



Genesis Edel Ortiz Perez

25490115

Fundamentos de base de datos

Juan Ramon Bogarin Valenzuela

Examen Final de Unidad 4

Viernes 23 de mayo del 2023

9:00 pm

Descripción: Documento explicativo del diseño de la base de datos

Problema Técnico: Gestionando la Información de una Universidad

Imagina que estás trabajando en el equipo de desarrollo de software para una universidad. La universidad necesita un sistema para gestionar la información de sus estudiantes, los cursos que ofrecen, las inscripciones de los estudiantes a los cursos, los profesores que imparten los cursos y los departamentos a los que pertenecen los profesores. Tu tarea es diseñar y trabajar con la base de datos que almacenará toda esta información.

Objetivo General:

Diseñar una base de datos relacional y realizar diversas operaciones para gestionar la información de la universidad. Esto incluye la creación y modificación de la estructura de las tablas, la manipulación de los datos y la realización de consultas complejas para obtener información específica.

- ◆ ****Planteamiento del problema****

El objetivo es diseñar una base de datos que permita gestionar la información de una universidad, incluyendo estudiantes, profesores, cursos, horarios, inscripciones y programas de estudio. Se busca crear una estructura que optimice la consulta de datos y facilite la administración académica.

- ◆ ****Solución propuesta****

Para solucionar este problema, se diseñó una base de datos relacional con las siguientes tablas:

Estudiantes: Almacena información básica de los estudiantes.

Departamentos: Organiza las facultades dentro de la universidad.

Aulas: Registra las ubicaciones de los cursos.

Cursos: Almacena la información de cada materia impartida.

Horarios: Contiene el calendario de clases para cada curso.

Profesores: Registra los datos de los docentes.

Inscripciones: Relaciona estudiantes con cursos específicos.

Cursos_Profesores: Asocia cursos con profesores.

ProgramasEstudio: Define los programas académicos.

Programa_Cursos: Relaciona los programas con los cursos que contienen.

Diseño de la base de datos

La base de datos sigue un esquema relacional con claves primarias y foráneas para garantizar la integridad de los datos:

Claves primarias ('PRIMARY KEY')

- Cada tabla cuenta con una clave primaria, utilizando 'SERIAL' para asegurar unicidad.

Claves foráneas ('FOREIGN KEY'):

- 'Cursos' se relaciona con 'Departamentos', asegurando que cada curso pertenece a una facultad.
- 'Horarios' vincula 'Cursos' con 'Aulas', organizando la distribución de clases.
- 'Inscripciones' enlaza 'Estudiantes' con 'Cursos', registrando qué alumno está en qué materia.
- 'Cursos_Profesores' asocia cursos con profesores.

Observaciones y justificaciones

Uso de 'SERIAL' en claves primarias: Se optó por 'SERIAL' para asignar IDs automáticamente, simplificando la administración.

Restricciones en atributos: Se definieron reglas como 'CHECK (Creditos > 0)' para evitar valores inválidos.

Uso de índices implícitos: Las claves primarias y foráneas permiten mejorar la velocidad de consulta.

Organización de relaciones: Se estructuró la información de manera modular para garantizar un diseño escalable y eficiente.

Resultados obtenidos

- Se crearon las tablas exitosamente sin conflictos en integridad referencial.
- Se probaron consultas 'SELECT', 'JOIN', 'GROUP BY', obteniendo resultados esperados.
- Se verificó que el esquema es escalable para futuras mejoras.

Esquema de la Base de Datos:

Aquí se presenta un esquema de la base de datos con 10 tablas y sus relaciones:

Modelo de Datos Lógico (MDL):

- **Tabla: Estudiantes**

- IDEstudiante (Clave Principal)
- Nombre
- Apellido
- FechaNacimiento
- Direccion
- Ciudad

- Email
- **Tabla: Cursos**
 - IDCurso (Clave Principal)
 - NombreCurso
 - Descripcion
 - Creditos
 - Semestre
 - IDDepartamento (Clave Foránea, referencia a la tabla Departamentos)
- **Tabla: Inscripciones**
 - IDInscripcion (Clave Principal)
 - IDEstudiante (Clave Foránea, referencia a la tabla Estudiantes)
 - IDCurso (Clave Foránea, referencia a la tabla Cursos)
 - Fechalincripcion
 - Calificacion
- **Tabla: Profesores**
 - IDProfesor (Clave Principal)
 - Nombre
 - Apellido
 - Titulo
 - IDDepartamento (Clave Foránea, referencia a la tabla Departamentos)
- **Tabla: Departamentos**
 - IDDepartamento (Clave Principal)
 - NombreDepartamento
 - Edificio
- **Tabla: Aulas**
 - IDAula (Clave Principal)
 - NombreAula
 - Capacidad
 - Ubicacion
- **Tabla: Horarios**
 - IDHorario (Clave Principal)
 - IDCurso (Clave Foránea, referencia a la tabla Cursos)
 - IDAula (Clave Foránea, referencia a la tabla Aulas)
 - Fechalinicio
 - FechaFin
 - HoralInicio
 - HoraFin

- **Tabla Intermedia: CursosProfesores** (Relación Muchos a Muchos entre Cursos y Profesores)
 - IDCursoProfesor (Clave Principal)
 - IDCurso (Clave Foránea, referencia a la tabla Cursos)
 - IDProfesor (Clave Foránea, referencia a la tabla Profesores)
- **Tabla: ProgramasEstudio**
 - IDPrograma (Clave Principal)
 - NombrePrograma
 - DescripcionPrograma
- **Tabla Intermedia: ProgramasCursos** (Relación Muchos a Muchos entre ProgramasEstudio y Cursos)
 - IDProgramaCurso (Clave Principal)
 - IDPrograma (Clave Foránea, referencia a la tabla ProgramasEstudio)
 - IDCurso (Clave Foránea, referencia a la tabla Cursos)

Capturas de pantalla

Tablas

```

1 v CREATE TABLE Estudiantes(
2   id_estudiante SERIAL PRIMARY KEY,
3   Nombre VARCHAR(100)NOT NULL,
4   Apellido VARCHAR(100)NOT NULL,
5   FechaNac DATE,
6   Direccion VARCHAR(250),
7   Ciudad VARCHAR(100),
8   Email VARCHAR(250)UNIQUE
9 );
10 v select*from Estudiantes
11
Data Output  Mensajes  Notificaciones

```

id_estudiante	nombre	apellido	fechanac	direccion	ciudad	email
[PK] integer	character varying (100)	character varying (100)	date	character varying (250)	character varying (100)	character varying (250)

```

1 v CREATE TABLE Departamentos(
2   id_departamento SERIAL PRIMARY KEY,
3   NombreDepartamento VARCHAR(100)NOT NULL,
4   Edificio VARCHAR(50)NOT NULL
5 );
6 v select*from Departamentos
7
Data Output  Mensajes  Notificaciones

```

id_departamento	nombredepartamento	edificio
[PK] integer	character varying (100)	character varying (50)

```

1 ✓ CREATE TABLE Aulas(
2   id_aula SERIAL PRIMARY KEY,
3   NombreAula VARCHAR(100)NOT NULL,
4   Capacidad INT NOT NULL CHECK (Capacidad>0),
5   Ubicacion VARCHAR(100)
6 );
7 ✓ select*from Aulas
8

```

Data Output Mensajes Notificaciones

	id_aula [PK] integer	nombreaula character varying (100)	capacidad integer	ubicacion character varying (100)
--	-------------------------	---------------------------------------	----------------------	--------------------------------------

```

1 ✓ CREATE TABLE Cursos(
2   id_curso SERIAL PRIMARY KEY,
3   NombreCurso VARCHAR(100)NOT NULL,
4   Descripcion TEXT,
5   Creditos INT NOT NULL CHECK(Creditos>0),
6   Semestre INT NOT NULL CHECK(Semestre>0),
7   id_departamento INT,
8   FOREIGN KEY (id_departamento)REFERENCES Departamentos (id_departamento)
9 );
10 ✓ select*from Cursos

```

Data Output Mensajes Notificaciones

	id_curso [PK] integer	nombrcurso character varying (100)	descripcion text	creditos integer	semestre integer	id_departamento integer
--	--------------------------	---------------------------------------	---------------------	---------------------	---------------------	----------------------------

```

1 ✓ CREATE TABLE Horarios(
2   id_horario SERIAL PRIMARY KEY, |
3   FechaInicio DATE,
4   FechaFin DATE CHECK (FechaFin>FechaInicio),
5   HoraInicio TIME NOT NULL,
6   HoraFin TIME NOT NULL CHECK (HoraFin>HoraInicio),
7   id_curso INT,
8   id_Aula INT,
9   FOREIGN KEY (id_curso)REFERENCES Cursos(id_curso),
10  FOREIGN KEY (id_aula)REFERENCES Aulas(id_aula)
11 );
12 ✓ SELECT*FROM Horarios
13

```

Data Output Mensajes Notificaciones

	id_horario [PK] integer	fechainicio date	fechafin date	horainicio time without time zone	horafin time without time zone	id_curso integer	id_aula integer
--	----------------------------	---------------------	------------------	--------------------------------------	-----------------------------------	---------------------	--------------------

```

1 ✓ CREATE TABLE Profesores(
2   id_profesor SERIAL PRIMARY KEY,
3   Nombre VARCHAR(50)NOT NULL,
4   Apellido VARCHAR(100)NOT NULL,
5   Titulo VARCHAR(100) NOT NULL,
6   id_departamento INT,
7   FOREIGN KEY (id_departamento)REFERENCES Departamentos(id_departamento)
8 );
9 ✓ select*from Profesores
10

```

Data Output Mensajes Notificaciones

	id_profesor [PK] integer	nombre character varying (50)	apellido character varying (100)	titulo character varying (100)	id_departamento integer
--	-----------------------------	----------------------------------	-------------------------------------	-----------------------------------	----------------------------

```

1 ✓ CREATE TABLE Inscripciones(
2   id_inscripcion SERIAL PRIMARY KEY,
3   FechaIncripcion DATE NOT NULL,
4   Calificacion DECIMAL(10,2),
5   id_estudiante INT,
6   id_curso INT,
7   FOREIGN KEY (id_estudiante)REFERENCES Estudiantes(id_estudiante),
8   FOREIGN KEY (id_curso)REFERENCES Cursos(id_curso)
9 );
10 ✓ select*from Inscripciones
11

```

Data Output Mensajes Notificaciones

	id_inscripcion [PK] integer	fechainscripcion date	calificacion numeric (10,2)	id_estudiante integer	id_curso integer
--	--------------------------------	--------------------------	--------------------------------	--------------------------	---------------------

```

1 ✓ CREATE TABLE Cursos_Profesores (
2   id_CursosProfesores SERIAL PRIMARY KEY,
3   id_curso INT,
4   id_profesor INT,
5   FOREIGN KEY (id_curso)REFERENCES Cursos(id_curso),
6   FOREIGN KEY (id_profesor)REFERENCES Profesores(id_profesor)
7 );
8 ✓ select*from Cursos_Profesores
9

```

Data Output Mensajes Notificaciones

	id_cursosprofesores [PK] integer	id_curso integer	id_profesor integer
--	-------------------------------------	---------------------	------------------------

```

1 v CREATE TABLE ProgramasEstudio(
2   id_programa SERIAL PRIMARY KEY,
3   NombrePrograma VARCHAR(100) NOT NULL,
4   DescripcionPrograma VARCHAR(100)NOT NULL
5 );
6 v select*from ProgramasEstudio
7

```

Data Output Mensajes Notificaciones

	id_programa [PK] integer	nombreprograma character varying (100)	descripcionprograma character varying (100)

```

1 v CREATE TABLE Programa_Cursos(
2   id_ProgramaCurso SERIAL PRIMARY KEY,
3   id_programa INT,
4   id_curso INT,
5   FOREIGN KEY (id_programa)REFERENCES ProgramasEstudio(id_programa),
6   FOREIGN KEY (id_curso)REFERENCES Cursos(id_curso)
7 );
8 v select*from Programa_Cursos
9
10
11

```

Data Output Mensajes Notificaciones

	id_programacurso [PK] integer	id_programa integer	id_curso integer

```

1 v CREATE TABLE Carreras (
2   id_carrera SERIAL PRIMARY KEY,
3   NombreCarrera VARCHAR(100) NOT NULL,
4   TituloOtorgado VARCHAR(100) NOT NULL
5 );
6 v select*from Carreras
7

```

Data Output Mensajes Notificaciones

	id_carrera [PK] integer	nombrercarrera character varying (100)	titulootorgado character varying (100)

Consulta Historial de Consultas

```
1 ▼ CREATE TABLE Campus (
2   id_campus SERIAL PRIMARY KEY,
3   NombreCampus VARCHAR(255) NOT NULL,
4   DireccionCampus VARCHAR(255) NOT NULL
5 );
6 ▼ select*from Campus
7
```

Data Output Mensajes Notificaciones

	id_campus [PK] integer	nombrecampus character varying (255)	direccioncampus character varying (255)

```
1 ▼ |CREATE TABLE Estudiantes_Carreras (
2   id_estudiantes_carreras SERIAL PRIMARY KEY,
3   id_estudiante INT,
4   id_carrera INT,
5   FechaInscripcion DATE NOT NULL,
6   FOREIGN KEY (id_estudiante) REFERENCES Estudiantes(id_estudiante),
7   FOREIGN KEY (id_carrera) REFERENCES Carreras(id_carrera)
8 );
9 select*from Estudiantes_Carreras
```

Data Output Mensajes Notificaciones

	id_estudiantes_carreras [PK] integer	id_estudiante integer	id_carrera integer	fechainscripcion date

Consultas Solicitadas

```
--Selección Básica: Muestra todos los nombres y apellidos de los estudiantes
22 SELECT Nombre, Apellido FROM Estudiantes;
23
24
```

Data Output Mensajes Notificaciones

	nombre character varying (100)	apellido character varying (100)
1	Juan	Pérez
2	María	González
3	Luis	Martínez
4	Fernanda	Hernández
5	Ricardo	Gómez
6	Natalia	Torres

```

24
25  --Cláusula WHERE: Encuentra todos los cursos que tienen 3 créditos.
26
27 v SELECT NombreCurso FROM Cursos
28 WHERE Creditos=3;
29

```

Data Output Mensajes Notificaciones

SQL

	nombrcurso character varying (100)
1	Gestión Empresarial
2	Psicología Social

```

35
36  --LEFT JOIN: Muestra todos los estudiantes y, si están inscritos en algún curso,
37  --el nombre del curso. Si un estudiante no está inscrito en ningún curso,
38  --el campo del nombre del curso debe mostrar un valor que lo indique (ej: NULL).
39
40 SELECT e.Nombre, e.Apellido, c.NombreCurso
41 FROM Estudiantes e
42 LEFT JOIN Inscripciones i ON e.id_estudiante= i.id_estudiante
43 LEFT JOIN Cursos c ON c.id_curso=i.id_curso
44

```

Data Output Mensajes Notificaciones

SQL

	nombre character varying (100)	apellido character varying (100)	nombrcurso character varying (100)
1	Juan	Pérez	Programación Avanzada
2	Luis	Martínez	Gestión Empresarial
3	Juan	Pérez	Gestión Empresarial
4	Maria	González	Introducción al Derecho
5	Fernanda	Hernández	Programación Avanzada
6	Ricardo	Gómez	[null]
7	Natalia	Torres	[null]

```

29
30  --INNER JOIN: Obtén una lista que muestre el nombre del estudiante y el nombre del curso en el que está inscrito.
31 v SELECT e.Nombre, e.Apellido, c.NombreCurso
32 FROM Inscripciones i
33 INNER JOIN Estudiantes e ON e.id_estudiante= i.Id_estudiante
34 INNER JOIN Cursos c ON c.id_curso=i.id_curso
35
36
37

```

Data Output Mensajes Notificaciones

SQL

	nombre character varying (100)	apellido character varying (100)	nombrcurso character varying (100)
1	Juan	Pérez	Programación Avanzada
2	Luis	Martínez	Gestión Empresarial
3	Juan	Pérez	Gestión Empresarial
4	Maria	González	Introducción al Derecho
5	Fernanda	Hernández	Programación Avanzada

Consulta Historial de Consultas

```
52
53  --GROUP BY y COUNT: Calcula cuántos estudiantes están inscritos en cada curso.
54  --Muestra el nombre del curso y la cantidad de estudiantes.
55
56  SELECT c.NombreCurso, COUNT(i.id_estudiante) AS Cantidad_Estudiantes
57  FROM Inscripciones i
58  INNER JOIN Cursos c ON i.id_curso=c.id_curso
59  GROUP BY c.NombreCurso
60
```

Data Output Mensajes Notificaciones

	nombrecurso	cantidad_estudiantes
1	Introducción al Derecho	1
2	Programación Avanzada	2
3	Gestión Empresarial	2

```
44
45  --RIGHT JOIN: Lista todos los cursos y, si tienen estudiantes inscritos,
46  --el nombre de los estudiantes. Muestra todos los cursos,
47  --incluso si no tienen estudiantes inscritos actualmente.
48  SELECT e.Nombre, e.Apellido, c.NombreCurso
49  FROM Estudiantes e |
50  RIGHT JOIN Inscripciones i ON e.id_estudiante= i.id_estudiante
51  RIGHT JOIN Cursos c ON c.id_curso=i.id_curso
52
```

Data Output Mensajes Notificaciones

	nombre	apellido	nombrecurso
1	Juan	Pérez	Programación Avanzada
2	Luis	Martínez	Gestión Empresarial
3	Juan	Pérez	Gestión Empresarial
4	Maria	González	Introducción al Derecho
5	Fernanda	Hernández	Programación Avanzada
6	[null]	[null]	Diseño Arquitectónico
7	[null]	[null]	Psicología Social

```
68  --BETWEEN: Encuentra todos los estudiantes que nacieron entre el 1 de enero de 1995 y el 31 de diciembre de 1998.
69  SELECT e.Nombre, e.Apellido, e.FechaNac
70  FROM Estudiantes e
71  WHERE FechaNac BETWEEN '1995-01-01' AND '1998-12-31'
72
```

Data Output Mensajes Notificaciones

	nombre	apellido	fechanac
1	Juan	Pérez	1996-05-21
2	Maria	González	1997-04-17
3	Luis	Martínez	1998-10-24

```
72
73 --ORDER BY: Muestra todos los cursos ordenados alfabéticamente por su nombre.
74 SELECT NombreCurso FROM Cursos
75 ORDER BY NombreCurso;
```

Barra de herramientas: Data, Output, Mensajes, Notificaciones.

	nombrerecurso
	character varying (100) 
1	Diseño Arquitectónico
2	Gestión Empresarial
3	Introducción al Derecho
4	Programación Avanzada
5	Psicología Social

```
12
13
14 v WITH CursoMasInscrito AS (
15     SELECT i.id_curso, COUNT(*) AS total_inscripciones
16     FROM Inscripciones i
17     GROUP BY i.id_curso
18 )
19 SELECT d.NombreDepartamento, c.NombreCurso
20 FROM Departamentos d
21 JOIN Cursos c ON d.id_departamento = c.id_departamento
22 JOIN CursoMasInscrito ci ON c.id_curso = ci.id_curso
23 WHERE ci.total_inscripciones = (
24     SELECT MAX(total_inscripciones)
25     FROM CursoMasInscrito
26     WHERE id_curso IN (
27         SELECT id_curso FROM Cursos WHERE id_departamento = d.id_departamento
28     )
29 );
```

Data Output		Mensajes	Notificaciones
			
1	nombredepartamento character varying (100)	lock	nombrerecurso character varying (100)
2	Departamento de Ingenier...	Programación Avanzada	
2	Departamento de Negocios	Gestión Empresarial	

```
1 ✓ SELECT p.Nombre, p.Apellido, COUNT(cp.id_curso) AS cantidad_cursos
2 FROM Profesores p
3 JOIN Cursos_Profesores cp ON p.id_profesor = cp.id_profesor
4 GROUP BY p.id_profesor, p.Nombre, p.Apellido
5 HAVING COUNT(cp.id_curso) > 2;
6
```

Data Output		Mensajes	Notificaciones
      			
nombre	character varying (50)	apellido	character varying (100)
character varying (50)		character varying (100)	
		cantidad_cursos	bigint
			

```

56
57 WITH PromedioCursos AS (
58     SELECT id_curso, AVG(Calificacion) AS promedio_calificacion
59     FROM Inscripciones
60     GROUP BY id_curso
61 )
62 SELECT pe.NombrePrograma, c.NombreCurso
63 FROM ProgramasEstudio pe
64 JOIN Programa_Cursos pc ON pe.id_programa = pc.id_programa
65 JOIN Cursos c ON pc.id_curso = c.id_curso
66 JOIN PromedioCursos p ON c.id_curso = p.id_curso
67 WHERE p.promedio_calificacion = (
68     SELECT MAX(promedio_calificacion)
69     FROM PromedioCursos
70     WHERE id_curso IN (
71         SELECT id_curso FROM Programa_Cursos WHERE id_programa = pe.id_programa
72     )
73 );
74

```

Data Output Mensajes Notificaciones

	nombreprograma character varying (100)	nombrcurso character varying (100)
1	Ingeniería en Sistemas	Gestión Empresarial
2	Administración de Empres...	Introducción al Derecho

```

1 WITH InscripcionesPorEstudiante AS (
2     SELECT id_estudiante, COUNT(*) AS total_inscripciones
3     FROM Inscripciones
4     GROUP BY id_estudiante
5 )
6 SELECT e.Nombre, e.Apellido, i.total_inscripciones
7 FROM InscripcionesPorEstudiante i
8 JOIN Estudiantes e ON i.id_estudiante = e.id_estudiante
9 ORDER BY i.total_inscripciones DESC
10 LIMIT 3;

```

Data Output Mensajes Notificaciones

	nombre character varying (100)	apellido character varying (100)	total_inscripciones bigint
1	Juan	Pérez	2
2	Luis	Martínez	1
3	María	González	1