



**ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS
INGENIERÍA DE SISTEMAS INFORMÁTICOS Y DE COMPUTACIÓN**

Implementación de Esquemas Locales y extracción de información

Grupo No.3

Integrantes: Bravo Francis, Freire Ismael, Vasco Alexis

Tema: Implementación de Esquemas Locales y extracción de información (P)

Objetivos:

- Diseñar los esquemas de fragmentación horizontal y mixta basándose en criterios institucionales para asegurar la integridad y la no redundancia de los datos.
- Establecer el esquema de ubicación y asignación de fragmentos en los nodos correspondientes para reducir la latencia y el tráfico de red.
- Desarrollar los esquemas lógicos locales y los procesos de migración desde la base centralizada para garantizar la autonomía operativa de cada sede.
- Representar la arquitectura del sistema a través de diagramas relacionales globales y locales para validar la correcta reconstrucción y consistencia de la base de datos.

Marco teórico:

1. Esquema de Fragmentación

La fragmentación consiste en descomponer una relación global en unidades lógicas menores (fragmentos) para optimizar el acceso a los datos. Se clasifica en horizontal (selección de filas por predicados), vertical (proyección de columnas) o mixta. Para ser válida, toda fragmentación debe cumplir con tres reglas de correctitud: completitud (no perder datos), reconstrucción (posibilidad de recuperar la tabla original) y disyunción (evitar duplicidad de datos no planificada).

2. Esquema de Ubicación y Asignación

Este esquema define la correspondencia física entre los fragmentos lógicos y los nodos de la red. Su objetivo es minimizar los costos de comunicación situando los datos cerca de donde se consultan. La asignación puede ser simple (un fragmento en un solo nodo) o replicada (copias en múltiples nodos), buscando un equilibrio entre la alta disponibilidad y la complejidad de mantener la consistencia de la información.

3. Esquemas Locales y Extracción de Datos

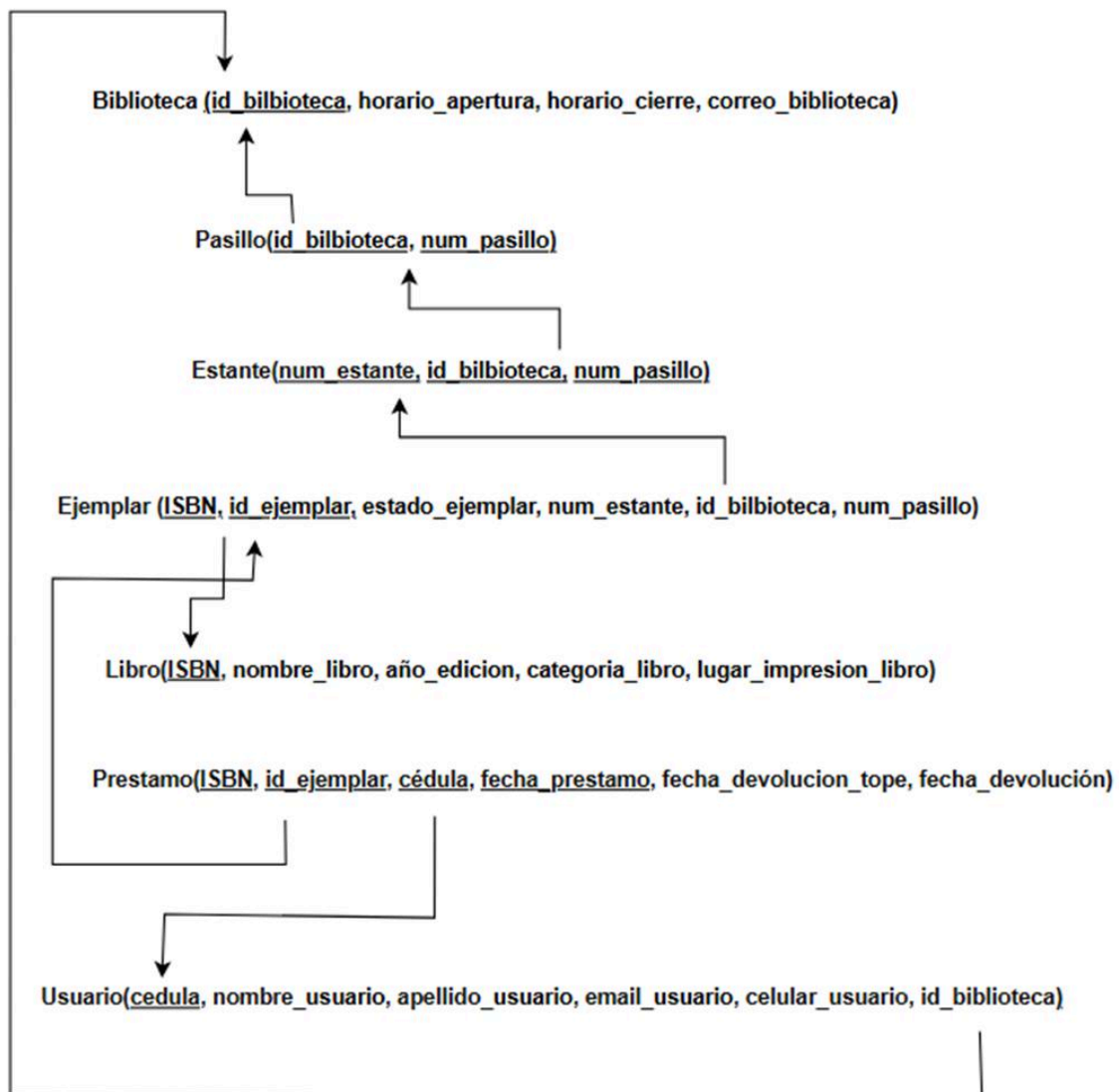
La implementación física requiere traducir el diseño global en Esquemas Locales, que son las estructuras de tablas (DDL) que residen efectivamente en cada nodo. Una vez creadas, se procede a la extracción y carga de datos, proceso técnico que migra la información desde una base centralizada hacia los nodos distribuidos. Esta fase garantiza que los criterios de fragmentación se apliquen correctamente y que se mantenga la integridad referencial en cada sitio.

4. Modelado Gráfico (Diagramas)

El diseño distribuido se visualiza mediante dos niveles de abstracción: el Diagrama Global, que muestra la base de datos como una entidad única y centralizada, y los Diagramas Locales, que representan la estructura fragmentada en cada nodo. Estos diagramas permiten verificar la arquitectura del sistema, facilitando la comprensión de cómo se distribuyen las tablas y cómo se relacionan entre sí a través de la red.

Desarrollo de la práctica:

Grafo Relacional



1.- Esquema de fragmentación

En este caso vamos a trabajar con dos sedes (facultades) = {01 => FIS, 02 => FIQA}

1. Fragmentación horizontal primaria

Campo de fragmentación: id_biblioteca

Condición de fragmentación: id_biblioteca = {01, 02}

- BIBLIOTECA (Por Facultad):

$$Biblioteca_i = \sigma_{id_biblioteca=i}(BIBLIOTECA)$$

- PASILLO (Por Biblioteca/Ubicación):

$$Pasillo_i = \sigma_{id_biblioteca=i}(PASILLO)$$

- ESTANTE (Por Biblioteca/Ubicación):

$$Estante_i = \sigma_{id_biblioteca=i}(ESTANTE)$$

- EJEMPLAR (Por Biblioteca):

$$Ejemplar_i = \sigma_{id_biblioteca=i}(EJEMPLAR)$$

2. Fragmentación Horizontal Derivada

- PRESTAMO (Deriva de Ejemplar):

$$Prestamo_i = PRESTAMO \bowtie Ejemplar_i$$

3. Fragmentación Mixta

Estrategia híbrida para aislar datos sensibles y ubicarlos por facultad.

- Paso A: Fragmentación Vertical (FV)

$$Usuario_info = \pi_{cédula, nombre, apellido, id_biblioteca}(Usuario)$$

$$Usuario_contacto = \pi_{cédula, email, celular}(Usuario)$$

- Paso B: Fragmentación Horizontal Primaria (FHP)

Campo de fragmentación = id_biblioteca

Condición de fragmentación = id_biblioteca = i ; i = {01, 02}

$$Usuarios_info_i = \sigma_{id_biblioteca = i}(Usuarios_info)$$

2.- Esquema de ubicación

Definición del esquema de replicación

Tabla: Libro

1.- Se aplica replicación total para garantizar la alta disponibilidad del catálogo bibliográfico en todas las sedes, permitiendo consultas de búsqueda inmediatas y eliminando la latencia de red al verificar la existencia de títulos.

2.- **Nodos de replicación:** 01, 02

2.- **Unidireccional**

3.- **Nodo de gestión:** FIS

Esquema de Ubicación

Entidad	Nodo 1	Nodo 2
Libro	Libro	Libro
Usuario	Usuario_contacto Usuario_info_01	Usuario_info_02
Biblioteca	Biblioteca_01	Biblioteca_02
Pasillo	Pasillo_01	Pasillo_02
Estante	Estante_01	Estante_02
Ejemplar	Ejemplar_01	Ejemplar_02
Prestamo	Prestamo_01	Prestamo_02

3.- Creación de los esquemas locales (tablas) y extracción de datos de la base centralizada a los dos nodos de la base distribuida

Creación de la BD Centralizada:

```

/* * Script de Creación - BD Centralizada Gestión Bibliotecaria
 * Autores: Alexis Vasco, Ismael Freire, Francis Bravo
 */

USE master;
GO

-- 1. Crear la Base de Datos
IF NOT EXISTS (SELECT name FROM sys.databases WHERE name = 'BibliotecaUCentral')
BEGIN
    CREATE DATABASE BibliotecaUCentral;
END
GO

USE BibliotecaUCentral;
GO

-- =====
-- 1. Tablas (Entidades Fuertes)
-- =====

-- Tabla: Biblioteca
CREATE TABLE Biblioteca (
    id_biblioteca CHAR(2) NOT NULL,
    horario_apertura TIME NOT NULL,
    horario_cierre TIME NOT NULL,
    correo_biblioteca VARCHAR(100),
    CONSTRAINT PK_Biblioteca PRIMARY KEY (id_biblioteca)
);
GO

-- Tabla: Libro (Catálogo)
CREATE TABLE Libro (
    ISBN VARCHAR(20) NOT NULL,
    nombre_libro VARCHAR(150) NOT NULL,
    anio_edicion INT,
    categoria_libro VARCHAR(50),
    lugar_impresion_libro VARCHAR(100),
    CONSTRAINT PK_Libro PRIMARY KEY (ISBN)
);
GO

-- =====
-- 2. Tablas de Infraestructura
-- =====

-- Tabla: Pasillo
-- Dependencia: Biblioteca -> Pasillo
CREATE TABLE Pasillo (
    id_biblioteca CHAR(2) NOT NULL,
    num_pasillo INT NOT NULL,
    CONSTRAINT PK_Pasillo PRIMARY KEY (id_biblioteca, num_pasillo),
    CONSTRAINT FK_Pasillo_Biblioteca FOREIGN KEY (id_biblioteca)
        REFERENCES Biblioteca(id_biblioteca)
);
GO

```

```

-- Tabla: Estante
-- Dependencia: Pasillo -> Estante
CREATE TABLE Estante (
    id_biblioteca CHAR(2) NOT NULL,
    num_pasillo INT NOT NULL,
    num_estante INT NOT NULL,

    CONSTRAINT PK_Estante PRIMARY KEY (id_biblioteca, num_pasillo, num_estante),
    CONSTRAINT FK_Estante_Pasillo FOREIGN KEY (id_biblioteca, num_pasillo)
        REFERENCES Pasillo(id_biblioteca, num_pasillo)
);
GO

```

```

-- =====
-- 3. Tablas de Inventario y Usuarios
-- =====
-- Tabla: Ejemplar
-- Dependencia: Estante -> Ejemplar y Libro
CREATE TABLE Ejemplar (
    ISBN VARCHAR(20) NOT NULL,
    id_ejemplar INT NOT NULL,
    estado_ejemplar VARCHAR(20) DEFAULT 'Disponible',

    id_biblioteca CHAR(2) NOT NULL,
    num_pasillo INT NOT NULL,
    num_estante INT NOT NULL,

    CONSTRAINT PK_Ejemplar PRIMARY KEY (ISBN, id_ejemplar),
    -- Relación con el Catálogo
    CONSTRAINT FK_Ejemplar_Libro FOREIGN KEY (ISBN)
        REFERENCES Libro(ISBN),
    -- Relación con la ubicación física
    CONSTRAINT FK_Ejemplar_Estante FOREIGN KEY (id_biblioteca, num_pasillo, num_estante)
        REFERENCES Estante(id_biblioteca, num_pasillo, num_estante),

    -- Restricción
    CONSTRAINT CK_Ejemplar_RangoID CHECK (
        (id_biblioteca = '01' AND id_ejemplar BETWEEN 1 AND 199) OR -- Rango Sede 01
        (id_biblioteca = '02' AND id_ejemplar BETWEEN 200 AND 399) -- Rango Sede 02
    ),
);
GO

```

```

-- Tabla: Usuario
-- Dependencia: Biblioteca (Facultad de origen)
CREATE TABLE Usuario (
    cedula VARCHAR(15) NOT NULL,
    nombre_usuario VARCHAR(50) NOT NULL,
    apellido_usuario VARCHAR(50) NOT NULL,
    email_usuario VARCHAR(100),
    celular_usuario VARCHAR(20),
    id_biblioteca CHAR(2) NOT NULL, -- Facultad a la que pertenece

    CONSTRAINT PK_Usuario PRIMARY KEY (cedula),
    CONSTRAINT FK_Usuario_Biblioteca FOREIGN KEY (id_biblioteca)
        REFERENCES Biblioteca(id_biblioteca)
);
GO

```



```

-- =====
-- 4. Tabla Transaccional
-- =====

-- Tabla: Prestamo
CREATE TABLE Prestamo (
    ISBN VARCHAR(20) NOT NULL,
    id_ejemplar INT NOT NULL,
    cedula VARCHAR(15) NOT NULL,
    fecha_prestamo DATETIME DEFAULT GETDATE(),
    fecha_devolucion_tope DATETIME NOT NULL,
    fecha_devolucion DATETIME NULL,

    CONSTRAINT PK_Prestamo PRIMARY KEY (ISBN, id_ejemplar, cedula, fecha_prestamo),

    CONSTRAINT FK_Prestamo_Usuario FOREIGN KEY (cedula)
        REFERENCES Usuario(cedula),

    CONSTRAINT FK_Prestamo_Ejemplar FOREIGN KEY (ISBN, id_ejemplar)
        REFERENCES Ejemplar(ISBN, id_ejemplar)
);
GO

```

Población de las tablas:

```

/* * Script de Creación - BD Centralizada Gestión Bibliotecaria
 * Autores: Alexis Vasco, Ismael Freire, Francis Bravo
 */

USE BibliotecaUCentral;
GO

-- =====
-- 1. Insertar Bibliotecas (Sedes)
-- =====
INSERT INTO Biblioteca (id_biblioteca, horario_apertura, horario_cierre, correo_biblioteca) VALUES
('01', '08:00', '20:00', 'biblioteca.fis@epn.edu.ec'), -- Sistemas (FIS)
('02', '07:00', '18:00', 'biblioteca.fiq@epn.edu.ec'); -- Química (FIQA)
GO

-- =====
-- 2. Insertar Infraestructura Física (Pasillos y Estantes)
-- =====

-- --- SEDE SISTEMAS (01) ---
INSERT INTO Pasillo (id_biblioteca, num_pasillo) VALUES
('01', 101), -- Pasillo de Redes
('01', 102); -- Pasillo de Bases de Datos

INSERT INTO Estante (id_biblioteca, num_pasillo, num_estante) VALUES
('01', 101, 1), -- Estante 1 en pasillo 101
('01', 101, 2), -- Estante 2 en pasillo 101
('01', 102, 1); -- Estante 1 en pasillo 102

```

```

-- --- SEDE QUÍMICA (02) ---
INSERT INTO Pasillo (id_biblioteca, num_pasillo) VALUES
('02', 201), -- Pasillo de Orgánica
('02', 202); -- Pasillo de Inorgánica

INSERT INTO Estante (id_biblioteca, num_pasillo, num_estante) VALUES
('02', 201, 1), -- Estante 1 en pasillo 201
('02', 202, 1); -- Estante 1 en pasillo 202
GO

-- =====
-- 3. Insertar Catálogo de Libros (Universal)
-- =====
INSERT INTO Libro (ISBN, nombre_libro, anio_edicion, categoria_libro, lugar_impresion_libro) VALUES
('978-0136086208', 'Clean Code', 2008, 'Software Engineering', 'USA'),
('978-0321125217', 'Domain-Driven Design', 2003, 'Software Architecture', 'UK'),
('978-6073235777', 'Química: La Ciencia Central', 2017, 'Química General', 'Mexico'),
('978-0073511245', 'Database System Concepts', 2019, 'Bases de Datos', 'USA');
GO

-- =====
-- 4. Insertar Ejemplares (Inventario Real con Nuevos Rangos)
-- =====

-- --- SEDE SISTEMAS (01): RANGO 1 - 199 ---
INSERT INTO Ejemplar (ISBN, id_ejemplar, estado_ejemplar, id_biblioteca, num_pasillo, num_estante) VALUES
('978-0136086208', 1, 'Disponible', '01', 101, 1), -- Clean Code (ID 1 es válido para Sede 01)
('978-0073511245', 1, 'Disponible', '01', 102, 1), -- DB Concepts (ID 1 es válido)
('978-0073511245', 2, 'Prestado', '01', 102, 1); -- DB Concepts (ID 2 es válido)

-- --- SEDE QUÍMICA (02): RANGO 200 - 399 ---
INSERT INTO Ejemplar (ISBN, id_ejemplar, estado_ejemplar, id_biblioteca, num_pasillo, num_estante) VALUES
('978-6073235777', 201, 'Disponible', '02', 201, 1), -- Química Central (ID 201 válido para Sede 02)
('978-6073235777', 202, 'Reparación', '02', 201, 1); -- Química Central (ID 202 válido para Sede 02)
GO

-- =====
-- 5. Insertar Usuarios
-- =====
INSERT INTO Usuario (cedula, nombre_usuario, apellido_usuario, email_usuario, celular_usuario, id_biblioteca) VALUES
('1700000001', 'Ismael', 'Freire', 'ismael.freire@epn.edu.ec', '0991234567', '01'),
('1700000002', 'Francis', 'Bravo', 'francis.bravo@epn.edu.ec', '0997654321', '01'),
('1700000003', 'Maria', 'Curie', 'maria.curie@epn.edu.ec', '0981112223', '02'),
('1700000004', 'Alexis', 'Vasco', 'alex.vasco@epn.edu.ec', '0981112224', '02');
GO

-- =====
-- 6. Insertar Préstamos
-- =====

-- CASO A: Usuario de SISTEMAS pide libro de SISTEMAS (Local Sede 01)
-- Ismael pide 'Clean Code' (Ejemplar ID 1 de Sistemas)
INSERT INTO Prestamo (ISBN, id_ejemplar, cedula, fecha_prestamo, fecha_devolucion_tope) VALUES
('978-0136086208', 1, '1700000001', GETDATE(), DATEADD(day, 7, GETDATE()));

-- CASO B: Usuario de QUÍMICA pide libro de QUÍMICA (Local Sede 02)
-- Maria Curie (Sede 02) pide 'Química Central' (Ejemplar ID 201 de Química)
INSERT INTO Prestamo (ISBN, id_ejemplar, cedula, fecha_prestamo, fecha_devolucion_tope) VALUES
('978-6073235777', 201, '1700000003', GETDATE(), DATEADD(day, 7, GETDATE()));

-- CASO C: Usuario de QUÍMICA pide libro de QUÍMICA (Local Sede 02)
-- Alexis pide 'Química Central'.
INSERT INTO Prestamo (ISBN, id_ejemplar, cedula, fecha_prestamo, fecha_devolucion_tope) VALUES
('978-6073235777', 201, '1700000004', GETDATE(), DATEADD(day, 7, GETDATE()));
GO

```

Nodo 1 (FIS):

Creación de la base de datos:

```

/* * Script de Creación - NODO FIS
 * Autores: Alexis Vasco, Ismael Freire, Francis Bravo
 */

-- Si la base de datos ya existe, la borramos para empezar de cero
IF EXISTS (SELECT name FROM sys.databases WHERE name = 'FIS')
BEGIN
    DROP DATABASE FIS;
END
GO

-- 1. Crear la Base de Datos
CREATE DATABASE FIS COLLATE Modern_Spanish_CI_AS
use FIS
go

```

Creación de tablas:

```

-- =====
-- Creacion de tablas
-- =====

-- Tabla: Libro (REPLICADA)
CREATE TABLE Libro (
    ISBN VARCHAR(20) NOT NULL,
    nombre_libro VARCHAR(150) NOT NULL,
    anio_edicion INT,
    categoria_libro VARCHAR(50),
    lugar_impresion_libro VARCHAR(100)
);
GO

-- Tabla: Biblioteca FIS
CREATE TABLE Biblioteca_01 (
    id_biblioteca CHAR(2) NOT NULL,
    horario_apertura TIME NOT NULL,
    horario_cierre TIME NOT NULL,
    correo_biblioteca VARCHAR(100)
);
GO

```

```
-- Pasillo_01
CREATE TABLE Pasillo_01 (
    id_biblioteca CHAR(2) NOT NULL,
    num_pasillo INT NOT NULL
);
GO

-- Estante_01
CREATE TABLE Estante_01 (
    num_estante INT NOT NULL,
    id_biblioteca CHAR(2) NOT NULL,
    num_pasillo INT NOT NULL
);
GO

-- Ejemplar_01
CREATE TABLE Ejemplar_01 (
    ISBN VARCHAR(20) NOT NULL,
    id_ejemplar INT NOT NULL,
    estado_ejemplar VARCHAR(20),
    -- Columnas de ubicación física necesarias para la relación
    id_biblioteca CHAR(2) NOT NULL,
    num_pasillo INT NOT NULL,
    num_estante INT NOT NULL
);
GO
```

```
-- Prestamo_01
CREATE TABLE Prestamo_01 (
    ISBN VARCHAR(20) NOT NULL,
    id_ejemplar INT NOT NULL,
    cedula VARCHAR(15) NOT NULL,
    fecha_prestamo DATETIME NOT NULL,
    fecha_devolucion_tope DATETIME NOT NULL,
    fecha_devolucion DATETIME NULL
);
GO

-- Usuarios_info_01
CREATE TABLE Usuarios_info_01 (
    cedula VARCHAR(15) NOT NULL,
    nombre_usuario VARCHAR(50) NOT NULL,
    apellido_usuario VARCHAR(50) NOT NULL,
    id_biblioteca CHAR(2) NOT NULL
);
GO

-- Usuario_contacto
CREATE TABLE Usuario_contacto (
    cedula VARCHAR(15) NOT NULL,
    email_usuario VARCHAR(100),
    celular_usuario VARCHAR(20)
);
GO
```

Definición de claves primarias y foráneas:

```

-- =====
-- Definicion de claves (PK) y relaciones (FK)
-- =====

-- Libro
ALTER TABLE Libro
ADD CONSTRAINT PK_Libro PRIMARY KEY (ISBN);

-- Biblioteca_01
ALTER TABLE Biblioteca_01
ADD CONSTRAINT PK_Biblioteca PRIMARY KEY (id_biblioteca);

-- Pasillo_01
ALTER TABLE Pasillo_01
ADD CONSTRAINT PK_Pasillo PRIMARY KEY (id_biblioteca, num_pasillo);
ALTER TABLE Pasillo_01
ADD CONSTRAINT FK_Pasillo_Biblioteca FOREIGN KEY (id_biblioteca)
REFERENCES Biblioteca_01 (id_biblioteca);

-- Estante_01
ALTER TABLE Estante_01
ADD CONSTRAINT PK_Estante PRIMARY KEY (id_biblioteca, num_pasillo, num_estante);
ALTER TABLE Estante_01
ADD CONSTRAINT FK_Estante_Pasillo FOREIGN KEY (id_biblioteca, num_pasillo)
REFERENCES Pasillo_01 (id_biblioteca, num_pasillo);

-- Ejemplar_01
ALTER TABLE Ejemplar_01
ADD CONSTRAINT PK_Ejemplar PRIMARY KEY (ISBN, id_ejemplar);
ALTER TABLE Ejemplar_01
ADD CONSTRAINT FK_Ejemplar_Libro FOREIGN KEY (ISBN)
REFERENCES Libro (ISBN);
ALTER TABLE Ejemplar_01
ADD CONSTRAINT FK_Ejemplar_Estante FOREIGN KEY (id_biblioteca, num_pasillo, num_estante)
REFERENCES Estante_01 (id_biblioteca, num_pasillo, num_estante);

-- agregar restriccion
ALTER TABLE Ejemplar_01
ADD CONSTRAINT CK_Ejemplar_RangoID CHECK (
    (id_biblioteca = '01' AND id_ejemplar BETWEEN 1 AND 199) OR -- Rango Sede 01
    (id_biblioteca = '02' AND id_ejemplar BETWEEN 200 AND 399) -- Rango Sede 02
);

-- Prestamo_01
ALTER TABLE Prestamo_01
ADD CONSTRAINT PK_Prestamo PRIMARY KEY (ISBN, id_ejemplar, cedula, fecha_prestamo);
ALTER TABLE Prestamo_01
ADD CONSTRAINT FK_Prestamo_Usuarios FOREIGN KEY (cedula)
REFERENCES Usuarios_info_01 (cedula);
ALTER TABLE Prestamo_01
ADD CONSTRAINT FK_Prestamo_Ejemplar FOREIGN KEY (ISBN, id_ejemplar)
REFERENCES Ejemplar_01 (ISBN, id_ejemplar);

-- 7. Usuarios_info_01
ALTER TABLE Usuarios_info_01
ADD CONSTRAINT PK_Usuarios_info PRIMARY KEY (cedula);
ALTER TABLE Usuarios_info_01
ADD CONSTRAINT FK_Usuarios_info_Biblioteca FOREIGN KEY (id_biblioteca)
REFERENCES Biblioteca_01 (id_biblioteca);

-- 8. Usuario_contacto
ALTER TABLE Usuario_contacto
ADD CONSTRAINT PK_Usuario_contacto PRIMARY KEY (cedula);
GO

```


Extracción de los datos de la base de datos centralizada:

```
-- =====
--  Extraccion de datos
--  =====

-- 1. Carga Libro
INSERT INTO Libro
SELECT * FROM BibliotecaUCentral.dbo.Libro;

-- Carga Biblioteca_01 (Solo FIS)
INSERT INTO Biblioteca_01
SELECT * FROM BibliotecaUCentral.dbo.Biblioteca
WHERE id_biblioteca = '01';

-- Carga Pasillo_01 (Solo FIS)
INSERT INTO Pasillo_01
SELECT * FROM BibliotecaUCentral.dbo.Pasillo
WHERE id_biblioteca = '01';

-- Carga Estante_01 (Solo FIS)
INSERT INTO Estante_01 (num_estante, id_biblioteca, num_pasillo)
SELECT num_estante, id_biblioteca, num_pasillo
FROM BibliotecaUCentral.dbo.Estante
WHERE id_biblioteca = '01';

-- Carga Ejemplar_01 (Solo FIS - Derivada)
INSERT INTO Ejemplar_01
SELECT * FROM BibliotecaUCentral.dbo.Ejemplar
WHERE id_biblioteca = '01';

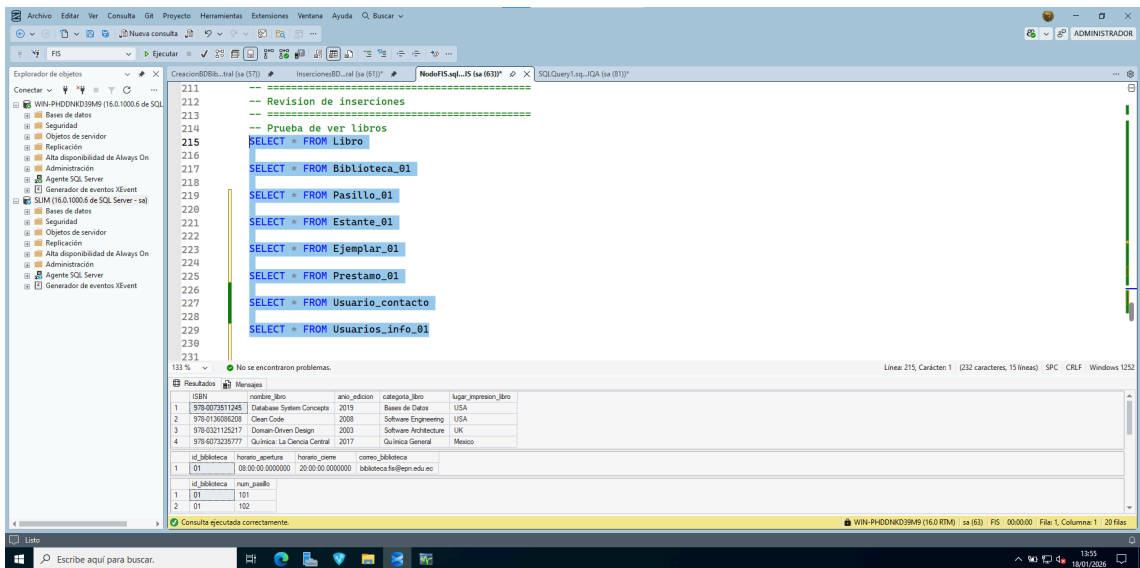
-- Carga Prestamo_01 (Derivada por Join)
INSERT INTO Prestamo_01
SELECT P.* FROM BibliotecaUCentral.dbo.Prestamo P
INNER JOIN BibliotecaUCentral.dbo.Ejemplar E
    ON P.ISBN = E.ISBN AND P.id_ejemplar = E.id_ejemplar
INNER JOIN BibliotecaUCentral.dbo.Usuario U
    ON P.cedula = U.cedula
WHERE
    E.id_biblioteca = '01'
    AND U.id_biblioteca = '01';

-- Carga Usuarios_info_01 (Solo FIS)
INSERT INTO Usuarios_info_01 (cedula, nombre_usuario, apellido_usuario, id_biblioteca)
SELECT cedula, nombre_usuario, apellido_usuario, id_biblioteca
FROM BibliotecaUCentral.dbo.Usuario
WHERE id_biblioteca = '01';

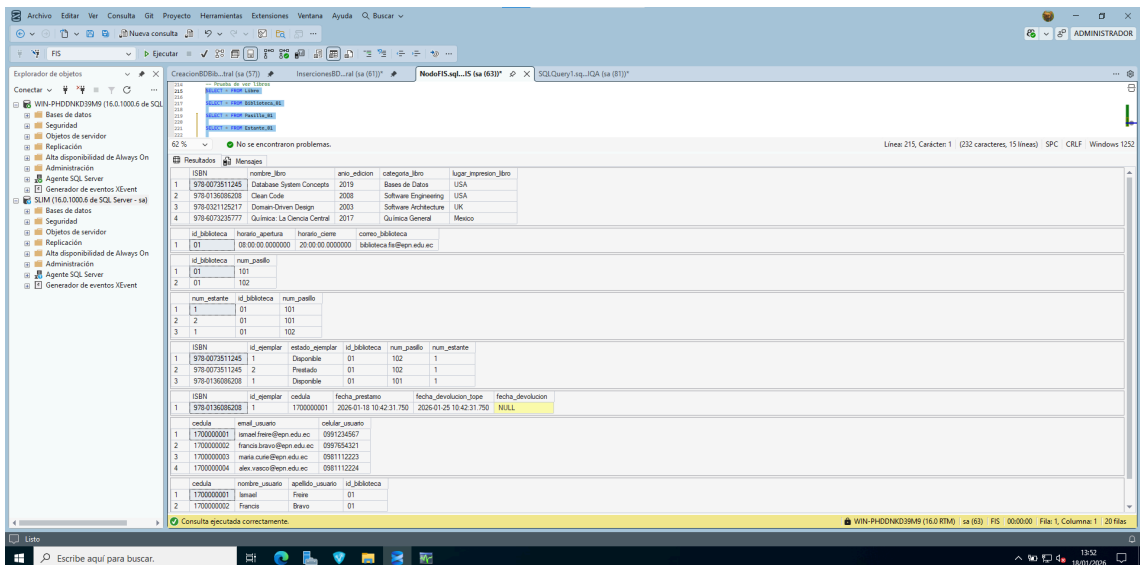
-- Carga Usuario_contacto (Todos - Vertical)
INSERT INTO Usuario_contacto (cedula, email_usuario, celular_usuario)
SELECT cedula, email_usuario, celular_usuario
FROM BibliotecaUCentral.dbo.Usuario;
```

Verificación de migración - nodo FIS:

Consultas ejecutadas:



Vista de las tablas:



Nodo 2 (FIQA):

Creación de la base de datos:

```

CREATE DATABASE FIQA COLLATE Modern_Spanish_CI_AS;
USE FIQA;
GO

```

Creación de tablas:

```
-- =====
-- 1. Tabla: Biblioteca_02 (Fragmento Horizontal)
-- =====
```

```
CREATE TABLE Biblioteca_02 (
    id_biblioteca CHAR(2) NOT NULL,
    horario_apertura TIME NOT NULL,
    horario_cierre TIME NOT NULL,
    correo_biblioteca VARCHAR(100)
);
```

```
-- =====
-- 2. Tabla: Pasillo_02 (Fragmento Horizontal)
-- =====
```

```
CREATE TABLE Pasillo_02 (
    id_biblioteca CHAR(2) NOT NULL,
    num_pasillo INT NOT NULL
);
```

```
-- =====
-- 3. Tabla: Estante_02 (Fragmento Horizontal)
-- =====
```

```
CREATE TABLE Estante_02 (
    num_estante INT NOT NULL,
    id_biblioteca CHAR(2) NOT NULL,
    num_pasillo INT NOT NULL
);
```

```
-- =====
-- 4. Tabla: Ejemplar_02 (Fragmento Horizontal)
-- =====
```

```
CREATE TABLE Ejemplar_02 (
    ISBN VARCHAR(20) NOT NULL,
    id_ejemplar INT NOT NULL,
    estado_ejemplar VARCHAR(20),
    -- Columnas de ubicación física
    id_biblioteca CHAR(2) NOT NULL,
    num_pasillo INT NOT NULL,
    num_estante INT NOT NULL
);
```

```

-- =====
-- 5. Tabla: Usuarios_info_02 (Fragmento Mixto)
-- =====
-- Contendrá solo a los estudiantes de la FIQA.
CREATE TABLE Usuarios_info_02 (
    cedula VARCHAR(15) NOT NULL,
    nombre_usuario VARCHAR(50) NOT NULL,
    apellido_usuario VARCHAR(50) NOT NULL,
    id_biblioteca CHAR(2) NOT NULL
);

-- =====
-- 6. Tabla: Prestamo_02 (Fragmento Derivado)
-- =====
CREATE TABLE Prestamo_02 (
    ISBN VARCHAR(20) NOT NULL,
    id_ejemplar INT NOT NULL,
    cedula VARCHAR(15) NOT NULL,
    fecha_prestamo DATETIME NOT NULL,
    fecha_devolucion_tope DATETIME NOT NULL,
    fecha_devolucion DATETIME NULL
);

```

Definiciones de claves (PKs y FKs):

```

-- Biblioteca_02
ALTER TABLE Biblioteca_02
ADD CONSTRAINT PK_Biblioteca PRIMARY KEY (id_biblioteca);

-- Pasillo_02
ALTER TABLE Pasillo_02
ADD CONSTRAINT PK_Pasillo PRIMARY KEY (id_biblioteca, num_pasillo);
ALTER TABLE Pasillo_02
ADD CONSTRAINT FK_Pasillo_Biblioteca FOREIGN KEY (id_biblioteca)
REFERENCES Biblioteca_02 (id_biblioteca);

-- Estante_02
ALTER TABLE Estante_02
ADD CONSTRAINT PK_Estante PRIMARY KEY (id_biblioteca, num_pasillo, num_estante);
ALTER TABLE Estante_02
ADD CONSTRAINT FK_Estante_Pasillo FOREIGN KEY (id_biblioteca, num_pasillo)
REFERENCES Pasillo_02 (id_biblioteca, num_pasillo);

--Ejemplar_02
ALTER TABLE Ejemplar_02
ADD CONSTRAINT PK_Ejemplar PRIMARY KEY (ISBN, id_ejemplar);
-- Estante -> Ejemplar (Ubicación física)
ALTER TABLE Ejemplar_02
ADD CONSTRAINT FK_Ejemplar_Estante FOREIGN KEY (id_biblioteca, num_pasillo, num_estante)
REFERENCES Estante_02 (id_biblioteca, num_pasillo, num_estante);
-- Libro -> Ejemplar
ALTER TABLE Ejemplar_02
ADD CONSTRAINT FK_Ejemplar_Libro FOREIGN KEY (ISBN)
REFERENCES Libro (ISBN);
ALTER TABLE Ejemplar_02
ADD CONSTRAINT CK_Ejemplar_RangoID CHECK (
(id_biblioteca = '01' AND id_ejemplar BETWEEN 1 AND 199)      -- Rango Sede 01 (FIS)
OR
(id_biblioteca = '02' AND id_ejemplar BETWEEN 200 AND 399)   -- Rango Sede 02 (FIQA)
);

```

```

--Usuarios_info_02
ALTER TABLE Usuarios_info_02
ADD CONSTRAINT PK_Usuarios_info PRIMARY KEY (cedula);
-- Biblioteca -> Usuario
ALTER TABLE Usuarios_info_02
ADD CONSTRAINT FK_Usuarios_info_Biblioteca FOREIGN KEY (id_biblioteca)
REFERENCES Biblioteca_02 (id_biblioteca);

--Prestamo

-- PK Histórica para préstamos
ALTER TABLE Prestamo_02
ADD CONSTRAINT PK_Prestamo PRIMARY KEY (ISBN, id_ejemplar, cedula, fecha_prestamo);
GO
-- Usuario -> Prestamo (Validación Local Estricta)
ALTER TABLE Prestamo_02
ADD CONSTRAINT FK_Prestamo_Usuarios FOREIGN KEY (cedula)
REFERENCES Usuarios_info_02 (cedula);
GO

-- Ejemplar -> Prestamo
ALTER TABLE Prestamo_02
ADD CONSTRAINT FK_Prestamo_Ejemplar FOREIGN KEY (ISBN, id_ejemplar)
REFERENCES Ejemplar_02 (ISBN, id_ejemplar);

```

Extracción de los datos de la base de datos centralizada:

```

-- 1. Carga Biblioteca_02 (FIQA)
INSERT INTO Biblioteca_02
SELECT * FROM [WIN-PHDDNKD39M9].[BibliotecaUCentral].dbo.Biblioteca
WHERE id_biblioteca = '02';

-- 2. Carga Pasillo_02 (FIQA)
INSERT INTO Pasillo_02
SELECT * FROM [WIN-PHDDNKD39M9].[BibliotecaUCentral].dbo.Pasillo
WHERE id_biblioteca = '02';

-- 3. Carga Estante_02 (FIQA)
INSERT INTO Estante_02 (num_estante, id_biblioteca, num_pasillo)
SELECT num_estante, id_biblioteca, num_pasillo
FROM [WIN-PHDDNKD39M9].[BibliotecaUCentral].dbo.Estante
WHERE id_biblioteca = '02';

-- 4. Carga Ejemplar_02 (FIQA)
INSERT INTO Ejemplar_02
SELECT * FROM [WIN-PHDDNKD39M9].[BibliotecaUCentral].dbo.Ejemplar
WHERE id_biblioteca = '02';

-- 5. Carga Usuarios_info_02 (FIQA)
INSERT INTO Usuarios_info_02 (cedula, nombre_usuario, apellido_usuario, id_biblioteca)
SELECT cedula, nombre_usuario, apellido_usuario, id_biblioteca
FROM [WIN-PHDDNKD39M9].[BibliotecaUCentral].dbo.Usuario
WHERE id_biblioteca = '02';

```



```
-- 6. Carga Prestamo_02 (FIQA)
INSERT INTO Prestamo_02 (ISBN, id_ejemplar, cedula, fecha_prestamo, fecha_devolucion_tope, fecha_devolucion)
SELECT
    P.ISBN,
    P.id_ejemplar,
    P.cedula,
    P.fecha_prestamo,
    P.fecha_devolucion_tope,
    P.fecha_devolucion
FROM [WIN-PHDDNKD39M9].[BibliotecaUCentral].dbo.Prestamo P
INNER JOIN [WIN-PHDDNKD39M9].[BibliotecaUCentral].dbo.Ejemplar E
    ON P.ISBN = E.ISBN AND P.id_ejemplar = E.id_ejemplar
INNER JOIN [WIN-PHDDNKD39M9].[BibliotecaUCentral].dbo.Usuario U
    ON P.cedula = U.cedula
WHERE
    E.id_biblioteca = '02'
    AND U.id_biblioteca = '02';
```

Verificación de migración - nodo FIQA:

The screenshot displays the SQL Server Enterprise Manager interface. The left pane shows the 'Explorador de objetos' (Object Explorer) with the 'FIQA' node selected. The central pane shows the 'SQLQuery1.s...A (sa (67))' window with the following query:

```
-- Verificación de Migración - Nodo FIQA
--
Select * from Libro
Select * from Biblioteca_02
Select * from Pasillo_02
Select * from Estante_02
Select * from Ejemplar_02
Select * from Prestamo_02
Select * from Usuarios_info_02
```

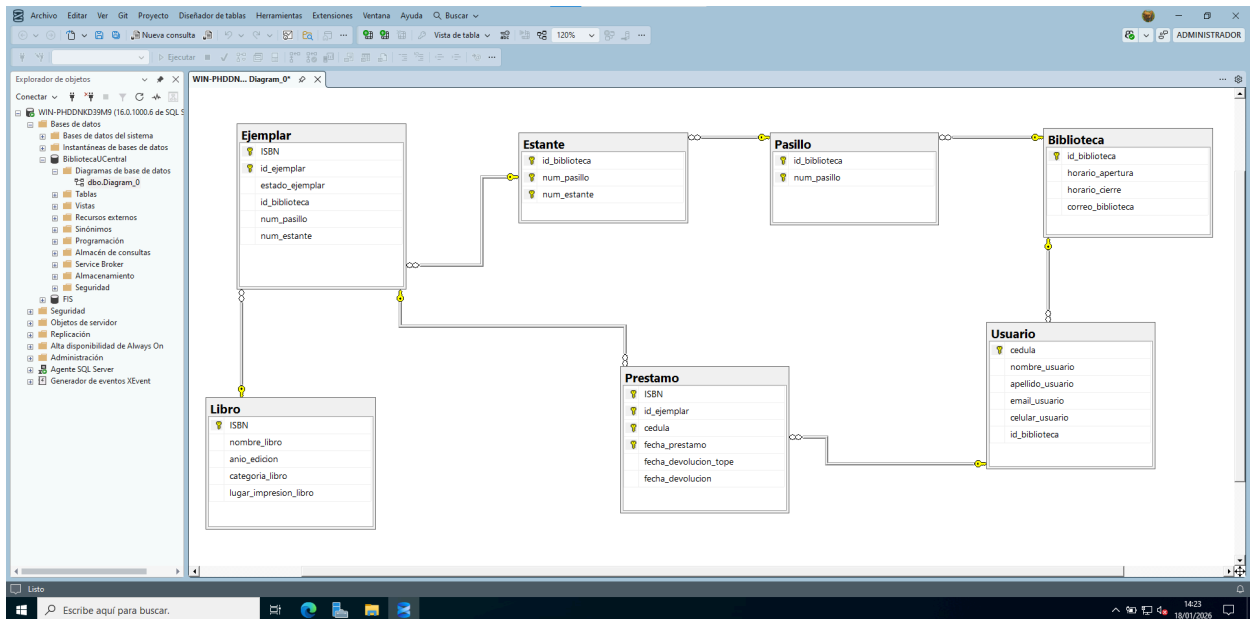
The bottom pane shows the 'Resultados' (Results) window with the following data:

ISBN	nombre_libro	anio_edicion	categoria_libro	lugar_impresion_libro
978-0073511245	Databases System Concepts	2018	Bases de Datos	USA
978-0136086208	Clean Code	2008	Software Engineering	USA
978-0321125217	Domain-Driven Design	2003	Software Architecture	UK
978-6073235777	Química: La Ciencia Central	2017	Química General	Mexico

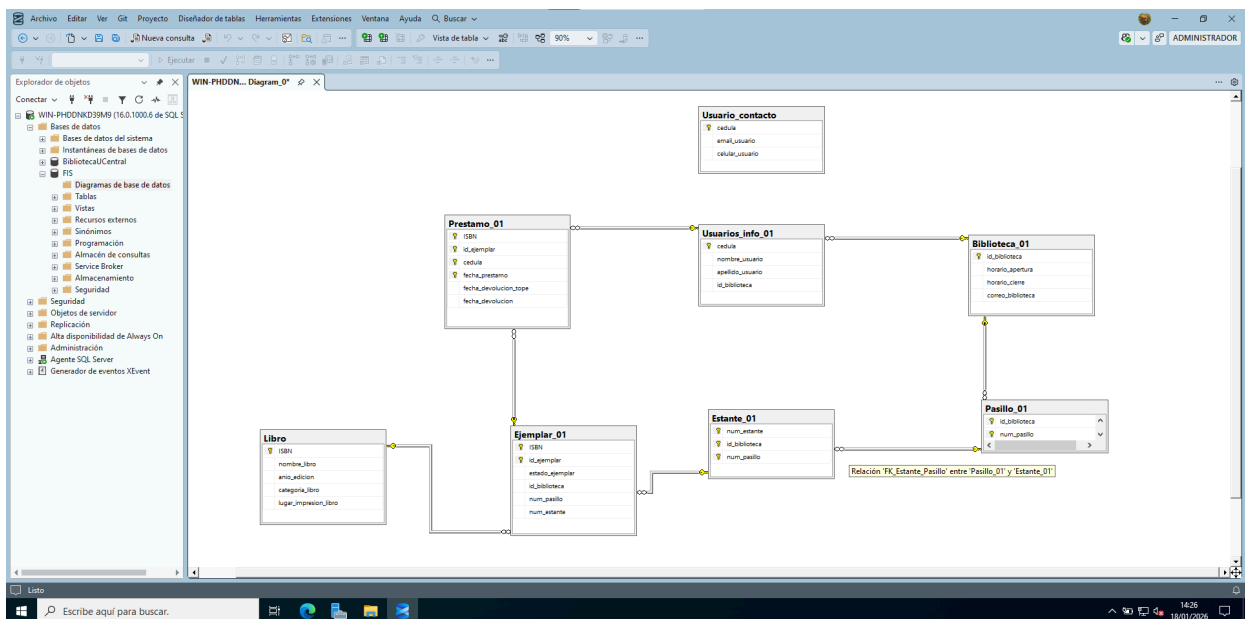
The status bar at the bottom indicates: 'Consulta ejecutada correctamente. SLIM (16.0 RTM) - sa (67) FIQA 00:00:00 Fila: 1, Columna: 1 2 filas'.

4.- Presentar el diagrama de las bases de datos locales y la base centralizada

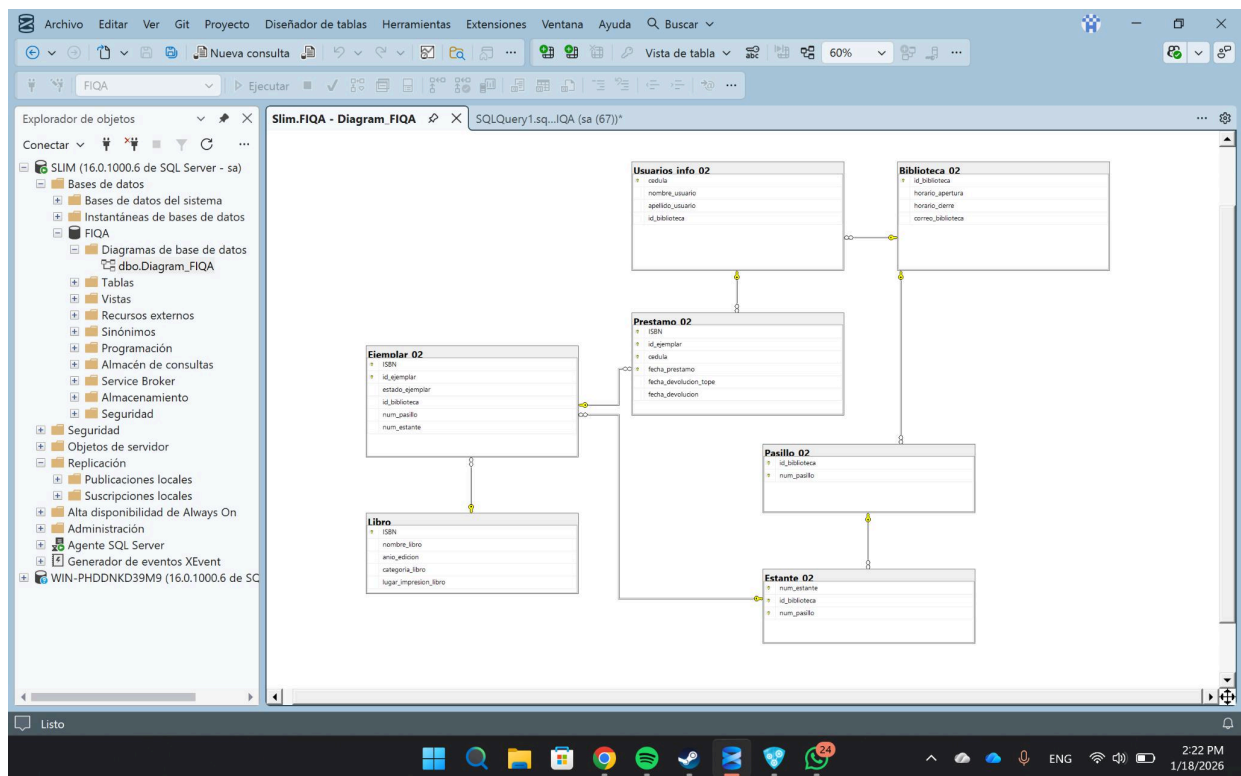
4.1 Diagrama de la base de datos central



4.2 Diagrama de la base de datos de FIS



4.3 Diagrama de la base de datos de FIQA



Conclusiones:

- Se implementó la fragmentación horizontal y mixta bajo criterios de sede, garantizando la completitud y disyunción de la información en el sistema.
- Se estableció un esquema de ubicación eficiente que minimiza la latencia de red al situar los datos en sus respectivos nodos de consulta frecuente.
- Se verificó la correcta migración de datos desde la base centralizada, asegurando que los esquemas locales mantengan su integridad y autonomía operativa.
- Se consolidó la arquitectura técnica mediante diagramas que muestran la correspondencia entre la visión global y la distribución física de los datos.

Recomendaciones:

- Monitorear periódicamente el rendimiento de las consultas en cada nodo para ajustar los criterios de fragmentación en caso de un crecimiento exponencial de los datos.
- Automatizar los procesos de extracción y carga de datos mediante tareas programadas, con el fin de garantizar que la base centralizada y los nodos locales se mantengan sincronizados.
- Implementar protocolos de seguridad y planes de contingencia en cada esquema local para asegurar la disponibilidad de la información ante posibles fallos de conexión en la red.

Bibliografía:

[1] M. T. Ozsú and P. Valduriez, *Principles of Distributed Database Systems*, 3rd ed. New York, NY, USA: Springer, 2011.

[2] D. Cuadra et al., *Desarrollo de Bases de Datos: Casos prácticos desde el análisis a la implementación*, 2nd ed. Madrid, Spain: Ra-Ma, 2014.

[3] R. Elmasri and S. Navathe, *Fundamentals of Database Systems*, 7th ed. Boston, MA, USA: Pearson, 2016.

[4] T. Connolly and C. Begg, *Database Systems: A Practical Approach to Design, Implementation and Management*. Harlow, UK: Pearson Education Limited, 2015.

[5] A. Siberschatz, H. Korth, and S. Sudarshan, *Database System Concepts*, 7th ed. New York, NY, USA: McGraw-Hill Education, 2019.

[6] M. T. Ozsu and P. Valduriez, *Principles of Distributed Database Systems*, 4th ed. Cham, Switzerland: Springer, 2020.