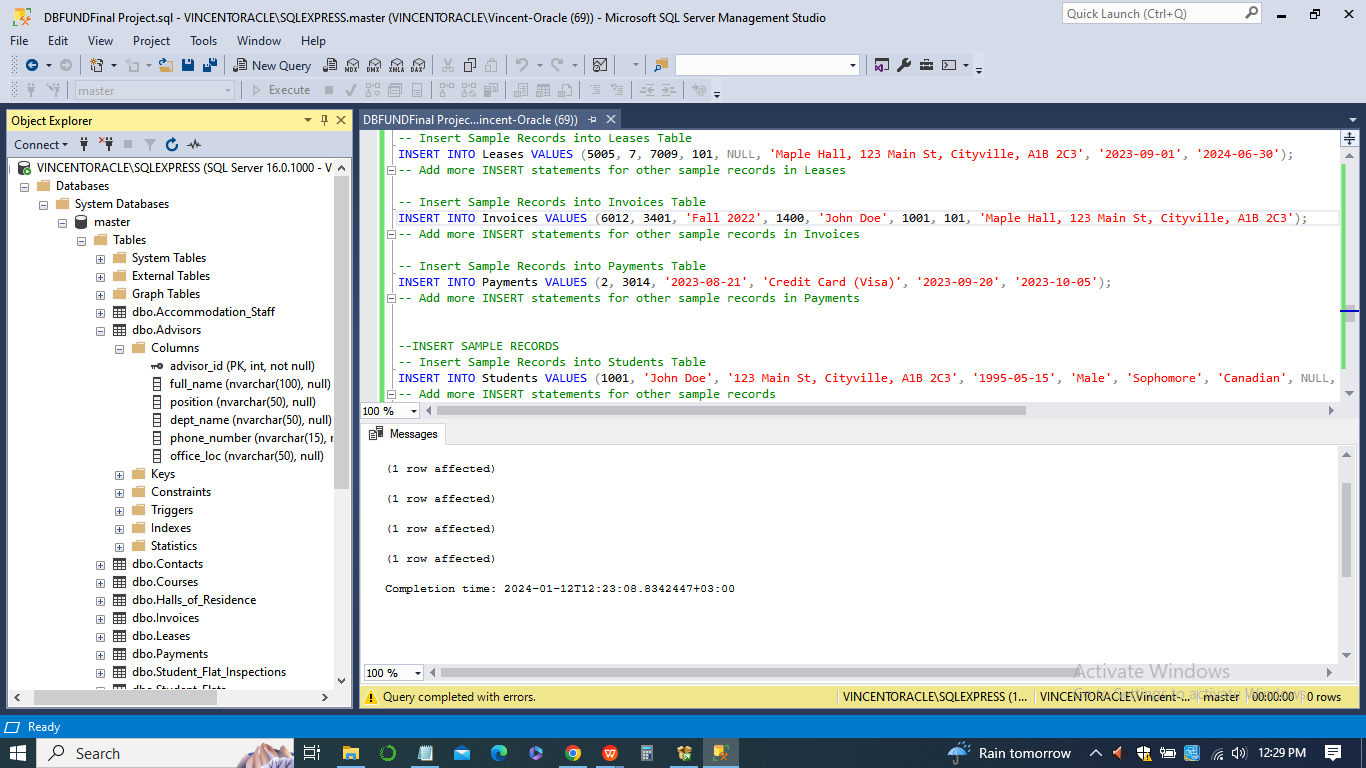
*-- Insert Sample Records into Students Table*

*INSERT INTO Students VALUES (1001, 'John Doe', '123 Main St, Cityville, A1B 2C3', '1995-05-15', 'Male', 'Sophomore', 'Canadian', NULL, 'Placed', 'Excellent student', 2001);*



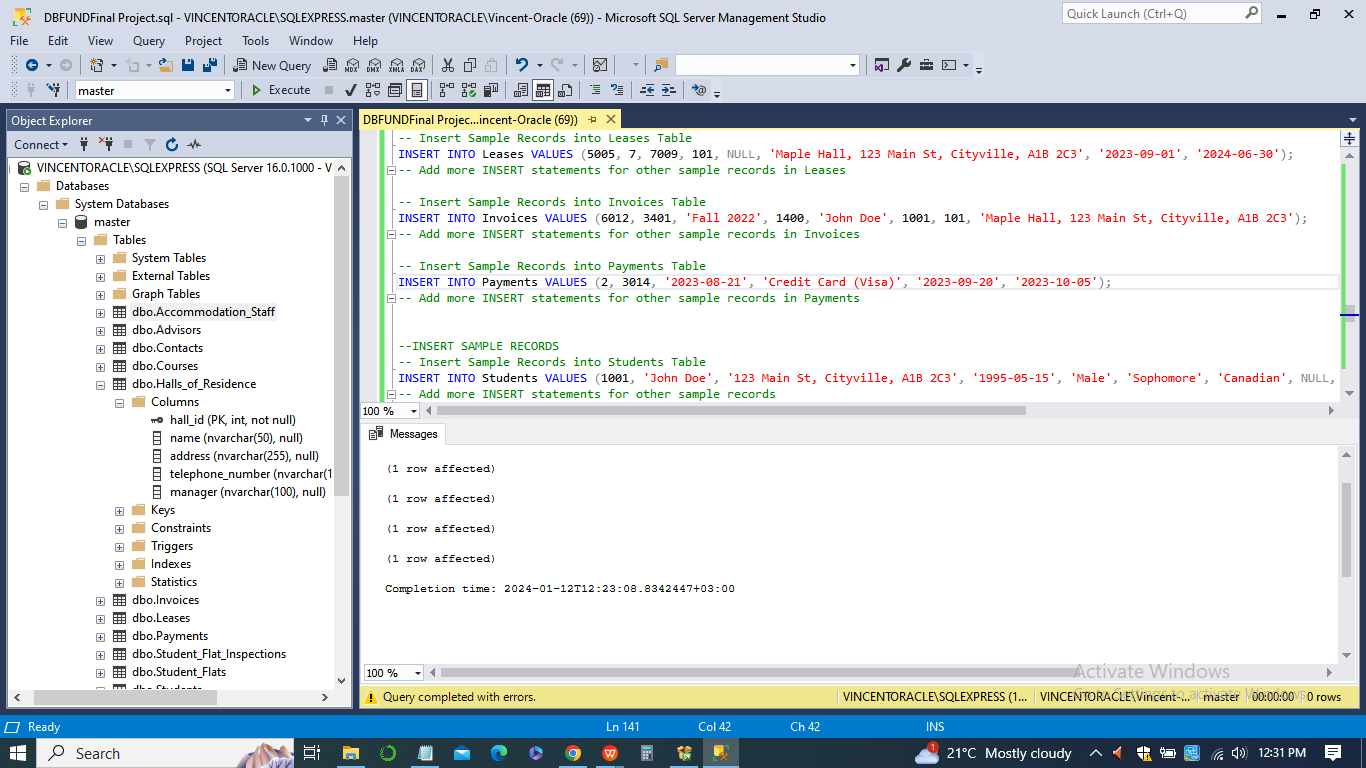
*-- Insert Sample Records into Advisors Table*

*INSERT INTO Advisors VALUES (2001, 'Dr. Alice Johnson', 'Academic Advisor', 'Computer Science', '555-1234', 'CS Building, Room 101');*



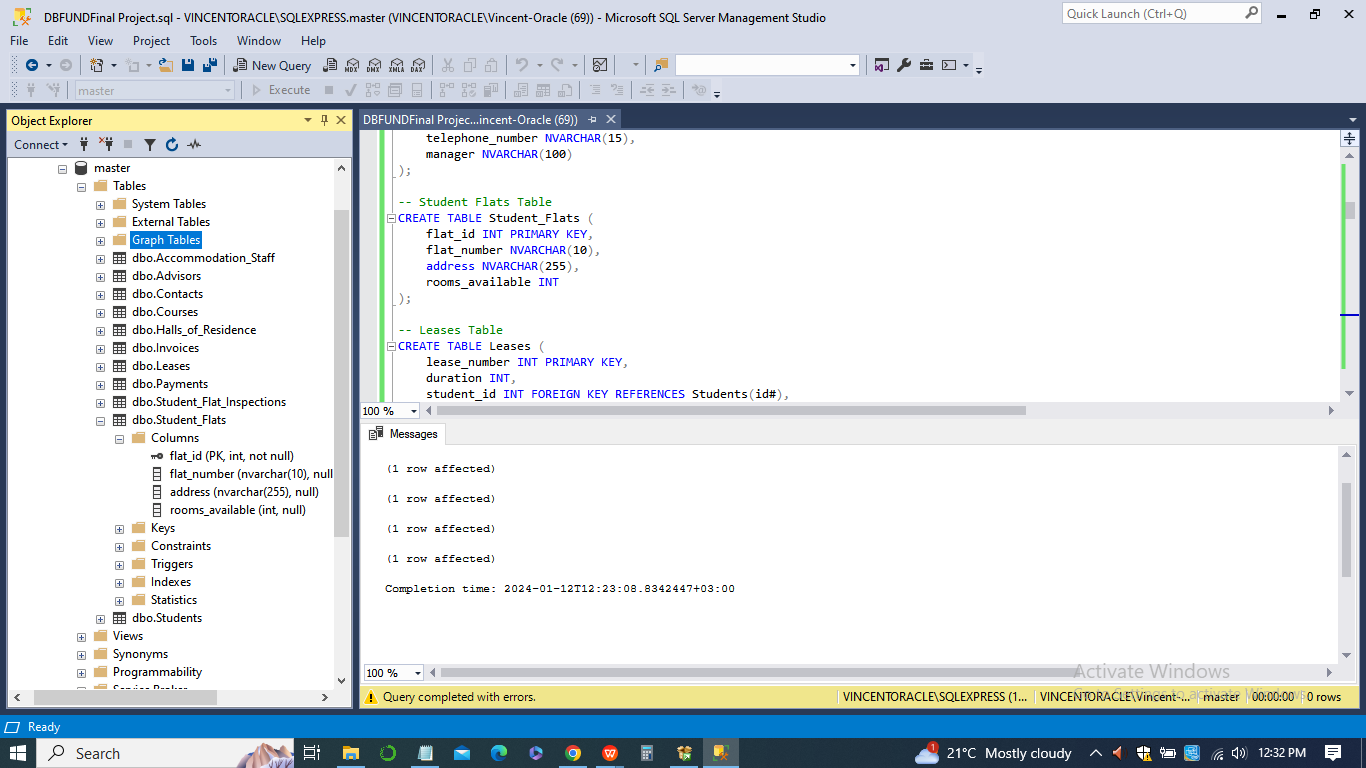
*-- Insert Sample Records into Halls\_of\_Residence Table*

*INSERT INTO Halls\_of\_Residence VALUES (3001, 'Maple Hall', '789 Pine St, Villagetown, M2N 3P4', '555-9876', 'Mr. James Brown');*



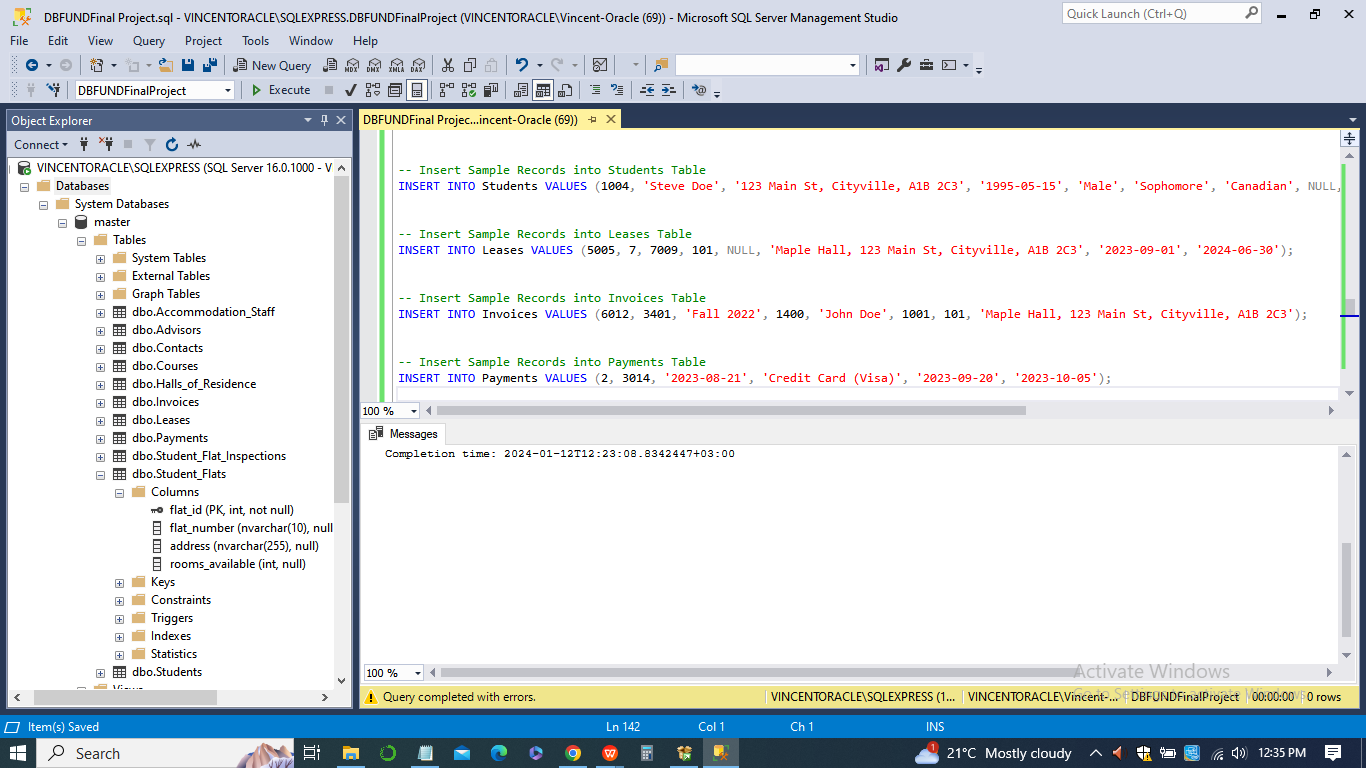
*-- Insert Sample Records into Student\_Flats Table*

*INSERT INTO Student\_Flats VALUES (4001, 'F101', '111 Elm St, Cityville, A1B 2C3', 3);*



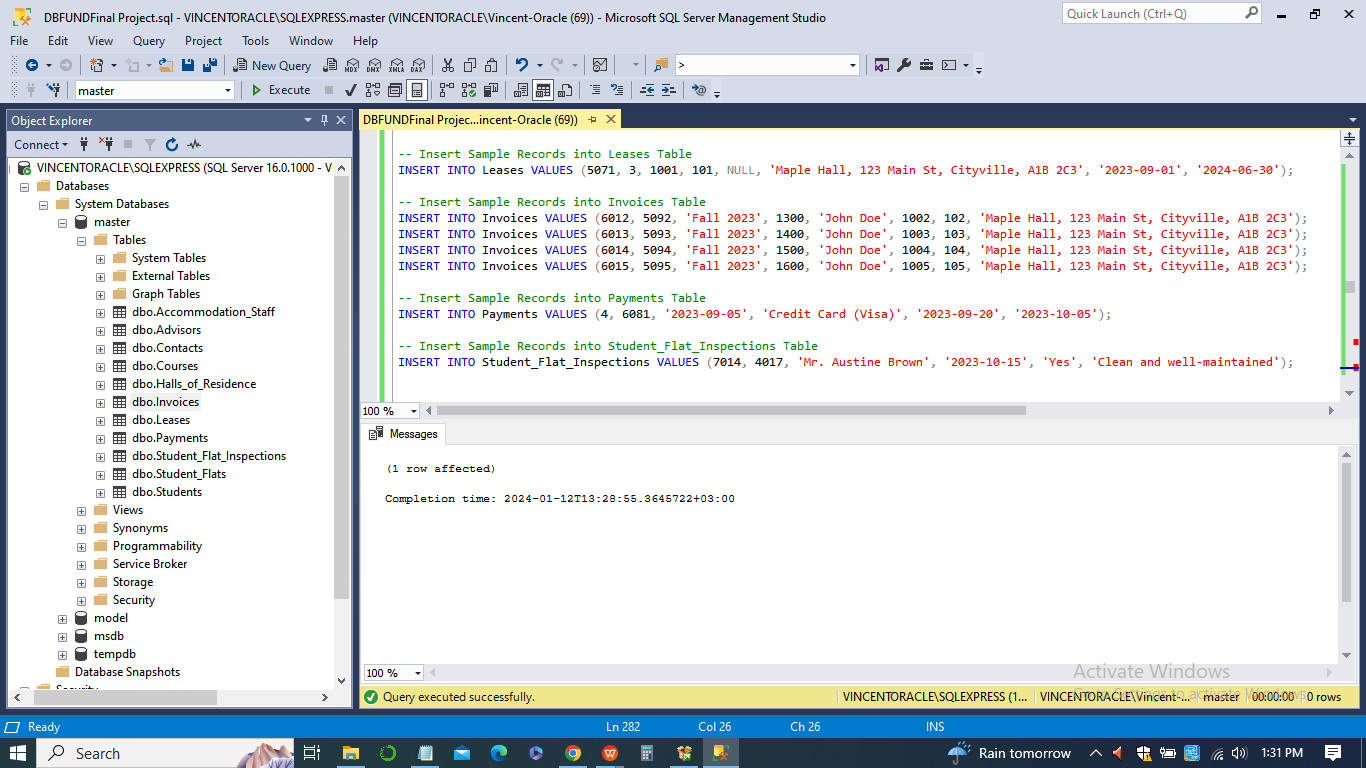
*-- Insert Sample Records into Leases Table*

*INSERT INTO Leases VALUES (5001, 2, 1001, 101, NULL, 'Maple Hall, 123 Main St, Cityville, A1B 2C3', '2023-09-01', '2024-06-30');*



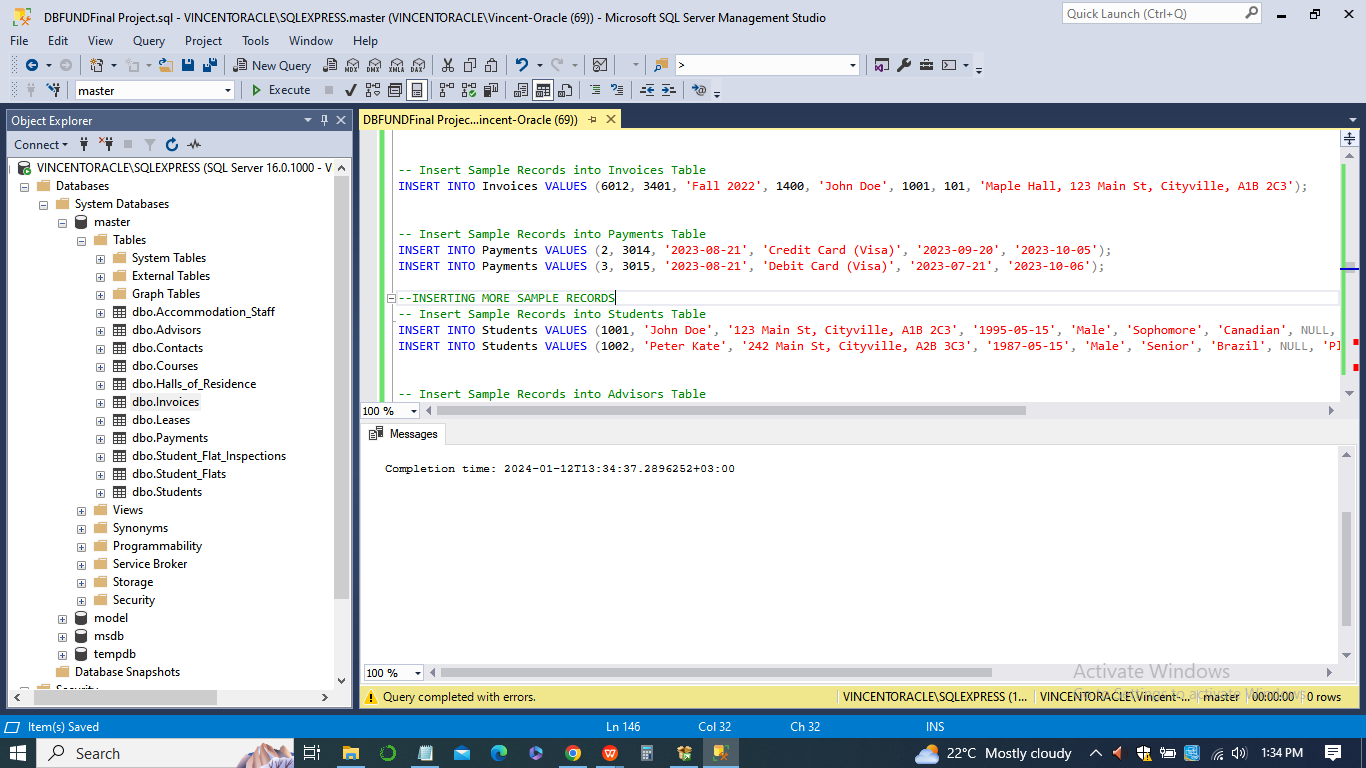
*-- Insert Sample Records into Invoices Table*

*INSERT INTO Invoices VALUES (6001, 5001, 'Fall 2023', 1200, 'John Doe', 1001, 101, 'Maple Hall, 123 Main St, Cityville, A1B 2C3');*



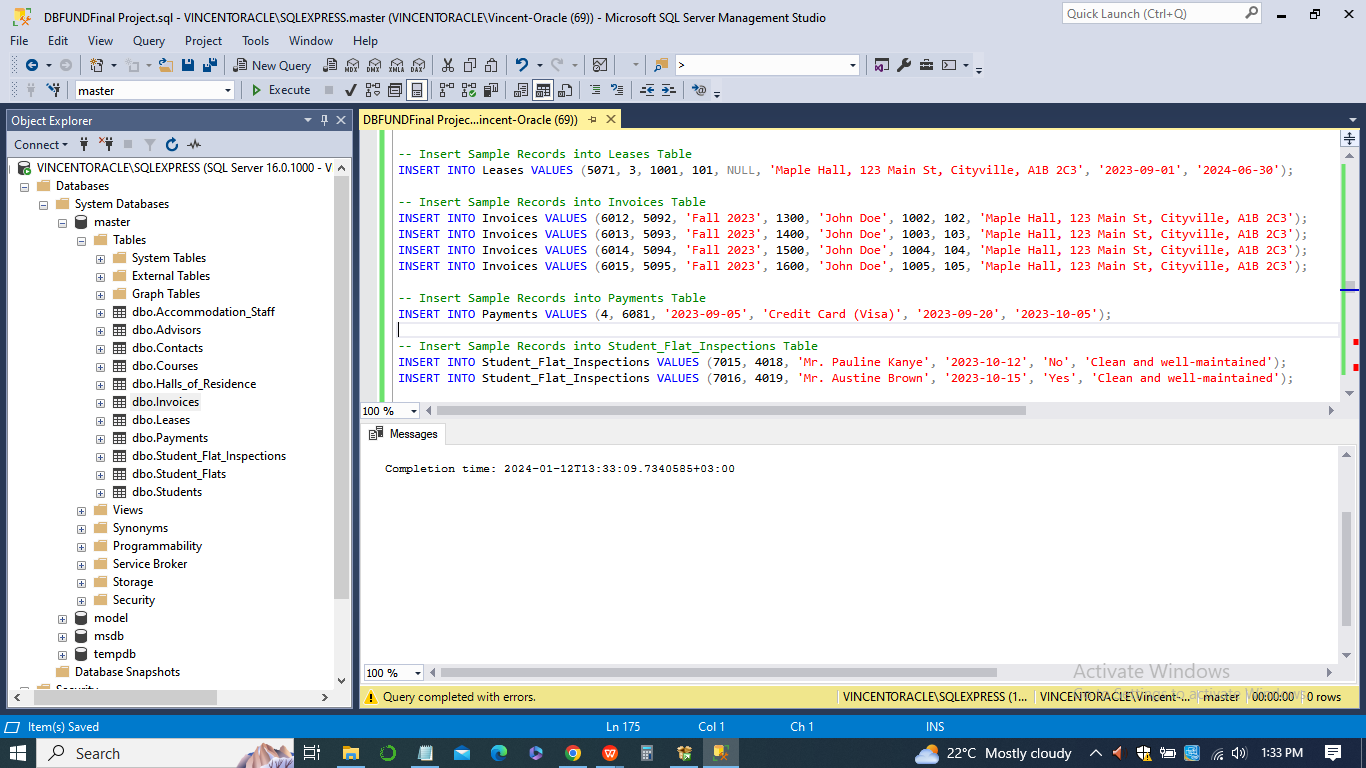
*-- Insert Sample Records into Payments Table*

*INSERT INTO Payments VALUES (1, 6001, '2023-09-05', 'Credit Card (Visa)', '2023-09-20', '2023-10-05');*



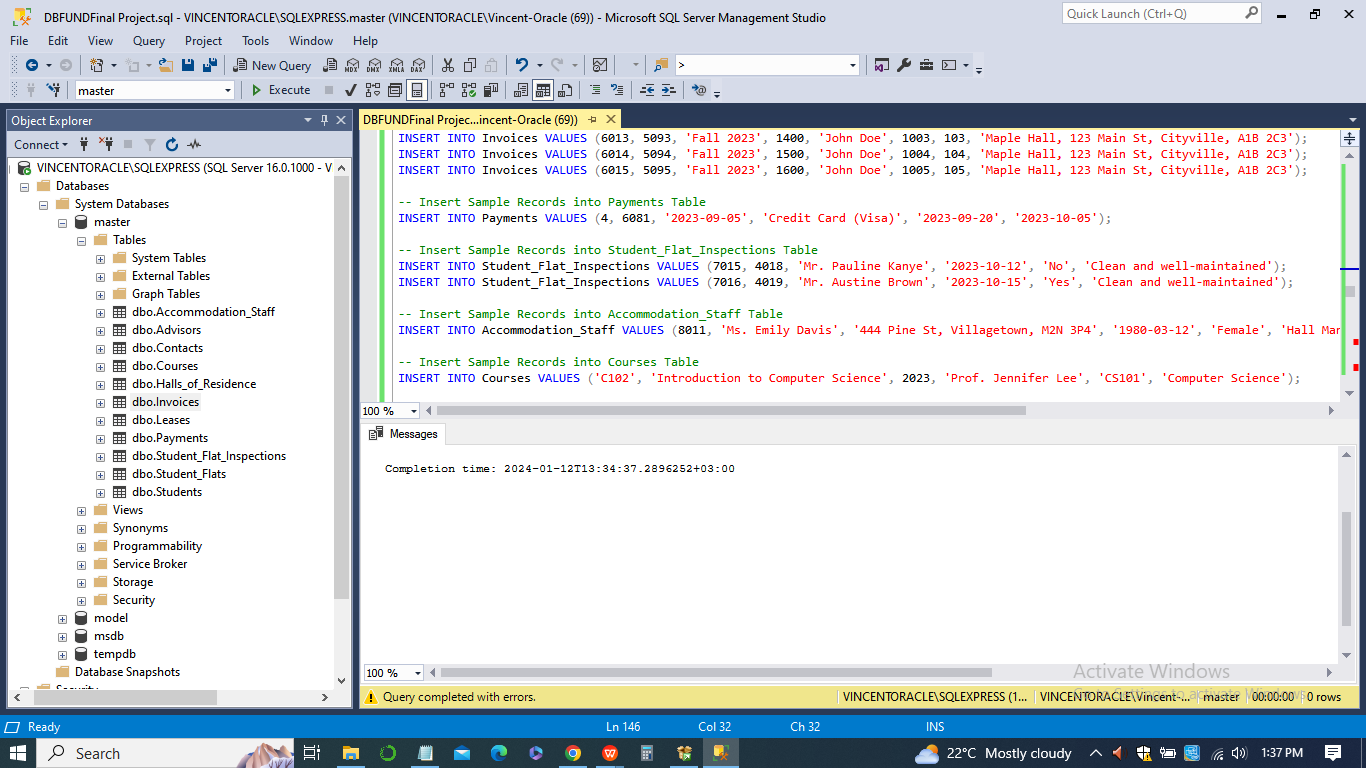
*-- Insert Sample Records into Student\_Flat\_Inspections Table*

*INSERT INTO Student\_Flat\_Inspections VALUES (7001, 4001, 'Mr. James Brown', '2023-10-15', 'Yes', 'Clean and well-maintained');*



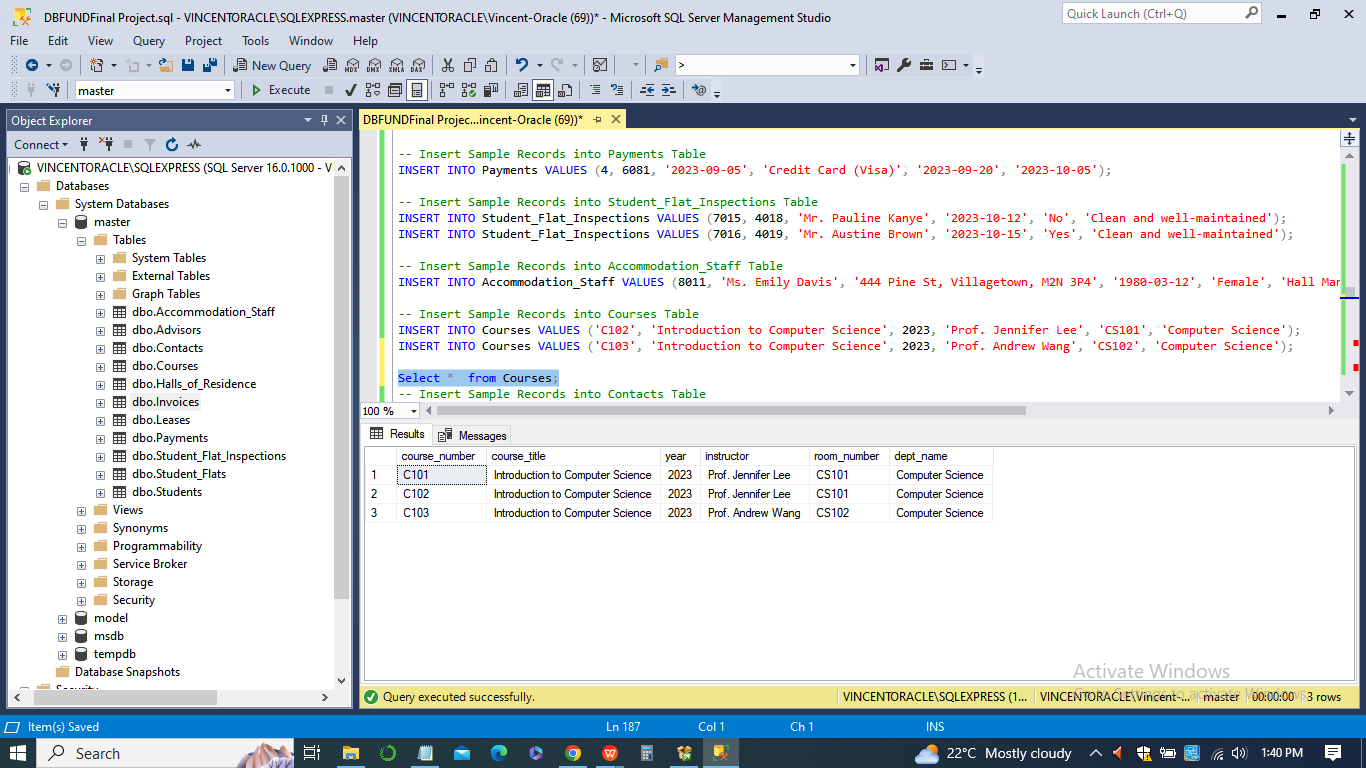
*-- Insert Sample Records into Accommodation\_Staff Table*

*INSERT INTO Accommodation\_Staff VALUES (8001, 'Ms. Emily Davis', '444 Pine St, Villagetown, M2N 3P4', '1980-03-12', 'Female', 'Hall Manager', 'Maple Hall');*



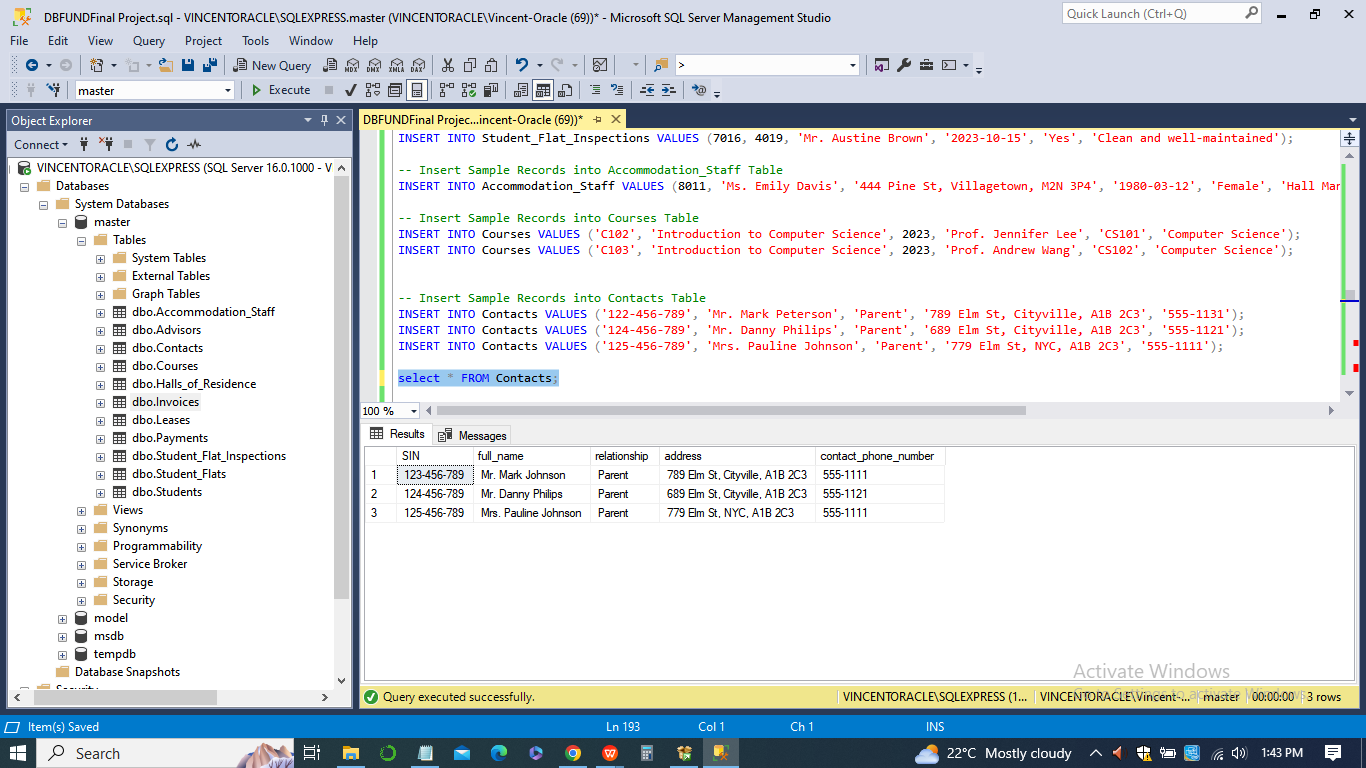
*-- Insert Sample Records into Courses Table*

*INSERT INTO Courses VALUES ('C101', 'Introduction to Computer Science', 2023, 'Prof. Jennifer Lee', 'CS101', 'Computer Science');*

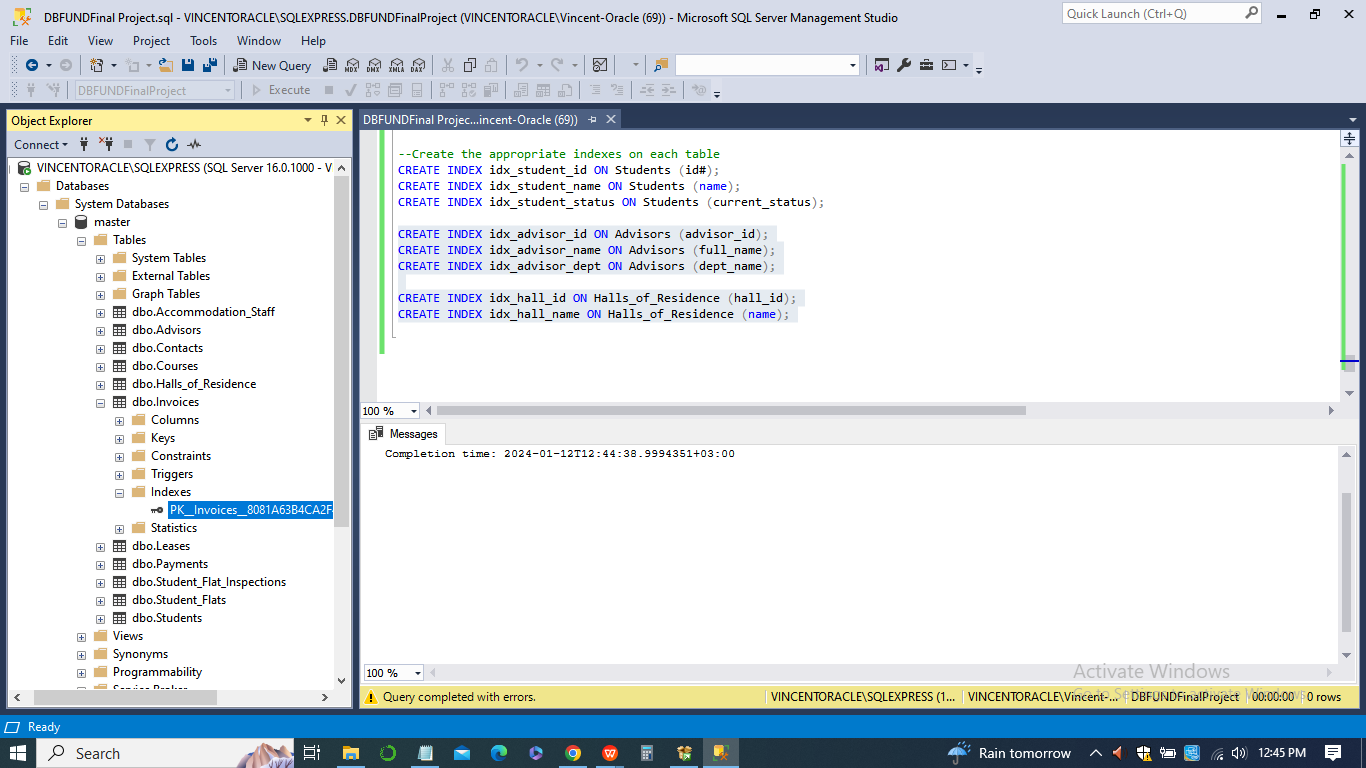


*-- Insert Sample Records into Contacts Table*

*INSERT INTO Contacts VALUES ('123-456-789', 'Mr. Mark Johnson', 'Parent', '789 Elm St, Cityville, A1B 2C3', '555-1111');*



1. **Create the appropriate indexes on each table**



1. ***Students Table***

*CREATE INDEX idx\_student\_id ON Students (id#);*

*CREATE INDEX idx\_student\_name ON Students (name);*

*CREATE INDEX idx\_student\_status ON Students (current\_status);*

1. ***Advisors Table***

*CREATE INDEX idx\_advisor\_id ON Advisors (advisor\_id);*

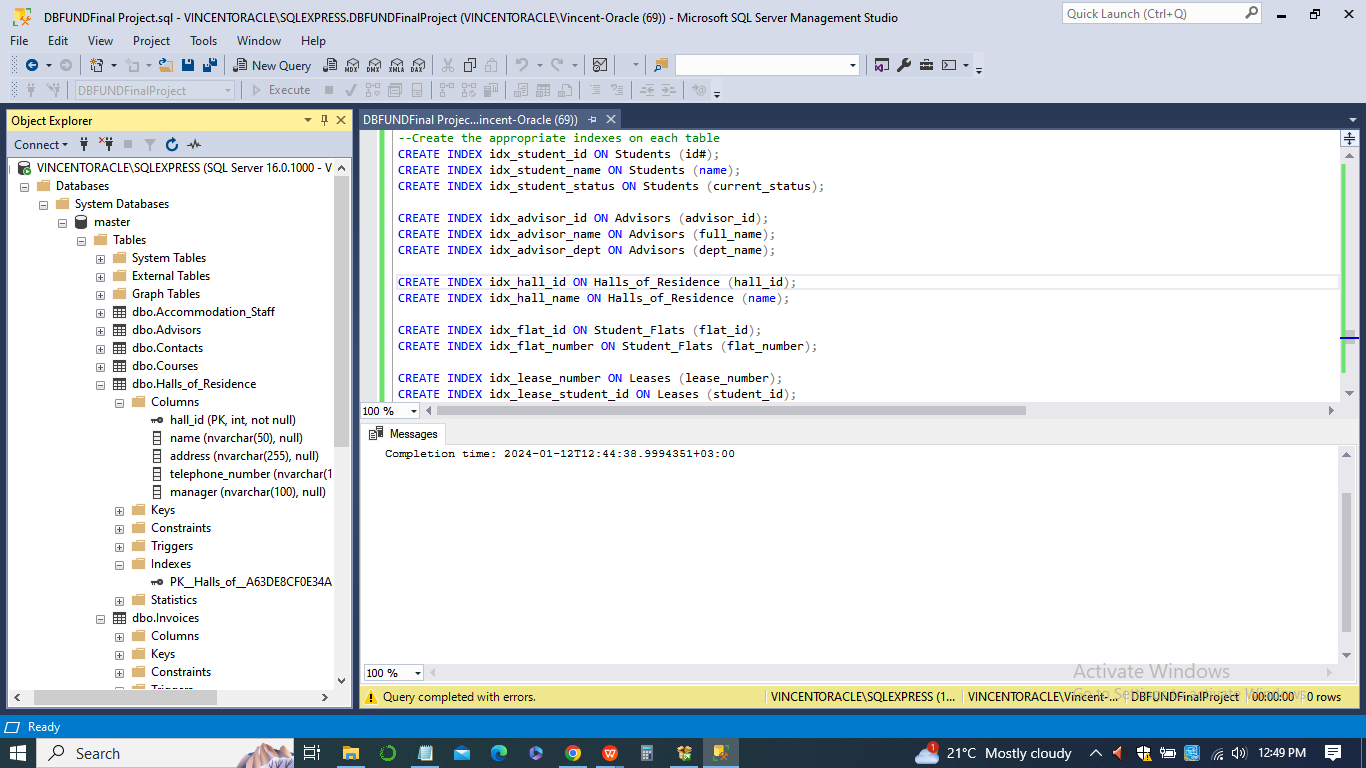
*CREATE INDEX idx\_advisor\_name ON Advisors (full\_name);*

*CREATE INDEX idx\_advisor\_dept ON Advisors (dept\_name);*

1. ***Halls of Residence Table***

*CREATE INDEX idx\_hall\_id ON Halls\_of\_Residence (hall\_id);*

*CREATE INDEX idx\_hall\_name ON Halls\_of\_Residence (name);*



1. ***Student Flats Table***

*CREATE INDEX idx\_flat\_id ON Student\_Flats (flat\_id);*

*CREATE INDEX idx\_flat\_number ON Student\_Flats (flat\_number);*

1. ***Leases Table***

*CREATE INDEX idx\_lease\_number ON Leases (lease\_number);*

*CREATE INDEX idx\_lease\_student\_id ON Leases (student\_id);*

1. ***Invoices Table***

*CREATE INDEX idx\_invoice\_number ON Invoices (invoice\_number);*

*CREATE INDEX idx\_invoice\_student\_id ON Invoices (student\_id);*

1. ***Payments Table***

*CREATE INDEX idx\_payment\_invoice\_number ON Payments (invoice\_number);*

1. ***Student Flat Inspections Table***

*CREATE INDEX idx\_inspection\_id ON Student\_Flat\_Inspections (inspection\_id);*

*CREATE INDEX idx\_inspection\_date ON Student\_Flat\_Inspections (inspection\_date);*

1. ***Accommodation Staff Table***

*CREATE INDEX idx\_staff\_id ON Accommodation\_Staff (staff\_id);*

*CREATE INDEX idx\_staff\_name ON Accommodation\_Staff (full\_name);*

1. ***Courses Table***

*CREATE INDEX idx\_course\_number ON Courses (course\_number);*

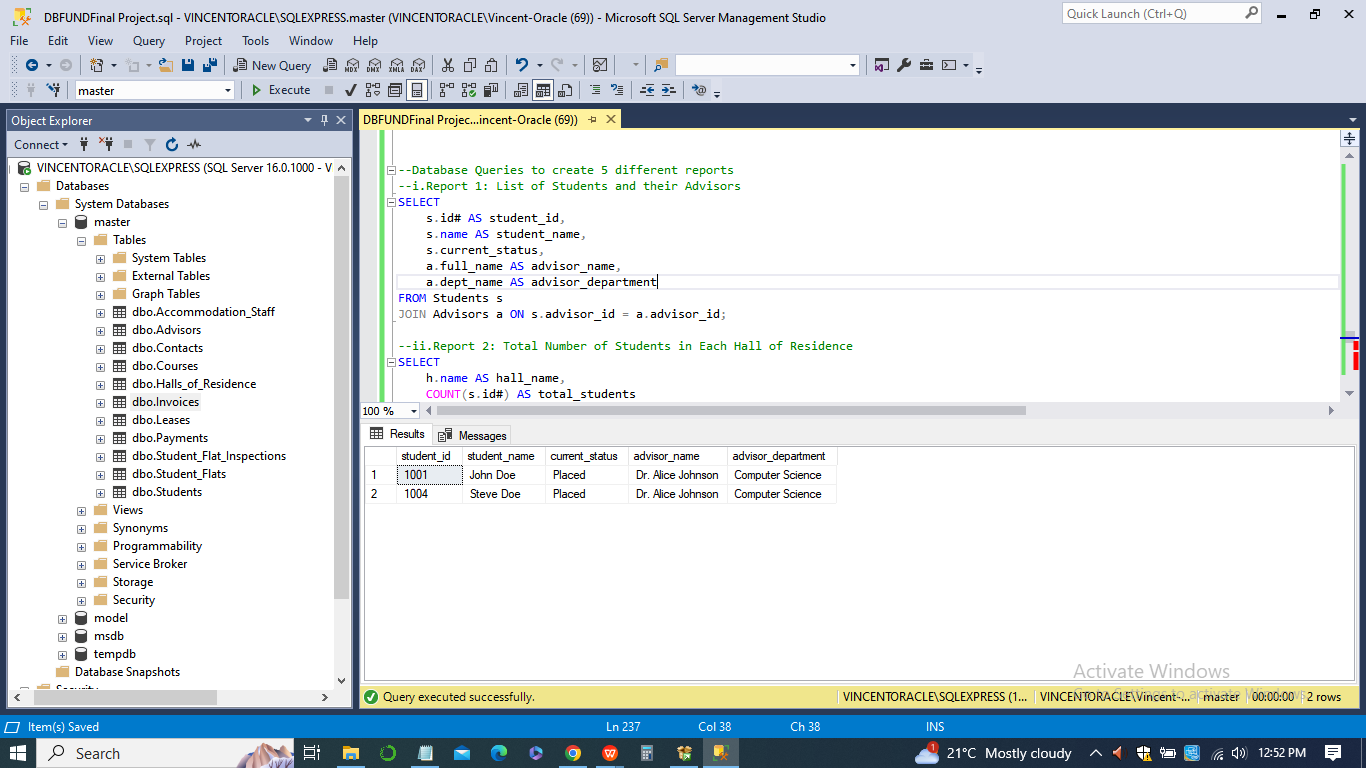
*CREATE INDEX idx\_course\_title ON Courses (course\_title);*

1. ***Contacts Table***

*CREATE INDEX idx\_contact\_sin ON Contacts (SIN);*

*CREATE INDEX idx\_contact\_name ON Contacts (full\_name);*

1. **Database Queries to create 5 different reports**
2. ***Report 1: List of Students and their Advisors***



*SELECT*

*s.id# AS student\_id,*

*s.name AS student\_name,*

*s.current\_status,*

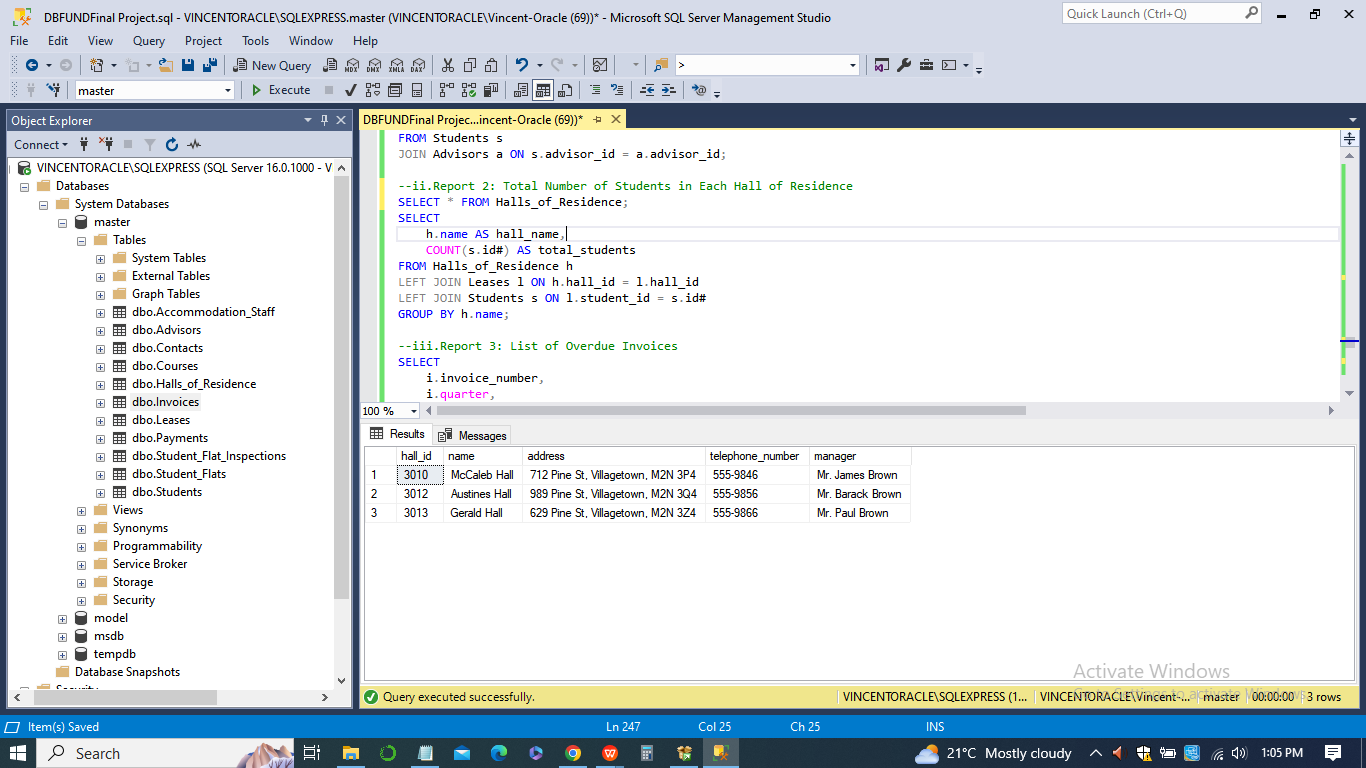
*a.full\_name AS advisor\_name,*

*a.dept\_name AS advisor\_department*

*FROM Students s*

*JOIN Advisors a ON s.advisor\_id = a.advisor\_id;*

1. ***Report 2: Total Number of Students in Each Hall of Residence***



*SELECT*

*h.name AS hall\_name,*

*COUNT(s.id#) AS total\_students*

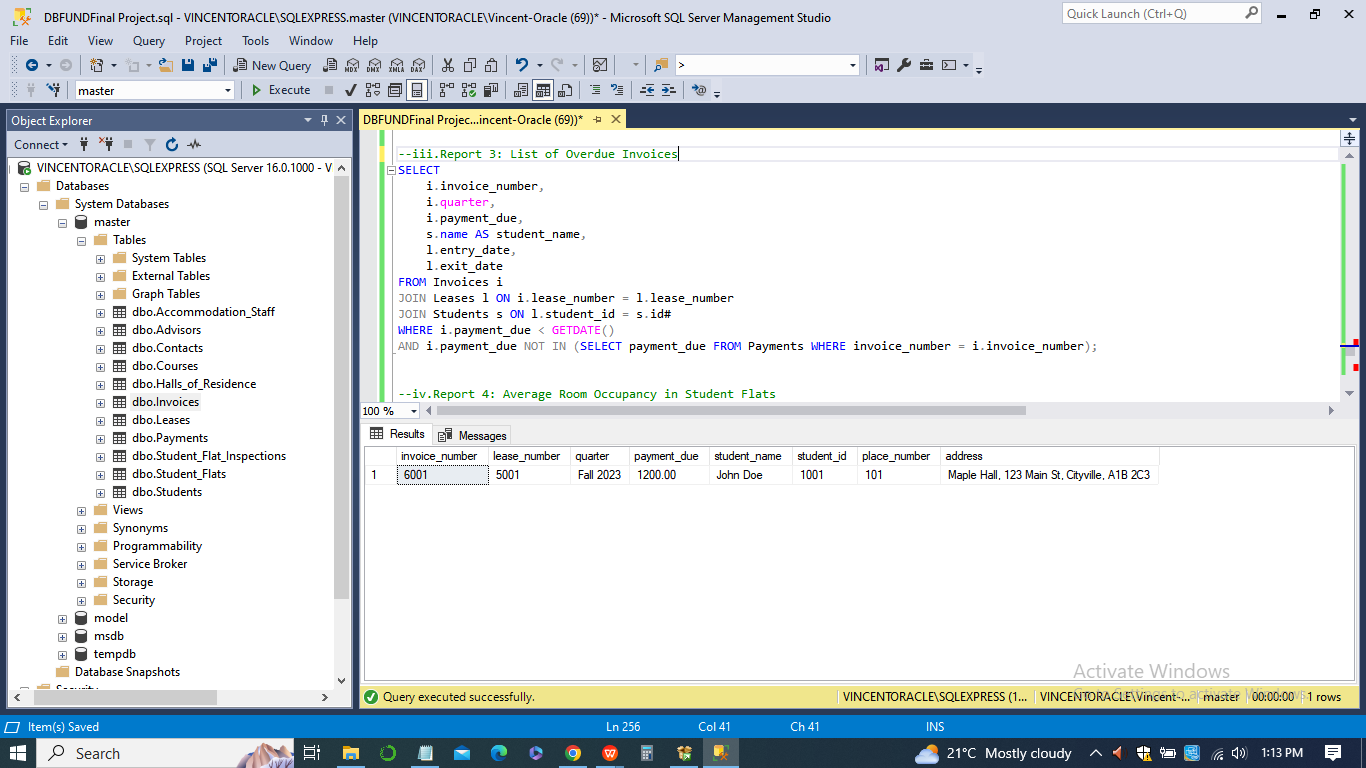
*FROM Halls\_of\_Residence h*

*LEFT JOIN Leases l ON h.hall\_id = l.hall\_id*

*LEFT JOIN Students s ON l.student\_id = s.id#*

*GROUP BY h.name;*

1. ***Report 3: List of Overdue Invoices***



*SELECT*

*i.invoice\_number,*

*i.quarter,*

*i.payment\_due,*

*s.name AS student\_name,*

*l.entry\_date,*

*l.exit\_date*

*FROM Invoices i*

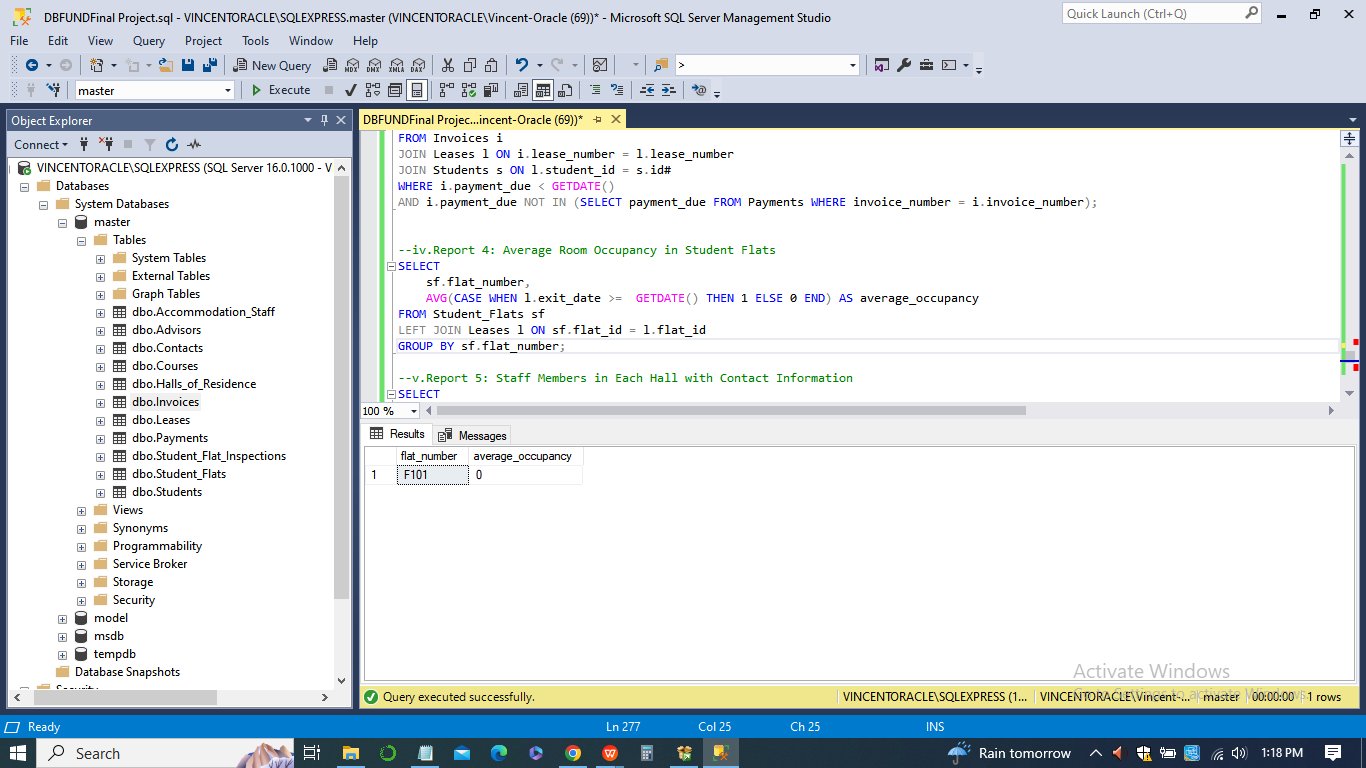
*JOIN Leases l ON i.lease\_number = l.lease\_number*

*JOIN Students s ON l.student\_id = s.id#*

*WHERE i.payment\_due < GETDATE()*

*AND i.payment\_due NOT IN (SELECT payment\_due FROM Payments WHERE invoice\_number = i.invoice\_number);*

1. ***Report 4: Average Room Occupancy in Student Flats***



*SELECT*

*sf.flat\_number,*

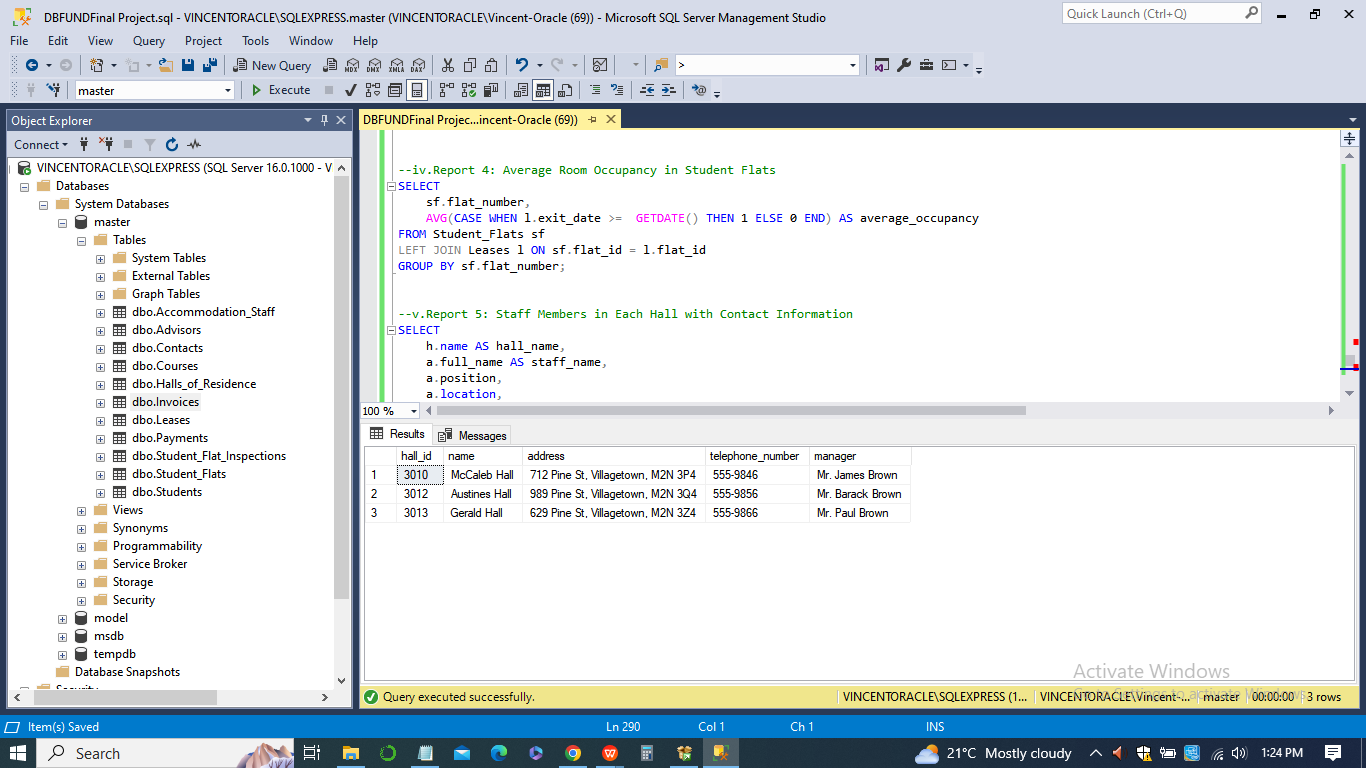
*AVG(CASE WHEN l.exit\_date >= GETDATE() THEN 1 ELSE 0 END) AS average\_occupancy*

*FROM Student\_Flats sf*

*LEFT JOIN Leases l ON sf.flat\_id = l.flat\_id*

*GROUP BY sf.flat\_number;*

1. ***Report 5: Staff Members in Each Hall with Contact Information***



*SELECT*

*h.name AS hall\_name,*

*a.full\_name AS staff\_name,*

*a.position,*

*a.location,*

*a.phone\_number*

*FROM Halls\_of\_Residence h*

*JOIN Accommodation\_Staff a ON h.hall\_id = a.hall\_id;*