Pregunta 1 (2 p.)

• Cree las sentencias que permitan la creación del modelo de datos presentados, incluyendo los tipos de datos, las llaves primarias y las llaves foráneas.

CREATE TABLE Asociado (

codigo\_asociado int NOT NULL,

apellido\_paterno varchar(30) NOT NULL,

apellido\_materno varchar(30) NOT NULL,

nombre varchar(30) NOT NULL,

tipo\_asociado char(1) NOT NULL,

numeroDocumento varchar(11) NOT NULL,

fechaUltimaCap date NOT NULL,

telefonoFijo varchar(7) NOT NULL,

telefonoTrabajo varchar(7) NOT NULL,

telefonoMovil varchar(9) NOT NULL,

Distrito\_codigo\_distrito int NOT NULL,

Especialidad\_codigo\_especialidad int NOT NULL,

Centro\_Ocupacional\_codigo\_co int NOT NULL,

CONSTRAINT Asociado\_pk PRIMARY KEY (codigo\_asociado)

);

-- Table: Centro\_Ocupacional

CREATE TABLE Centro\_Ocupacional (

codigo\_co int NOT NULL,

razon\_social varchar(30) NOT NULL,

telefonoFijo varchar(7) NOT NULL,

Distrito\_codigo\_distrito int NOT NULL,

Giro\_codigo\_giro int NOT NULL,

CONSTRAINT Centro\_Ocupacional\_pk PRIMARY KEY (codigo\_co)

);

-- Table: Curso

CREATE TABLE Curso (

codigo\_curso int NOT NULL,

descripcion\_curso varchar(30) NOT NULL,

nro\_HorasT int NOT NULL,

nro\_HorasP int NOT NULL,

nivelComplejidad char(1) NOT NULL,

CONSTRAINT Curso\_pk PRIMARY KEY (codigo\_curso)

);

-- Table: Distrito

CREATE TABLE Distrito (

codigo\_distrito int NOT NULL,

nombre\_distrito varchar(30) NOT NULL,

CONSTRAINT Distrito\_pk PRIMARY KEY (codigo\_distrito)

);

-- Table: Especialidad

CREATE TABLE Especialidad (

codigo\_especialidad int NOT NULL,

descripcion\_especialidad varchar(30) NOT NULL,

CONSTRAINT Especialidad\_pk PRIMARY KEY (codigo\_especialidad)

);

-- Table: Evento

CREATE TABLE Evento (

numero\_evento int NOT NULL,

fecha\_inicio date NOT NULL,

fecha\_final date NOT NULL,

Curso\_codigo\_curso int NOT NULL,

Centro\_Ocupacional\_codigo\_co int NOT NULL,

Expositor\_codigo\_expositor int NOT NULL,

frecuencia char(1) NOT NULL,

turno char(1) NOT NULL,

presupuesto decimal(10,2) NOT NULL,

CONSTRAINT Evento\_pk PRIMARY KEY (numero\_evento)

);

-- Table: Expositor

CREATE TABLE Expositor (

codigo\_expositor int NOT NULL,

apellido\_paterno varchar(30) NOT NULL,

apellido\_materno varchar(30) NOT NULL,

nombres varchar(30) NOT NULL,

telefonoMovil varchar(9) NOT NULL,

eMail varchar(80) NOT NULL,

Distrito\_codigo\_distrito int NOT NULL,

Centro\_Ocupacional\_codigo\_co int NOT NULL,

CONSTRAINT Expositor\_pk PRIMARY KEY (codigo\_expositor)

);

-- Table: Giro

CREATE TABLE Giro (

codigo\_giro int NOT NULL,

nombre\_giro varchar(30) NOT NULL,

CONSTRAINT Giro\_pk PRIMARY KEY (codigo\_giro)

);

-- Table: Inscripcion

CREATE TABLE Inscripcion (

Evento\_numero\_evento int NOT NULL,

Asociado\_codigo\_asociado int NOT NULL,

asistido bit NOT NULL,

CONSTRAINT Inscripcion\_pk PRIMARY KEY (Evento\_numero\_evento,Asociado\_codigo\_asociado)

);

-- foreign keys

-- Reference: Asociado\_Centro\_Ocupacional (table: Asociado)

ALTER TABLE Asociado ADD CONSTRAINT Asociado\_Centro\_Ocupacional

FOREIGN KEY (Centro\_Ocupacional\_codigo\_co)

REFERENCES Centro\_Ocupacional (codigo\_co);

-- Reference: Asociado\_Distrito (table: Asociado)

ALTER TABLE Asociado ADD CONSTRAINT Asociado\_Distrito

FOREIGN KEY (Distrito\_codigo\_distrito)

REFERENCES Distrito (codigo\_distrito);

-- Reference: Asociado\_Especialidad (table: Asociado)

ALTER TABLE Asociado ADD CONSTRAINT Asociado\_Especialidad

FOREIGN KEY (Especialidad\_codigo\_especialidad)

REFERENCES Especialidad (codigo\_especialidad);

-- Reference: Centro\_Ocupacional\_Distrito (table: Centro\_Ocupacional)

ALTER TABLE Centro\_Ocupacional ADD CONSTRAINT Centro\_Ocupacional\_Distrito

FOREIGN KEY (Distrito\_codigo\_distrito)

REFERENCES Distrito (codigo\_distrito);

-- Reference: Centro\_Ocupacional\_Giro (table: Centro\_Ocupacional)

ALTER TABLE Centro\_Ocupacional ADD CONSTRAINT Centro\_Ocupacional\_Giro

FOREIGN KEY (Giro\_codigo\_giro)

REFERENCES Giro (codigo\_giro);

-- Reference: Evento\_Centro\_Ocupacional (table: Evento)

ALTER TABLE Evento ADD CONSTRAINT Evento\_Centro\_Ocupacional

FOREIGN KEY (Centro\_Ocupacional\_codigo\_co)

REFERENCES Centro\_Ocupacional (codigo\_co);

-- Reference: Evento\_Curso (table: Evento)

ALTER TABLE Evento ADD CONSTRAINT Evento\_Curso

FOREIGN KEY (Curso\_codigo\_curso)

REFERENCES Curso (codigo\_curso);

-- Reference: Evento\_Expositor (table: Evento)

ALTER TABLE Evento ADD CONSTRAINT Evento\_Expositor

FOREIGN KEY (Expositor\_codigo\_expositor)

REFERENCES Expositor (codigo\_expositor);

-- Reference: Expositor\_Centro\_Ocupacional (table: Expositor)

ALTER TABLE Expositor ADD CONSTRAINT Expositor\_Centro\_Ocupacional

FOREIGN KEY (Centro\_Ocupacional\_codigo\_co)

REFERENCES Centro\_Ocupacional (codigo\_co);

-- Reference: Expositor\_Distrito (table: Expositor)

ALTER TABLE Expositor ADD CONSTRAINT Expositor\_Distrito

FOREIGN KEY (Distrito\_codigo\_distrito)

REFERENCES Distrito (codigo\_distrito);

-- Reference: Inscripcion\_Asociado (table: Inscripcion)

ALTER TABLE Inscripcion ADD CONSTRAINT Inscripcion\_Asociado

FOREIGN KEY (Asociado\_codigo\_asociado)

REFERENCES Asociado (codigo\_asociado);

-- Reference: Inscripcion\_Evento (table: Inscripcion)

ALTER TABLE Inscripcion ADD CONSTRAINT Inscripcion\_Evento

FOREIGN KEY (Evento\_numero\_evento)

REFERENCES Evento (numero\_evento);

Pregunta 2 (2 p.)

• Cree las sentencias que permita insertar por lo menos cinco (5) registros en dos (2) de las tablas creadas en la pregunta 1.

INSERT INTO Curso (codigo\_curso, descripcion\_curso, nro\_HorasT, nro\_HorasP, nivelComplejidad)

VALUES

(1, 'Curso Diseño de Paginas Web', 20, 10, 'B'),

(2, 'Curso Base de Datos', 15, 5, 'A'),

(3, 'Curso Cocina', 30, 20, 'C'),

(4, 'Curso Derecho Empresarial', 25, 15, 'B'),

(5, 'Curso Electricidad', 10, 8, 'A');

INSERT INTO Distrito (codigo\_distrito, nombre\_distrito)

VALUES

(1, 'Distrito VMT'),

(2, 'Distrito SJL'),

(3, 'Distrito VES'),

(4, 'Distrito ATE'),

(5, 'Distrito ANCON');

Tabla

Descripción generada automáticamente

**Pregunta 3 (2 p.)**

 Diseñar la función o procedimiento almacenado que permita determinar la cantidad de asociados por tipos de especialidad para un determinado centro ocupacional.

CREATE FUNCTION ContarAsociadosPorEspecialidad (

@codigo\_co INT

)

RETURNS TABLE

AS

RETURN (

SELECT E.descripcion\_especialidad, COUNT(A.codigo\_asociado) AS cantidad\_asociados

FROM Asociado A

INNER JOIN Especialidad E ON A.Especialidad\_codigo\_especialidad = E.codigo\_especialidad

WHERE A.Centro\_Ocupacional\_codigo\_co = @codigo\_co

GROUP BY E.descripcion\_especialidad

);

SELECT \*

FROM dbo.ContarAsociadosPorEspecialidad(1)

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Pregunta 4 (2 p.)**

Diseñar el procedimiento almacenado que permita imprimir la cantidad de eventos y el monto total de presupuesto asignado para un rango de fechas determinado.

CREATE PROCEDURE ImprimirEventosPorFecha (

@fecha\_inicio DATE,

@fecha\_final DATE

)

AS

BEGIN

SELECT COUNT(\*) AS cantidad\_eventos, SUM(presupuesto) AS total\_presupuesto

FROM Evento

WHERE fecha\_inicio >= @fecha\_inicio AND fecha\_final <= @fecha\_final;

END;

EXEC ImprimirEventosPorFecha '2023-01-01', '2023-12-31'

Tabla

Descripción generada automáticamente

**Pregunta 5 (4 p.)**

 Diseñar la función o procedimiento almacenado que permita determinar el o (los) asociado(s) que no asistieron a ningún evento que se dio en un rango de fechas determinado.

CREATE PROCEDURE BuscarAsociadosSinAsistencia (

@fecha\_inicio DATE,

@fecha\_final DATE

)

AS

BEGIN

SELECT A.codigo\_asociado, A.apellido\_paterno, A.apellido\_materno, A.nombre

FROM Asociado A

WHERE NOT EXISTS (

SELECT 1

FROM Inscripcion I

INNER JOIN Evento E ON I.Evento\_numero\_evento = E.numero\_evento

WHERE I.Asociado\_codigo\_asociado = A.codigo\_asociado

AND E.fecha\_inicio >= @fecha\_inicio AND E.fecha\_final <= @fecha\_final

);

END;

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Pregunta 6 (4 p.)**

 Diseñar la función o procedimiento almacenado que permita determinar el(los) expositor(es) que más eventos han realizado en un rango de fechas determinado.

CREATE PROCEDURE BuscarExpositoresMasActivos (

@fecha\_inicio DATE,

@fecha\_final DATE

)

AS

BEGIN

SELECT TOP 1 WITH TIES E.codigo\_expositor, E.apellido\_paterno, E.apellido\_materno, E.nombres, COUNT(\*) AS cantidad\_eventos

FROM Evento EV

INNER JOIN Expositor E ON EV.Expositor\_codigo\_expositor = E.codigo\_expositor

WHERE EV.fecha\_inicio >= @fecha\_inicio AND EV.fecha\_final <= @fecha\_final

GROUP BY E.codigo\_expositor, E.apellido\_paterno, E.apellido\_materno, E.nombres

ORDER BY COUNT(\*) DESC;

END;

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Pregunta 7 (4 p.)**

 Diseñar la función o procedimiento almacenado que permita determinar el(los) curso(s) con la mayor cantidad de eventos en un rango de fechas determinado.

CREATE PROCEDURE BuscarCursosMasPopulares (

@fecha\_inicio DATE,

@fecha\_final DATE

)

AS

BEGIN

SELECT TOP 1 WITH TIES C.codigo\_curso, C.descripcion\_curso, COUNT(\*) AS cantidad\_eventos

FROM Evento E

INNER JOIN Curso C ON E.Curso\_codigo\_curso = C.codigo\_curso

WHERE E.fecha\_inicio >= @fecha\_inicio AND E.fecha\_final <= @fecha\_final

GROUP BY C.codigo\_curso, C.descripcion\_curso

ORDER BY COUNT(\*) DESC;

END;

EXEC BuscarCursosMasPopulares '2021-06-07', '2023-08-05';

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente