

Final Project SQL Part 3

By Holden Weber

Create PHYSICIAN_TABLE

```
CREATE PROC create_PHYSICIAN_Table
AS
BEGIN
    SET NOCOUNT ON;

    IF NOT EXISTS (SELECT * FROM sysobjects WHERE name='PHYSICIAN' AND xtype='U')
    BEGIN
        CREATE TABLE PHYSICIAN (
            PHYSICIAN_ID INT CONSTRAINT PK_PHYSICIAN_ID PRIMARY KEY,
            FNAME VARCHAR(25) NOT NULL,
            LNAME VARCHAR(25) NOT NULL,
            MIDINIT CHAR(1),
            SALARY DECIMAL(10, 2) NOT NULL,
            CELLPHONE VARCHAR(15),
            HOMEPHONE VARCHAR(15),
            WORKPHONE VARCHAR(15),
            WORKEMAIL VARCHAR(50),
            HOMEEMAIL VARCHAR(50),
            GENDER VARCHAR(6) NOT NULL CONSTRAINT CHK_GENDER2 CHECK (GENDER
IN ('F','M','Female', 'Male', 'NA','f','m','female', 'male', 'na')),
            STREET VARCHAR(30) NOT NULL,
            CITY VARCHAR(30) NOT NULL,
            STATE_ID VARCHAR(2) NOT NULL,
            ZIP VARCHAR(6) NOT NULL,
            HIRED_DATE DATE NOT NULL,
            SPECIALTY_MAIN VARCHAR(50),
            SPECIALTY_OTHER2 VARCHAR(50),
            SPECIALTY_OTHER3 VARCHAR(50),
            SPECIALTY_OTHER4 VARCHAR(50),
            SPECIALTY_OTHER5 VARCHAR(50)
        );

        PRINT 'PHYSICIAN table created successfully!';
    END
    ELSE
    BEGIN
        PRINT 'Table PHYSICIAN already exists!';
    END
END
```

```
END;  
GO
```

```
EXEC create_PHYSICIAN_Table;
```

Add a New Physician

```
CREATE PROC add_PHYSICIAN (  
    @PHYSICIAN_ID INT,  
    @FNAME VARCHAR(25),  
    @LNAME VARCHAR(25),  
    @MIDINIT CHAR(1),  
    @SALARY DECIMAL(10, 2),  
    @CELLPHONE VARCHAR(15),  
    @HOMEPHONE VARCHAR(15),  
    @WORKPHONE VARCHAR(15),  
    @WORKEMAIL VARCHAR(50),  
    @HOMEEMAIL VARCHAR(50),  
    @GENDER VARCHAR(6),  
    @STREET VARCHAR(30),  
    @CITY VARCHAR(30),  
    @STATE_ID VARCHAR(2),  
    @ZIP VARCHAR(6),  
    @HIRED_DATE DATE,  
    @SPECIALTY_MAIN VARCHAR(50),  
    @SPECIALTY_OTHER2 VARCHAR(50),  
    @SPECIALTY_OTHER3 VARCHAR(50),  
    @SPECIALTY_OTHER4 VARCHAR(50),  
    @SPECIALTY_OTHER5 VARCHAR(50)  
)  
AS  
BEGIN  
    SET NOCOUNT ON;  
  
    BEGIN TRY  
        INSERT INTO PHYSICIAN (PHYSICIAN_ID, FNAME, LNAME, MIDINIT, SALARY,  
CELLPHONE, HOMEPHONE, WORKPHONE, WORKEMAIL, HOMEEMAIL, GENDER,  
STREET, CITY, STATE_ID, ZIP, HIRED_DATE, SPECIALTY_MAIN, SPECIALTY_OTHER2,  
SPECIALTY_OTHER3, SPECIALTY_OTHER4, SPECIALTY_OTHER5)  
        VALUES (@PHYSICIAN_ID, @FNAME, @LNAME, @MIDINIT, @SALARY,  
@CELLPHONE, @HOMEPHONE, @WORKPHONE, @WORKEMAIL, @HOMEEMAIL,  
@GENDER, @STREET, @CITY, @STATE_ID, @ZIP, @HIRED_DATE, @SPECIALTY_MAIN,
```

```
@SPECIALTY_OTHER2, @SPECIALTY_OTHER3, @SPECIALTY_OTHER4,  
@SPECIALTY_OTHER5);
```

```
    PRINT 'Physician added successfully!';  
END TRY  
BEGIN CATCH  
    PRINT 'Error occurred while adding the physician!';  
END CATCH  
END;  
GO
```

```
EXEC add_PHYSICIAN 1, 'John', 'Doe', 'M', 100000, '1234567890', '0987654321',  
'1122334455', 'john.doe@work.com', 'john.doe@home.com', 'Male', '123 Main St', 'Citytown',  
'NY', '12345', '2023-01-01', 'Cardiology', 'Oncology', 'Pediatrics', 'Dermatology', 'Neurology';
```

Delete a Physician by PHYSICIAN_ID

```
CREATE PROC delete_PHYSICIAN (  
    @PHYSICIAN_ID INT  
)  
AS  
BEGIN  
    SET NOCOUNT ON;  
  
    BEGIN TRY  
        DELETE FROM PHYSICIAN WHERE PHYSICIAN_ID = @PHYSICIAN_ID;  
  
        IF @@ROWCOUNT = 0  
            PRINT 'No physician found with the given PHYSICIAN_ID!';  
        ELSE  
            PRINT 'Physician deleted successfully!';  
    END TRY  
    BEGIN CATCH  
        PRINT 'Error occurred while deleting the physician!';  
    END CATCH  
END;  
GO
```

```
EXEC delete_PHYSICIAN 1;
```

Update Physician Data

```
CREATE PROC modify_PHYSICIAN (  
    @PHYSICIAN_ID INT,
```

```

@FNAME VARCHAR(25) = NULL,
@LNAME VARCHAR(25) = NULL,
@MIDINIT CHAR(1) = NULL,
@SALARY DECIMAL(10, 2) = NULL,
@CELLPHONE VARCHAR(15) = NULL,
@HOMEPHONE VARCHAR(15) = NULL,
@WORKPHONE VARCHAR(15) = NULL,
@WORKEMAIL VARCHAR(50) = NULL,
@HOMEEMAIL VARCHAR(50) = NULL,
@GENDER VARCHAR(6) = NULL,
@STREET VARCHAR(30) = NULL,
@CITY VARCHAR(30) = NULL,
@STATE_ID VARCHAR(2) = NULL,
@ZIP VARCHAR(6) = NULL,
@HIRED_DATE DATE = NULL,
@SPECIALTY_MAIN VARCHAR(50) = NULL,
@SPECIALTY_OTHER2 VARCHAR(50) = NULL,
@SPECIALTY_OTHER3 VARCHAR(50) = NULL,
@SPECIALTY_OTHER4 VARCHAR(50) = NULL,
@SPECIALTY_OTHER5 VARCHAR(50) = NULL
)
AS
BEGIN
    SET NOCOUNT ON;

    BEGIN TRY
        UPDATE PHYSICIAN
        SET
            FNAME = COALESCE(@FNAME, FNAME),
            LNAME = COALESCE(@LNAME, LNAME),
            MIDINIT = COALESCE(@MIDINIT, MIDINIT),
            SALARY = COALESCE(@SALARY, SALARY),
            CELLPHONE = COALESCE(@CELLPHONE, CELLPHONE),
            HOMEPHONE = COALESCE(@HOMEPHONE, HOMEPHONE),
            WORKPHONE = COALESCE(@WORKPHONE, WORKPHONE),
            WORKEMAIL = COALESCE(@WORKEMAIL, WORKEMAIL),
            HOMEEMAIL = COALESCE(@HOMEEMAIL, HOMEEMAIL),
            GENDER = COALESCE(@GENDER, GENDER),
            STREET = COALESCE(@STREET, STREET),
            CITY = COALESCE(@CITY, CITY),
            STATE_ID = COALESCE(@STATE_ID, STATE_ID),
            ZIP = COALESCE(@ZIP, ZIP),
            HIRED_DATE = COALESCE(@HIRED_DATE, HIRED_DATE),
            SPECIALTY_MAIN = COALESCE(@SPECIALTY_MAIN, SPECIALTY_MAIN),

```

```

    SPECIALTY_OTHER2 = COALESCE(@SPECIALTY_OTHER2, SPECIALTY_OTHER2),
    SPECIALTY_OTHER3 = COALESCE(@SPECIALTY_OTHER3, SPECIALTY_OTHER3),
    SPECIALTY_OTHER4 = COALESCE(@SPECIALTY_OTHER4, SPECIALTY_OTHER4),
    SPECIALTY_OTHER5 = COALESCE(@SPECIALTY_OTHER5, SPECIALTY_OTHER5)
WHERE PHYSICIAN_ID = @PHYSICIAN_ID;

```

```

IF @@ROWCOUNT = 0
    PRINT 'No physician found with the given PHYSICIAN_ID!';
ELSE
    PRINT 'Physician updated successfully!';
END TRY
BEGIN CATCH
    PRINT 'Error occurred while updating the physician!';
END CATCH
END;
GO

```

```

EXEC modify_PHYSICIAN 1, 'Jane', 'Smith', 'M', 150000, '1234567890', NULL, NULL,
'jane.smith@work.com', NULL, 'Female', '456 Elm St', 'Cityville', 'NY', '67890', '2023-05-01',
'Neurology', NULL, NULL, NULL, NULL;

```

Display Physician Data

```

CREATE PROC display_PHYSICIAN (
    @PHYSICIAN_ID INT = NULL
)
AS
BEGIN
    SET NOCOUNT ON;

    IF @PHYSICIAN_ID IS NULL
    BEGIN
        SELECT * FROM PHYSICIAN;
    END
    ELSE
    BEGIN
        SELECT * FROM PHYSICIAN WHERE PHYSICIAN_ID = @PHYSICIAN_ID;
    END
END;
GO

```

```

EXEC display_PHYSICIAN; -- Display all physicians
EXEC display_PHYSICIAN 1; -- Display specific physician with ID 1

```

Create PATIENT Table

```
CREATE PROC create_PATIENT_TABLE
AS
BEGIN
    SET NOCOUNT ON;

    IF NOT EXISTS (SELECT * FROM sysobjects WHERE name='PATIENT' AND xtype='U')
    BEGIN
        CREATE TABLE PATIENT (
            PATIENT_ID INT CONSTRAINT PK_PATIENT_ID PRIMARY KEY,
            FNAME VARCHAR(25) NOT NULL,
            LNAME VARCHAR(25) NOT NULL,
            MIDINIT CHAR(1),
            CELLPHONE VARCHAR(15),
            HOMEPHONE VARCHAR(15),
            WORKPHONE VARCHAR(15),
            WORKEMAIL VARCHAR(50),
            HOMEEMAIL VARCHAR(50),
            DOB DATE NOT NULL CHECK(DOB < GETDATE()),
            GENDER VARCHAR(6) NOT NULL CONSTRAINT CHK_GENDER CHECK (GENDER
IN ('F','M','Female', 'Male', 'NA')),
            STREET VARCHAR(30),
            CITY VARCHAR(30),
            STATE_ID VARCHAR(2),
            ZIP VARCHAR(6),
            PH_STREET VARCHAR(30),
            PH_CITY VARCHAR(30),
            PH_STATE_ID VARCHAR(2),
            PH_ZIP VARCHAR(6),
            BALANCE DECIMAL(9,2) DEFAULT 0
        );

        PRINT 'PATIENT table created successfully!';
    END
    ELSE
    BEGIN
        PRINT 'PATIENT table already exists!';
    END
END;
GO

EXEC create_PATIENT_TABLE;
```

Create a New Patient

```
CREATE PROC add_PATIENT (  
    @PATIENT_ID INT,  
    @FNAME VARCHAR(25),  
    @LNAME VARCHAR(25),  
    @MIDINIT CHAR(1) = NULL,  
    @CELLPHONE VARCHAR(15) = NULL,  
    @HOMEPHONE VARCHAR(15) = NULL,  
    @WORKPHONE VARCHAR(15) = NULL,  
    @WORKEMAIL VARCHAR(50) = NULL,  
    @HOMEEMAIL VARCHAR(50) = NULL,  
    @DOB DATE,  
    @GENDER VARCHAR(6),  
    @STREET VARCHAR(30) = NULL,  
    @CITY VARCHAR(30) = NULL,  
    @STATE_ID VARCHAR(2) = NULL,  
    @ZIP VARCHAR(6) = NULL,  
    @PH_STREET VARCHAR(30) = NULL,  
    @PH_CITY VARCHAR(30) = NULL,  
    @PH_STATE_ID VARCHAR(2) = NULL,  
    @PH_ZIP VARCHAR(6) = NULL,  
    @BALANCE DECIMAL(9,2) = 0  
)  
AS  
BEGIN  
    SET NOCOUNT ON;  
  
    BEGIN TRY  
        INSERT INTO PATIENT (PATIENT_ID, FNAME, LNAME, MIDINIT, CELLPHONE,  
        HOMEPHONE, WORKPHONE, WORKEMAIL, HOMEEMAIL, DOB, GENDER, STREET, CITY,  
        STATE_ID, ZIP, PH_STREET, PH_CITY, PH_STATE_ID, PH_ZIP, BALANCE)  
        VALUES (@PATIENT_ID, @FNAME, @LNAME, @MIDINIT, @CELLPHONE,  
        @HOMEPHONE, @WORKPHONE, @WORKEMAIL, @HOMEEMAIL, @DOB, @GENDER,  
        @STREET, @CITY, @STATE_ID, @ZIP, @PH_STREET, @PH_CITY, @PH_STATE_ID,  
        @PH_ZIP, @BALANCE);  
  
        PRINT 'Patient added successfully!';  
    END TRY  
    BEGIN CATCH
```

```
        PRINT 'Error occurred while adding the patient!';
    END CATCH
END;
GO
```

```
EXEC add_PATIENT 1, 'Alice', 'Johnson', 'A', '1234567890', NULL, NULL, 'alice.j@work.com',
'alice.j@home.com', '1985-06-15', 'F', '123 Main St', 'Cityville', 'NY', '12345', '456 Pine St',
'Townsville', 'CA', '67890', 50.00;
```

Modify Patient Data

```
CREATE PROC modify_PATIENT (
    @PATIENT_ID INT,
    @FNAME VARCHAR(25) = NULL,
    @LNAME VARCHAR(25) = NULL,
    @MIDINIT CHAR(1) = NULL,
    @CELLPHONE VARCHAR(15) = NULL,
    @HOMEPHONE VARCHAR(15) = NULL,
    @WORKPHONE VARCHAR(15) = NULL,
    @WORKEMAIL VARCHAR(50) = NULL,
    @HOMEEMAIL VARCHAR(50) = NULL,
    @DOB DATE = NULL,
    @GENDER VARCHAR(6) = NULL,
    @STREET VARCHAR(30) = NULL,
    @CITY VARCHAR(30) = NULL,
    @STATE_ID VARCHAR(2) = NULL,
    @ZIP VARCHAR(6) = NULL,
    @PH_STREET VARCHAR(30) = NULL,
    @PH_CITY VARCHAR(30) = NULL,
    @PH_STATE_ID VARCHAR(2) = NULL,
    @PH_ZIP VARCHAR(6) = NULL,
    @BALANCE DECIMAL(9,2) = NULL
)
AS
BEGIN
    SET NOCOUNT ON;

    BEGIN TRY
        UPDATE PATIENT
        SET
            FNAME = COALESCE(@FNAME, FNAME),
            LNAME = COALESCE(@LNAME, LNAME),
            MIDINIT = COALESCE(@MIDINIT, MIDINIT),
            CELLPHONE = COALESCE(@CELLPHONE, CELLPHONE),
```



```

HOMEPHONE = COALESCE(@HOMEPHONE, HOMEPHONE),
WORKPHONE = COALESCE(@WORKPHONE, WORKPHONE),
WORKEMAIL = COALESCE(@WORKEMAIL, WORKEMAIL),
HOMEEMAIL = COALESCE(@HOMEEMAIL, HOMEEMAIL),
DOB = COALESCE(@DOB, DOB),
GENDER = COALESCE(@GENDER, GENDER),
STREET = COALESCE(@STREET, STREET),
CITY = COALESCE(@CITY, CITY),
STATE_ID = COALESCE(@STATE_ID, STATE_ID),
ZIP = COALESCE(@ZIP, ZIP),
PH_STREET = COALESCE(@PH_STREET, PH_STREET),
PH_CITY = COALESCE(@PH_CITY, PH_CITY),
PH_STATE_ID = COALESCE(@PH_STATE_ID, PH_STATE_ID),
PH_ZIP = COALESCE(@PH_ZIP, PH_ZIP),
BALANCE = COALESCE(@BALANCE, BALANCE)
WHERE PATIENT_ID = @PATIENT_ID;

```

```

IF @@ROWCOUNT = 0
    PRINT 'No patient found with the given PATIENT_ID!';
ELSE
    PRINT 'Patient updated successfully!';
END TRY
BEGIN CATCH
    PRINT 'Error occurred while updating the patient!';
END CATCH
END;
GO

```

```

EXEC modify_PATIENT 1, 'Alice', NULL, NULL, '9876543210', NULL, NULL, NULL, NULL,
NULL, 'Female', NULL, NULL, NULL, '54321', NULL, NULL, NULL, NULL, 100.00;

```

Delete a Patient

```

CREATE PROC delete_PATIENT (
    @PATIENT_ID INT
)
AS
BEGIN
    SET NOCOUNT ON;

    BEGIN TRY
        DELETE FROM PATIENT

```

```

WHERE PATIENT_ID = @PATIENT_ID;

IF @@ROWCOUNT = 0
    PRINT 'No patient found with the given PATIENT_ID!';
ELSE
    PRINT 'Patient deleted successfully!';
END TRY
BEGIN CATCH
    PRINT 'Error occurred while deleting the patient!';
END CATCH
END;
GO

EXEC delete_PATIENT 1;

```

Display Patient Data

```

CREATE PROC display_PATIENT (
    @PATIENT_ID INT = NULL
)
AS
BEGIN
    SET NOCOUNT ON;

    IF @PATIENT_ID IS NULL
        BEGIN
            SELECT * FROM PATIENT;
        END
    ELSE
        BEGIN
            SELECT * FROM PATIENT WHERE PATIENT_ID = @PATIENT_ID;
        END
END;
GO

```

EXEC display_PATIENT; -- This will show all patients

EXEC display_PATIENT 1; -- This will show details for patient with ID 1

Create the PAYMENT Table

```

CREATE PROC create_PAYMENT_TABLE
AS
BEGIN

```

```

SET NOCOUNT ON;
IF NOT EXISTS (SELECT * FROM sysobjects WHERE name='PAYMENT' AND xtype='U')
BEGIN
    CREATE TABLE PAYMENT (
        PAYMENT_ID INT NOT NULL IDENTITY(1,1) CONSTRAINT PK_PAYMENT_ID
PRIMARY KEY,
        PAYMENT_DT DATETIME NOT NULL DEFAULT GETDATE(),
        PAYMENT_AMT DECIMAL(10,2) NOT NULL,
        PATIENT_ID INT NOT NULL,
        FOREIGN KEY (PATIENT_ID) REFERENCES PATIENT(PATIENT_ID),
        PAYMENT_TYPE VARCHAR(11) NOT NULL CONSTRAINT CHK_PAYMENT_TYPE
CHECK (PAYMENT_TYPE IN ('Cash', 'Check', 'Debit_Card', 'Credit_Card', 'cash', 'check',
'debit_card', 'credit_card'))
    );

    PRINT 'PAYMENT table created successfully!';
END
ELSE
BEGIN
    PRINT 'PAYMENT table already exists!';
END
END;
GO

EXEC create_PAYMENT_TABLE;

```

Create a New Payment

```

CREATE PROC add_PAYMENT (
    @PAYMENT_AMT DECIMAL(10,2),
    @PATIENT_ID INT,
    @PAYMENT_TYPE VARCHAR(11) -- Assuming the type is validated through a constraint
)
AS
BEGIN
    SET NOCOUNT ON;

    BEGIN TRY
        INSERT INTO PAYMENT (PAYMENT_AMT, PATIENT_ID, PAYMENT_TYPE)
        VALUES (@PAYMENT_AMT, @PATIENT_ID, @PAYMENT_TYPE);

        PRINT 'Payment added successfully!';
    END TRY
    BEGIN CATCH
        PRINT 'Error occurred while adding the payment!';
    END CATCH

```

```
END CATCH
END;
GO

EXEC add_PAYMENT 150.00, 1, 'Cash';
```

Modify Payment Data

```
CREATE PROC modify_PAYMENT (
    @PAYMENT_ID INT,
    @PAYMENT_AMT DECIMAL(10,2) = NULL,
    @PATIENT_ID INT = NULL,
    @PAYMENT_TYPE VARCHAR(11) = NULL
)
AS
BEGIN
    SET NOCOUNT ON;

    BEGIN TRY
        UPDATE PAYMENT
        SET
            PAYMENT_AMT = COALESCE(@PAYMENT_AMT, PAYMENT_AMT),
            PATIENT_ID = COALESCE(@PATIENT_ID, PATIENT_ID),
            PAYMENT_TYPE = COALESCE(@PAYMENT_TYPE, PAYMENT_TYPE)
        WHERE PAYMENT_ID = @PAYMENT_ID;

        IF @@ROWCOUNT = 0
            PRINT 'No payment found with the given PAYMENT_ID!';
        ELSE
            PRINT 'Payment updated successfully!';
    END TRY
    BEGIN CATCH
        PRINT 'Error occurred while updating the payment!';
    END CATCH
END;
GO

EXEC modify_PAYMENT 1, 200.00, NULL, 'Credit_Card';
```

Delete a Payment

```
CREATE PROC delete_PAYMENT (
    @PAYMENT_ID INT
```

```

)
AS
BEGIN
    SET NOCOUNT ON;

    BEGIN TRY
        DELETE FROM PAYMENT
        WHERE PAYMENT_ID = @PAYMENT_ID;

        IF @@ROWCOUNT = 0
            PRINT 'No payment found with the given PAYMENT_ID!';
        ELSE
            PRINT 'Payment deleted successfully!';
    END TRY
    BEGIN CATCH
        PRINT 'Error occurred while deleting the payment!';
    END CATCH
END;
GO

```

```
EXEC delete_PAYMENT 1;
```

Display Payment Data

```

CREATE PROC display_PAYMENT (
    @PAYMENT_ID INT = NULL
)
AS
BEGIN
    SET NOCOUNT ON;

    IF @PAYMENT_ID IS NULL
        BEGIN
            SELECT * FROM PAYMENT;
        END
    ELSE
        BEGIN
            SELECT * FROM PAYMENT WHERE PAYMENT_ID = @PAYMENT_ID;
        END
END;
GO

```

```

EXEC display_PAYMENT; -- Display all payments
EXEC display_PAYMENT 1; -- Display specific payment with ID 1

```

Creating the VISIT table

```
CREATE PROC create_VISIT_TABLE
AS
BEGIN
    SET NOCOUNT ON;

    IF NOT EXISTS (SELECT * FROM sysobjects WHERE name='VISIT' AND xtype='U')
    BEGIN
        CREATE TABLE VISIT (
            VISIT_ID INT NOT NULL IDENTITY(1,1) CONSTRAINT PK_VISIT_ID PRIMARY KEY,
            VISIT_DT DATETIME NOT NULL DEFAULT GETDATE(),
            CHARGED_AMT DECIMAL(8,2) NOT NULL DEFAULT 0,
            PATIENT_ID INT NOT NULL,
                FOREIGN KEY (PATIENT_ID) REFERENCES PATIENT(PATIENT_ID),
            PHYSICIAN_ID INT NOT NULL,
                FOREIGN KEY (PHYSICIAN_ID) REFERENCES PHYSICIAN(PHYSICIAN_ID),
            ALLERGIES_1 VARCHAR(30),
            ALLERGIES_2 VARCHAR(30),
            ALLERGIES_3 VARCHAR(30),
            ALLERGIES_4 VARCHAR(30),
            ALLERGIES_5 VARCHAR(30),
            VACCINES_1 VARCHAR(30),
            VACCINES_2 VARCHAR(30),
            VACCINES_3 VARCHAR(30),
            VACCINES_4 VARCHAR(30),
            VACCINES_5 VARCHAR(30),
            NOTES_1 VARCHAR(5000),
            DIAGNOSES_1 VARCHAR(200),
            DIAGNOSES_2 VARCHAR(200),
            DIAGNOSES_3 VARCHAR(200),
            DIAGNOSES_4 VARCHAR(200),
            DIAGNOSES_5 VARCHAR(200)
        );

        PRINT 'VISIT table created successfully!';
    END
    ELSE
    BEGIN
        PRINT 'VISIT table already exists!';
    END
END;
GO
```

```
EXEC create_VISIT_TABLE;
```

Add a VISIT

```
CREATE PROC add_VISIT
  @VISIT_DT DATETIME,
  @CHARGED_AMT DECIMAL(8,2),
  @PATIENT_ID INT,
  @PHYSICIAN_ID INT,
  @ALLERGIES_1 VARCHAR(30) = NULL,
  @ALLERGIES_2 VARCHAR(30) = NULL,
  @ALLERGIES_3 VARCHAR(30) = NULL,
  @ALLERGIES_4 VARCHAR(30) = NULL,
  @ALLERGIES_5 VARCHAR(30) = NULL,
  @VACCINES_1 VARCHAR(30) = NULL,
  @VACCINES_2 VARCHAR(30) = NULL,
  @VACCINES_3 VARCHAR(30) = NULL,
  @VACCINES_4 VARCHAR(30) = NULL,
  @VACCINES_5 VARCHAR(30) = NULL,
  @NOTES_1 VARCHAR(5000) = NULL,
  @DIAGNOSES_1 VARCHAR(200) = NULL,
  @DIAGNOSES_2 VARCHAR(200) = NULL,
  @DIAGNOSES_3 VARCHAR(200) = NULL,
  @DIAGNOSES_4 VARCHAR(200) = NULL,
  @DIAGNOSES_5 VARCHAR(200) = NULL
AS
BEGIN
  SET NOCOUNT ON;

  INSERT INTO VISIT (
    VISIT_DT, CHARGED_AMT, PATIENT_ID, PHYSICIAN_ID,
    ALLERGIES_1, ALLERGIES_2, ALLERGIES_3, ALLERGIES_4, ALLERGIES_5,
    VACCINES_1, VACCINES_2, VACCINES_3, VACCINES_4, VACCINES_5,
    NOTES_1, DIAGNOSES_1, DIAGNOSES_2, DIAGNOSES_3, DIAGNOSES_4,
    DIAGNOSES_5
  )
  VALUES (
    @VISIT_DT, @CHARGED_AMT, @PATIENT_ID, @PHYSICIAN_ID,
    @ALLERGIES_1, @ALLERGIES_2, @ALLERGIES_3, @ALLERGIES_4,
    @ALLERGIES_5,
    @VACCINES_1, @VACCINES_2, @VACCINES_3, @VACCINES_4, @VACCINES_5,
    @NOTES_1, @DIAGNOSES_1, @DIAGNOSES_2, @DIAGNOSES_3,
    @DIAGNOSES_4, @DIAGNOSES_5
  );
```

```
PRINT 'VISIT added successfully!';
END;
GO
```

```
EXEC add_VISIT '2024-02-10', 200.00, 1, 2, 'Peanuts', 'Dust', NULL, NULL, NULL, 'Hepatitis B',
'Polio', NULL, NULL, NULL, 'Notes here', 'Diabetes', NULL, NULL, NULL, NULL;
```

Modify a VISIT

```
CREATE PROC modify_VISIT
    @VISIT_ID INT,
    @VISIT_DT DATETIME = NULL,
    @CHARGED_AMT DECIMAL(8,2) = NULL,
    @PATIENT_ID INT = NULL,
    @PHYSICIAN_ID INT = NULL,
    @ALLERGIES_1 VARCHAR(30) = NULL,
    @ALLERGIES_2 VARCHAR(30) = NULL,
    @ALLERGIES_3 VARCHAR(30) = NULL,
    @ALLERGIES_4 VARCHAR(30) = NULL,
    @ALLERGIES_5 VARCHAR(30) = NULL,
    @VACCINES_1 VARCHAR(30) = NULL,
    @VACCINES_2 VARCHAR(30) = NULL,
    @VACCINES_3 VARCHAR(30) = NULL,
    @VACCINES_4 VARCHAR(30) = NULL,
    @VACCINES_5 VARCHAR(30) = NULL,
    @NOTES_1 VARCHAR(5000) = NULL,
    @DIAGNOSES_1 VARCHAR(200) = NULL,
    @DIAGNOSES_2 VARCHAR(200) = NULL,
    @DIAGNOSES_3 VARCHAR(200) = NULL,
    @DIAGNOSES_4 VARCHAR(200) = NULL,
    @DIAGNOSES_5 VARCHAR(200) = NULL
AS
BEGIN
    SET NOCOUNT ON;

    UPDATE VISIT
    SET
        VISIT_DT = COALESCE(@VISIT_DT, VISIT_DT),
        CHARGED_AMT = COALESCE(@CHARGED_AMT, CHARGED_AMT),
        PATIENT_ID = COALESCE(@PATIENT_ID, PATIENT_ID),
        PHYSICIAN_ID = COALESCE(@PHYSICIAN_ID, PHYSICIAN_ID),
        ALLERGIES_1 = COALESCE(@ALLERGIES_1, ALLERGIES_1),
        ALLERGIES_2 = COALESCE(@ALLERGIES_2, ALLERGIES_2),
```



```

ALLERGIES_3 = COALESCE(@ALLERGIES_3, ALLERGIES_3),
ALLERGIES_4 = COALESCE(@ALLERGIES_4, ALLERGIES_4),
ALLERGIES_5 = COALESCE(@ALLERGIES_5, ALLERGIES_5),
VACCINES_1 = COALESCE(@VACCINES_1, VACCINES_1),
VACCINES_2 = COALESCE(@VACCINES_2, VACCINES_2),
VACCINES_3 = COALESCE(@VACCINES_3, VACCINES_3),
VACCINES_4 = COALESCE(@VACCINES_4, VACCINES_4),
VACCINES_5 = COALESCE(@VACCINES_5, VACCINES_5),
NOTES_1 = COALESCE(@NOTES_1, NOTES_1),
DIAGNOSES_1 = COALESCE(@DIAGNOSES_1, DIAGNOSES_1),
DIAGNOSES_2 = COALESCE(@DIAGNOSES_2, DIAGNOSES_2),
DIAGNOSES_3 = COALESCE(@DIAGNOSES_3, DIAGNOSES_3),
DIAGNOSES_4 = COALESCE(@DIAGNOSES_4, DIAGNOSES_4),
DIAGNOSES_5 = COALESCE(@DIAGNOSES_5, DIAGNOSES_5)
WHERE VISIT_ID = @VISIT_ID;

```

```

PRINT 'VISIT modified successfully!';
END;
GO

```

```

EXEC modify_VISIT 1, NULL, 250.00, NULL, NULL, 'Nuts', NULL, NULL, NULL, NULL, 'Flu
Shot', NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL;

```

Delete a VISIT

```

CREATE PROC delete_VISIT
    @VISIT_ID INT
AS
BEGIN
    SET NOCOUNT ON;

    DELETE FROM VISIT
    WHERE VISIT_ID = @VISIT_ID;

    PRINT 'VISIT deleted successfully!';
END;
GO

```

```

EXEC delete_VISIT 1;

```

Display VISITs

```

CREATE PROC display_VISIT
AS

```

```

BEGIN
  SET NOCOUNT ON;

  SELECT
    V.VISIT_ID, V.VISIT_DT, V.CHARGED_AMT,
    P.FNAME AS Patient_FirstName, P.LNAME AS Patient_LastName,
    PH.FNAME AS Physician_FirstName, PH.LNAME AS Physician_LastName,
    V.ALLERGIES_1, V.ALLERGIES_2, V.ALLERGIES_3, V.ALLERGIES_4, V.ALLERGIES_5,
    V.VACCINES_1, V.VACCINES_2, V.VACCINES_3, V.VACCINES_4, V.VACCINES_5,
    V.NOTES_1, V.DIAGNOSES_1, V.DIAGNOSES_2, V.DIAGNOSES_3, V.DIAGNOSES_4,
    V.DIAGNOSES_5
  FROM VISIT V
  INNER JOIN PATIENT P ON V.PATIENT_ID = P.PATIENT_ID
  INNER JOIN PHYSICIAN PH ON V.PHYSICIAN_ID = PH.PHYSICIAN_ID;

  PRINT 'VISIT records displayed successfully!';
END;
GO

EXEC display_VISIT;

```

Display all the tables in the FINALPROJECT database

```

CREATE PROC display_all_tables
AS
BEGIN
  SET NOCOUNT ON;

  SELECT *
    FROM PATIENT
  SELECT *
  FROM PAYMENT
  SELECT *
  FROM PHYSICIAN
  SELECT *
  FROM VISIT

  PRINT 'All tables in the FINALPROJECT database displayed successfully!';
END;
GO

EXEC display_all_tables;

```

Display summary of visits by diagnosis

```
CREATE PROC display_summary_VISIT
@diagnoses varchar(500)
AS
BEGIN
    SET NOCOUNT ON;

    SELECT *
    FROM VISIT
    WHERE DIAGNOSES_1 = @diagnoses
        OR DIAGNOSES_2 = @diagnoses
        OR DIAGNOSES_3 = @diagnoses
        OR DIAGNOSES_4 = @diagnoses
        OR DIAGNOSES_5 = @diagnoses;

    PRINT 'VISIT records displayed successfully!';
END;
GO

EXEC display_summary_VISIT 'stress_disorder';
```

Create Trigger for Updating Balance on VISIT new row

```
CREATE TRIGGER trg_update_balance_on_visit
ON VISIT
AFTER INSERT
AS
BEGIN
    UPDATE PATIENT
    SET BALANCE = BALANCE + i.CHARGED_AMT
    FROM INSERTED i
    WHERE PATIENT.PATIENT_ID = i.PATIENT_ID;

    PRINT 'Patient balance updated after visit!';
END;
GO
```

Create Trigger for Updating Balance on PAYMENT new row

```
CREATE TRIGGER trg_update_balance_on_payment
ON PAYMENT
```

```

AFTER INSERT
AS
BEGIN
    UPDATE PATIENT
    SET BALANCE = BALANCE - i.PAYMENT_AMT
    FROM INSERTED i
    WHERE PATIENT.PATIENT_ID = i.PATIENT_ID;

    PRINT 'Patient balance updated after payment!';
END;
GO

```

Use Crystal Reports to show the VISIT Information report

Visit summary report

SAP Crystal Reports - [Report1]

File SAP Edit View Insert Format Database Report Window Help

Start Page Report1 x

Design Preview x

Groups

Report1

RF-
PF-

10/10/2024

<u>VISIT_ID</u>	<u>VISIT_DT</u>	<u>COST</u>	<u>PATIENTID</u>	<u>PHYSICIANID</u>	<u>NOTES_1</u>	<u>DIAGNOSES_1</u>
2	3/4/2024 3:45:54PM	6,000.00	2	3	broken shoulder bone	broken shoulder
3	4/2/2024 4:00:00PM	100,000.00	1	1	Heart transplant success	Heart Transplant
4	2/10/2024 12:00:00A	200.00	1	2	Notes here	Diabetes
5	2/10/2024 12:00:00A	200.00	1	2	Notes here	Diabetes

RF

Advanced Trigger Track Salary

```
CREATE TRIGGER tr_UpdatePhysicianSalary
ON dbo.PHYSICIAN
FOR UPDATE
AS
BEGIN
    DECLARE @OldSalary DECIMAL(10, 2);
    DECLARE @NewSalary DECIMAL(10, 2);
    DECLARE @PhysicianID INT;
    DECLARE @ChangedDate DATETIME;

    SELECT @OldSalary = d.SALARY, @PhysicianID = d.PHYSICIAN_ID
    FROM DELETED d;

    SELECT @NewSalary = i.SALARY
    FROM INSERTED i;

    IF @OldSalary <> @NewSalary
    BEGIN
        SET @ChangedDate = GETDATE();

        INSERT INTO PHYSICIAN_SALARY_LOG (PHYSICIAN_ID, OLD_SALARY,
NEW_SALARY, CHANGED_DATE)
        VALUES (@PhysicianID, @OldSalary, @NewSalary, @ChangedDate);
    END
END;
```

Log table for Advanced Trigger Track Salary

```
CREATE TABLE PHYSICIAN_SALARY_LOG (
    LOG_ID INT IDENTITY(1,1) PRIMARY KEY,
    PHYSICIAN_ID INT,
    OLD_SALARY DECIMAL(10, 2),
    NEW_SALARY DECIMAL(10, 2),
    CHANGED_DATE DATETIME
);
```