# CSE 581:

# INTRODUCTION TO DATABASE MANAGEMENT SYSTEMS

PROJECT: 2

*Submitted by*

## KAVYA NIRMAL SHAH

## SUID: 330126500

## 

## 

## *Table Of Contents*

*Abstract……………..…………………………………………………………………………………. 03*

*Design Introduction………….……………………………………………………………………….. 32*

*Views……………………………….………………………………………………………………….. 33*

*Stored Procedures……………………………………………………………………………………..41*

*Function…………………………………………………………………………………………………48*

*Conclusion………………………………………………………………………………………………. 49*

## ***Abstract***

*This project focuses on the design and implementation of a database for a company's Human Resources department, which will aid in boosting the organization's effectiveness.*

*This project assists in tracking the candidates' path from applying to the firm to successfully becoming a member of it. The major goal of this project is to create a database that will assist in tracking the status from time to time.*

*This project is built with SQL Server Management Studio and contains everything from the execution of simple queries to build tables and databases to the automated updation and efficient operation of the process.*

*This project aids in the implementation of concepts learned in lectures such as how to design, load, and manipulate databases using SQL Server Management Studio, as well as basic queries, keywords, statements, operators, joins, subqueries, summary queries, aggregate functions, datatypes, Stored Procedures, Views, Security, and much more in tables. Insertion, deletion, and updating are examples of data alteration procedures. Views, scripts, scalar and table valued functions, database implementation, and other subjects are covered. Furthermore, using an E-R diagram to build a database is a crucial feature. The bulk of the essential SQL and database principles are covered in this project.*

DESIGN INTRODUCTION

# 

# PROJECT STATEMENT

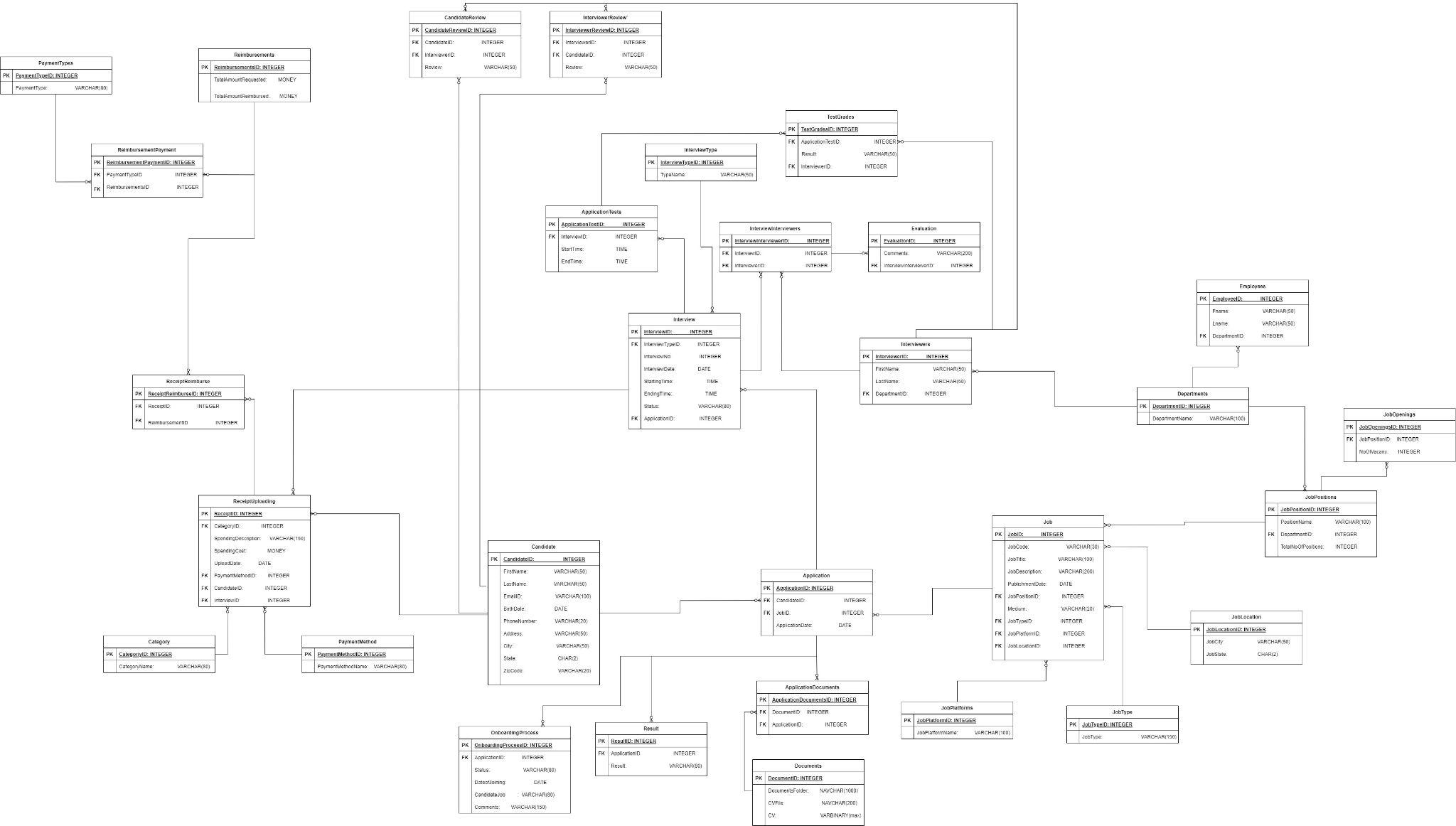
* *Candidate applies for job openings at Your Company.*
* *Company rejects or selects the candidate for 1st interview, which can be online or onsite.*
* *Company rejects or selects the candidate for the following interviews till the final ‘nth’ interview.*
* *Any of these interviews can be online or onsite.*
* *Interviews have start time and end time. One interview may involve multiple interviewers.*
* *Interviews may include multiple tests, which can again be online or onsite. Tests would have start and end time. A test may need to be graded by multiple interviewers with a simple result, passed or failed.*
* *The candidate can be rejected after any of the ‘n’ interviews.*
* *If a rejected candidate re-applies for any job-opening, a brand-new status flow starts for that candidate. There is no limit on the number of times a candidate can re-apply or the time after which a candidate can re-apply.*
* *If a rejected candidate complains about the hiring or interviewing process, the company puts the candidate on a ‘waiting’ status. The ‘Complaint Handling’ branch of the HR department investigates the complaint (with whose working we are not concerned). If the Complaint department finds that the complaint is correct, a re-interview takes place and status is changed accordingly. However, if the Complaint Department finds that the complaint is invalid, status is again changed to ‘rejected’.*
* *If after the final ‘nth’ interview the company likes the candidate but there are no more job openings available for the position interviewed for, the candidate is put on a “on-call for next job opportunity” status. This means that the company will themselves contact this candidate for interviews during the next round of recruitment.*
* *If after the final ‘nth’ interview the company likes the candidate and there is a job opening available, an offer is extended to the candidate.*
* *Whether a candidate is selected or rejected after (1) the ‘application scanning’ phase in the beginning or (2) after any of the ‘n’ interviews, is decided by the interview team of the company. You as a database admin are not concerned with the internal working of this interview team. When someone from the interview team updates the answer for a particular interview, you as admin will change the candidate’s status accordingly.*
* *The candidate can either decline or accept this offer or can negotiate ‘n’ number of times and then decline or accept the negotiated offer. Status is changed accordingly (Declined, Accepted, Negotiating).*
* *If the given or negotiated offer is accepted, the onboarding procedure starts. The onboarding procedure involves candidate’s background check, and collection and checking of all documents required for employment.*
* *If the candidate declines the given or negotiated offer, s/he is again put on the “on-call for next job opportunity” status.*
* *If onboarding is successfully completed, the candidate becomes an employee. For this project, this is where our recruitment procedure will end.*
* *If the onboarding is unsuccessful (some joining requirements were not met) the candidate is again put on the “on-call for next job opportunity” status.*
* *If after accepting the offer and/or after successful completion of onboarding, the candidate does not join the company, s/he is blacklisted. This means that the candidate cannot again apply for any job-opening in the company.*
* *If any interview is onsite, your database needs to keep track of the following things per candidate: Airline reservation details, Hotel reservation details, Car Rental details.*
* *The candidate will be asked to submit their expense receipts - all expenses that the company says it will cover for food, etc. during the period of the onsite recruitment process. The reimbursement team will update whether the reimbursement that the candidate is asking for is valid and how much reimbursement will be given.*
* *If any interview is online or at a location close to the candidate’s residence, one or more of the above given details would not be needed.*
* *The job openings can be for any department’s any position with the following details: actual job description including code, name, description, job start date, job type, job, type, job medium, job category, job platform, job start date, number of positions, etc.*
* *Job Type can be Summer Internship, Full-time Job, Part-time job, or Contract-based.*
* *Job Medium can be Online or Onsite.*
* *Job Categories can be such as “IT”, “software design”, “testing”, “finance”, etc.*
* *Job Position can be anything like “IT Manager”, “Software Developer”, etc.*
* *Job Platform is the medium used to post the job opening such as an online job board, company webpage, etc.*
* *The number of job openings per position are fixed and need to be tracked. Every time a position is filled by a candidate (that is when the status becomes ‘offer extended’), the number of job openings for that position should be reduced by 1).*
* *And every time an offer is declined, or somebody is blacklisted, the corresponding job opening should be increased by 1.*
* *Although here we have only one database admin, the database has other roles of interviewers, onboarding team specialist, etc., with the power to change a few things in the database. Thus, when the reducing or increasing of the number of job-openings by 1 is taking place, make sure that nobody else can do that at the same time, else there would be a synchronization issue.*
* *Per interview, the candidate can give a review for the interviewer, and the interviewer can give a review for the candidate.*
* *Make the status changes due to the on-going interview process per candidate as automatic as possible as the design of your database can allow.*

# Design Considerations

The HR Database is created by following the six fundamental stages outlined in the presentation.

* They identified the data items by examining the structure of numerous firms and the project statement's needs.
* Subdivide these aspects as much as feasible and into as many usable components as possible, such as which table will require which columns and data, for example, candidates like The candidate table will provide basic information on the people who are applying to the firm.
* I identified the tables and allocated the relevant columns to them.
* By understanding the link between the tables, I was able to identify the Primary Keys, Foreign Keys, and so on.
* As specified in the project requirements, I normalized the whole database into its 3rd Normal Form to decrease data redundancy, repeated columns, repeating values, and other issues.
* Normalization is a database design approach that lowers data redundancy and eliminates undesired features such as Insertion, Updation, and Deletion Anomalies. It separates bigger databases into smaller tables and connects them via relationships. The goal of SQL normalisation is to remove superfluous (repetitive) data and guarantee that data is stored correctly.
* When a relation is in 2NF but has no transitive partial dependency, it is said to be in its third normal form.
* Partial reliance, on the other hand, happens when one main key determines some other attribute/attributes.
* Identified the Indexes that are the data utilized in repetitive manner.

# E - R DIAGRAM



# DATABASE OBJECTS

| **ENTITY** | **ATTRIBUTES** | **CONSTRAINTS** |
| --- | --- | --- |
| 1. Candidate | CandidateID : int  FirstName: Varchar(50)  LastName : Varchar(50)  EmailID : Varchar(100) BirthDate : Date  PhoneNumber:Varchar(20)  Address : Varchar(200)  City : Varchar(50)  State : Char(2)  ZipCode : Varchar(20) | PRIMARY KEY  NOT NULL  NOT NULL  NOT NULL  NOT NULL  NOT NULL  NOT NULL |
| 1. Department | DepartmentID : int  DepartmentName : Varchar(150) | PRIMARY KEY  NOT NULL |
| 1. JobPositions | JobPositionID: int  JobPositionName : Varchar(100)  DepartmentID: int  TotalNoOfPositions: int | PRIMARY KEY  NOT NULL  FK |
| 1. JobType | JobTypeID : int  JobType : Varchar(150) | PRIMARY KEY  NOT NULL |
| 1. JobPlatforms | JobPlatformID : int  PlatformName : Varchar(100) | PRIMARY KEY  NOT NULL |
| 1. JobLocations | JobLocationID : int  JobCity : Varchar(50)  JobState : Char(2) | PRIMARY KEY  NOT NULL  NOT NULL |
| 1. JobOpening | JobOpeningID : int  JobPositionID : int  NoOfVacancies : int | PRIMARY KEY  FK  NOT NULL |
| 1. Job | JobID: int  JobCode: varchar(30)  JobTitle: varchar(100)  JobDescription : varchar(200)  PublishmentDate : date  JobPositionID: int JobMedium: varchar(20)  JobTypeID: int JobPlatformID: int  JobLocationID: int | PRIMARY KEY  NOT NULL  NOT NULL  NOT NULL  FK  NOT NULL  FK  FK  FK |
| 1. Applications | ApplicationID : int  CandidateID : int  JobID : int  ApplicationDate: Date | PRIMARY KEY  FK  FK  NOT NULL |
| 1. DocumentUpload | DocumentID: int DocumentsFolder: nvarchar(1000)  CVFile : nvarchar(200)  CV : varbinary(max) | PRIMARY KEY |
| 11)ApplicationDocuments | ApplicationDocumentsID: Int  DocumentID: Int  ApplicationID: Int | PRIMARY KEY  FK  FK |
| 12) InterviewTypes | InterviewTypeID : Int  TypeName: Varchar(50) | PRIMARY KEY  NOT NULL |
| 13) Interview | InterviewID : int  InterviewTypeID: int  InterviewNo : int  InterviewDate : date  StartingTime : time  EndingTime : time  Status : Varchar(80)  ApplicationID : int | PRIMARY KEY  FK  NOT NULL  NULL  NULL  NULL  NOT NULL  FK |
| 14) Interviewers | InterviewerID : int  FirstName : Varchar(50)  LastName : Varchar(50)  DepartmentID : int | PRIMARY KEY  NOT NULL  NOT NULL  FK |
| 15) InterviewInterviewer | InterviewInterviewerID : int  InterviewID : int  InterviewerID :int | PRIMARY KEY  FK  FK |
| 16) ApplicationTest | ApplicationTestID: int  InterviewID: int  StartTime : time  EndTime : time | PRIMARY KEY  FK  NOT NULL  NOT NULL |
| 17) TestGrade | TestGradeID : int ApplicationTestID : int Results : varchar(50) InterviewerID : int | PRIMARY KEY  FK  NOT NULL  FK |
| 18) CandidateReviews | CandidateReviewID: int  CandidateID: int  InterviewerID: int  Reviews varchar(200) | PRIMARY KEY  FK  FK |
| 19) InterviewerReviews | InterviewerReviewID : int  InterviewerID : int  CandidateID : int  Reviews : Varchar(200) | PRIMARY KEY  FK  FK |
| 20) Evaluation | EvaluationID: int Comments:varchar(200) InterviewInterviewerID: int | PRIMARY KEY  FK |
| 21) Category | CategoryID: int CategoryName: varchar(80) | PRIMARY KEY  NOT NULL |
| 22)PaymentMethods | PaymentMethodID: int  PaymentMethodName: varchar(80) | PRIMARY KEY  NOT NULL |
| 23) ReceiptUploading | ReceiptID :int  CategoryID : int  SpendingDescription : Varchar(150)  Spendingcost: money  UploadDate : date  PaymentMethodID: Int  CandidateID: int  InterviewID: int | PRIMARY KEY  FK  NOT NULL  NOT NULL  FK  FK  FK |
| 24) Reimbursements | ReimbursementID : int  TotalAmountRequested: money  TotalAmountReimbursed: money | PRIMARY KEY  NOT NULL  DEFAULT 0, NOT NULL |
| 25) PaymentTypes | PaymentTypeID : int  PaymentType: varchar(80) | PRIMARY KEY  NOT NULL |
| 26) ReimbursementPayment | ReimbursementPaymentID : int  PaymentTypeID: Int  ReimbursementID : int | PRIMARY KEY  FK  FK |
| 27) ReceiptReimbursement | ReceiptReimbursementID: int  ReceiptID: int  ReimbursementID: int | PRIMARY KEY  FK  FK |
| 28) Results | ResultID: int ApplicationID: int Result: varchar(80) | PRIMARY KEY  FK  NOT NULL |
| 29) OnboardingProcess | OnboardingProcessID: int  ApplicationID: int  Status : varchar(80)  DateOfJoining: date  CandidateJob : varchar(80)  Comments : varchar(150) | PRIMARY KEY  FK |
| 30)Employees | EmployeeID: int FirstNmame: varchar(50)  LastName: varchar(50)  DepartmentID: int | PRIMARY KEY  NOT NULL  NOT NULL  FK |

# 

# DDL SCRIPT:

USE master

GO

--new database

CREATE DATABASE HRDatabase

GO

USE HRDatabase

GO

--database tables

DECLARE @FOLDER as nvarchar(1000)

DECLARE @FILE as nvarchar(200)

DECLARE @SQL as nvarchar(1000)

--1) Candidate table

CREATE TABLE Candidate(

CandidateID int PRIMARY KEY,

FirstName varchar(50) NOT NULL,

LastName varchar(50) NOT NULL,

EmailId varchar(100),

BirthDate date,

PhoneNumber varchar(20),

Address varchar(200) NOT NULL,

City varchar(50) NOT NULL,

State char(2) NOT NULL,

ZipCode varchar(20) NOT NULL

);

--2) Department table

CREATE TABLE Department(

DepartmentID int PRIMARY KEY,

DepartmentName varchar(150) NOT NULL

);

--3) JobPosition table

CREATE TABLE JobPosition(

JobPositionID int PRIMARY KEY,

JobPositionName varchar(100) NOT NULL,

DepartmentID int FOREIGN KEY REFERENCES Department(DepartmentID),

TotalNoOfPositions int

);

--4) JobType table

CREATE TABLE JobType(

JobTypeID int PRIMARY KEY,

JobType varchar(150) NOT NULL

);

--5) JobPlatform table

CREATE TABLE JobPlatform(

JobPlatformID int PRIMARY KEY,

JobPlatformName varchar(100) NOT NULL

);

--6) JobLocation table

CREATE TABLE JobLocation(

JobLocationID int PRIMARY KEY,

JobCity varchar(50) NOT NULL,

JobState char(2) NOT NULL

);

--7) JobOpening table

CREATE TABLE JobOpening(

JobOpeningID int PRIMARY KEY,

JobPositionID int FOREIGN KEY REFERENCES JobPosition(JobPositionID),

NoOfVacancies int

);

--8) Job table

CREATE TABLE Job(

JobID int PRIMARY KEY,

JobCode varchar(30) NOT NULL,

JobTitle varchar(100) NOT NULL,

JobDescription varchar(200),

PublishmentDate date NOT NULL,

JobPositionID int FOREIGN KEY REFERENCES JobPosition(JobPositionID),

JobMedium varchar(20) NOT NULL,

JobTypeID int FOREIGN KEY REFERENCES JobType(JobTypeID),

JobPlatformID int FOREIGN KEY REFERENCES JobPlatform(JobPlatformID),

JobLocationID int FOREIGN KEY REFERENCES JobLocation(JobLocationID),

);

--9) Application table

CREATE TABLE Application(

ApplicationID int PRIMARY KEY,

CandidateID int FOREIGN KEY REFERENCES Candidate(CandidateID),

JobID int FOREIGN KEY REFERENCES Job(JobID),

ApplicationDate date NOT NULL

);

--10) DocumentUpload table

CREATE TABLE DocumentUpload(

DocumentID int PRIMARY KEY,

DocumentsFolder nvarchar(1000),

CVFile nvarchar(200),

CV varbinary(max)

);

--11) ApplicationDocuments table

CREATE TABLE ApplicationDocuments(

ApplicationDocumentsID int PRIMARY KEY,

DocumentID int FOREIGN KEY REFERENCES DocumentUpload(DocumentID),

ApplicationID int FOREIGN KEY REFERENCES Application(ApplicationID)

);

--12) InterviewTypes table

CREATE TABLE InterviewTypes(

InterviewTypeID int PRIMARY KEY,

TypeName varchar(50) NOT NULL

);

--13) Interview table

CREATE TABLE Interview(

InterviewID int PRIMARY KEY,

InterviewTypeID int FOREIGN KEY REFERENCES InterviewTypes(InterviewTypeID),

InterviewNo int NOT NULL,

InterviewDate date NULL,

StartingTime time NULL,

EndingTime time NULL,

Status varchar(80) NOT NULL,

ApplicationID int FOREIGN KEY REFERENCES Application(ApplicationID) NOT NULL

);

--14) Interviewers table

CREATE TABLE Interviewers(

InterviewerID int PRIMARY KEY,

FirstName varchar(50) NOT NULL,

LastName varchar(50) NOT NULL,

DepartmentID int FOREIGN KEY REFERENCES Department(DepartmentID) NOT NULL

);

--15) InterviewInterviewers table

CREATE TABLE InterviewInterviewers(

InterviewInterviewerID int PRIMARY KEY,

InterviewID int FOREIGN KEY REFERENCES Interview(InterviewID),

InterviewerID int FOREIGN KEY REFERENCES Interviewers(InterviewerID)

);

--16) ApplicationTest table

CREATE TABLE ApplicationTest(

ApplicationTestID int PRIMARY KEY,

InterviewID int FOREIGN KEY REFERENCES Interview(InterviewID),

StartTime time NOT NULL,

EndTime time NOT NULL

);

--17) TestGrade table

CREATE TABLE TestGrade(

TestGradeID int PRIMARY KEY,

ApplicationTestID int FOREIGN KEY REFERENCES ApplicationTest(ApplicationTestID),

Results varchar(50) NOT NULL,

InterviewerID int FOREIGN KEY REFERENCES Interviewers(InterviewerID)

);

--18) CandidateReviews table

CREATE TABLE CandidateReviews(

CandidateReviewID int PRIMARY KEY,

CandidateID int FOREIGN KEY REFERENCES Candidate(CandidateID),

InterviewerID int FOREIGN KEY REFERENCES Interviewers(InterviewerID),

Reviews varchar(200)

);

--19) InterviewerReviews table

CREATE TABLE InterviewerReviews(

InterviewerReviewID int PRIMARY KEY,

InterviewerID int FOREIGN KEY REFERENCES Interviewers(InterviewerID),

CandidateID int FOREIGN KEY REFERENCES Candidate(CandidateID),

Reviews varchar(200)

);

--20) Evaluation table

CREATE TABLE Evaluation(

EvaluationID int PRIMARY KEY,

Comments varchar(200),

InterviewInterviewerID int FOREIGN KEY REFERENCES InterviewInterviewers(InterviewInterviewerID)

);

--21) Category table

CREATE TABLE Category(

CategoryID int PRIMARY KEY,

CategoryName varchar(80) NOT NULL

);

--22) PaymentMethods table

CREATE TABLE PaymentMethods(

PaymentMethodID int PRIMARY KEY,

PaymentMethodName varchar(80) NOT NULL

);

--23) ReceiptUploading table

CREATE TABLE ReceiptUploading(

ReceiptID int PRIMARY KEY,

CategoryID int FOREIGN KEY REFERENCES Category(CategoryID),

SpendingDescription varchar(150),

SpendigCost money NOT NULL,

UploadDate date NOT NULL,

PaymentMethodID int FOREIGN KEY REFERENCES PaymentMethods(PaymentMethodID),

CandidateID int FOREIGN KEY REFERENCES Candidate(CandidateID),

InterviewID int FOREIGN KEY REFERENCES Interview(InterviewID),

);

--24) Reimbursements table

CREATE TABLE Reimbursements(

ReimbursementID int PRIMARY KEY,

TotalAmountRequested money NOT NULL,

TotalAmountReimbursed money DEFAULT 0 NOT NULL,

);

--25) PaymentTypes table

CREATE TABLE PaymentTypes(

PaymentTypeID int PRIMARY KEY,

PaymentType varchar(80) NOT NULL

);

--26) ReimbursementPayment table

CREATE TABLE ReimbursementPayment(

ReimbursementPaymentID int PRIMARY KEY,

PaymentTypeID int FOREIGN KEY REFERENCES PaymentTypes(PaymentTypeID),

ReimbursementID int FOREIGN KEY REFERENCES Reimbursements(ReimbursementID)

);

--27)ReceiptReimbursement table

CREATE TABLE ReceiptReimbursement(

ReceiptReimbursementID int PRIMARY KEY,

ReceiptID int FOREIGN KEY REFERENCES ReceiptUploading(ReceiptID),

ReimbursementID int FOREIGN KEY REFERENCES Reimbursements(ReimbursementID)

);

--28)Results table

CREATE TABLE Results(

ResultID int PRIMARY KEY,

ApplicationID int FOREIGN KEY REFERENCES Application(ApplicationID),

Result varchar(80) NOT NULL

);

--29)OnboardingProcess table

CREATE TABLE OnboardingProcess(

OnboardingProcessID int PRIMARY KEY,

ApplicationID int FOREIGN KEY REFERENCES Application(ApplicationID),

Status varchar(80),

DateOfJoining date,

CandidateJob varchar(80),

Comments varchar(150)

);

--30)Employee table

CREATE TABLE Employee(

EmployeeID int PRIMARY KEY,

FirstNmame varchar(50) NOT NULL,

LastName varchar(50) NOT NULL,

DepartmentID int FOREIGN KEY REFERENCES Department(DepartmentID)

);

--Database population

--1)Candidate table

INSERT INTO Candidate VALUES

(1, 'James', 'Elsher', 'je@syr.edu', '1986-05-16', '(601) 915-3900', '411 Ashley Ave', 'Norrthbrook', 'IL', 60062),

(2, 'Mary', 'Solace', 'ms@syr.edu', '1987-01-13', '(639) 821-3926', '69 Belmont Street', 'West Chicago', 'IL', 60185),

(3, 'Robert', 'Levine', 'rl@syr.edu', '1991-04-18', '(813) 612-7343', '8371 Fieldstone St', 'Moorhead', 'MN', 56560),

(4, 'Amy', 'Thatcher', 'at@syr.edu', '1992-02-29', '(577) 226-6990', '34 Airport Avenue', 'Groton', 'CT', 06340),

(5, 'John', 'Raven', 'jr@syr.edu', '1999-03-30', '(742) 599-3981', '18 Pilgrim St.', 'Helena', 'MT', 59601),

(6, 'Linda', 'Bardot', 'lb@syr.edu', '1994-02-14', '(554) 833-5746', '9422 Linden St', 'Essex', 'MD', 21221),

(7, 'David', 'Hansley', 'dh@syr.edu', '1996-03-04', '(620) 706-4539', '63 Gulf St.', 'Queensbury', 'NY', 12804),

(8, 'Sarah', 'Cromwell', 'sc@syr.edu', '1997-12-31', '(208) 942-3261', '8830 Vale St.', 'Amber', 'PA', 19002),

(9, 'Lisa', 'Ashley', 'la@syr.edu', '2000-01-03', '(851) 431-5329', '9017 Oak Valley Street', 'Johnston', 'RI', 02919),

(10, 'Karen', 'Monroe', 'km@syr.edu', '2001-10-12', '(753) 433-5263', '9894 W. Monroe Drive', 'Minneapolis', 'MN', 55406);

--2)Departments table

INSERT INTO Department VALUES

(1, 'Engineering & Technology'),

(2, 'Sales Service & Support'),

(3, 'Marketing & Communications'),

(4, 'Design'),

(5, 'Business Strategy'),

(6, 'Finance'),

(7, 'Legal'),

(8, 'People'),

(9, 'Facilities'),

(10, 'Operations');

--3)JobPosition table

INSERT INTO JobPosition VALUES

(1, 'Software Engineer', 1, 10),

(2, 'Product Manager', 1, 8),

(3, 'Account Manager', 2, 7),

(4, 'Business Partnerships', 2, 10),

(5, 'Marketing', 3, 9),

(6, 'Analyst', 3, 6),

(7, 'UX Engineer', 4, 7),

(8, 'Visual Designer', 4, 8),

(9, 'Strategy Planner', 5, 10),

(10, 'Program Manager', 5, 10),

(11, 'Accountant', 6, 7),

(12, 'Auditor', 6, 6),

(13, 'Counsel', 7, 3),

(14, 'Legal Support', 7, 4),

(15, 'Recruiter', 8, 6),

(16, 'HR Business Partner', 8, 2),

(17, 'Facility Management', 9, 1),

(18, 'Security', 9, 4),

(19, 'Operation Analyst', 10, 8),

(20, 'Program Manager', 10, 10);

--4)JobType table

INSERT INTO JobType VALUES

(1, 'Summer Internship'),

(2, 'Full-Time Job'),

(3, 'Part-Time Job'),

(4, 'Contract-Based');

--5)JobPlatform table

INSERT INTO JobPlatform VALUES

(1, 'Google Jobs'),

(2, 'LinkedIn'),

(3, 'GlassDoor'),

(4, 'Indeed'),

(5, 'Company Portal');

--6)JobLocation table

INSERT INTO JobLocation VALUES

(1, 'Seattle', 'WA'),

(2, 'San Jose', 'CA'),

(3, 'San Francisco', 'CA'),

(4, 'Madison', 'WI'),

(5, 'Raleigh', 'NC'),

(6, 'Austin', 'TX'),

(7, 'Boston', 'MA'),

(8, 'Sacramento', 'CA'),

(9, 'Portland', 'OR'),

(10, 'Atlanta', 'GA');

--7)JobOpening table

INSERT INTO JobOpening VALUES

(1, 1, 2),

(2, 2, 2),

(3, 3, 1),

(4, 4, 2),

(5, 5, 1),

(6, 6, 2),

(7, 7, 1),

(8, 8, 1),

(9, 9, 2),

(10,10, 1),

(11,11, 0),

(12,12, 0),

(13,13, 1),

(14,14, 0),

(15,15, 0),

(16,16, 1),

(17,17, 0),

(18,18,0),

(19,19,0),

(20,20, 0);

--8)Job table

INSERT INTO Job VALUES

(1, '#J512345', 'Software Engineer, Full Stack', ' ','2021-11-10', 1, 'Online', 1, 2, 8),

(2, '#J512346', 'Software Developer',' ', '2021-11-21', 1, 'Online', 1, 5, 2),

(3, '#J512347', 'Product Manager', ' ', '2021-11-25', 2, 'Onsite', 1, 1, 5),

(4, '#J512348', 'Group Product Manager',' ', '2021-12-10', 2, 'Onsite', 2, 3, 4),

(5, '#J512349', 'Technical Account Manager', ' ', '2021-12-11', 3, 'Online', 2, 2, 4),

(6, '#J512350', 'Strategic Partnerships Development Manager', ' ', '2021-11-16', 4, 'Onsite', 2, 5, 3),

(7, '#J512351', 'Product Partnership Manager',' ', '2021-11-18', 4, 'Online', 2, 1, 7),

(8, '#J512352', 'Senior Content Marketing Manager',' ', '2021-11-21', 5, 'Onsite', 2, 2, 8),

(9, '#J512353', 'Senior Marketing Analyst',' ', '2021-12-11', 6, 'Onsite', 2, 5, 6),

(10,'#J512354', 'Senior Analytics Manager, Media',' ', '2021-12-18', 6, 'Online', 2, 4, 3),

(11,'#J512355', 'Senior UX Engineer',' ', '2021-11-10', 7, 'Onsite', 2, 2, 2),

(12,'#J512356', 'Senior Visual Designer',' ', '2021-11-14', 8, 'Onsite', 2, 4, 10),

(13,'#J512357', 'Strategy And Operations Manager',' ', '2021-11-18', 9, 'Online', 3, 1, 7),

(14,'#J512358', 'Strategy And Operations Lead', ' ','2021-11-25', 9, 'Online', 4, 3, 9),

(15,'#J512359', 'Program Manager',' ', '2021-12-12', 10, 'Onsite', 2, 2, 1),

(16,'#J512360', 'Associate Product Counsel',' ', '2021-11-10', 13, 'Onsite', 1, 4, 4),

(17,'#J512361', 'Market HR Partner',' ', '2021-11-22', 16, 'Onsite', 2, 1, 6);

--9)Application table

INSERT INTO Application VALUES

(1, 1, 1, '2021-11-11'),

(2, 1, 2, '2021-11-23'),

(3, 1, 3, '2021-11-30'),

(4, 1, 4, '2021-12-23'),

(5, 2, 5, '2021-12-28'),

(6, 3, 6, '2021-11-29'),

(7, 3, 7, '2021-12-18'),

(8, 4, 11, '2021-12-19'),

(9, 4, 12, '2021-12-12'),

(10, 5, 5, '2021-12-13'),

(11, 5, 6, '2021-12-14'),

(12, 6, 17, '2021-12-16'),

(13, 7, 12, '2021-12-19'),

(14, 8, 13, '2021-12-23'),

(15, 8, 14, '2021-12-26'),

(16, 8, 15, '2021-12-27'),

(17, 9, 11, '2021-12-28'),

(18, 10, 8, '2021-12-29'),

(19, 10, 9, '2021-12-31'),

(20, 10, 10, '2021-12-29');

--10)DocumentUpload table

INSERT INTO DocumentUpload

SELECT 1,'C:\Users\ADMiN\Desktop\DatabaseFiles\','file1.txt', BulkColumn

FROM OPENROWSET(BULK 'C:\Users\ADMiN\Desktop\DatabaseFiles\file1.txt', SINGLE\_BLOB) AS FName;

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file2.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 2, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file3.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 3, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file4.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 4, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file5.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 5, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file6.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 6, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file7.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 7, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file8.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 8, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file9.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 9, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file10.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 10, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file11.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 11, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file12.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 12, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file13.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 13, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file14.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 14, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file15.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 15, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file16.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 16, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file17.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 17, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file18.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 18, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file19.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 19, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

SET @FOLDER = 'C:\Users\ADMiN\Desktop\DatabaseFiles\'

SET @FILE = 'file20.txt'

SET @SQL = 'INSERT INTO DocumentUpload

SELECT 20, '''+@FOLDER+''','''+@FILE+''', BulkColumn

FROM OPENROWSET (BULK '''+@FOLDER+@FILE+''',SINGLE\_BLOB) AS FName';

EXEC(@SQL);

--11)ApplicationDocuments table

INSERT INTO ApplicationDocuments VALUES

(1, 1, 1),

(2, 2, 2),

(3, 3, 3),

(4, 4, 4),

(5, 5, 5),

(6, 6, 6),

(7, 7, 7),

(8, 8, 8),

(9, 9, 9),

(10,10, 10),

(11,11, 11),

(12,12, 12),

(13,13, 13),

(14,14, 14),

(15,15, 15),

(16,16, 16),

(17,17, 17),

(18,18, 18),

(19,19, 19),

(20,20, 20);

--12)InterviewTypes table

INSERT INTO InterviewTypes VALUES

(1, 'Online'),

(2, 'Onsite');

--13)Interview table

INSERT INTO Interview VALUES

(1, 1, 1, '2021-11-23', '01:35:20', '03:35:20', 'Pass', 1),

(2, 1, 2, '2021-12-27', '01:59:59', '02:59:59', 'Pass', 1),

(3, 1, 1, '2021-12-31', '03:08:30', '04:08:30', 'Fail', 3),

(4, 2, 1, '2022-02-13', '03:21:23', '04:21:23', 'Pass', 5),

(5, 1, 1, '2021-12-30', '03:58:39', '05:58:39', 'Fail', 6),

(6, 1, 1, '2022-02-12', '04:04:49', '06:04:49', 'Pass', 9),

(7, 2, 2, '2022-03-22', '05:12:10', '07:05:11', 'Fail', 9),

(8, 2, 1, '2022-01-31', '05:19:56', '06:19:53', 'Pass', 12),

(9, 1, 1, '2022-01-19', '07:42:01', '09:42:01', 'Pass', 13),

(10, 1, 2, '2022-02-23', '09:35:15', '10:35:15', 'Pass', 13),

(11, 2, 3, '2022-03-13', '10:48:17', '11:48:17', 'Pass', 13),

(12, 1, 1, '2022-02-10', '12:35:49', '14:35:49', 'Pass', 15),

(13, 2, 2, '2022-03-11', '14:34:39', '16:34:39', 'Fail', 15),

(14, 1, 1, '2022-01-30', '15:20:12', '17:20:12', 'Pass', 17),

(15, 1, 2, '2022-02-08', '16:01:46', '18:01:46', 'Pass', 17);

--14)Interviewers table

INSERT INTO Interviewers VALUES

(1, 'Violet', 'Markey', 1),

(2, 'Jasmine', 'Wood', 1),

(3, 'Phoebe', 'Carr', 2),

(4, 'Aysha', 'Russel', 2),

(5, 'Penny', 'Hughes', 3),

(6, 'Sheldon', 'Copper', 3),

(7, 'Raj', 'Kotharpali', 4),

(8, 'Lenord', 'Hofstader', 4),

(9, 'Howard', 'Wolowitz', 5),

(10, 'Harry', 'Potter', 5),

(11, 'Ron', 'Wesely', 6),

(12, 'Finch', 'Theodore', 6),

(13, 'Tara', 'Warren', 7),

(14, 'Alice', 'Nichollas', 7),

(15, 'Liam', 'Nicholls', 8),

(16, 'Harry', 'Lewis', 8),

(17, 'Emma', 'Watson', 9),

(18, 'Calvin', 'Palmer', 9),

(19, 'John', 'Alexander', 10),

(20, 'Abdul', 'Khan', 10);

--15)InterviewInterviewers table

INSERT INTO InterviewInterviewers VALUES

(1, 1, 1),

(2, 2, 1),

(3, 2, 2),

(4, 3, 2),

(5, 4, 3),

(6, 4, 4),

(7, 5, 3),

(8, 6, 7),

(9, 7, 7),

(10,7, 8),

(11,8, 15),

(12,8, 16),

(13,9, 7),

(14,10, 8),

(15,11, 7),

(16,11, 8),

(17,12, 9),

(18,12, 10),

(19,13, 9),

(20,14, 7),

(21,15, 8);

--16) ApplicationTest table

INSERT INTO ApplicationTest VALUES

(1, 1, '01:35:20', '03:29:20'),

(2, 2, '01:59:59', '02:59:59'),

(3, 3, '03:08:30', '04:01:30'),

(4, 4, '03:21:23', '04:21:23'),

(5, 5, '03:58:39', '04:48:38'),

(6, 5, '04:48:38', '05:48:38'),

(7, 6, '04:04:49', '05:04:49'),

(8, 6, '05:04:49', '06:02:49'),

(9, 7, '05:12:10', '07:05:11'),

(10, 8, '05:19:56', '06:19:53'),

(11, 9, '07:42:01', '08:42:01'),

(12, 9, '08:42:01', '09:42:01'),

(13, 10, '09:35:15', '10:33:10'),

(14, 11, '10:48:17', '11:43:17'),

(15, 12, '12:35:49', '14:35:49'),

(16, 13, '14:34:39', '15:34:39'),

(17, 13, '15:34:39', '16:33:39'),

(18, 14, '15:20:12', '17:22:12'),

(19, 15, '16:01:46', '18:02:47');

--17) TestGrade table

INSERT INTO TestGrade VALUES

(1, 1, 'Pass', 1),

(2, 2, 'Pass', 1),

(3, 2, 'Pass', 2),

(4, 3, 'Fail', 2),

(5, 4, 'Pass', 3),

(6, 4, 'Pass', 4),

(7, 5, 'Pass', 3),

(8, 6, 'Fail', 3),

(9, 7, 'Pass', 7),

(10, 8, 'Pass', 7),

(11, 9, 'Pass', 7),

(12, 9, 'Fail', 8),

(13, 10, 'Pass', 15),

(14, 10, 'Pass', 16),

(15, 11, 'Pass', 7),

(16, 12, 'Pass', 7),

(17, 13, 'Pass', 8),

(18, 14, 'Pass', 7),

(19, 14, 'Pass', 8),

(20, 15, 'Pass', 9),

(21, 15, 'Pass', 10),

(22, 16, 'Pass', 9),

(23, 17, 'Fail', 9),

(24, 18, 'Pass', 7),

(25, 19, 'Pass', 8);

--18) CandidateReviews table

INSERT INTO CandidateReviews VALUES

(1, 1, 1, 'Good'),

(2, 1, 2, 'Better'),

(3, 2, 3, 'Good'),

(4, 2, 4, 'Good'),

(5, 3, 3, 'Good'),

(6, 4, 7, 'Better'),

(7, 4, 8, 'Good'),

(8, 6, 15, 'Good'),

(9, 6, 16, 'Good'),

(10, 7, 7, 'Good'),

(11, 7, 8, 'Better'),

(12, 8, 9, 'Good'),

(13, 8, 10, 'Better'),

(14, 9, 7, 'Good'),

(15, 9, 8, 'Better');

--19) InterviewerReviews table

INSERT INTO InterviewerReviews VALUES

(1, 1, 1, 'Good'),

(2, 2, 1, 'Better'),

(3, 3, 2, 'Good'),

(4, 3, 3, 'Needs Improvement'),

(5, 4, 2, 'Good'),

(6, 7, 4, 'Good'),

(7, 7, 7, 'Good'),

(8, 7, 9, 'Good'),

(9, 8, 4, 'Needs Improvement'),

(10, 8, 7, 'Good'),

(11, 8, 9, 'Good'),

(12, 9, 8, 'Good'),

(13, 10, 8, 'Needs Improvement'),

(14, 15, 6, 'Good'),

(15, 16, 6, 'Good');

--20) Evaluation table

INSERT INTO Evaluation VALUES

(1, 'Good', 1),

(2, 'Good', 2),

(3, 'Better', 3),

(4, 'Needs Improvement', 4),

(5, 'Good', 5),

(6, 'Excellent', 6),

(7, 'Good', 7),

(8, 'Good', 8),

(9, 'Good', 9),

(10, 'Needs Improvement', 10),

(11, 'Good', 11),

(12, 'Good', 12),

(13, 'Good', 13),

(14, 'Good', 14),

(15, 'Excellent', 15),

(16, 'Excellent', 16),

(17, 'Better', 17),

(18, 'Excellent', 18),

(19, 'Needs Improvement', 19),

(20, 'Good', 20),

(21, 'Good', 21);

--21) Category table

INSERT INTO CategorY VALUES

(1, 'Transportation'),

(2, 'Hotel'),

(3, 'Food');

--22) PaymentMethods table

INSERT INTO PaymentMethodS VALUES

(1, 'Credit/ Debit Card'),

(2, 'Cash');

--23) ReceiptUploading table

INSERT INTO ReceiptUploading VALUES

(1, 1, 'Air Fare', 2000, '2021-12-31', 1, 1, 3),

(2, 2, 'Hotel Reservation', 1000, '2021-12-31', 1, 1, 3),

(3, 3, 'Food', 100, '2021-12-31', 2, 1, 3),

(4, 1, 'Car Rent', 200, '2022-03-22', 1, 4, 7),

(5, 2, 'Hotel Reservation', 1000, '2022-03-22', 2, 4, 7),

(6, 1, 'Air Fare', 1000, '2022-02-10', 1, 8, 12),

(7, 1, 'Air Fare', 1000, '2022-03-11', 1, 8, 13);

--24) Reimbursements table

INSERT INTO Reimbursements VALUES

(1, 3100, 3100),

(2, 1200, 1200),

(3, 2000, 2000);

--25) PaymentTypes table

INSERT INTO PaymentTypes VALUES

(1, 'Direct Deposit'),

(2, 'Cash'),

(3, 'Cheque');

--26) ReimbursementPayment table

INSERT INTO ReimbursementPayment VALUES

(1, 1, 1),

(2, 2, 2),

(3, 3, 3);

--27)ReceiptReimbursement table

INSERT INTO ReceiptReimbursement VALUES

(1, 1, 1),

(2, 2, 1),

(3, 3, 1),

(4, 4, 2),

(5, 5, 2),

(6, 6, 3),

(7, 7, 3);

--28)Results table

INSERT INTO Results VALUES

(1, 1, 'Accepted'),

(2, 3, 'Rejected'),

(3, 5, 'Interview In Process'),

(4, 6, 'Rejected'),

(5, 9, 'Rejected'),

(6, 12, 'Interview In Process'),

(7, 13, 'Accepted'),

(8, 15, 'Rejected'),

(9, 17, 'Accepted');

--29)OnboardingProcess table

INSERT INTO OnboardingProcess VALUES

(1, 1,'Successful', '2022-01-27', 'Accepted', 'NULL'),

(2, 13,'Unscucessful', NULL, 'Denied', 'NULL'),

(3, 17,'Unscucessful', NULL, 'Accepted', 'On Call For next Job Opportunity');

--30)Employee table

INSERT INTO Employee VALUES

(1, 'Smith', 'Cindy', 2),

(2, 'Jones', 'Elmer', 4),

(3, 'Simonian', 'Ralph', 2),

(4, 'Hernandez', 'Olivia', 1),

(5, 'Aaronsen', 'Robert', 2),

(6, 'Watson', 'Denise', 6),

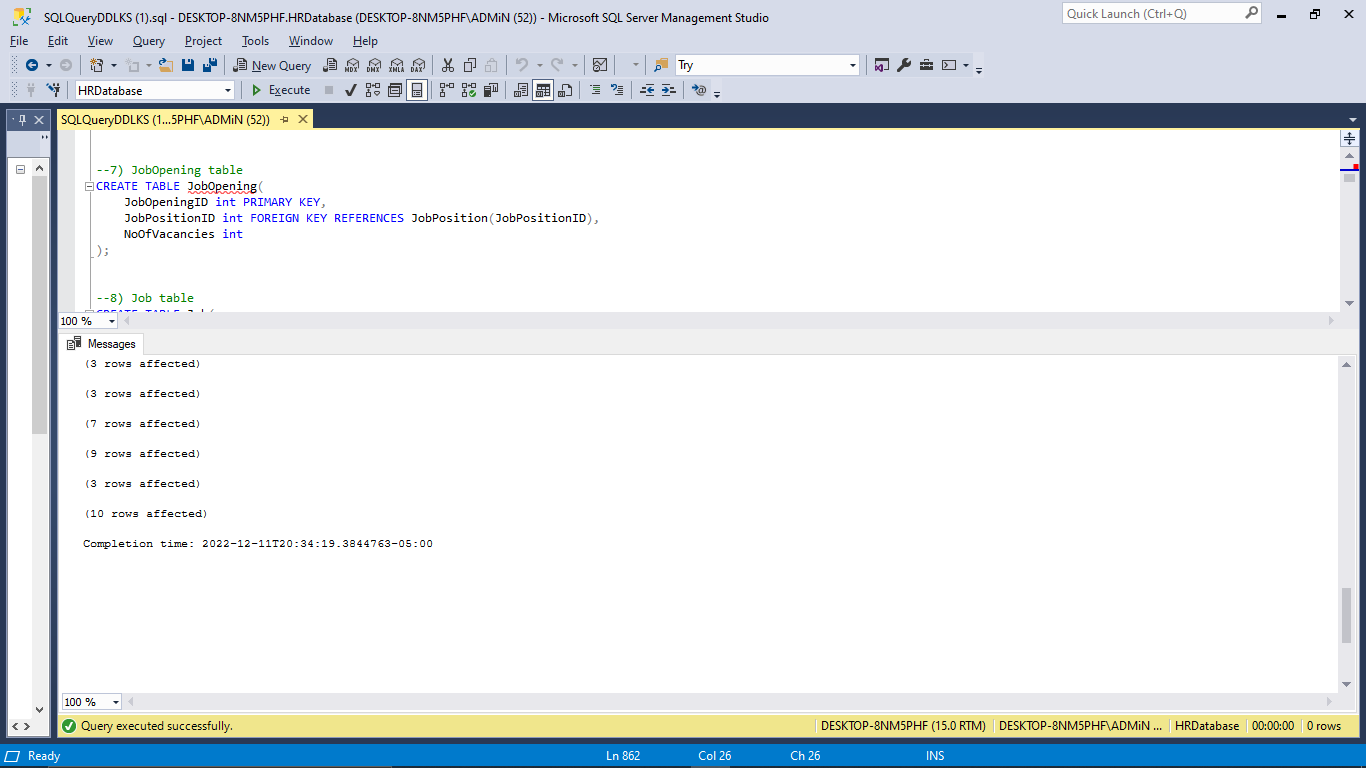
(7, 'Hardy', 'Thomas', 5),

(8, 'Leary', 'Rhea', 4),

(9, 'Locario', 'Paulo', 6),

(10, 'James', 'Elsher', 1);

**OUTPUT :**



**Comment:**

DDL instructions were successfully executed, and 30 tables were created in the HR Database.

# 

# 

# VIEWS:

***Introduction:***

*A view is a virtual table in SQL that is based on the result set of a SQL operation. A view, like a true table, has rows and columns. A view's fields are taken from one or more actual tables in the database.*

1. Create a view for all the available jobs which are located in TX and are onsite jobs

**Comments:**

A VIEW is created called TexasOnsiteJobs using Create View Statement to retrieve the required data.

**Query:**

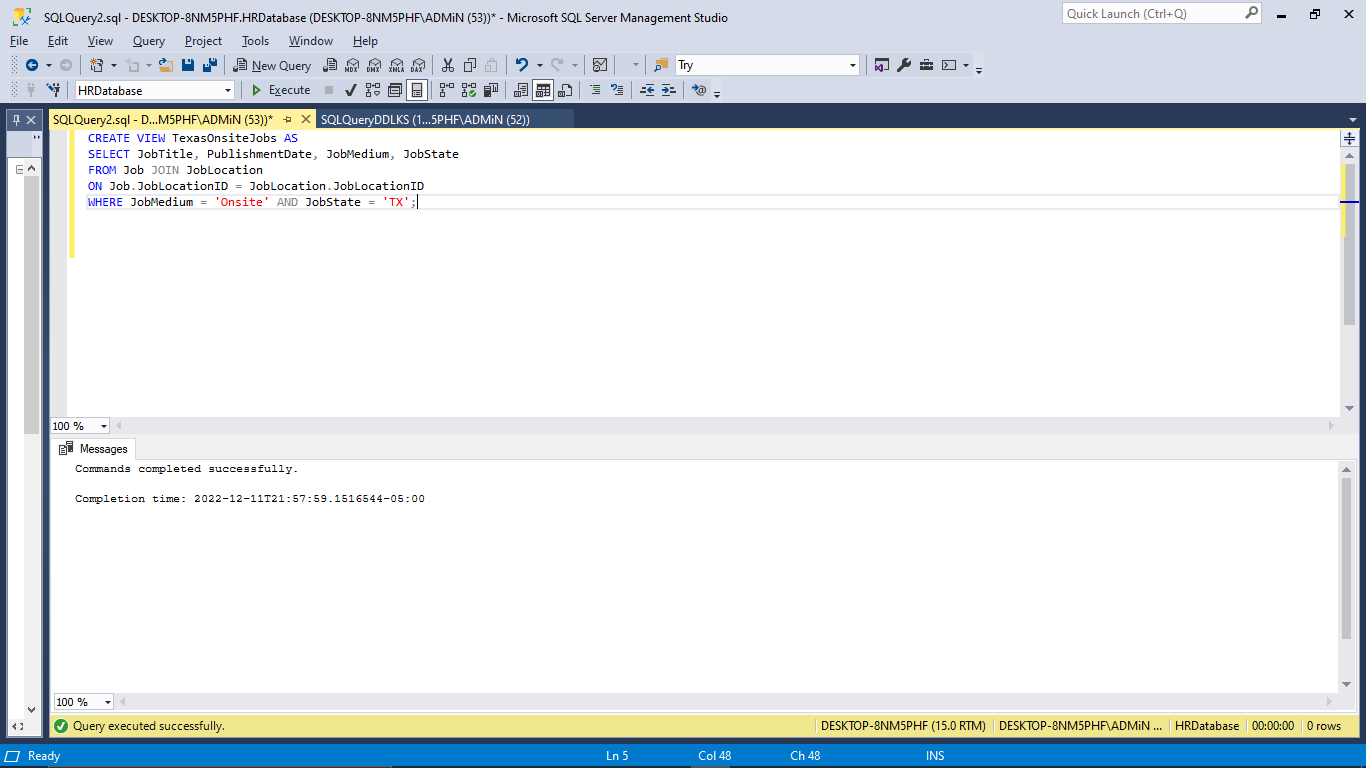
CREATE VIEW TexasOnsiteJobs AS

SELECT JobTitle, PublishmentDate, JobMedium, JobState

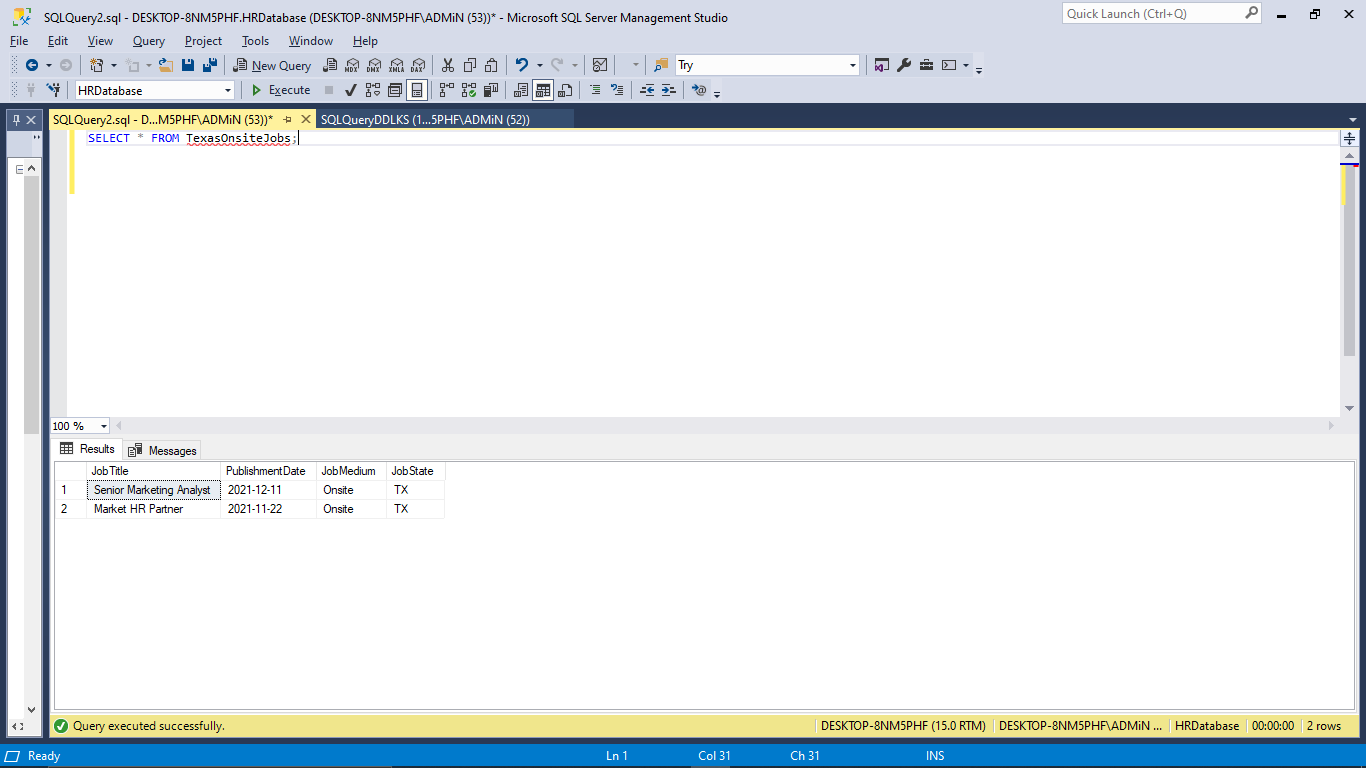
FROM Job JOIN JobLocation

ON Job.JobLocationID = JobLocation.JobLocationID

WHERE JobMedium = 'Onsite' AND JobState = 'TX';



SELECT \* FROM TexasOnsiteJobs;



1. Create a view for all those candidates who applied before 23rd December, 2022

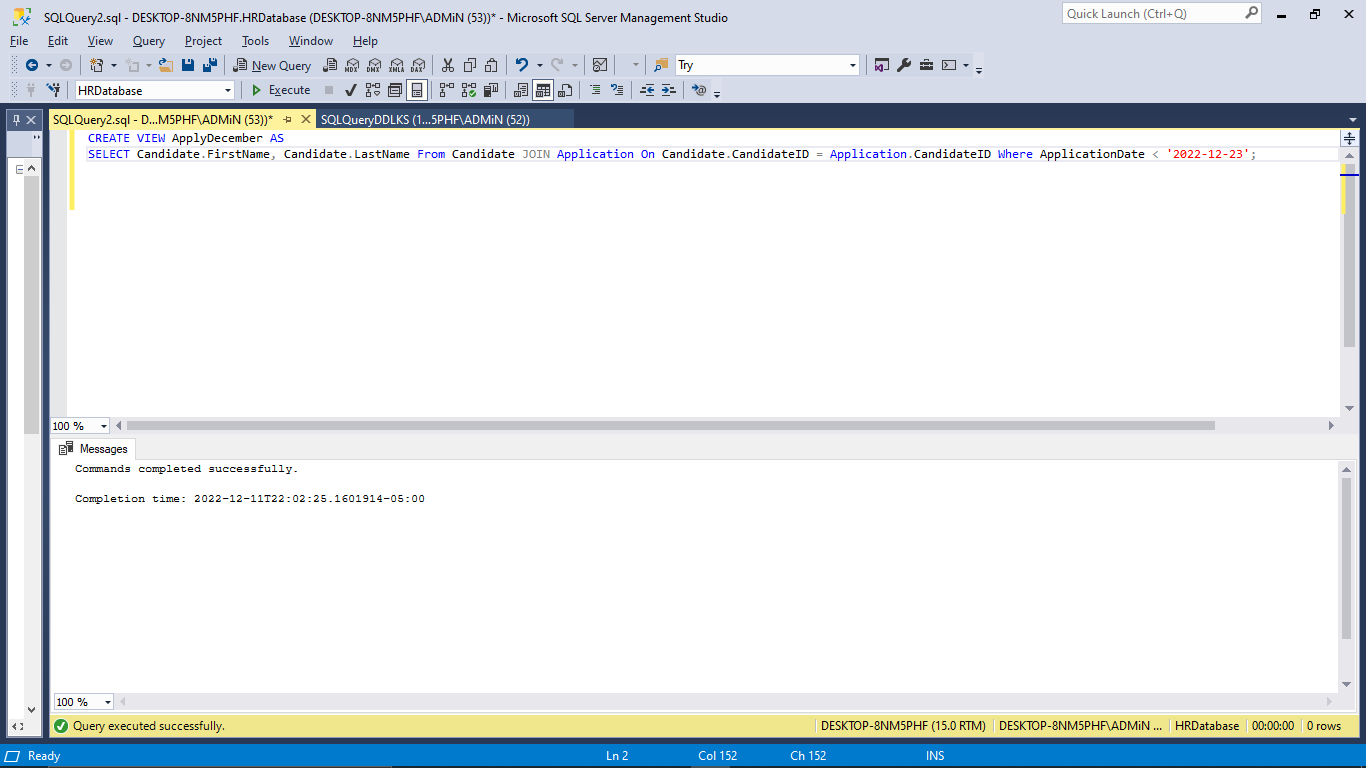
**Comments:**

A VIEW is created called ApplyDecember using Create View Statement to retrieve the required data.

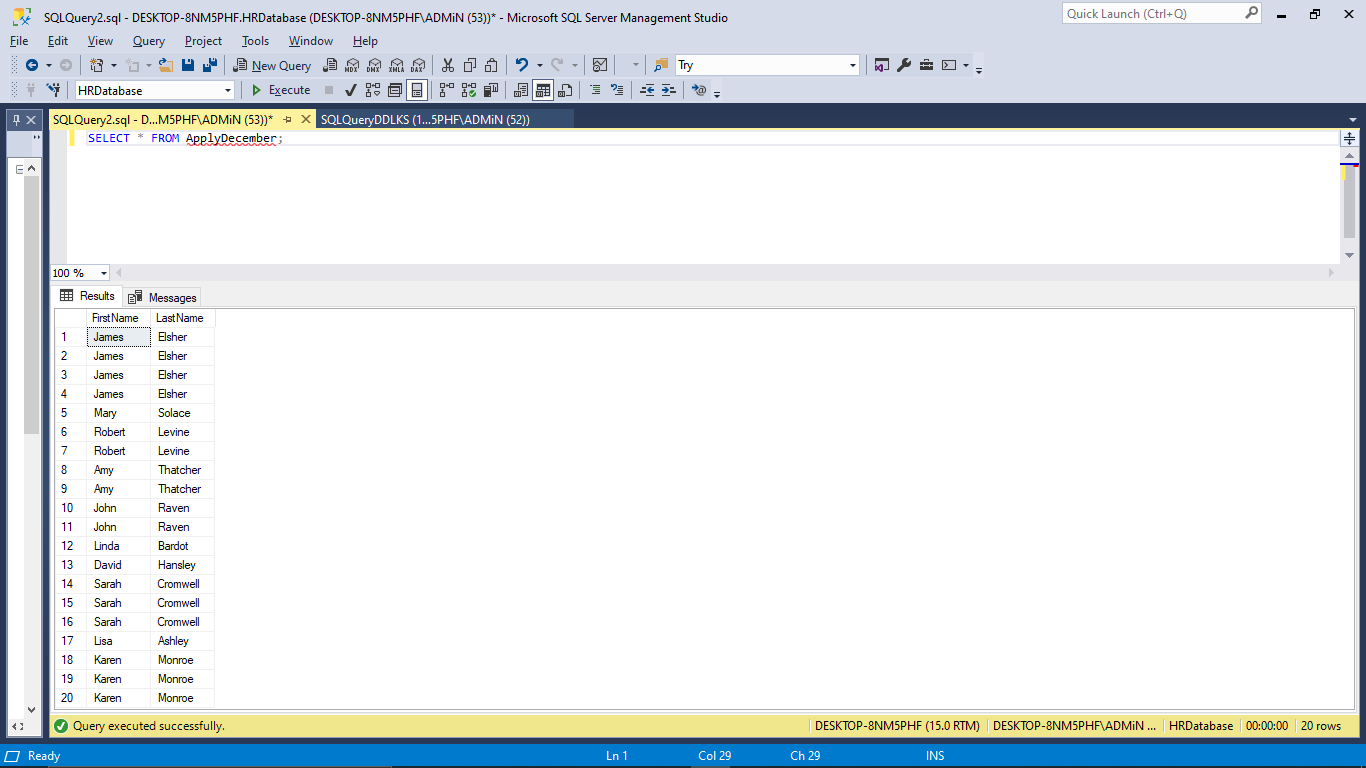
**Query:**

CREATE VIEW ApplyDecember AS

SELECT Candidate.FirstName, Candidate.LastName From Candidate JOIN Application On Candidate.CandidateID = Application.CandidateID Where ApplicationDate < '2022-12-23';



SELECT \* FROM ApplyDecember;



1. Create a view that shows the list of Candidates whose interviews were taken before 1st DECEMBER 2021

**Comments:**

A VIEW is created called CandidateBefDecember using Create View Statement to retrieve the required data.

**Query:**

CREATE VIEW CandidatesBefDecember AS

SELECT Candidate.CandidateID, LastName + ' ' + FirstName AS FullName, InterviewNo, InterviewDate, Status

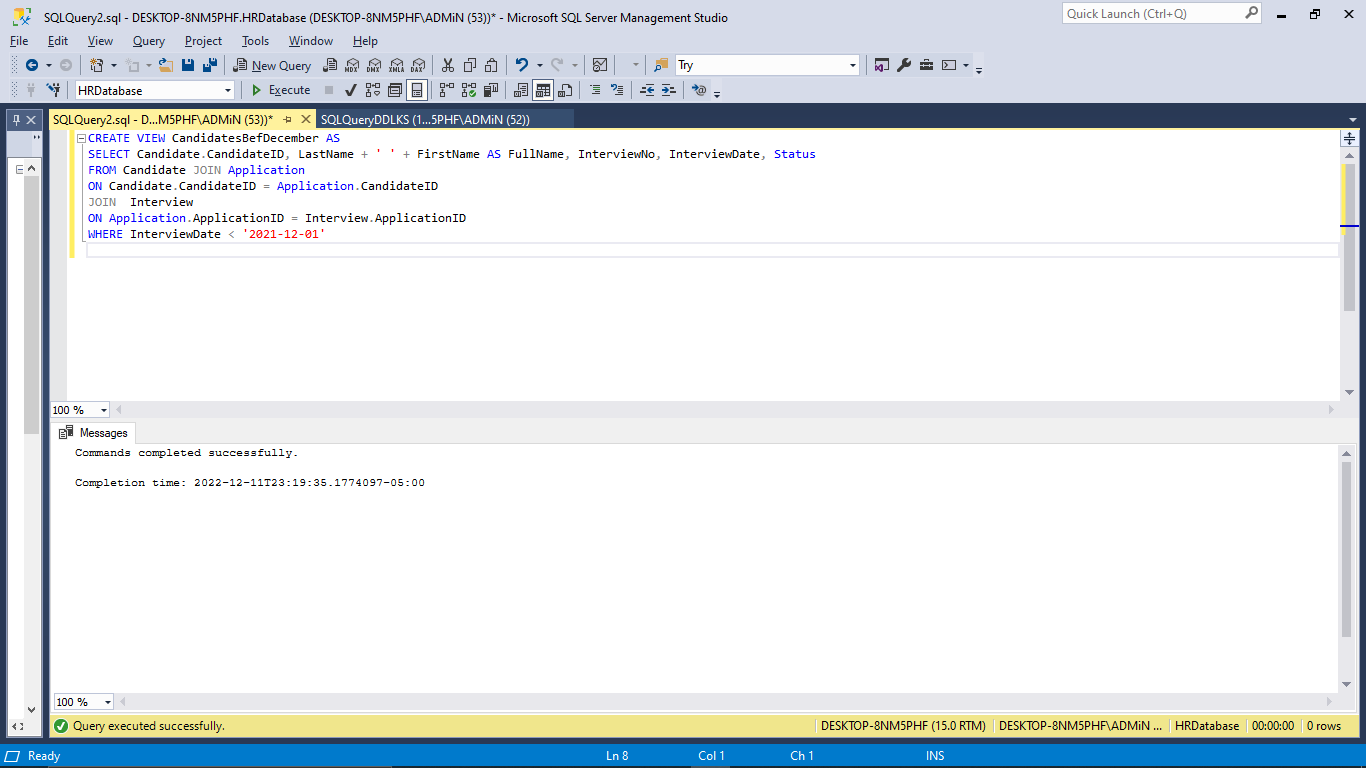
FROM Candidate JOIN Application

ON Candidate.CandidateID = Application.CandidateID

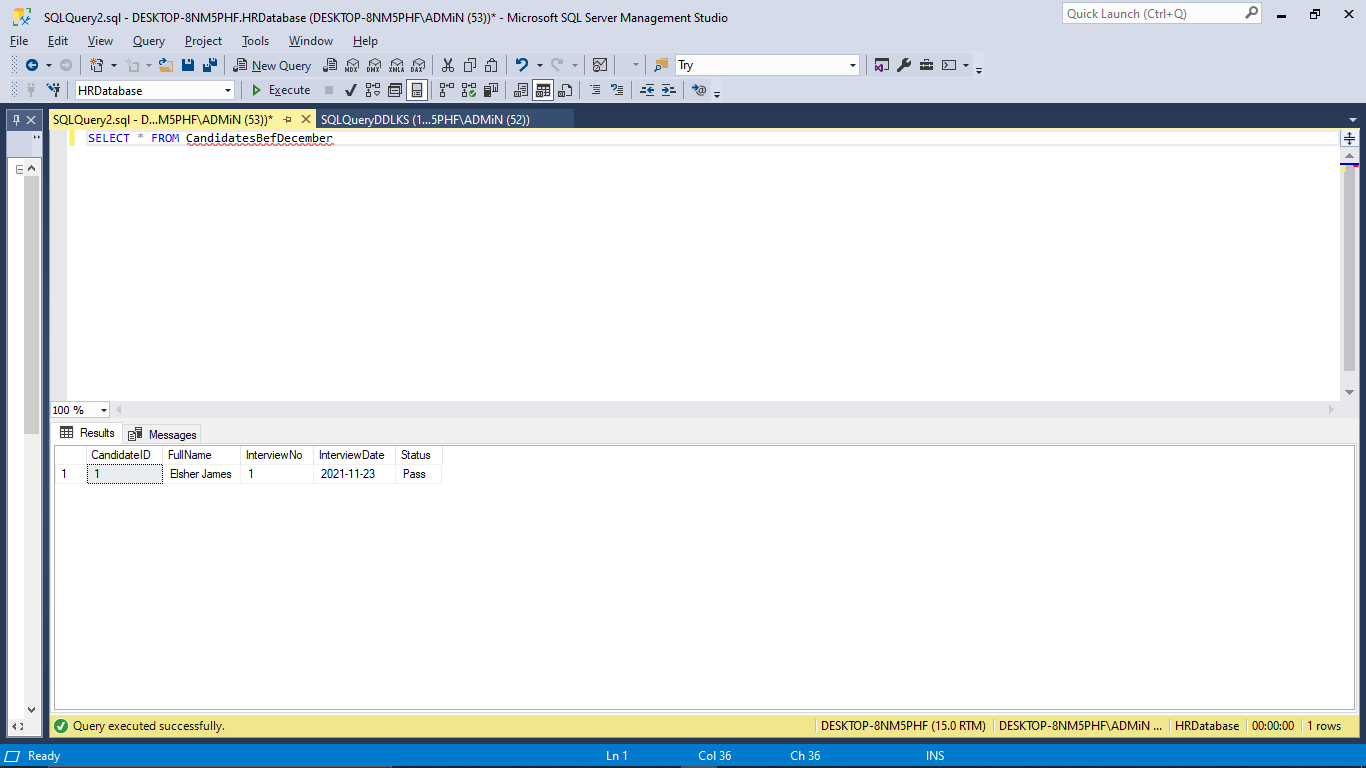
JOIN Interview

ON Application.ApplicationID = Interview.ApplicationID

WHERE InterviewDate < '2021-12-01'



SELECT \* FROM CandidatesBefDecember;



1. Create a view that shows the Candidate's total sum of the receipt cost that is the reimbursement amount.

**Comments:**

A VIEW is created called CandidateBefDecember using Create View Statement to retrieve the required data.

**Query:**

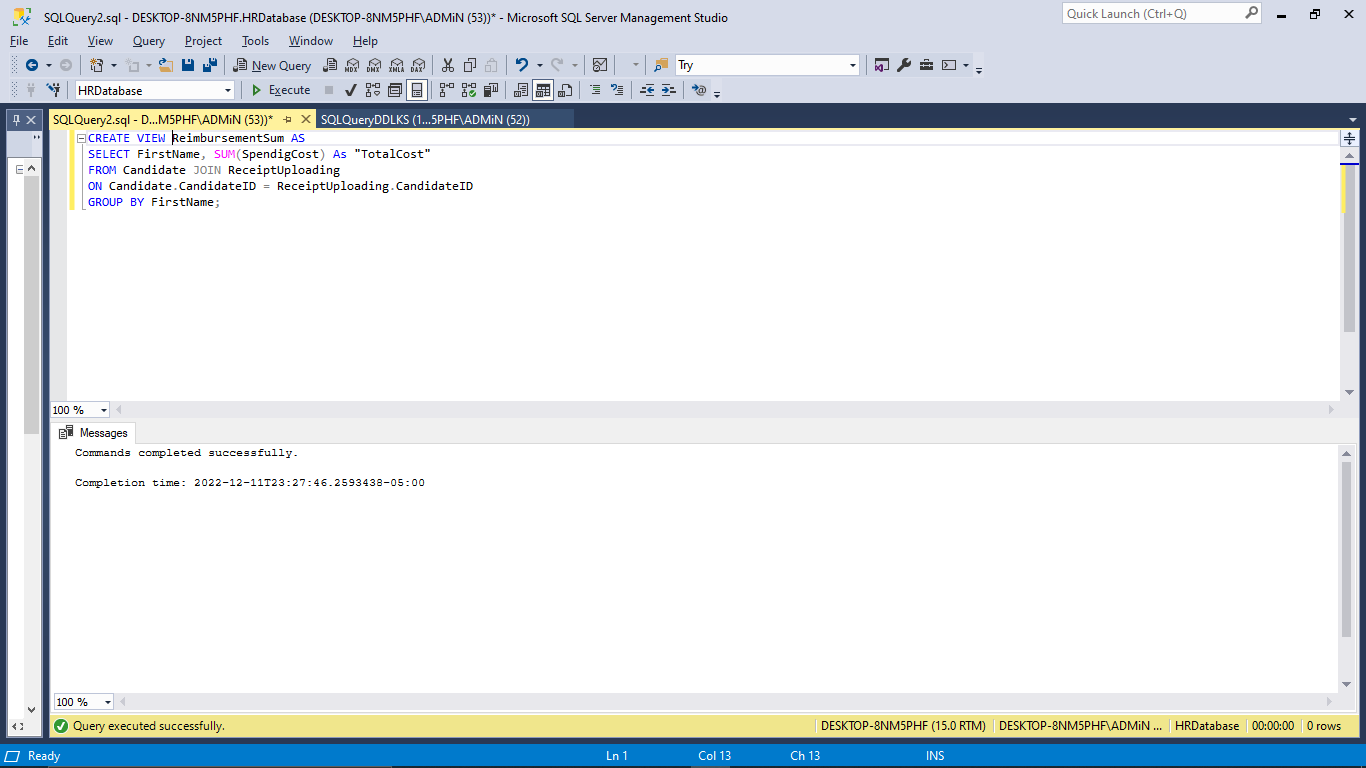
CREATE VIEW ReimbursementSum AS

SELECT FirstName, SUM(SpendigCost) As "TotalCost"

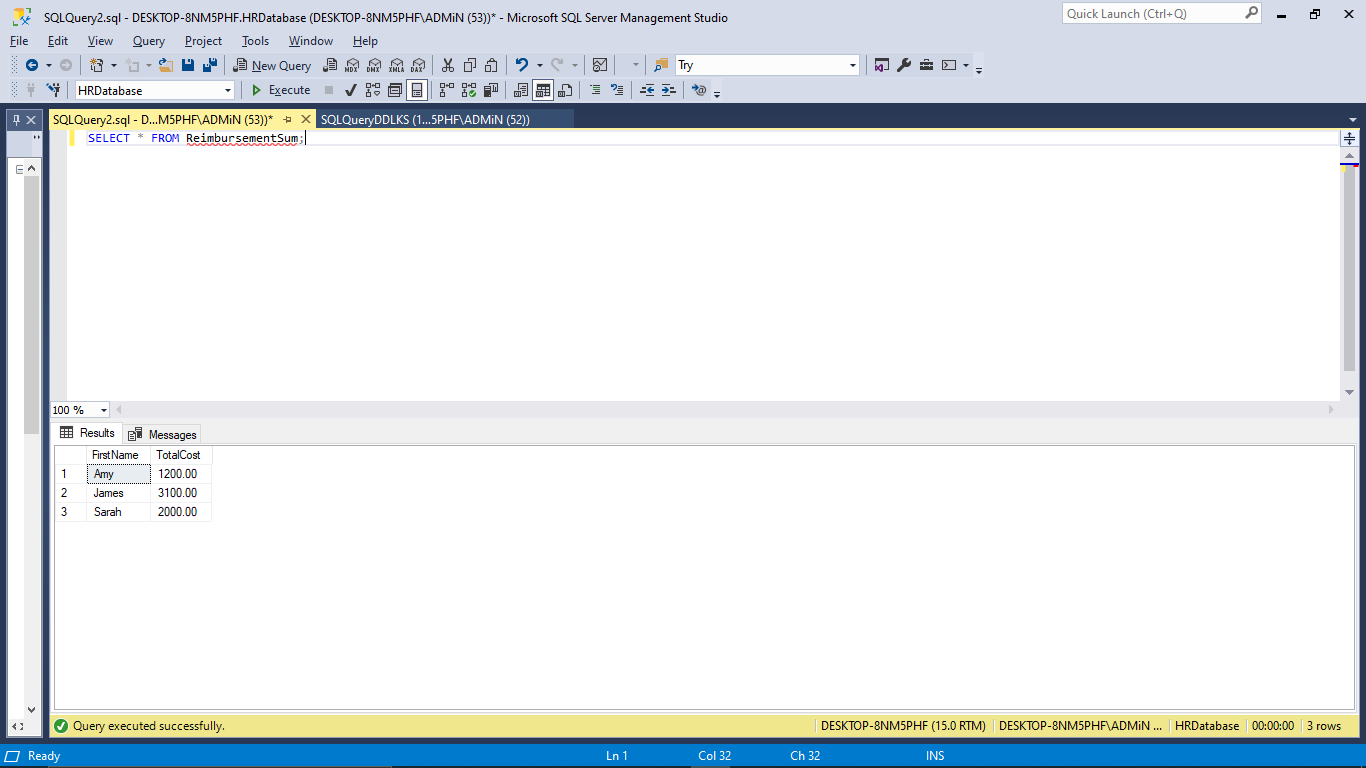
FROM Candidate JOIN ReceiptUploading

ON Candidate.CandidateID = ReceiptUploading.CandidateID

GROUP BY FirstName;



SELECT \* FROM ReimbursementSum;



# 

# Stored Procedures:

***Introduction:***

*A stored procedure is a subroutine that may be accessed by programs that use a relational database management system. These processes are kept in the database data dictionary.*

**1**. Create a stored procedure that contains all the candidates whose job id = 5;

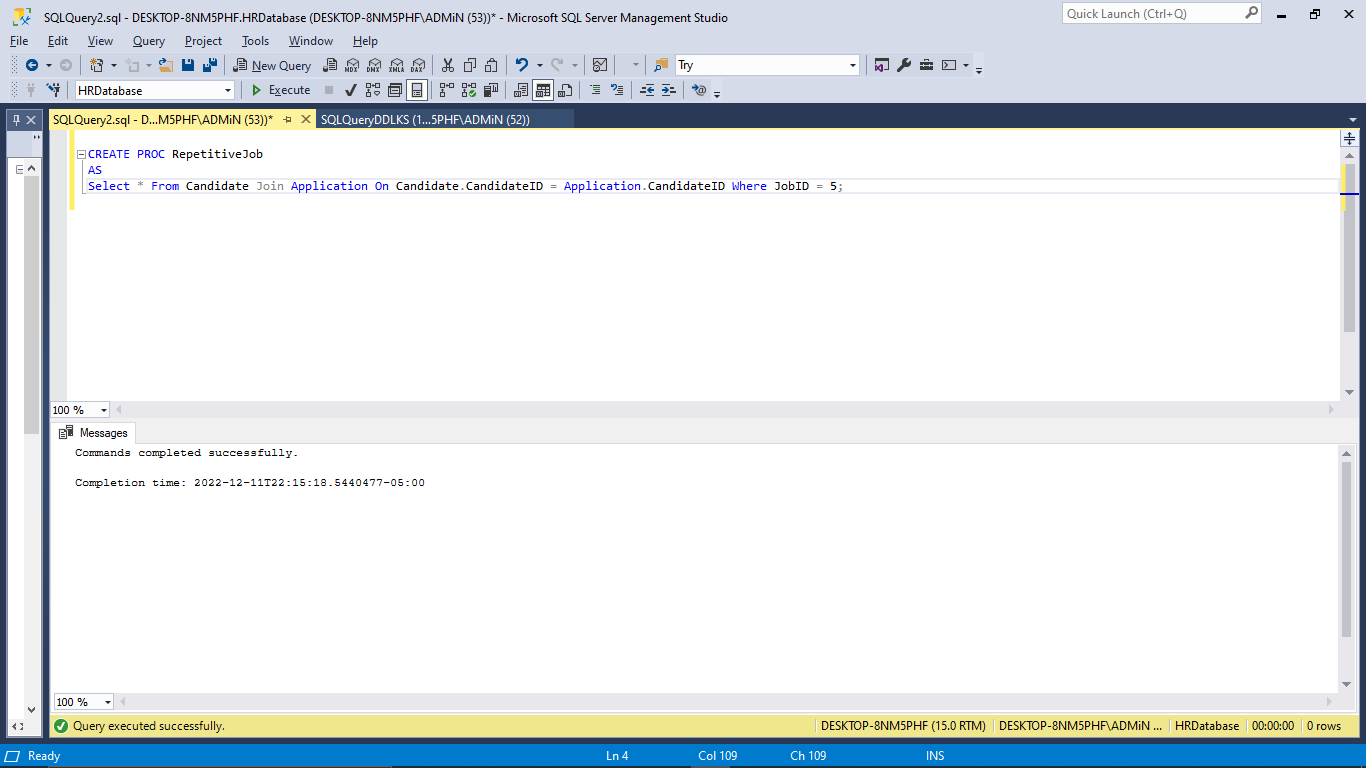
**Comment:** Created a Stored Procedure using Create PROC statement as per the requirement as executed it by using EXEC <Procedure name>

**Query:**

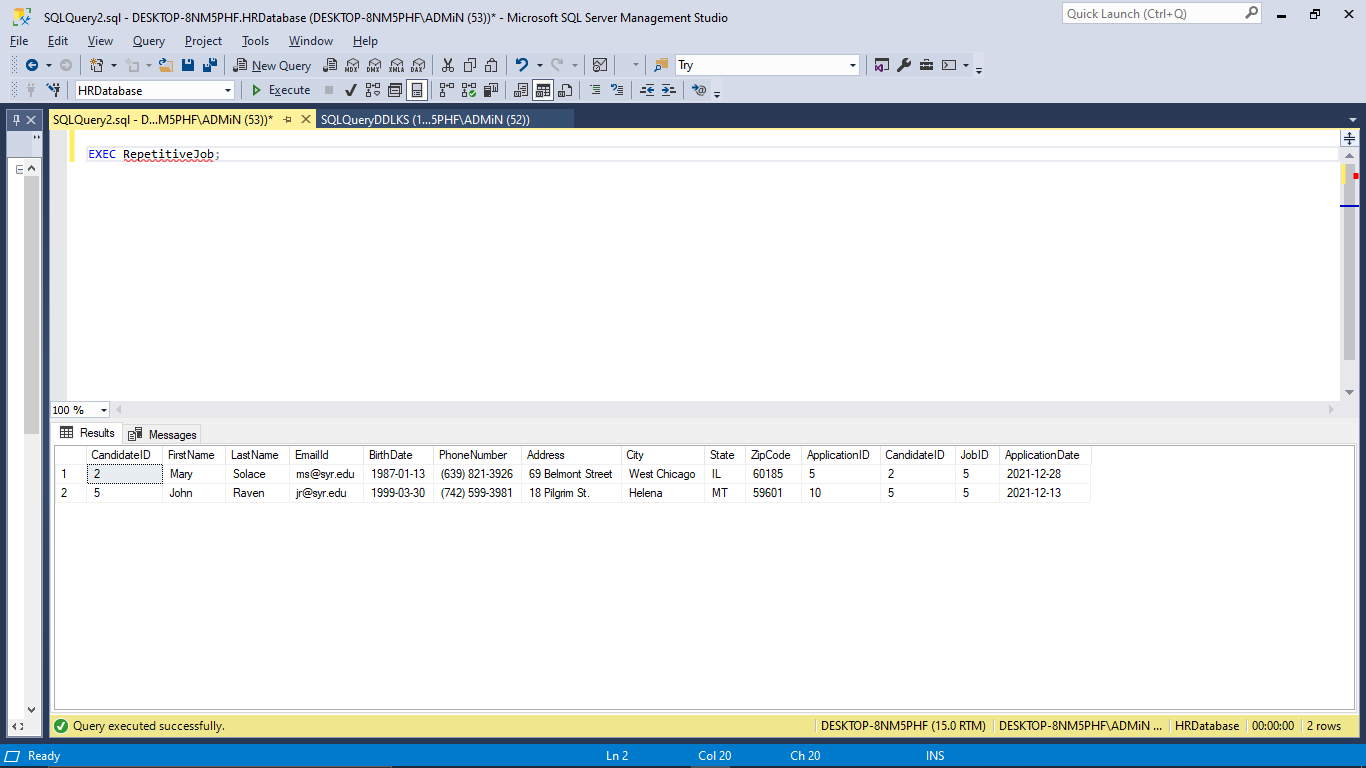
CREATE PROC RepetitiveJob

AS

Select \* From Candidate Join Application On Candidate.CandidateID = Application.CandidateID Where JobID = 5;



EXEC RepetitiveJob;



**2**. Create a stored procedure that contains all the Interviews that has been passed by the candidates;

**Comment:**

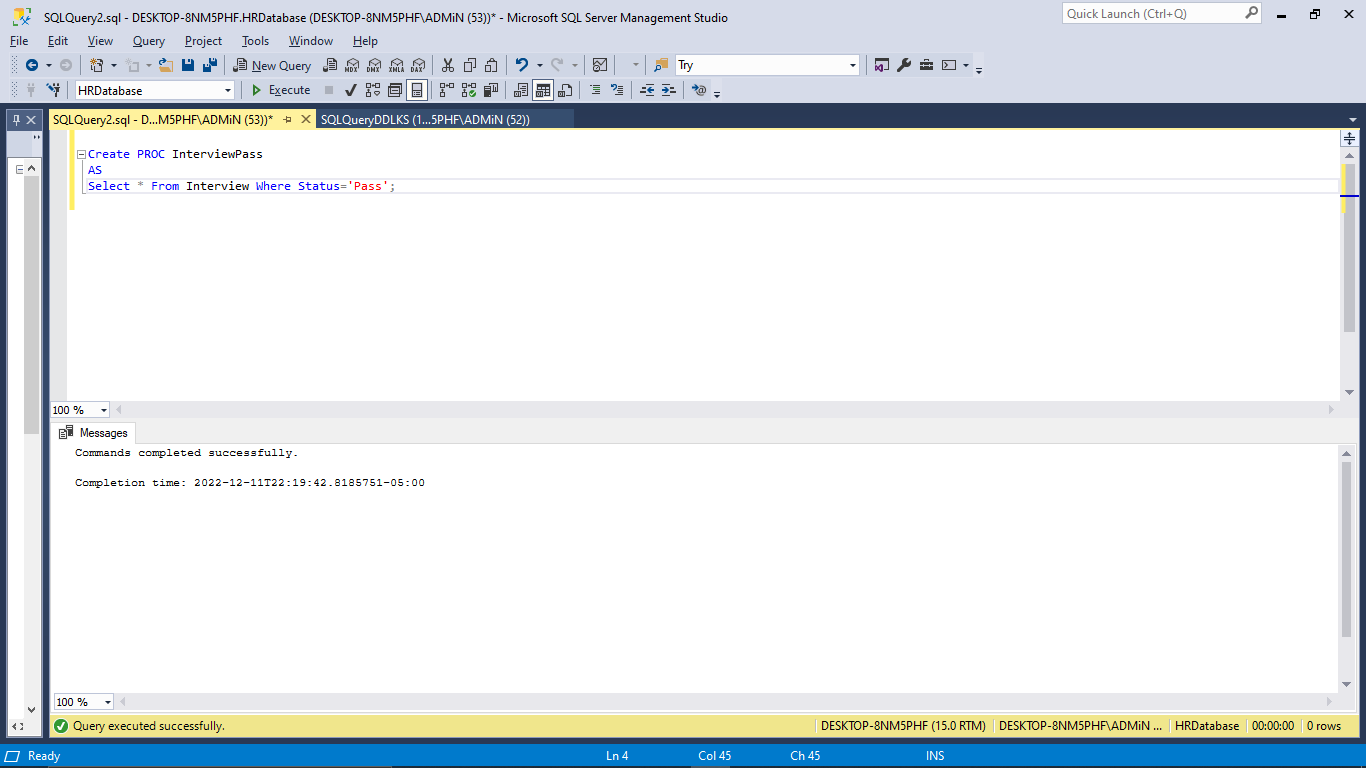
Created a Stored Procedure using Create PROC statement as per the requirement as executed it by using EXEC <Procedure name>

**Query:**

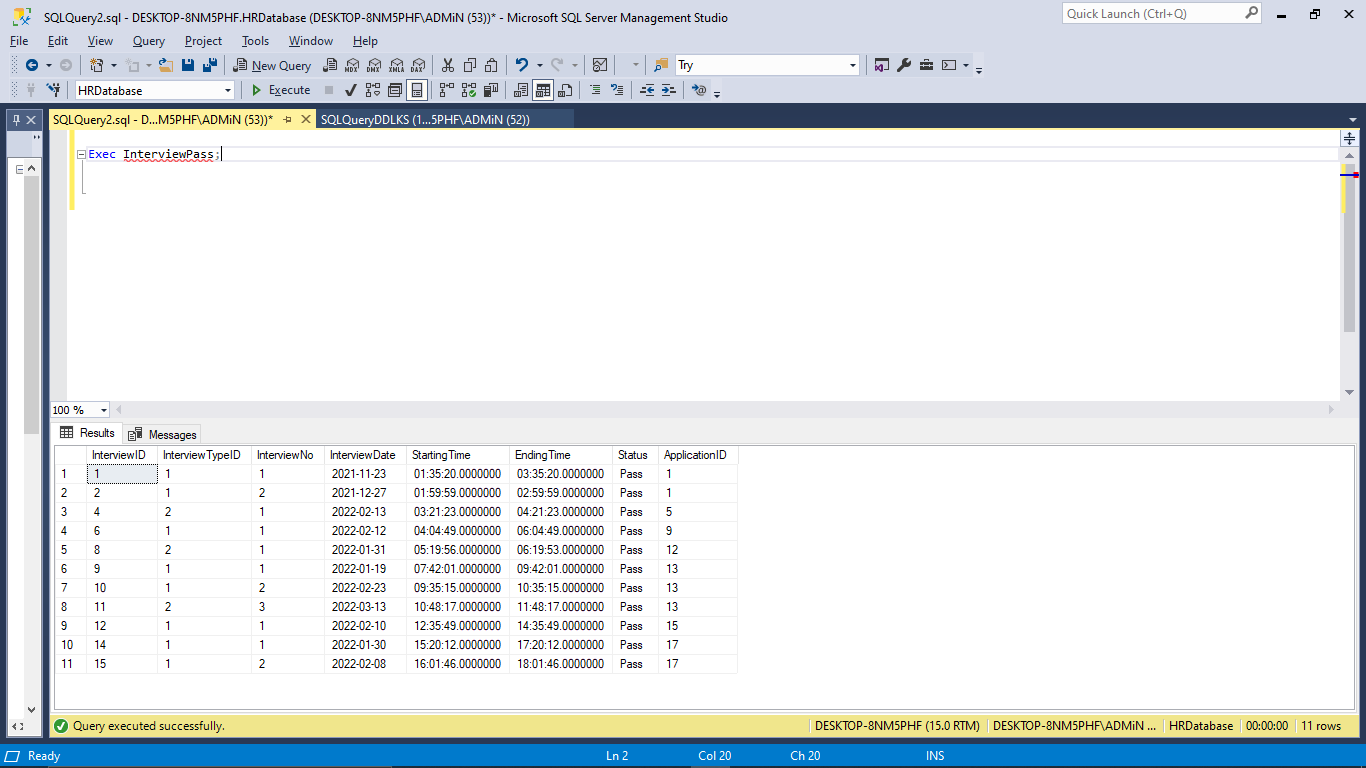
Create PROC InterviewPass

AS

Select \* From Interview Where Status='Pass';



Exec InterviewPass;



**3.**  Create a stored procedure to count the number of interviewers whose InterviewID = 4

**Comment:**

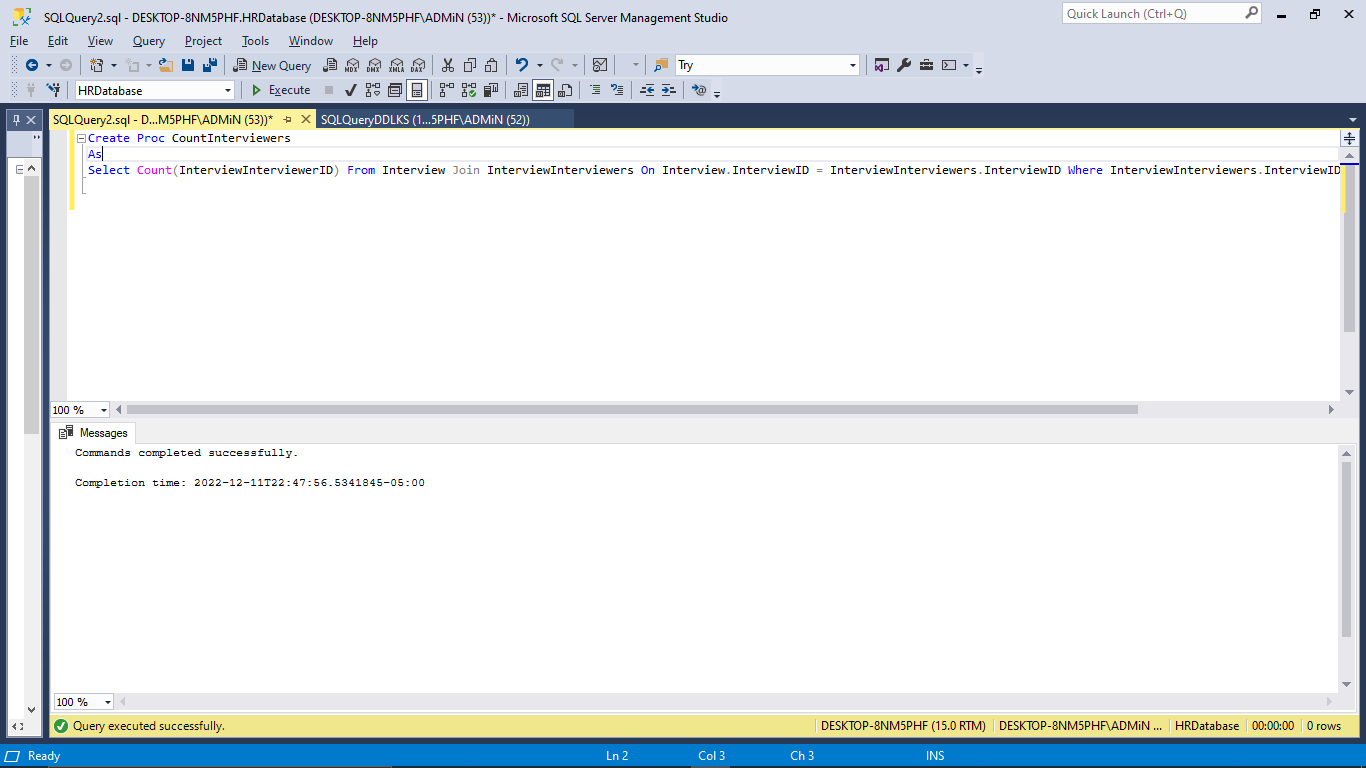
Created a Stored Procedure using Create PROC statement as per the requirement as executed it by using EXEC <Procedure name>

**Query:**

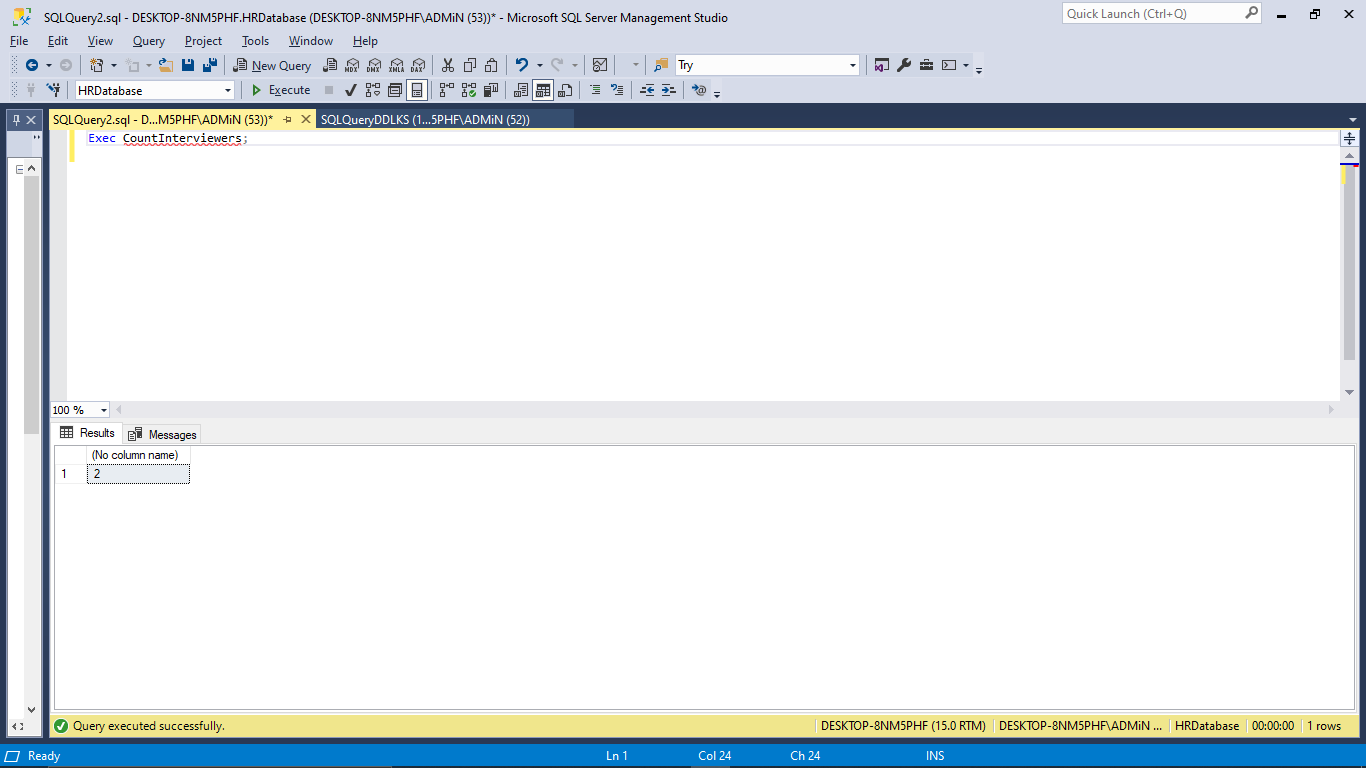
Create Proc CountInterviewers

As

Select Count(InterviewInterviewerID) From Interview Join InterviewInterviewers On Interview.InterviewID = InterviewInterviewers.InterviewID Where InterviewInterviewers.InterviewID=4;



Exec CountInterviewers;



**4.** Create a Stored Procedure which returns the job titles of the application of the candidates who are willing to work onsite.

**Comments:**

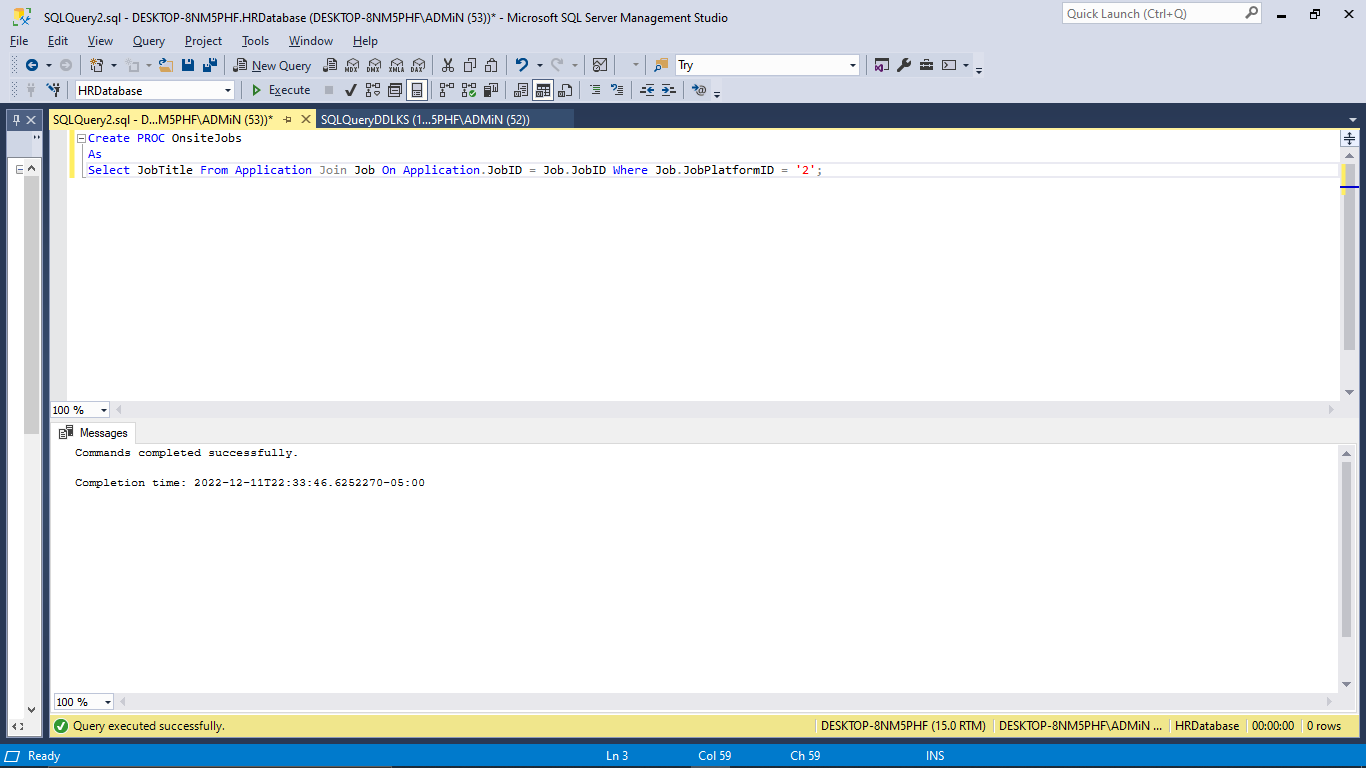
Created a Stored Procedure using Create PROC statement as per the requirement as executed it by using EXEC <Procedure name>

**Query:**

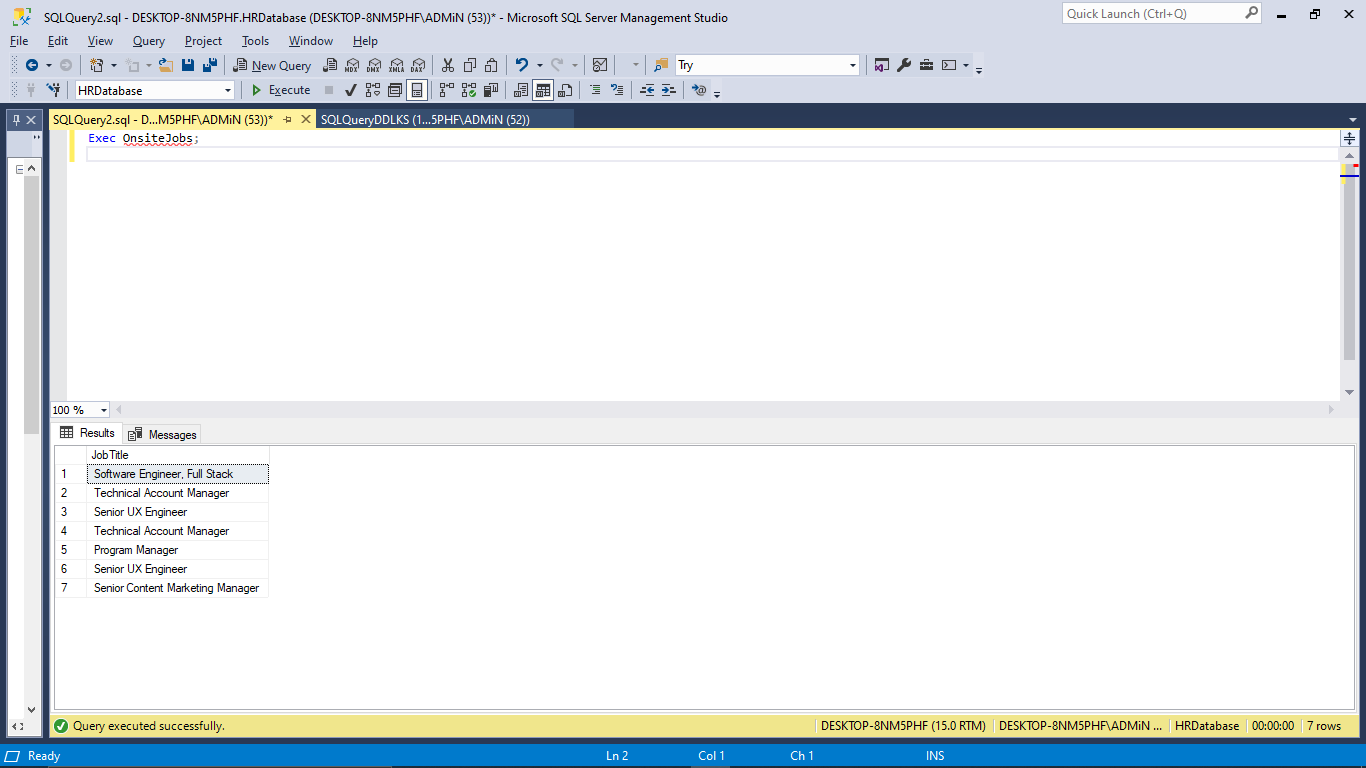
Create PROC OnsiteJobs

As

Select JobTitle From Application Join Job On Application.JobID = Job.JobID Where Job.JobPlatformID = '2';



Exec OnsiteJobs;



# Functions

1.Write a tabled function to return the candidates whose interview is in progress.

**Comments:**

Wrote a tabled function to get those candidates whose interview is still in process

Used HRDatabase

Declared the function by using Declare Statement and Inserted values using Insert Keyword.

**Query:**

USE HRDatabase;

DECLARE @InterviewInProcessCandidate table

(CandidateID int,

FName varchar(50),

LName varchar(50),

Result varchar(50));

INSERT @InterviewInProcessCandidate

SELECT Candidate.CandidateID, FirstName, LastName, Result

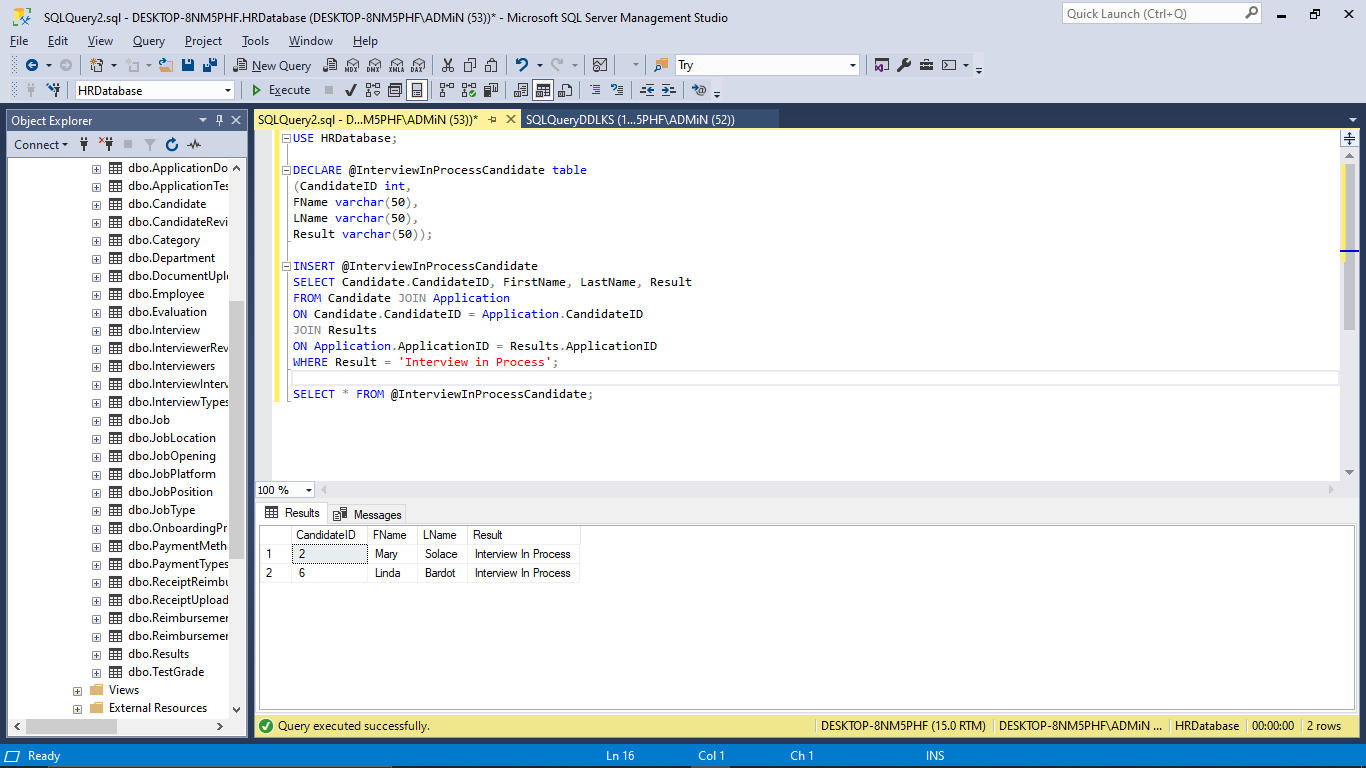
FROM Candidate JOIN Application

ON Candidate.CandidateID = Application.CandidateID

JOIN Results

ON Application.ApplicationID = Results.ApplicationID

WHERE Result = 'Interview in Process';

SELECT \* FROM @InterviewInProcessCandidate;

# **Conclusion**

*This project effectively uses SQL Server Management Studio to load and manipulate databases. The following topics are covered: fundamental queries, keywords, statements, operators, joins, subqueries, summary queries, aggregate functions, datatypes, and tables. Insertion, deletion, and updating are all types of data change. The subjects addressed include views, scripts, scalar and table valued functions, database implementation, and others. Additionally, creating a database using an E-R diagram is a crucial step. The bulk of the important SQL and database principles are the subject of this project.*