

CSE 581 SPRING 2020 Intro to Database Management Systems

Project – 2

Design and Implementation of Recruitment Database

Abstract:

In this project we design and implement Recruitment Database for ‘xyz’ company. I’ve written all the tables which are required during the design and then I’ve written its associated columns. I’ve designed the database using ER diagrams and then we normalize it so that to improve efficiency and reducing redundancy. After designing the database, we create tables and constraints. I’ve added triggers according to the problem statement. I’ve populated data and inserted into the tables. I’ve used database concepts like Stored Procedures, Server Roles, Security, Views, Functions. After Implementing the database, we run few test scripts which shows that database is correct and useful. Recruitment is important part of the company and with this project we can maintain the recruitment hiring of any company.

VENKATA SAI PAWAN KOMARAVOLU

SUID : 425210282

Table of Contents

1) Database Name.....	3
2) Design Phase of Project	3
2.1 Introduction.....	3
2.2 Tables Used.....	4
2.3 Logic.....	6
2.4 E/R Diagram.....	7
2.5 Potential Integrity and Security Issues.....	9
3) Implementation Phase.....	10
3.1 Database Creation.....	10
3.2 Constraints.....	10
3.3 Tables Creation.....	14
3.4 Triggers.....	16
4) Testing Phase.....	20
4.1 Populating Database.....	20
4.2 Views.....	29
4.3 Stored Procedure.....	36
4.4 Functions.....	40
4.5 Scripts.....	44
4.6 Transaction.....	46
5) Conclusion.....	48

1. DATABASE NAME

I've named my database as XYZRecruitment Database

2. DESIGN PHASE OF PROJECT

2.1 INTRODUCTION

In today's world, almost most of the companies have HR department which develops and administers programs which increase an organization effectiveness. The HR department focuses on Recruitment, Compensation and benefits, Training and Learning, Labor and Employee Relations and Organization Development. In this project we design, implement and test Recruitment branch of a company 'xyz' . The recruitment Database keeps track of steps beginning from the job opening to Onboarding process of the selected candidates. In between there are multiple steps where we need to keep track of. Our first step is to design a database for that we have taken tables and their columns, and we use ER diagrams to design our database.

We design the database in way that we can retrieve records of all the candidates who have applied and who's interview process is on going or those which are completed. We keep track of candidate status in the who process of the Interview.

The Interview is either online or onsite. Depending on the location of the interview we maintain tables for Hotel reservation, Airline reservation and Car rentals and we maintain amount for each candidate so that it can be reimbursed by the candidate. We also keep track of the candidate applying to specific job role and job type.

We maintain an onboarding table where a selected candidate training is done and all his/her documents are verified and with this our database management is done as a database admin.

2.2 TABLES USED

The following are the tables used in my project. I've added appropriate columns to each table so that we can meet all the conditions which were asked in the problem statement.

1) Address Table :

- AddressID
- AddressLine1
- AddressLine2
- City
- State
- ZipCode

2) ContactDetails Table

- ContactID
- EmailAddress
- PhoneNumber

3) Department Table

- DepartmentID
- DepartmentName
- DepartmentDescription

4) CandidateQualification Table

- CandidateQualificationID
- GradGpa
- WorkExperienceYears

5) Interviewer Table

- InterviewerID
- InterviewerName
- InterviewerStatus

6) For InterviewLocation Table

- BuildingName
- InterviewLocationID
- City
- State
- ZipCode

7) JobOpening Table

- JobOpeningID
- JobRole

- JobDescription
 - DepartmentID
 - JobType
 - NoOfJobOpenings
- 8) CandidateDetails Table
- CandidateDetailsID
 - CandidateName
 - AddressID
 - ContactID
 - CandidateQualificationID
 - CandidateQualification(CandidateQualificationID)
- 9) JobApplication Table
- CandidateID
 - JobOpeningID
 - CandidateDetailsID
 - CandidateStatus
- 10) Interview Table
- InterviewerID
 - InterviewID
 - CandidateID
 - InterviewResult
 - InterviewType
- 11) ComplaintHandling Table
- CandidateID
 - ComplaintID
 - InterviewerID
 - ComplaintDescription
- 12) AirlineReservation Table
- TicketID
 - InterviewID
 - CandidateID
 - InterviewLocationID
 - FlightDetails
 - AirlineFare money
- 13) Onboarding Table
- OnboardingID
 - CandidateID

- JobOpeningID
- DepartmentID
- CandidateDocuments

14) HotelReservation Table

- ReservationID
- InterviewID
- CandidateID
- InterviewLocationID
- HotelDetails
- HotelFare money

15) CarRental Table

- RentalID
- InterviewID
- CandidateID
- InterviewLocationID
- CarDetails
- RentalFare money

16) For Reimbursement Table

- ReimbursementID
- InterviewID
- CandidateID
- RentalID
- TicketID
- ReservationID
- ReimbursementAmount

17) JobOfferNegotiation Table

- NegotiationID
- CandidateID
- NegotitationStatus

2.3 LOGIC

First Starting from JobApplication table with CandidateID as the primary key. The Candidate who will apply he needs to fill his details and this will be stored in CandidateDetails table. In JobApplication table we will have CandidateStatus column where the status of the interview is stored and updated when ever there is change in the interview status for that particular Candidate

In CandidateDetails table we have CandidateDetailsID as the primary key which stores the CandidateDetails

A candidate apply to a particular Jobrole and Jobtype for that I have created JobOpening table with JobOpeningID as the primary key. In this table we keep track of JobRole , DepartmentID and number of JobOpening. When a candidate gets selected and the opening aren't there then his status is done as On call or next Interview.

After Applying the Interview can be Online and Onsite and we update these in Interview table with InterviewID as primary key.

Depending on the InterviewType and InterviewLocation if its onsite then company pays for the Candidate. Company allocates AirlineFare , HotelFare and CarRental Fare which are three tables in the database with TicketID , ReservationID and RentalID as the primary key which keeps track of travel expenses , Living expenses of the candidate. We have InterviewLocationID as the primary key for the InterviewLocation.

These fare are later reimbursed and thus for this we have reimbursement table with ReimbursementID as the primary key.

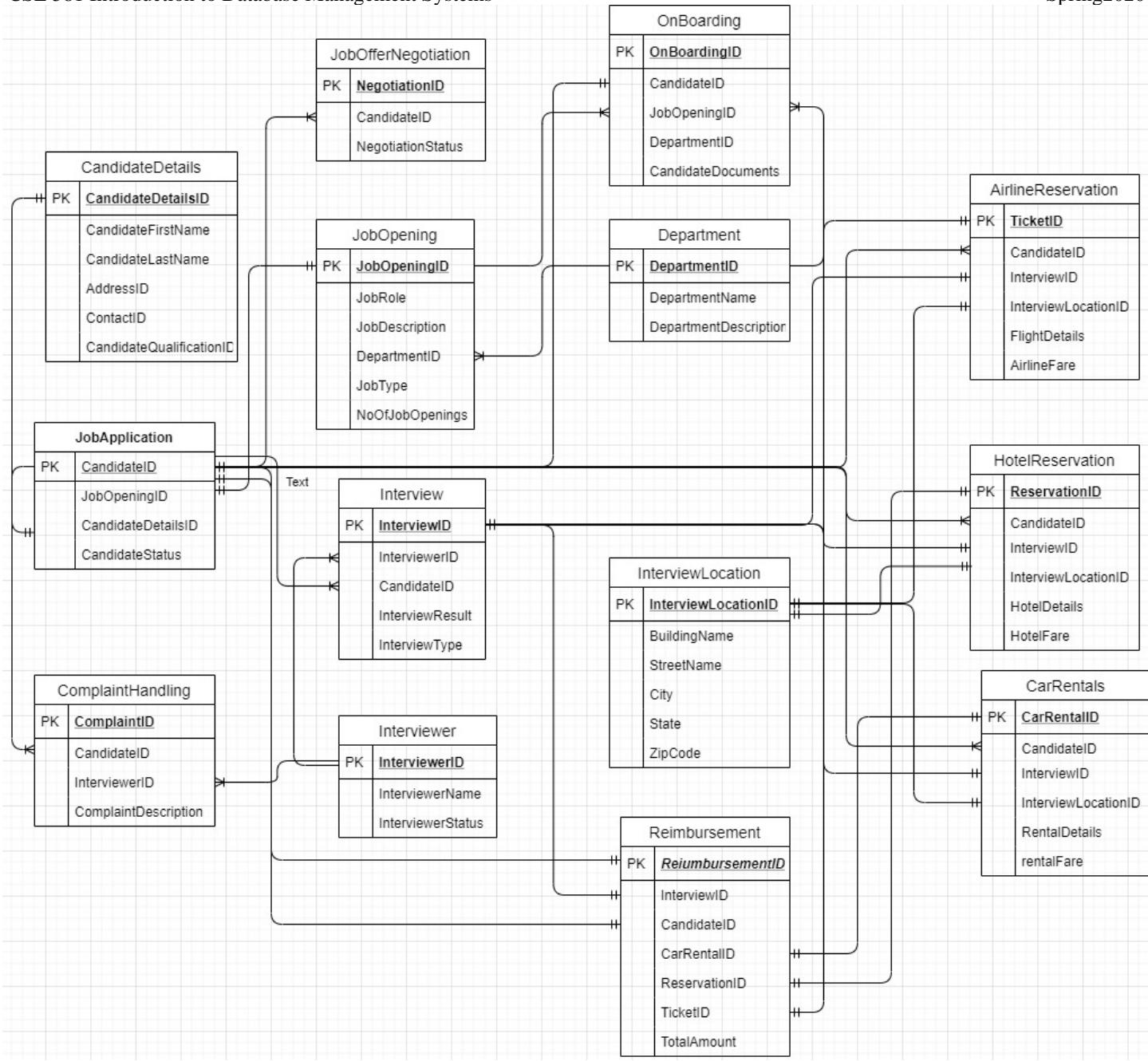
If a candidate gets selected then offer is made to him but the candidate can negotiate with the company and therefore we create OfferNegotiation table with OfferNegotiationID as the primary key.

After accepting the offer the candidate has to do Onboarding process for this we have a table with OnboardingID as the primary key. After the Onboarding process is completed the status of the Candidate is updated and with th

2.4 E/R DIAGRAM

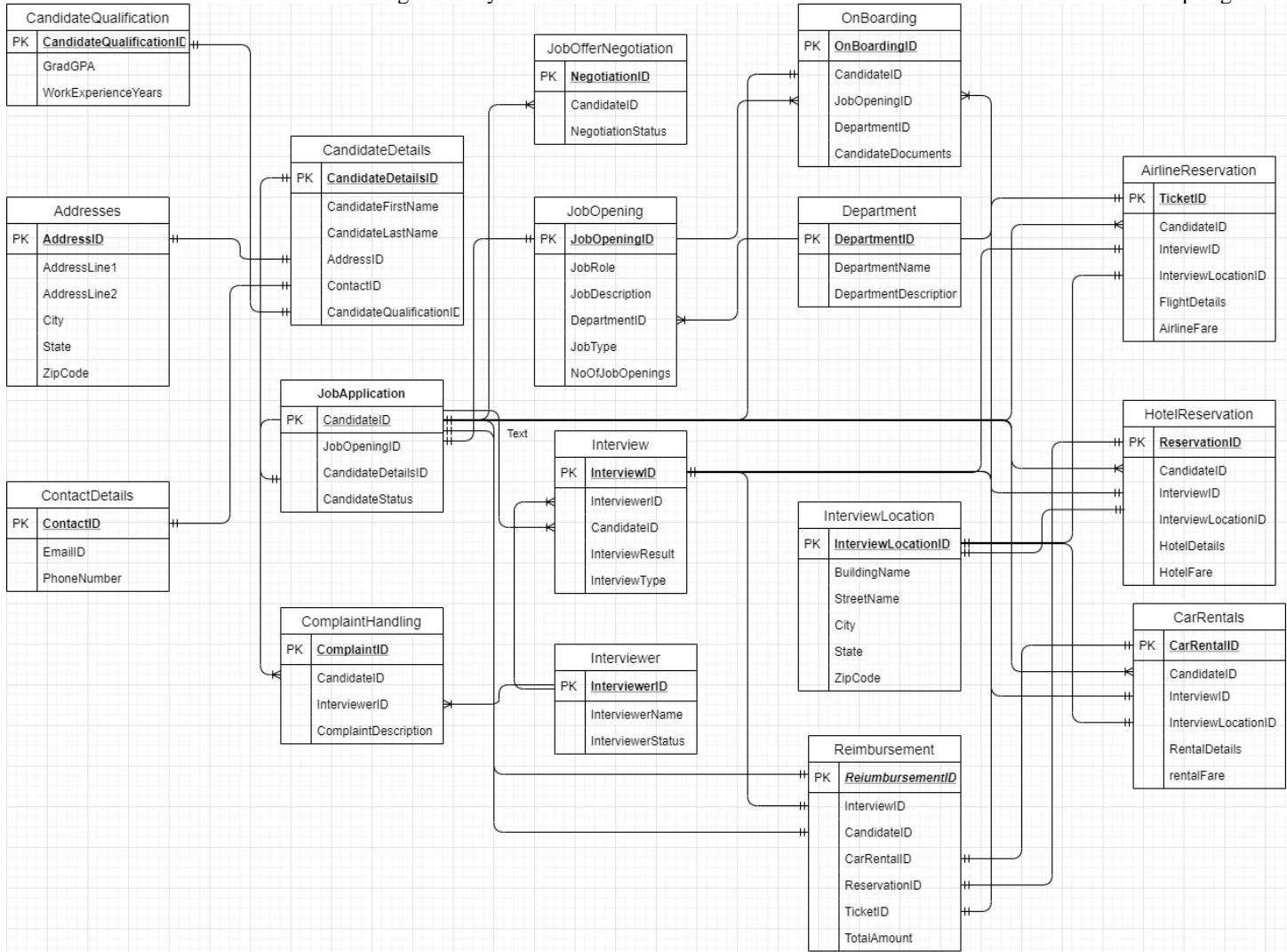
The Entity Relationship(E/R) diagram mainly focusses on how entities in the database are related to each other and how to link them.

Here is the E/R diagram given below that is used for my project.



NORMALIZED DATABASE DESIGN

For a database design to be in 3rd Normal form, it must be in First Normal Form i.e. it must not contain any non repeating columns and should contain scalar values in all rows, and it must also be in 2nd Normal form(i.e. no partial dependencies) and also all the columns must only depend on the primary key(i.e. no transitive dependencies). So here is my database design in 3rd Normal form given below:



2.5 Potential Integrity and Security issues

The potential security and integrity is reduced by transforming the database into the 3NF format.

For referential integrity to hold in a relational database, any column in a base table that is declared a foreign key can only contain either null values or values from a parent table's primary key or a candidate key. In other words, when a foreign key value is used it must reference a valid, existing primary key in the parent table. For instance, deleting a record that contains a value referred to by a foreign key in another table would break referential integrity.

3. IMPLEMENTATION PHASE OF THE PROJECT

We implement the database of this project by firstly creating database, creating tables and columns which we have discussed in the design phase and adding the trigger to solve the problem statement.

3.1 DATABASE CREATION

We create Database and name it ‘XYZRecruitmentDatabase’.

3.2 Constraints

There are two types of constraints will be there. They are:

1. Normal Constraints:- Normal constraints are used to specify rules for the data in a table. Constraints are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the table. If there is any violation between the constraint and the data action, the action is aborted. Various types of normal constraints are: NOT NULL, PRIMARY KEY, FOREIGN KEY, CHECK, DEFAULT, UNIQUE, INDEX.
2. Business Constraints: They include the relationships between the tables such as One-to-One, Many-to-One, Many-to-One.

Following are few Constraints like primary key , foreign key , Nullable and relationships which are used in this project.

1) For Address Table

AddressID INT NOT NULL PRIMARY KEY

AddressLine1 VARCHAR(50) NOT NULL

AddressLine2 VARCHAR(50) Default NULL

City CHAR(50) NOT NULL

State CHAR(50) NOT NULL

ZipCode INT NOT NULL

2) For ContactDetails Table

ContactID INT NOT NULL PRIMARY KEY

EmailAddress VARCHAR(50) NOT NULL

PhoneNumber VARCHAR(12) NOT NULL

3) For Department Table

DepartmentID INT NOT NULL PRIMARY KEY

DepartmentName CHAR(30) NOT NULL

4) For CandidateQualification Table

CandidateQualificationID INT NOT NULL PRIMARY KEY

GradGpa INT NOT NULL

WorkExperienceYears INT NOT NULL

5) For Interviewer Table

InterviewerID INT NOT NULL PRIMARY KEY

InterviewerName CHAR(50) NOT NULL

InterviewerStatus Varchar(50) NULL

6) For InterviewLocation Table

InterviewLocationID INT NOT NULL PRIMARY KEY

BuildingName VARCHAR(50) NOT NULL

City CHAR(50) NOT NULL

State CHAR(50) NOT NULL

ZipCode INT NOT NULL

7) For JobOpening Table

JobOpeningID INT NOT NULL PRIMARY KEY

JobRole VARCHAR(50) NOT NULL

JobDescription VARCHAR(50) NOT NULL

DepartmentID INT NOT NULL REFERENCES Department(DepartmentID)

JobType VARCHAR(50) NOT NULL

NoOfJobOpenings INT NOT NULL

8) For CandidateDetails Table

CandidateDetailsID INT NOT NULL PRIMARY KEY

CandidateName CHAR(50)

AddressID INT NOT NULL REFERENCES Addresses(AddressID)

ContactID INT NOT NULL REFERENCES ContactDetails(ContactID)

CandidateQualificationID INT NOT NULL REFERENCES CandidateQualification(CandidateQualificationID)

9) JobApplication Table

CandidateID INT NOT NULL PRIMARY KEY

CandidateDetailsID INT NOT NULL REFERENCES CandidateDetails(CandidateDetailsID)
CandidateStatus VARCHAR(50) NOT NULL

10) For Interview Table

InterviewID INT NOT NULL PRIMARY KEY
InterviewerID INT NOT NULL References Interviewer(InterviewerID)
CandidateID INT NOT NULL References JobApplication(CandidateID)
InterviewResult CHAR(50) NOT NULL
InterviewType VARCHAR(20) NOT NULL

11) For ComplaintHandling Table

ComplaintID INT NOT NULL PRIMARY KEY
CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID)
InterviewerID INT NOT NULL References Interviewer(InterviewerID)
ComplaintDescription VARCHAR(100)

12) For AirlineReservation Table

TicketID INT NOT NULL PRIMARY KEY
InterviewID INT NOT NULL REFERENCES Interview(InterviewID)
CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID)
InterviewLocationID INT NOT NULL REFERENCES InterviewLocation(InterviewLocationID),
FlightDetails VARCHAR(100)
AirlineFare money

13) Onboarding Table

OnboardingID INT NOT NULL PRIMARY KEY
CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID)
JobOpeningID INT NOT NULL REFERENCES JobOpening(JobOpeningID)
DepartmentID INT NOT NULL REFERENCES Department(DepartmentID)
CandidateDocuments Varchar(50) NOT NULL

14) HotelReservation Table

ReservationID INT NOT NULL PRIMARY KEY
InterviewID INT NOT NULL REFERENCES Interview(InterviewID)

InterviewLocationID INT NOT NULL REFERENCES InterviewLocation(InterviewLocationID)

HotelDetails VARCHAR(100) NOT NULL

HotelFare money NOT NULL

15) For CarRental Table

RentalID INT NOT NULL PRIMARY KEY

InterviewID INT NOT NULL REFERENCES Interview(InterviewID)

CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID)

InterviewLocationID INT NOT NULL REFERENCES InterviewLocation(InterviewLocationID)

CarDetails VARCHAR(50) NOT NUL

RentalFare money NOT NULL

16) For Reimbursement Table

ReimbursementID INT NOT NULL PRIMARY KEY

InterviewID INT NOT NULL REFERENCES Interview(InterviewID)

CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID)

RentalID INT NOT NULL REFERENCES CarRental(RentalID)

TicketID INT NOT NULL REFERENCES AirlineReservation(TicketID)

ReservationID INT NOT NULL REFERENCES HotelReservation(ReservationID)

ReimbursementAmount MONEY

17) For JobOfferNegotiation Table

NegotiationID INT NOT NULL PRIMARY KEY

CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID)

NegotitationStatus varchar(50) NOT NULL

Here are the list of relations that are used between the tables in my project.

In this project I have used 17 One-to-One relations, 11 Many-to-One relations

- There is One to One relation between JobApplication and CandidateDetails on CandidateDetailsID
- There is One to Many relation between JobApplication and ComplaintHandling on CandidateID
- There is One to One relation between CandidateDetails and CandidateQualification on CandidateQualificationID
- There is One to One relation between CandidateDetails and Addresses on AddressID
- There is One to One relation between CandidateDetails and ContactDetails on ContactID
- There is One to Many relation between JobApplication and JobOfferNegotiation on CandidateID

- There is One to One relation between JobApplication and JobOpening on JobOpeningID
- There is One to Many relation between Interviewer and Interview on InterviewerID
- There is One to Many relation between Interviewer and ComplaintHandling on InterviewerID
- There is One to Many relation between JobApplication and Interview on CandidateID
- There is One to One relation between JobApplication and OnBoarding on CandidateID
- There is One to Many relation between JobOpening and OnBoarding on JobOpeningID
- There is One to Many relation between Department and OnBoarding on JobOpeningID
- There is One to Many relation between Department and JobOpening on DepartmentID
- There is One to One relation between InterviewLocation and AirlineReservation on InterviewLocationID
- There is One to One relation between InterviewLocation and HotelReservation on InterviewLocationID
- There is One to One relation between InterviewLocation and CarRentals on InterviewLocationID
- There is One to One relation between Reimbursement and AirlineReservation on TicketID
- There is One to One relation between Reimbursement and HotelReservation on ReservationID
- There is One to One relation between Reimbursement and CarRentals on CarRentalID
- There is One to One relation between Reimbursement and JobApplication on CandidateID
- There is One to One relation between Reimbursement and Interview on InterviewID
- There is One to Many relation between JobApplication and AirlineReservation on CandidateID
- There is One to One relation between Interview and AirlineReservation on InterviewID
- There is One to Many relation between JobApplication and HotelReservation on CandidateID
- There is One to One relation between Interview and HotelReservation on InterviewID
- There is One to Many relation between JobApplication and CarRentals on CandidateID
- There is One to One relation between Interview and CarRentals on InterviewID

3.3 TABLE CREATING

WE CREATE DATABASE AND THEN WE CREATE TABLES WITH ABOVE CONSTRAINTS

CSE 581 Introduction to Database Management Systems

Spring2020

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SQLQuery1.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (62))* - Microsoft SQL Server Management Studio". The left pane is the Object Explorer, showing the database structure. The right pane is the Query Editor, displaying T-SQL code for creating a database and a table.

```
IF DB_ID('XYZRecruitmentDatabase') IS NOT NULL
    DROP DATABASE XYZRecruitmentDatabase
GO

CREATE DATABASE XYZRecruitmentDatabase;
GO

USE XYZRecruitmentDatabase;
GO

CREATE TABLE Addresses (
    AddressID INT NOT NULL PRIMARY KEY,
    AddressLine1 VARCHAR(50) NOT NULL,
```

Messages

Commands completed successfully.

Completion time: 2020-05-02T01:24:15.2995979-07:00

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The title bar indicates the file path: "SQLQuery1.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (61))* - Microsoft SQL Server Management Studio". The menu bar includes File, Edit, View, Query, Project, Tools, Window, and Help. The toolbar contains various icons for database management tasks.

The Object Explorer on the left shows the database structure, including the "XYZRecruitmentDatabase" selected. The main pane displays two SQL scripts:

```
CREATE TABLE Interview (
    InterviewLocationID INT NOT NULL REFERENCES InterviewLocation(InterviewLocationID),
    CarDetails VARCHAR(50),
    RentalFare money ,
);

CREATE TABLE Reimbursement (
    ReimbursementID INT NOT NULL PRIMARY KEY,
    InterviewID INT NOT NULL REFERENCES Interview(InterviewID),
    CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID),
    RentalID INT NOT NULL REFERENCES CarRental(RentalID),
    TicketID INT NOT NULL REFERENCES AirlineReservation(TicketID),
)
```

The "Messages" section at the bottom of the main pane shows the execution results:

```
Commands completed successfully.  
Completion time: 2020-05-02T02:15:38.7810677-07:00
```

The status bar at the bottom provides information about the query: "Query executed successfully.", "DESKTOP-A19REFF\SQLEXPRESSS...", "sa (61)", "XYZRecruitmentDatabase", "00:00:01", and "0 rows". The status bar also includes icons for connectivity, search, and system status.

3.4 TRIGGERS

In this project I've used 2 triggers. According to the problem statement whenever a blacklisted candidate applies It should say that the candidate is blacklisted therefore the candidate is prohibited to apply. For this I've created a trigger. When ever a candidate applies we update the candidate details column and when ever the candidate provides his name we check if the name is already present and if the name is matched with the JobApplication table's name we see the status of the candidate and if its Blacklisted the trigger works and we get the message stating he cant apply.

```
USE XYZRecruitmentDatabase;
GO
```

```
CREATE TRIGGER tr_blacklisted_candidate
ON CandidateDetails
FOR INSERT
AS
IF EXISTS
(Select * FROM JobApplication JOIN CandidateDetails
ON JobApplication.CandidateDetailsID = CandidateDetails.CandidateDetailsID
WHERE (JobApplication.CandidateName = CandidateDetails.CandidateName)
AND
(JobApplication.CandidateStatus = 'Blacklisted'))
BEGIN
    PRINT 'Candidate has been blacklisted. Therefore Candidate cannot apply';
END;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects like External Resources, Synonyms, Programmability, Stored Procedures, Functions, Types, Rules, Defaults, Sequences, Service Broker, Storage, Security, and Management. The central pane displays the T-SQL code for creating the trigger:

```
USE XYZRecruitmentDatabase;
GO
CREATE TRIGGER tr_blacklisted_candidate
ON CandidateDetails
FOR INSERT
AS
IF EXISTS
(Select * FROM JobApplication JOIN CandidateDetails
ON JobApplication.CandidateDetailsID = CandidateDetails.CandidateDetailsID
WHERE (JobApplication.CandidateName = CandidateDetails.CandidateName)
AND
(JobApplication.CandidateStatus = 'Blacklisted'))
BEGIN
    PRINT 'Candidate has been blacklisted. Therefore Candidate cannot apply';
END;
```

The code is highlighted with syntax coloring. Below the code, the Messages pane shows the output: "Commands completed successfully." and "Completion time: 2020-05-04T02:30:06.5539710-07:00". At the bottom, a status bar indicates "Query executed successfully." and shows the system information: DESKTOP-A19REFF\SQLEXPRESSS... sa (53) XYZRecruitmentDatabase 00:00:00 0 rows.

INSERT INTO CandidateDetails

```
([CandidateDetailsID],[CandidateName],[AddressID],[ContactID],[CandidateQualificationID])
values (810,'ChandlerBing',101,201,401);
```

Here ChandlerBing is blacklisted and when he applies again name is checked and if the status is blacklisted then we get the message.

```
SQLQuery2.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (55)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
New Query File New Database New Table New View New StoredProcedure New Function New Assembly New Type New Rule New Default New Sequence New ServiceBroker New Storage New Security
XYZRecruitmentDatabase Execute
Object Explorer
SQLQuery2.sql - DE...tDatabase (sa (55))*
SQLQuery1.sql - DE...tDatabase (sa (53))
INSERT INTO CandidateDetails ([CandidateDetailsID],[CandidateName],[AddressID],[ContactID],[CandidateQualificationID])
values (810,'ChandlerBing',101,201,401);
--Venkata sai pawan komaravolu

Messages
Candidate has been blacklisted. Therefore Candidate cannot apply
Candidate Cannot apply as he is been blacklisted

(1 row affected)

Completion time: 2020-05-04T02:35:48.8811616-07:00

100 % 100 %
Query executed successfully.

DESKTOP-A19REFF\SQLEXPRESSS... sa (55) XYZRecruitmentDatabase | 00:00:00 | 0 rows
Ready
Type here to search
Ln 3 Col 31 Ch 31 INS
2:36 AM 5/4/2020
```

Another condition mentioned is that when ever a candidate gets rejected he can complain and if the complaint is approved he will be kept in the interview list otherwise he will be rejected. For this I've created a trigger and when ever there is update in the Complaint Description and it says Invalid then the candidate is rejected and his status is update to rejected.

GO

```
CREATE TRIGGER CandidateComplaintsStatus_Update
```

ON

ComplaintHandling

AFTER UPDATE

AS

BEGIN

```
DECLARE @ComplaintDescription varchar (30), @CandidateID INT;
SELECT @CandidateID = CandidateID , @ComplaintDescription = ComplaintDescription FROM
inserted;
if @ComplaintDescription = 'Invalid'
UPDATE JobApplication
SET CandidateStatus = 'Rejected' where CandidateID = @CandidateID;
```

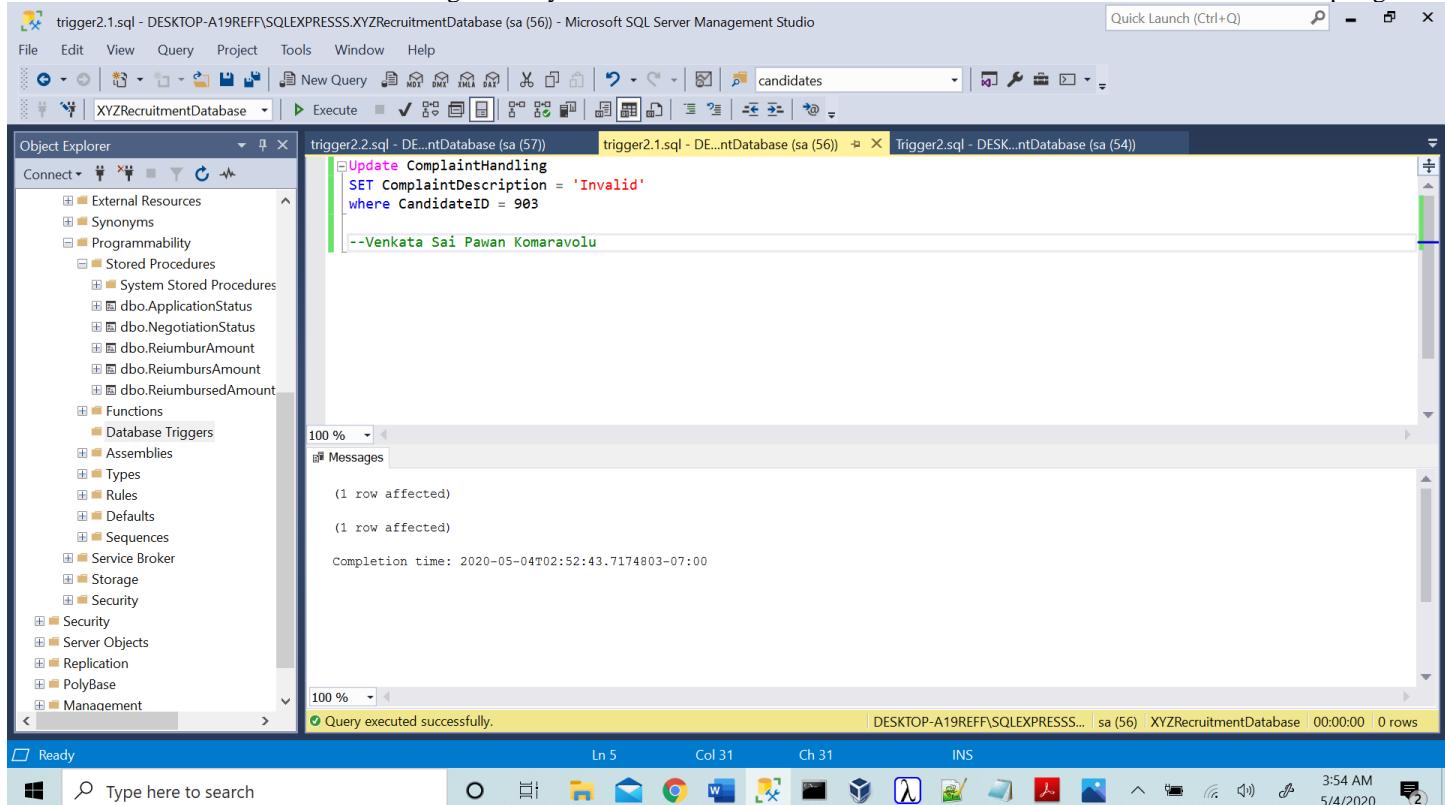
END;

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. In the Object Explorer on the left, there is a tree view of database objects under the 'XYZRecruitmentDatabase' node. The 'Stored Procedures' node is expanded, showing several system stored procedures like 'dbo.ApplicationStatus', 'dbo.NegotiationStatus', etc. In the center pane, a query window titled 'SQLQuery3.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (54))' is open. It contains the following T-SQL code:

```
GO
CREATE TRIGGER CandidateComplaintsStatus_Update
ON
    --Venkata Sai Pawan Komaravolu
    ComplaintHandling
AFTER UPDATE
AS
BEGIN
    DECLARE @ComplaintDescription varchar (30), @CandidateID INT;
    SELECT @CandidateID = CandidateID , @ComplaintDescription = ComplaintDescription FROM
        inserted;
    if @ComplaintDescription = 'Invalid'
        UPDATE JobApplication
        SET CandidateStatus = 'Rejected' where CandidateID = @CandidateID;
END;
```

The status bar at the bottom of the SSMS window indicates 'Query executed successfully.' and '0 rows'. The bottom right corner of the screen shows the Windows taskbar with the date and time.

Update ComplaintHandling
SET ComplaintDescription = 'Invalid'
where CandidateID = 903



trigger2.2.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (56)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

Object Explorer

trigger2.2.sql - DE...ntDatabase (sa (57)) trigger2.1.sql - DE...ntDatabase (sa (56)) Trigger2.sql - DESK...ntDatabase (sa (54))

```
Update ComplaintHandling
SET ComplaintDescription = 'Invalid'
where CandidateID = 903

--Venkata Sai Pawan Komaravolu
```

100 % Messages

(1 row affected)

(1 row affected)

Completion time: 2020-05-04T02:52:43.7174803-07:00

100 % Results

Query executed successfully.

DESKTOP-A19REFF\SQLEXPRESSS... sa (56) XYZRecruitmentDatabase 00:00:00 0 rows

Ready

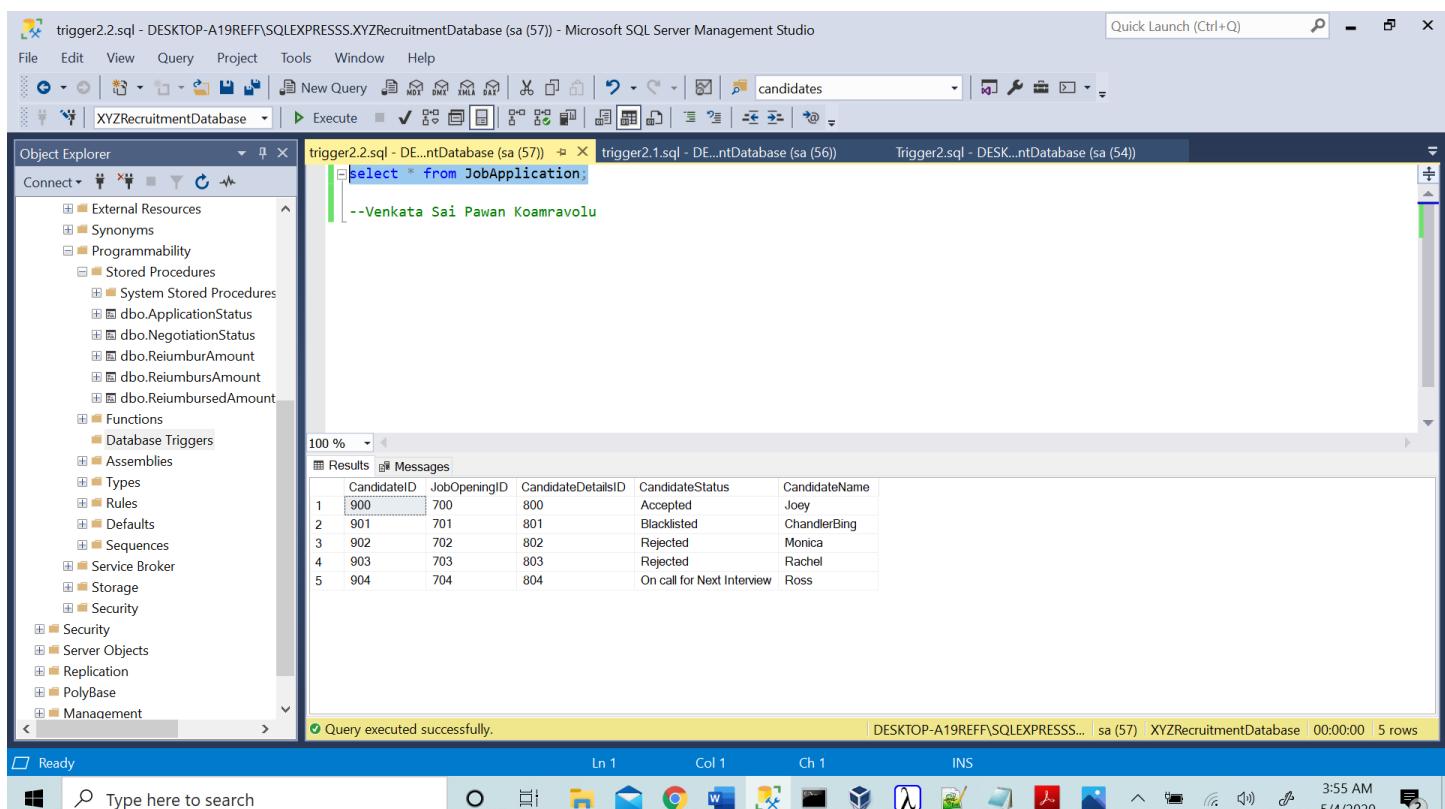
Type here to search

Ln 5 Col 31 Ch 31 INS

3:54 AM 5/4/2020

This screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer pane on the left lists various database objects like External Resources, Synonyms, Programmability, and Stored Procedures. The central pane displays an SQL script for updating a table named 'ComplaintHandling'. The command sets the 'ComplaintDescription' column to 'Invalid' for the row where 'CandidateID' is 903. A comment at the end of the script identifies the developer as 'Venkata Sai Pawan Komaravolu'. Below the script, the output window shows two messages indicating rows were affected. The status bar at the bottom right shows the completion time as 2020-05-04T02:52:43.7174803-07:00. The taskbar at the bottom includes icons for File Explorer, Mail, Google Chrome, Word, Excel, and others.

select * from JobApplication;



trigger2.2.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (57)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

Object Explorer

trigger2.2.sql - DE...ntDatabase (sa (57)) trigger2.1.sql - DE...ntDatabase (sa (56)) Trigger2.sql - DESK...ntDatabase (sa (54))

```
select * from JobApplication
```

--Venkata Sai Pawan Komaravolu

100 % Results

CandidateID	JobOpeningID	CandidateDetailsID	CandidateStatus	CandidateName
900	700	800	Accepted	Joey
901	701	801	Blacklisted	ChandlerBing
902	702	802	Rejected	Monica
903	703	803	Rejected	Rachel
904	704	804	On call for Next Interview	Ross

100 % Messages

Query executed successfully.

DESKTOP-A19REFF\SQLEXPRESSS... sa (57) XYZRecruitmentDatabase 00:00:00 5 rows

Ready

Type here to search

Ln 1 Col 1 Ch 1 INS

3:55 AM 5/4/2020

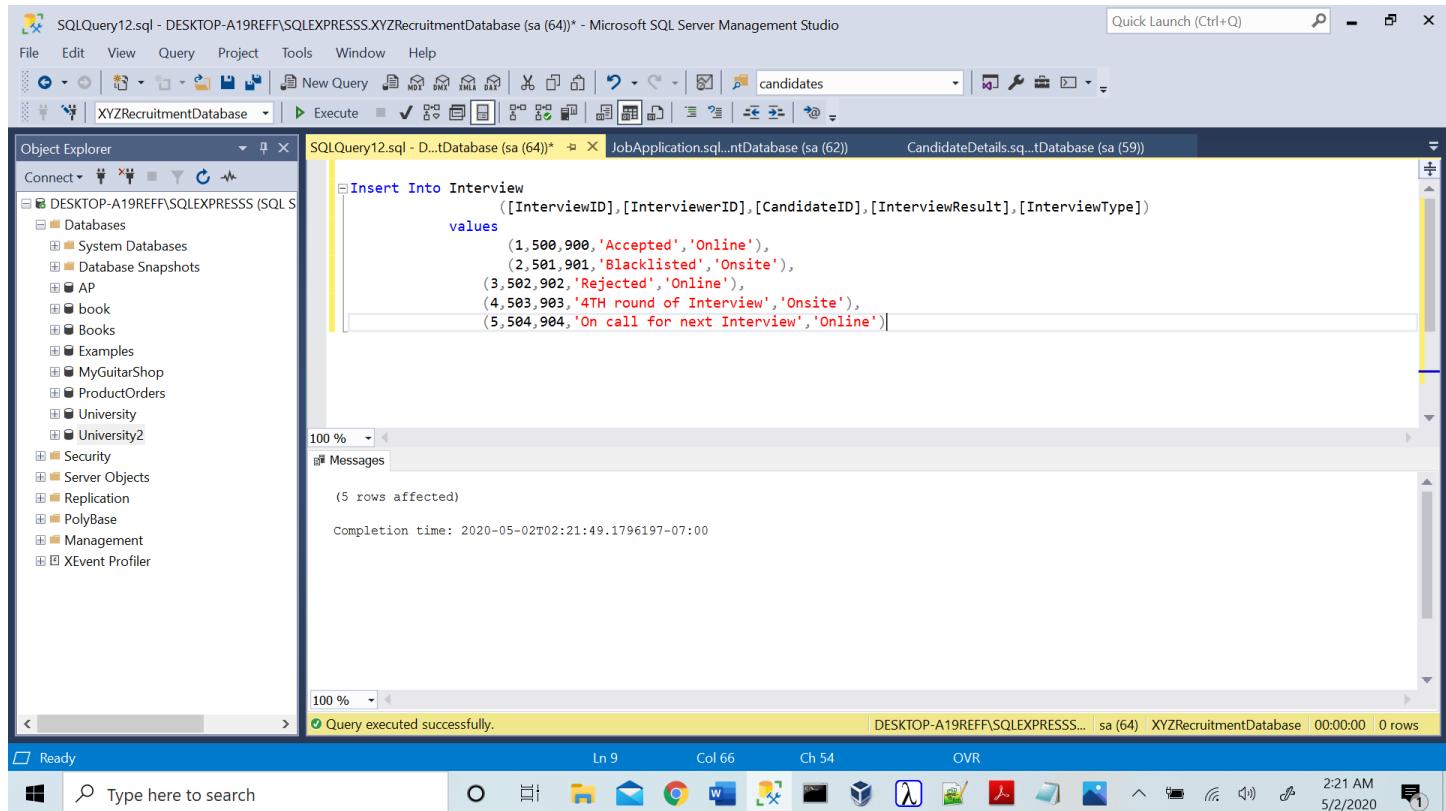
This screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer pane on the left lists various database objects. The central pane displays a simple SELECT query against the 'JobApplication' table. The results are shown in a grid, listing five rows with columns: CandidateID, JobOpeningID, CandidateDetailsID, CandidateStatus, and CandidateName. The data includes entries for Joey (Accepted), ChandlerBing (Blacklisted), Monica (Rejected), Rachel (Rejected), and Ross (On call for Next Interview). The status bar at the bottom right shows the execution time as 00:00:00 and 5 rows affected. The taskbar at the bottom includes icons for File Explorer, Mail, Google Chrome, Word, Excel, and others.

4. TESTING PHASE

In testing phase we populate the database by inserting values into tables of the database which we have created and then to test the scenarios we have used views, stored procedure, Functions, Scripts and transaction.

4.1 POPULATING DATABASE

We now insert values into the tables and columns using the INSERT INTO statement.



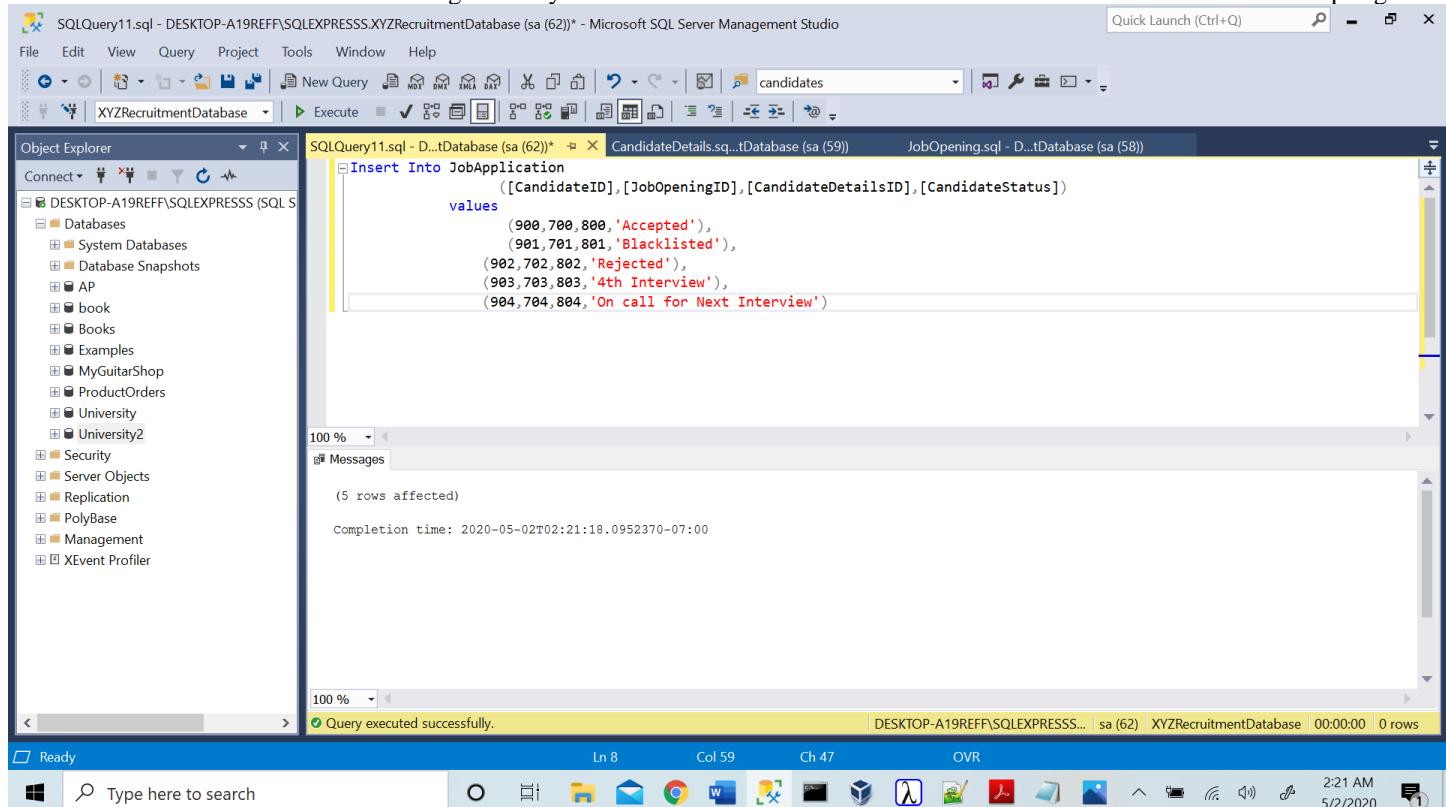
The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with various databases listed. The central pane contains a query window with the following SQL code:

```
Insert Into Interview
([InterviewID],[InterviewerID],[CandidateID],[InterviewResult],[InterviewType])
values
(1,500,900,'Accepted','Online'),
(2,501,901,'Blacklisted','Onsite'),
(3,502,902,'Rejected','Online'),
(4,503,903,'4TH round of Interview','Onsite'),
(5,504,904,'On call for next Interview','Online')
```

Below the code, the message pane shows:

(5 rows affected)
Completion time: 2020-05-02T02:21:49.1796197-07:00

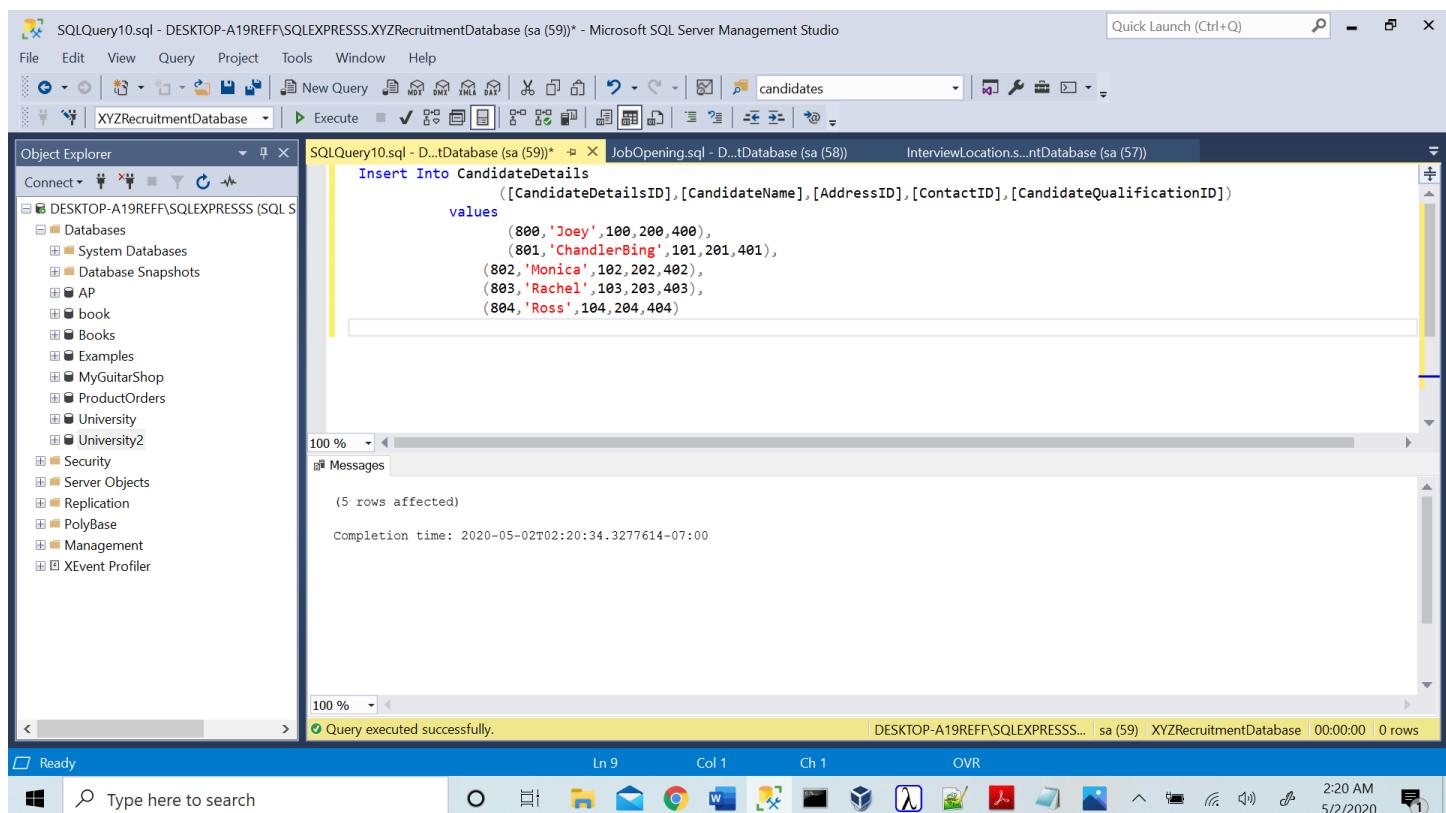
The status bar at the bottom indicates "Query executed successfully." and shows the system date and time as 5/2/2020 2:21 AM.



SQLQuery11.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (62)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query MDI DML XML Execute
XYZRecruitmentDatabase Candidates
Quick Launch (Ctrl+Q) X
Object Explorer
Connect > DESKTOP-A19REFF\SQLEXPRESSS (SQL Server)
  Databases
    System Databases
    Database Snapshots
    AP
    book
    Books
    Examples
    MyGuitarShop
    ProductOrders
    University
    University2
    Security
    Server Objects
    Replication
    PolyBase
    Management
    XEvent Profiler
SQLQuery11.sql - D...tDatabase (sa (62))> CandidateDetails.sq...tDatabase (sa (59)) JobOpening.sql - D...tDatabase (sa (58))
Insert Into JobApplication
([CandidateID], [JobOpeningID], [CandidateDetailsID], [CandidateStatus])
values
(900, 700, 800, 'Accepted'),
(901, 701, 801, 'Blacklisted'),
(902, 702, 802, 'Rejected'),
(903, 703, 803, '4th Interview'),
(904, 704, 804, 'On call for Next Interview')

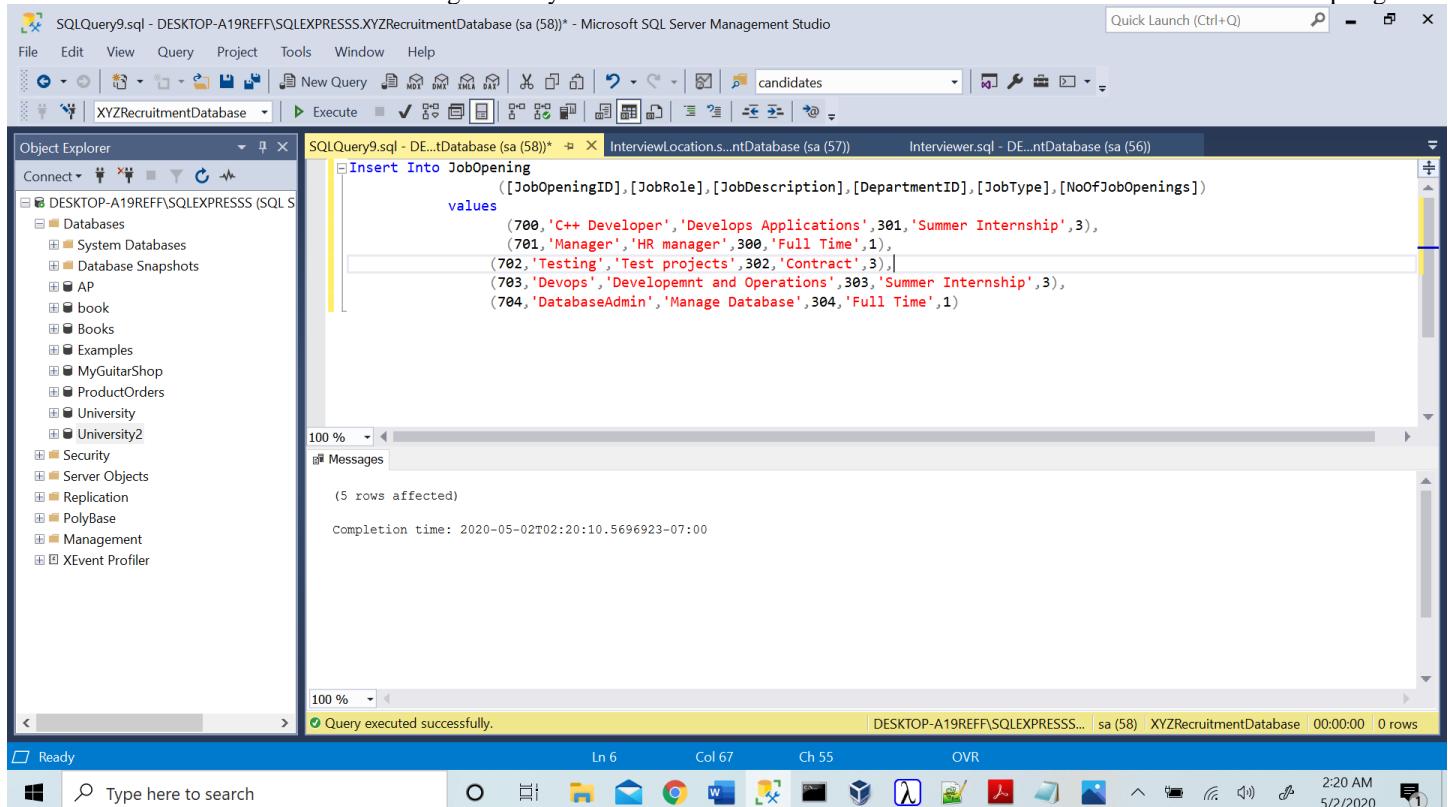
100 % < > Messages
(5 rows affected)
Completion time: 2020-05-02T02:18:09.952370-07:00
100 % < > Messages
Query executed successfully.
DESKTOP-A19REFF\SQLEXPRESSS... sa (62) XYZRecruitmentDatabase 00:00:00 0 rows
Ready Type here to search Ln 8 Col 59 Ch 47 OVR
2:21 AM 5/2/2020
```



SQLQuery10.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (59)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query MDI DML XML Execute
XYZRecruitmentDatabase Candidates
Quick Launch (Ctrl+Q) X
Object Explorer
Connect > DESKTOP-A19REFF\SQLEXPRESSS (SQL Server)
  Databases
    System Databases
    Database Snapshots
    AP
    book
    Books
    Examples
    MyGuitarShop
    ProductOrders
    University
    University2
    Security
    Server Objects
    Replication
    PolyBase
    Management
    XEvent Profiler
SQLQuery10.sql - D...tDatabase (sa (59))> JobOpening.sql - D...tDatabase (sa (58)) InterviewLocation.s...ntDatabase (sa (57))
Insert Into CandidateDetails
([CandidateDetailsID], [CandidateName], [AddressID], [ContactID], [CandidateQualificationID])
values
(800, 'Joey', 100, 200, 400),
(801, 'ChandlerBing', 101, 201, 401),
(802, 'Monica', 102, 202, 402),
(803, 'Rachel', 103, 203, 403),
(804, 'Ross', 104, 204, 404)

100 % < > Messages
(5 rows affected)
Completion time: 2020-05-02T02:20:34.3277614-07:00
100 % < > Messages
Query executed successfully.
DESKTOP-A19REFF\SQLEXPRESSS... sa (59) XYZRecruitmentDatabase 00:00:00 0 rows
Ready Type here to search Ln 9 Col 1 Ch 1 OVR
2:20 AM 5/2/2020
```



SQLQuery9.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (58)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query MDW BME XML Execute
XYZRecruitmentDatabase Candidates
Object Explorer
Connect > DESKTOP-A19REFF\SQLEXPRESSS (SQL S
  Databases
    System Databases
    Database Snapshots
  AP
  book
  Books
  Examples
  MyGuitarShop
  ProductOrders
  University
  University2
  Security
  Server Objects
  Replication
  PolyBase
  Management
  XEvent Profiler

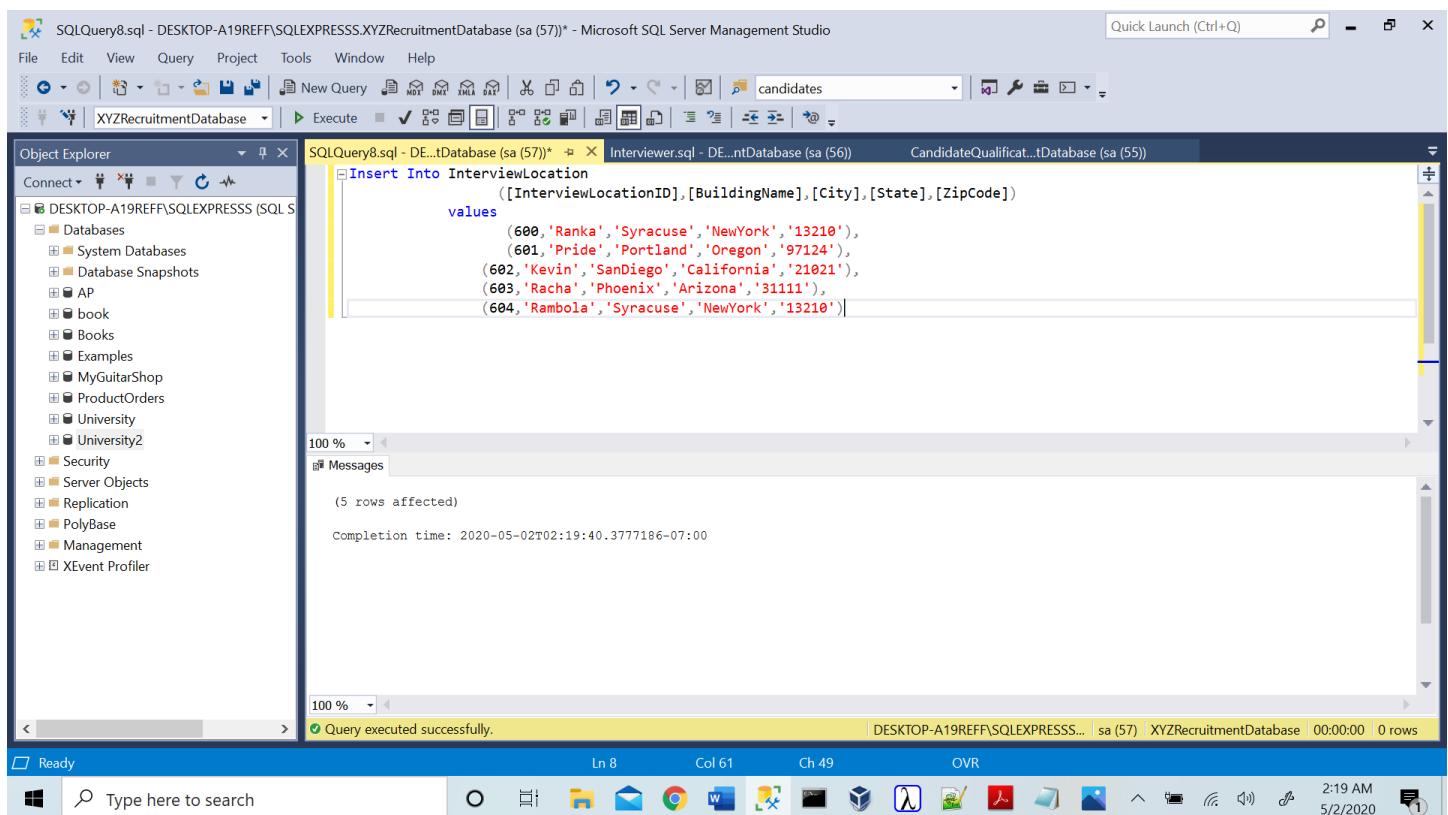
SQLQuery9.sql - DE...tDatabase (sa (58))> InterviewLocation.s...ntDatabase (sa (57))
Interviewer.sql - DE...ntDatabase (sa (56))

Insert Into JobOpening
([JobOpeningID], [JobRole], [JobDescription], [DepartmentID], [JobType], [NoOfJobOpenings])
values
(700, 'C++ Developer', 'Develops Applications', 301, 'Summer Internship', 3),
(701, 'Manager', 'HR manager', 300, 'Full Time', 1),
(702, 'Testing', 'Test projects', 302, 'Contract', 3),
(703, 'Devops', 'Development and Operations', 303, 'Summer Internship', 3),
(704, 'DatabaseAdmin', 'Manage Database', 304, 'Full Time', 1)

100 %
Messages
(5 rows affected)
Completion time: 2020-05-02T02:20:10.5696923-07:00

100 %
100 %
Ready
Type here to search Ln 6 Col 67 Ch 55 OVR
2:20 AM 5/2/2020
```

Query executed successfully.



SQLQuery8.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (57)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query MDW BME XML Execute
XYZRecruitmentDatabase Candidates
Object Explorer
Connect > DESKTOP-A19REFF\SQLEXPRESSS (SQL S
  Databases
    System Databases
    Database Snapshots
  AP
  book
  Books
  Examples
  MyGuitarShop
  ProductOrders
  University
  University2
  Security
  Server Objects
  Replication
  PolyBase
  Management
  XEvent Profiler

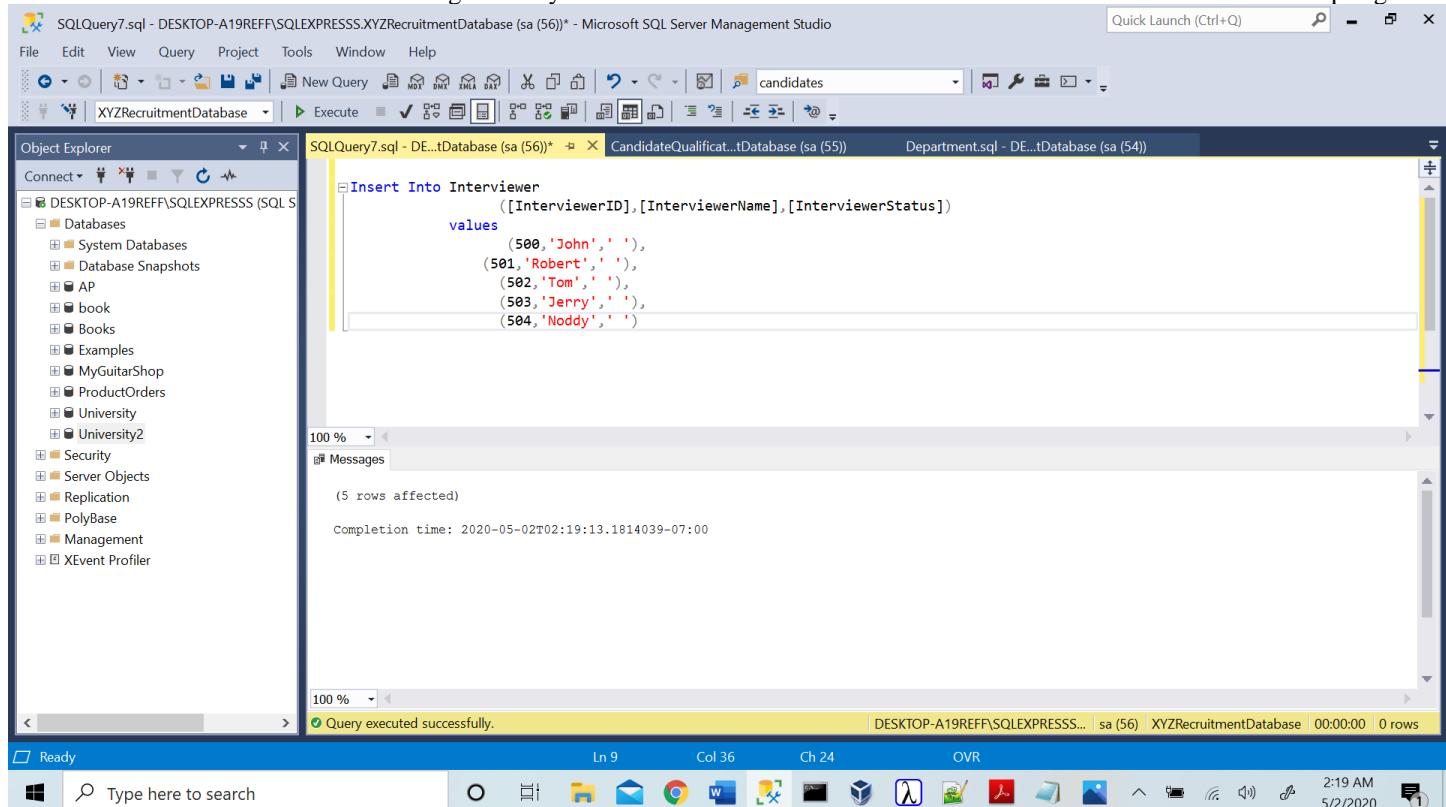
SQLQuery8.sql - DE...tDatabase (sa (57))> Interviewer.sql - DE...ntDatabase (sa (56))
CandidateQualificat...tDatabase (sa (55))

Insert Into InterviewLocation
([InterviewLocationID], [BuildingName], [City], [State], [ZipCode])
values
(600, 'Ranka', 'Syracuse', 'NewYork', '13210'),
(601, 'Pride', 'Portland', 'Oregon', '97124'),
(602, 'Kevin', 'SanDiego', 'California', '21021'),
(603, 'Racha', 'Phoenix', 'Arizona', '31111'),
(604, 'Rambola', 'Syracuse', 'NewYork', '13210')

100 %
Messages
(5 rows affected)
Completion time: 2020-05-02T02:19:40.3777186-07:00

100 %
100 %
Ready
Type here to search Ln 8 Col 61 Ch 49 OVR
2:19 AM 5/2/2020
```

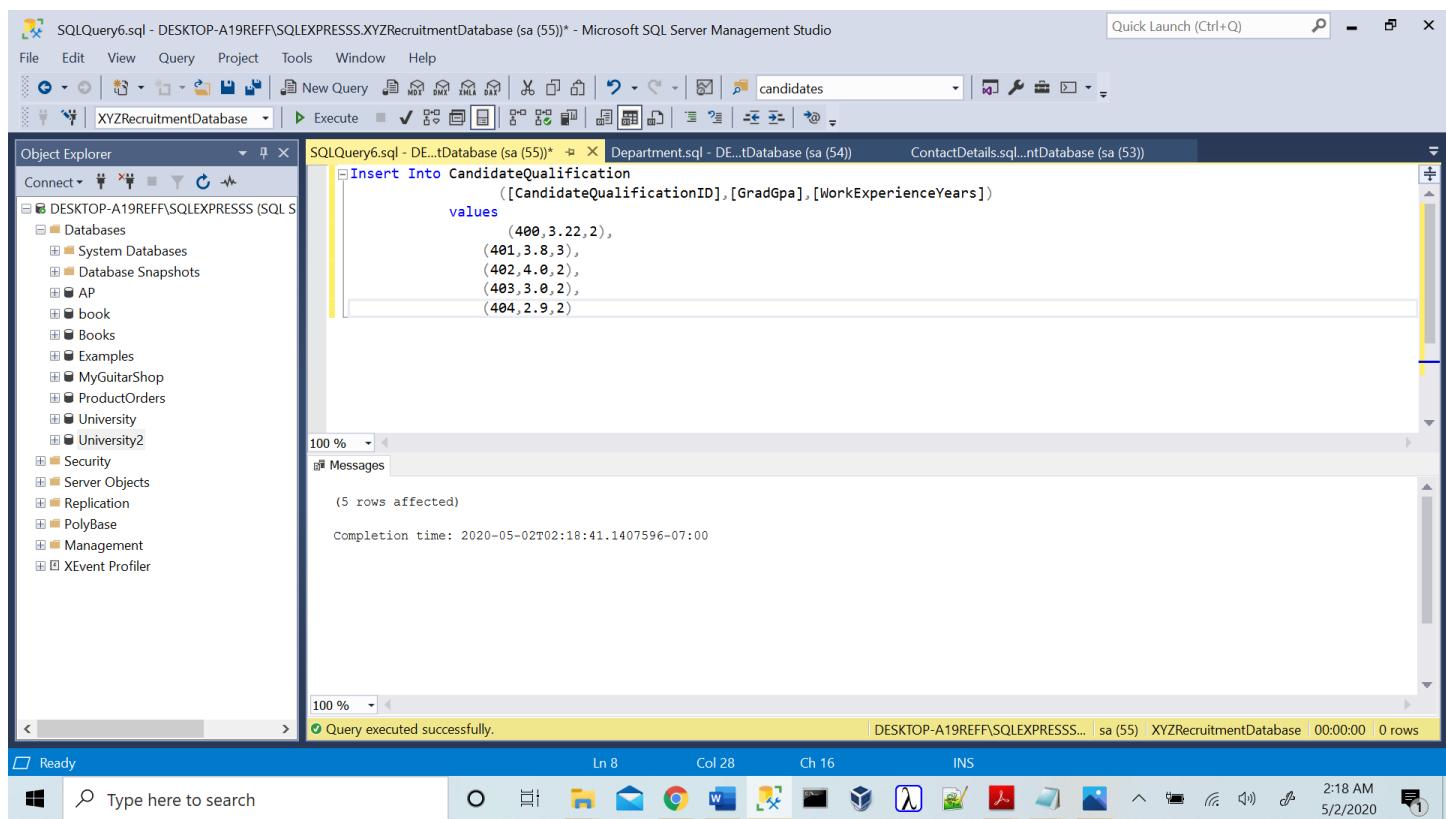
Query executed successfully.



SQLQuery7.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (56)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query MDW BME XML Execute
XYZRecruitmentDatabase Candidates
Quick Launch (Ctrl+Q) X
Object Explorer
Connect > DESKTOP-A19REFF\SQLEXPRESSS (SQL S
  Databases
    System Databases
    Database Snapshots
    AP
    book
    Books
    Examples
    MyGuitarShop
    ProductOrders
    University
    University2
    Security
    Server Objects
    Replication
    PolyBase
    Management
    XEvent Profiler
SQLQuery7.sql - DE...tDatabase (sa (56))*
CandidateQualificat...tDatabase (sa (55))
Department.sql - DE...tDatabase (sa (54))
Insert Into Interviewer
([InterviewerID],[InterviewerName],[InterviewerStatus])
values
(500,'John',' '),
(501,'Robert',' '),
(502,'Tom',' '),
(503,'Jerry',' '),
(504,'Noddy',' ')
100 % Messages
(5 rows affected)
Completion time: 2020-05-02T02:19:13.1814039-07:00
100 % < >
Query executed successfully.
DESKTOP-A19REFF\SQLEXPRESSS... sa (56) XYZRecruitmentDatabase 00:00:00 0 rows
Ready Type here to search Ln 9 Col 36 Ch 24 OVR
2:19 AM 5/2/2020

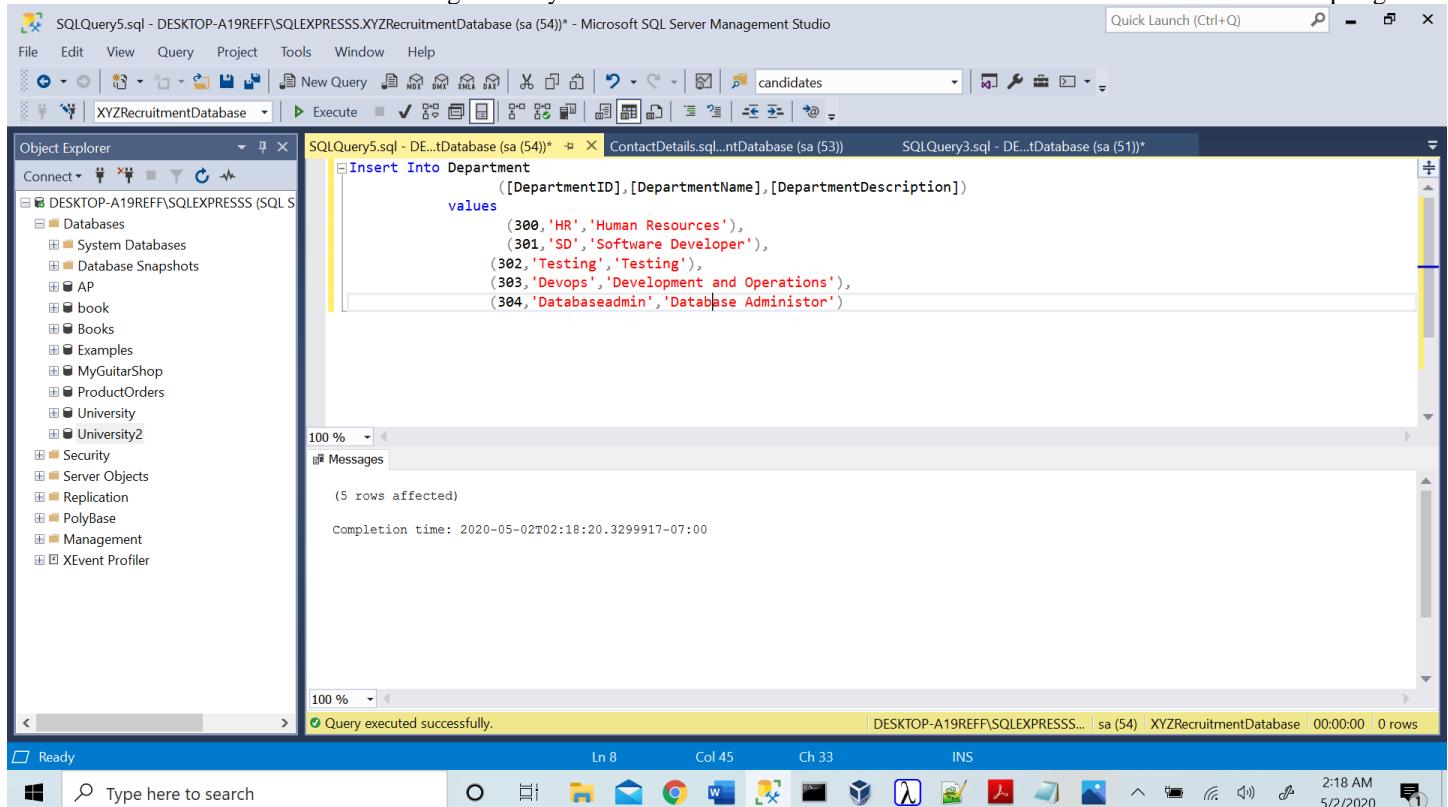
```



SQLQuery6.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (55)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query MDW BME XML Execute
XYZRecruitmentDatabase Candidates
Quick Launch (Ctrl+Q) X
Object Explorer
Connect > DESKTOP-A19REFF\SQLEXPRESSS (SQL S
  Databases
    System Databases
    Database Snapshots
    AP
    book
    Books
    Examples
    MyGuitarShop
    ProductOrders
    University
    University2
    Security
    Server Objects
    Replication
    PolyBase
    Management
    XEvent Profiler
SQLQuery6.sql - DE...tDatabase (sa (55))*
Department.sql - DE...tDatabase (sa (54))
ContactDetails.sql...ntDatabase (sa (53))
Insert Into CandidateQualification
([CandidateQualificationID],[GradGpa],[WorkExperienceYears])
values
(400,3.22,2),
(401,3.8,3),
(402,4.0,2),
(403,3.0,2),
(404,2.9,2)
100 % Messages
(5 rows affected)
Completion time: 2020-05-02T02:18:41.1407596-07:00
100 % < >
Query executed successfully.
DESKTOP-A19REFF\SQLEXPRESSS... sa (55) XYZRecruitmentDatabase 00:00:00 0 rows
Ready Type here to search Ln 8 Col 28 Ch 16 INS
2:18 AM 5/2/2020

```



SQLQuery5.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (54))* - Microsoft SQL Server Management Studio

```
Object Explorer
SQLQuery5.sql - DE...tDatabase (sa (54))*
XYZRecruitmentDatabase (sa (53))
SQLQuery3.sql - DE...tDatabase (sa (51))*
```

```
Insert Into Department
([DepartmentID],[DepartmentName],[DepartmentDescription])
values
(300,'HR','Human Resources'),
(301,'SD','Software Developer'),
(302,'Testing','Testing'),
(303,'Devops','Development and Operations'),
(304,'Databaseadmin','Database Administrator')
```

100 %

Messages

(5 rows affected)

Completion time: 2020-05-02T02:18:20.3299917-07:00

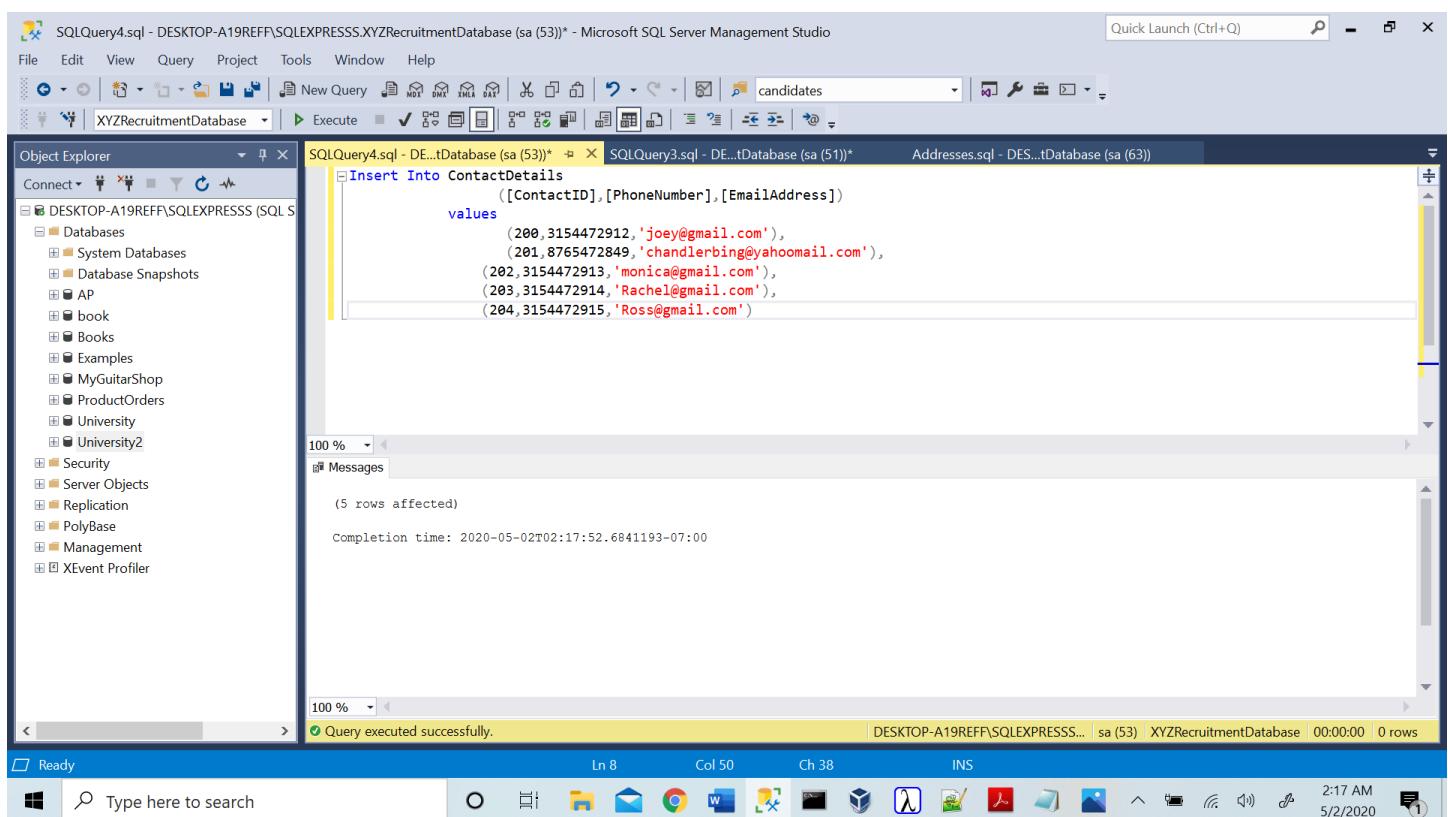
Ready

Type here to search

Ln 8 Col 45 Ch 33 INS

DESKTOP-A19REFF\SQLEXPRESSS... sa (54) XYZRecruitmentDatabase 00:00:00 0 rows

2:18 AM 5/2/2020



SQLQuery4.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (53))* - Microsoft SQL Server Management Studio

```
Object Explorer
SQLQuery4.sql - DE...tDatabase (sa (53))*
XYZRecruitmentDatabase (sa (51))
SQLQuery3.sql - DE...tDatabase (sa (51))*
Addresses.sql - DES...tDatabase (sa (63))*
```

```
Insert Into ContactDetails
([ContactID],[PhoneNumber],[EmailAddress])
values
(200,3154472912,'joey@gmail.com'),
(201,8765472849,'chandlerbing@yahooemail.com'),
(202,3154472913,'monica@gmail.com'),
(203,3154472914,'Rachel@gmail.com'),
(204,3154472915,'Ross@gmail.com')
```

100 %

Messages

(5 rows affected)

Completion time: 2020-05-02T02:17:52.6841193-07:00

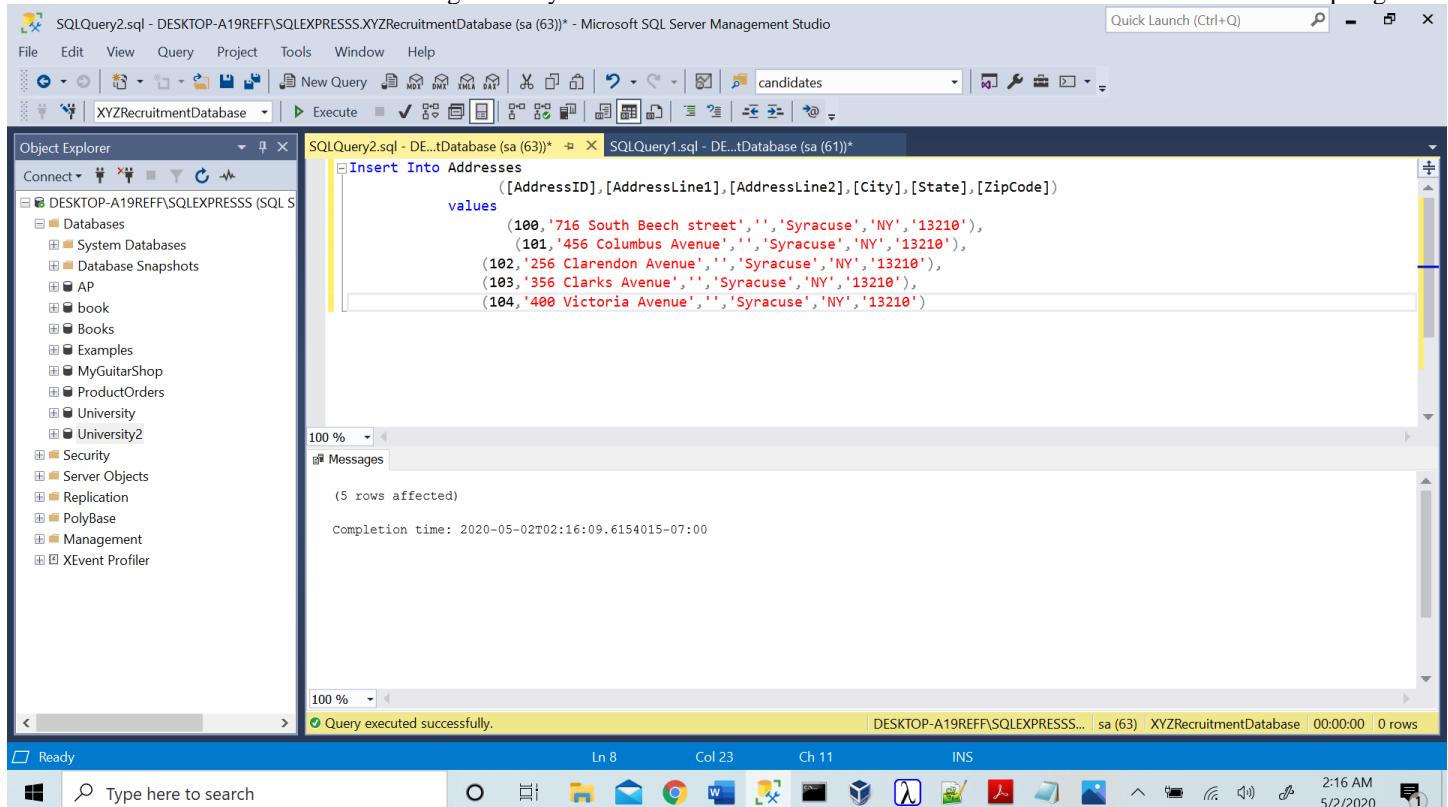
Ready

Type here to search

Ln 8 Col 50 Ch 38 INS

DESKTOP-A19REFF\SQLEXPRESSS... sa (53) XYZRecruitmentDatabase 00:00:00 0 rows

2:17 AM 5/2/2020



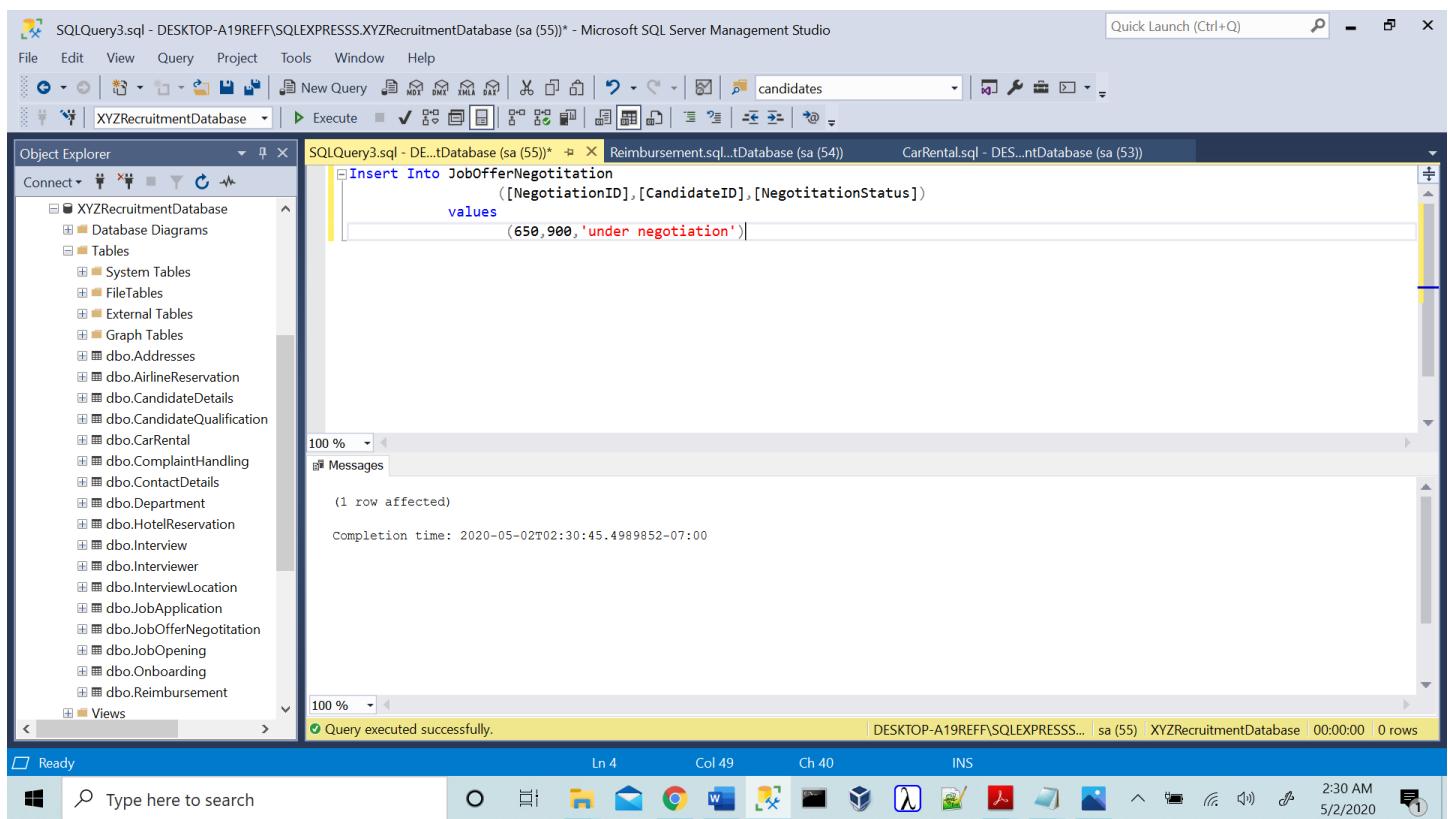
SQLQuery2.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (63)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query MDI DMS XML Execute
XYZRecruitmentDatabase Candidates
Object Explorer
Connect > XYZRecruitmentDatabase
DESKTOP-A19REFF\SQLEXPRESSS (SQL Server)
  Databases
    System Databases
    Database Snapshots
    AP
    book
    Books
    Examples
    MyGuitarShop
    ProductOrders
    University
    University2
    Security
    Server Objects
    Replication
    PolyBase
    Management
    XEvent Profiler
SQLQuery2.sql - DE...tDatabase (sa (63))*
SQLQuery1.sql - DE...tDatabase (sa (61))*
Insert Into Addresses
([AddressID],[AddressLine1],[AddressLine2],[City],[State],[ZipCode])
values
(100,'716 South Beech street','','Syracuse','NY','13210'),
(101,'456 Columbus Avenue','','Syracuse','NY','13210'),
(102,'256 Clarendon Avenue','','Syracuse','NY','13210'),
(103,'356 Clarks Avenue','','Syracuse','NY','13210'),
(104,'400 Victoria Avenue','','Syracuse','NY','13210')

100 %
Messages
(5 rows affected)

Completion time: 2020-05-02T02:16:09.6154015-07:00
```

Ready Type here to search Ln 8 Col 23 Ch 11 INS 2:16 AM 5/2/2020



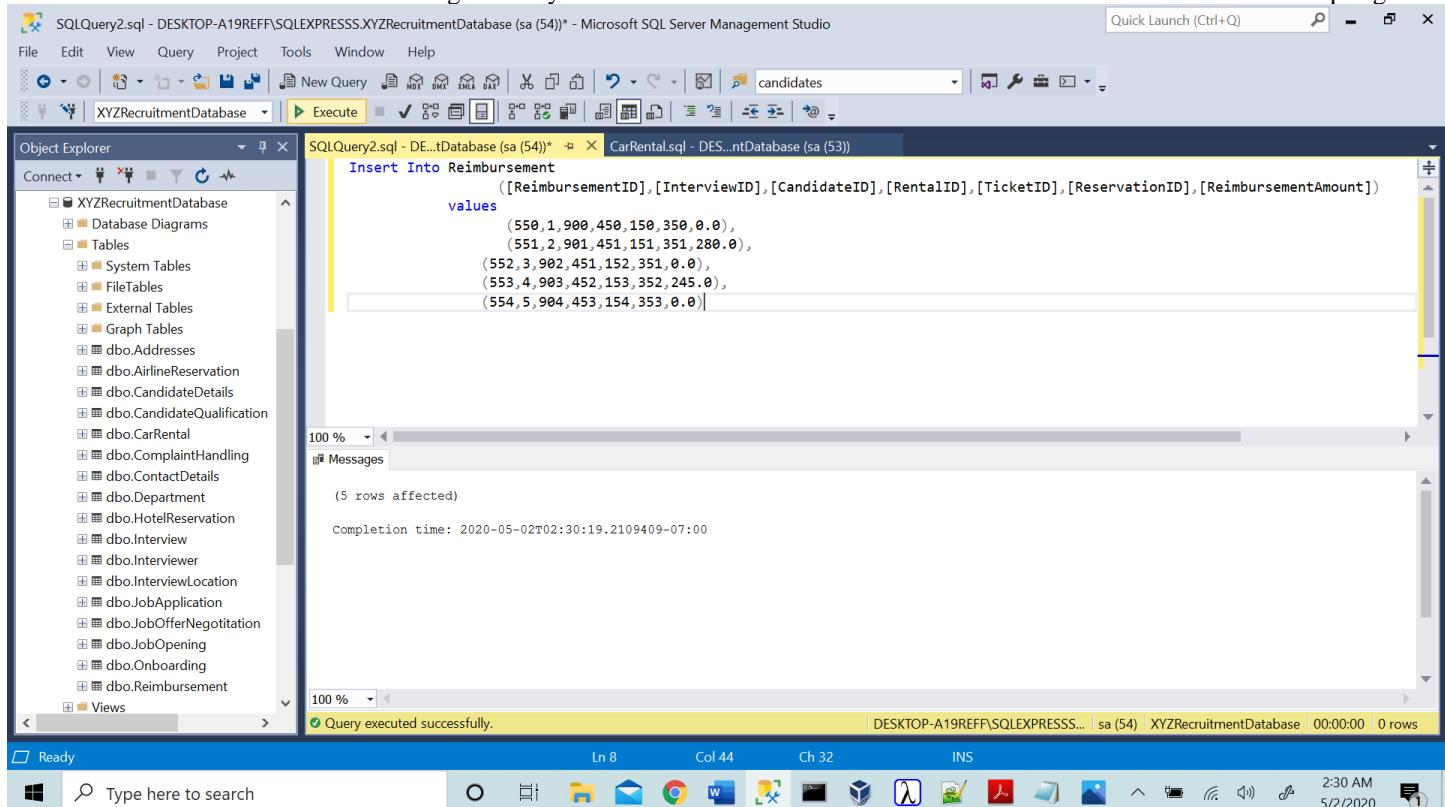
SQLQuery3.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (55)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query MDI DMS XML Execute
XYZRecruitmentDatabase Candidates
Object Explorer
Connect > XYZRecruitmentDatabase
  Database Diagrams
  Tables
    System Tables
    FileTables
    External Tables
    Graph Tables
    dbo.Addresses
    dbo.AirlineReservation
    dbo.CandidateDetails
    dbo.CandidateQualification
    dbo.CarRental
    dbo.ComplaintHandling
    dbo.ContactDetails
    dbo.Department
    dbo.HotelReservation
    dbo.Interview
    dbo.Interviewer
    dbo.InterviewLocation
    dbo.JobApplication
    dbo.JobOfferNegotiation
    dbo.JobOpening
    dbo.Onboarding
    dbo.Reimbursement
  Views
SQLQuery3.sql - DE...tDatabase (sa (55))*
Reimbursement.sql - tDatabase (sa (54))
CarRental.sql - DES...ntDatabase (sa (53))
Insert Into JobOfferNegotiation
([NegotiationID],[CandidateID],[NegotitationStatus])
values
(650,900,'under negotiation')

100 %
Messages
(1 row affected)

Completion time: 2020-05-02T02:30:45.4989852-07:00
```

Ready Type here to search Ln 4 Col 49 Ch 40 INS 2:30 AM 5/2/2020



SQLQuery2.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (54)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query Execute
XYZRecruitmentDatabase Candidates
Object Explorer
Connect
SQLQuery2.sql - DE...tDatabase (sa (54))*
CarRental.sql - DES...ntDatabase (sa (53))
Insert Into Reimbursement
([ReimbursementID],[InterviewID],[CandidateID],[RentalID],[TicketID],[ReservationID],[ReimbursementAmount])
values
(550,1,900,450,150,350,0.0),
(551,2,901,451,151,351,280.0),
(552,3,902,452,152,351,0.0),
(553,4,903,452,153,352,245.0),
(554,5,904,453,154,353,0.0)
```

100 % Messages
(5 rows affected)
Completion time: 2020-05-02T02:30:19.2109409-07:00

100 % 100 % 100 %

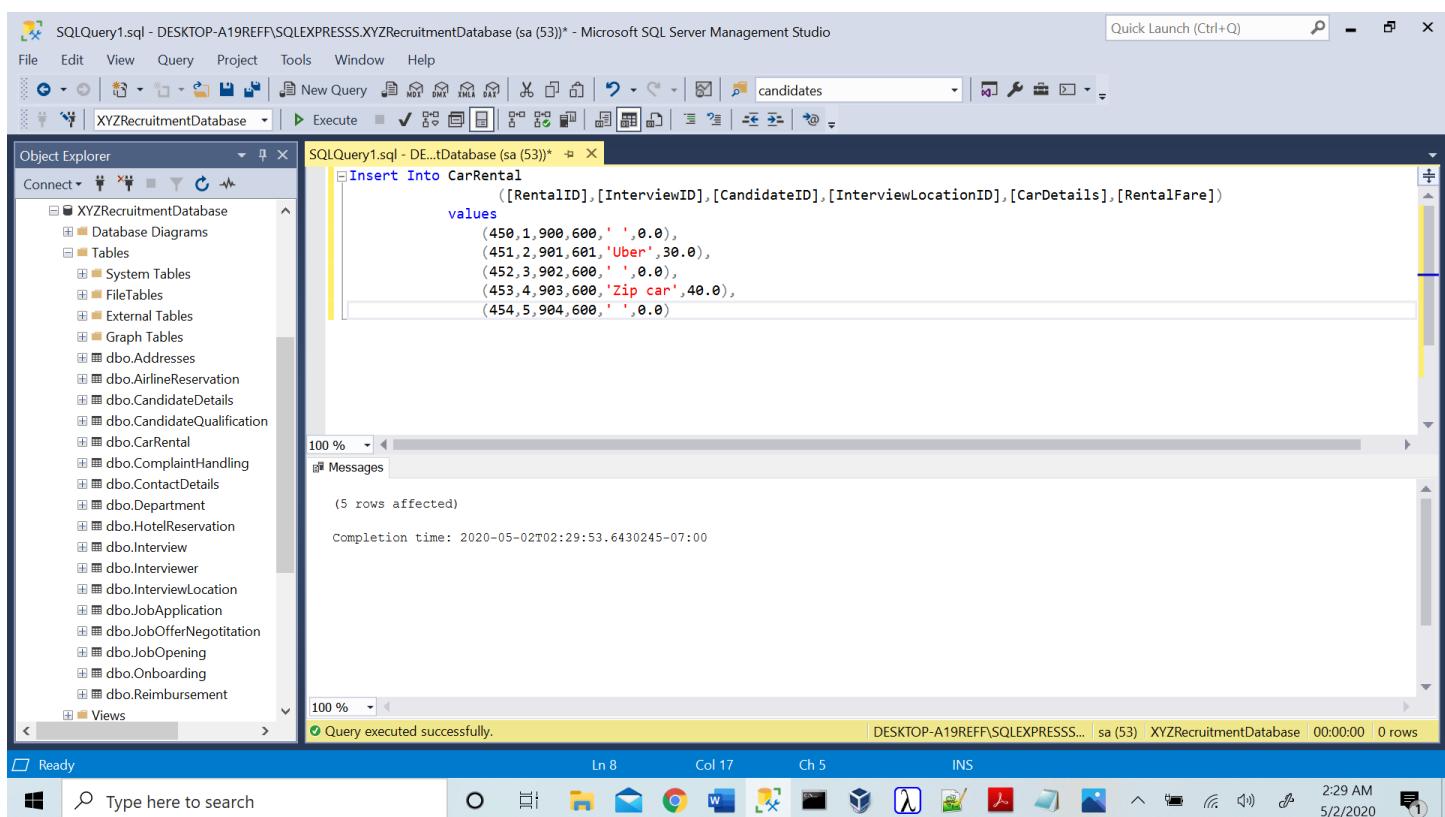
Query executed successfully.

DESKTOP-A19REFF\SQLEXPRESSS... sa (54) XYZRecruitmentDatabase 00:00:00 0 rows

Ready Type here to search

Ln 8 Col 44 Ch 32 INS

2:30 AM 5/2/2020



SQLQuery1.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (53)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query Execute
XYZRecruitmentDatabase Candidates
Object Explorer
Connect
SQLQuery1.sql - DE...tDatabase (sa (53))*
Insert Into CarRental
([RentalID],[InterviewID],[CandidateID],[InterviewLocationID],[CarDetails],[RentalFare])
values
(450,1,900,600,' ',0.0),
(451,2,901,601,'Uber',30.0),
(452,3,902,600,' ',0.0),
(453,4,903,600,'Zip car',40.0),
(454,5,904,600,' ',0.0)
```

100 % Messages
(5 rows affected)
Completion time: 2020-05-02T02:29:53.6430245-07:00

100 % 100 % 100 %

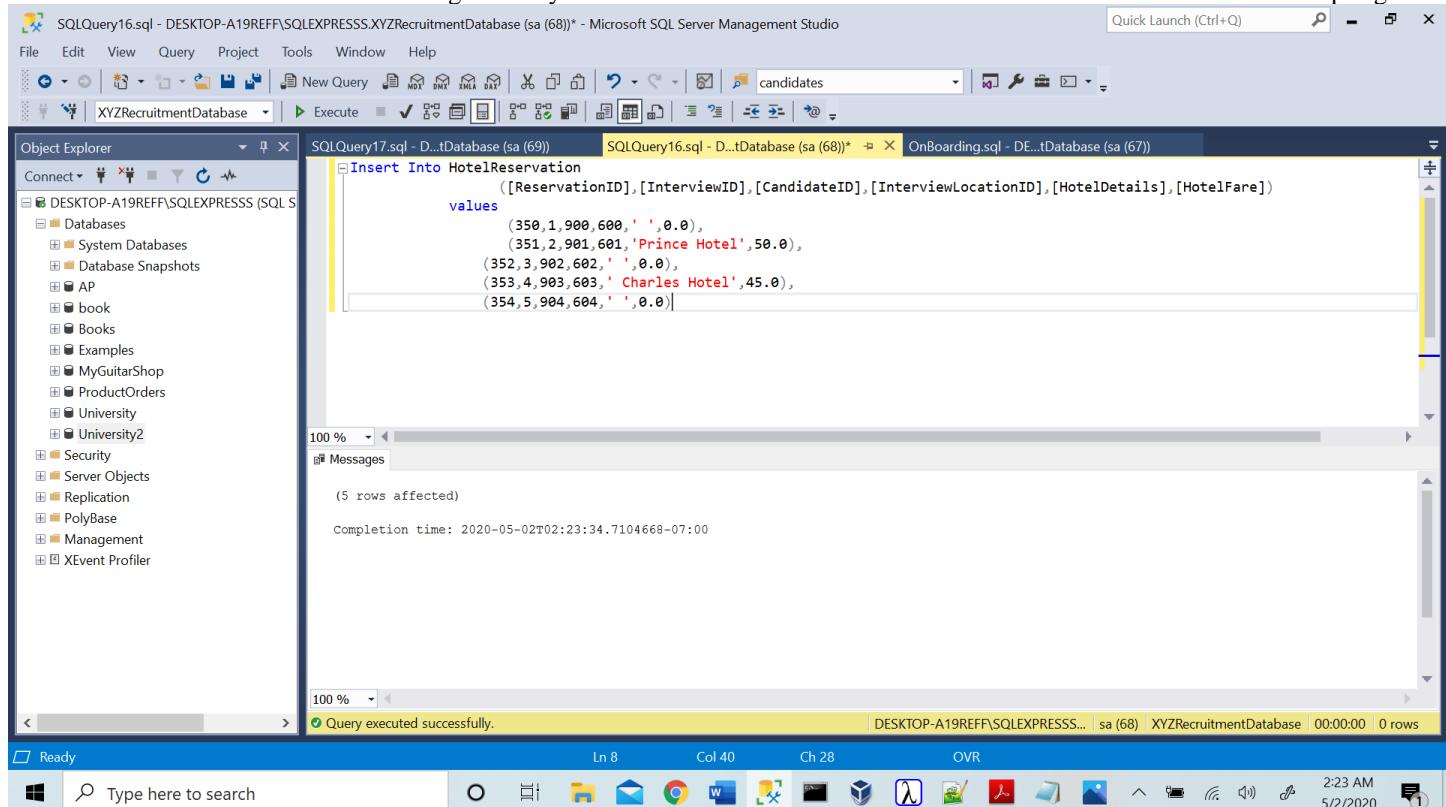
Query executed successfully.

DESKTOP-A19REFF\SQLEXPRESSS... sa (53) XYZRecruitmentDatabase 00:00:00 0 rows

Ready Type here to search

Ln 8 Col 17 Ch 5 INS

2:29 AM 5/2/2020

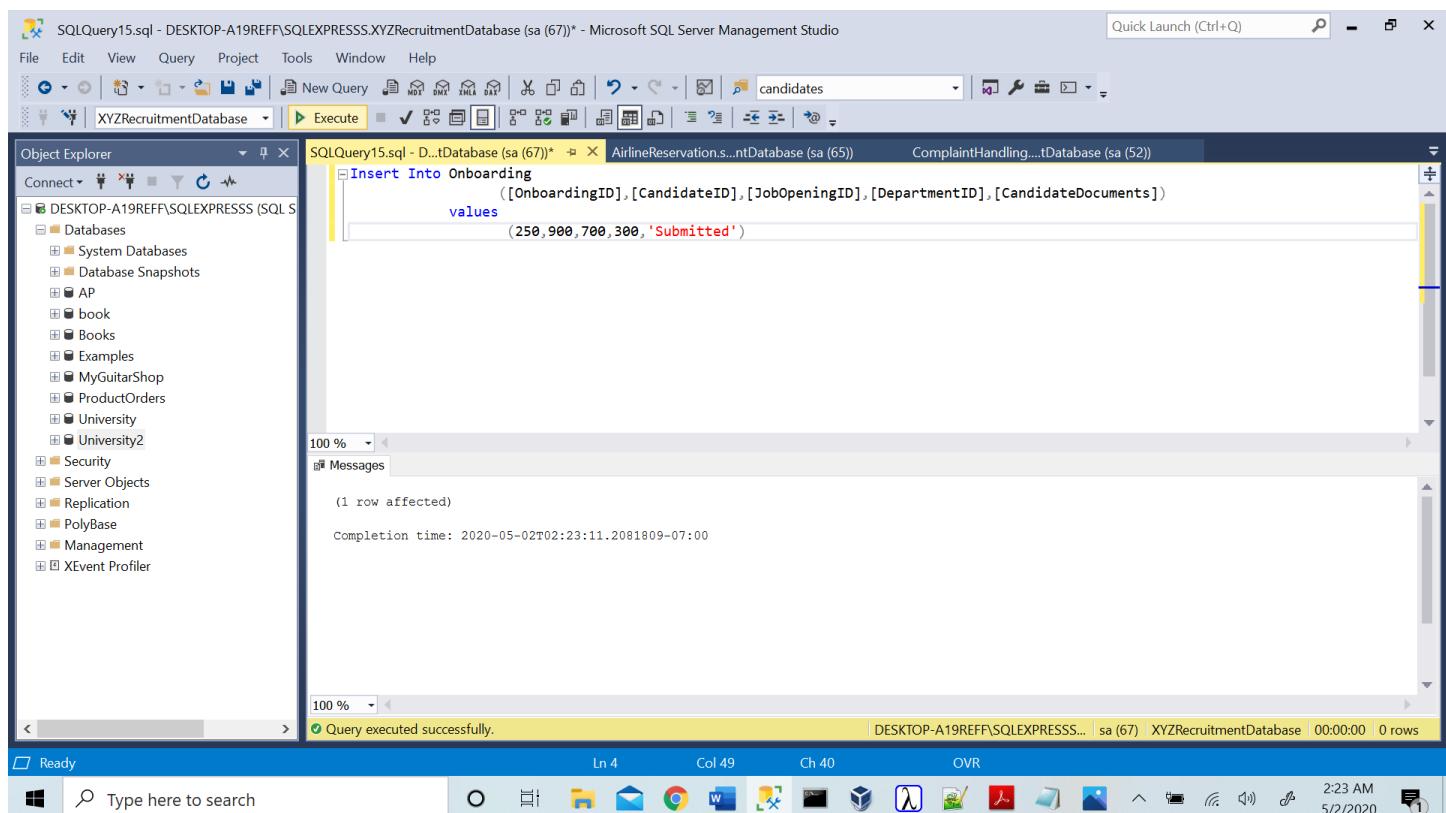


SQLQuery16.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (68)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query MDI DML XML XMLEXEC Execute
XYZRecruitmentDatabase Object Explorer Candidates
SQLQuery17.sql - D...tDatabase (sa (69)) SQLQuery16.sql - D...tDatabase (sa (68)) OnBoarding.sql - DE...tDatabase (sa (67))
Insert Into HotelReservation
([ReservationID],[InterviewID],[CandidateID],[InterviewLocationID],[HotelDetails],[HotelFare])
values
(350,1,900,600,' ',0.0),
(351,2,901,601,'Prince Hotel',50.0),
(352,3,902,602,' ',0.0),
(353,4,903,603,' Charles Hotel',45.0),
(354,5,904,604,' ',0.0)|
```

100 % Messages
(5 rows affected)
Completion time: 2020-05-02T02:23:34.7104668-07:00

Ready Type here to search Ln 8 Col 40 Ch 28 OVR 2:23 AM 5/2/2020

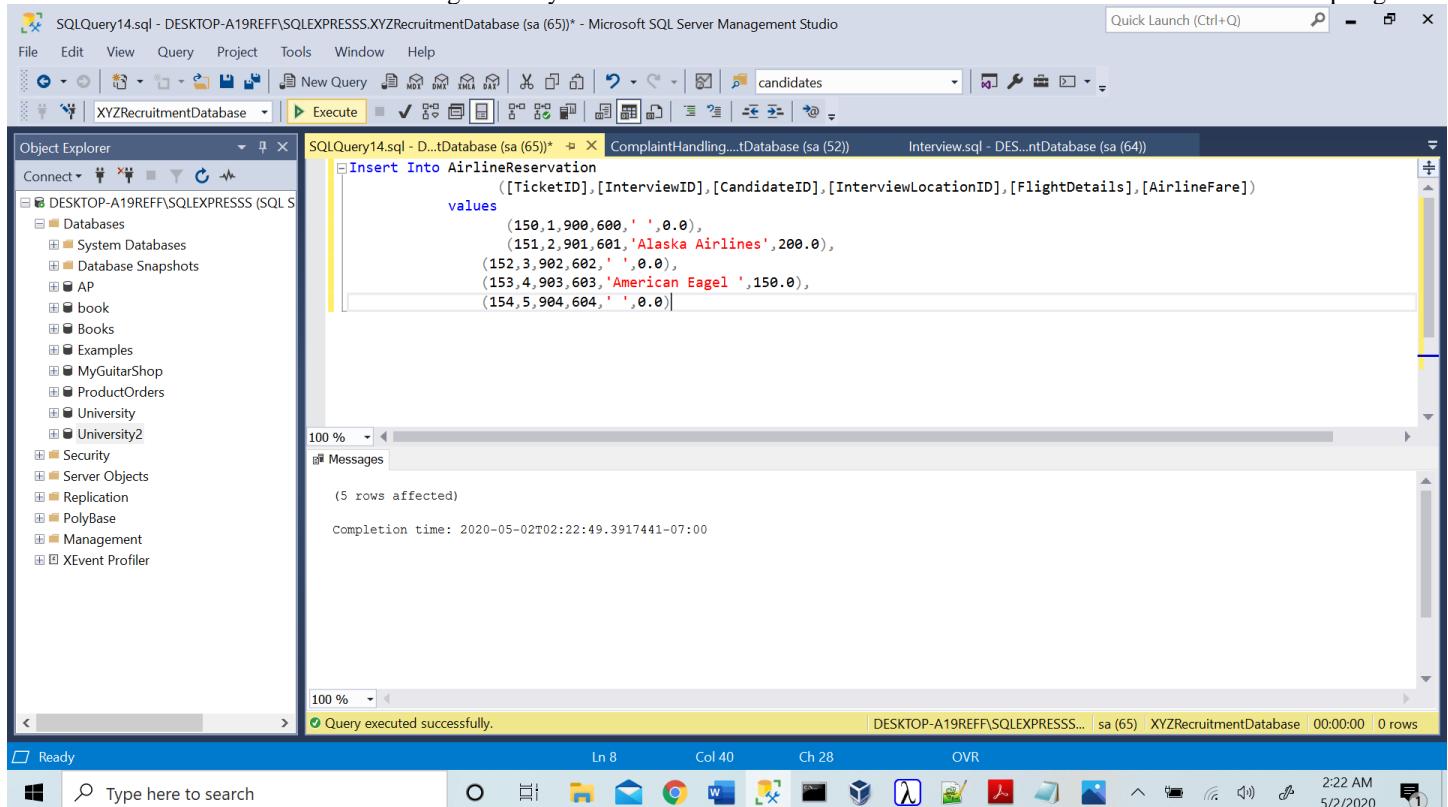


SQLQuery15.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (67)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query MDI DML XML XMLEXEC Execute
XYZRecruitmentDatabase Object Explorer Candidates
SQLQuery15.sql - D...tDatabase (sa (67)) AirlineReservation.s...ntDatabase (sa (65)) ComplaintHandling...tDatabase (sa (52))
Insert Into Onboarding
([OnboardingID],[CandidateID],[JobOpeningID],[DepartmentID],[CandidateDocuments])
values
(250,900,700,300,'Submitted')|
```

100 % Messages
(1 row affected)
Completion time: 2020-05-02T02:23:11.2081809-07:00

Ready Type here to search Ln 4 Col 49 Ch 40 OVR 2:23 AM 5/2/2020

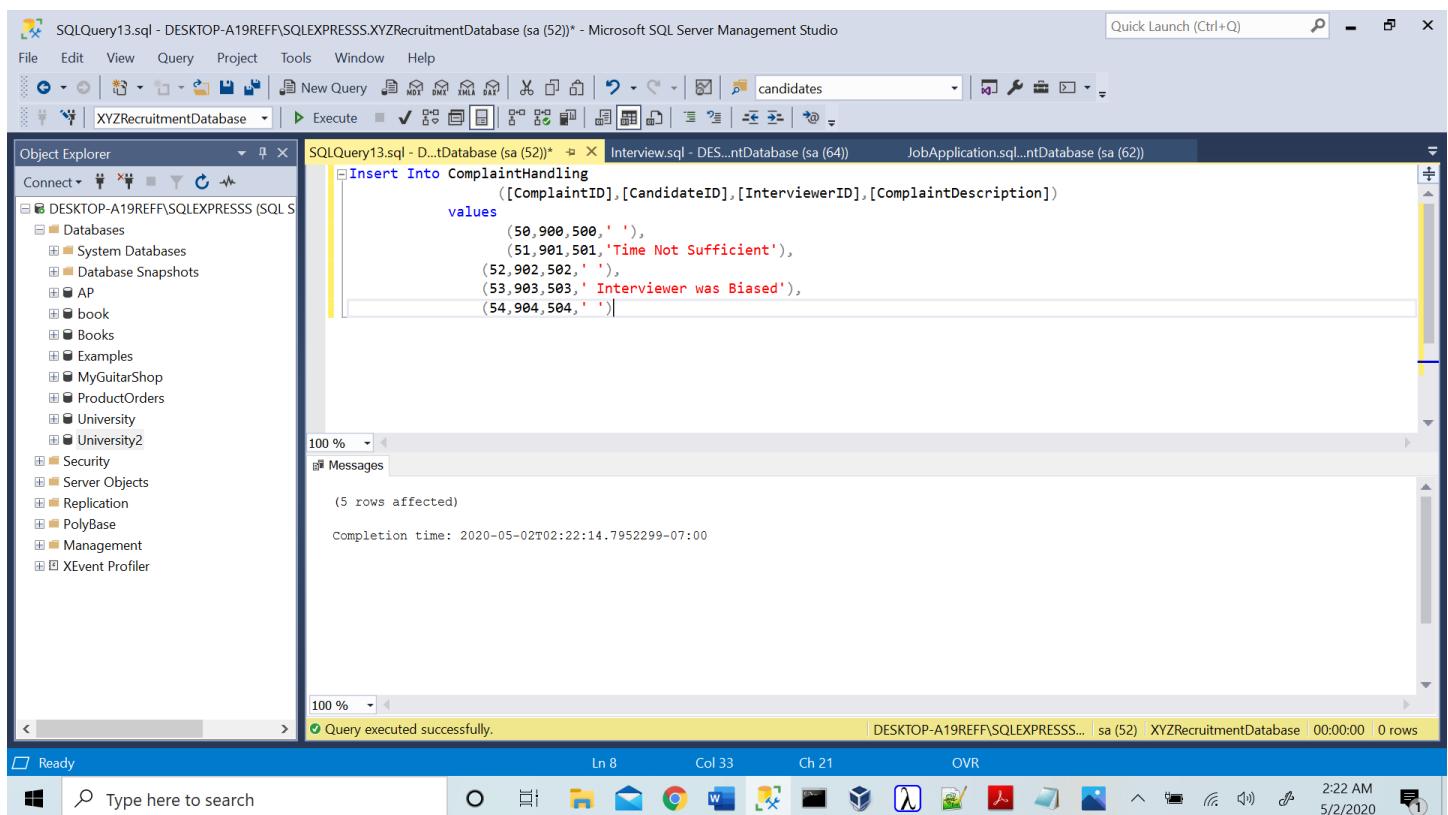


SQLQuery14.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (65)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query MDW BME XML Execute
XYZRecruitmentDatabase Candidates
Object Explorer
Connect Databases System Databases Database Snapshots AP book Books Examples MyGuitarShop ProductOrders University University2 Security Server Objects Replication PolyBase Management XEvent Profiler
SQLQuery14.sql - D...tDatabase (sa (65)) * > X ComplaintHandling....tDatabase (sa (52)) Interview.sql - DES...ntDatabase (sa (64))
Insert Into AirlineReservation
([TicketID],[InterviewID],[CandidateID],[InterviewLocationID],[FlightDetails].[AirlineFare])
values
(150,1,900,600,' ',0.0),
(151,2,901,601,'Alaska Airlines',200.0),
(152,3,902,602,' ',0.0),
(153,4,903,603,'American Eagle ',150.0),
(154,5,904,604,' ',0.0)|
```

100 % Messages
(5 rows affected)
Completion time: 2020-05-02T02:22:49.3917441-07:00

Ready Type here to search Ln 8 Col 40 Ch 28 OVR 2:22 AM 5/2/2020



SQLQuery13.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (52)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query MDW BME XML Execute
XYZRecruitmentDatabase Candidates
Object Explorer
Connect Databases System Databases Database Snapshots AP book Books Examples MyGuitarShop ProductOrders University University2 Security Server Objects Replication PolyBase Management XEvent Profiler
SQLQuery13.sql - D...tDatabase (sa (52)) * > X Interview.sql - DES...ntDatabase (sa (64)) JobApplication.sql...ntDatabase (sa (62))
Insert Into ComplaintHandling
([ComplaintID],[CandidateID],[InterviewerID],[ComplaintDescription])
values
(50,900,500,' '),
(51,901,501,'Time Not Sufficient'),
(52,902,502,' '),
(53,903,503,'Interviewer was Biased'),
(54,904,504,' ')|
```

100 % Messages
(5 rows affected)
Completion time: 2020-05-02T02:22:14.7952299-07:00

Ready Type here to search Ln 8 Col 33 Ch 21 OVR 2:22 AM 5/2/2020

4.2 VIEW

Views in SQL are virtual tables. A view also has rows and columns as they are in a real table in the database. We can create a view by selecting fields from one or more tables present in the database. A View can either have all the rows of a table or specific rows based on certain condition. We create a view to show current status of the candidate interview . For this we have view that joins CandidateDetails table and JobApplication table and return the candidate name and his/her job status.

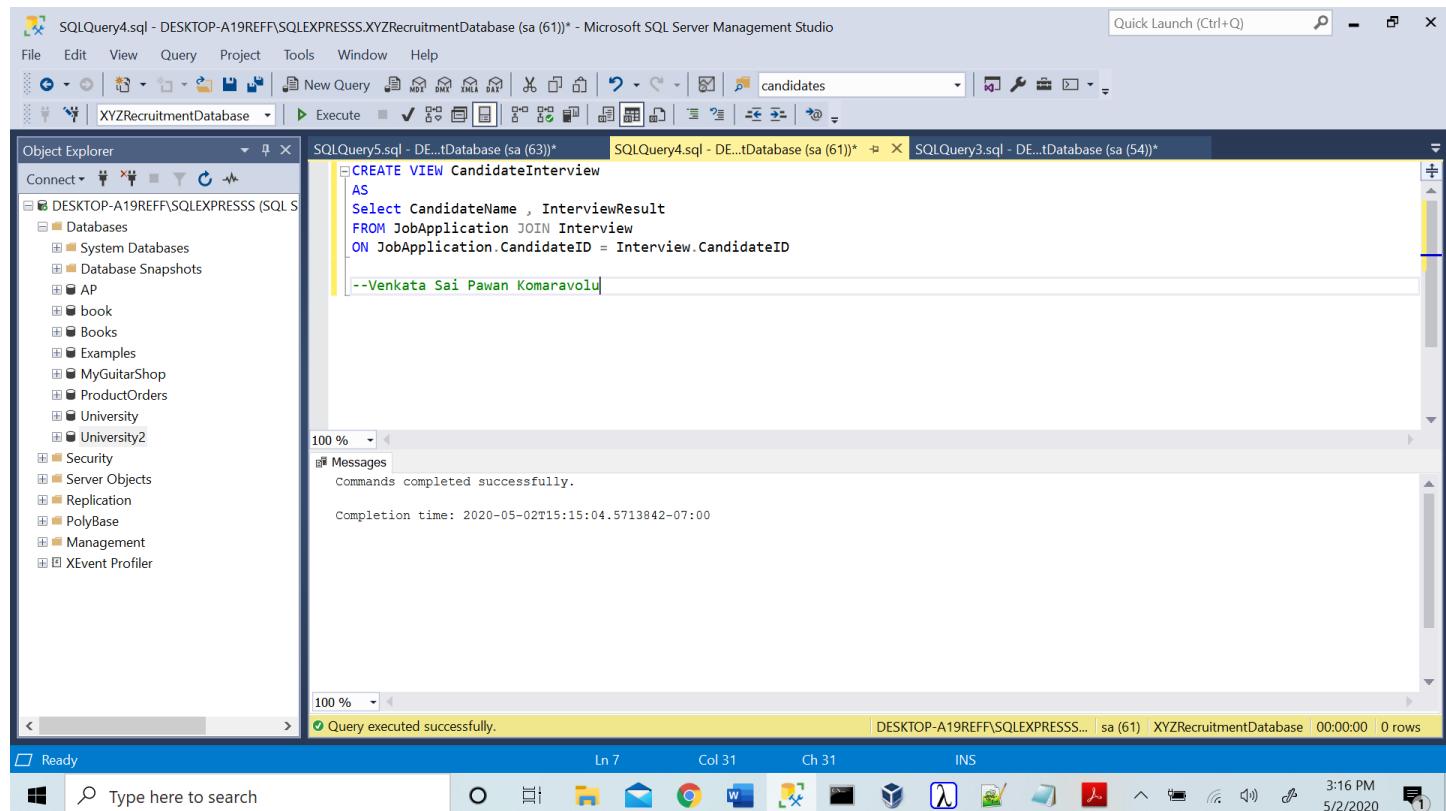
1st view : CREATE VIEW CandidateInterview

AS

Select CandidateName , InterviewResult

FROM JobApplication JOIN Interview

ON JobApplication.CandidateID = Interview.CandidateID



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the 'XYZRecruitmentDatabase'. The central pane displays a query window with the following SQL code:

```

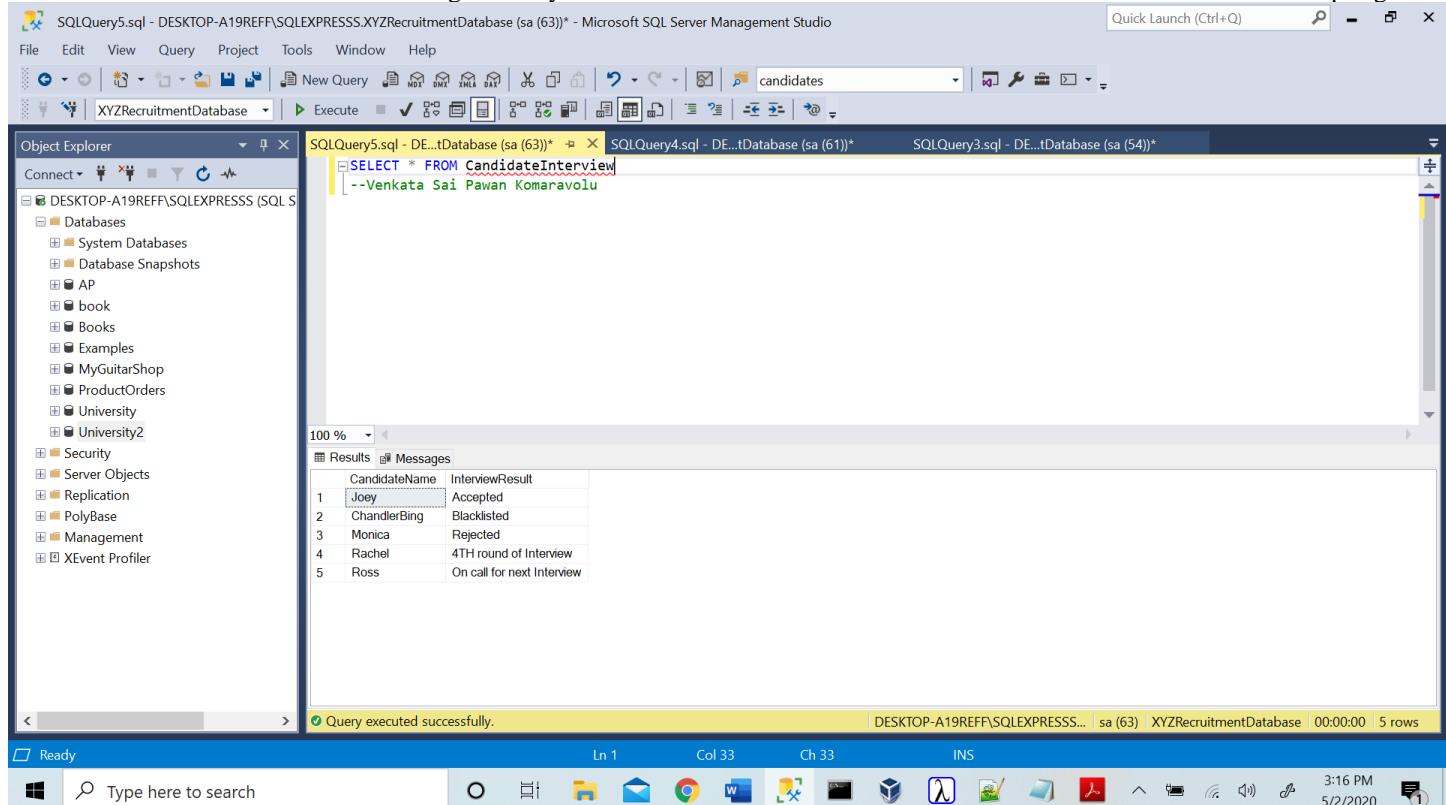
CREATE VIEW CandidateInterview
AS
Select CandidateName , InterviewResult
FROM JobApplication JOIN Interview
ON JobApplication.CandidateID = Interview.CandidateID
--Venkata Sai Pawan Komaravolu

```

The status bar at the bottom indicates "Query executed successfully." and "Completion time: 2020-05-02T15:15:04.5713842-07:00". The taskbar at the bottom right shows the date and time as "5/2/2020 3:16 PM".

We now write select statement for selecting everything from the view

Select * from CandidateInterview



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, there are several databases listed under 'DESKTOP-A19REFF\SQLEXPRESSS (SQL Server)'. In the center pane, a query window displays the following SQL code:

```
SELECT * FROM CandidateInterview
```

The results grid shows the following data:

CandidateName	InterviewResult
Joey	Accepted
ChandlerBing	Blacklisted
Monica	Rejected
Rachel	4TH round of Interview
Ross	On call for next Interview

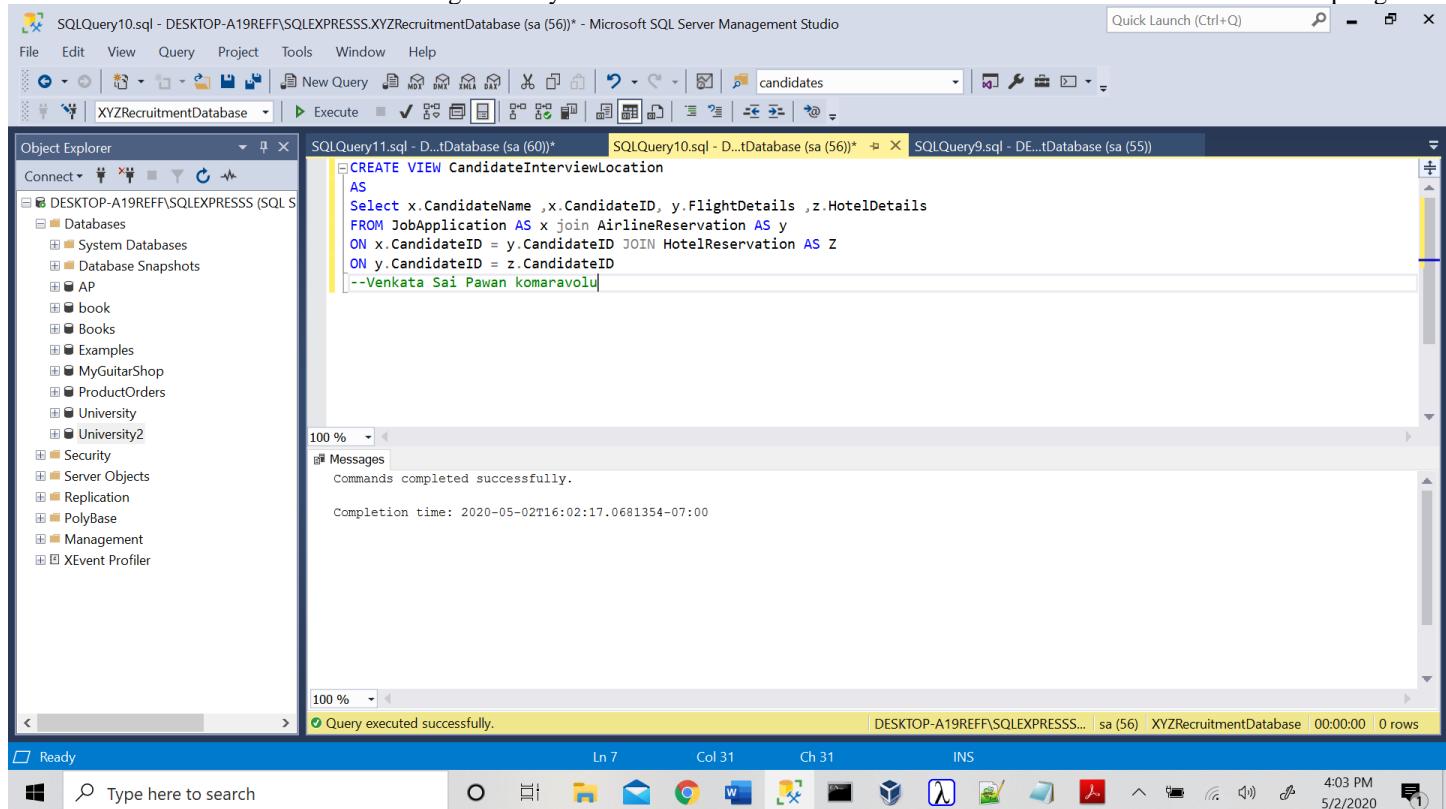
At the bottom of the screen, the taskbar shows the date and time as 3:16 PM 5/2/2020.

We create another view to get the interview location of the candidate which is onsite and all details of the travel i.e. Airline details, Hotels details and car rentals details are combined into a table using views. We create view named CandidateInterviewLocation and get CandidateName, CandidateID , Airline details, hotel details and car rental details using join and on common CandidateID.

2nd view: CREATE VIEW CandidateInterviewLocation

AS

```
Select x.CandidateName ,x.CandidateID, y.FlightDetails ,z.HotelDetails
FROM JobApplication AS x join AirlineReservation AS y
ON x.CandidateID = y.CandidateID JOIN HotelReservation AS Z
ON y.CandidateID = z.CandidateID
```



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'XYZRecruitmentDatabase' is selected. In the center pane, a query window titled 'SQLQuery11.sql - D...tDatabase (sa (60))*' contains the following SQL code:

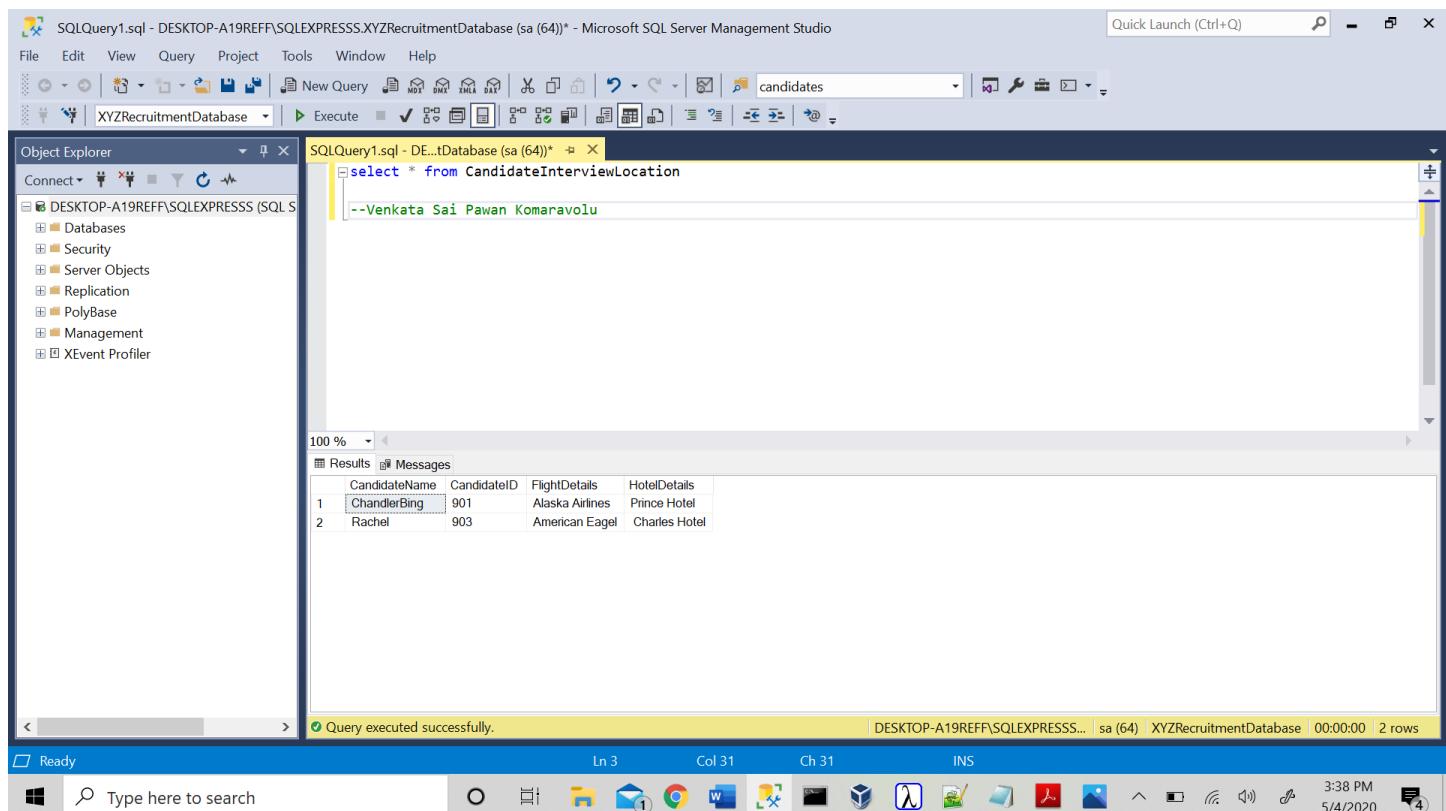
```

CREATE VIEW CandidateInterviewLocation
AS
Select x.CandidateName ,x.CandidateID ,y.FlightDetails ,z.HotelDetails
FROM JobApplication AS x join AirlineReservation AS y
ON x.CandidateID = y.CandidateID JOIN HotelReservation AS z
ON y.CandidateID = z.CandidateID
--Venkata Sai Pawan komarovolu

```

The 'Messages' pane below the query window displays the message: "Commands completed successfully." and "Completion time: 2020-05-02T16:02:17.0681354-07:00". The status bar at the bottom indicates "Query executed successfully.", "DESKTOP-A19REFF\SQLEXPRESSS... sa (64) XYZRecruitmentDatabase | 00:00:00 | 0 rows", and the current date and time as "5/2/2020 4:03 PM".

We display results of this view



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'XYZRecruitmentDatabase' is selected. In the center pane, a query window titled 'SQLQuery1.sql - DE...tDatabase (sa (64))*' contains the following SQL code:

```

select * from CandidateInterviewLocation
--Venkata Sai Pawan Komarovolu

```

The 'Results' pane below the query window displays the following data:

	CandidateName	CandidateID	FlightDetails	HotelDetails
1	ChandlerBing	901	Alaska Airlines	Prince Hotel
2	Rachel	903	American Eagle	Charles Hotel

The 'Messages' pane below the query window displays the message: "Commands completed successfully." and "Completion time: 2020-05-04T16:02:17.0681354-07:00". The status bar at the bottom indicates "Query executed successfully.", "DESKTOP-A19REFF\SQLEXPRESSS... sa (64) XYZRecruitmentDatabase | 00:00:00 | 2 rows", and the current date and time as "5/4/2020 3:38 PM".

We create another view to get the details of the reimbursement amount from each of the tables: Airlines, hotel and rental on common CadidateID.

AS

```
Select x.CandidateID , x.AirlineFare ,y.HotelFare , Z.RentalFAre  
FROM AirlineReservation AS x JOIN HotelReservation AS y  
on x.CandidateID = y.CandidateID JOIN CarRental as z  
on y.CandidateID = z.CandidateID
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several databases, including DESKTOP-A19REFF\SQLEXPRESSS (SQL Server). The central pane displays a query window with the following SQL code:

```
CREATE VIEW ReimbursementAmount  
AS  
Select x.CandidateID , x.AirlineFare ,y.HotelFare , Z.RentalFAre  
FROM AirlineReservation AS x JOIN HotelReservation AS y  
on x.CandidateID = y.CandidateID JOIN CarRental as z  
on y.CandidateID = z.CandidateID
```

Below the code, a message indicates successful execution:

--Venkata Sai Pawan Komaravolu

Messages

Commands completed successfully.

Completion time: 2020-05-02T16:16:34.3091879-07:00

The status bar at the bottom shows "Query executed successfully." and other details like the session ID and date.

We now get the result of candidateID and the fares which need to be reimbursed

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'XYZRecruitmentDatabase' is selected. In the center pane, a query window displays the following SQL code:

```
Select * from ReimbursementAmount;
```

The results pane shows the following data:

	CandidateID	AirlineFare	HotelFare	RentalFare
1	901	200.00	50.00	30.00
2	903	150.00	45.00	40.00

At the bottom of the screen, the taskbar shows the following icons: Start button, Task View, File Explorer, Mail, Google Chrome, Word, File Explorer, Task View, and a search bar. The system tray indicates the date and time as 4:17 PM, 5/2/2020.

We create another view for getting the complaint status for candidate and interviewer with complaint description. We get the CandidateID, InterviewerID and the complaint description.

4th view : CREATE VIEW ComplaintCandidate

AS

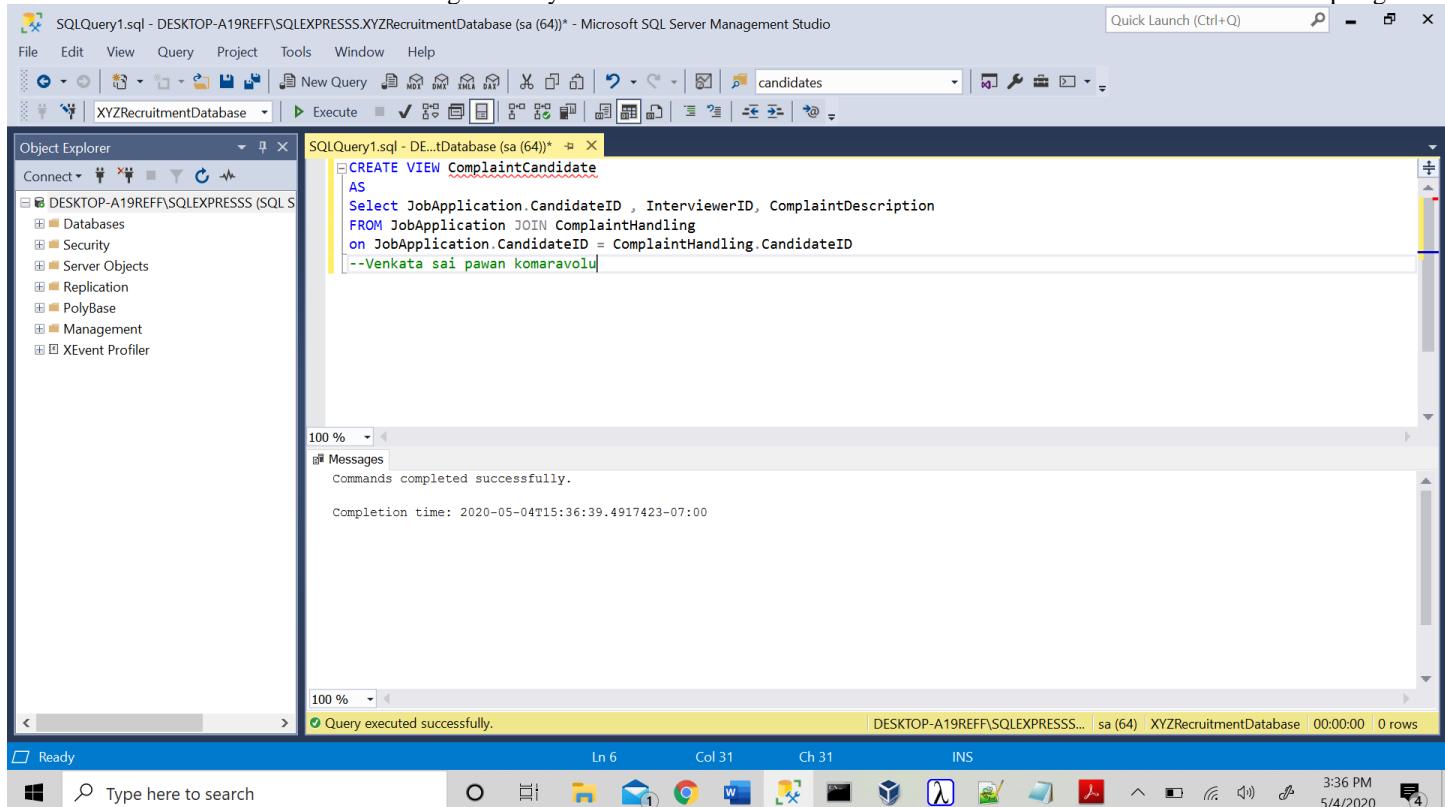
Select JobApplication.CandidateID , InterviewerID, ComplaintDescription

FROM JobApplication JOIN ComplaintHandling

on JobApplication.CandidateID = ComplaintHandling.CandidateID

Select * from ComplaintCandidate ;

Select * from CandidateInterviewLocation



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'XYZRecruitmentDatabase' is selected. In the center pane, a query window titled 'SQLQuery1.sql - DE...tDatabase (sa (64))' contains the following SQL code:

```

CREATE VIEW ComplaintCandidate
AS
Select JobApplication.CandidateID , InterviewerID, ComplaintDescription
FROM JobApplication JOIN ComplaintHandling
on JobApplication.CandidateID = ComplaintHandling.CandidateID
--Venkata sai pawan komaravolu

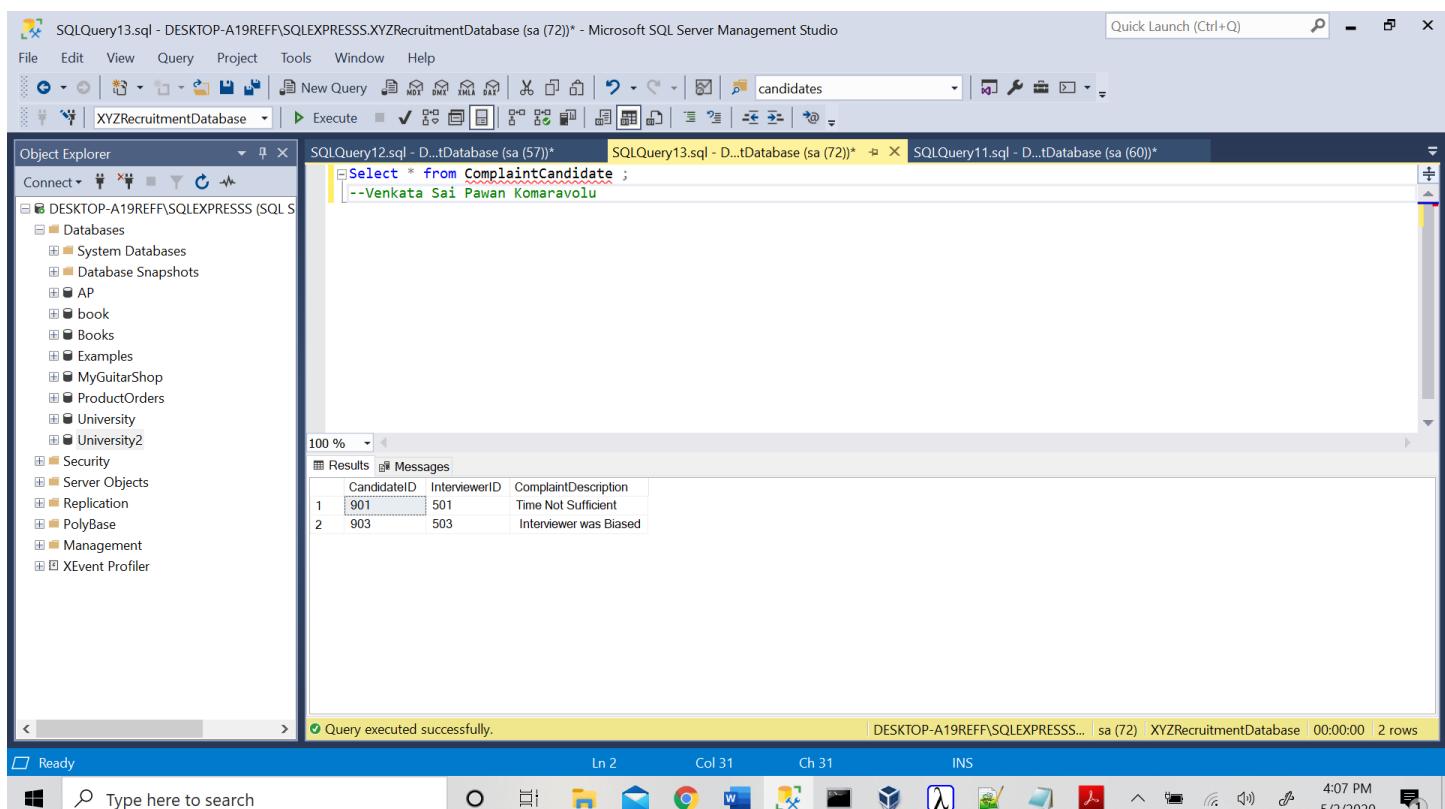
```

The 'Messages' pane below the query window shows the execution results:

- Commands completed successfully.
- Completion time: 2020-05-04T15:36:39.4917423-07:00

The status bar at the bottom indicates: DESKTOP-A19REFF\SQLEXPRESSS... sa (64) XYZRecruitmentDatabase | 00:00:00 | 0 rows.

We select everything from the ComplaintCandidate view.



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'XYZRecruitmentDatabase' is selected. In the center pane, a query window titled 'SQLQuery13.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (72))' contains the following SQL code:

```

Select * from ComplaintCandidate ;
--Venkata Sai Pawan Komaravolu

```

The 'Results' pane below the query window displays the following data:

CandidateID	InterviewerID	ComplaintDescription
901	501	Time Not Sufficient
903	503	Interviewer was Biased

The status bar at the bottom indicates: DESKTOP-A19REFF\SQLEXPRESSS... sa (72) XYZRecruitmentDatabase | 00:00:00 | 2 rows.

We create another view to get the contact details of the candidate for this we join ContactDetails table and CandidateDetails table on common CandidateDetailsID.

5th view: Create View CandidateConatact

AS

Select CandidateName , PhoneNumber,EmailAddress

from CandidateDetails join ContactDetails

on CandidateDetails.ContactID = ContactDetails.ContactID

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays the database structure of 'DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase'. In the center, the 'SQLQuery17.sql - D...tDatabase (sa (78))' window contains the T-SQL code for creating a view:

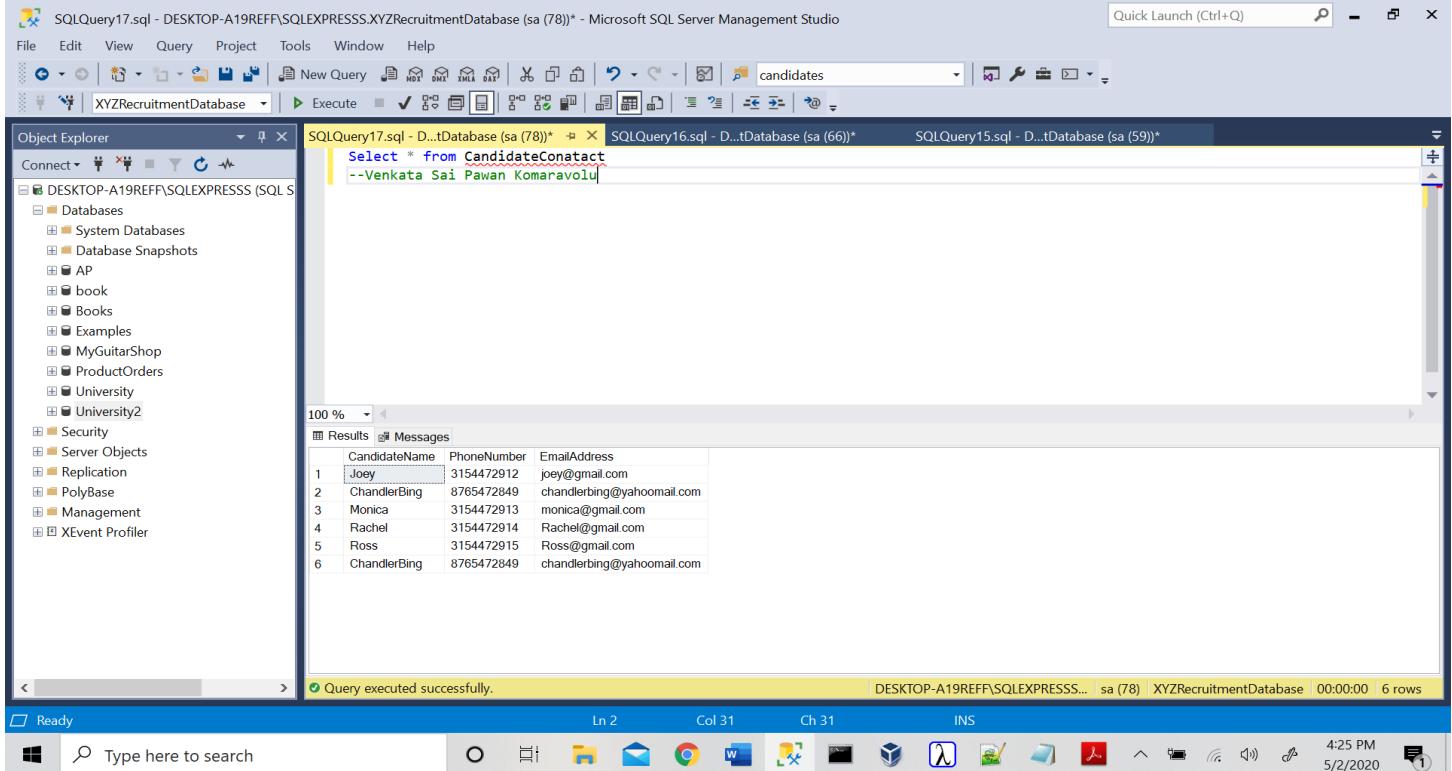
```
Create View CandidateConatact
AS
Select CandidateName , PhoneNumber,EmailAddress
from CandidateDetails join ContactDetails
on CandidateDetails.ContactID = ContactDetails.ContactID
--Venkata Sai Pawan Komaravolu
```

Below the code, the 'Messages' pane shows the execution results:

Commands completed successfully.
Completion time: 2020-05-02T16:25:09.6128524-07:00

At the bottom, the status bar indicates: DESKTOP-A19REFF\SQLEXPRESSS... sa (66) | XYZRecruitmentDatabase | 00:00:00 | 0 rows

We select everything from CandidateContact view



4.3 STORED PROCEDURE

A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again. So if you have an SQL query that you write over and over again, save it as a stored procedure, and then just call it to execute it. You can also pass parameters to a stored procedure, so that the stored procedure can act based on the parameter value(s) that is passed. We create a stored procedure to get the reimbursed amount for a candidate. If we give the procedure name and name of the candidate, we get the reimbursed amount for that candidate.

```
1st Procedure : CREATE PROC ReimbursedAmount @CandidateName VARCHAR (50) = NULL
AS
SELECT x.CandidateID, x.CandidateName , y.InterviewID , y.ReimbursementAmount
FROM Candidate AS x join Reimbursement AS y
on x.CandidateID = y.CandidateID
WHERE CandidateName = @CandidateName;
```

We execute the procedure using the query

EXEC ReimbursedAmount @ CandidateName = 'ChandlerBing' will give results of the reimbursement amount for the CandidateName ChandlerBing.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'master' is selected. In the center pane, a script window titled 'SQLQuery2.sql' contains the following T-SQL code:

```

CREATE PROC ReimburAmount @CandidateName VARCHAR (50) = NULL
AS
SELECT x.CandidateID, x.CandidateName , y.InterviewID , y.ReimbursementAmount
FROM JobApplication AS x join Reimbursement AS y
on x.CandidateID = y.CandidateID
WHERE CandidateName = @CandidateName;
--Venkata Sai Pawan Komaravolu

```

The 'Messages' tab at the bottom shows the execution results:

```

Commands completed successfully.

Completion time: 2020-05-02T16:44:23.8758903-07:00

```

The status bar at the bottom right indicates: DESKTOP-A19REFF\SQLEXPRESSSS... | sa (53) | master | 00:00:00 | 0 rows.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'XYZRecruitmentDatabase' is selected. In the center pane, a script window titled 'SQLQuery2.sql' contains the following T-SQL code:

```

EXEC ReimburAmount @CandidateName='ChandlerBing';

```

The 'Results' tab at the bottom shows the output of the query:

CandidateID	CandidateName	InterviewID	ReimbursementAmount	
1	901	ChandlerBing	2	280.00
1	900	Joey	1	0.00
1	900	Joey	1	0.00
1	900	Joey	1	0.00
1	900	Joey	1	0.00
1	900	Joey	1	0.00
1	900	Joey	1	0.00

The status bar at the bottom right indicates: DESKTOP-A19REFF\SQLEXPRESSSS... | sa (54) | XYZRecruitmentDatabase | 00:00:02 | 0 rows.

We create another procedure for the application status. If have application status in stored procedure and when we execute this procedure with candidate name we get the status of the candidate

2nd procedure: CREATE PROC [dbo].[ApplicationStatus] @Status VARCHAR(50) = NULL
AS

SELECT x.CandidateName , y.CandidateID,y.CandidateStatus
FROM CandidateDetails as x JOIN JobApplication as y

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure. The main pane displays a T-SQL script for creating a stored procedure named ApplicationStatus. The script uses a cursor to select candidate details from two tables, CandidateDetails and JobApplication, based on CandidateDetailsID equality and a status filter. A comment at the end credits Venkata Sai Pawan Komaravolu.

```
USE XYZRecruitmentDatabase
GO

CREATE PROC ApplicationStatus @Status VARCHAR(50) = NULL
AS
SELECT x.CandidateName , y.CandidateID,y.CandidateStatus
FROM CandidateDetails as x JOIN JobApplication as y
on x.CandidateDetailsID = y.CandidateDetailsID
where CandidateStatus = @Status
--Venkata Sai Pawan Komaravolu
```

The Messages pane shows the command completed successfully with a completion time of 2020-05-02T16:49:19.7007408-07:00. The status bar at the bottom indicates the query was executed successfully and returned 0 rows.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure. The main pane displays a T-SQL script executing the ApplicationStatus stored procedure with a parameter value of 'Blacklisted'. The results pane shows a single row returned, indicating candidates with a blacklisted status.

```
EXEC ApplicationStatus @Status = 'Blacklisted'
```

CandidateName	CandidateID	CandidateStatus
ChandlerBing	901	Blacklisted

The status bar at the bottom indicates the query was executed successfully and returned 1 row.

We create another procedure to get the status of the negotiation which is still under negotiation

3rd procedure: CREATE PROC [dbo].[NegotiationStatus] @Status VARCHAR(50) = NULL

AS

```
SELECT x.CandidateName ,x.CandidateID,y.NegotitationStatus  
FROM JobApplication as x JOIN JobOfferNegotitation as y  
on x.CandidateID = y.CandidateID  
where NegotitationStatus = @Status  
GO
```

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, a connection to 'DESKTOP-A19REFF\SQLEXPRESSS' is selected. In the center pane, three queries are visible: 'SQLQuery5.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (57))*', 'SQLQuery4.sql - DE...tDatabase (sa (56))*', and 'SQLQuery3.sql - DE...tDatabase (sa (55))'. The 'SQLQuery3.sql' tab is active, displaying the T-SQL code for creating a stored procedure:

```
USE XYZRecruitmentDatabase  
GO  
CREATE PROC NegotiationStatus @Status VARCHAR(50) = NULL  
AS  
SELECT x.CandidateName ,x.CandidateID,y.NegotitationStatus  
FROM JobApplication as x JOIN JobOfferNegotitation as y  
on x.CandidateID = y.CandidateID  
where NegotitationStatus = @Status  
--Venkata Sai Pawan
```

In the 'Messages' pane below, the output is shown:

```
Commands completed successfully.  
Completion time: 2020-05-02T16:53:52.9083101-07:00
```

The status bar at the bottom right indicates: DESKTOP-A19REFF\SQLEXPRESSS... | sa (57) | XYZRecruitmentDatabase | 00:00:00 | 0 rows | 4:54 PM | 5/2/2020 | 1.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'XYZRecruitmentDatabase' is selected. In the center pane, a query window titled 'SQLQuery6.sql' is open, displaying the following T-SQL code:

```
EXEC NegotiationStatus @Status = 'under negotiation'
```

The results pane shows a table with three columns: CandidateName, CandidateID, and NegotiationStatus. The data is as follows:

	CandidateName	CandidateID	NegotiationStatus
1	Joey	900	under negotiation
2	Joey	900	under negotiation

Below the results, a message bar indicates: 'Query executed successfully.' The status bar at the bottom right shows the session details: DESKTOP-A19REFF\SQLEXPRESSSS... | sa (58) | XYZRecruitmentDatabase | 00:00:00 | 2 rows.

4.4 FUNCTIONS

We have 2 types of functions 1. Built in functions such as aggregate functions and 2. UDF : User defined function. Below are the functions used in this project

`SELECT SUM (ReimbursementAmount) from Reimbursement`

We select the sum of the reimbursement amount in the reimbursement table.

```

SQLQuery6.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (61)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
New Query Object Explorer XYZRecruitmentDatabase Execute Quick Launch (Ctrl+Q)
Object Explorer
Connect > DESKTOP-A19REFF\SQLEXPRESSS (SQL Server)
  Databases Security Server Objects Replication PolyBase Management XEvent Profiler
SQLQuery6.sql - DE...tDatabase (sa (61))*
--Venkata Sai Pawan Komaravolu
SELECT SUM ( ReimbursementAmount) from Reimbursement
Results Messages
(No column name)
1 525.00
91 %
Query executed successfully.
DESKTOP-A19REFF\SQLEXPRESSS... sa (61) XYZRecruitmentDatabase | 00:00:00 | 1 rows
Ready Ln 2 Col 31 Ch 31 INS
Type here to search
6:09 PM 5/4/2020

```

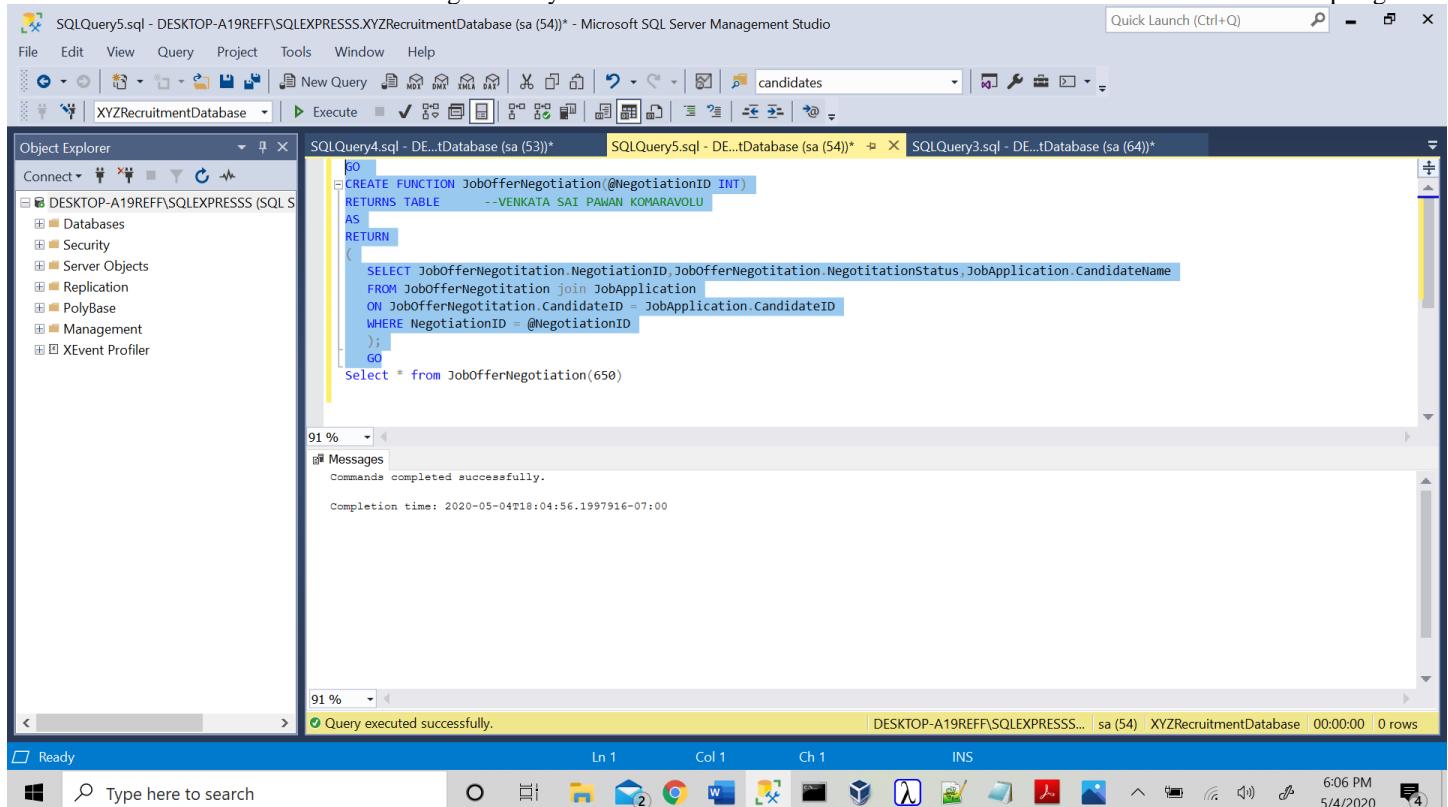
2nd Function :

```

GO
CREATE FUNCTION JobOfferNegotiation(@NegotiationID INT)
RETURNS TABLE
AS
RETURN
(
    SELECT
        JobOfferNegotiation.NegotiationID, JobOfferNegotiation.NegotitationStatus, JobApplication.CandidateName
    FROM JobOfferNegotiation join JobApplication
    ON JobOfferNegotiation.CandidateID = JobApplication.CandidateID
    WHERE NegotiationID = @NegotiationID
);
GO
Select * from JobOfferNegotiation(650)

```

We create a function named JobOfferNegotiation and in this we select NegotiationID, NegotitationStatus from JobOfferNegotiation table and Candidate name from JobApplication table on Common CandidateID and we have NegotiationID as the parameter and the final result we get is the candidate name and his negotiation status from both the tables and we pass the NegotiationID which is 650 in this case.



SQLQuery5.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (54)) - Microsoft SQL Server Management Studio

```

GO
CREATE FUNCTION JobOfferNegotiation(@NegotiationID INT)
RETURNS TABLE
--VENKATA SAI PAWAN KOMARAVOLU
AS
RETURN
(
    SELECT JobOfferNegotiation.NegotiationID, JobOfferNegotiation.NegotitationStatus, JobApplication.CandidateName
    FROM JobOfferNegotiation join JobApplication
    ON JobOfferNegotiation.CandidateID = JobApplication.CandidateID
    WHERE NegotiationID = @NegotiationID
);
GO
Select * from JobOfferNegotiation(650)

```

91 %

Messages

Commands completed successfully.

Completion time: 2020-05-04T18:04:56.1997916-07:00

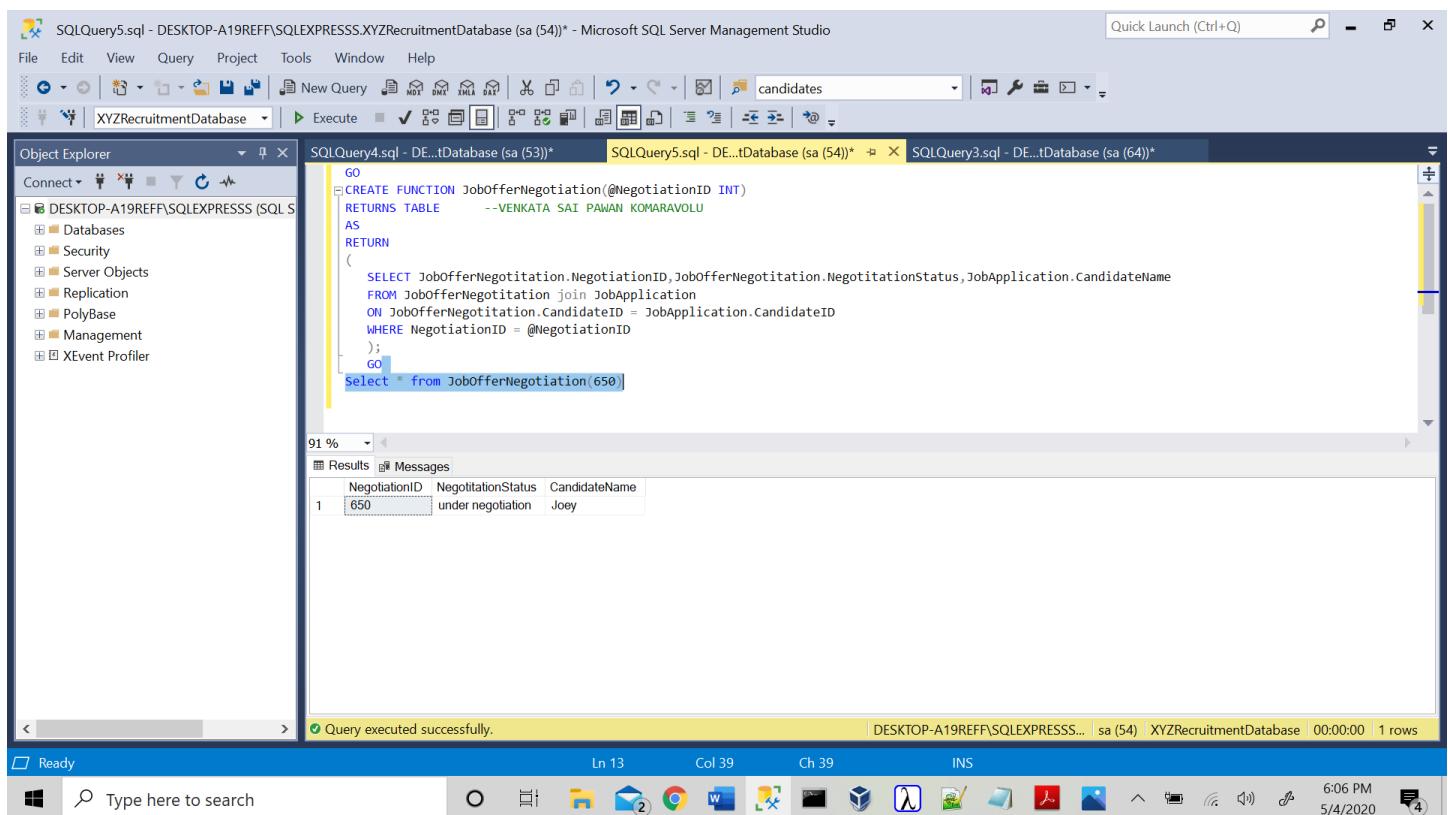
Ready

Type here to search

Ln 1 Col 1 Ch 1 INS

DESKTOP-A19REFF\SQLEXPRESSS... sa (54) XYZRecruitmentDatabase 00:00:00 0 rows

6:06 PM 5/4/2020



SQLQuery5.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (54)) - Microsoft SQL Server Management Studio

```

GO
CREATE FUNCTION JobOfferNegotiation(@NegotiationID INT)
RETURNS TABLE
--VENKATA SAI PAWAN KOMARAVOLU
AS
RETURN
(
    SELECT JobOfferNegotiation.NegotiationID, JobOfferNegotiation.NegotitationStatus, JobApplication.CandidateName
    FROM JobOfferNegotiation join JobApplication
    ON JobOfferNegotiation.CandidateID = JobApplication.CandidateID
    WHERE NegotiationID = @NegotiationID
);
GO
Select * from JobOfferNegotiation(650)

```

91 %

Results

NegotiationID	NegotitationStatus	CandidateName
650	under negotiation	Joey

Messages

Query executed successfully.

DESKTOP-A19REFF\SQLEXPRESSS... sa (54) XYZRecruitmentDatabase 00:00:00 1 rows

Ready

Type here to search

Ln 13 Col 39 Ch 39 INS

DESKTOP-A19REFF\SQLEXPRESSS... sa (54) XYZRecruitmentDatabase 00:00:00 1 rows

6:06 PM 5/4/2020

3rd Function :

We now create a function to get the Onboarding status of the candidate along with his/her name and for this we create the function named OnBoardingDetails and Onboarding ID as the object and returns the table where we have selected OnboardingID , CandidateDocuments and CandidateName and CandidateStatus from the join of

Onboarding and JobApplication table on common CandidateID and later we select everything from this function where Onboarding ID is 250.

GO

```
CREATE FUNCTION OnboardingDetails(@OnboardingID INT)
```

```
RETURNS TABLE
```

```
AS
```

```
RETURN
```

```
(
```

```
    SELECT Onboarding.OnboardingID , Onboarding.CandidateDocuments , JobApplication.CandidateName
```

```
,JobApplication.CandidateStatus
```

```
FROM Onboarding join JobApplication
```

```
ON Onboarding.CandidateID = JobApplication.CandidateID
```

```
WHERE OnboardingID =@OnboardingID
```

```
);
```

```
GO
```

```
SQLQuery1.sql - DESKTOP-A19REFF\SQLEXPRESSSS.XYZRecruitmentDatabase (sa (56)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
New Query MDW BME XML DAX | candidates
XYZRecruitmentDatabase Execute
Object Explorer
Connect ▾
DESKTOP-A19REFF\SQLEXPRESSSS (SQLS
  Databases
  Security
  Server Objects
  Replication
  PolyBase
  Management
  XEvent Profiler
SQLQuery1.sql - DE...tDatabase (sa (56))
GO
CREATE FUNCTION OnboardingDetails(@OnboardingID INT)
RETURNS TABLE --VENKATA SAI PAWAN KOMARAVOLU
AS
RETURN
(
    SELECT Onboarding.OnboardingID , Onboarding.CandidateDocuments , JobApplication.CandidateName ,JobApplication.CandidateStatus
    FROM Onboarding join JobApplication
    ON Onboarding.CandidateID = JobApplication.CandidateID
    WHERE OnboardingID =@OnboardingID
);
GO
SELECT * FROM OnboardingDetails ('250');

100 %
Messages
Commands completed successfully.
Completion time: 2020-05-04T16:29:21.5862590-07:00

100 %
100 %
Ready
Type here to search
Ln 1 Col 1 Ch 1 INS
DESKTOP-A19REFF\SQLEXPRESSSS... sa (56) XYZRecruitmentDatabase 00:00:00 0 rows
4:33 PM
5/4/2020 4
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SQLQuery1.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (56)) - Microsoft SQL Server Management Studio". The main area displays a query window with the following T-SQL code:

```

GO
CREATE FUNCTION OnboardingDetails(@OnboardingID INT)
RETURNS TABLE
--VENKATA SAI PAWAN KOMARAVOLU
AS
RETURN
(
    SELECT Onboarding.OnboardingID , Onboarding.CandidateDocuments , JobApplication.CandidateName ,JobApplication.CandidateStatus
    FROM Onboarding join JobApplication
    ON Onboarding.CandidateID = JobApplication.CandidateID
    WHERE OnboardingID = @OnboardingID
);
GO
SELECT * FROM OnboardingDetails ('250');

```

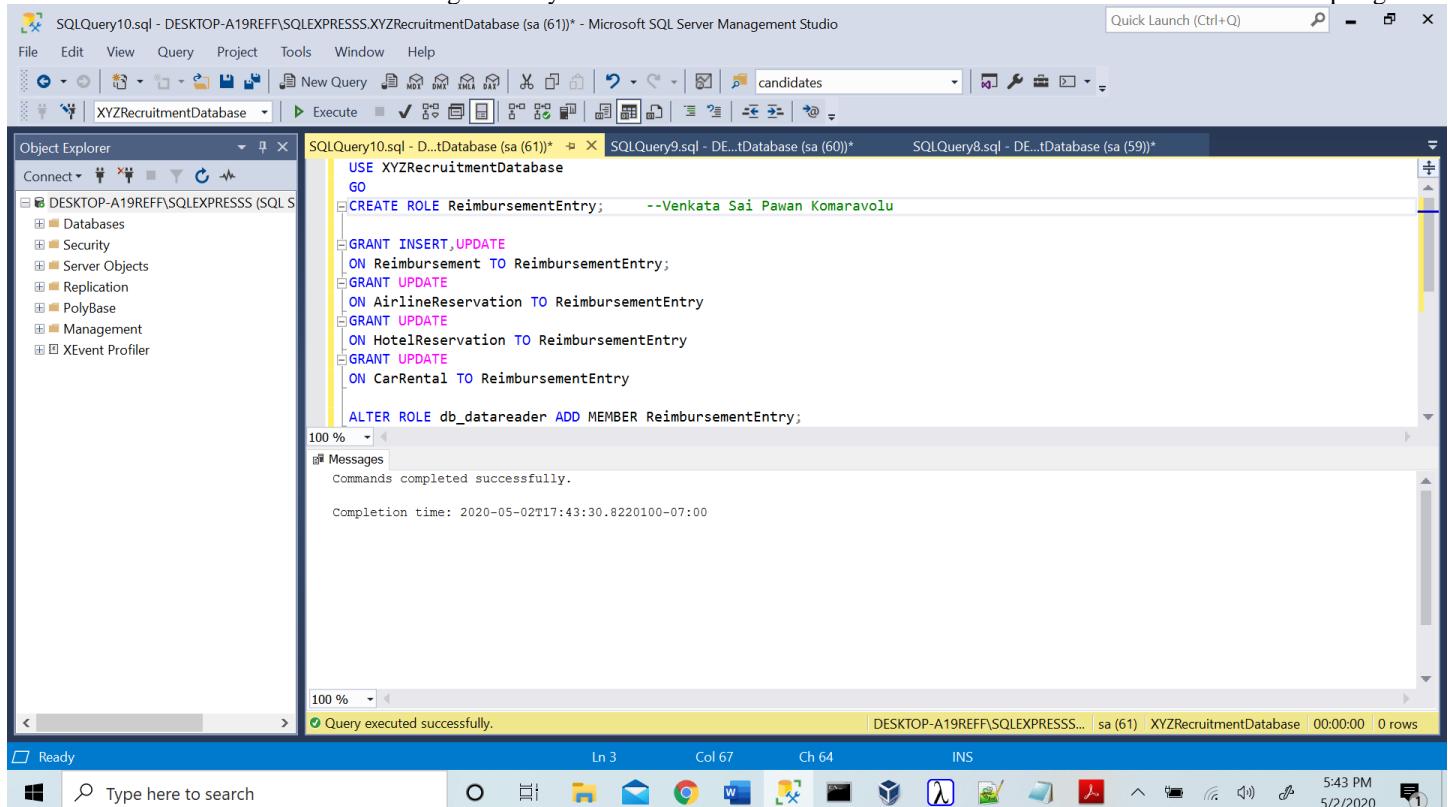
The results pane shows a single row of data:

	OnboardingID	CandidateDocuments	CandidateName	CandidateStatus
1	250	Submitted	Joey	Accepted

At the bottom of the interface, a status bar indicates "Query executed successfully." and "DESKTOP-A19REFF\SQLEXPRESSS... sa (56) XYZRecruitmentDatabase | 00:00:00 | 1 rows". The taskbar at the bottom of the screen shows various application icons.

4.5 SCRIPTS

I've created a role ReiumburseEntry and I've granted INSERT and UPDATE on Reiumbursement table to the roleReiumbursementEntry and also to the role we grant UPDATE permission on the tables AirlineReservation, HotelReservation , CarRental to the role ReiumbursementEntry and later alter the role db_datareader and add the member reiumbursementEntry. I have created a Login named 'Venkata' with password as 'Saipawan@897' and with default database as XYZRecruitmentDatabase. Also I have created a User named 'Phoebe' for Login named 'Venkata' I've created a user for this and for the role. I've added the member Phoebe.



SQLQuery10.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (61)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

XYZRecruitmentDatabase Execute Quick Launch (Ctrl+Q)

Object Explorer

```

USE XYZRecruitmentDatabase
GO
CREATE ROLE ReimbursementEntry; --Venkata Sai Pawan Komaravolu

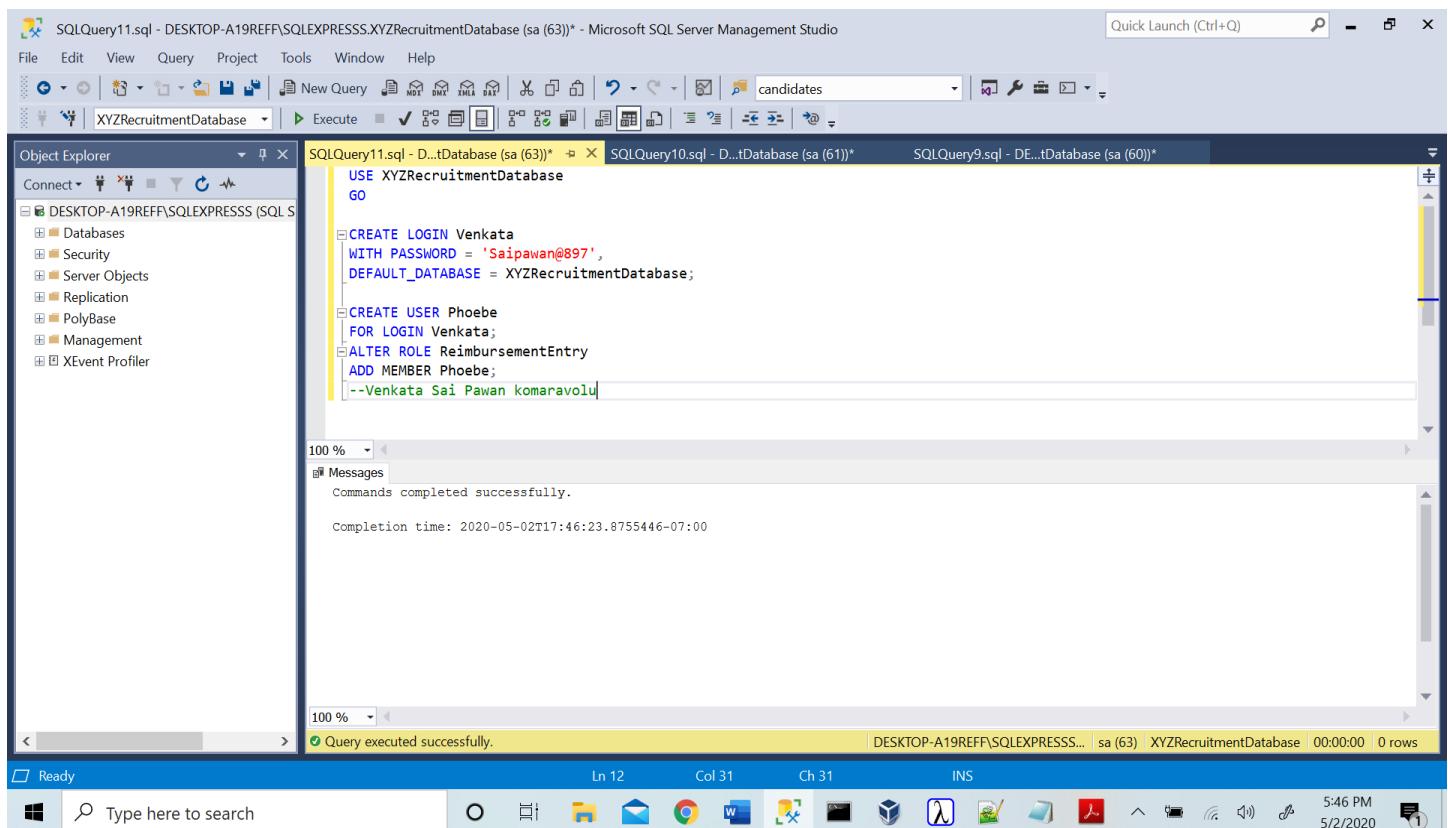
GRANT INSERT,UPDATE
ON Reimbursement TO ReimbursementEntry;
GRANT UPDATE
ON AirlineReservation TO ReimbursementEntry
GRANT UPDATE
ON HotelReservation TO ReimbursementEntry
GRANT UPDATE
ON CarRental TO ReimbursementEntry

ALTER ROLE db_datareader ADD MEMBER ReimbursementEntry;
  
```

100 % Messages Commands completed successfully.

Completion time: 2020-05-02T17:43:30.8220100-07:00

Ready Type here to search Ln 3 Col 67 Ch 64 INS 5:43 PM 5/2/2020



SQLQuery11.sql - DESKTOP-A19REFF\SQLEXPRESSS.XYZRecruitmentDatabase (sa (63)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

XYZRecruitmentDatabase Execute Quick Launch (Ctrl+Q)

Object Explorer

```

USE XYZRecruitmentDatabase
GO

CREATE LOGIN Venkata
WITH PASSWORD = 'Saipawan@897',
DEFAULT_DATABASE = XYZRecruitmentDatabase;

CREATE USER Phoebe
FOR LOGIN Venkata;
ALTER ROLE ReimbursementEntry
ADD MEMBER Phoebe;
--Venkata Sai Pawan komaravolu
  
```

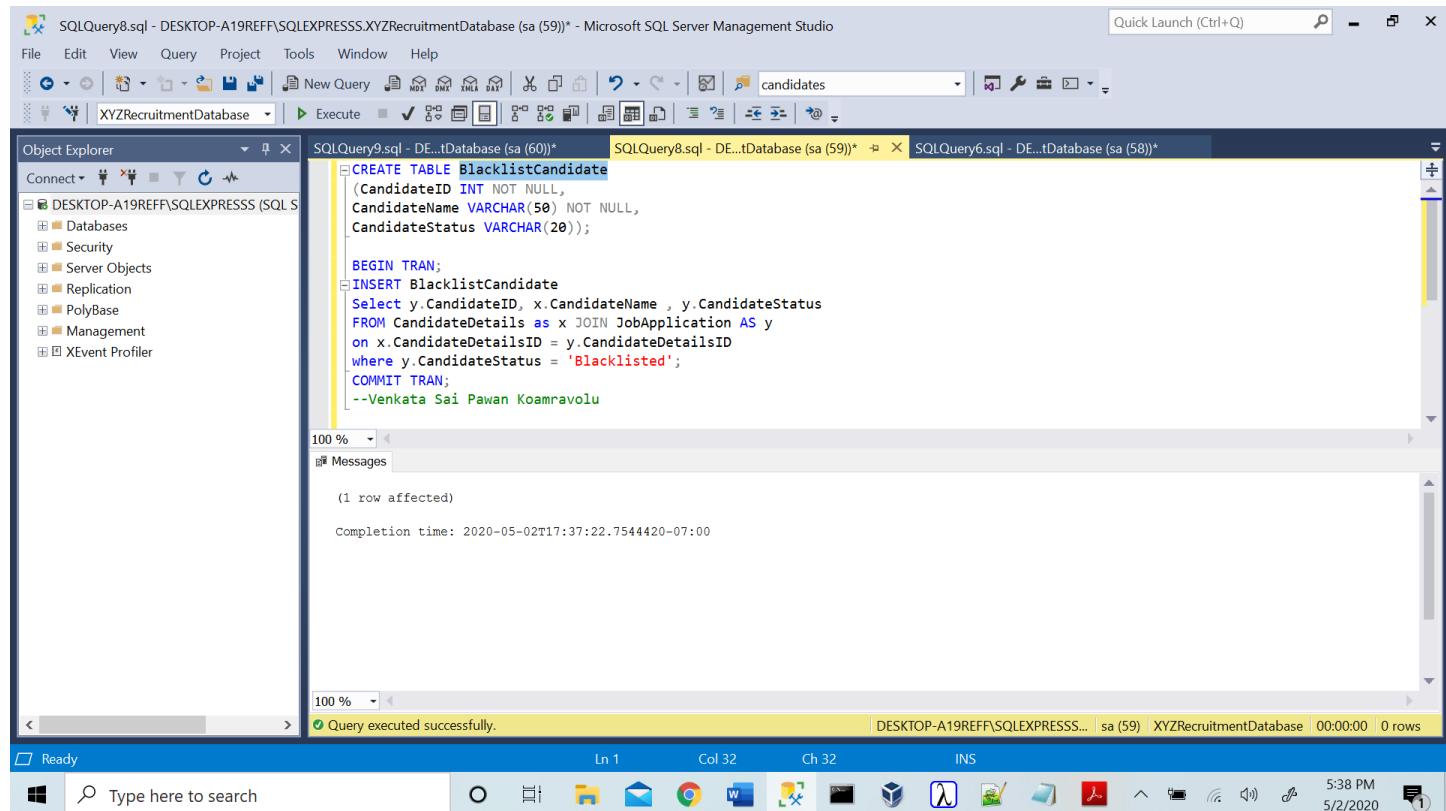
100 % Messages Commands completed successfully.

Completion time: 2020-05-02T17:46:23.8755446-07:00

Ready Type here to search Ln 12 Col 31 Ch 31 INS 5:46 PM 5/2/2020

4.6 TRANSACTION

We create a transaction whenever the status of the candidate is changed to blacklisted we send that candidate details to a table which is BlacklistCandidate. For this we create a table and declare the table. We begin the transaction and insert into the BlacklistCandidate the candidateID,CandidateName,CandidateStatus from Candidate details table join JobApplication table on common CandidateDetailsID and where the CandidateStatus is blacklisted and then we Commit the transaction.



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'XYZRecruitmentDatabase' is selected. In the center pane, a query window titled 'SQLQuery9.sql - DE...tDatabase (sa (60))' contains the following SQL code:

```

CREATE TABLE BlacklistCandidate
(
    CandidateID INT NOT NULL,
    CandidateName VARCHAR(50) NOT NULL,
    CandidateStatus VARCHAR(20));
BEGIN TRAN;
INSERT BlacklistCandidate
Select y.CandidateID, x.CandidateName , y.CandidateStatus
FROM CandidateDetails as x JOIN JobApplication AS y
on x.CandidateDetailsID = y.CandidateDetailsID
where y.CandidateStatus = 'Blacklisted';
COMMIT TRAN;
--Venkata Sai Pawan Koamravolu

```

Execution results are shown below the code:

- (1 row affected)
- Completion time: 2020-05-02T17:37:22.7544420-07:00
- Message bar: Query executed successfully.

We get the new table for Blacklisted Candidate and details are mentioned in the below screenshot

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'XYZRecruitmentDatabase' is selected. In the center pane, a query window displays the results of the following SQL statement:

```
Select * from BlacklistCandidate
```

The results show one row:

	CandidateID	CandidateName	CandidateStatus
1	901	ChandlerBing	Blacklisted

At the bottom of the screen, the taskbar shows the Windows Start button, a search bar with 'Type here to search', and several pinned application icons including File Explorer, Mail, Google Chrome, Word, Excel, and Power BI. The system tray indicates the date and time as 5/2/2020 at 5:38 PM.

5. CONCLUSION

The Implementation and Design of Recruitment Database for a company ‘xyz’ is done in this project. I’ve created database from scratch. In this project, I’ve used all the basic database concepts are implemented i.e. creation of tables, creation of columns, constraints, relationships between the tables used, Entity Relationship diagram, populating data, Normalizing the database, usage of Triggers ,creation of Views, usage of functions, Stored Procedures and transactions. By doing this project I have got great command over various skills on all the database concepts also got experience on working with the real-world project from scratch. We now have a recruitment database and can we implemented as all the test cases are successful and meets the problem statement.

Source Code for creating tables:

```
IF DB_ID('XYZRecruitmentDatabase') IS NOT NULL  
    DROP DATABASE XYZRecruitmentDatabase  
GO
```

```
CREATE DATABASE XYZRecruitmentDatabase;
```

```
GO
```

```
USE XYZRecruitmentDatabase;
```

```
GO
```

```
CREATE TABLE Addresses (  
    AddressID INT NOT NULL PRIMARY KEY,  
    AddressLine1 VARCHAR(50) NOT NULL,  
    AddressLine2 VARCHAR(50) Default NULL,  
    City CHAR(50) NOT NULL,  
    State CHAR(50) NOT NULL,  
    ZipCode INT NOT NULL  
)
```

```
CREATE TABLE ContactDetails (  
    ContactID INT NOT NULL PRIMARY KEY,  
    EmailAddress VARCHAR(50) NOT NULL,  
    PhoneNumber VARCHAR(12) NOT NULL,  
)
```

```
CREATE TABLE Department (
    DepartmentID INT NOT NULL PRIMARY KEY,
    DepartmentName CHAR(30) NOT NULL,
    DepartmentDescription varchar(50) NULL
);
```



```
CREATE TABLE CandidateQualification (
    CandidateQualificationID INT NOT NULL PRIMARY KEY,
    GradGpa INT NOT NULL,
    WorkExperienceYears INT NOT NULL,
);
```

```
CREATE TABLE Interviewer (
    InterviewerID INT NOT NULL PRIMARY KEY,
    InterviewerName CHAR(50) NOT NULL,
    InterviewerStatus Varchar(50) NULL
);
```

```
CREATE TABLE InterviewLocation (
    InterviewLocationID INT NOT NULL PRIMARY KEY,
    BuildingName VARCHAR(50) NOT NULL,
    City CHAR(50) NOT NULL,
    State CHAR(50) NOT NULL,
    ZipCode INT NOT NULL,
);
```

```
CREATE TABLE JobOpening (
    JobOpeningID INT NOT NULL PRIMARY KEY,
    JobRole VARCHAR(50) NOT NULL,
    JobDescription VARCHAR(50) NOT NULL,
```

```
DepartmentID INT NOT NULL REFERENCES Department(DepartmentID),  
JobType VARCHAR(50) NOT NULL,  
NoOfJobOpenings INT NOT NULL,  
);
```

```
CREATE TABLE CandidateDetails (  
CandidateDetailsID INT NOT NULL PRIMARY KEY,  
CandidateName CHAR(50),  
AddressID INT NOT NULL REFERENCES Addresses(AddressID),  
ContactID INT NOT NULL REFERENCES ContactDetails(ContactID),  
CandidateQualificationID      INT      NOT      NULL      REFERENCES  
CandidateQualification(CandidateQualificationID),  
);
```

```
CREATE TABLE JobApplication (  
CandidateID INT NOT NULL PRIMARY KEY,  
JobOpeningID INT NOT NULL REFERENCES JobOpening(JobOpeningID),  
CandidateDetailsID INT NOT NULL REFERENCES CandidateDetails(CandidateDetailsID),  
CandidateStatus VARCHAR(50) NOT NULL,  
CandidateName VARCHAR(50) NOT NULL,  
);
```

```
CREATE TABLE Interview (  
InterviewID INT NOT NULL PRIMARY KEY,  
InterviewerID INT NOT NULL References Interviewer(InterviewerID),  
CandidateID INT NOT NULL References JobApplication(CandidateID),  
InterviewResult CHAR(50) NOT NULL,  
InterviewType VARCHAR(20) NOT NULL,  
);
```

```
CREATE TABLE ComplaintHandling (  
ComplaintID INT NOT NULL PRIMARY KEY,
```

```
CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID),  
InterviewerID INT NOT NULL References Interview(InterviewerID),  
ComplaintDescription VARCHAR(100),  
);
```

```
CREATE TABLE AirlineReservation (  
TicketID INT NOT NULL PRIMARY KEY,  
InterviewID INT NOT NULL REFERENCES Interview(InterviewID),  
CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID),  
InterviewLocationID INT NOT NULL REFERENCES InterviewLocation(InterviewLocationID),  
FlightDetails VARCHAR(100),  
AirlineFare money  
);
```

```
CREATE TABLE Onboarding (  
OnboardingID INT NOT NULL PRIMARY KEY,  
CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID),  
JobOpeningID INT NOT NULL REFERENCES JobOpening(JobOpeningID),  
DepartmentID INT NOT NULL REFERENCES Department(DepartmentID),  
CandidateDocuments Varchar(50) NOT NULL  
);
```

```
CREATE TABLE HotelReservation (  
ReservationID INT NOT NULL PRIMARY KEY,  
InterviewID INT NOT NULL REFERENCES Interview(InterviewID),  
CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID),  
InterviewLocationID INT NOT NULL REFERENCES InterviewLocation(InterviewLocationID),  
HotelDetails VARCHAR(100) NOT NULL,  
HotelFare money NOT NULL,  
);
```

```
CREATE TABLE CarRental (
    RentalID INT NOT NULL PRIMARY KEY,
    InterviewID INT NOT NULL REFERENCES Interview(InterviewID),
    CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID),
    InterviewLocationID INT NOT NULL REFERENCES InterviewLocation(InterviewLocationID),
    CarDetails VARCHAR(50) ,
    RentalFare money ,
);
```

```
CREATE TABLE Reimbursement (
    ReimbursementID INT NOT NULL PRIMARY KEY,
    InterviewID INT NOT NULL REFERENCES Interview(InterviewID),
    CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID),
    RentalID INT NOT NULL REFERENCES CarRental(RentalID),
    TicketID INT NOT NULL REFERENCES AirlineReservation(TicketID),
    ReservationID INT NOT NULL REFERENCES HotelReservation(ReservationID),
    ReimbursementAmount MONEY
);
```

```
CREATE TABLE JobOfferNegotitation (
    NegotiationID INT NOT NULL PRIMARY KEY,
    CandidateID INT NOT NULL REFERENCES JobApplication(CandidateID),
    NegotitationStatus varchar(50) NOT NULL,
);
```