

CSE 581
INTRODUCTION TO DATABASE
MANAGEMENT SYSTEMS
Project - 2

TITLE
HR Recruitment Management System

Raj Nimish Shah
SU ID - 287463717
rshah26@syr.edu

Table of Content

SR NO	TITLE	PAGE NO
1	Abstract	3
2	Design <ul style="list-style-type: none"> 1. Introduction - Database Overview 2. Design Description with ER Diagram 3. Database Design Objects 	4
3	Implementation <ul style="list-style-type: none"> 1. Database creation 2. Tables creation 3. Inserting data in Tables 4. Retrieving data from Tables 	12
4	Testing <ul style="list-style-type: none"> 1. Views 2. Stored Procedures 3. Functions 4. Triggers 5. Transactions 6. Test Scripts 7. Business reports 	28
5	Conclusions	55
6	Appendix	56

ABSTRACT

Database management system (DBMS) controls the creation, maintenance, and use of databases. Databases can either be created, edited, deleted and read by the DBA team or other experts only. The purpose of DBMS is to offer a practical and efficient method for storing and retrieving database data. The importance of database management systems could be explained by the fact that they provide programmers, database administrators, and end users with a consolidated view of the data, alleviating applications and users the need to understand the actual location of the data. Internet requests and responses for certain types of data are managed through APIs (application program interfaces).

Business organizations utilize this HR Database to hire new employees. For a certain vacancy, the organization will publish a request for job applications. Based on the availability of openings and test results, the employer will assess each applicant's eligibility and determine if they are qualified for the employment. This system will take care of employee hiring as well as the administrative module. This database contains each and every essential DBMS concept, including DDL, DML, Views, Role Assignment, Stored Procedures, Transactions, Scripts, Functions, Triggers, and Business Reports. My Human Resource Database are created keeping into consideration Database Design Structure(Normalization).

DESIGN

a. Introduction

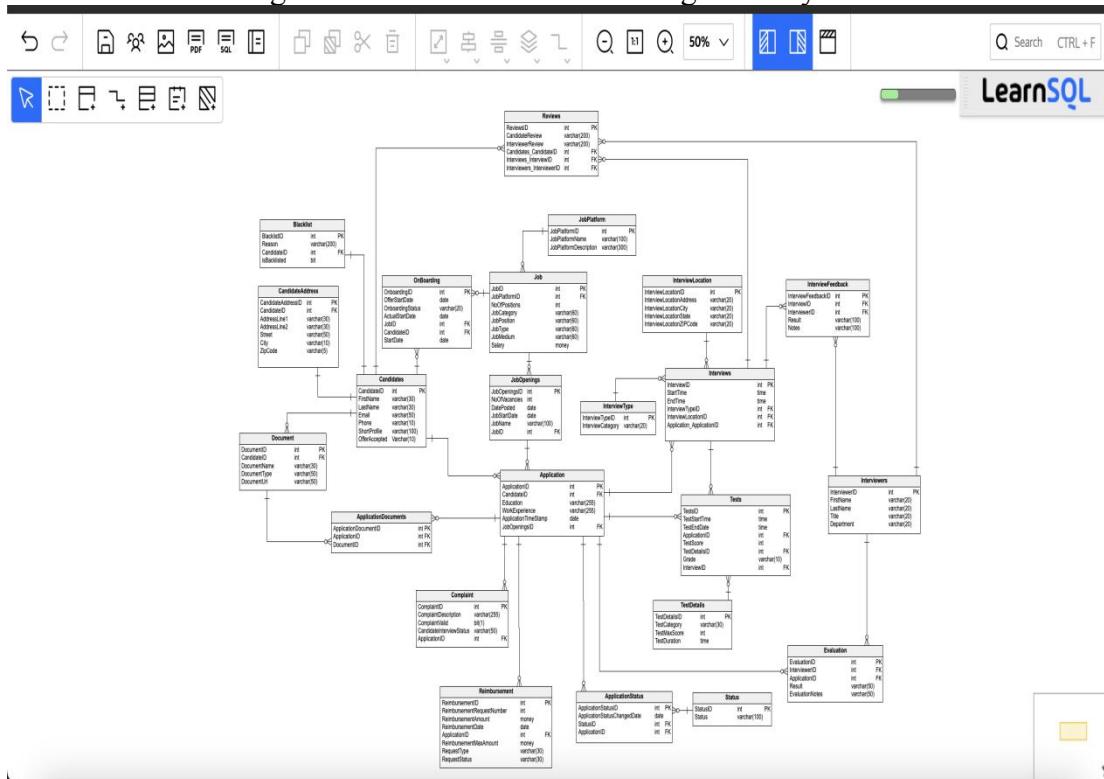
Employers are hired by companies with the assistance of recruiting firms. In rare circumstances, the corporation may also recruit employees directly. The person in charge of hiring commences the hiring process. There are many different phases in this process, including the first screening, a written exam, the first interview, the follow-up interview, the actual hiring decision, etc.

The developed Human Resource Database System includes the following steps for recruiting new employees:

- 1) Candidates apply for a particular job opening.
- 2) Candidates get classified based on their overall job application. They may be directly called for an interview or can be sent a test link.
- 3) Upon either of the scenario completion, candidates are graded and feedback is also provided by the interviewer.
- 6) Evaluation depends on tests and interviews.
- 7) The ultimate decision will then be made based on the hiring decision.
- 8) The interviews can also take place in person or online, depending on the candidate's ability and the company's requirements.
- 9) Re-interviews may be scheduled if a rejected candidate complains about the interview procedure, depending on whether the complaint is justified or not.
- 10) Reimbursement is granted to the candidate who comes for an onsite interview or test if he/she fills out the reimbursement form.
- 11) Candidate and the interviewer can post reviews regarding each other during each interview.

b. Design Description with ER Diagram :

Below is the ER Diagram for HR Recruitment Management System:



1. **Candidates table** - Conserves candidate's personal information.
2. **CandidateAddress table** - Stores Candidate's Address.
3. **Application table** - Saves application data of candidate such as date of application, work history, education, etc.
4. **Document table** - Stores document details like document name, url and type which is required for the job application.(This include documents like CV, Resume).
5. **ApplicationDocument table** - Enacts as an linking table between Application and Documents table.
6. **Complaint table** - If a rejected applicant finds the interview was unfair, he or she can file a complaint, and the information regarding the complaint is saved in this table.
7. **Reimbursement table** : Keeps data about candidate's expenses. Column included here are reimbursement status, request type, request date, request number and amount.
8. **Interviewer table** - Saves Interviewer's details who are responsible for the recruiting employees like FirstName, Lastname, Title and Department .
9. **Interview table** - Stores Interview details such as interview's start time, end time, location and interview type.
10. **InterviewType** - It saves information about interviewtype either onsite or online.
11. **InterviewLocation** - It contains detail information like interview address, city, state and ZipCode.
12. **InterviewFeedback table** - Each applicant receives interview result and notes from the interviewer based on their interview, which is saved in the InterviewFeedback table. Moreover, it acts as a linking table.
13. **Evaluation table** - The interviewer completes and stores the evaluation like evaluationresult and notes in this table.
14. **ApplicationStatus table** - It stores details like application status changed date, statusid of an applicant who has applied for a job position. It acts as a linking table between the Application table and Status table.
15. **Status table** - Status like accepted, declined or negotiating are stored in this status column.
16. **Tests table** - Saves the start and end time of the examination, the candidate's scores, and the grade attained based on the results.
17. **TestDetails table** - Contains information on the test's category, time, and maximum possible score.
18. **Job table** - Contains information of the job's details, including its type, medium, and number of positions.
19. **JobOpenings table** - Keep information about the job description as well as the number of positions.
20. **JobPlatform table** - It saves information about platforms either onsite or online.
21. **Onboarding table** - Stores information about the onboarding process like start date, onboarding status.
22. **Reviews table** - Enacts as an linking table between Candidates , Interview and Interviewer table. Each candidate can give review for an interviewer for each interview. Similarly for each interview each interviewer can give review for an candidate.
23. **Blacklist table** - Stores the Blacklist reason for a particular applicant.

Database Design Objects

The following is a list of the primary keys (PK), data types, nullables, and relationships table entries that are utilized in my project:

1) Candidates Table

```
CandidateID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
FirstName VARCHAR(30) NOT NULL,  
LastName VARCHAR(30) NULL,  
Email VARCHAR(50) NOT NULL,  
Phone VARCHAR(10) NOT NULL,  
ShortProfile VARCHAR(100) NULL DEFAULT 'Not Given',  
OfferAccepted VARCHAR(10) NULL
```

2) CandidateAddress Table

```
CandidateAddressID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
CandidateID INT NOT NULL,  
AddressLine1 VARCHAR(30) NOT NULL,  
AddressLine2 VARCHAR(30) NULL,  
Street VARCHAR(50) NULL,  
City VARCHAR(10) NOT NULL,  
ZipCode VARCHAR(5) NOT NULL,  
FOREIGN KEY(CandidateID) REFERENCES Candidates(CandidateID)
```

3) JobPlatform Table

```
JobPlatformID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
JobPlatformName VARCHAR(100) NOT NULL,  
JobPlatformDescription VARCHAR(300) NOT NULL
```

4) Job Table

```
JobID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
JobPlatformID INT NOT NULL,  
NoOfPositions INT NOT NULL,  
JobCategory VARCHAR(60) NOT NULL,  
JobPosition VARCHAR(60) NOT NULL,  
JobType VARCHAR(60) NOT NULL,  
JobMedium VARCHAR(60) NOT NULL,  
Salary MONEY NULL  
FOREIGN KEY(JobPlatformID) REFERENCES JobPlatform(JobPlatformID)
```

5) JobOpenings Table

```
JobOpeningsID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
NoOfVacancies INT NOT NULL,  
DatePosted DATE NOT NULL,  
JobStartDate DATE NOT NULL,  
JobName VARCHAR(100) NOT NULL,  
JobID INT NOT NULL,  
FOREIGN KEY(JobID) REFERENCES Job(JobID)
```

6) Documents Table

```
DocumentID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
CandidateID INT NOT NULL,  
DocumentName VARCHAR(30) NOT NULL,  
DocumentType VARCHAR(30) NOT NULL,  
DocumentUrl VARCHAR(50) NOT NULL  
FOREIGN KEY(CandidateID) REFERENCES Candidates(CandidateID)
```

7) OnBoarding Table

```
OnBoardingID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
CandidateID INT NOT NULL,  
OnboardingStatus VARCHAR(20) NULL,  
ActualStartDate DATETIME NULL,  
JobID INT NOT NULL,  
FOREIGN KEY(JobID) REFERENCES Job(JobID),  
FOREIGN KEY(CandidateID) REFERENCES Candidates(CandidateID)
```

8) Application Table

```
ApplicationID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
CandidateID INT NOT NULL,  
Education VARCHAR(255) NOT NULL,  
WorkExperience VARCHAR(255) DEFAULT NULL,  
ApplicationTimeStamp DATE NOT NULL,  
JobOpeningsID INT NOT NULL  
FOREIGN KEY(JobOpeningsID) REFERENCES JobOpenings(JobOpeningsID),  
FOREIGN KEY(CandidateID) REFERENCES Candidates(CandidateID)
```

9) ApplicationDocuments Table

```
ApplicationDocumentID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
DocumentID INT NOT NULL,  
ApplicationID INT NOT NULL
```

FOREIGN KEY(ApplicationID) REFERENCES Application(ApplicationID),
FOREIGN KEY(DocumentID) REFERENCES Documents(DocumentID)

10) Reimbursement Table

ReimbursementID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
ReimbursementRequestNumber INT NOT NULL,
ReimbursementAmount MONEY NOT NULL,
ReimbursementDate DATE NOT NULL,
ReimbursementMaxAmount MONEY DEFAULT 30000 NOT NULL,
ApplicationID INT NOT NULL,
RequestType VARCHAR(30) NOT NULL,
RequestStatus VARCHAR(30) NOT NULL
FOREIGN KEY(ApplicationID) REFERENCES Application(ApplicationID)

11) Complaint Table

ComplaintID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
ComplaintDescription VARCHAR(255) NOT NULL,
ComplaintValid BIT NOT NULL,
ComplaintInterviewStatus VARCHAR(50) NOT NULL,
ApplicationID INT NOT NULL
FOREIGN KEY(ApplicationID) REFERENCES Application(ApplicationID)

12) ApplicationStatus Table

ApplicationStatusID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
ApplicationStatusChangedDate DATETIME NOT NULL,
StatusID INT NOT NULL,
ApplicationID INT NOT NULL
FOREIGN KEY(ApplicationID) REFERENCES Application(ApplicationID)

13) Status Table

StatusID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
Status VARCHAR(100) NOT NULL

14) InterviewLocation Table

InterviewLocationID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
InterviewLocationAddress VARCHAR(20) NULL,
InterviewLocationCity VARCHAR(20) NULL,
InterviewLocationState VARCHAR(20) NULL,
InterviewLocationZIPCode VARCHAR(20) NULL

15) InterviewType Table

InterviewTypeID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
InterviewCategory VARCHAR(20) NOT NULL

16) Interviews Table

InterviewID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
StartTime TIME NOT NULL,
EndTime TIME NOT NULL,
ApplicationID INT NOT NULL,
InterviewTypeID INT NOT NULL,
InterviewLocationID INT NOT NULL,
FOREIGN KEY(ApplicationID) REFERENCES Application(ApplicationID),
FOREIGN KEY(InterviewLocationID) REFERENCES
InterviewLocation(InterviewLocationID),
FOREIGN KEY(InterviewTypeID) REFERENCES InterviewType(InterviewTypeID)

17) TestDetails Table

TestDetailsID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
TestCategory VARCHAR(30) NOT NULL,
TestMaxScore INT NOT NULL,
TestDuration TIME NULL

18) Tests Table

TestsID int IDENTITY(1,1) NOT NULL PRIMARY KEY,
TestStartTime TIME NOT NULL,
TesEndTime TIME NOT NULL,
TestDetailsID INT NOT NULL,
TestScore INT NOT NULL,
TestGrade varchar(10) NOT NULL,
ApplicationID INT NOT NULL,
InterviewID INT NOT NULL
FOREIGN KEY(InterviewID) REFERENCES Interviews(InterviewID),
FOREIGN KEY(TestDetailsID) REFERENCES TestDetails(TestDetailsID),
FOREIGN KEY(ApplicationID) REFERENCES Application(ApplicationID)

19) Interviewers Table

InterviewerID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
FirstName VARCHAR(20) NOT NULL,

LastName VARCHAR(20) NOT NULL,
Title VARCHAR(20) NULL,
Department VARCHAR(20) NOT NULL

20) InterviewFeedback Table

InterviewFeedbackID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
InterviewID INT NOT NULL,
InterviewerID INT NOT NULL,
Result VARCHAR(100) NOT NULL,
Notes VARCHAR(100) NULL
FOREIGN KEY(InterviewID) REFERENCES Interviews(InterviewID),
FOREIGN KEY(InterviewerID) REFERENCES Interviewers(InterviewerID)

21) Evaluation Table

EvaluationID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
InterviewerID INT NOT NULL,
ApplicationID INT NOT NULL,
Result VARCHAR(50) NOT NULL,
EvaluationNotes VARCHAR(50) NULL,
FOREIGN KEY(InterviewerID) REFERENCES Interviewers(InterviewerID),
FOREIGN KEY(ApplicationID) REFERENCES Application(ApplicationID)

22) Reviews Table

ReviewsID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
CandidateReview VARCHAR(200) NULL,
InterviewerReview VARCHAR(200) NULL,
InterviewID INT NOT NULL,
InterviewerID INT NOT NULL,
CandidateID INT NOT NULL,
FOREIGN KEY(CandidateID) REFERENCES Candidates(CandidateID),
FOREIGN KEY(InterviewID) REFERENCES Interviews(InterviewID),
FOREIGN KEY(InterviewerID) REFERENCES Interviewers(InterviewerID)

23) Blacklist Table

BlacklistID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
Reason VARCHAR(200) NULL,
CandidateID INT NOT NULL,
isBlacklisted BIT NOT NULL

FOREIGN KEY(CandidateID) REFERENCES Candidates(CandidateID)

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

1. One-to-One relationship between Candidates and CandidateAddress
2. One-to-One relationship between Candidates and Blacklist
3. One-to-Many relationship between Candidates and Application
4. One-to-Many relationship between Candidates and Document
5. One-to-Many relationship between Application and ApplicationDocument
6. One-to-Many relationship between Document and ApplicationDocument
7. One-to-Many relationship between Application and Complaint
8. One-to-Many relationship between Application and Reimbursement
9. One-to-Many relationship between Application and ApplicationStatus
10. Many-to-One relationship between ApplicationStatus and Status
11. Many-to-One relationship between Application and JobOpening
12. Many-to-One relationship between JobOpening and Job
13. Many-to-One relationship between Job and JobPlatform
14. One-to-Many relationship between Candidate and Onboarding
15. One-to-Many relationship between Job and Onboarding
16. One-to-Many relationship between Application and Tests
17. Many-to-One relationship between Tests and TestDetails
18. One-to-Many relationship between Application and Interviews
19. One-to-Many relationship between Interviews and Tests
20. Many-to-One relationship between Interviews and InterviewType
21. Many-to-One relationship between Interviews and InterviewLocation
22. One-to-Many relationship between Interviews and InterviewFeedback
23. One-to-Many relationship between Interviewers and InterviewFeedback
24. One-to-Many relationship between Interviewers and Evaluation
25. Many-to-One relationship between Evaluation and Application
26. One-to-Many relationship between Interviews and Reviews
27. One-to-Many relationship between Interviewers and Reviews
28. One-to-Many relationship between Candidate and Reviews

IMPLEMENTATION

1. Database creation

Code for HR_RajShah database creation

```
USE master
```

```
GO
```

```
IF DB_ID('HR_RajShah') IS NOT NULL
DROP DATABASE HR_RajShah
GO
CREATE DATABASE HR_RajShah;
```

Schema

Code for HR schema creation

```
CREATE SCHEMA HR;
```

2. Tables creation

Creation of all the tables present in the database.

Code for Candidates table creation

```
CREATE TABLE HR.Candidates(
CandidateID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
FirstName VARCHAR(30) NOT NULL,
LastName VARCHAR(30) NULL,
Email VARCHAR(50) NOT NULL,
Phone VARCHAR(10) NOT NULL,
ShortProfile VARCHAR(100) NULL DEFAULT 'Not Given',
OfferAccepted VARCHAR(10) NULL);
```

Code for CandidateAddress table creation

```
CREATE TABLE HR.CandidateAddress(
CandidateAddressID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
CandidateID INT NOT NULL,
AddressLine1 VARCHAR(30) NOT NULL,
AddressLine2 VARCHAR(30) NULL,
Street VARCHAR(50) NULL,
City VARCHAR(10) NOT NULL,
ZipCode VARCHAR(5) NOT NULL,
FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID)
);
```

Code for JobPlatform table creation

```
CREATE TABLE HR.JobPlatform(
```

```
JobPlatformID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
JobPlatformName VARCHAR(100) NOT NULL,
JobPlatformDescription VARCHAR(300) NOT NULL
);
```

Code for Job table creation

```
CREATE TABLE HR.Job(
JobID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
JobPlatformID INT NOT NULL,
NoOfPositions INT NOT NULL,
JobCategory VARCHAR(60) NOT NULL,
JobPosition VARCHAR(60) NOT NULL,
JobType VARCHAR(60) NOT NULL,
JobMedium VARCHAR(60) NOT NULL,
Salary MONEY NULL
FOREIGN KEY(JobPlatformID) REFERENCES HR.JobPlatform(JobPlatformID),
);
```

Code for JobOpenings table creation

```
CREATE TABLE HR.JobOpenings(
JobOpeningsID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
NoOfVacancies INT NOT NULL,
DatePosted DATE NOT NULL,
JobStartDate DATE NOT NULL,
JobName VARCHAR(100) NOT NULL,
JobID INT NOT NULL,
FOREIGN KEY(JobID) REFERENCES HR.Job(JobID)
);
```

Code for Documents table creation

```
CREATE TABLE HR.Documents(
DocumentID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
CandidateID INT NOT NULL,
DocumentName VARCHAR(30) NOT NULL,
DocumentType VARCHAR(30) NOT NULL,
DocumentUrl VARCHAR(50) NOT NULL
FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID)
);
```

Code for OnBoarding table creation

```
CREATE TABLE HR.OnBoarding(
    OnBoardingID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
    CandidateID INT NOT NULL,
    OnboardingStatus VARCHAR(20) NULL,
    ActualStartDate DATETIME NULL,
    JobID INT NOT NULL,
    StartDate DATETIME NOT NULL,
    FOREIGN KEY(JobID) REFERENCES HR.Job(JobID),
    FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID)
);
```

Code for Application table creation

```
CREATE TABLE HR.Application(
    ApplicationID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
    CandidateID INT NOT NULL,
    Education VARCHAR(255) NOT NULL,
    WorkExperience VARCHAR(255) DEFAULT NULL,
    ApplicationTimeStamp DATE NOT NULL,
    JobOpeningsID INT NOT NULL
    FOREIGN KEY(JobOpeningsID) REFERENCES HR.JobOpenings(JobOpeningsID),
    FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID)
);
```

Code for ApplicationDocuments table creation

```
CREATE TABLE HR.ApplicationDocuments(
    ApplicationDocumentID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
    DocumentID INT NOT NULL,
    ApplicationID INT NOT NULL
    FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID),
    FOREIGN KEY(DocumentID) REFERENCES HR.Documents(DocumentID)
);
```

Code for Reimbursement table creation

```
CREATE TABLE HR.Reimbursement(
    ReimbursementID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
    ReimbursementRequestNumber INT NOT NULL,
    ReimbursementAmount MONEY NOT NULL,
    ReimbursementDate DATE NOT NULL,
    ReimbursementMaxAmount MONEY DEFAULT 30000 NOT NULL,
);
```

```
ApplicationID INT NOT NULL,  
RequestType VARCHAR(30) NOT NULL,  
RequestStatus VARCHAR(30) NOT NULL  
FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID)  
);
```

Code for Complaint table creation

```
CREATE TABLE HR.Complaint(  
ComplaintID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
ComplaintDescription VARCHAR(255) NOT NULL,  
ComplaintValid BIT NOT NULL,  
ComplaintInterviewStatus VARCHAR(50) NOT NULL,  
ApplicationID INT NOT NULL  
FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID)  
);
```

Code for ApplicationStatus table creation

```
CREATE TABLE HR.ApplicationStatus(  
ApplicationStatusID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
ApplicationStatusChangedDate DATETIME NOT NULL,  
StatusID INT NOT NULL,  
ApplicationID INT NOT NULL  
FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID)  
);
```

Code for Status table creation

```
CREATE TABLE HR.Status(  
StatusID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
Status VARCHAR(100) NOT NULL  
);
```

Code for InterviewLocation table creation

```
CREATE TABLE HR.InterviewLocation(  
InterviewLocationID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
InterviewLocationAddress VARCHAR(20) NULL,  
InterviewLocationCity VARCHAR(20) NULL,  
InterviewLocationState VARCHAR(20) NULL,  
InterviewLocationZIPCode VARCHAR(20) NULL  
);
```

Code for InterviewType table creation

```
CREATE TABLE HR.InterviewType(
InterviewTypeID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
InterviewCategory VARCHAR(20) NOT NULL
);
```

Code for Interviews table creation

```
CREATE TABLE HR.Interviews(
InterviewID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
StartTime TIME NOT NULL,
EndTime TIME NOT NULL,
ApplicationID INT NOT NULL,
InterviewTypeID INT NOT NULL,
InterviewLocationID INT NOT NULL,
FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID),
FOREIGN KEY(InterviewLocationID) REFERENCES
HR.InterviewLocation(InterviewLocationID),
FOREIGN KEY(InterviewTypeID) REFERENCES
HR.InterviewType(InterviewTypeID)
);
```

Code for TestDetails table creation

```
CREATE TABLE HR.TestDetails(
TestDetailsID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
TestCategory VARCHAR(30) NOT NULL,
TestMaxScore INT NOT NULL,
TestDuration TIME NULL
);
```

Code for Tests table creation

```
CREATE TABLE HR.Tests(
TestsID int IDENTITY(1,1) NOT NULL PRIMARY KEY,
TestStartTime TIME NOT NULL,
TestEndTime TIME NOT NULL,
TestDetailsID INT NOT NULL,
TestScore INT NOT NULL,
TestGrade varchar(10) NOT NULL,
ApplicationID INT NOT NULL,
InterviewID INT NOT NULL
)
```

```

FOREIGN KEY(InterviewID) REFERENCES HR.Interviews(InterviewID),
FOREIGN KEY(TestDetailsID) REFERENCES HR.TestDetails(TestDetailsID),
FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID)
);

```

Code for Interviewers table creation

```

CREATE TABLE HR.Interviewers(
InterviewerID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
FirstName VARCHAR(20) NOT NULL,
LastName VARCHAR(20) NOT NULL,
Title VARCHAR(20) NULL,
Department VARCHAR(20) NOT NULL
);

```

Code for InterviewFeedback table creation

```

CREATE TABLE HR.InterviewFeedback(
InterviewFeedbackID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
InterviewID INT NOT NULL,
InterviewerID INT NOT NULL,
Result VARCHAR(100) NOT NULL,
Notes VARCHAR(100) NULL
FOREIGN KEY(InterviewID) REFERENCES HR.Interviews(InterviewID),
FOREIGN KEY(InterviewerID) REFERENCES HR.Interviewers(InterviewerID)
);

```

Code for Evaluation table creation

```

CREATE TABLE HR.Evaluation(
EvaluationID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
InterviewerID INT NOT NULL,
ApplicationID INT NOT NULL,
Result VARCHAR(50) NOT NULL,
EvaluationNotes VARCHAR(50) NULL
FOREIGN KEY(InterviewerID) REFERENCES HR.Interviewers(InterviewerID),
FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID)
);

```

Code for Reviews table creation

```

CREATE TABLE HR.Reviews(
ReviewsID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
CandidateReview VARCHAR(200) NULL,

```

```

InterviewerReview VARCHAR(200) NULL,
InterviewID INT NOT NULL,
InterviewerID INT NOT NULL,
CandidateID INT NOT NULL,
FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID),
FOREIGN KEY(InterviewID) REFERENCES HR.Interviews(InterviewID),
FOREIGN KEY(InterviewerID) REFERENCES HR.Interviewers(InterviewerID)
);

```

Code for Blacklist table creation

```

CREATE TABLE HR.Blacklist(
BlacklistID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
Reason VARCHAR(200) NULL,
isBlacklisted BIT NOT NULL,
CandidateID INT NOT NULL,
FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID),
);

```

3. Inserting data in Tables

Code for Candidates table insertion

```

INSERT INTO
HR.Candidates(FirstName,LastName,Email,Phone,ShortProfile,OfferAccepted)
VALUES
('Raj','Shah','rshah26@syr.edu','9876534564','Skilled in SQL','Yes'),
('Harsh','Mandviya','hmand6@syr.edu','6534564123','Skilled in Java',"),
('Jill','Goshrani','jigos2@syr.edu','3456446743','Proficient in DBMS, Python',"),
('Jaynam','Shah','jaysa8@syr.edu','3154214546','Skilled in C++','Yes'),
('Mohit','Thakkar','mohit36@syr.edu','3154564546','Skilled in MongoDB',"),
('Manav','Nisar','manan76@syr.edu','3123567548','Skilled in Data Science','Yes'),
('Kavish','Shah','kawah5@syr.edu','3874164564','Proficient in DBMS, Python',"),
('Bhavik','Shah','bhavh96@syr.edu','9126534564','Skilled in MERN','Yes'),
('Rithik','Gujar','rit456@syr.edu','9316534564','Proficient in Java, Python',"),
('Ayush','Shah','ayushh6@syr.edu','3187534564','Proficient in Node.js, Python','Yes');

```

Code for CandidateAddress table insertion

```

INSERT INTO HR.CandidateAddress(CandidateID,AddressLine1,City,ZipCode)
VALUES
(1,'12 Westcott Street','Syracuse','13210'),
(2,'712 Westcott Street','Syracuse','13210'),

```

```
(3,'822 Westcott Street','Syracuse','13210'),
(4,'456 Westcott Street','Syracuse','13210'),
(5,'321 Westcott Street','Syracuse','13210'),
(6,'897 Westcott Street','Syracuse','13210'),
(7,'567 Westcott Street','Syracuse','13210'),
(8,'1743 Westcott Street','Syracuse','13210'),
(9,'678 Westcott Street','Syracuse','13210'),
(10,'245 Westcott Street','Syracuse','13210');
```

Code for Documents table insertion

```
INSERT INTO HR.Documents(CandidateID, DocumentName, DocumentType,
DocumentURL)
```

VALUES

```
(1,'Resume','PDF','https://yourwebsite.com/pages/'),
(2,'CV','PDF','https://yourwebsite.com/pages/'),
(3,'CV','PDF','https://yourwebsite.com/pages/'),
(4,'Resume','Word','https://yourwebsite.com/pages/'),
(5,'CV','Word','https://yourwebsite.com/pages/'),
(6,'Resume','Word','https://yourwebsite.com/pages/'),
(7,'CV','Word','https://yourwebsite.com/pages/'),
(8,'Resume','Word','https://yourwebsite.com/pages/'),
(9,'Resume','PDF','https://yourwebsite.com/pages/'),
(10,'Resume','PDF','https://yourwebsite.com/pages/');
```

Code for JobPlatform table insertion

```
INSERT INTO HR.JobPlatform(JobPlatformName,JobPlatformDescription)
```

VALUES

```
('Job board','Company Job Board'),
('Social Media','LinkedIn'),
('Social Media','Indeed'),
('Social Media','Glassdoor'),
('College Connect','Handshake'),
('Referral','Company Employee'),
('Email','Email'),
('SMS','SMS'),
('Career Fair','College connect'),
('HR Recruiter','Inperson');
```

Code for Job table insertion

INSERT INTO

```
HR.Job(JobPlatformID,JobMedium,JobType,JobCategory,JobPosition,NoOfPositions,  
Salary)
```

VALUES

```
(1,'Online','Full Time','IT','Manager',10,100000),  
(2,'Online','Summer Internship','Sales','Associate',10,40000),  
(3,'Online','Full Time','Testing','Testing',10,120000),  
(4,'Online','Full Time','Software','Developer',10,190000),  
(5,'Online','Part Time','Software','Associate Developer',10,123000),  
(6,'Online','Part Time','Testing','Associate Tester',10,113000),  
(7,'Onsite','Full Time','Testing','Associate Manager',10,110000),  
(8,'Onsite','Summer Internship','Test','Associate',10,71000),  
(9,'Onsite','Full Time','IT','Developer',10,120000),  
(10,'Onsite','Summer Internship','IT','Developer',10,50000);
```

Code for Job Openings table insertion**INSERT INTO**

```
HR.JobOpenings(JobName,JobStartDate,DatePosted,NoOfVacancies,JobID)
```

VALUES

```
('SDE-1','12-12-2022','10-10-2022',7,4),  
(‘SDE-2’,'12-18-2023','10-10-2022',8,4),  
(‘Test Engineer’,'2-2-2023','10-10-2022',6,3),  
(‘Data Science Engineer’,'10-1-2023','10-10-2022',4,9),  
(‘Machine Learning Engineer’,'12-22-2023','10-10-2022',9,9),  
(‘Intern - SDE’,'12-25-2022','10-20-2022',4,10),  
(‘Test Intern’,'12-11-2022','10-10-2022',8,8),  
(‘Sales Intern’,'12-17-2022','10-10-2022',5,2),  
(‘Marketing Intern’,'12-18-2022','12-10-2022',6,2),  
(‘Associate Developer’,'12-27-2022','11-10-2022',7,5);
```

Code for Status table insertion**INSERT INTO HR.Status(Status)****VALUES**

```
(‘Declined’),  
(‘Accepted’),  
(‘Negotiating’);
```

Code for TestDetails table insertion**INSERT INTO HR.TestDetails**

```
(TestCategory,TestMaxScore,TestDuration)
```

VALUES

```
('online',100,'01:00:00'),  
('onsite',100,'01:00:00');
```

Code for InterviewType table insertion

```
INSERT INTO HR.InterviewType(InterviewCategory)
```

VALUES

```
('Onsite'),  
(('Online'));
```

Code for InterviewLocation table insertion

```
INSERT INTO
```

```
HR.InterviewLocation(InterviewLocationAddress,InterviewLocationState,InterviewL  
ocationCity,InterviewLocationZIPCode)
```

VALUES

```
('PDC1A Main Office' , 'NY' , 'NYC' , '10132'),  
('PDC1B Main Office' , 'NY' , 'NYC' , '10132'),  
('PDC1C Main Office' , 'NY' , 'NYC' , '10132'),  
('PDC1D Main Office' , 'NY' , 'NYC' , '10132'),  
('PDC1E Main Office' , 'NY' , 'NYC' , '10132'),  
('PDC1F Main Office' , 'NY' , 'NYC' , '10132'),  
('PDC1G Main Office' , 'NY' , 'NYC' , '10132'),  
('PDC1H Main Office' , 'NY' , 'NYC' , '10132'),  
('PDC1I Main Office' , 'NY' , 'NYC' , '10132'),  
('PDC1J Main Office' , 'NY' , 'NYC' , '10132'),  
('REMOTE' , " , " );
```

Code for Interviewers table insertion

```
INSERT INTO
```

```
HR.Interviewers(InterviewerID,FirstName,LastName,Title,Department)
```

VALUES

```
(1,'Robin','Sharma','Executives','IT'),  
(2,'Jon','Repp','Hiring manager','IT'),  
(3,'Marvin','Mandela','HR manager','HR'),  
(4,'Aaron','Wanda','Executives','IT'),  
(5,'Cristiano','Repp','Hiring manager','IT'),  
(6,'Yash','Joshi','Hiring manager','Marketing'),  
(7,'Jainam','Gosrani','Executives','Marketing'),  
(8,'Rohan','Singh','Hiring manager','IT'),  
(9,'Sonali','Kubde','Executives','IT'),
```

(10,'Saniya','Jain','Hiring manager','IT');

Code for Application table insertion

INSERT INTO HR.Application

(CandidateID,ApplicationTimeStamp,Education,WorkExperience,JobOpeningsID)

VALUES

(1,'12-07-2022','MS in Computer Science','Accenture',2),
(2,'12-07-2022','BSC in Computer Science','LTI',1),
(3,'12-07-2022','MS in CS','MindTree',3),
(4,'12-07-2022','MS in Computer Science','Amazon',4),
(5,'12-07-2022','MS in CS','Chewy',5),
(6,'12-07-2022','MS in CS','MindTree',4),
(7,'12-07-2022','MS in Computer Science','Amazon',3),
(8,'12-07-2022','BSC in Computer Science','LTI',4),
(8,'11-22-2022','BSC in Computer Science','LTI',5),
(9,'11-30-2022','MS in Computer Science','Accenture',6),
(10,'11-30-2022','MS in Computer Science','Accenture',6);

Code for ApplicationDocuments table insertion

INSERT INTO HR.ApplicationDocuments(ApplicationID,DocumentID)

VALUES

(1,1),
(2,2),
(3,3),
(4,4),
(5,5),
(6,6),
(7,7),
(8,8),
(9,8),
(10,9),
(11,10);

Code for Interviews table insertion

INSERT INTO

HR.Interviews(InterviewID,StartTime,EndTime,InterviewTypeID,InterviewLocationID,ApplicationID)

VALUES

(1,'09:30:00','10:30:00',1,2,1),
(2,'09:30:00','10:30:00',1,3,2),

```
(3,'09:30:00','10:30:00',1,4,3),
(4,'09:30:00','10:30:00',1,6,4),
(5,'09:30:00','10:30:00',1,6,5),
(6,'10:30:00','11:30:00',2,11,6),
(7,'10:30:00','11:30:00',2,11,7),
(8,'10:30:00','11:30:00',2,11,8),
(9,'10:30:00','11:30:00',2,11,9),
(10,'10:30:00','11:30:00',2,11,10),
(11,'10:30:00','11:30:00',1,8,1);
```

Code for Tests table insertion**INSERT INTO**

HR.Tests(ApplicationID,TestStartTime,TesEndTime,TestDetailsID,TestScore,TestGrade,InterviewID)

VALUES

```
(1,'09:30:00','10:30:00',1,81,'A',1),
(2,'09:30:00','10:30:00',1,71,'B+',2),
(3,'09:30:00','10:30:00',1,51,'C+',3),
(4,'09:30:00','10:30:00',1,100,'A+',4),
(5,'09:30:00','10:30:00',1,51,'C+',5),
(6,'10:30:00','11:30:00',2,71,'B+',6),
(7,'10:30:00','11:30:00',2,81,'A',7),
(8,'10:30:00','11:30:00',2,71,'B+',8),
(9,'10:30:00','11:30:00',2,51,'C+',9),
(10,'10:30:00','11:30:00',2,71,'B+',10),
(1,'10:30:00','11:30:00',1,91,'A+',1),
(1,'10:30:00','11:30:00',1,85,'A',1);
```

Code for InterviewFeedback table insertion**INSERT INTO** HR.InterviewFeedback(InterviewID,InterviewerID,Result,Notes)**VALUES**

```
(1,1,'Pass','Passed the interview'),
(1,5,'Pass','Passed the interview'),
(2,2,'Fail','Failed the interview'),
(3,3,'Fail','Failed the interview'),
(4,4,'Pass','Passed the interview'),
(5,5,'Pass','Passed the interview'),
(6,6,'Pass','Passed the interview'),
(7,7,'Fail','Failed the interview'),
```

```
(8,8,'Pass','Passed the interview'),
(9,9,'Fail','Failed the interview'),
(10,10,'Pass','Passed the interview'),
(11,2,'Pass','Passed the interview');
```

Code for Evaluation table insertion

```
INSERT INTO HR.Evaluation(ApplicationID, InterviewerID, Result,
EvaluationNotes)
```

VALUES

```
(1,1,'Accepted','Candidate fit for the role'),
(1,5,'Accepted','Candidate fit for the role'),
(1,2,'Accepted','Candidate fit for the role'),
(2,2,'Rejected','Candidate not fit for the role'),
(3,3,'Rejected','Candidate not fit for the role'),
(4,4,'Accepted','Candidate fit for the role'),
(5,5,'Rejected','Candidate not fit for the role'),
(6,6,'Accepted','Candidate fit for the role'),
(7,7,'Rejected','Candidate not fit for the role'),
(8,8,'Accepted','Candidate fit for the role'),
(9,9,'Accepted','Candidate fit for the role'),
(10,10,'Accepted','Candidate fit for the role');
```

Code for Reimbursement table insertion

```
INSERT INTO
```

```
HR.Reimbursement(ReimbursementRequestNumber,ReimbursementAmount,Reimbu
rsementDate,RequestStatus,RequestType,ApplicationID)
```

VALUES

```
(1001,10000,'12-06-2022','Open','HotelReservation',1),
(1002,18000,'11-16-2022','Open','Air Tickets',2),
(1003,20000,'12-05-2022','Open','Air Tickets',3),
(1004,10000,'11-14-2022','Open','Car Rental',4),
(1005,20000,'11-16-2022','Open','HotelReservation',5),
(1006,10000,'12-12-2022','Open','HotelReservation',6),
(1007,14000,'11-06-2022','Open','Car Rental',7),
(1008,10000,'11-9-2022','Open','Air Tickets',8),
(1009,16000,'12-06-2022','Open','Car Rental',9),
(1010,30000,'11-01-2022','Open','Air Tickets',10);
```

Code for OnBoarding table insertion

```
INSERT INTO
HR.OnBoarding(OnboardingStatus,ActualStartDate,JobID,CandidateID, StartDate)
VALUES
('Completed','2023-12-28',4,1,'2023-12-28'),
('OnHold','2021-12-12',9,4,'2023-12-12'),
('Completed','2021-12-25',10,6,'2023-12-25'),
('OnHold','2022-12-28',9,8,'2022-12-28');
```

Code for Complaint table insertion

```
INSERT INTO
HR.Complaint(ApplicationID,ComplaintDescription,ComplaintInterviewStatus,Comp
laintValid)
VALUES
(2,'Did not get an interview call','Waiting',"),
(3,'Did not get an interview call','Waiting',"),
(5,'Did not get an interview call','Re-Interview','1');
```

Code for ApplicationStatus table insertion

```
INSERT INTO HR.ApplicationStatus(StatusID,
ApplicationID,ApplicationStatusChangedDate)
VALUES
(2,1,'12-06-2022 09:00:10'),
(3,2,'12-06-2022 09:00:10'),
(3,3,'12-06-2022 09:00:10'),
(2,4,'12-06-2022 09:00:10'),
(3,5,'12-06-2022 09:00:10'),
(2,6,'12-06-2022 09:00:10'),
(1,7,'12-06-2022 09:00:10'),
(2,8,'12-06-2022 09:00:10'),
(3,9,'12-06-2022 09:00:10'),
(1,10,'12-06-2022 09:00:10');
```

Code for Reviews table insertion

```
INSERT into
HR.Reviews(InterviewerID,InterviewID,CandidateID,InterviewerReview,CandidateR
eview)
VALUES
(5,1,1,'Good Candidate','Smooth Process'),
```

```
(2,1,1,'Good Candidate','Smooth Process'),
(1,1,1,'Good Candidate','Smooth Process'),
(2,2,2,'Bad Candidate','Bad Process'),
(3,3,3,'Bad Candidate','Bad Process'),
(4,4,4,'Good Candidate','Smooth Process'),
(5,5,5,'Bad Candidate','Bad Process'),
(6,6,6,'Good Candidate','Smooth Process'),
(7,7,7,'Good Candidate','Smooth Process'),
(8,8,8,'Good Candidate','Process Good');
```

Code for BlackList table insertion

```
INSERT INTO HR.Blacklist(BlacklistID,CandidateID,Reason,isBlacklisted)
VALUES(1,10,'Not Joined the Company',1);
```

4. Retrieving data from Tables**Code for retrieving from Candidates table**

```
SELECT * FROM HR.Candidates;
```

Code for retrieving from CandidateAddress table

```
SELECT * FROM HR.CandidateAddress;
```

Code for retrieving from Documents table

```
SELECT * FROM HR.Documents;
```

Code for retrieving from Blacklist table

```
SELECT * FROM HR.Blacklist;
```

Code for retrieving from OnBoarding table

```
SELECT * FROM HR.OnBoarding;
```

Code for retrieving from JobPlatform table

```
SELECT * from HR.JobPlatform;
```

Code for retrieving from Complaint table

```
SELECT * FROM HR.Complaint;
```

Code for retrieving from Job table

```
SELECT * FROM HR.Job;
```

Code for retrieving from JobOpenings table

```
SELECT * FROM HR.JobOpenings;
```

Code for retrieving from Application table

```
SELECT * FROM HR.Application;
```

Code for retrieving from Interviews table

```
SELECT * FROM HR.Interviews;
```

Code for retrieving from Reimbursement table

```
SELECT * FROM HR.Reimbursement;
```

Code for retrieving from Evaluation table

```
SELECT * FROM HR.Evaluation;
```

Code for retrieving from Tests table

```
SELECT * FROM HR.Tests;
```

Code for retrieving from TestDetails table

```
SELECT * FROM HR.TestDetails;
```

Code for retrieving from Status table

```
SELECT * FROM HR.Status;
```

Code for retrieving from ApplicationStatus table

```
SELECT * FROM HR.ApplicationStatus;
```

Code for retrieving from Reviews table

```
SELECT * FROM HR.Reviews;
```

Code for retrieving from ApplicationDocuments table

```
SELECT * FROM HR.ApplicationDocuments;
```

Code for retrieving from InterviewType table

```
SELECT * FROM HR.InterviewType;
```

Code for retrieving from InterviewLocation table

```
SELECT * FROM HR.InterviewLocation;
```

Code for retrieving from Interviewers table

```
SELECT * FROM HR.Interviewers;
```

Code for retrieving from InterviewFeedBack table

```
SELECT * FROM HR.InterviewFeedBack;
```

TESTING

1. Views

View 1:

To gather all of the job details in one place, this view is constructed.

```
CREATE VIEW JobView
```

```
AS
```

```
SELECT j.JobID, j.JobCategory, j.JobMedium, jp.JobPlatformName, j.JobPosition
FROM HR.Job j
INNER JOIN HR.JobPlatform jp
ON j.JobPlatformID= jp.JobPlatformID;
SELECT * FROM JobView;
```

JobID	JobCategory	JobMedium	JobPlatformName	JobPosition
1 1	IT	Online	Job board	Manager
2 2	Sales	Online	Social Media	Associate
3 3	Testing	Online	Social Media	Testing
4 4	Software	Online	Social Media	Developer
5 5	Software	Online	College Connect	Associate Developer
6 6	Testing	Online	Referral	Associate Tester
7 7	Testing	Onsite	Email	Associate Manager
8 8	Test	Onsite	SMS	Associate
9 9	IT	Onsite	Career Fair	Developer
10 10	IT	Onsite	HR Recruiter	Developer

View 2:

It presents all of the application's information in one view, including the applicant name and currentStatus.

```
CREATE VIEW Application_Details
```

```
AS
```

```
SELECT a.ApplicationID, c.FirstName, c.LastName, s.Status
FROM HR.Application a
JOIN HR.Candidates c
ON a.CandidateID=c.CandidateID
JOIN HR.ApplicationStatus ac
ON a.ApplicationID=ac.ApplicationID
```

```
JOIN HR.Status s
ON ac.StatusID=s.StatusID;
SELECT * FROM Application_Details;
```

```

CREATE VIEW InterviewDetails
AS
SELECT a.ApplicationID, c.FirstName AS CandidateFirstName,c.LastName AS CandidateLastName,i.StartTime,i.EndTime,f.Result,s.FirstName,s.LastName,s.Title,
s.Department
FROM HR.Candidates c
JOIN HR.Application a
ON a.CandidateID=c.CandidateID
JOIN HR.Interviews i
ON a.ApplicationID=i.ApplicationID
JOIN HR.InterviewFeedback f
ON i.InterviewID=f.InterviewID
JOIN HR.Interviewers s
ON f.InterviewerID=s.InterviewerID;
SELECT * from InterviewDetails;

```

	ApplicationID	FirstName	LastName	Status
1	1	Raj	Shah	Accepted
2	2	Harsh	Mandviya	Negotiating
3	3	Jill	Goshrani	Negotiating
4	4	Jaynam	Shah	Accepted
5	5	Mohit	Thakkar	Negotiating
6	6	Manav	Nisar	Accepted
7	7	Kavish	Shah	Declined
8	8	Bhavik	Shah	Accepted
9	9	Bhavik	Shah	Negotiating
10	10	Rithik	Gujar	Declined

View 3:

This view gives the all the interview details in one view.

(candidate, application, interviewer , interview feedback, and interview details)

```
CREATE VIEW InterviewDetails
```

```
AS
```

```

SELECT a.ApplicationID, c.FirstName AS CandidateFirstName,c.LastName AS CandidateLastName,i.StartTime,i.EndTime,f.Result,s.FirstName,s.LastName,s.Title,
s.Department
FROM HR.Candidates c
JOIN HR.Application a
ON a.CandidateID=c.CandidateID
JOIN HR.Interviews i
ON a.ApplicationID=i.ApplicationID
JOIN HR.InterviewFeedback f
ON i.InterviewID=f.InterviewID
JOIN HR.Interviewers s
ON f.InterviewerID=s.InterviewerID;
SELECT * from InterviewDetails;

```

The screenshot shows the SSMS interface with a query window open. The query is attempting to create a view named 'InterviewDetails' but includes multiple statements, which is causing an error:

```

CREATE VIEW InterviewDetails
AS
SELECT a.ApplicationID, c.FirstName AS CandidateFirstName, c.LastName AS CandidateLastName, i.StartTime, i.EndTime, f.Result, s.FirstName, s.LastName
FROM HR.Candidates_C
JOIN HR.Application_A
ON a.CandidateID=a.CandidateID
JOIN HR.Interviews_I
ON f.InterviewerID=I.InterviewerID
column InterviewerID(PK, int, not null)
Incorrect syntax: 'CREATE VIEW' must be the only statement in the batch.

```

The results grid displays 12 rows of interview data:

	ApplicationID	CandidateFirstName	CandidateLastName	StartTime	EndTime	Result	FirstName	LastName	Title
1	1	Raj	Shah	09:30:00	10:30:00	Pass	Robin	Sharma	Ex
2	1	Raj	Shah	09:30:00	10:30:00	Pass	Cristiano	Repp	Hi
3	2	Harsh	Mandviya	09:30:00	10:30:00	Fail	Jon	Repp	Hi
4	3	Jill	Goshrani	09:30:00	10:30:00	Fail	Marvin	Mandela	HR
5	4	Jaynam	Shah	09:30:00	10:30:00	Pass	Aaron	Wanda	Ex
6	5	Mohit	Thakkar	09:30:00	10:30:00	Pass	Cristiano	Repp	Hi
7	6	Manav	Nisar	10:30:00	11:30:00	Pass	Yash	Joshi	Hi
8	7	Kavish	Shah	10:30:00	11:30:00	Fail	Jainam	Gosrani	Ex
9	8	Bhavik	Shah	10:30:00	11:30:00	Pass	Rohan	Singh	Hi
10	9	Bhavik	Shah	10:30:00	11:30:00	Fail	Sonali	Kubde	Ex
11	10	Rithik	Gujar	10:30:00	11:30:00	Pass	Saniya	Jain	H
12	1	Raj	Shah	10:30:00	11:30:00	Pass	Jon	Repp	Hi

View 4:

This view provides the candidate's personally identifiable information.

`CREATE VIEW CandidatePersonalInfo`

`AS`

```

SELECT c.FirstName,c.LastName, c.Email,c.Phone,a.AddressLine1,a.City,a.ZipCode
FROM HR.Candidates c
JOIN HR.CandidateAddress a
ON c.CandidateID=a.CandidateID;
SELECT * FROM CandidatePersonalInfo;

```

The screenshot shows the SSMS interface with a query window open. The query creates a view named 'CandidatePersonalInfo' and selects personal information from candidates and their addresses:

```

CREATE VIEW CandidatePersonalInfo
AS
SELECT c.FirstName, c.LastName, c.Email, c.Phone, a.AddressLine1, a.City, a.ZipCode
FROM HR.Candidates c
JOIN HR.CandidateAddress a
ON c.CandidateID=a.CandidateID;
SELECT * FROM CandidatePersonalInfo;

```

The results grid displays 18 rows of candidate personal information:

	FirstName	LastName	Email	Phone	AddressLine1	City	ZipCode
1	Raj	Shah	rshaw26@syr.edu	9876534564	12 Westcott Street	Syracuse	13210
2	Harsh	Mandviya	hmand@syrs.edu	6534564123	712 Westcott Street	Syracuse	13210
3	Jill	Goshrani	jigos2@syr.edu	3456446743	822 Westcott Street	Syracuse	13210
4	Jaynam	Shah	jaysa0@syr.edu	3154214546	456 Westcott Street	Syracuse	13210
5	Mohit	Thakkar	mohit36@syr.edu	3154564546	321 Westcott Street	Syracuse	13210
6	Manav	Nisar	manan76@syr.edu	3123567548	897 Westcott Street	Syracuse	13210
7	Kavish	Shah	kavah6@syr.edu	3874164564	567 Westcott Street	Syracuse	13210
8	Bhavik	Shah	bhavh96@syr.edu	9126534564	1743 Westcott Street	Syracuse	13210
9	Rithik	Gujar	rith45@syr.edu	9316534564	678 Westcott Street	Syracuse	13210
10	Ayush	Shah	ayushh0@syr.edu	3187534564	245 Westcott Street	Syracuse	13210

2. Stored Procedures

Stored Procedure 1

The procedure is designed to give candidates grades based on their obtained scores.

```
CREATE PROCEDURE ReassignGrades
AS
BEGIN
    UPDATE HRTests
    SET TestGrade = CASE
        WHEN TestScore >35 then 'PASS'
        ELSE 'FAIL'
    END
    END
    GO
SELECT * FROM HR.Tests;
```

```
CREATE PROCEDURE ReassignGrades
AS
BEGIN
    UPDATE HRTests
    SET TestGrade = CASE
        WHEN TestScore >35 then 'PASS'
        ELSE 'FAIL'
    END
    END
    GO
SELECT * FROM HR.Tests;
```

TestID	TestStartTime	TestEndTime	TestDetailsID	TestScore	TestGrade	ApplicationID	InterviewID
1	09:30:00	10:30:00	1	81	PASS	1	1
2	09:30:00	10:30:00	1	71	PASS	2	2
3	09:30:00	10:30:00	1	51	PASS	3	3
4	09:30:00	10:30:00	1	100	PASS	4	4
5	09:30:00	10:30:00	1	51	PASS	5	5
6	10:30:00	11:30:00	2	71	PASS	6	6
7	10:30:00	11:30:00	2	81	PASS	7	7
8	10:30:00	11:30:00	2	71	PASS	8	8
9	10:30:00	11:30:00	2	51	PASS	9	9
10	10:30:00	11:30:00	2	71	PASS	10	10
11	10:30:00	11:30:00	1	91	PASS	1	1
12	10:30:00	11:30:00	1	85	PASS	1	1

Stored Procedure 2

If the complaint is genuine, the candidate has a second meeting; otherwise, the status of the interview is changed to Rejected. This technique was developed to set the Candidate interview status to Reinterview or Rejected based on the ComplaintValid bit.

```
Create PROCEDURE StatusUpdate_Complaint AS
BEGIN
    UPDATE HRComplaint
    SET ComplaintInterviewStatus = CASE
        WHEN ComplaintValid = 0 then 'Rejected'
```

```

ELSE 'Reinterview'
END
END
GO
EXEC StatusUpdate_Complaint;
SELECT * from HR.Complaint;

```

```

CREATE PROCEDURE StatusUpdate_Complaint AS
BEGIN
    UPDATE HRComplaint
    SET ComplaintInterviewStatus = CASE
        WHEN ComplaintValid = 0 then 'Rejected'
        ELSE 'Reinterview'
    END
    GO
    EXEC StatusUpdate_Complaint;
SELECT * from HR.Complaint;

```

ComplaintID	ComplaintDescription	ComplaintValid	ComplaintInterviewStatus	ApplicationID
1	Did not get an interview -	0	Rejected	2
2	Did not get an interview -	0	Rejected	3
3	Did not get an interview -	1	Reinterview	5

Stored Procedure 3

This procedure is used to assign feedback to the applicant depending on the outcome of the interview; specifically, whenever the outcome contains the term "Passed," the feedback is set to Deserving Candidate, and otherwise to Not Deserving Candidate.

```

CREATE PROC InterviewerFeedbackOnInterview as
BEGIN
    UPDATE HR.InterviewFeedback
    SET Notes = CASE
        WHEN Notes LIKE '%Passed%' THEN 'Deserving Candidate'
        ELSE 'Not Deserving Candidate'
    END
    GO
    EXEC InterviewerFeedbackOnInterview;

    SELECT * from HR.InterviewFeedback;

```

```

CREATE PROC InterviewerFeedbackOnInterview
BEGIN
    UPDATE HR.InterviewerFeedback
    SET Notes = CASE
        WHEN Notes LIKE '%Passed%' THEN 'Deserving Candidate'
        ELSE 'Not Deserving Candidate'
    END
    GO
    EXEC InterviewerFeedbackOnInterview;
    SELECT * from HR.InterviewerFeedback;
END
GO

```

The screenshot shows the SSMS interface with the 'Results' tab selected, displaying the execution results of the stored procedure. The results table has columns: InterviewFeedbackID, InterviewID, InterviewerID, Result, and Notes. The data shows various rows with different results and notes.

Stored Procedure 4

This procedure is used to change candidate's offeraccepted column value.

CREATE PROCEDURE OfferDecision AS

BEGIN

UPDATE HR.Candidates

SET OfferAccepted = CASE

WHEN OfferAccepted ='Yes' then 'Accepted'

WHEN OfferAccepted ='No' then 'Declined'

ELSE NULL

END

END

GO

EXEC OfferDecision;

SELECT * FROM HR.Candidates;

```

CREATE PROCEDURE OfferDecision AS
BEGIN
    UPDATE HR.Candidates
    SET OfferAccepted = CASE
        WHEN OfferAccepted ='Yes' then 'Accepted'
        WHEN OfferAccepted ='No' then 'Declined'
        ELSE NULL
    END
    END
    GO
    EXEC OfferDecision;
    SELECT * FROM HR.Candidates;
END
GO

```

The screenshot shows the SSMS interface with the 'Results' tab selected, displaying the execution results of the stored procedure. The results table has columns: CandidateID, FirstName, LastName, Email, Phone, ShortProfile, and OfferAccepted. The data shows various rows with different names and offer accepted status.

3. User Defined Functions

Function 1

The job type variable is taken as an input by the onboarding function, which then provides details about the applicant who was recruited for that kind. The applicant's contact information, the job for which they were employed, and the start date are all returned by this function.

Create Function HR.OnboardingFunc(@Job_type varchar(200))

Returns Table

Return

(

SELECT o.StartDate, c.Firstname, c.Lastname, c.offeraccepted, j.JobPosition,

j.jobMedium

FROM HR.Onboarding o

JOIN HR.Job j

ON o.jobID=j.jobID

JOIN Hr.Candidates c

ON o.CandidateID=c.CandidateID

WHERE j.JobType=@Job_type

);

Select * from HR.OnboardingFunc('Full Time');

```

CREATE FUNCTION HR.OnboardingFunc(@Job_type varchar(200))
RETURNS TABLE
RETURN
(
    SELECT o.StartDate, c.Firstname, c.Lastname, c.offeraccepted, j.JobPosition, j.jobMedium
    FROM HR.Onboarding o
    JOIN HR.Job j
    ON o.jobID=j.jobID
    JOIN Hr.Candidates c
    ON o.CandidateID=c.CandidateID
    WHERE j.JobType=@Job_type
);

```

Start Date	First Name	Last Name	Offer Accepted	Job Position	Job Medium
2022-12-28 00:00:00.000	Raj	Shah	Accepted	Developer	Online
2021-12-12 00:00:00.000	Jaynam	Shah	Accepted	Developer	Onsite
2022-12-28 00:00:00.000	Bhavik	Shah	Accepted	Developer	Onsite

Function 2

This function accepts a job name as an input and returns a list of all applicable jobs along with details about each one, including the job's title, position, type, method of execution, and duties.

Create function HR.Jobfunc(@Job_name varchar(100))

Returns Table

Return

(

```
Select jo.JobName,j.JobPosition,j.JobCategory,j.JobType,j.Salary
from HR.Job j
JOIN HR.JobOpenings jo
ON j.JobID= jo.JobID
Where jo.JobName=@Job_name
);
Select * from HR.Jobfunc('SDE-1');
```

JobName	JobPosition	JobCategory	JobType	Salary
SDE-1	Developer	Software	Full Time	190000.00

Function 3

This function accepts a test id as an input and returns a list of all applicable test along with details about each one, including the TestsID, candidateID, Candidate Name, TestCategory, and TestGrade.

Create function HR.Test_Details(@test int)

Returns Table

Return

(

```
select att.TestsID,c.candidateID,concat(c.FirstName,' ',c.LastName) as
Candidate_Name,t.TestCategory,att.TestGrade
from HR.testdetails t
join HR.tests att
on t.TestDetailsID = att.TestDetailsID
JOIN HR.application a
```

```

on att.ApplicationID = a.ApplicationID
JOIN HR.candidates c
on c.CandidateID = a.CandidateID
where att.testsID = @test
);
GO
select * from Hr.Test_Details('1');

```

The screenshot shows the SSMS interface with the following details:

- Servers:** A tree view showing various database objects such as Triggers, Indexes, Statistics, and several HR-related tables and procedures.
- Query Editor:** The main window contains the T-SQL code for the stored procedure. The code includes a function definition for `HR.Test_Details` and a select statement that joins multiple tables (HR.Candidates, HR.Tests, HR.Application, etc.) to retrieve test details for a specific candidate.
- Results:** A grid table titled "Results" displays the output of the query. It has columns: TestsID, candidateID, Candidate_Name, TestCategory, and TestGrade. The single row returned is: 1, 1, Raj Shah, online, PASS.
- Status Bar:** At the bottom, it shows the message "The changes have been successfully published." and various status indicators like Line: 877, Col: 1, Spaces: 4, etc.

Function 4

This function accepts a interview id as an input and returns a list of all applicable Interview along with details about each one, including the InterviewID, Candidate Name, InterviewerName, and Result.

Create function `HR.Interview_Details(@interview int)`

Returns Table

Return

```

(
select t.InterviewID,concat(c.FirstName,' ',c.LastName) as CandidateName,
concat(I.FirstName,' ',I.LastName) as InterviewerName, att.Result
from HR.Interviews t
join HR.InterviewFeedback att
on t.InterviewID = att.InterviewID
join HR.Interviewers I
on I.InterviewerID=att.InterviewerID
JOIN HR.application a
on t.ApplicationID = a.ApplicationID

```

```

JOIN HR.candidates c
on c.CandidateID = a.CandidateID
where t.InterviewID = @interview
);
GO
SELECT * FROM Hr.Interview_Details('1');

```

```

CREATE FUNCTION Hr.Interview_Details(@interview INT)
RETURNS TABLE
AS
BEGIN
    RETURN (
        SELECT t.InterviewID, concat(c.FirstName, ' ', c.LastName) AS CandidateName, concat(I.FirstName, ' ', I.LastName) AS InterviewerName, att.Result
        FROM HR.Interviews t
        JOIN HR.InterviewFeedback att
        ON t.InterviewID = att.InterviewID
        JOIN HR.Interviewers I
        ON I.InterviewerID = att.InterviewerID
        JOIN HR.Application a
        ON t.ApplicationID = a.ApplicationID
        JOIN HR.Candidates c
        ON c.CandidateID = a.CandidateID
        WHERE t.InterviewID = @interview
    );
END;
GO
SELECT * FROM Hr.Interview_Details('1');

+-----+-----+-----+-----+
| InterviewID | CandidateName | InterviewerName | Result |
+-----+-----+-----+-----+
| 1 | Raj Shah | Robin Sharma | Pass |
| 2 | Raj Shah | Cristiano Repp | Pass |
+-----+-----+-----+-----+

```

4. Triggers

Trigger 1

This trigger is activated whenever a row is inserted into the Reviews table.

```
Create Trigger TriggerReview
```

```
ON HR.Reviews
```

```
AFTER INSERT
```

```
AS
```

```
Declare @InterviewerID INT, @InterviewID INT
```

```
BEGIN
```

```
UPDATE HR.Reviews
```

```
SET InterviewerReview = CASE
```

```
WHEN InterviewerReview LIKE '%Good%' THEN 'POSITIVE'
```

```
ELSE 'NEGATIVE'
```

```
END
```

```
END
```

INSERT INTO

```
HR.Reviews(CandidateReview,InterviewerReview,CandidateID,InterviewID,InterviewerID) VALUES('Good','Need Work',2,2,8)
```

```
SELECT * from HR.Reviews;
```

The screenshot shows the SSMS interface with a query window titled 'HRProject2.sql - localhost...ah (sa)'. The code in the window includes a trigger definition and an insert statement:

```

Create Trigger SetReviews
ON HR.Reviews
AFTER INSERT
AS
Declare @InterviewerID int, @InterviewerID int
BEGIN
    UPDATE HR.Reviews
    SET InterviewerReview = CASE
        WHEN InterviewerReview LIKE '%Good%' THEN 'Positive'
        ELSE 'Negative'
    END
    INSERT INTO HR.Reviews(CandidateReview,InterviewerReview,CandidateID,InterviewID,InterviewerID) VALUES('Good','Need Work',2,2,8)
    SELECT * from HR.Reviews;
END

```

The 'Results' tab shows the output of the 'SELECT * from HR.Reviews;' query, which displays 11 rows of data:

	ReviewsID	CandidateReview	InterviewerReview	InterviewID	InterviewerID	CandidateID
1	1	Smooth Process	Positive	1	5	1
2	2	Smooth Process	Positive	1	2	1
3	3	Smooth Process	Positive	1	1	1
4	4	Bad Process	NEGATIVE	2	2	2
5	5	Bad Process	NEGATIVE	3	3	3
6	6	Smooth Process	Positive	4	4	4
7	7	Bad Process	NEGATIVE	5	5	5
8	8	Smooth Process	Positive	6	6	6
9	9	Smooth Process	Positive	7	7	7
10	18	Process Good	Positive	8	8	8
11	11	Good	NEGATIVE	2	8	2

Trigger 2

This trigger is activated whenever a row is inserted into the Tests table. The value is set.

```

Create Trigger TriggerTestResult
ON HR.Tests
AFTER Insert
AS
DECLARE @TestsId INT
BEGIN
    SELECT @TestsId=TestsID FROM HR.Tests WHERE TestGrade = NULL
    UPDATE HR.Tests Set TestGrade = CASE
        WHEN TestScore>=60 THEN 'PASS'
        ELSE 'FAIL'
    END
    WHERE TestsID = @TestsId
END

```

```
INSERT INTO HR.Tests VALUES(12,'09:30:00','10:30:00',1,85,NULL,1,1);
```

```
SELECT * from hr.Tests;
```

The screenshot shows the SSMS interface. In the Object Explorer, under the 'HRProject2.sql - local...ah (sa)' connection, the 'HR.Tests' table is selected. The 'Script' tab of the query editor displays the following T-SQL code:

```

980 Create Trigger TriggerTestResult
981 ON HR.Tests
982 AFTER Insert
983 AS
984 DECLARE @TestsID INT
985 BEGIN
986     SELECT @TestsId=TestsID FROM HR.Tests WHERE TestGrade = NULL
987     UPDATE HR.Tests Set TestGrade = CASE
988         WHEN TestScore>=60 THEN 'PASS'
989         ELSE 'FAIL'
990     END
991     WHERE TestsID = @TestsId
992 END
993
994 INSERT INTO HR.Tests VALUES(12,'09:30:00','10:30:00',1,85,NULL,1,1);
995
996 SELECT * from hr.Tests;
997

```

The 'Results' tab shows the output of the 'SELECT * from hr.Tests;' query, displaying 12 rows of test data:

	TestsID	TestStartTime	TesEndTime	TestDetailsID	TestScore	TestGrade	ApplicationID	InterviewID
1	1	09:30:00	10:30:00	1	81	PASS	1	1
2	2	09:30:00	10:30:00	1	71	PASS	2	2
3	3	09:30:00	10:30:00	1	51	FAIL	3	3
4	4	09:30:00	10:30:00	1	100	PASS	4	4
5	5	09:30:00	10:30:00	1	51	FAIL	5	5
6	6	10:30:00	11:30:00	2	71	PASS	6	6
7	7	10:30:00	11:30:00	2	81	PASS	7	7
8	8	10:30:00	11:30:00	2	71	PASS	8	8
9	9	10:30:00	11:30:00	2	51	FAIL	9	9
10	10	10:30:00	11:30:00	2	71	PASS	10	10
11	11	10:30:00	11:30:00	1	91	PASS	1	1
12	12	10:30:00	11:30:00	1	85	PASS	1	1

Trigger 3

When the candidate's status is changed to Declined, this query adds them to blacklist table.

```

CREATE TRIGGER TriggerBlacklist ON HR.ApplicationStatus
AFTER UPDATE
AS
DECLARE @CandidateID INT, @Reason VARCHAR(50)
IF EXISTS
(SELECT c.CandidateID
FROM HR.Candidates c,HR.Application a,HR.ApplicationStatus ac
WHERE c.CandidateId=a.CandidateId AND a.ApplicationId=ac.ApplicationId AND
ac.StatusID=1 AND c.CandidateID NOT IN (SELECT candidateid FROM
HR.Blacklist))
BEGIN
SELECT @CandidateID = c.CandidateID, @Reason = 'Not joined the company'
FROM
HR.Candidates c, HR.Application a,HR.ApplicationStatus ac
WHERE c.CandidateId=a.CandidateId AND a.ApplicationId=ac.ApplicationId AND
ac.StatusID=1 and c.CandidateID

```

```

NOT IN (SELECT CandidateID FROM HR.Blacklist)
INSERT INTO HR.Blacklist VALUES (4,@Reason, @CandidateID,1)
END
UPDATE HR.ApplicationStatus SET StatusId = 1 WHERE ApplicationID = 9;
SELECT * FROM HR.Blacklist;

```

```

CREATE TRIGGER TriggerBlacklist ON HR.ApplicationStatus
AFTER UPDATE
AS
DECLARE @CandidateID INT, @Reason VARCHAR(50)
IF EXISTS
(
    SELECT c.CandidateID
    FROM HR.Candidates c,HR.Application a,HR.ApplicationStatus ac
    WHERE c.CandidateId=a.CandidateID AND a.ApplicationId=ac.ApplicationId AND
    ac.StatusId=1 AND c.CandidateID NOT IN (SELECT CandidateID FROM HR.Blacklist)
)
BEGIN
SELECT @CandidateID = c.CandidateID, @Reason = 'Not joined the company' FROM
HR.Candidates c, HR.Application a,HR.ApplicationStatus ac,
WHERE c.CandidateId=a.CandidateID AND a.ApplicationId=ac.ApplicationId AND ac.StatusID=1 and c.CandidateID
NOT IN (SELECT CandidateID FROM HR.Blacklist)
INSERT INTO HR.Blacklist VALUES (@CandidateID,1)
END
UPDATE HR.ApplicationStatus SET StatusId = 1 WHERE ApplicationID = 9;
SELECT * FROM HR.Blacklist;

```

BlacklistID	Reason	CandidateID	isBlacklisted
1	Not Joined the Company	10	1
2	Not joined the company	9	1

Trigger 4

When the application status is changed from accepted to declined, this query executes the triggers. There are now 1 more vacancies available.

Create Trigger TriggerUpdateVacancies

ON HR.ApplicationStatus

After Update AS

DECLARE @jobOpeningID INT

IF EXISTS (SELECT * FROM HR.Application a,HR.ApplicationStatus sc

WHERE a.ApplicationID=sc.ApplicationID AND a.CandidateID NOT IN (SELECT CandidateID from HR.Blacklist) AND sc.StatusId=1)

BEGIN

SELECT @jobOpeningID = jo.jobOpeningsId

FROM HR.JobOpenings jo JOIN HR.Job j

ON jo.JobId=j.JobId

JOIN HR.Application a

ON a.JobOpeningsId = jo.JobOpeningsId

JOIN HR.Candidates c

```

ON c.CandidateId= a.CandidateId
JOIN HR.ApplicationStatus sc
ON sc.ApplicationId=a.ApplicationId
WHERE c.CandidateId NOT IN (SELECT candidateId FROM HR.Blacklist) AND
StatusId=1
UPDATE HR.JobOpenings SET NoOfVacancies = NoOfVacancies+1 WHERE
JobOpeningsID = @jobOpeningID
END
UPDATE HR.ApplicationStatus
SET StatusID = 1 WHERE ApplicationID = 7;
SELECT * FROM HR.ApplicationStatus;
SELECT * from HR.JobOpenings;

```

The screenshot shows the SSMS interface with the following details:

- Connections:** HRPProject2.sql - localhost...ah (sa) 9+ •
- Servers:** localhost, <default> (sa)
- Databases:** HR, HR_RajShah
- Script:**

```

CREATE Trigger TriggerUpdateVacancies
ON HR.ApplicationStatus
After Update AS
DECLARE @jobOpeningID INT
IF EXISTS (SELECT * FROM HR.Application a,HR.ApplicationStatus sc
           WHERE a.ApplicationID=sc.ApplicationID AND a.CandidateID NOT IN (SELECT CandidateID from HR.Blacklist) AND sc.StatusId=1)
BEGIN
    SELECT @jobOpeningID = jo.JobOpeningID
    FROM HR.JobOpenings jo JOIN HR.Job j
    ON jo.JobId=j.JobId
    JOIN HR.Application a
    ON a.JobOpeningID = jo.JobOpeningID
    JOIN HR.Candidates c
    ON c.CandidateId = a.CandidateId
    JOIN HR.ApplicationStatus sc
    ON sc.ApplicationId=a.ApplicationId
    WHERE c.CandidateId NOT IN (SELECT CandidateID FROM HR.Blacklist) AND StatusId=1
    UPDATE HR.JobOpenings SET NoOfVacancies = NoOfVacancies+1 WHERE JobOpeningID = @jobOpeningID
END
UPDATE HR.ApplicationStatus
SET StatusID = 1 WHERE ApplicationID = 7;
SELECT * FROM HR.ApplicationStatus;

```
- Results:** A table with columns ApplicationStatusID, ApplicationStatusChangedDate, StatusID, and ApplicationID. The data is as follows:

ApplicationStatusID	ApplicationStatusChangedDate	StatusID	ApplicationID
1	2022-12-06 09:00:10.000	2	1
2	2022-12-06 09:00:10.000	3	2
3	2022-12-06 09:00:10.000	3	3
4	2022-12-06 09:00:10.000	2	4
5	2022-12-06 09:00:10.000	3	5
6	2022-12-06 09:00:10.000	2	6
7	2022-12-06 09:00:10.000	1	7
8	2022-12-06 09:00:10.000	2	8
9	2022-12-06 09:00:10.000	1	9
10	2022-12-06 09:00:10.000	1	10

The screenshot shows the SSMS interface with the following details:

- Connections:** HRPProject2.sql - localhost...ah (sa) 9+ •
- Servers:** localhost, <default> (sa)
- Databases:** HR, HR_RajShah
- Script:**

```

CREATE Trigger TriggerUpdateVacancies
ON HR.ApplicationStatus
After Update AS
DECLARE @jobOpeningID INT
IF EXISTS (SELECT * FROM HR.Application a,HR.ApplicationStatus sc
           WHERE a.ApplicationID=sc.ApplicationID AND a.CandidateID NOT IN (SELECT CandidateID from HR.Blacklist) AND sc.StatusId=1)
BEGIN
    SELECT @jobOpeningID = jo.JobOpeningID
    FROM HR.JobOpenings jo JOIN HR.Job j
    ON jo.JobId=j.JobId
    JOIN HR.Application a
    ON a.JobOpeningID = jo.JobOpeningID
    JOIN HR.Candidates c
    ON c.CandidateId = a.CandidateId
    JOIN HR.ApplicationStatus sc
    ON sc.ApplicationId=a.ApplicationId
    WHERE c.CandidateId NOT IN (SELECT CandidateID FROM HR.Blacklist) AND StatusId=1
    UPDATE HR.JobOpenings SET NoOfVacancies = NoOfVacancies+1 WHERE JobOpeningID = @jobOpeningID
END
UPDATE HR.ApplicationStatus
SET StatusID = 1 WHERE ApplicationID = 7;
SELECT * FROM HR.ApplicationStatus;

```
- Results:** A table with columns ApplicationStatusID, ApplicationStatusChangedDate, StatusID, and ApplicationID. The data is as follows:

ApplicationStatusID	ApplicationStatusChangedDate	StatusID	ApplicationID
1	2022-12-06 09:00:10.000	2	1
2	2022-12-06 09:00:10.000	3	2
3	2022-12-06 09:00:10.000	3	3
4	2022-12-06 09:00:10.000	2	4
5	2022-12-06 09:00:10.000	3	5
6	2022-12-06 09:00:10.000	2	6
7	2022-12-06 09:00:10.000	1	7
8	2022-12-06 09:00:10.000	2	8
9	2022-12-06 09:00:10.000	1	9
10	2022-12-06 09:00:10.000	1	10

5. Transactions

Transaction 1

This transaction removes one job vacancy for which the candidate accepted the offer.

CREATE VIEW Update_Vacancy AS

```
SELECT c.CandidateId,c.FirstName,c.LastName,c.OfferAccepted,j.NoOfVacancies
FROM HR.Candidates c JOIN HR.Application a
ON c.CandidateID = a.CandidateID
JOIN HR.JobOpenings j
ON a.JobOpeningsID=j.JobOpeningsID
WHERE c.OfferAccepted='Accepted';
```

```
select * from Update_Vacancy;
```

```
BEGIN TRAN
```

```
DECLARE @vacancy int;
```

```
SELECT @vacancy = NoOfVacancies
```

```
FROM Update_Vacancy;
```

```
UPDATE Update_Vacancy
```

```
SET NoOfVacancies = @vacancy - 1
```

```
COMMIT TRAN
```

```
1083 CREATE VIEW Update_Vacancy AS
1084 SELECT c.CandidateId,c.FirstName,c.LastName,c.OfferAccepted,j.NoOfVacancies
1085 FROM HR.Candidates c JOIN HR.Application a
1086 ON c.CandidateID = a.CandidateID
1087 JOIN HR.JobOpenings j
1088 ON a.JobOpeningsID=j.JobOpeningsID
1089 WHERE c.OfferAccepted='Accepted';
1090
1091 select * from Update_Vacancy;
1092
1093 BEGIN TRAN
1094 DECLARE @vacancy int;
1095 SELECT @vacancy = NoOfVacancies
1096 FROM Update_Vacancy;
1097 UPDATE UpdateVacancy
1098 SET NoOfVacancies = @vacancy - 1
1099
1100 COMMIT TRAN
1101
1102
```

CandidateId	FirstName	LastName	OfferAccepted	NoOfVacancies
1	Raj	Shah	Accepted	3
2	Jaynam	Shah	Accepted	3
3	Manav	Nisar	Accepted	3
4	Bhavik	Shah	Accepted	3
5	Bhavik	Shah	Accepted	3
6	Ayush	Shah	Accepted	3

The screenshot shows the SSMS interface with a query window titled 'HRProject2.sql - localhost...ah (sa) 0+'. The code in the window is:

```

1083
1084 CREATE VIEW Update_Vacancy_AS
1085     SELECT c.CandidateID,c.FirstName,c.LastName,c.OfferAccepted,j.NoOfVacancies
1086     FROM HR.Candidates c JOIN HR.Application a
1087     ON c.CandidateID = a.CandidateID
1088     JOIN HR.JobOpenings_j
1089     ON a.JobOpeningsID=j.JobOpeningsID
1090     WHERE c.OfferAccepted='Accepted';
1091
1092 select * from Update_Vacancy;
1093
1094 BEGIN TRAN
1095 DECLARE @vacancy int;
1096 SELECT @vacancy = NoOfVacancies
1097 FROM UpdateVacancy;
1098 UPDATE UpdateVacancy
1099 SET NoOfVacancies = @vacancy - 1
1100 COMMIT TRAN
1101
1102

```

The results grid shows the following data:

	CandidateID	FirstName	LastName	OfferAccepted	NoOfVacancies
1	1	Raj	Shah	Accepted	2
2	4	Jaynam	Shah	Accepted	2
3	6	Manav	Nisar	Accepted	2
4	8	Bhavik	Shah	Accepted	2
5	8	Bhavik	Shah	Accepted	2
6	10	Ayush	Shah	Accepted	2

Transaction 2

This Transaction modifies the table's reimbursement status based on the amount requested and maximum amount. The reimbursement status is changed to "Rejected" if the amount desired is greater than the maximum amount authorized, otherwise it is changed to "Processed."

```

BEGIN TRAN
DECLARE @reimbursestatus varchar(20);
SELECT @reimbursestatus = RequestStatus
FROM HR.Reimbursement
UPDATE HR.Reimbursement
SET requestStatus = 'Processed'
WHERE ReimbursementAmount <= ReimbursementMaxAmount;
UPDATE HR.Reimbursement
SET RequestStatus = 'Declined'
WHERE ReimbursementAmount > ReimbursementMaxAmount;
COMMIT TRAN

SELECT * from HR.Reimbursement;

```

The screenshot shows the SSMS interface with a query window titled 'HRProject2.sql - localhost...ah (sa)'. The code in the window is:

```

1821
1822
1823
1824 BEGIN TRAN
1825 DECLARE @reimbursestatus varchar(20);
1826 SELECT @reimbursestatus = RequestStatus
1827 FROM HR.Reimbursement
1828 UPDATE HR.Reimbursement
1829 SET requeststatus = 'Processed'
1830 WHERE ReimbursementAmount > ReimbursementMaxAmount;
1831 UPDATE HR.Reimbursement
1832 SET RequestStatus = 'Declined'
1833 WHERE ReimbursementAmount > ReimbursementMaxAmount;
1834 COMMIT TRAN
1835
1836 SELECT * from HR.Reimbursement;
1837
1838
1839
1840

```

The results grid below shows 10 rows of data:

idNumber	ReimbursementAmount	ReimbursementDate	ReimbursementMaxAmount	ApplicationID	RequestType	RequestStatus
10000.00	2022-12-06	30000.00	1	HotelReservation	Processed	
18000.00	2022-11-16	30000.00	2	Air Tickets	Processed	
20000.00	2022-12-05	30000.00	3	Air Tickets	Processed	
10000.00	2022-11-14	30000.00	4	Car Rental	Processed	
20000.00	2022-11-16	30000.00	5	HotelReservation	Processed	
10000.00	2022-12-12	30000.00	6	HotelReservation	Processed	
14000.00	2022-11-06	30000.00	7	Car Rental	Processed	
10000.00	2022-11-09	30000.00	8	Air Tickets	Processed	
16000.00	2022-12-06	30000.00	9	Car Rental	Processed	
30000.00	2022-11-01	30000.00	10	Air Tickets	Processed	

Transaction 3

The Job table shows how the salary has been modified since the transaction was completed for Developer and Testing Job Positions.

BEGIN TRAN

```

DECLARE @jobsalary int;
SELECT @jobsalary = Salary
FROM HR.Job
UPDATE HR.Job
SET Salary = @jobsalary + 40000
WHERE JobPosition = 'Developer'
UPDATE HR.Job
SET Salary = @jobsalary + 23000
WHERE JobPosition = 'Testing'
COMMIT TRAN
SELECT * FROM HR.JOB;

```

```

1049
1050
1051
1052
1053 SELECT * FROM HR.JOB;
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068

```

JobID	JobPlatformID	NoOfVacancies	JobCategory	JobPosition	JobType	JobMedium	Salary	
1	1	10	IT	Manager	Full Time	Online	100000.00	
2	2	2	Sales	Associate	Summer Internship	Online	40000.00	
3	3	3	10	Testing	Testing	Full Time	Online	67000.00
4	4	4	10	Software	Developer	Full Time	Online	50000.00
5	5	5	10	Software	Associate Developer	Part Time	Online	123000.00
6	6	6	10	Testing	Associate Tester	Part Time	Online	113000.00
7	7	7	10	Testing	Associate Manager	Full Time	Onsite	110000.00
8	8	8	10	Test	Associate	Summer Internship	Onsite	71000.00
9	9	9	10	IT	Developer	Full Time	Onsite	50000.00
10	10	10	IT	Developer	Summer Internship	Onsite	50000.00	

```

1036
1037
1038 BEGIN TRAN
1039 DECLARE @jobsalary int;
1040 SELECT @jobsalary = Salary
1041 FROM HR.Job
1042 UPDATE HR.Job
1043 SET Salary = @jobsalary + 40000
1044 WHERE JobPosition = 'Developer'
1045 UPDATE HR.Job
1046 SET Salary = @jobsalary + 23000
1047 WHERE JobPosition = 'Testing'
1048 COMMIT TRAN
1049
1050 SELECT * FROM HR.JOB;
1051
1052
1053
1054
1055

```

JobID	JobPlatformID	NoOfVacancies	JobCategory	JobPosition	JobType	JobMedium	Salary	
1	1	10	IT	Manager	Full Time	Online	100000.00	
2	2	2	Sales	Associate	Summer Internship	Online	40000.00	
3	3	3	10	Testing	Testing	Full Time	Online	73000.00
4	4	4	10	Software	Developer	Full Time	Online	90000.00
5	5	5	10	Software	Associate Developer	Part Time	Online	123000.00
6	6	6	10	Testing	Associate Tester	Part Time	Online	113000.00
7	7	7	10	Testing	Associate Manager	Full Time	Onsite	110000.00
8	8	8	10	Test	Associate	Summer Internship	Onsite	71000.00
9	9	9	10	IT	Developer	Full Time	Onsite	90000.00
10	10	10	IT	Developer	Summer Internship	Onsite	90000.00	

Transaction 4

The Tests table shows how the TestScore has been increased by 10 marks since the transaction was completed.

Condition is applied for test score less than 60 increase it by 10 marks.

BEGIN TRAN

UPDATE HR.Tests

SET TestScore = TestScore + 10

WHERE TestScore <= 60

COMMIT TRAN

`SELECT * FROM HR.Tests;`

TestsID	TestStartTime	TestEndTime	TestDetailsID	TestScore	TestGrade	ApplicationID	InterviewID
1	09:30:00	10:30:00	1	81	PASS	1	1
2	09:30:00	10:30:00	1	71	PASS	2	2
3	09:30:00	10:30:00	1	51	FAIL	3	3
4	09:30:00	10:30:00	1	100	PASS	4	4
5	09:30:00	10:30:00	1	51	FAIL	5	5
6	10:30:00	11:30:00	2	71	PASS	6	6
7	10:30:00	11:30:00	2	81	PASS	7	7
8	10:30:00	11:30:00	2	71	PASS	8	8
9	10:30:00	11:30:00	2	51	FAIL	9	9
10	10:30:00	11:30:00	2	71	PASS	10	10
11	10:30:00	11:30:00	1	91	PASS	1	1
12	10:30:00	11:30:00	1	85	PASS	1	1

TestsID	TestStartTime	TestEndTime	TestDetailsID	TestScore	TestGrade	ApplicationID	InterviewID
1	09:30:00	10:30:00	1	81	PASS	1	1
2	09:30:00	10:30:00	1	71	PASS	2	2
3	09:30:00	10:30:00	1	61	FAIL	3	3
4	09:30:00	10:30:00	1	100	PASS	4	4
5	09:30:00	10:30:00	1	61	FAIL	5	5
6	10:30:00	11:30:00	2	71	PASS	6	6
7	10:30:00	11:30:00	2	81	PASS	7	7
8	10:30:00	11:30:00	2	71	PASS	8	8
9	10:30:00	11:30:00	2	61	FAIL	9	9
10	10:30:00	11:30:00	2	71	PASS	10	10
11	10:30:00	11:30:00	1	91	PASS	1	1
12	10:30:00	11:30:00	1	85	PASS	1	1

6. Test Scripts

TestScript 1

This script creates a role with the INSERT, UPDATE, and DELETE privileges for the Complaint table. The purpose of this function is to manage information relating to complaints, such as deciding if the candidate's complaint is legitimate or not and updating the table appropriately. Since changing the Complaints table may require

accessing other tables in the database, the Complaints Department role has also been granted Select permission for the whole HR_RajShah database. The person in charge of the complaints department can be given this job, which would provide him access to and control over all data pertaining to complaints.

```
CREATE ROLE ComplaintDepartment
```

```
GO
```

```
GRANT INSERT,UPDATE,DELETE on HR_RajShah.dbo.Complaint to
```

```
ComplaintDepartment
```

```
GO
```

```
GRANT SELECT on DATABASE:: HR_RajShah to ComplaintDepartment;
```

```
762 CREATE ROLE ComplaintDepartment;
763 GRANT INSERT,UPDATE,DELETE on HR_RajShah.dbo.Complaint to ComplaintDepartment;
765 GRANT SELECT on DATABASE:: HR_RajShah to ComplaintDepartment;
```

Messages

7:55:04 PM Started executing query at Line 761
Commands completed successfully.
Total execution time: 00:00:00.030

TestScript 2

The role Interviewing Department is created by this script, and it has access to all tables connected to interviews, including interview, interviewer, feedback, and assessment. The user who will oversee the interviewing process can be given this position, which will provide him access to and control over all interview-related data.

```
CREATE ROLE InterviewingDepartment
```

```
GO
```

```
GRANT SELECT ON HR_RajShah.dbo.Interviews to InterviewingDepartment
```

```
GO
```

```
GRANT SELECT ON HR_RajShah.dbo.Interviewers to InterviewingDepartment
```

```
GO
```

```
GRANT SELECT ON HR_RajShah.dbo.InterviewFeedback to
```

```
InterviewingDepartment
```

GO

GRANT SELECT ON HR_RajShah.dbo.Evaluation TO InterviewingDepartment

GO

GRANT SELECT, UPDATE ON HR_RajShah.dbo.JobOpenings TO

InterviewingDepartment;

```

CREATE ROLE InterviewingDepartment
GRANT SELECT ON HR_RajShah.dbo.Interviews TO InterviewingDepartment
GRANT SELECT ON HR_RajShah.dbo.Interviewers TO InterviewingDepartment
GRANT SELECT ON HR_RajShah.dbo.InterviewFeedback TO InterviewingDepartment
GRANT SELECT ON HR_RajShah.dbo.Evaluation TO InterviewingDepartment
GRANT SELECT, UPDATE ON HR_RajShah.dbo.JobOpenings TO InterviewingDepartment
    
```

Messages

- 8:02:20 PM Started executing query at Line 770
- 8:02:20 PM Commands completed successfully.
- 8:02:20 PM Started executing query at Line 772
- 8:02:20 PM Commands completed successfully.
- 8:02:20 PM Started executing query at Line 774
- 8:02:20 PM Commands completed successfully.
- 8:02:20 PM Started executing query at Line 776
- 8:02:20 PM Commands completed successfully.
- 8:02:20 PM Started executing query at Line 778
- 8:02:20 PM Commands completed successfully.
- 8:02:20 PM Started executing query at Line 780
- 8:02:20 PM Commands completed successfully.

Total execution time: 00:00:00.029

TestScript 3

Create role CandidateEntry and login user Raj with login details and password.

CREATE ROLE CandidateEntry;

GRANT UPDATE ON Candidates TO CandidateEntry

GO

CREATE LOGIN RajShah WITH PASSWORD = 'rshah@123',

DEFAULT_DATABASE = HR_RajShah;

CREATE USER Raj FOR LOGIN RajShah;

ALTER ROLE CandidateEntry ADD MEMBER Raj;

```

CREATE ROLE CandidateEntry;
GRANT UPDATE ON Candidates TO CandidateEntry
GO

CREATE LOGIN RajShah WITH PASSWORD = 'rshaw@123',
DEFAULT_DATABASE = HR_RajShah;
CREATE USER Raj FOR LOGIN RajShah;
ALTER ROLE CandidateEntry ADD MEMBER Raj;
GO

```

Messages

8:57:15 PM Started executing query at Line 798
Commands completed successfully.
8:57:15 PM Started executing query at Line 791
Commands completed successfully.
Total execution time: 00:00:00.036

TestScript 4

Create role InsertNewJob and assign it to JobOpenings.

`CREATE ROLE InsertNewJob;`

`GRANT INSERT, UPDATE ON JobOpenings TO InsertNewJob`

`GO`

```

CREATE ROLE ComplaintDepartment;
GRANT INSERT,UPDATE,DELETE on HR_RajShah.dbo.Complaint to ComplaintDepartment;
GRANT SELECT ON DATABASE:: HR_RajShah TO ComplaintDepartment;

```

Messages

7:55:04 PM Started executing query at Line 761
Commands completed successfully.
Total execution time: 00:00:00.030

7. Business reports

Business reports 1

Count the number of evaluation result a interviewer gives

`CREATE PROCEDURE InterviewerEvaluationResult @InterviewerName`

`VARCHAR(30)`

```
AS
```

```
BEGIN
```

```
WITH InterviewerEvaluation AS (
```

```
SELECT i.InterviewerID as InterviewerID,concat(i.FirstName, ' ', i.LastName) as
```

```
InterviewerName, e.result
```

```
FROM HR.Interviewers i
```

```
JOIN HR.Evaluation e
```

```
ON i.InterviewerID=e.InterviewerID
```

```
WHERE concat(i.FirstName, ' ', i.LastName) = @InterviewerName)
```

```
SELECT InterviewerID,InterviewerName,COUNT(Result) AS [Evaluation Result]
```

```
FROM InterviewerEvaluation
```

```
GROUP BY InterviewerID,InterviewerName
```

```
ORDER BY InterviewerID;
```

```
END
```

```
EXEC InterviewerEvaluationResult @InterviewerName='Jon Repp';
```

```

CREATE PROCEDURE InterviewerEvaluationResult @InterviewerName VARCHAR(30)
AS
BEGIN
    WITH InterviewerEvaluation AS (
        SELECT i.InterviewerID as InterviewerID,concat(i.FirstName, ' ', i.LastName) as InterviewerName, e.result
        FROM HR.Interviewers i
        JOIN HR.Evaluation e
        ON i.InterviewerID=e.InterviewerID
        WHERE concat(i.FirstName, ' ', i.LastName) = @InterviewerName)
    SELECT InterviewerID,InterviewerName,COUNT(Result) AS [Evaluation Result]
    FROM InterviewerEvaluation
    GROUP BY InterviewerID,InterviewerName
    ORDER BY InterviewerID;
END
EXEC InterviewerEvaluationResult @InterviewerName='Jon Repp';

```

InterviewerID	InterviewerName	Evaluation Result	
1	2	Jon Repp	2

Business Reports 2

The below parameterized procedure calculates Average Salary from custom JobType and JobPosition.

```
SELECT * from HR.Job;
```

The screenshot shows the SSMS interface with the following details:

- Connections:** HRProject2.sql - localhost\sa (sa)
- Servers:** SERVERS > HR.interviewType > HR.Job
- Tables:** Columns, Keys, Constraints, Triggers, Indexes, Statistics, JobOpenings
- Query:** SELECT * from HR.Job;
- Results:** A grid of 10 rows of data from the HR.Job table.

JobID	JobPlatformID	NoOfVacancies	JobCategory	JobPosition	JobType	JobMedium	Salary
1	1	10	IT	Manager	Full Time	Online	100000.00
2	2	10	Sales	Associate	Summer Internship	Online	40000.00
3	3	10	Testing	Testing	Full Time	Online	67000.00
4	4	10	Software	Developer	Full Time	Online	70000.00
5	5	10	Software	Associate Developer	Part Time	Online	123000.00
6	6	10	Testing	Associate Tester	Part Time	Online	113000.00
7	7	10	Testing	Associate Manager	Full Time	Onsite	110000.00
8	8	10	Test	Associate	Summer Internship	Onsite	71000.00
9	9	10	IT	Developer	Full Time	Onsite	50000.00
10	10	10	IT	Developer	Summer Internship	Onsite	50000.00

```

CREATE PROCEDURE AveragePayForJobposition @JobType
VARCHAR(30),@JobPosition VARCHAR(30)
AS
BEGIN
SELECT JobType,JobPosition,AVG(Salary) AS [Job Salary]
FROM HR.Job
WHERE JobType = @JobType AND JobPosition=@JobPosition
GROUP BY JobType,JobPosition
END
EXEC AveragePayForJobposition @JobType='Full Time', @JobPosition='Developer';

```

```

CREATE PROCEDURE AveragePayForJobposition @JobType VARCHAR(30), @JobPosition VARCHAR(30)
AS
BEGIN
    SELECT JobType, JobPosition, AVG(Salary) AS [Job_Salary]
    FROM HR.Job
    WHERE JobType = @JobType AND JobPosition=@JobPosition
    GROUP BY JobType, JobPosition
END
EXEC AveragePayForJobposition @JobType='Full Time', @JobPosition='Developer';

```

JobType	JobPosition	Job_Salary
Full Time	Developer	60000.00

Business reports 3

This procedure displays reimbursement amount distribution based on date. Parameter named ReimbursementDate is passed in parameterized procedure because we want records of distribution after 1 november 2022 only.

```

CREATE PROCEDURE ReimbursementDistributionResult @ReimbursementDate
date
AS
BEGIN
SELECT ReimbursementDate, ReimbursementID, ReimbursementAmount,
PERCENT_RANK() OVER (PARTITION BY ReimbursementDate
ORDER BY ReimbursementAmount) AS PctRank,
CUME_DIST() OVER (PARTITION BY ReimbursementDate
ORDER BY ReimbursementAmount) AS CumeDist,
PERCENTILE_CONT(.5) WITHIN GROUP (ORDER BY ReimbursementAmount)
OVER (PARTITION BY ReimbursementDate) AS PercentileCont,
PERCENTILE_DISC(.5) WITHIN GROUP (ORDER BY ReimbursementAmount)
OVER (PARTITION BY ReimbursementDate) AS PercentileDisc
FROM HR.Reimbursement
WHERE ReimbursementDate > @ReimbursementDate;
END
EXEC ReimbursementDistributionResult @ReimbursementDate ='11-01-2022';

```

```

CREATE PROCEDURE ReimbursementDistributionResult @ReimbursementDate DATE
AS
BEGIN
SELECT ReimbursementDate, ReimbursementID, ReimbursementAmount,
PERCENT_RANK() OVER (PARTITION BY ReimbursementDate
ORDER BY ReimbursementAmount) AS PctRank,
CUME_DIST() OVER (PARTITION BY ReimbursementDate
ORDER BY ReimbursementAmount) AS CumeDist,
PERCENTILE_CONT(.5) WITHIN GROUP (ORDER BY ReimbursementAmount)
OVER (PARTITION BY ReimbursementDate) AS PercentileCont,
PERCENTILE_DISC(.5) WITHIN GROUP (ORDER BY ReimbursementAmount)
OVER (PARTITION BY ReimbursementDate) AS PercentileDisc
FROM HR.Reimbursement
WHERE ReimbursementDate > @ReimbursementDate;
END
EXEC ReimbursementDistributionResult @ReimbursementDate = '11-01-2022';

```

	ReimbursementDate	ReimbursementID	ReimbursementAmount	PctRank	CumeDist	PercentileCont	PercentileDisc
1	2022-11-06	7	14000.00	0	1	14000	14000.00
2	2022-11-09	8	10000.00	0	1	10000	10000.00
3	2022-11-14	4	10000.00	0	1	10000	10000.00
4	2022-11-16	2	18000.00	0	0.5	19000	18000.00
5	2022-11-16	5	20000.00	1	1	19000	18000.00
6	2022-12-05	3	20000.00	0	1	20000	20000.00
7	2022-12-06	1	10000.00	0	0.5	13000	10000.00
8	2022-12-06	9	15000.00	1	1	13000	10000.00
9	2022-12-12	6	10000.00	0	1	10000	10000.00

Business reports 4

Count the number of interview a interviewer takes

```

CREATE PROCEDURE InterviewerTakesInvterview @InterviewerID INT
AS
BEGIN
WITH TakesInvterview AS
(SELECT i.InterviewerID as InterviewerID, concat(i.FirstName, ' ', i.LastName) as
InterviewerName,ij.InterviewID
FROM HR.Interviewers i
JOIN HR.InterviewFeedback ij
ON i.InterviewerID=ij.InterviewerID)
SELECT InterviewerID,InterviewerName, COUNT(InterviewID) AS
InterviewConducted
FROM TakesInvterview
WHERE InterviewerID=@InterviewerID
GROUP BY InterviewerID,InterviewerName
ORDER BY InterviewerID;
END
EXEC InterviewerTakesInvterview @InterviewerID=5;

```

```
1114
1115  /* Count the number of interview a interviewer takes */
1116  CREATE PROCEDURE InterviewerTakesInterview @InterviewerID INT
1117  AS
1118  BEGIN
1119      WITH TakesInterview AS
1120      (SELECT i.InterviewerID AS InterviewerID, concat(i.FirstName, ' ', i.LastName) AS InterviewerName, ii.InterviewID
1121      FROM HR.Interviewers i
1122      JOIN HR.InterviewFeedback ii
1123      ON i.InterviewerID=ii.InterviewerID
1124  )
1125      SELECT InterviewerID, InterviewerName, COUNT(InterviewID) AS InterviewConducted
1126      FROM TakesInterview
1127      WHERE InterviewerID=@InterviewerID
1128      GROUP BY InterviewerID, InterviewerName
1129      ORDER BY InterviewerID;
1130  END
1131  GO
1132  EXEC InterviewerTakesInterview @InterviewerID=5;
1133
1134
1135
1136
```

InterviewerID	InterviewerName	InterviewConducted	
1	5	Cristiano Repp	2

Ln 1135, Col 1 Spaces: 4 UTF-8 LF SQL 1 rows MSSQL 00:00:00 localhost : HR_RajShah

CONCLUSION

In this project, I have successfully built a HR database for the recruitment procedure of the company. This project is under the direction of the company's HR Recruitment division. All of the tables necessary for the recruitment process have been created. Along with an ER Diagram that matches our database design, we have established tables for jobs, applications, interview, etc. Additionally, there exists adequate amount of information in each table. In addition, we've developed and tested the database with a number of queries such as Views, User Defined Functions, Stored Procedures, Scripts, Triggers, and Transactions. All the SQL codes as well as the associated screenshots are attached in the appendix.

APPENDIX

Creation of Database HR_RajShah

The screenshot shows the SSMS interface with the following details:

- Connections:** Shows the server 'localhost <default> (sa)' and the database 'HR_RajShah'.
- Object Explorer:** Shows the schema of the database, including Tables, Views, Synonyms, External Resources, Storage, Service Broker, Security, and other system objects.
- SQL Editor:** Contains the T-SQL script for creating the database:


```

USE master
GO
IF DB_ID('HR_RajShah') IS NOT NULL
    DROP DATABASE HR_RajShah
GO
CREATE DATABASE HR_RajShah
GO
    
```
- Messages:** Displays log messages indicating the successful execution of the commands.
- Bottom Status Bar:** Shows the command count (1), spaces used (4), and execution time (00:00:00.047).

Creation of HR Schema:

The screenshot shows the SSMS interface with the following details:

- Connections:** Shows the server 'localhost <default> (sa)' and the database 'HR_RajShah'.
- Object Explorer:** Shows the schema of the database, including MyCollege, ProductOrders, RajShah, School, Security, and Server Objects.
- SQL Editor:** Contains the T-SQL script for creating the schema:


```

CREATE SCHEMA HR;
    
```
- Messages:** Displays log messages indicating the successful execution of the command.
- Bottom Status Bar:** Shows the command count (1), spaces used (4), and execution time (00:00:00.030).

Screenshots for creation of all the tables

The screenshot shows the SSMS interface with the following details:

- Connections:** Shows the server 'localhost <default> (sa)' and the database 'HR_RajShah'.
- Object Explorer:** Shows the schema of the database, including master, model, msdb, tempdb, AP, Examples, HR_RajShah, MyCollege, ProductOrders, School, and Security.
- SQL Editor:** Contains the T-SQL script for creating the table:


```

CREATE TABLE HR.Candidates(
    CandidateID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
    FirstName VARCHAR(30) NOT NULL,
    LastName VARCHAR(30) NOT NULL,
    Email VARCHAR(50) NOT NULL,
    Phone VARCHAR(10) NOT NULL,
    ShortProfile VARCHAR(100) NULL DEFAULT 'Not Given',
    OfferAccepted VARCHAR(10) NULL);
    
```
- Messages:** Displays log messages indicating the successful execution of the command.
- Bottom Status Bar:** Shows the command count (12), spaces used (4), and execution time (00:00:00.021).

The screenshot shows the SSMS interface with the following details:

- Connections:** HRRProject2.sql - localhost, <default> (sa)
- Databases:** System Databases (selected)
- Table:** HR.CandidateAddress
- Code:**

```

24
25   CREATE TABLE HR.CandidateAddress(
26     CandidateAddressID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
27     CandidateID INT NOT NULL,
28     AddressLine1 VARCHAR(30) NOT NULL,
29     AddressLine2 VARCHAR(30) NULL,
30     Street VARCHAR(50) NULL,
31     City VARCHAR(10) NOT NULL,
32     ZipCode VARCHAR(5) NOT NULL,
33     FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID)
34   );
35

```
- Messages:**
 - Started executing query at Line 11
 - Commands completed successfully.
 - Total execution time: 00:00:00.021
- Status Bar:** Ln 33, Col 40 | Spaces: 4 | UTF-8 | LF | 0 rows | MSSQL | 00:00:00 | localhost : master

The screenshot shows the SSMS interface with the following details:

- Connections:** HRRProject2.sql - localhost, <default> (sa)
- Databases:** System Databases (selected)
- Table:** HR.JobPlatform
- Code:**

```

36
37
38   CREATE TABLE HR.JobPlatform(
39     JobPlatformID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
40     JobPlatformName VARCHAR(100) NOT NULL,
41     JobPlatformDescription VARCHAR(300) NOT NULL
42   );
43
44
45
46

```
- Messages:**
 - Started executing query at Line 38
 - Commands completed successfully.
 - Total execution time: 00:00:00.015
- Status Bar:** Ln 42, Col 3 | Spaces: 4 | UTF-8 | LF | 0 rows | MSSQL | 00:00:00 | localhost : master

```

57 );
58 ;
59 CREATE TABLE HR.JobOpenings(
60 JobOpeningsID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
61 NoOfVacancies INT NOT NULL,
62 DatePosted DATE NOT NULL,
63 JobStartDate DATE NOT NULL,
64 JobName VARCHAR(100) NOT NULL,
65 JobID INT NOT NULL,
66 FOREIGN KEY(JobID) REFERENCES HR.Job(JobID)
67 );
68

```

Messages

8:19:09 PM Started executing query at Line 59
Commands completed successfully.
Total execution time: 00:00:00.016

Ln 58, Col 1 Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master

```

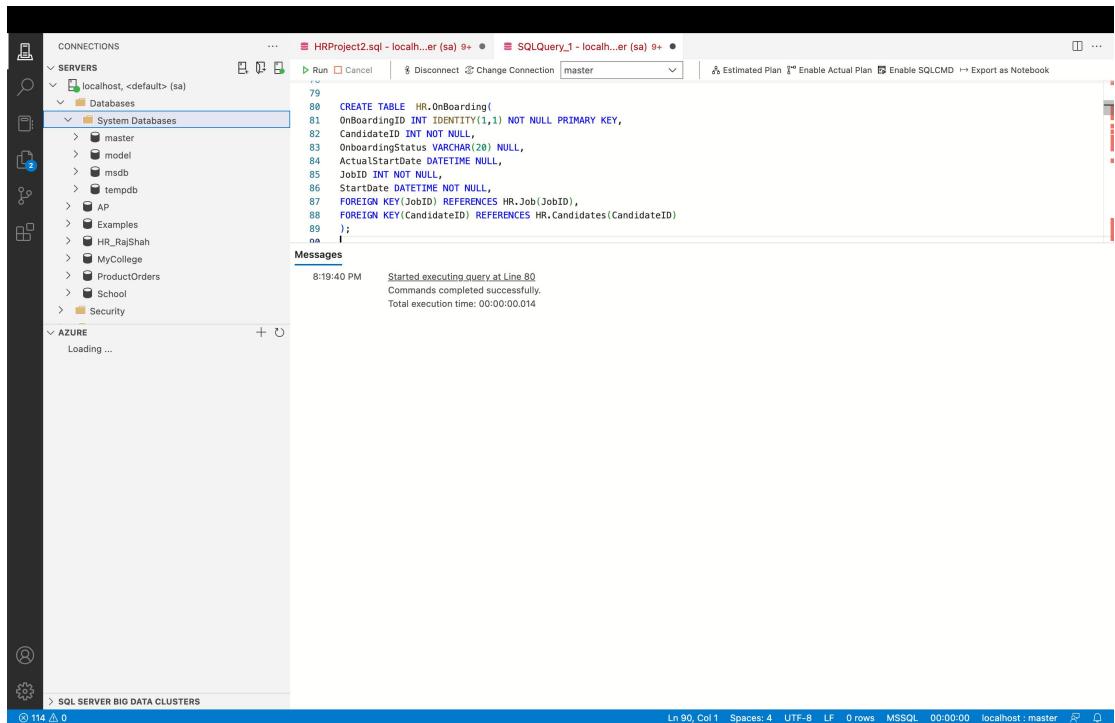
68 );
69 CREATE TABLE HR.Documents(
70 DocumentID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
71 CandidateID INT NOT NULL,
72 DocumentName VARCHAR(30) NOT NULL,
73 DocumentType VARCHAR(30) NOT NULL,
74 DocumentUrl VARCHAR(50) NOT NULL,
75 FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID)
76 );
77 ;
78 ;
79

```

Messages

8:19:26 PM Started executing query at Line 69
Commands completed successfully.
Total execution time: 00:00:00.012

Ln 77, Col 1 Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master



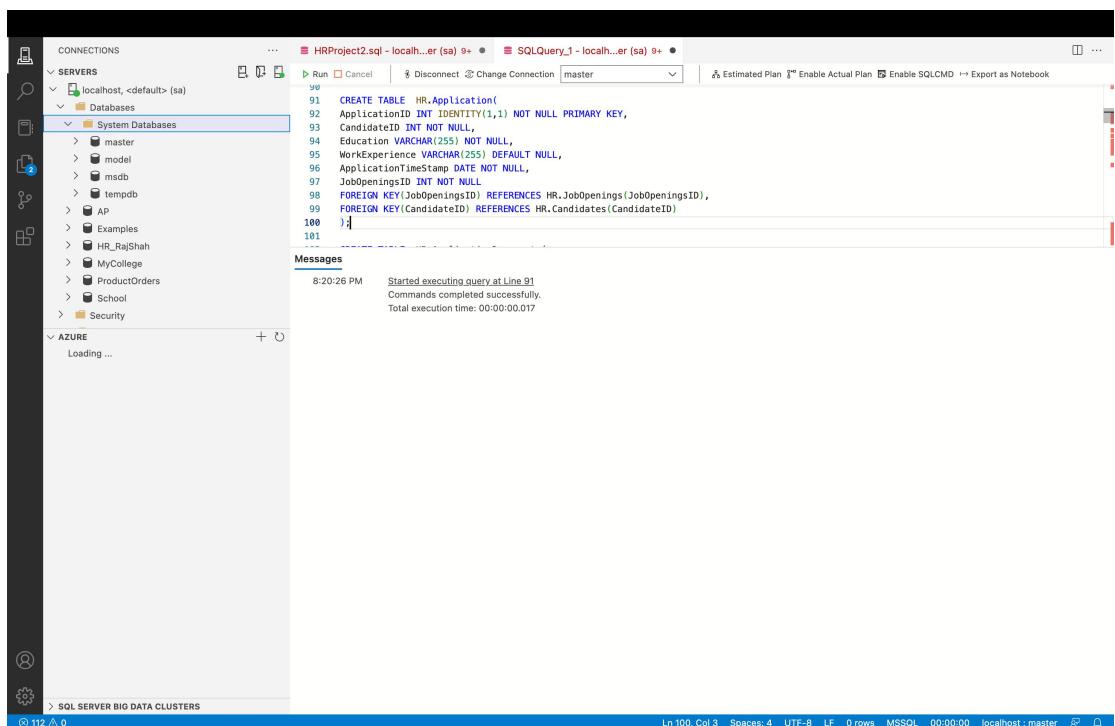
```

79
80 CREATE TABLE HR.OnBoarding(
81     OnBoardingID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
82     CandidateID INT NOT NULL,
83     OnboardingStatus VARCHAR(20) NULL,
84     ActualStartDate DATETIME NULL,
85     JobID INT NOT NULL,
86     StartDate DATETIME NOT NULL,
87     FOREIGN KEY(JobID) REFERENCES HR.Job(JobID),
88     FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID)
89 );
90

```

Messages

8:19:40 PM Started executing query at Line 80
Commands completed successfully.
Total execution time: 00:00:00.014



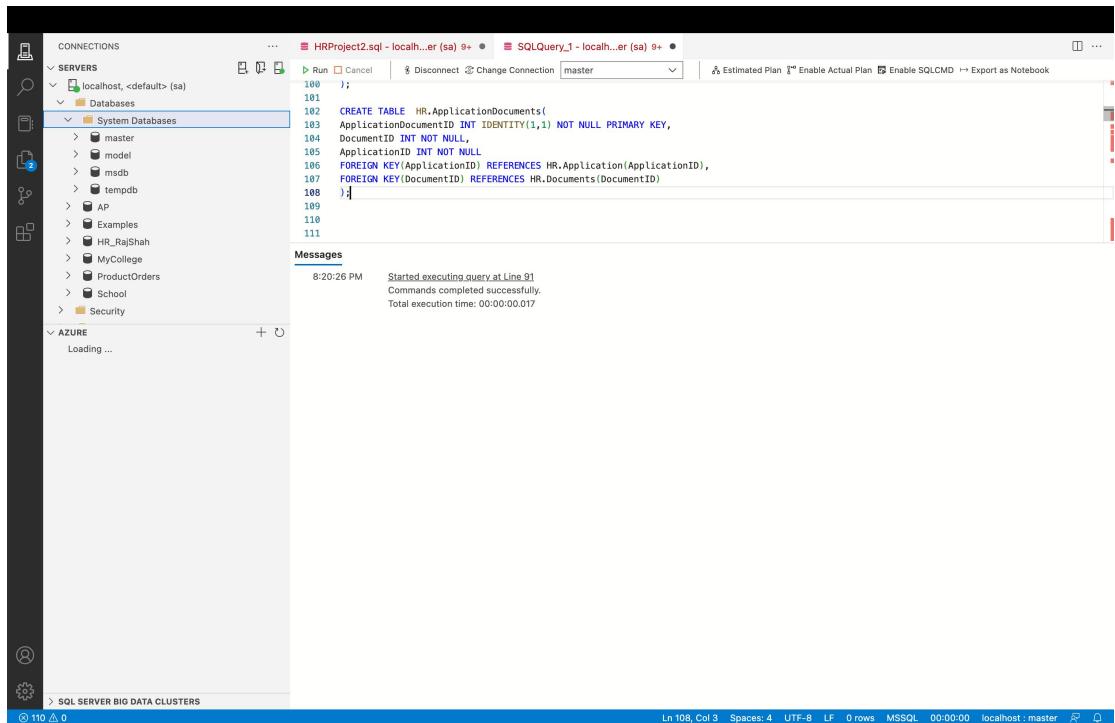
```

91 CREATE TABLE HR.Application(
92     ApplicationID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
93     CandidateID INT NOT NULL,
94     EducationLevel VARCHAR(255) NOT NULL,
95     WorkExperience VARCHAR(255) DEFAULT NULL,
96     ApplicationTimeStamp DATE NOT NULL,
97     JobOpeningID INT NOT NULL,
98     FOREIGN KEY(JobOpeningID) REFERENCES HR.JobOpenings(JobOpeningID),
99     FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID)
100 );
101

```

Messages

8:20:26 PM Started executing query at Line 91
Commands completed successfully.
Total execution time: 00:00:00.017



```

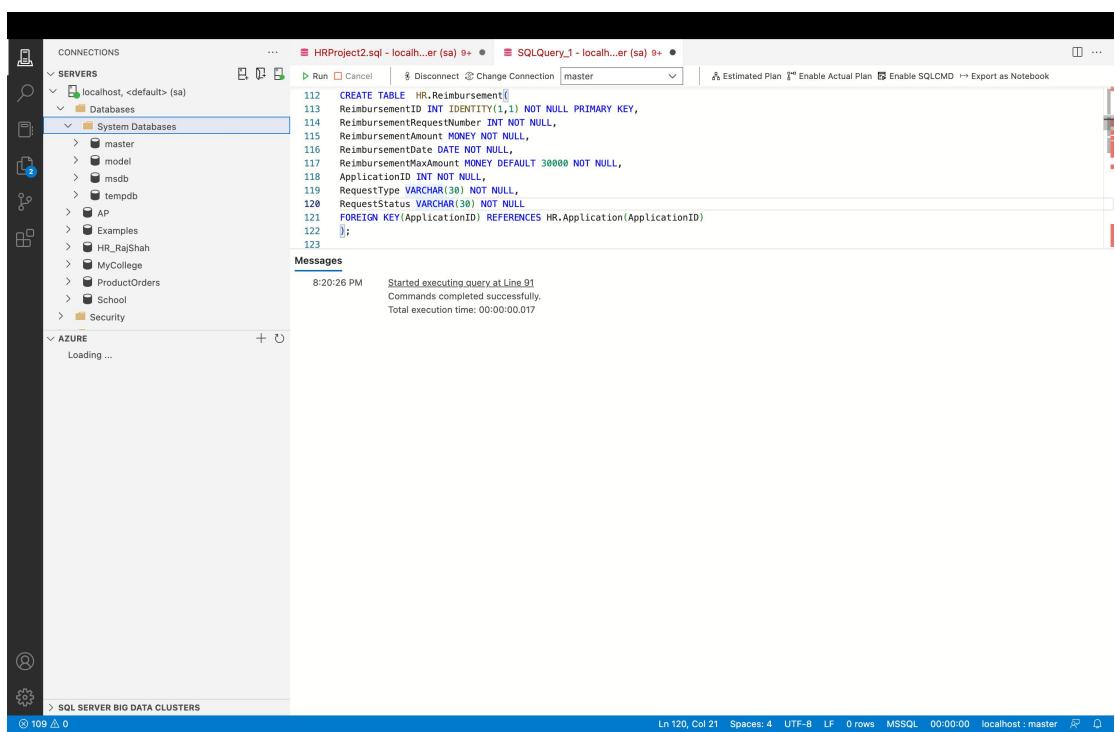
100  };
101
102 CREATE TABLE HR.ApplicationDocuments(
103     ApplicationDocumentID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
104     DocumentID INT NOT NULL,
105     ApplicationID INT NOT NULL,
106     FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID),
107     FOREIGN KEY(DocumentID) REFERENCES HR.Documents(DocumentID)
108 );
109
110
111

```

Messages

8:20:26 PM Started executing query at Line 91
Commands completed successfully.
Total execution time: 00:00:00.017

Ln 110, Col 3 Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master



```

112 CREATE TABLE HR.Reimbursement(
113     ReimbursementID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
114     ReimbursementRequestNumber INT NOT NULL,
115     ReimbursementAmount MONEY NOT NULL,
116     ReimbursementDate DATE NOT NULL,
117     ReimbursementMaxAmount MONEY DEFAULT 30000 NOT NULL,
118     ApplicationID INT NOT NULL,
119     RequestType VARCHAR(30) NOT NULL,
120     RequestStatus VARCHAR(30) NOT NULL,
121     FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID)
122 );
123

```

Messages

8:20:26 PM Started executing query at Line 91
Commands completed successfully.
Total execution time: 00:00:00.017

Ln 120, Col 21 Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master

```

122    );
123
124    CREATE TABLE HR.Complaint(
125        ComplaintID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
126        ComplaintDescription VARCHAR(255) NOT NULL,
127        ComplaintIsValid BIT NOT NULL,
128        ComplaintInterviewStatus VARCHAR(50) NOT NULL,
129        ApplicationID INT NOT NULL
130        FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID)
131    );
132
133

```

Messages

8:20:56 PM Started executing query at Line 124
Commands completed successfully.
Total execution time: 00:00:00.025

Ln 133, Col 3 Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master

```

132
133
134    CREATE TABLE HR.ApplicationStatus(
135        ApplicationStatusID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
136        ApplicationStatusChangedDate DATETIME NOT NULL,
137        StatusID INT NOT NULL,
138        ApplicationID INT NOT NULL
139        FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID)
140    );
141
142
143

```

Messages

8:21:09 PM Started executing query at Line 134
Commands completed successfully.
Total execution time: 00:00:00.014

Ln 143, Col 3 Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master

The screenshot shows the SSMS interface with the following details:

- Connections:** HRRProject2.sql - localhost\sa (sa) 9+ - SQLQuery_1 - localhost\sa (sa) 9+
- Servers:** localhost, <default> (sa)
- Databases:** System Databases
- System Databases:** master, model, msdb, tempdb, AP, Examples, HR_RajShah, MyCollege, ProductOrders, School, Security
- Code Editor:** CREATE TABLE HR.Status
StatusID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
Status VARCHAR(100) NOT NULL;
141
142
143
144
145
146
147
148
149
150
151
152
- Messages:** Started executing query at Line 144. Commands completed successfully. Total execution time: 00:00:00.017
- Status Bar:** Ln 154, Col 1 (264 selected) Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master

The screenshot shows the SSMS interface with the following details:

- Connections:** HRRProject2.sql - localhost\sa (sa) 9+ - SQLQuery_1 - localhost\sa (sa) 9+
- Servers:** localhost, <default> (sa)
- Databases:** System Databases
- System Databases:** master, model, msdb, tempdb, AP, Examples, HR_RajShah, MyCollege, ProductOrders, School, Security
- Code Editor:** CREATE TABLE HR.InterviewLocation
InterviewLocationID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
InterviewlocationAddress VARCHAR(20) NULL,
InterviewlocationCity VARCHAR(20) NULL,
InterviewlocationState VARCHAR(20) NULL,
InterviewlocationZIPCode VARCHAR(20) NULL;
151
152
153
154
155
156
157
158
159
160
161
162
- Messages:** Started executing query at Line 154. Commands completed successfully. Total execution time: 00:00:00.020
- Status Bar:** Ln 161, Col 1 Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master

```

162
163 CREATE TABLE HR.InterviewType(
164 InterviewTypeID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
165 InterviewCategory VARCHAR(28) NOT NULL
166 );
167
168
169
170
171
172

```

Messages

8:21:56 PM Started executing query at Line 163
Commands completed successfully.
Total execution time: 00:00:00.011

```

173 CREATE TABLE HR.Interviews(
174 InterviewID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
175 StartTime TIME NOT NULL,
176 EndTime TIME NOT NULL,
177 ApplicationID INT NOT NULL,
178 InterviewTypeID INT NOT NULL,
179 InterviewLocationID INT NOT NULL,
180 FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID),
181 FOREIGN KEY(InterviewLocationID) REFERENCES HR.InterviewLocation(InterviewLocationID),
182 FOREIGN KEY(InterviewTypeID) REFERENCES HR.InterviewType(InterviewTypeID)
183 );
184

```

Messages

8:22:25 PM Started executing query at Line 173
Commands completed successfully.
Total execution time: 00:00:00.019

The screenshot shows the SSMS interface with the following details:

- Connections:** HRPProject2.sql - localhost, <default> (sa)
- Databases:** System Databases (selected)
- Table:** HR.TestDetails
- SQL Script:**

```

184 CREATE TABLE HR.TestDetails(
185     TestDetailsID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
186     TestCategory VARCHAR(30) NOT NULL,
187     TestMaxScore INT NOT NULL,
188     TestDuration TIME NULL
189 );
190
191
192
193
194
195

```
- Messages:**
 - Started executing query at Line 186
 - Commands completed successfully.
 - Total execution time: 00:00:00.012
- Status Bar:** Ln 192, Col 1 Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master

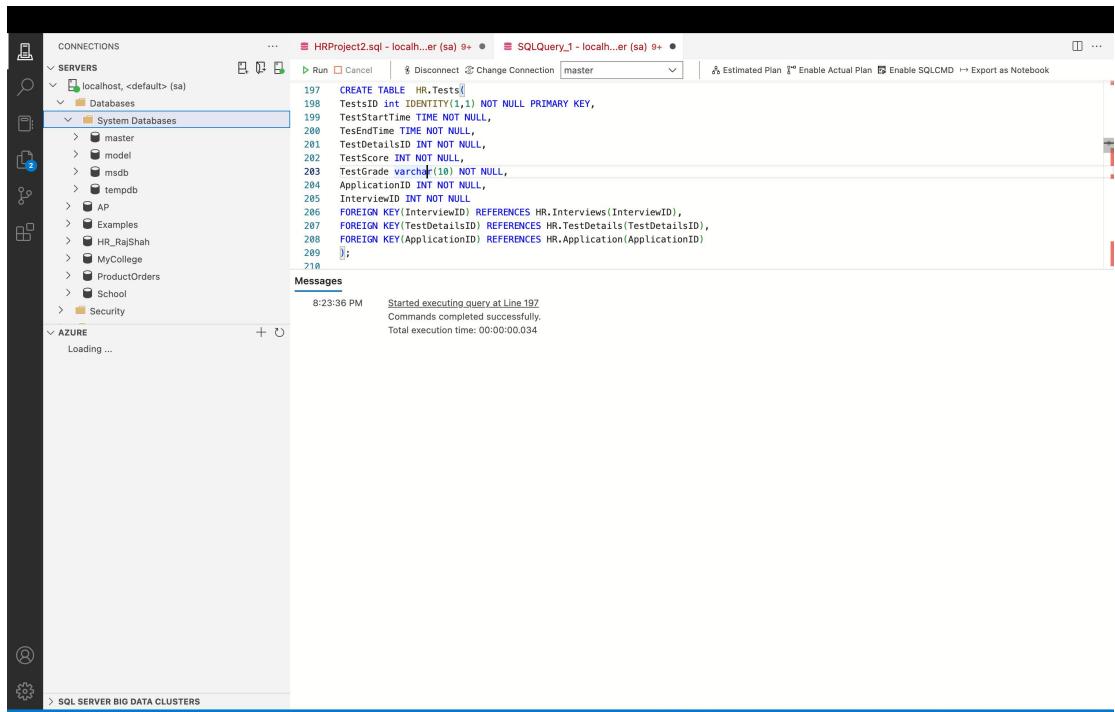
The screenshot shows the SSMS interface with the following details:

- Connections:** HRPProject2.sql - localhost, <default> (sa)
- Databases:** System Databases (selected)
- Table:** HR.Tests
- SQL Script:**

```

196 CREATE TABLE HR.Tests(
197     TestsID int IDENTITY(1,1) NOT NULL PRIMARY KEY,
198     TestStartTime TIME NOT NULL,
199     TestEndTime TIME NOT NULL,
200     TestDetailsID INT NOT NULL,
201     TestScore INT NOT NULL,
202     TestGrade varchar(10) NOT NULL,
203     ApplicationID INT NOT NULL,
204     InterviewID INT NOT NULL
205     FOREIGN KEY(InterviewID) REFERENCES HR.Interviews(InterviewID),
206     FOREIGN KEY(TestDetailsID) REFERENCES HR.TestDetails(TestDetailsID),
207     FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID)
208 );
209

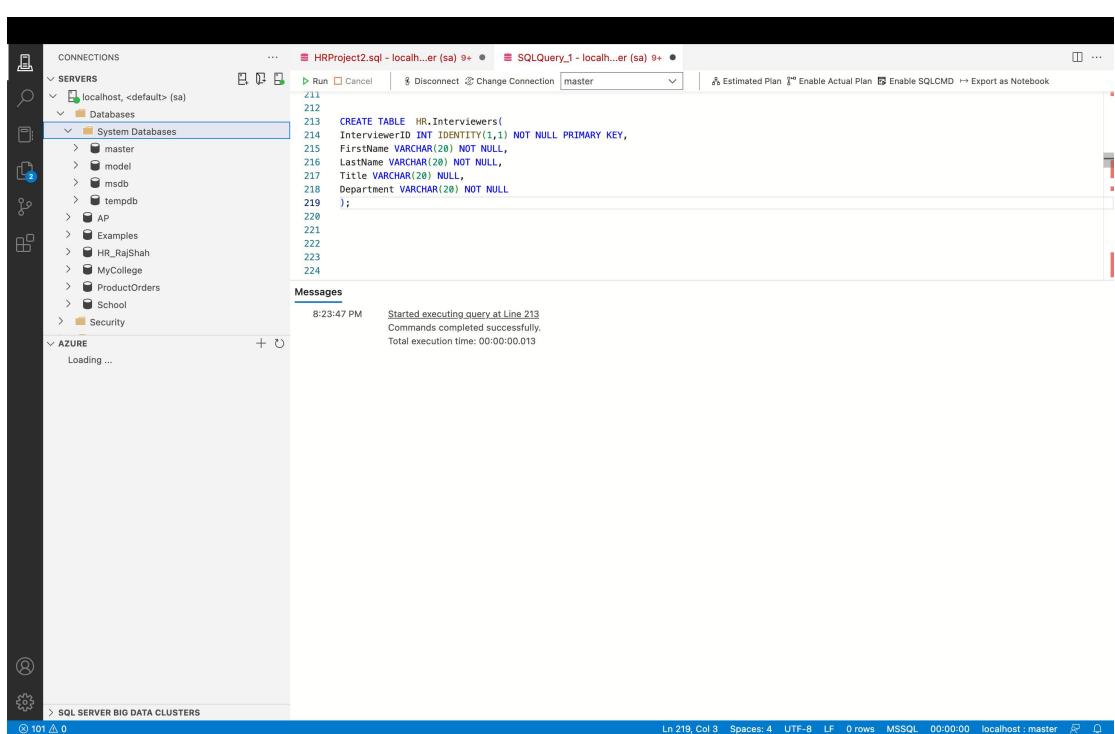
```
- Messages:**
 - Started executing query at Line 186
 - Commands completed successfully.
 - Total execution time: 00:00:00.012
- Status Bar:** Ln 208, Col 3 (467 selected) Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master



```

CREATE TABLE HR.Tests(
    TestID int IDENTITY(1,1) NOT NULL PRIMARY KEY,
    TestStartTime TIME NOT NULL,
    TestEndTime TIME NOT NULL,
    TestDetailsID INT NOT NULL,
    TestScore INT NOT NULL,
    TestGrade varchar(10) NOT NULL,
    ApplicationID INT NOT NULL,
    InterviewID INT NOT NULL
)
FOREIGN KEY(InterviewID) REFERENCES HR.Interviews(InterviewID),
FOREIGN KEY(TestDetailsID) REFERENCES HR.TestDetails(TestDetailsID),
FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID)
;
  
```

Ln 203, Col 17 Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master



```

CREATE TABLE HR.Interviewers(
    InterviewerID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
    FirstName VARCHAR(20) NOT NULL,
    LastName VARCHAR(20) NOT NULL,
    Title VARCHAR(20) NULL,
    Department VARCHAR(20) NOT NULL
)
  
```

Ln 219, Col 3 Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master

```

CREATE TABLE HR.InterviewFeedback(
    InterviewFeedbackID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
    InterviewID INT NOT NULL,
    InterviewerID INT NOT NULL,
    Result VARCHAR(100) NOT NULL,
    Notes VARCHAR(100) NULL
);
FOREIGN KEY(InterviewID) REFERENCES HR.Interviews(InterviewID),
FOREIGN KEY(InterviewerID) REFERENCES HR.Interviewers(InterviewerID)
);

```

Messages

8:24:02 PM Started executing query at Line 225
Commands completed successfully.
Total execution time: 0:00:00.015

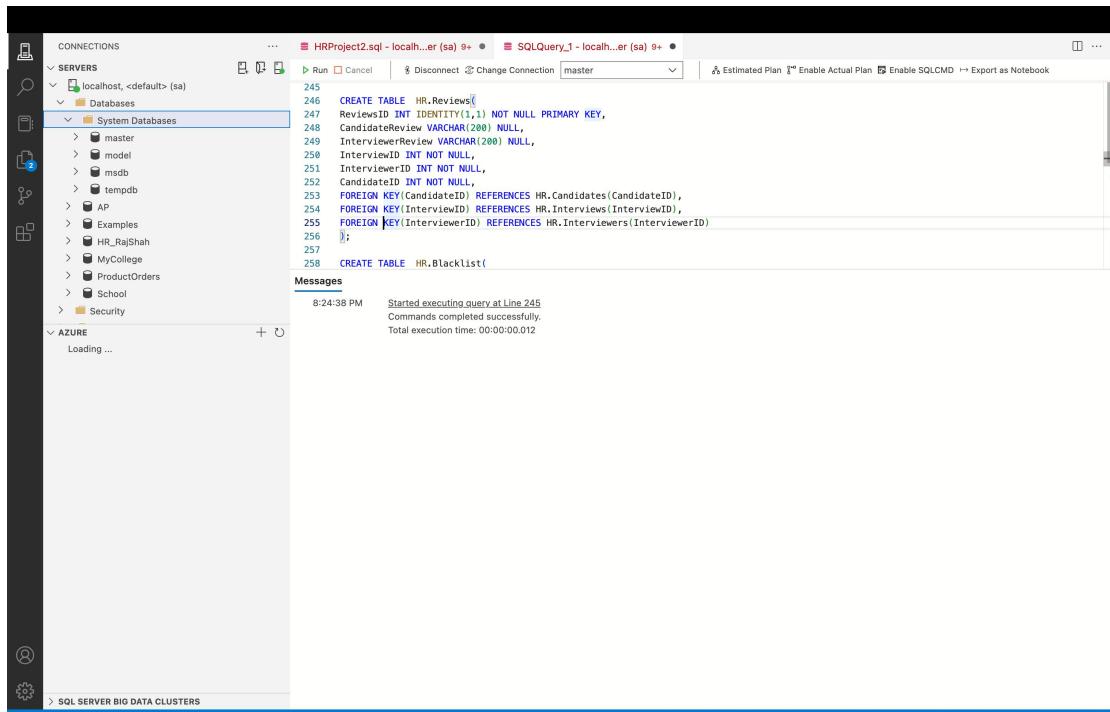
```

CREATE TABLE HR.Evaluation(
    EvaluationID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
    InterviewerID INT NOT NULL,
    ApplicationID INT NOT NULL,
    Result VARCHAR(50) NOT NULL,
    EvaluationNotes VARCHAR(50) NULL
);
FOREIGN KEY(InterviewerID) REFERENCES HR.Interviewers(InterviewerID),
FOREIGN KEY(ApplicationID) REFERENCES HR.Application(ApplicationID)
);

```

Messages

8:24:17 PM Started executing query at Line 235
Commands completed successfully.
Total execution time: 0:00:00.019



```

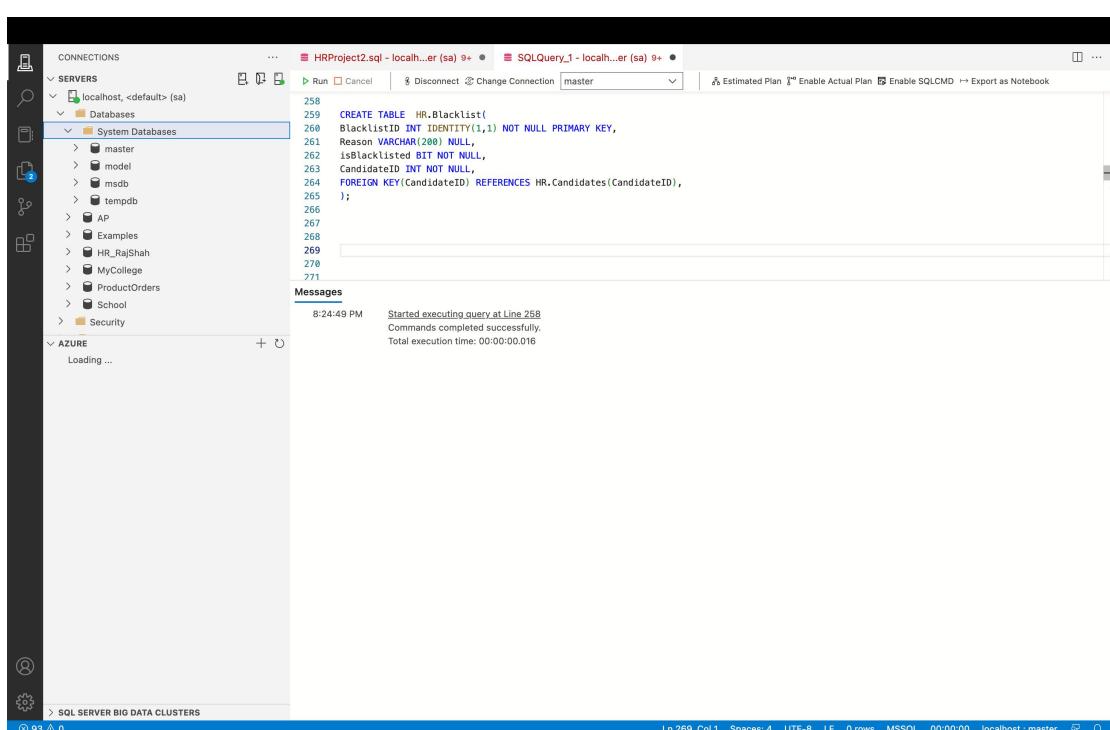
245 CREATE TABLE HR.Reviews(
246     ReviewerID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
247     CandidateReview VARCHAR(200) NULL,
248     InterviewerReview VARCHAR(200) NULL,
249     InterviewID INT NOT NULL,
250     InterviewerID INT NOT NULL,
251     CandidateID INT NOT NULL,
252     FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID),
253     FOREIGN KEY(InterviewID) REFERENCES HR.Interviews(InterviewID),
254     FOREIGN KEY(InterviewerID) REFERENCES HR.Interviewers(InterviewerID)
255 );
256
257 CREATE TABLE HR.Blacklist(
258
259     BlacklistID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
260     Reason VARCHAR(200) NULL,
261     IsBlacklisted BIT NOT NULL,
262     CandidateID INT NOT NULL,
263     FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID),
264 );
265
266
267
268
269
270
271

```

Messages

8:24:38 PM Started executing query at Line 245
Commands completed successfully.
Total execution time: 00:00:00.12

Ln 255, Col 9 Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master



```

258
259     BlacklistID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
260     Reason VARCHAR(200) NULL,
261     IsBlacklisted BIT NOT NULL,
262     CandidateID INT NOT NULL,
263     FOREIGN KEY(CandidateID) REFERENCES HR.Candidates(CandidateID),
264 );
265
266
267
268
269
270
271

```

Messages

8:24:49 PM Started executing query at Line 258
Commands completed successfully.
Total execution time: 00:00:00.016

Ln 269, Col 1 Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 localhost : master

Screenshots for insertion of data in the tables

```

-- Top Query: Insert into Reviews
INSERT INTO Reviews(InterviewerID, InterviewerID, CandidateID, InterviewerReview, CandidateReview)
VALUES
(5,1,1,'Good Candidate', 'Smooth Process'),
(5,1,1,'Good Candidate', 'Smooth Process'),
(2,1,1,'Good Candidate', 'Smooth Process'),
(1,1,1,'Good Candidate', 'Smooth Process'),
(2,1,2,'Bad Candidate', 'Bad Process'),
(3,1,3,'Bad Candidate', 'Bad Process'),
(4,1,4,'Good Candidate', 'Smooth Process'),
(5,1,5,'Bad Candidate', 'Bad Process'),
(6,1,6,'Good Candidate', 'Smooth Process'),
(7,1,7,'Good Candidate', 'Smooth Process'),
(8,1,8,'Good Candidate', 'Process Good')

-- Bottom Query: Insert into Blacklist
INSERT INTO Blacklist(BlacklistID, CandidateID, Reason)
VALUES(1,10,'Not Joined the Company');

```

The changes have been successfully published.

```

-- Interview Feedback Data
INSERT INTO InterviewFeedback (CandidateID, InterviewerID, Rating, Comment)
VALUES
    ('I1', 'R1', 4.5, 'Great interview!'),
    ('I2', 'R2', 4.8, 'Excellent!'),
    ('I3', 'R3', 4.2, 'Good, needs improvement.'), 
    ('I4', 'R4', 4.9, 'Outstanding!'),
    ('I5', 'R5', 4.7, 'Very good.'), 
    ('I6', 'R6', 4.3, 'Satisfactory.'), 
    ('I7', 'R7', 4.6, 'Good.'), 
    ('I8', 'R8', 4.4, 'Good.'), 
    ('I9', 'R9', 4.0, 'Needs improvement.'), 
    ('I10', 'R10', 4.1, 'Good.');

-- Interview Data
SELECT * FROM Interviews;

```



```

-- Test Data
INSERT INTO Tests (ApplicationID, TestStartTime, TestEndTime, TestDetailsID, TestScore, TestGrade, InterviewID)
VALUES
    ('A1', '2023-01-01 10:00:00', '2023-01-01 10:30:00', 'T1', 85, 'A', 'I1'),
    ('A2', '2023-01-01 10:00:00', '2023-01-01 10:30:00', 'T2', 92, 'A+', 'I2'),
    ('A3', '2023-01-01 10:00:00', '2023-01-01 10:30:00', 'T3', 78, 'B+', 'I3'),
    ('A4', '2023-01-01 10:00:00', '2023-01-01 10:30:00', 'T4', 90, 'A+', 'I4'),
    ('A5', '2023-01-01 10:00:00', '2023-01-01 10:30:00', 'T5', 88, 'A+', 'I5'),
    ('A6', '2023-01-01 10:00:00', '2023-01-01 10:30:00', 'T6', 80, 'B+', 'I6'),
    ('A7', '2023-01-01 10:00:00', '2023-01-01 10:30:00', 'T7', 83, 'B+', 'I7'),
    ('A8', '2023-01-01 10:00:00', '2023-01-01 10:30:00', 'T8', 87, 'A+', 'I8'),
    ('A9', '2023-01-01 10:00:00', '2023-01-01 10:30:00', 'T9', 81, 'B+', 'I9'),
    ('A10', '2023-01-01 10:00:00', '2023-01-01 10:30:00', 'T10', 89, 'A+', 'I10');

```

```

    HRProject2.sql - localhost\ah (sa) 9 •
    HRProject2.sql

Run Cancel Disconnect Change Connection HR_RajShah Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook
511 INSERT INTO Evaluation(ApplicationID, InterviewerID, Result, EvaluationNotes)
512 VALUES
513 (1,1,'Accepted','Candidate fit for the role'),
514 (1,5,'Accepted','Candidate fit for the role'),
515 (1,2,'Accepted','Candidate fit for the role'),
516 (2,2,'Rejected','Candidate not fit for the role'),
517 (3,3,'Rejected','Candidate not fit for the role'),
518 (4,4,'Accepted','Candidate fit for the role'),
519 (5,5,'Rejected','Candidate not fit for the role'),
520 (6,6,'Accepted','Candidate fit for the role'),
521 (7,7,'Rejected','Candidate not fit for the role'),
522 (8,8,'Accepted','Candidate fit for the role'),
523 (9,9,'Accepted','Candidate fit for the role'),
524 (10,10,'Accepted','Candidate fit for the role');
525
526
527 SELECT * FROM Evaluation;
528
529

```

Messages

1:35:12 AM Started executing query at Line 511
(12 rows affected)
Total execution time: 00:00:00.031

Ln 526, Col 1 Spaces: 4 UTF-8 LF SQL 0 rows MSSQL 00:00:00 localhost : HR_RajShah

```

    HRProject2.sql - localhost\ah (sa) 9 •
    HRProject2.sql

Run Cancel Disconnect Change Connection HR_RajShah Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook
529 INSERT INTO Reimbursement(RequestNumber,ReimbursementAmount,ReimbursementDate,RequestStatus,RequestType,ApplicationID)
530 VALUES
531
532 (1001,10000,'12-06-2022','Open','Hotel Reservation',1),
533 (1002,10000,'11-16-2022','Open','Air Tickets',2),
534 (1003,20000,'12-05-2022','Open','Air Tickets',3),
535 (1004,10000,'11-14-2022','Open','Car Rental',4),
536 (1005,20000,'11-16-2022','Open','Hotel Reservation',5),
537 (1006,10000,'12-12-2022','Open','Hotel Reservation',6),
538 (1007,14000,'11-06-2022','Open','Car Rental',7),
539 (1008,10000,'11-9-2022','Open','Air Tickets',8),
540 (1009,16000,'12-06-2022','Open','Car Rental',9),
541 (1010,30000,'11-01-2022','Open','Air Tickets',10);
542
543
544 SELECT * FROM Reimbursement;

```

Messages

1:37:41 AM Started executing query at Line 530
(10 rows affected)
Total execution time: 00:00:00.028

Ln 642, Col 1 Spaces: 4 UTF-8 LF SQL 0 rows MSSQL 00:00:00 localhost : HR_RajShah

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. On the left, the Object Explorer tree shows the connection to 'HRProject2.sql - localhost...ah (sa)' and the schema 'dbo'. The 'Candidates' table is selected, with its 'Columns' node expanded. The main pane displays a T-SQL script for inserting data into the 'OnBoarding' table:

```
INSERT INTO OnBoarding(OnboardingStatus,ActualStartDate,JobID,CandidateID)
VALUES
    ('Completed','2023-12-28',4,1),
    ('OnHold','2021-12-12',9,4),
    ('Completed','2021-12-25',10,6),
    ('OnHold','2022-12-28',9,8);

SELECT * FROM OnBoarding;
```

The status bar at the bottom indicates the following details: Line 561, Col 1; Spaces: 4; UTF-8; LF; SQL; 0 rows; MSSQL; 00:00:00; localhost : HR_RajShah.

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. On the left, the Object Explorer displays a database structure for 'HRProject2.sql'. The 'Tables' node under 'dbo' is expanded, showing 'Candidates' as the selected table. The 'Messages' pane at the bottom right shows the execution of a query starting at line 575, which inserts three rows into the 'Complaint' table. The status message indicates 3 rows affected and a total execution time of 0:00:00.0025.

```
563
564
565 INSERT INTO Complaint(ApplicationID,ComplaintDescription,ComplaintInterviewStatus,ComplaintValid)
566     VALUES
567     (2,'Did not get an interview call','Waiting',''),
568     (3,'Did not get an interview call','Waiting',''),
569     (5,'Did not get an interview call','Re-Interview','1');
570
571
572 SELECT * FROM Complaint;
573
574
575
576
577
578
579
580
581
582
583
584
585
586
```

Messages

Started executing query at Line 575
(3 rows affected)
Total execution time: 0:00:00.0025

```

591 INSERT INTO ApplicationStatus(StatusID, ApplicationID, ApplicationStatusChangedDate)
592     VALUES
593         (2,1,'12-06-2022 09:00:10'),
594         (3,2,'12-06-2022 09:00:10'),
595         (3,3,'12-06-2022 09:00:10'),
596         (2,4,'12-06-2022 09:00:10'),
597         (3,5,'12-06-2022 09:00:10'),
598         (2,6,'12-06-2022 09:00:10'),
599         (1,7,'12-06-2022 09:00:10'),
600         (2,8,'12-06-2022 09:00:10'),
601         (3,9,'12-06-2022 09:00:10'),
602         (1,10,'12-06-2022 09:00:10');

603 SELECT * FROM ApplicationStatus;

```

Messages

2:10:56 AM Started executing query at Line 591
(10 rows affected)
Total execution time: 00:00:00.027

Ln 611, Col 1 Spaces: 4 UTF-8 LF SQL 0 rows MSSQL 00:00:00 localhost : HR_RajShah

```

379
380
381
382
383
384
385
386 INSERT INTO TestDetails
387     (TestCategory,TestMaxScore,TestDuration)
388     VALUES
389     ('online','100','01:00:00'),
390     ('onsite','100','01:00:00');

391
392
393 SELECT * FROM TestDetails;
394
395
396

```

Messages

12:46:27 AM Started executing query at Line 386
(2 rows affected)
Total execution time: 00:00:00.016

Ln 386, Col 1 Spaces: 4 UTF-8 LF SQL 0 rows MSSQL 00:00:00 localhost : HR_RajShah

```

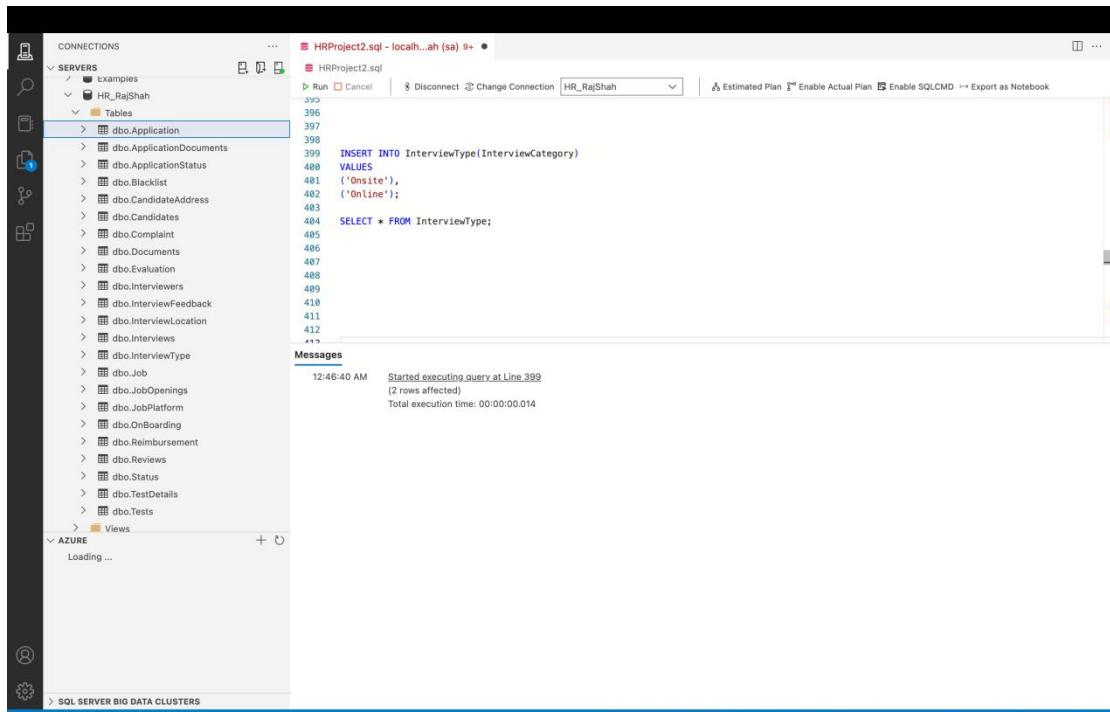
-- Query 1: Insert into Status table
INSERT INTO Status(Status)
VALUES
    ('Declined'),
    ('Accepted'),
    ('Negotiating');

SELECT * from Status;

-- Query 2: Insert into Documents and JobPlatform tables
-- Data for Documents table
INSERT INTO Documents(CandidateID, DocumentName, DocumentType, DocumentURL)
VALUES
    (1, 'Resume', 'PDF', 'https://yourwebsite.com/pages/1'),
    (2, 'CV', 'PDF', 'https://yourwebsite.com/pages/2'),
    (3, 'CV', 'PDF', 'https://yourwebsite.com/pages/3'),
    (4, 'Resume', 'Word', 'https://yourwebsite.com/pages/4'),
    (5, 'CV', 'Word', 'https://yourwebsite.com/pages/5'),
    (6, 'Resume', 'Word', 'https://yourwebsite.com/pages/6'),
    (7, 'CV', 'Word', 'https://yourwebsite.com/pages/7'),
    (8, 'Resume', 'Word', 'https://yourwebsite.com/pages/8'),
    (9, 'Resume', 'PDF', 'https://yourwebsite.com/pages/9'),
    (10, 'Resume', 'PDF', 'https://yourwebsite.com/pages/10');

-- Data for JobPlatform table
INSERT INTO JobPlatform(JobPlatformName, JobPlatformDescription)
VALUES
    ('Indeed', 'Job search engine'),
    ('Glassdoor', 'Job search engine'),
    ('LinkedIn', 'Professional networking site'),
    ('SimplyHired', 'Job search engine'),
    ('monster.com', 'Job search engine'),
    ('Hiring.com', 'Job search engine'),
    ('SmartRecruiters', 'Recruitment software'),
    ('Handshake', 'Recruitment software'),
    ('Craigslist', 'Job search engine'),
    ('Glassdoor', 'Job search engine'),
    ('Indeed', 'Job search engine');

```



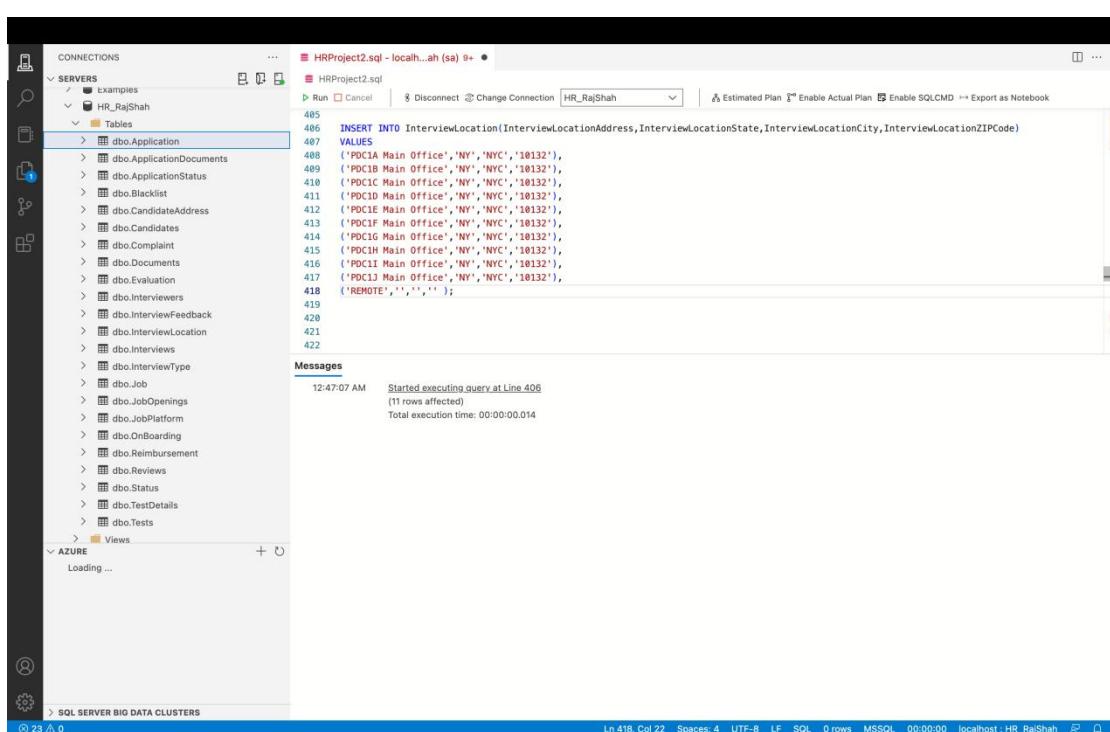
```

    399   INSERT INTO InterviewType(InterviewCategory)
400     VALUES
401     ('Onsite'),
402     ('Online');
403
404   SELECT * FROM InterviewType;
405
406
407
408
409
410
411
412
  
```

Messages

12:46:40 AM Started executing query at Line 399
(2 rows affected)
Total execution time: 00:00:00.014

Ln 413, Col 1 Spaces: 4 UTF-8 LF SQL 0 rows MSSQL 00:00:00 localhost : HR_RajShah



```

405
406   INSERT INTO InterviewLocation(InterviewLocationAddress,InterviewLocationState,InterviewLocationCity,InterviewLocationZIPCode)
407     VALUES
408     ('POC1A Main Office','NY','NYC','10132'),
409     ('POC1B Main Office','NY','NYC','10132'),
410     ('POC1C Main Office','NY','NYC','10132'),
411     ('POC1D Main Office','NY','NYC','10132'),
412     ('POC1E Main Office','NY','NYC','10132'),
413     ('POC1F Main Office','NY','NYC','10132'),
414     ('POC1G Main Office','NY','NYC','10132'),
415     ('POC1H Main Office','NY','NYC','10132'),
416     ('POC1I Main Office','NY','NYC','10132'),
417     ('POC1J Main Office','NY','NYC','10132'),
418     ('REMOTE','','','');
  
```

Messages

12:47:07 AM Started executing query at Line 406
(11 rows affected)
Total execution time: 00:00:00.014

Ln 418, Col 22 Spaces: 4 UTF-8 LF SQL 0 rows MSSQL 00:00:00 localhost : HR_RajShah

```

    INSERT INTO Interviewers(InterviewerID,FirstName,LastName,Title,Department)
    VALUES
    (1,'Robin','Sharma','Executives','IT'),
    (2,'Jon','Repp','Hiring manager','IT'),
    (3,'Marvin','Mandela','HR manager','HR'),
    (4,'Aaron','Wanda','Executives','IT'),
    (5,'Cristiano','Repp','Hiring manager','Marketing'),
    (6,'Yash','Joshi','Hiring manager','Marketing'),
    (7,'Jainam','Gosrani','Executives','Marketing'),
    (8,'Rohan','Singh','Hiring manager','IT'),
    (9,'Sonali','Kubde','Executives','IT'),
    (10,'Saniya','Jain','Hiring manager','IT');
  
```

The changes have been successfully published.

```

    INSERT INTO Application
    (CandidateID,ApplicationTimeStamp,Education,WorkExperience,JobOpeningsID)
    VALUES
    ('1','12-07-2022','MS in Computer Science','Accenture',2),
    ('2','12-07-2022','BSC in Computer Science','LTI',1),
    ('3','12-07-2022','MS in CS','MindTree',3),
    ('4','12-07-2022','MS in Computer Science','Amazon',4),
    ('5','12-07-2022','MS in CS','Chevy',5),
    ('6','12-07-2022','MS in CS','MindTree',4),
    ('7','12-07-2022','MS in Computer Science','Amazon',3),
    ('8','12-07-2022','BSC in Computer Science','LTI',4),
    ('9','11-22-2022','BSC in Computer Science','LTI',5),
    ('10','11-30-2022','MS in Computer Science','Accenture',6);
  
```

The changes have been successfully published.

```

    337  INSERT INTO Job(JobPlatformID,JobMedium,JobType,JobCategory,JobPosition,NoOfVacancies,Salary)
    338      VALUES
    339      (1,'Online','Full Time','IT','Manager',10,100000),
    340      (2,'Online','Summer Internship','Sales','Associate',10,40000),
    341      (3,'Online','Full Time','Testing','Testing',10,120000),
    342      (4,'Online','Full Time','Software','Developer',10,150000),
    343      (5,'Online','Part Time','Software','Associate Developer',10,123000),
    344      (6,'Online','Part Time','Testing','Associate Tester',10,113000),
    345      (7,'Onsite','Full Time','Testing','Associate Manager',10,118000),
    346      (8,'Onsite','Summer Internship','Test','Associate',10,71000),
    347      (9,'Onsite','Full Time','IT','Developer',10,126000),
    348      (10,'Onsite','Summer Internship','IT','Developer',10,50000);
    349
    350  SELECT * FROM Job;
    351
    352
  
```

12:51:25 AM Started executing query at Line 338
(10 rows affected)
Total execution time: 0:00:00.024

The changes have been successfully published.

```

    350  SELECT * FROM Job;
    351
    352
    353
    354  INSERT INTO JobOpenings(JobName,JobStartDate,DatePosted,NoOfPositions,JobID)
    355      VALUES
    356      ('SDE-1','12-12-2022','10-10-2022',10,4),
    357      ('SDE-2','12-18-2023','10-10-2022',10,4),
    358      ('Test Engineer','12-2-2023','10-10-2022',10,3),
    359      ('Data Science Engineer','10-1-2023','10-10-2022',10,9),
    360      ('Machine Learning Engineer','12-22-2023','10-10-2022',10,9),
    361      ('Intern - SDE','12-25-2022','10-20-2022',10,10),
    362      ('Test Intern','12-11-2022','10-10-2022',10,8),
    363      ('Sales Intern','12-17-2022','10-10-2022',10,2),
    364      ('Marketing Intern','12-18-2022','12-10-2022',10,2),
    365      ('Associate Developer','12-27-2022','11-10-2022',10,5);
    366
    367  SELECT * FROM JobOpenings;
    368
    369
    370
  
```

12:54:30 AM Started executing query at Line 354
(10 rows affected)
Total execution time: 0:00:00.048

```

    INSERT INTO JobPlatform(JobPlatformName,JobPlatformDescription)
    VALUES
    ('Job board','Company Job Board'),
    ('Social Media','LinkedIn'),
    ('Social Media','Indeed'),
    ('Social Media','Glassdoor'),
    ('College Connect','Handshake'),
    ('Referral','Company Employee'),
    ('Email','Email'),
    ('SMS','SMS'),
    ('Career Fair','College connect'),
    ('HR Recruiter','Inperson');
    SELECT * from JobPlatform;
  
```

Messages

12:45:11 AM Started executing query at Line 320
(10 rows affected)
Total execution time: 00:00:00.015

Ln 326, Col 13 Spaces: 4 UTF-8 LF SQL 0 rows MSSQL 00:00:00 localhost : HR_RajShah

```

    INSERT INTO CandidateAddress(CandidateID,AddressLine1,City,ZipCode)
    VALUES
    (1,'111 Westcott Street','Syracuse','13210'),
    (2,'712 Westcott Street','Syracuse','13210'),
    (3,'1000 Westcott Street','Syracuse','13210'),
    (4,'456 Westcott Street','Syracuse','13210'),
    (5,'321 Westcott Street','Syracuse','13210'),
    (6,'897 Westcott Street','Syracuse','13210'),
    (7,'567 Westcott Street','Syracuse','13210'),
    (8,'1743 Westcott Street','Syracuse','13210'),
    (9,'678 Westcott Street','Syracuse','13210'),
    (10,'245 Westcott Street','Syracuse','13210');
    SELECT * FROM CandidateAddress;
  
```

Messages

12:44:14 AM Started executing query at Line 286
(10 rows affected)
Total execution time: 00:00:00.029

Ln 299, Col 1 Spaces: 4 UTF-8 LF SQL 0 rows MSSQL 00:00:00 localhost : HR_RajShah

```

610  SELECT * from Application;
611  SELECT * from Candidates;
612
613  INSERT INTO Reviews(InterviewerID, InterviewID, CandidateID, InterviewerReview, CandidateReview)
614  VALUES
615  (5,1,1,'Good Candidate','Smooth Process'),
616  (2,1,1,'Good Candidate','Smooth Process'),
617  (1,1,1,'Good Candidate','Smooth Process'),
618  (2,2,2,'Bad Candidate','Bad Process'),
619  (3,3,3,'Bad Candidate','Bad Process'),
620  (4,4,4,'Good Candidate','Smooth Process'),
621  (5,5,5,'Bad Candidate','Bad Process'),
622  (6,6,6,'Good Candidate','Smooth Process'),
623  (7,7,7,'Good Candidate','Smooth Process'),
624  (8,8,8,'Good Candidate','Process Good')

```

```

478  ON Application.CandidateID=Documents.CandidateID;
479
480
481
482  INSERT INTO ApplicationDocuments(ApplicationID, DocumentID)
483  VALUES
484  (1,1),
485  (2,2),
486  (3,3),
487  (4,4),
488  (5,5),
489  (6,6),
490  (7,7),
491  (8,8),
492  (9,9),
493  (10,10);
494
495
496

```

```

    469
    470
    471
    472     INSERT INTO Interviews(InterviewID,StartTime,EndTime,InterviewTypeID,InterviewLocationID,ApplicationID)
    473     VALUES
    474         (1,'09:30:00','10:30:00',1,2,1),
    475         (2,'09:30:00','10:30:00',1,3,2),
    476         (3,'09:30:00','10:30:00',1,4,3),
    477         (4,'09:30:00','10:30:00',1,6,4),
    478         (5,'09:30:00','10:30:00',1,6,5),
    479         (6,'10:30:00','11:30:00',2,11,6),
    480         (7,'10:30:00','11:30:00',2,11,7),
    481         (8,'10:30:00','11:30:00',2,11,8),
    482         (9,'10:30:00','11:30:00',2,11,9),
    483         (10,'10:30:00','11:30:00',2,11,10),
    484         (11,'10:30:00','11:30:00',1,8,1);
    485
    486     SELECT * FROM Interviews;
    487
    488
  
```

Messages

1:18:24 AM Started executing query at Line 472
(11 rows affected)
Total execution time: 0:00:00.026

The changes have been successfully published.

Screenshots for retrieval of data from the tables

```

    465
    466     INSERT INTO InterviewLocation(InterviewLocationAddress,InterviewLocationState,InterviewLocationCity,InterviewLocationZIPCode)
    467     VALUES
    468         ('PDC1A Main Office','NY','NYC','10132'),
    469         ('PDC1B Main Office','NY','NYC','10132'),
    470         ('PDC1C Main Office','NY','NYC','10132'),
    471         ('PDC1D Main Office','NY','NYC','10132'),
    472         ('PDC1E Main Office','NY','NYC','10132'),
    473         ('PDC1F Main Office','NY','NYC','10132'),
    474         ('PDC1G Main Office','NY','NYC','10132'),
    475         ('PDC1H Main Office','NY','NYC','10132'),
    476         ('PDC1I Main Office','NY','NYC','10132'),
    477         ('PDC1J Main Office','NY','NYC','10132'),
    478         ('REMOTE','','','');
    479
    480     SELECT * FROM InterviewLocation;
    481
    482
  
```

	InterviewLocationID	InterviewLocationAddress	InterviewLocationCity	InterviewLocationState	InterviewLocationZIPCode
1	1	PDC1A Main Office	NYC	NY	10132
2	2	PDC1B Main Office	NYC	NY	10132
3	3	PDC1C Main Office	NYC	NY	10132
4	4	PDC1D Main Office	NYC	NY	10132
5	5	PDC1E Main Office	NYC	NY	10132
6	6	PDC1F Main Office	NYC	NY	10132
7	7	PDC1G Main Office	NYC	NY	10132
8	8	PDC1H Main Office	NYC	NY	10132
9	9	PDC1I Main Office	NYC	NY	10132
10	10	PDC1J Main Office	NYC	NY	10132
11	11	REMOTE			

HRProject2.sql - localhost (sa) 8+ ●

```

350  SELECT * FROM Job;
351
352
353
354  INSERT INTO JobOpenings(JobName,JobStartDate,DatePosted,NoOfPositions,JobID)
355  VALUES
356  ('SDE-1','12-12-2022','10-10-2022',10,4),
357  ('SDE-2','12-18-2023','10-10-2022',10,4),
358  ('Test Engineer','12-2-2023','10-10-2022',10,3),
359  ('Data Scientist','12-22-2023','10-10-2022',10,9),
360  ('Machine Learning Engineer','12-22-2023','10-10-2022',10,9),
361  ('Intern - SDE','12-25-2022','10-20-2022',10,1),
362  ('Test Intern','12-11-2022','10-10-2022',10,9),
363  ('Sales Intern','12-17-2022','10-10-2022',10,7),
364  ('Marketing Intern','12-18-2022','12-18-2022',10,2),
365  ('Associate Developer','12-27-2022','11-10-2022',10,5);
366
367  SELECT * FROM JobOpenings;
368
369
370

```

Results Messages

JobOpeningsID	NoOfPositions	DatePosted	JobStartDate	JobName	JobID
1	10	2022-10-10	2022-12-12	SDE-1	4
2	10	2022-10-10	2023-02-02	SDE-2	4
3	10	2022-10-10	2023-02-02	Test Engineer	3
4	10	2022-10-10	2023-10-01	Data Science Engineer	9
5	10	2022-10-10	2023-12-22	Machine Learning Engineer	9
6	10	2022-10-20	2022-12-25	Intern - SDE	10
7	10	2022-10-10	2022-12-11	Test Intern	8
8	10	2022-10-10	2022-12-17	Sales Intern	2
9	10	2022-12-10	2022-12-18	Marketing Intern	2
10	10	2022-11-10	2022-12-27	Associate Developer	5

Ln 369, Col 1 Spaces: 4 UTF-8 LF SQL 10 rows MSSQL 00:00:00 localhost : HR_RajShah

HRProject2.sql - localhost (sa) 8+ ●

```

335
336
337  INSERT INTO Job(JobPlatformID,JobMedium,JobType,JobCategory,JobPosition,NoOfVacancies,Salary)
338  VALUES
339  (1,'Online','Full Time','IT','Manager',10,100000),
340  (2,'Online','Summer Internship','Sales','Associate',10,40000),
341  (3,'Online','Full Time','Testing','Testing',10,120000),
342  (4,'Online','Full Time','Software','Developer',10,190000),
343  (5,'Online','Part Time','Software','Associate Developer',10,123000),
344  (6,'Online','Part Time','Testing','Associate Tester',10,113000),
345  (7,'Onsite','Full Time','Testing','Associate Manager',10,110000),
346  (8,'Onsite','Summer Internship','Test','Associate',10,71000),
347  (9,'Onsite','Part Time','Developer',10,120000),
348  (10,'Onsite','Summer Internship','IT','Developer',10,50000);
349
350  SELECT * FROM Job];
351
352

```

Results Messages

JobID	JobPlatformID	NoOfVacancies	JobCategory	JobPosition	JobType	JobMedium	Salary
1	1	10	IT	Manager	Full Time	Online	100000.00
2	2	10	Sales	Associate	Summer Internship	Online	40000.00
3	3	10	Testing	Testing	Full Time	Online	120000.00
4	4	10	Software	Developer	Full Time	Online	190000.00
5	5	10	Software	Associate Developer	Part Time	Online	123000.00
6	6	10	Testing	Associate Tester	Part Time	Online	113000.00
7	7	10	Testing	Associate Manager	Full Time	Onsite	110000.00
8	8	8	Test	Associate	Summer Internship	Onsite	71000.00
9	9	9	IT	Developer	Full Time	Onsite	120000.00
10	10	10	IT	Developer	Summer Internship	Onsite	50000.00

The changes have been successfully published.

Ln 350, Col 19 Spaces: 4 UTF-8 LF SQL 10 rows MSSQL 00:00:00 localhost : HR_RajShah

The screenshot shows the SSMS interface with a connection to 'HRProject2.sql - localhost\sa'. The code window contains the following SQL:

```

423 INSERT INTO Interviewers(InterviewerID,FirstName,LastName,Title,Department)
424 VALUES
425 (1,'Robin','Sharma','Executives','IT'),
426 (2,'Jon','Repp','Hiring manager','IT'),
427 (3,'Marvin','Mandela','HR manager','HR'),
428 (4,'Aaron','Wanda','Executives','IT'),
429 (5,'Cristiano','Repp','Hiring manager','IT'),
430 (6,'Yash','Joshi','Hiring manager','Marketing'),
431 (7,'Jainam','Gosrani','Executives','Marketing'),
432 (8,'Rohan','Singh','Hiring manager','IT'),
433 (9,'Sonali','Kubde','Executives','IT'),
434 (10,'Saniya','Jain','Hiring manager','IT');
435
436 SELECT * FROM Interviewers;
437
438
439

```

The results pane displays the data inserted into the Interviewers table:

	InterviewerID	FirstName	LastName	Title	Department
1	1	Robin	Sharma	Executives	IT
2	2	Jon	Repp	Hiring manager	IT
3	3	Marvin	Mandela	HR manager	HR
4	4	Aaron	Wanda	Executives	IT
5	5	Cristiano	Repp	Hiring manager	IT
6	6	Yash	Joshi	Hiring manager	Marketing
7	7	Jainam	Gosrani	Executives	Marketing
8	8	Rohan	Singh	Hiring manager	IT
9	9	Sonali	Kubde	Executives	IT
10	10	Saniya	Jain	Hiring manager	IT

A message box at the bottom right says: 'The changes have been successfully published.'

The screenshot shows the SSMS interface with a connection to 'HRProject2.sql - localhost\sa'. The code window contains the following SQL:

```

399 INSERT INTO InterviewType(InterviewCategory)
400 VALUES
401 ('Onsite'),
402 ('Online');
403
404 SELECT * FROM InterviewType;
405
406
407
408
409
410
411
412

```

The results pane displays the data inserted into the InterviewType table:

	InterviewTypeID	InterviewCategory
1	1	Onsite
2	2	Online

Intro to DBMS

CSE 581

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The left sidebar displays the 'CONNECTIONS' tree, which includes 'System Databases', 'AP', 'Examples', and 'HR_RajShah'. Under 'HR_RajShah', the 'Tables' node is expanded, listing various tables such as Application, ApplicationDocuments, ApplicationStatus, Blacklist, CandidateAddress, Candidates, Complaint, Documents, Evaluation, Interviewers, InterviewFeedback, InterviewLocation, Interviews, and InterviewType. The 'Job' table is currently selected. The main pane shows a T-SQL script for inserting data into the ApplicationDocuments table and selecting all rows from it. Below the script, the 'Results' tab is active, displaying a grid of data with columns: ApplicationDocumentID, DocumentID, and ApplicationID. The data consists of 10 rows, each with values (1, 1), (2, 2), (3, 3), (4, 4), (5, 5), (6, 6), (7, 7), (8, 8), (9, 9), and (10, 10). The bottom status bar indicates the script is at line 491, column 7, with 10 rows affected.

```
... ■ HRProject2.sql - localhost\sa (sa) 9+ ●
  HRProject2.sql
  Run Cancel Disconnect Change Connection HR_RajShah
  Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook

  479 on Application.CandidateID=Documents.CandidateID;
  480
  481
  482 INSERT INTO ApplicationDocuments(ApplicationID,DocumentID)
  483 VALUES
  484 (1,1),
  485 (2,2),
  486 (3,3),
  487 (4,4),
  488 (5,5),
  489 (6,6),
  490 (7,7),
  491 (8,8),
  492 (9,8),
  493 (10,9);
  495 SELECT * FROM ApplicationDocuments;
  496
  497

Results Messages
ApplicationDocumentID DocumentID ApplicationID
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
```

Ln 491, Col 7 Spaces: 4 UTF-8 LF SQL 10 rows MSSQL 00:00:00 localhost : HR RajShah

CONNECTIONS

... HRProject2.sql - localhost (sa) 9+ ...

SERVERS

- System Databases
- AP
- Examples
- HR_RajShah
 - Tables
 - dbo.Application
 - dbo.ApplicationDocuments
 - dbo.ApplicationStatus
 - dbo.Blacklist
 - dbo.CandidateAddress
 - dbo.Candidates
 - dbo.Complaint
 - dbo.Documents
 - dbo.Evaluation
 - dbo.Interviewers
 - dbo.InterviewFeedback
 - dbo.InterviewLocation
 - dbo.Interviews
 - dbo.InterviewType
 - Job
 - JobOpenings
 - JobPlatform
 - Onboarding
 - Reimbursement
 - Reviews
 - Status

RESULTS

VALUES

```
INSERT INTO Application
(CandidateID,ApplicationTimeStamp,Education,WorkExperience,JobOpeningsID)
VALUES
(1,'12-07-2022','MS in Computer Science','Accenture',2),
(2,'12-07-2022','BSC in Computer Science','LTI',1),
(3,'12-07-2022','MS in CS','MindTree',3),
(4,'12-07-2022','MS in Computer Science','Amazon',4),
(5,'12-07-2022','MS in CS','Chewy',5),
(6,'12-07-2022','MS in CS','MindTree',4),
(7,'12-07-2022','MS in Computer Science','Amazon',3),
(8,'12-07-2022','BSC in Computer Science','LTI',4),
(9,'11-22-2022','BSC in Computer Science','LTI',5),
(9,'11-30-2022','MS in Computer Science','Accenture',6);
```

SELECT * FROM Application;

	ApplicationID	CandidateID	Education	WorkExperience	ApplicationTimeStamp	JobOpeningsID	
> dbo.Job	1	1	1	MS in Computer Science	Accenture	2022-12-07	2
> dbo.JobOpenings	2	2	2	BSC in Computer Science	LTI	2022-12-07	1
> dbo.Onboarding	3	3	3	MS in CS	MindTree	2022-12-07	3
> dbo.Reimbursement	4	4	4	MS in Computer Science	Amazon	2022-12-07	4
> dbo.Reviews	5	5	5	MS in CS	Chewy	2022-12-07	5
> dbo.Status	6	6	6	MS in CS	MindTree	2022-12-07	4
+ AZURE	7	7	7	MS in Computer Science	Amazon	2022-12-07	3
	8	8	8	BSC in Computer Science	LTI	2022-12-07	4
	9	9	8	BSC in Computer Science	LTI	2022-11-22	5
	10	10	9	MS in Computer Science	Accenture	2022-11-30	6

SQL SERVER BIO DATA CLUSTERS

CONNECTIONS

... HRProject2.sql - localhost\sa (sa) 9+ ●

SERVERS

✓ Examples

✓ HR_RajShah

Tables

dbo.Application

dbo.ApplicationDocuments

dbo.ApplicationStatus

dbo.Blacklist

dbo.CandidateAddress

dbo.Candidates

dbo.Complaint

dbo.Documents

dbo.Evaluation

dbo.Interviewers

dbo.InterviewFeedback

dbo.InterviewLocation

dbo.Interviews

dbo.InterviewType

dbo.Job

dbo.JobOpenings

dbo.JobPlatform

dbo.Onboarding

dbo.Reimbursement

dbo.Reviews

dbo.Status

dbo.TestDetails

dbo.Tests

Views

AZURE

Loading ...

Run Cancel Disconnect Change Connection HR_RajShah Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook

```
INSERT INTO Status(Status)
VALUES
    ('Declined'),
    ('Accepted'),
    ('Negotiating');

SELECT * from Status;
```

Results Messages

StatusID	Status
1	Declined
2	Accepted
3	Negotiating

SQL SERVER BIG DATA CLUSTERS

Ln 381, Col 1 Spaces: 4 UTF-8 LF SQL 3 rows MSSQL 00:00:00 localhost: HR_RajShah

CONNECTIONS

... HRProject2.sql - localhost (sa) 9+ ●

SERVERS

/ Examples

✓ HR_RajShah

Tables

dbo.Application

dbo.ApplicationDocuments

dbo.ApplicationStatus

dbo.Blacklist

dbo.CandidateAddress

dbo.Candidates

dbo.Complaint

dbo.Documents

dbo.Evaluation

dbo.Interviewers

dbo.InterviewFeedback

dbo.InterviewLocation

dbo.Interviews

dbo.InterviewType

dbo.Job

dbo.JobOpenings

dbo.JobPlatform

dbo.Onboarding

dbo.Reimbursement

dbo.Reviews

dbo.Status

dbo.TestDetails

dbo.Tests

Views

AZURE

Loading ...

Run Cancel Disconnect Change Connection HR_RajShah Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook

379

380

381

382

383

384

385

386

387 **INSERT INTO TestDetails**

388 (**TestCategory**,**TestMaxScore**,**TestDuration**)

389 **VALUES**

390 ('online',100,'01:00:00'),

391 ('onsite',100,'01:00:00');

392

393 **SELECT * FROM TestDetails;**

394

395

396

Results Messages

TestDetailsID	TestCategory	TestMaxScore	TestDuration
1	online	100	01:00:00
2	onsite	100	01:00:00

SQL SERVER BIDS DATA CLUSTERS

Ln 395 Col 1 Spacing: A UTS-B LF SQL 2 rows MSSQL 00:00:00 localhost · HR_RajShah

CONNECTIONS

Servers

- System Databases
- AP
- Examples
- HR_RajShah
- Tables
- dbo.Application
- dbo.ApplicationDocuments
- dbo.ApplicationStatus
- dbo.Blacklist
- dbo.CandidateAddress
- dbo.Candidates
- dbo.Complaint
- dbo.Documents
- dbo.Evaluation
- dbo.Interviewers
- dbo.InterviewFeedback
- dbo.InterviewLocation
- dbo.Interviews
- Columns
- Keys
- Constraints
- Triggers
- Indexes
- Statistics
- dbo.InterviewType
- dbo.Job

AZURE

Loading ...

SQL SERVER BIG DATA CLUSTERS

Ln 487, Col 1 (26 selected) Spaces: 4 UTF-8 LF SQL 11 rows MSSQL 00:00:00 localhost : HR_RajShah

```
HRProject2.sql - localh...ah (sa) 0+ •
HRProject2.sql
Run Cancel Disconnect Change Connection HR_RajShah Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook
469
470
471
472 INSERT INTO Interviews(InterviewID,StartTime,EndTime,InterviewTypeID,InterviewLocationID,ApplicationID)
VALUES
473
474 (1,'09:30:00','10:30:00',1,2,1),
475 (2,'09:30:00','10:30:00',1,3,2),
476 (3,'09:30:00','10:30:00',1,4,3),
477 (4,'09:30:00','10:30:00',1,6,4),
478 (5,'09:30:00','10:30:00',1,6,5),
479 (6,'10:30:00','11:30:00',2,11,6),
480 (7,'10:30:00','11:30:00',2,11,7),
481 (8,'10:30:00','11:30:00',2,11,8),
482 (9,'10:30:00','11:30:00',2,11,9),
483 (10,'10:30:00','11:30:00',2,11,10),
484 (11,'10:30:00','11:30:00',1,8,1);
485
486 SELECT * FROM Interviews;
487
488
489
```

Results Messages

	InterviewID	StartTime	EndTime	ApplicationID	InterviewTypeID	InterviewLocationID
1	1	09:30:00	10:30:00	1	1	2
2	2	09:30:00	10:30:00	2	1	3
3	3	09:30:00	10:30:00	3	1	4
4	4	09:30:00	10:30:00	4	1	6
5	5	09:30:00	10:30:00	5	1	6
6	6	10:30:00	11:30:00	6	2	11
7	7	10:30:00	11:30:00	7	2	11
8	8	10:30:00	11:30:00	8	2	11
9	9	10:30:00	11:30:00	9	2	11
10	10	10:30:00	11:30:00	10	2	11
11	11	10:30:00	11:30:00	1	1	8

CONNECTIONS

Servers

- Examples
- HR_RajShah
- Tables
- dbo.Application
- dbo.ApplicationDocuments
- dbo.ApplicationStatus
- dbo.Blacklist
- dbo.CandidateAddress
- dbo.Candidates
- dbo.Complaint
- dbo.Documents
- dbo.Evaluation
- dbo.Interviewers
- dbo.InterviewFeedback
- dbo.InterviewLocation
- dbo.Interviews
- dbo.InterviewType
- dbo.Job
- dbo.JobOpenings
- dbo.JobPlatform
- dbo.OnBoarding
- dbo.Reimbursement
- dbo.Reviews
- dbo.Status
- dbo.TestDetails
- dbo.Tests
- Views

AZURE

Loading ...

SQL SERVER BIG DATA CLUSTERS

Ln 336, Col 1 Spaces: 4 UTF-8 LF SQL 10 rows MSSQL 00:00:00 localhost : HR_RajShah

```
HRProject2.sql - localh...ah (sa) 0+ •
HRProject2.sql
Run Cancel Disconnect Change Connection HR_RajShah Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317 SELECT * FROM Documents;
318
319
320 INSERT INTO JobPlatform(JobPlatformName,JobPlatformDescription)
VALUES
321 ('Job board','Company Job Board'),
322 ('Social Media','LinkedIn'),
323 ('Social Media','Indeed'),
324 ('Social Media','Glassdoor'),
325 ('Social Media','Handshake'),
326 ('College Connect','Handshake'),
327 ('Referral','Company Employee'),
328 ('Email','Email'),
329 ('SMS','SMS'),
330 ('Career Fair','College connect'),
331 ('HR Recruiter','Inperson');
332
333 SELECT * from JobPlatform;
334
335
```

Results Messages

	JobPlatformID	JobPlatformName	JobPlatformDescription
1	1	Job board	Company Job Board
2	2	Social Media	LinkedIn
3	3	Social Media	Indeed
4	4	Social Media	Glassdoor
5	5	College Connect	Handshake
6	6	Referral	Company Employee
7	7	Email	Email
8	8	SMS	SMS
9	9	Career Fair	College connect
10	10	HR Recruiter	Inperson

```

511 INSERT INTO Evaluation(ApplicationID, InterviewerID, Result, EvaluationNotes)
512 VALUES
513 (1,1,'Accepted','Candidate fit for the role'),
514 (1,1,'Accepted','Candidate fit for the role'),
515 (1,2,'Accepted','Candidate fit for the role'),
516 (2,2,'Rejected','Candidate not fit for the role'),
517 (3,3,'Rejected','Candidate not fit for the role'),
518 (4,4,'Accepted','Candidate fit for the role'),
519 (5,5,'Rejected','Candidate not fit for the role'),
520 (6,6,'Accepted','Candidate fit for the role'),
521 (7,7,'Rejected','Candidate not fit for the role'),
522 (8,8,'Accepted','Candidate fit for the role'),
523 (9,9,'Accepted','Candidate fit for the role'),
524 (10,10,'Accepted','Candidate fit for the role');

525
526
527 SELECT * FROM Evaluation;
528
529

```

	EvaluationID	InterviewerID	ApplicationID	Result	EvaluationNotes
1	1	1	1	Accepted	Candidate fit for the role
2	2	5	1	Accepted	Candidate fit for the role
3	3	2	1	Accepted	Candidate fit for the role
4	4	2	2	Rejected	Candidate not fit for the role
5	5	3	3	Rejected	Candidate not fit for the role
6	6	4	4	Accepted	Candidate fit for the role
7	7	5	5	Rejected	Candidate not fit for the role
8	8	6	6	Accepted	Candidate fit for the role
9	9	7	7	Rejected	Candidate not fit for the role
10	10	8	8	Accepted	Candidate fit for the role
11	11	9	9	Accepted	Candidate fit for the role
12	12	10	10	Accepted	Candidate fit for the role

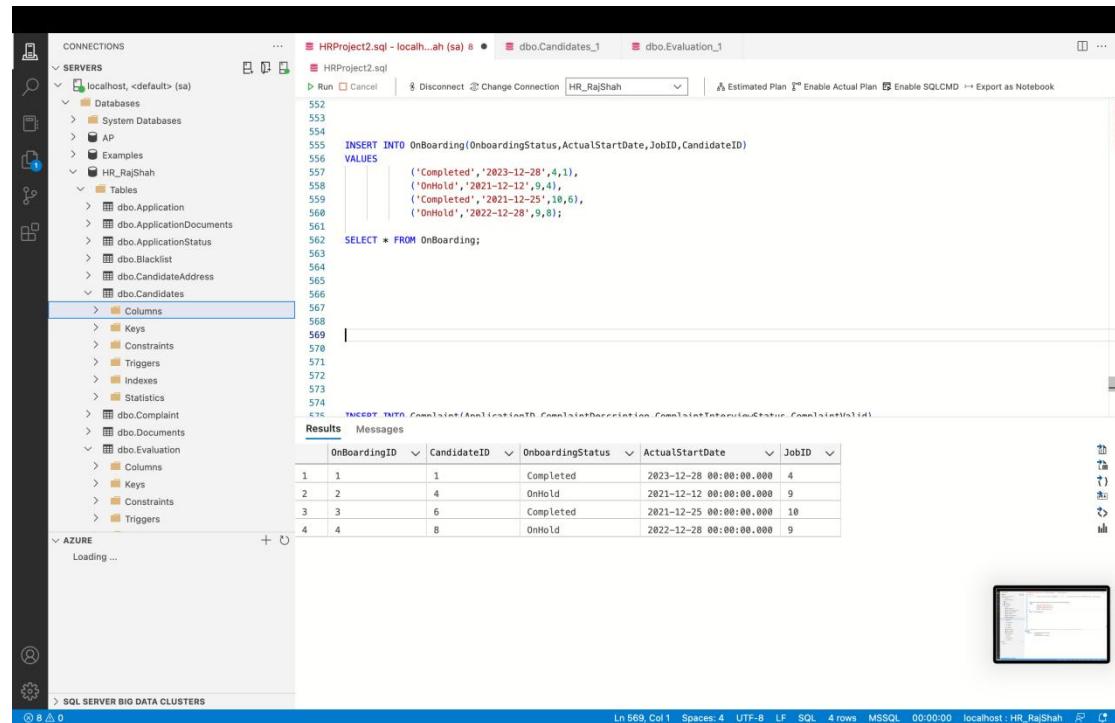
```

530 INSERT INTO Reimbursement(ReimbursementRequestNumber,ReimbursementAmount,ReimbursementDate,RequestStatus,RequestType,ApplicationID)
531 VALUES
532 (1001,10000,'12-06-2022','Open','Hotel Reservation',1),
533 (1002,10000,'11-16-2022','Open','Air Tickets',2),
534 (1003,20000,'12-05-2022','Open','Air Tickets',3),
535 (1004,10000,'11-14-2022','Open','Car Rental',4),
536 (1005,20000,'11-16-2022','Open','Hotel Reservation',5),
537 (1006,10000,'12-12-2022','Open','Hotel Reservation',6),
538 (1007,14000,'11-06-2022','Open','Car Rental',7),
539 (1008,10000,'11-9-2022','Open','Air Tickets',8),
540 (1009,16000,'12-06-2022','Open','Car Rental',9),
541 (1010,30000,'11-01-2022','Open','Air Tickets',10);

542
543
544 SELECT * FROM Reimbursement;
545
546

```

	ReimbursementID	ReimbursementRequestNumber	ReimbursementAmount	ReimbursementDate	ReimbursementMaxAmount	ApplicationID
1	1	1001	10000.00	2022-12-06	30000.00	1
2	2	1002	10000.00	2022-11-16	30000.00	2
3	3	1003	20000.00	2022-12-05	30000.00	3
4	4	1004	10000.00	2022-11-14	30000.00	4
5	5	1005	20000.00	2022-11-16	30000.00	5
6	6	1006	10000.00	2022-12-12	30000.00	6
7	7	1007	14000.00	2022-11-06	30000.00	7
8	8	1008	10000.00	2022-11-09	30000.00	8
9	9	1009	16000.00	2022-12-06	30000.00	9
10	10	1010	30000.00	2022-11-01	30000.00	10



The screenshot shows the SSMS interface with the following details:

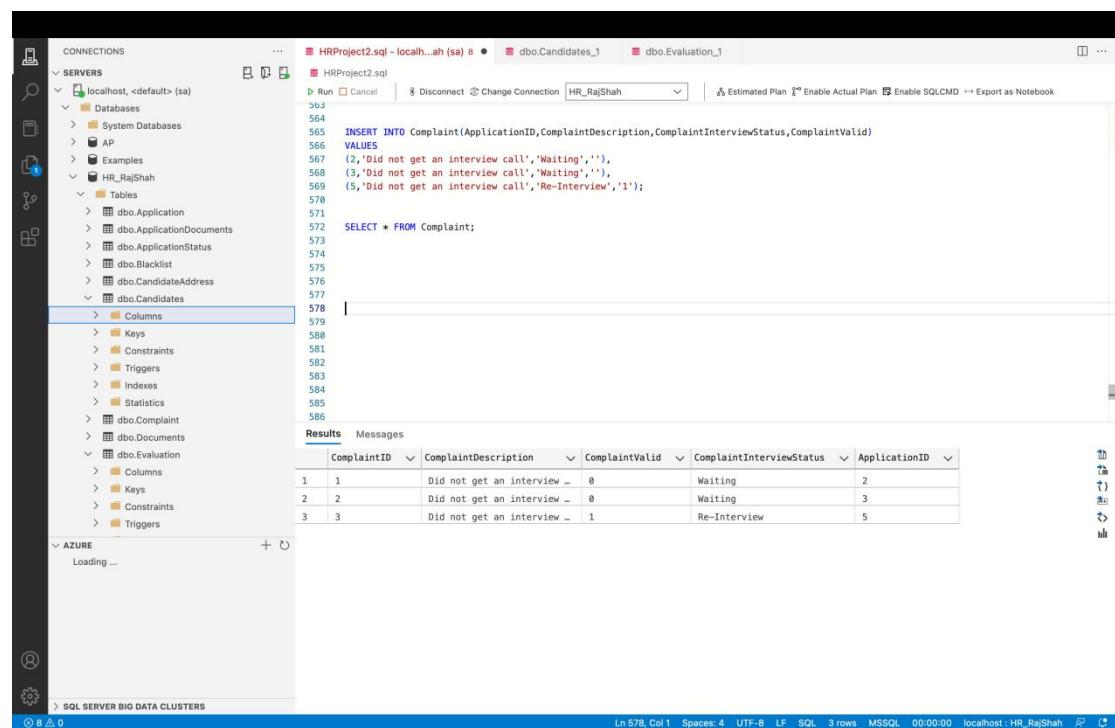
- Connections:** HRProject2.sql - localhost\sa (sa) 8
- Server:** localhost, <default> (sa)
- Database:** HR_RajShah
- Table:** dbo.OnBoarding
- Query:**

```

555  INSERT INTO OnBoarding(OnboardingStatus,ActualStartDate,JobID,CandidateID)
556  VALUES
557      ('Completed','2023-12-28',4,1),
558      ('OnHold','2021-12-12',9,4),
559      ('Completed','2021-12-25',10,6),
560      ('OnHold','2022-12-28',9,8);
561
562  SELECT * FROM OnBoarding;
  
```

- Results:** The results show four rows inserted into the OnBoarding table.

OnBoardingID	CandidateID	OnboardingStatus	ActualStartDate	JobID
1	1	Completed	2023-12-28 00:00:00.000	4
2	2	OnHold	2021-12-12 00:00:00.000	9
3	3	Completed	2021-12-25 00:00:00.000	10
4	4	OnHold	2022-12-28 00:00:00.000	9



The screenshot shows the SSMS interface with the following details:

- Connections:** HRProject2.sql - localhost\sa (sa) 8
- Server:** localhost, <default> (sa)
- Database:** HR_RajShah
- Table:** dbo.Complaint
- Query:**

```

563
564
565  INSERT INTO Complaint(ApplicationID,ComplaintDescription,ComplaintInterviewStatus,ComplaintValid)
566  VALUES
567      (2,'Did not get an interview call','Waiting',0),
568      (3,'Did not get an interview call','Waiting',0),
569      (5,'Did not get an interview call','Re-Interview',1);
570
571  SELECT * FROM Complaint;
  
```

- Results:** The results show three rows inserted into the Complaint table.

ComplaintID	ComplaintDescription	ComplaintValid	ComplaintInterviewStatus	ApplicationID
1	Did not get an interview	0	Waiting	2
2	Did not get an interview	0	Waiting	3
3	Did not get an interview	1	Re-Interview	5

HRProject2.sql - localhost\sa (sa) 9 ● dbo.Candidates_1 ● dbo.Evaluation_1

```

591 INSERT INTO ApplicationStatus(StatusID, ApplicationID, ApplicationStatusChangedDate)
592 VALUES
593 (2,1,'12-06-2022 09:00:10'),
594 (3,2,'12-06-2022 09:00:10'),
595 (3,3,'12-06-2022 09:00:10'),
596 (2,4,'12-06-2022 09:00:10'),
597 (3,5,'12-06-2022 09:00:10'),
598 (2,6,'12-06-2022 09:00:10'),
599 (1,7,'12-06-2022 09:00:10'),
600 (2,8,'12-06-2022 09:00:10'),
601 (3,9,'12-06-2022 09:00:10'),
602 (1,10,'12-06-2022 09:00:10');
603 SELECT * FROM ApplicationStatus;
604
605
606
607
608
609
610
611
612
613

```

Results Messages

	ApplicationStatusID	ApplicationStatusChangedDate	StatusID	ApplicationID
1	1	2022-12-06 09:00:10.000	2	1
2	2	2022-12-06 09:00:10.000	3	2
3	3	2022-12-06 09:00:10.000	3	3
4	4	2022-12-06 09:00:10.000	2	4
5	5	2022-12-06 09:00:10.000	3	5
6	6	2022-12-06 09:00:10.000	2	6
7	7	2022-12-06 09:00:10.000	1	7
8	8	2022-12-06 09:00:10.000	2	8
9	9	2022-12-06 09:00:10.000	3	9
10	10	2022-12-06 09:00:10.000	1	10

HRProject2.sql - localhost\sa (sa) 9 ● dbo.Reviews ● dbo.Candidates_1 ● dbo.Evaluation_1

```

613 INSERT INTO Reviews(InterviewerID, InterviewerID, CandidateID, InterviewerReview, CandidateReview)
614 VALUES
615 (5,1,1,'Good Candidate','Smooth Process'),
616 (2,1,1,'Good Candidate','Smooth Process'),
617 (1,1,1,'Good Candidate','Smooth Process'),
618 (2,2,2,'Bad Candidate','Bad Process'),
619 (3,3,3,'Bad Candidate','Bad Process'),
620 (4,4,4,'Good Candidate','Smooth Process'),
621 (5,5,5,'Bad Candidate','Bad Process'),
622 (6,6,6,'Good Candidate','Smooth Process'),
623 (7,7,7,'Good Candidate','Smooth Process'),
624 (8,8,8,'Good Candidate','Process Good');
625
626 SELECT * from Reviews;

```

Results Messages

	ReviewsID	CandidateReview	InterviewerReview	InterviewerID	InterviewerID	CandidateID
1	1	Smooth Process	Good Candidate	1	5	1
2	2	Smooth Process	Good Candidate	1	2	1
3	3	Smooth Process	Good Candidate	1	1	1
4	4	Bad Process	Bad Candidate	2	2	2
5	5	Bad Process	Bad Candidate	3	3	3
6	6	Smooth Process	Good Candidate	4	4	4
7	7	Bad Process	Bad Candidate	5	5	5
8	8	Smooth Process	Good Candidate	6	6	6
9	9	Smooth Process	Good Candidate	7	7	7
10	10	Process Good	Good Candidate	8	8	8

The changes have been successfully published.

HRProject2.sql - local...ah (sa) 0+ •

Run Cancel Disconnect Change Connection HR_RajShah Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook

```

270
271 INSERT INTO Candidates(FirstName,LastName,Email,Phone,ShortProfile,OfferAccepted)
272 VALUES
273 ('Raj','Shah','rshah26@syr.edu','9876534564','Skilled in SQL','Yes'),
274 ('Harsh','Mandviya','hmand6@syr.edu','6534564123','Skilled in Java',''),
275 ('Jill','Goshrani','jigos2@syr.edu','3456446743','Proficient in DBMS, Python',''),
276 ('Jaynam','Shah','jayna8@syr.edu','3154214546','Skilled in C++','Yes'),
277 ('Mohit','Thakkar','mohit36@syr.edu','3154564546','Skilled in MongoDB',''),
278 ('Manav','Nisar','manan76@syr.edu','3123567548','Skilled in Data Science','Yes'),
279 ('Kavish','Shah','kavash5@syr.edu','3874145464','Proficient in DBMS, Python',''),
280 ('Bhavik','Shah','bhavh96@syr.edu','9126534564','Skilled in MERN','Yes'),
281 ('Rithik','Gujar','rit156@syr.edu','9316534564','Proficient in Java, Python',''),
282 ('Ayush','Shah','ayushhh6@syr.edu','3187534564','Proficient in Node.js, Python','No');
283
284
285
286 SELECT * FROM Candidates;
287

```

Results Messages

CandidateID	FirstName	LastName	Email	Phone	ShortProfile	OfferAccepted
1	Raj	Shah	rshah26@syr.edu	9876534564	Skilled in SQL	Yes
2	Harsh	Mandviya	hmand6@syr.edu	6534564123	Skilled in Java	
3	Jill	Goshrani	jigos2@syr.edu	3456446743	Proficient in DBMS, Python	
4	Jaynam	Shah	jayna8@syr.edu	3154214546	Skilled in C++	Yes
5	Mohit	Thakkar	mohit36@syr.edu	3154564546	Skilled in MongoDB	
6	Manav	Nisar	manan76@syr.edu	3123567548	Skilled in Data Science	Yes
7	Kavish	Shah	kavash5@syr.edu	3874145464	Proficient in DBMS, Python	
8	Bhavik	Shah	bhavh96@syr.edu	9126534564	Skilled in MERN	Yes
9	Rithik	Gujar	rit156@syr.edu	9316534564	Proficient in Java, Python	
10	Ayush	Shah	ayushhh6@syr.edu	3187534564	Proficient in Node.js, Python	No

Ln 285, Col 1 Spaces:4 UTF-8 LF SQL 10 rows MSSQL 00:00:00 localhost : HR_RajShah

HRProject2.sql - local...ah (sa) 0+ •

Run Cancel Disconnect Change Connection HR_RajShah Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook

```

285
286 INSERT INTO CandidateAddress(CandidateID,AddressLine1,City,ZipCode)
287 VALUES
288 ('1','12 Westcott Street','Syracuse','13210'),
289 ('2','712 Westcott Street','Syracuse','13210'),
290 ('3','822 Westcott Street','Syracuse','13210'),
291 ('4','456 Westcott Street','Syracuse','13210'),
292 ('5','321 Westcott Street','Syracuse','13210'),
293 ('6','897 Westcott Street','Syracuse','13210'),
294 ('7','567 Westcott Street','Syracuse','13210'),
295 ('8','1743 Westcott Street','Syracuse','13210'),
296 ('9','678 Westcott Street','Syracuse','13210'),
297 ('10','245 Westcott Street','Syracuse','13210');
298
299
300
301
302 SELECT * FROM CandidateAddress;

```

Results Messages

CandidateAddressID	CandidateID	AddressLine1	AddressLine2	Street	City	ZipCode
1	1	12 Westcott Street	NULL	NULL	Syracuse	13210
2	2	712 Westcott Street	NULL	NULL	Syracuse	13210
3	3	822 Westcott Street	NULL	NULL	Syracuse	13210
4	4	456 Westcott Street	NULL	NULL	Syracuse	13210
5	5	321 Westcott Street	NULL	NULL	Syracuse	13210
6	6	897 Westcott Street	NULL	NULL	Syracuse	13210
7	7	567 Westcott Street	NULL	NULL	Syracuse	13210
8	8	1743 Westcott Street	NULL	NULL	Syracuse	13210
9	9	678 Westcott Street	NULL	NULL	Syracuse	13210
10	10	245 Westcott Street	NULL	NULL	Syracuse	13210

Ln 298, Col 1 Spaces:4 UTF-8 LF SQL 10 rows MSSQL 00:00:00 localhost : HR_RajShah

```

    INSERT INTO Documents(CandidateID, DocumentName, DocumentType, DocumentURL)
    VALUES
    (1, 'Resume', 'PDF', 'https://yourwebsite.com/pages/1'),
    (2, 'CV', 'PDF', 'https://yourwebsite.com/pages/2'),
    (3, 'CV', 'PDF', 'https://yourwebsite.com/pages/3'),
    (4, 'Resume', 'Word', 'https://yourwebsite.com/pages/4'),
    (5, 'CV', 'Word', 'https://yourwebsite.com/pages/5'),
    (6, 'Resume', 'Word', 'https://yourwebsite.com/pages/6'),
    (7, 'CV', 'Word', 'https://yourwebsite.com/pages/7'),
    (8, 'Resume', 'Word', 'https://yourwebsite.com/pages/8'),
    (9, 'Resume', 'PDF', 'https://yourwebsite.com/pages/9'),
    (10, 'Resume', 'PDF', 'https://yourwebsite.com/pages/10')

    SELECT * FROM Documents;

    INSERT INTO JobPlatform(JobPlatformName, JobPlatformDescription)
    VALUES
    ('LinkedIn', 'Job Platform for LinkedIn'),
    ('Indeed', 'Job Platform for Indeed'),
    ('Glassdoor', 'Job Platform for Glassdoor'),
    ('Hiring.com', 'Job Platform for Hiring.com'),
    ('SimplyHired', 'Job Platform for SimplyHired'),
    ('SmartRecruiters', 'Job Platform for SmartRecruiters'),
    ('Recruiter.com', 'Job Platform for Recruiter.com'),
    ('Jobvite', 'Job Platform for Jobvite'),
    ('Handshake', 'Job Platform for Handshake'),
    ('Pulse', 'Job Platform for Pulse')
  
```

Results

DocumentID	CandidateID	DocumentName	DocumentType	DocumentURL
1	1	Resume	PDF	https://yourwebsite.com/p...
2	2	CV	PDF	https://yourwebsite.com/p...
3	3	CV	PDF	https://yourwebsite.com/p...
4	4	Resume	Word	https://yourwebsite.com/p...
5	5	CV	Word	https://yourwebsite.com/p...
6	6	Resume	Word	https://yourwebsite.com/p...
7	7	CV	Word	https://yourwebsite.com/p...
8	8	Resume	Word	https://yourwebsite.com/p...
9	9	Resume	PDF	https://yourwebsite.com/p...
10	10	Resume	PDF	https://yourwebsite.com/p...

```

    INSERT INTO Blacklist(BlacklistID, CandidateID, Reason)
    VALUES(1, 10, 'Not Joined the Company');

    select * from Blacklist;
  
```

Results

BlacklistID	Reason	CandidateID
1	Not Joined the Company	10

The changes have been successfully published.

CONNECTIONS

SERVERS

- > dbo.Interviews
- > dbo.InterviewType
- > dbo.Job
- > dbo.JobOpenings
- > dbo.JobPlatform
- > dbo.OnBoarding
- > dbo.Reimbursement
- > dbo.Reviews
- > dbo.Status
- > dbo.TestDetails
- > **dbo.Tests**
- > **Columns**
- > Keys
- > Constraints
- > Triggers
- > Indexes
- > Statistics
- > Views

HRProject2.sql - localhost (sa) 8

Run Cancel Disconnect Change Connection HR_RajShah Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook

```
493 INSERT INTO InterviewFeedback(InterviewID, InterviewerID, Result, Notes)
494 VALUES
495 (1,1,'Pass','Passed the interview'),
496 (1,5,'Pass','Passed the interview'),
497 (2,2,'Fail','Failed the interview'),
498 (3,3,'Fail','Failed the interview'),
499 (4,4,'Pass','Passed the interview'),
500 (5,5,'Pass','Passed the interview'),
501 (6,6,'Pass','Passed the interview'),
502 (7,7,'Fail','Failed the interview'),
503 (8,8,'Pass','Passed the interview'),
504 (9,9,'Fail','Failed the interview'),
505 (10,10,'Pass','Passed the interview'),
506 (11,2,'Pass','Passed the interview');

508 SELECT * from InterviewFeedback;
```

509

510

511 TRUNCATE TABLE EvaluationApplications..InterviewFeedback..Result..EvaluationNotes

	InterviewFeedbackID	InterviewID	InterviewerID	Result	Notes
1	1	1	1	Pass	Passed the interview
2	2	1	5	Pass	Passed the interview
3	3	2	2	Fail	Failed the interview
4	4	3	3	Fail	Failed the interview
5	5	4	4	Pass	Passed the interview
6	6	5	5	Pass	Passed the interview
7	7	6	6	Pass	Passed the interview
8	8	7	7	Fail	Failed the interview
9	9	8	8	Pass	Passed the interview
10	10	9	9	Fail	Failed the interview
11	11	10	10	Pass	Passed the interview
12	12	11	2	Pass	Passed the interview

The screenshot shows the SSMS interface with the following details:

- Connections:** HRProject2.sql - localhost\sa (sa)
- Servers:** dbo
- Tables:** Tests
- Table Data:**

TestsID	TestStartTime	TesEndTime	TestDetailsID	TestScore	TestGrade	ApplicationID	InterviewID
1	09:30:00	10:30:00	1	81	A	1	1
2	09:30:00	10:30:00	1	71	B+	2	2
3	09:30:00	10:30:00	1	51	C+	3	3
4	09:30:00	10:30:00	1	100	A+	4	4
5	09:30:00	10:30:00	1	51	C+	5	5
6	10:30:00	11:30:00	2	71	B+	6	6
7	10:30:00	11:30:00	2	81	A	7	7
8	10:30:00	11:30:00	2	71	B+	8	8
9	10:30:00	11:30:00	2	51	C+	9	9
10	10:30:00	11:30:00	2	71	B+	10	10
11	10:30:00	11:30:00	1	91	A+	1	1
12	10:30:00	11:30:00	1	85	A	1	1

All the screenshots attached above contains my name in Database as HR RajShah.