



Inteligencia Artificial: Qué es, Cómo Funciona y su Impacto en el Empleo

La Inteligencia Artificial (IA) ha pasado de ser un concepto de ciencia ficción a una herramienta tecnológica fundamental que moldea nuestra vida diaria. Entenderla es crucial para navegar el presente y el futuro del trabajo. El término "Inteligencia Artificial" fue acuñado formalmente en 1956 por John McCarthy en la Conferencia de Dartmouth, marcando el nacimiento del campo.

¿Qué es la Inteligencia Artificial?

En esencia, la Inteligencia Artificial es un campo de la informática dedicado a crear sistemas que pueden realizar tareas que normalmente requieren inteligencia humana. Estas tareas incluyen el aprendizaje, el razonamiento, la percepción visual, la comprensión del lenguaje natural y la toma de decisiones.

Es importante distinguir dos tipos de IA:

1. **IA Estrecha (ANI):** Es la única que poseemos actualmente. Está diseñada y entrenada para realizar una tarea específica (por ejemplo, el sistema de recomendación de Netflix, el motor de búsqueda de Google, o un software que juega ajedrez).
2. **IA General (AGI):** Es una IA teórica que poseería la capacidad de entender, aprender y aplicar su inteligencia para resolver cualquier problema, de forma similar a un ser humano. Aún no existe, y su viabilidad sigue siendo un tema de intenso debate científico.

¿Cómo Funciona la Inteligencia Artificial?

La IA no es una sola cosa; es un campo amplio. El motor detrás de la mayoría de los avances modernos de la IA es el **Aprendizaje Automático** (Machine Learning o ML).

En lugar de programar un software con reglas explícitas para cada situación (programación tradicional), el ML utiliza algoritmos para "entrenar" un modelo.

El proceso funciona así:

1. **Recolección de Datos:** Se alimenta al sistema con una enorme cantidad de datos. Para crear una IA que reconozca gatos, se le mostrarían millones de imágenes, etiquetando cuáles contienen un gato y cuáles no. La calidad y cantidad de estos datos son cruciales.
2. **Entrenamiento:** El modelo (el "cerebro" de la IA) ajusta sus parámetros internos repetidamente para encontrar patrones en los datos. No "memoriza" las fotos, sino que "aprende" las características estadísticas de cómo se ve un gato (forma de las orejas, textura del pelaje, etc.).
3. **Inferencia (Predicción):** Una vez entrenado, el modelo puede tomar datos nuevos que nunca ha visto (una foto nueva) y hacer una predicción precisa (decir "esto es un gato").

Deep Learning y IA Generativa

Una subcategoría avanzada del ML es el **Deep Learning** (Aprendizaje Profundo), que utiliza estructuras llamadas **redes neuronales artificiales**. Estas redes están inspiradas en la estructura del cerebro humano, con múltiples capas que procesan la información de forma cada vez más compleja.



[Imagen de un diagrama de red neuronal]

El Deep Learning es lo que impulsa las herramientas más impresionantes de hoy, como la **IA Generativa**. Modelos como GPT-3 (lanzado en 2020) fueron entrenados con 175 mil millones de parámetros, y modelos más recientes como GPT-4 operan a una escala aún mayor. Estos modelos se entran con la totalidad de internet (texto, código e imágenes) y aprenden patrones tan complejos que pueden *crear* contenido nuevo, coherente y original.

El Impacto de la IA en los Trabajos (Hasta Ahora)

Contrario a la creencia popular de un reemplazo masivo e inmediato, el impacto de la IA hasta la fecha ha sido más de **transformación y aumento** que de eliminación pura.

Según el informe "**Future of Jobs Report 2023**" del Foro Económico Mundial (WEF), se estima que mientras la IA y la digitalización desplazarán millones de empleos, también crearán millones de nuevos roles. El resultado neto es una **transformación de las habilidades** requeridas en el mercado laboral.

El impacto verificable se puede dividir en tres categorías:

1. Automatización de Tareas Rutinarias

La IA es extremadamente eficaz en automatizar tareas repetitivas y basadas en reglas, tanto físicas como cognitivas.

- **Ejemplos:** Análisis básico de datos financieros, entrada de datos (data entry), filtrado inicial de currículums (RRHH), atención al cliente de primer nivel (chatbots) y control de calidad visual en líneas de ensamblaje.
- **Datos Verificables:**
 - Un informe de **McKinsey Global Institute** ha estimado que las tecnologías actuales (incluyendo la IA) tienen el potencial de automatizar actividades que consumen entre el 40% y 50% del tiempo de los empleados. Es crucial notar que esto se refiere a *actividades*, no a *empleos completos*, sugiriendo una redefinición de los roles.
 - **Gartner**, una firma líder en investigación tecnológica, proyectó que para 2022 (una tendencia que se ha consolidado), los chatbots y asistentes virtuales manejarían un porcentaje significativo de las interacciones de servicio al cliente, reduciendo costos operativos.

2. Augmentación de Capacidades (Aumento de Productividad)

Este es el impacto más significativo y medible hasta ahora. La IA actúa como un "copiloto" que hace a los trabajadores más eficientes, permitiéndoles enfocarse en tareas de mayor valor.

- **Medicina:** Radiólogos usan IA para pre-analizar resonancias magnéticas y tomografías, detectando tumores con mayor precisión y velocidad.
- **Programación:** Herramientas como GitHub Copilot sugieren bloques enteros de código, acelerando el desarrollo.
- **Creatividad y Marketing:** Se utiliza IA generativa para proponer borradores iniciales de textos (copywriting), traducir contenido a múltiples idiomas o crear imágenes base, acelerando drásticamente el proceso creativo.
- **Datos Verificables:**
 - Un estudio de **GitHub sobre Copilot** (su herramienta de IA para programadores) encontró que los desarrolladores que la utilizaban completaban sus tareas de programación hasta un **55% más rápido** que el grupo de control.



- Un *working paper* de la **Oficina Nacional de Investigación Económica (NBER) de EE.UU.** en 2023, que analizó el impacto de un asistente de IA generativa en un entorno real de soporte técnico, registró un **aumento de productividad promedio del 14%**. Notablemente, los trabajadores más nuevos o con menos habilidades fueron los más beneficiados, cerrando la brecha de rendimiento.
- En medicina, estudios publicados en revistas como **The Lancet Digital Health** han mostrado que los algoritmos de IA pueden igualar, y en algunos casos superar, la precisión de radiólogos humanos experimentados en la detección de cáncer de mama. Sin embargo, la combinación de **IA + Radiólogo** ha demostrado ser la más precisa de todas, evidenciando el modelo de colaboración.

3. Creación de Nuevos Roles

La adopción de la IA ha creado una demanda de trabajos que no existían hace una década, centrados en construir, mantener y gestionar estas nuevas tecnologías.

- **Ingeniero de Prompts (Prompt Engineer):** Especialistas en diseñar las instrucciones (prompts) precisas para que las IAs generativas produzcan los resultados deseados.
- **Especialista en Ética de IA:** Profesionales que auditán y aseguran que los sistemas de IA sean justos, imparciales, transparentes y no perpetúen sesgos discriminatorios.
- **Entrenador de IA / Anotador de Datos:** Personas que preparan, limpian y etiquetan los vastos conjuntos de datos necesarios para entrenar a los modelos, un paso fundamental para su precisión.
- **Datos Verificables:**
- El ya mencionado informe "**Future of Jobs 2023**" del WEF identifica a los "**Especialistas en IA y Machine Learning**" como la categoría de empleo de más rápido crecimiento a nivel global.
- Durante 2023, plataformas de empleo como **LinkedIn** reportaron un crecimiento exponencial en las ofertas de trabajo que mencionaban "IA Generativa" o "Ingeniería de Prompts" en sus descripciones.

Conclusión Parcial

Hasta ahora, la IA no ha causado el desempleo masivo que a veces se predice. Su principal efecto ha sido **cambiar la naturaleza de los trabajos existentes**.

El "**AI Index Report 2023**" de la **Universidad de Stanford** subraya que la inversión privada en IA se está desplazando de la investigación pura a la aplicación práctica, lo que acelera esta integración laboral. El desafío actual para la fuerza laboral no es tanto competir *contra* la IA, sino aprender a **colaborar con ella**. El WEF estima que, debido a la IA y otras transiciones tecnológicas, más del 40% de las habilidades laborales básicas cambiarán en los próximos cinco años, destacando la urgencia de la recapacitación y el aprendizaje continuo.

Inteligencia Artificial: Qué es, Cómo Funciona y su Impacto en el Empleo

La Inteligencia Artificial (IA) ha pasado de ser un concepto de ciencia ficción a una herramienta tecnológica fundamental que moldea nuestra vida diaria. Entenderla es crucial para navegar el presente y el futuro del trabajo. El término "Inteligencia Artificial" fue acuñado formalmente en 1956 por John McCarthy en la Conferencia de Dartmouth, marcando el nacimiento del campo.

¿Qué es la Inteligencia Artificial?



En esencia, la Inteligencia Artificial es un campo de la informática dedicado a crear sistemas que pueden realizar tareas que normalmente requieren inteligencia humana. Estas tareas incluyen el aprendizaje, el razonamiento, la percepción visual, la comprensión del lenguaje natural y la toma de decisiones.

Es importante distinguir dos tipos de IA:

1. **IA Estrecha (ANI):** Es la única que poseemos actualmente. Está diseñada y entrenada para realizar una tarea específica (por ejemplo, el sistema de recomendación de Netflix, el motor de búsqueda de Google, o un software que juega ajedrez).
2. **IA General (AGI):** Es una IA teórica que poseería la capacidad de entender, aprender y aplicar su inteligencia para resolver cualquier problema, de forma similar a un ser humano. Aún no existe, y su viabilidad sigue siendo un tema de intenso debate científico.

¿Cómo Funciona la Inteligencia Artificial?

La IA no es una sola cosa; es un campo amplio. El motor detrás de la mayoría de los avances modernos de la IA es el **Aprendizaje Automático** (Machine Learning o ML).

En lugar de programar un software con reglas explícitas para cada situación (programación tradicional), el ML utiliza algoritmos para "entrenar" un modelo.

El proceso funciona así:

1. **Recolección de Datos:** Se alimenta al sistema con una enorme cantidad de datos. Para crear una IA que reconozca gatos, se le mostrarían millones de imágenes, etiquetando cuáles contienen un gato y cuáles no. La calidad y cantidad de estos datos son cruciales.
2. **Entrenamiento:** El modelo (el "cerebro" de la IA) ajusta sus parámetros internos repetidamente para encontrar patrones en los datos. No "memoriza" las fotos, sino que "aprende" las características estadísticas de cómo se ve un gato (forma de las orejas, textura del pelaje, etc.).
3. **Inferencia (Predicción):** Una vez entrenado, el modelo puede tomar datos nuevos que nunca ha visto (una foto nueva) y hacer una predicción precisa (decir "esto es un gato").

Deep Learning y IA Generativa

Una subcategoría avanzada del ML es el **Deep Learning** (Aprendizaje Profundo), que utiliza estructuras llamadas **redes neuronales artificiales**. Estas redes están inspiradas en la estructura del cerebro humano, con múltiples capas que procesan la información de forma cada vez más compleja.

[Imagen de un diagrama de red neuronal]

El Deep Learning es lo que impulsa las herramientas más impresionantes de hoy, como la **IA Generativa**. Modelos como GPT-3 (lanzado en 2020) fueron entrenados con 175 mil millones de parámetros, y modelos más recientes como GPT-4 operan a una escala aún mayor. Estos modelos se entranan con la totalidad de internet (texto, código e imágenes) y aprenden patrones tan complejos que pueden *crear* contenido nuevo, coherente y original.

El Impacto de la IA en los Trabajos (Hasta Ahora)

Contrario a la creencia popular de un reemplazo masivo e inmediato, el impacto de la IA hasta la fecha ha sido más de **transformación y aumento** que de eliminación pura.



Según el informe "**Future of Jobs Report 2023**" del Foro Económico Mundial (WEF), se estima que mientras la IA y la digitalización desplazarán millones de empleos, también crearán millones de nuevos roles. El resultado neto es una **transformación de las habilidades** requeridas en el mercado laboral.

El impacto verificable se puede dividir en tres categorías:

1. Automatización de Tareas Rutinarias

La IA es extremadamente eficaz en automatizar tareas repetitivas y basadas en reglas, tanto físicas como cognitivas.

- **Ejemplos:** Análisis básico de datos financieros, entrada de datos (data entry), filtrado inicial de currículums (RRHH), atención al cliente de primer nivel (chatbots) y control de calidad visual en líneas de ensamblaje.
- **Datos Verificables:**
 - Un informe de **McKinsey Global Institute** ha estimado que las tecnologías actuales (incluyendo la IA) tienen el potencial de automatizar actividades que consumen entre el 40% y 50% del tiempo de los empleados. Es crucial notar que esto se refiere a *actividades*, no a *empleos completos*, sugiriendo una redefinición de los roles.
 - **Gartner**, una firma líder en investigación tecnológica, proyectó que para 2022 (una tendencia que se ha consolidado), los chatbots y asistentes virtuales manejarían un porcentaje significativo de las interacciones de servicio al cliente, reduciendo costos operativos.

2. Augmentación de Capacidades (Aumento de Productividad)

Este es el impacto más significativo y medible hasta ahora. La IA actúa como un "copiloto" que hace a los trabajadores más eficientes, permitiéndoles enfocarse en tareas de mayor valor.

- **Medicina:** Radiólogos usan IA para pre-analizar resonancias magnéticas y tomografías, detectando tumores con mayor precisión y velocidad.
- **Programación:** Herramientas como GitHub Copilot sugieren bloques enteros de código, acelerando el desarrollo.
- **Creatividad y Marketing:** Se utiliza IA generativa para proponer borradores iniciales de textos (copywriting), traducir contenido a múltiples idiomas o crear imágenes base, acelerando drásticamente el proceso creativo.
- **Datos Verificables:**
 - Un estudio de **GitHub sobre Copilot** (su herramienta de IA para programadores) encontró que los desarrolladores que la utilizaban completaban sus tareas de programación hasta un **55% más rápido** que el grupo de control.
 - Un *working paper* de la **Oficina Nacional de Investigación Económica (NBER) de EE.UU.** en 2023, que analizó el impacto de un asistente de IA generativa en un entorno real de soporte técnico, registró un **aumento de productividad promedio del 14%**. Notablemente, los trabajadores más nuevos o con menos habilidades fueron los más beneficiados, cerrando la brecha de rendimiento.
 - En medicina, estudios publicados en revistas como **The Lancet Digital Health** han mostrado que los algoritmos de IA pueden igualar, y en algunos casos superar, la precisión de radiólogos humanos experimentados en la detección de cáncer de mama. Sin embargo, la combinación de **IA + Radiólogo** ha demostrado ser la más precisa de todas, evidenciando el modelo de colaboración.

3. Creación de Nuevos Roles

La adopción de la IA ha creado una demanda de trabajos que no existían hace una década, centrados en construir, mantener y gestionar estas nuevas tecnologías.



- **Ingeniero de Prompts (Prompt Engineer):** Especialistas en diseñar las instrucciones (prompts) precisas para que las IAs generativas produzcan los resultados deseados.
- **Especialista en Ética de IA:** Profesionales que auditán y aseguran que los sistemas de IA sean justos, imparciales, transparentes y no perpetúen sesgos discriminatorios.
- **Entrenador de IA / Anotador de Datos:** Personas que preparan, limpian y etiquetan los vastos conjuntos de datos necesarios para entrenar a los modelos, un paso fundamental para su precisión.
- **Datos Verificables:**
 - El ya mencionado informe "**Future of Jobs 2023**" del WEF identifica a los "**Especialistas en IA y Machine Learning**" como la categoría de empleo de más rápido crecimiento a nivel global.
 - Durante 2023, plataformas de empleo como **LinkedIn** reportaron un crecimiento exponencial en las ofertas de trabajo que mencionaban "IA Generativa" o "Ingeniería de Prompts" en sus descripciones.

Conclusión Parcial

Hasta ahora, la IA no ha causado el desempleo masivo que a veces se predice. Su principal efecto ha sido **cambiar la naturaleza de los trabajos existentes**.

El "**AI Index Report 2023**" de la **Universidad de Stanford** subraya que la inversión privada en IA se está desplazando de la investigación pura a la aplicación práctica, lo que acelera esta integración laboral. El desafío actual para la fuerza laboral no es tanto competir *contra* la IA, sino aprender a **colaborar con ella**. El WEF estima que, debido a la IA y otras transiciones tecnológicas, más del 40% de las habilidades laborales básicas cambiarán en los próximos cinco años, destacando la urgencia de la recapacitación y el aprendizaje continuo.

Laboratorio de:

Materia: Fundamentos de Bases de Datos

Práctica No.: LABORATORIO PRÁCTICO - TÓPICO 4

Tema: ESTRUCTURA DE UNA BD RELACIONAL - LENGUAJE DDL

TABLA DE CONTENIDOS

1. [Objetivos](#)
2. [Requisitos](#)
3. [Caso de Estudio](#)
4. [Desarrollo Paso a Paso](#)
5. [Script Completo Oracle](#)

OBJETIVOS

Objetivo General

Comprender y aplicar los comandos del Lenguaje de Definición de Datos (DDL) para crear, modificar y eliminar objetos en una base de datos Oracle, estableciendo correctamente las estructuras, restricciones e integridad referencial.



Objetivos Específicos

1. **Crear usuarios y esquemas** en Oracle con permisos apropiados
2. **Definir tablas** con tipos de datos Oracle apropiados
3. **Implementar restricciones de integridad** (PRIMARY KEY, FOREIGN KEY, UNIQUE, CHECK, NOT NULL, DEFAULT)
4. **Modificar estructuras existentes** mediante ALTER TABLE
5. **Crear y gestionar índices** para optimización de consultas
6. **Utilizar secuencias** para auto-incremento
7. **Eliminar objetos** de base de datos de forma segura (DROP, TRUNCATE)

REQUISITOS

Marco teórico:

Conceptos de normalización (1FN, 2FN, 3FN)

Forma Normal	Requisito Teórico Principal	Problema que Resuelve	Dependencia Funcional que Elimina
1NF	Los dominios de todos los atributos deben ser atómicos .	Grupos repetidos, valores multivaluados y atributos anidados.	(No aplica a DF, sino a la estructura del atributo).
2NF	- Estar en 1NF + - Todo atributo no-clave debe tener una dependencia funcional completa de la clave primaria.	Dependencias Parciales (un atributo no-clave depende de <i>parte</i> de una clave primaria compuesta).	Dependencias parciales.
3NF	- Estar en 2NF + - Ningún atributo no-clave puede tener una dependencia transitiva de la clave primaria.	Dependencias Transitivas (un atributo no-clave depende de <i>otro</i> atributo no-clave: PK -> NoClave_A -> NoClave_B).	Dependencias transitivas.

Modelo Entidad-Relación

Modelo conceptual (Entidad – Relación)

Su función principal es representar las entidades del dominio del negocio, contado con: sus atributos esenciales y relaciones entre los atributos, sin preocuparse sobre la tecnología o detalles de implementación.

El fin de esta función es reflejar el “mundo real” la situación en la que los usuarios y analistas estén de acuerdo con el modelo, siendo el puente entre los desarrolladores y el cliente.

Para representar este modelo se utilizan diagramas ER (Entity-Relationship) .



Conceptos Clave de Bases de Datos

A continuación, se detallan los conceptos que solicitaste.

Cardinalidad de Relaciones

La cardinalidad describe la relación numérica entre dos entidades en una base de datos.

Cardinalidad	Nombre	Descripción	Ejemplo
1:1	Uno a Uno	Una instancia de la Entidad A solo puede estar relacionada con una instancia de la Entidad B, y viceversa.	Usuario y PerfilDeUsuario. Un usuario tiene un solo perfil, y un perfil pertenece a un solo usuario.
1:N	Uno a Muchos	Una instancia de la Entidad A puede estar relacionada con muchas instancias de la Entidad B, pero una instancia de la Entidad B solo puede estar relacionada con una instancia de la Entidad A.	Cliente y Pedido. Un cliente puede tener muchos pedidos, pero un pedido pertenece a un solo cliente.
N:N	Muchos a Muchos	Una instancia de la Entidad A puede estar relacionada con muchas instancias de la Entidad B, y viceversa. Este tipo de relación generalmente requiere una tabla intermedia (tabla de unión) para implementarse.	Estudiante y Curso. Un estudiante puede inscribirse en muchos cursos, y un curso puede tener muchos estudiantes.

Claves Primarias y Foráneas

Clave Primaria (Primary Key - PK)

Es una columna (o conjunto de columnas) que identifica de forma **única** cada fila en una tabla. No puede contener valores nulos (NULL) y sus valores no deben repetirse en toda la tabla. Es el identificador principal de un registro.

Clave Foránea (Foreign Key - FK)

Es una columna (o conjunto de columnas) en una tabla que establece un **enlace** con la clave primaria de otra tabla (o la misma). Se utiliza para mantener la integridad referencial de los datos, asegurando que un valor en la tabla "hija" exista en la tabla "padre".

Tipos de Datos Básicos

Tipo de Dato	Descripción	Ejemplos
INT (o INTEGER)	Números enteros, sin decimales.	1, 100, -45
VARCHAR(n)	Cadena de texto de longitud variable, donde n es el máximo número de caracteres.	"Hola", "Juan Pérez"



CHAR(n)	Cadena de texto de longitud fija, donde n es el número exacto de caracteres.	"S", "MX"
DECIMAL(p, s) (o NUMERIC)	Números exactos con decimales. p es la precisión (total de dígitos) y s es la escala (dígitos después del punto decimal).	123.45, 99.99
FLOAT (o REAL)	Números de punto flotante (decimales aproximados).	10.5e3
DATE	Almacena una fecha (Año, Mes, Día).	2025-11-13
DATETIME (o TIMESTAMP)	Almacena una fecha y hora (Año, Mes, Día, Hora, Minuto, Segundo).	2025-11-13 23:30:00
BOOLEAN (o BIT)	Almacena valores de verdadero o falso.	true, false, 1, 0
TEXT (o CLOB)	Almacena cadenas de texto de gran longitud.	Un artículo de blog, una descripción larga.
BLOB	Almacena datos binarios de gran tamaño (Binary Large Object).	Imágenes, archivos de audio, PDFs.

Requisitos Técnicos

Software requerido:

- Oracle Database 11g o superior
- SQL Developer o SQL*Plus
- Acceso con privilegios SYSDBA o DBA

Material de Apoyo

- Diagrama ER del caso de estudio
- Diccionario de datos
- Documentación de sintaxis DDL de Oracle

CASO DE ESTUDIO

Sistema de Gestión Académica Universitaria

Descripción del Dominio

La Escuela Politécnica Nacional requiere un sistema para gestionar la información académica de sus estudiantes, docentes, carreras y asignaturas. El sistema debe permitir:

- Registrar información de estudiantes matriculados en diferentes carreras
- Gestionar el catálogo de asignaturas organizadas por carrera y nivel
- Asignar docentes a asignaturas en diferentes períodos académicos
- Registrar las matrículas de estudiantes en asignaturas
- Almacenar las calificaciones finales de cada estudiante
- Controlar requisitos entre asignaturas



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

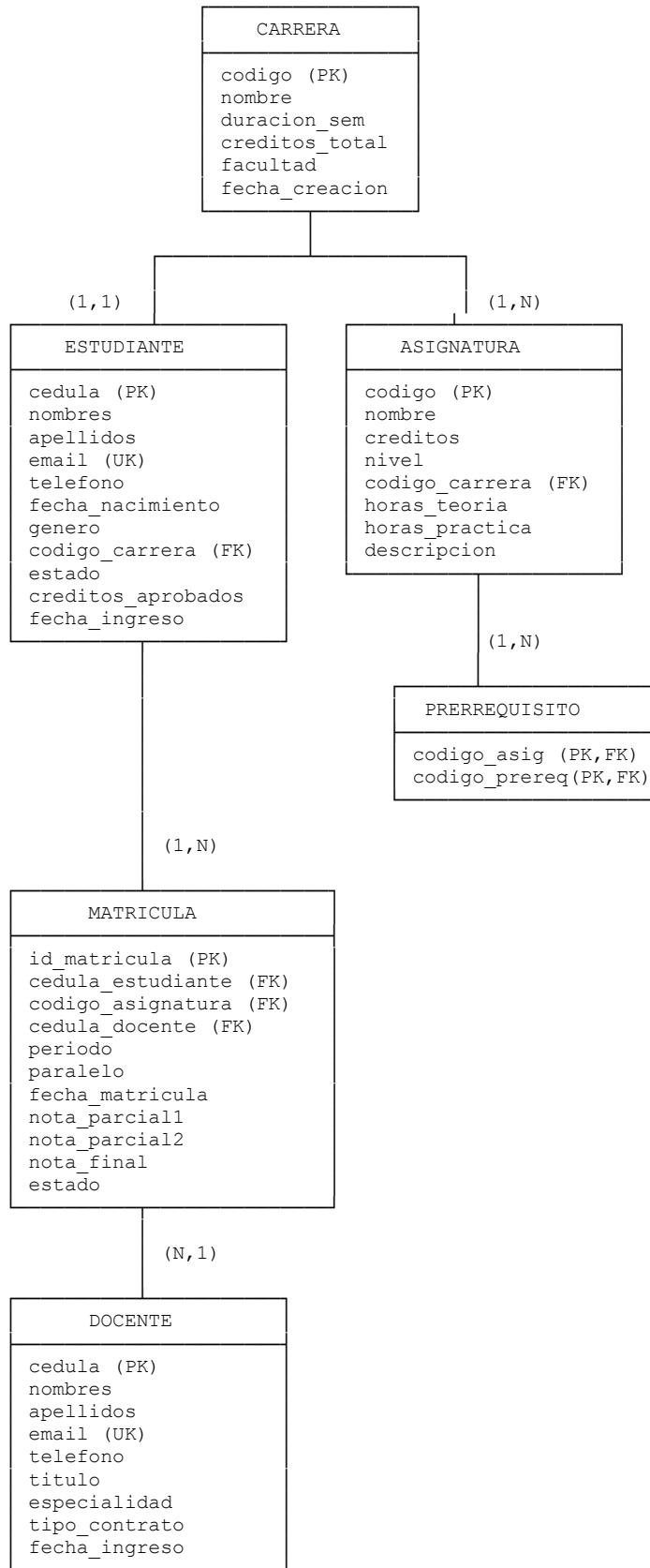
Reglas de Negocio

1. Cada estudiante pertenece a una única carrera
2. Una asignatura pertenece a una carrera específica y tiene un nivel definido
3. Una asignatura puede tener múltiples prerrequisitos
4. Un docente puede impartir múltiples asignaturas en diferentes períodos
5. Un estudiante puede matricularse en múltiples asignaturas por período
6. Las notas finales deben estar entre 0 y 10
7. Se considera aprobada una asignatura con nota ≥ 7.0
8. Los estados de matrícula válidos son: CURSANDO, APROBADO, REPROBADO, RETIRADO
9. Un estudiante no puede matricularse dos veces en la misma asignatura en el mismo período
10. El período académico tiene formato: YYYY-NS (ejemplo: 2024-1S, 2024-2S)

DIAGRAMA ENTIDAD-RELACIÓN



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS





Leyenda:

- PK = Primary Key (Clave Primaria)
FK = Foreign Key (Clave Foránea)
UK = Unique Key (Clave Única)
(1,1) = Cardinalidad uno a uno
(1,N) = Cardinalidad uno a muchos
(N,M) = Cardinalidad muchos a muchos

DICCIONARIO DE DATOS

Tabla: CARRERA

Campo	Tipo de Dato Oracle	Restricciones	Descripción
codigo	VARCHAR2(10)	PK, NOT NULL	Código único de carrera (ej: ING-SIS)
nombre	VARCHAR2(100)	NOT NULL, UNIQUE	Nombre completo de la carrera
duracion_semestres	NUMBER(2)	NOT NULL, CHECK (8-12)	Duración en semestres
creditos_totales	NUMBER(3)	NOT NULL, CHECK (>0)	Total de créditos para graduarse
facultad	VARCHAR2(100)	NOT NULL	Facultad a la que pertenece
fecha_creacion	DATE	DEFAULT SYSDATE	Fecha de creación del registro

Tabla: ESTUDIANTE

Campo	Tipo de Dato Oracle	Restricciones	Descripción
cedula	VARCHAR2(10)	PK, NOT NULL, CHECK (10 dígitos)	Cédula de identidad ecuatoriana
nombres	VARCHAR2(50)	NOT NULL	Nombres del estudiante
apellidos	VARCHAR2(50)	NOT NULL	Apellidos del estudiante
email	VARCHAR2(100)	NOT NULL, UNIQUE	Correo institucional
telefono	VARCHAR2(15)	NULL	Teléfono de contacto
fecha_nacimiento	DATE	NOT NULL, CHECK (edad>=16)	Fecha de nacimiento
genero	CHAR(1)	NOT NULL, CHECK ('M', 'F', 'O')	Género del estudiante
codigo_carrera	VARCHAR2(10)	FK → Carrera, NOT NULL	Carrera del estudiante
estado	VARCHAR2(20)	DEFAULT 'ACTIVO', CHECK	Estado actual del estudiante
creditos_aprobados	NUMBER(3)	DEFAULT 0, CHECK (>=0)	Créditos acumulados
fecha_ingreso	DATE	DEFAULT SYSDATE	Fecha de ingreso a la EPN

Valores válidos para estado: ACTIVO, INACTIVO, GRADUADO, RETIRADO

Tabla: ASIGNATURA

Campo	Tipo de Dato Oracle	Restricciones	Descripción
codigo	VARCHAR2(10)	PK, NOT NULL	Código único (ej: BD-101)
nombre	VARCHAR2(100)	NOT NULL	Nombre de la asignatura



creditos	NUMBER(1)	NOT NULL, CHECK (1-8)	Número de créditos
nivel	NUMBER(2)	NOT NULL, CHECK (1-10)	Nivel o semestre
codigo_carrera	VARCHAR2(10)	FK → Carrera, NOT NULL	Carrera a la que pertenece
horas_teoria	NUMBER(2)	DEFAULT 0, NOT NULL	Horas teóricas semanales
horas_practica	NUMBER(2)	DEFAULT 0, NOT NULL	Horas prácticas semanales
descripcion	CLOB	NULL	Descripción y objetivos

Restricción adicional: horas_teoria + horas_practica > 0

Tabla: DOCENTE

Campo	Tipo de Dato Oracle	Restricciones	Descripción
cedula	VARCHAR2(10)	PK, NOT NULL, CHECK (10 dígitos)	Cédula de identidad
nombres	VARCHAR2(50)	NOT NULL	Nombres del docente
apellidos	VARCHAR2(50)	NOT NULL	Apellidos del docente
email	VARCHAR2(100)	NOT NULL, UNIQUE	Correo institucional
telefono	VARCHAR2(15)	NULL	Teléfono de contacto
titulo	VARCHAR2(50)	NOT NULL	Título académico máximo
especialidad	VARCHAR2(100)	NULL	Área de especialización
tipo_contrato	VARCHAR2(20)	DEFAULT 'TIEMPO_COMPLETO'	Tipo de contrato
fecha_ingreso	DATE	DEFAULT SYSDATE	Fecha de ingreso

Valores válidos para tipo_contrato: TIEMPO_COMPLETO, MEDIO_TIEMPO, HORA_CLASE

Tabla: MATRICULA

Campo	Tipo de Dato Oracle	Restricciones	Descripción
id_matricula	NUMBER(10)	PK, SEQUENCE	ID único generado
cedula_estudiante	VARCHAR2(10)	FK → Estudiante, NOT NULL	Estudiante matriculado
codigo_asignatura	VARCHAR2(10)	FK → Asignatura, NOT NULL	Asignatura matriculada
cedula_docente	VARCHAR2(10)	FK → Docente, NOT NULL	Docente asignado
periodo	VARCHAR2(10)	NOT NULL, CHECK (formato)	Periodo académico
paralelo	CHAR(1)	NOT NULL, CHECK (A-Z)	Paralelo
fecha_matricula	DATE	DEFAULT SYSDATE	Fecha de matrícula
nota_parcial1	NUMBER(4,2)	NULL, CHECK (0-10)	Primera nota parcial
nota_parcial2	NUMBER(4,2)	NULL, CHECK (0-10)	Segunda nota parcial
nota_final	NUMBER(4,2)	NULL, CHECK (0-10)	Nota final
estado	VARCHAR2(20)	DEFAULT 'CURSANDO'	Estado de la matrícula

Restricciones adicionales:

- UNIQUE (cedula_estudiante, codigo_asignatura, periodo)
- Si estado='CURSANDO' → nota_final IS NULL
- Si estado IN ('APROBADO','REPROBADO') → nota_final IS NOT NULL
- Formato periodo: ^\d{4}-[12]\\$ (ej: 2024-1S)

Tabla: PRERREQUISITO



Campo	Tipo de Dato Oracle	Restricciones	Descripción
codigo_asignatura	VARCHAR2(10)	PK, FK → Asignatura	Asignatura que requiere
codigo_prerrequisito	VARCHAR2(10)	PK, FK → Asignatura	Asignatura prerrequisito

Restricción adicional: codigo_asignatura ≠ codigo_prerrequisito

DESARROLLO DE LA PRACTICA

PASO 1: Crear Usuario y Esquema en Oracle

```
-- =====
-- CREACIÓN DE USUARIO/ESQUEMA - ORACLE
-- =====
-- Conectarse como SYSDBA
-- sqlplus / as sysdba

-- Paso 1.1: Verificar y eliminar usuario si existe
DROP USER gestion_academica CASCADE;
```

```
SQL> DROP USER gestion_academica CASCADE;
DROP USER gestion_academica CASCADE
      *
ERROR en lÍnea 1:
ORA-01918: el usuario 'GESTION_ACADEMICA' no existe
```

En primera instancia no existe

-- Paso 1.2: Crear nuevo usuario

```
CREATE USER gestion_academica IDENTIFIED BY EPN2024Secure
DEFAULT TABLESPACE USERS
TEMPORARY TABLESPACE TEMP
PROFILE DEFAULT;
```

```
SQL> show user
USER es "SYSTEM"
SQL> CREATE USER gestion_academica IDENTIFIED BY EPN2024Secure
  2  DEFAULT TABLESPACE USERS
  3  TEMPORARY TABLESPACE TEMP
  4  PROFILE DEFAULT;
```



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

-- Paso 1.3: Otorgar privilegios de sistema

```
GRANT CONNECT TO gestion_academica;  
GRANT RESOURCE TO gestion_academica;  
GRANT CREATE VIEW TO gestion_academica;  
GRANT CREATE SEQUENCE TO gestion_academica;  
GRANT CREATE SYNONYM TO gestion_academica;  
GRANT CREATE TRIGGER TO gestion_academica;  
GRANT CREATE PROCEDURE TO gestion_academica;
```



```
SQL> GRANT CONNECT TO gestion_academica;  
Concesión terminada correctamente.  
  
SQL> GRANT RESOURCE TO gestion_academica;  
Concesión terminada correctamente.  
  
SQL> GRANT CREATE VIEW TO gestion_academica;  
Concesión terminada correctamente.  
  
SQL> GRANT CREATE SEQUENCE TO gestion_academica;  
Concesión terminada correctamente.  
  
SQL> GRANT CREATE SYNONYM TO gestion_academica;  
Concesión terminada correctamente.  
  
SQL> GRANT CREATE TRIGGER TO gestion_academica;  
Concesión terminada correctamente.  
  
SQL> GRANT CREATE PROCEDURE TO gestion_academica;  
Concesión terminada correctamente.
```

-- Paso 1.4: Otorgar cuota en tablespace

ALTER USER gestion_academica QUOTA UNLIMITED ON USERS;

```
SQL> ALTER USER gestion_academica QUOTA UNLIMITED ON USERS;  
SQL>
```



-- Paso 1.5: Verificar usuario creado

```
SELECT username, account_status, default_tablespace, created
FROM dba_users
WHERE username = 'GESTION_ACADEMICA';
```

```
SQL> SELECT username, account_status, default_tablespace, created
  FROM dba_users
 2 WHERE username = 'GESTION_ACADEMICA';
```

USERNAME

ACCOUNT_STATUS	DEFAULT_TABLESPACE	C
CREATED		
GESTION_ACADEMICA	USERS	0
OPEN		
7/11/25		

-- Paso 1.6: Conectarse como el nuevo usuario

```
CONN gestion_academica/EPN2024Secure;
```

```
SQL> CONN gestion_academica/EPN2024Secure;
Conectado.
```

-- Paso 1.7: Verificar conexión actual

```
SELECT USER FROM DUAL;
SELECT SYS_CONTEXT('USERENV', 'CURRENT_SCHEMA') AS esquema_actual FROM DUAL;
```



```
SQL> SELECT USER FROM DUAL;

USER
-----
-----
GESTION_ACADEMICA

SQL> SELECT SYS_CONTEXT('USERENV', 'CURRENT_SCHEMA') AS esquema_
actual FROM DUAL;

ESQUEMA_ACTUAL
-----
-----
GESTION_ACADEMICA
```

PASO 2: Crear Tabla CARRERA



```
-- =====
-- TABLA CARRERA - ORACLE
-- =====

-- Eliminar tabla si existe (para pruebas)
BEGIN
    EXECUTE IMMEDIATE 'DROP TABLE Carrera CASCADE CONSTRAINTS';
EXCEPTION
    WHEN OTHERS THEN
        IF SQLCODE != -942 THEN RAISE; END IF;
END;
/
SQL> BEGIN
  2 EXECUTE IMMEDIATE 'DROP TABLE Carrera CASCADE CONSTRAINTS';
EXCEPTION
  3 WHEN OTHERS THEN
  4 IF SQLCODE != -942 THEN RAISE; END IF; END;
  5 /
```

Procedimiento PL/SQL terminado correctamente.

```
-- Crear tabla CARRERA
CREATE TABLE Carrera (
    codigo VARCHAR2(10) NOT NULL,
    nombre VARCHAR2(100) NOT NULL,
    duracion_semestres NUMBER(2) NOT NULL,
    creditos_totales NUMBER(3) NOT NULL,
    facultad VARCHAR2(100) NOT NULL,
    fecha_creacion DATE DEFAULT SYSDATE NOT NULL,
    -- Restricción de clave primaria
    CONSTRAINT pk_carrera PRIMARY KEY (codigo),
    -- Restricción de unicidad
    CONSTRAINT uk_carrera_nombre UNIQUE (nombre),
    -- Restricciones de validación
    CONSTRAINT chk_carrera_duracion
        CHECK (duracion_semestres BETWEEN 8 AND 12),
    CONSTRAINT chk_carrera_creditos
        CHECK (creditos_totales > 0)
```



);

```
SQL> CREATE TABLE Carrera (
  2  codigo VARCHAR2(10) NOT NULL, nombre VARCHAR2(100) NOT NULL
  '
  3  duracion_semestres NUMBER(2) NOT NULL, creditos_totales NUM
BER(3) NOT NULL, facultad VARCHAR2(100) NOT NULL,
  4  fecha_creacion DATE DEFAULT SYSDATE NOT NULL,
  5  -- Restricción de clave primaria
  6  CONSTRAINT pk_carrera PRIMARY KEY (codigo),
  7  -- Restricción de unicidad
  8  CONSTRAINT uk_carrera_nombre UNIQUE (nombre),
  9  -- Restricciones de validación
 10 CONSTRAINT chk_carrera_duracion
 11 CHECK (duracion_semestres BETWEEN 8 AND 12),
 12 CONSTRAINT chk_carrera_creditos CHECK (creditos_totales > 0
)
 13 );
```

Tabla creada.

-- Agregar comentarios a la tabla

COMMENT ON TABLE Carrera IS

'Catálogo de carreras ofertadas por la institución';

COMMENT ON COLUMN Carrera.codigo IS

'Código único de la carrera (ej: ING-SIS, ING-CIV)';

COMMENT ON COLUMN Carrera.nombre IS



'Nombre completo de la carrera';

COMMENT ON COLUMN Carrera.duracion_semestres **IS**

'Duración de la carrera en semestres académicos';

COMMENT ON COLUMN Carrera.creditos_totales **IS**

'Total de créditos necesarios para graduarse';

COMMENT ON COLUMN Carrera.facultad **IS**

'Facultad a la que pertenece la carrera';

```
SQL> COMMENT ON TABLE Carrera IS
  2  'Catálogo de carreras ofertadas por la institución';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Carrera.codigo IS
  2  'Código único de la carrera (ej: ING-SIS, ING-CIV)'; COMMENT ON COLUMN Carrera.nombre IS
  3
```

```
SQL> 'Nombre completo de la carrera';
```

```
SP2-0734: inicio "'Nombre co..." de comando desconocido - resto de la lÍnea ignorado.
```

```
SQL> COMMENT ON COLUMN Carrera.duracion_semestres IS 'Duración de la carrera en semestres académicos';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Carrera.creditos_totales IS 'Total de créditos necesarios para graduarse';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Carrera.facultad IS 'Facultad a la que pertenece la carrera';
```

Comentario creado.

-- Verificar creación

DESC Carrera;



```
SQL> DESC Carrera;
  Nombre          Null?    Tipo
-----  -----
CODOGO           NOT NULL VARCHAR2(10)
NOMBRE           NOT NULL VARCHAR2(100)
)
DURACION_SEMESTRES      NOT NULL NUMBER(2)
CREDITOS_TOTALES      NOT NULL NUMBER(3)
FACULTAD          NOT NULL VARCHAR2(100)
)
FECHA_CREACION      NOT NULL DATE
```

```
SQL>
```

-- Verificar restricciones

```
SELECT constraint_name, constraint_type, search_condition
FROM user_constraints
WHERE table_name = 'CARRERA'
ORDER BY constraint_type;
```

CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1 SYS_C007671	C	"NOMBRE" IS NOT NULL
2 SYS_C007672	C	"DURACION_SEMESTRES" IS NOT NULL
3 SYS_C007673	C	"CREDITOS_TOTALES" IS NOT NULL
4 SYS_C007670	C	"CODOGO" IS NOT NULL
5 SYS_C007674	C	"FACULTAD" IS NOT NULL
6 CHK_CARRERA_CREDITOS	C	creditos_totales > 0
7 SYS_C007675	C	"FECHA_CREACION" IS NOT NULL
8 CHK_CARRERA_DURACION	C	duracion_semestres BETWEEN 8 AND 12
9 PK_CARRERA	P	(null)
10 UK_CARRERA_NOMBRE	U	(null)

PASO 3: Crear Tabla ESTUDIANTE

```
-- =====
-- TABLA ESTUDIANTE - ORACLE
```



```
-- ======  
  
-- Eliminar tabla si existe  
BEGIN  
    EXECUTE IMMEDIATE 'DROP TABLE Estudiante CASCADE CONSTRAINTS';  
EXCEPTION  
    WHEN OTHERS THEN  
        IF SQLCODE != -942 THEN RAISE; END IF;  
END;  
/  
  
-- Crear tabla ESTUDIANTE  
CREATE TABLE Estudiante (  
    cedula VARCHAR2(10) NOT NULL,  
    nombres VARCHAR2(50) NOT NULL,  
    apellidos VARCHAR2(50) NOT NULL,  
    email VARCHAR2(100) NOT NULL,  
    telefono VARCHAR2(15),  
    fecha_nacimiento DATE NOT NULL,  
    genero CHAR(1) NOT NULL,  
    codigo_carrera VARCHAR2(10) NOT NULL,  
    estado VARCHAR2(20) DEFAULT 'ACTIVO' NOT NULL,  
    creditos_aprobados NUMBER(3) DEFAULT 0 NOT NULL,  
    fecha_ingreso DATE DEFAULT SYSDATE NOT NULL,  
    -- Clave primaria  
    CONSTRAINT pk_estudiante PRIMARY KEY (cedula),  
    -- Unicidad de email  
    CONSTRAINT uk_estudiante_email UNIQUE (email),  
    -- Validación de cédula (10 dígitos ecuatorianos)  
    CONSTRAINT chk_estudiante_cedula  
        CHECK (REGEXP_LIKE(cedula, '^\\d{10}$')),  
    -- Validación de género  
    CONSTRAINT chk_estudiante_genero  
        CHECK (genero IN ('M', 'F', 'O')),  
    -- Validación de estado  
    CONSTRAINT chk_estudiante_estado  
        CHECK (estado IN ('ACTIVO', 'INACTIVO', 'GRADUADO', 'RETIRADO')),  
    -- Validación de créditos
```



CONSTRAINT chk_estudiante_creditos

 CHECK (creditos_aprobados >= 0 AND creditos_aprobados <= 300),
-- Validación de edad (mayor de 16 años) a través de trigger
-- Clave foránea a Carrera

CONSTRAINT fk_estudiante_carrera

 FOREIGN KEY (codigo_carrera)
 REFERENCES Carrera(codigo));

```
4  telefono VARCHAR2(15), fecha_nacimiento DATE NOT NULL, gene
ro CHAR(1) NOT NULL,
5  codigo_carrera VARCHAR2(10) NOT NULL,
6  estado VARCHAR2(20) DEFAULT 'ACTIVO' NOT NULL,
7  creditos_aprobados NUMBER(3) DEFAULT 0 NOT NULL, fecha_ingr
eso DATE DEFAULT SYSDATE NOT NULL,
8  -- Clave primaria
9  CONSTRAINT pk_estudiante PRIMARY KEY (cedula),
10 -- Unicidad de email
11 CONSTRAINT uk_estudiante_email UNIQUE (email),
12 -- Validación de cédula (10 dígitos ecuatorianos)
13 CONSTRAINT chk_estudiante_cedula
14 CHECK (REGEXP_LIKE(cedula, '^\\d{10}$$')),
15 -- Validación de género
16 CONSTRAINT chk_estudiante_genero CHECK (genero IN ('M', 'F'
, 'O')),
17 -- Validación de estado
18 CONSTRAINT chk_estudiante_estado
19 CHECK (estado IN ('ACTIVO', 'INACTIVO', 'GRADUADO', 'RETIRA
DO')),
20 -- Validación de créditos
21 CONSTRAINT chk_estudiante_creditos
22 CHECK (creditos_aprobados >= 0 AND creditos_aprobados <= 30
0),
23 -- Validación de edad (mayor de 16 años) a través de trigge
r
24 -- Clave foránea a Carrera
25 CONSTRAINT fk_estudiante_carrera FOREIGN KEY (codigo_carrer
a) REFERENCES Carrera(codigo) );
```

Tabla creada.



-- Índices para optimizar búsquedas frecuentes

```
CREATE INDEX idx_estudiante_apellidos  
    ON Estudiante(apellidos);
```

```
CREATE INDEX idx_estudiante_carrera  
    ON Estudiante(codigo_carrera);
```

```
CREATE INDEX idx_estudiante_estado  
    ON Estudiante(estado);
```

```
SQL> CREATE INDEX idx_estudiante_apellidos ON Estudiante(apellidos);
```

```
=ndice creado.
```

```
SQL>  
SQL> CREATE INDEX idx_estudiante_carrera ON Estudiante(codigo_carrera);
```

```
=ndice creado.
```

```
SQL>  
SQL> CREATE INDEX idx_estudiante_estado ON Estudiante(estado);
```

```
=ndice creado.
```

-- Índice compuesto para búsquedas combinadas

```
CREATE INDEX idx_estudiante_apellidos_carrera  
    ON Estudiante(apellidos, codigo_carrera);
```

```
SQL>  
SQL> CREATE INDEX idx_estudiante_estado ON Estudiante(estado);
```

```
=ndice creado.
```

```
SQL> CREATE INDEX idx_estudiante_apellidos_carrera ON Estudiante(apellidos, codigo_carrera);
```

```
=ndice creado.
```

-- Agregar comentarios



COMMENT ON TABLE Estudiante IS

'Información personal y académica de estudiantes matriculados en la institución';

COMMENT ON COLUMN Estudiante.cedula IS

'Cédula de identidad ecuatoriana (10 dígitos)';

COMMENT ON COLUMN Estudiante.email IS

'Correo electrónico institucional único';

COMMENT ON COLUMN Estudiante.creditos_aprobados IS

'Total de créditos acumulados hasta la fecha';

COMMENT ON COLUMN Estudiante.estado IS

'Estado actual del estudiante: ACTIVO, INACTIVO, GRADUADO, RETIRADO';

```
SQL> COMMENT ON TABLE Estudiante IS
```

```
2 'Información personal y académica de estudiantes matriculados en la institución';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Estudiante.cedula IS 'Cédula de identidad ecuatoriana (10 dígitos)';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Estudiante.email IS 'Correo electrónico institucional único';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Estudiante.creditos_aprobados IS 'Total de créditos acumulados hasta la fecha';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Estudiante.estado IS
```

```
2 'Estado actual del estudiante: ACTIVO, INACTIVO, GRADUADO, RETIRADO' ;
```

Comentario creado.

```
SQL>
```



-- Validar que la edad sea al menos 16 años

```
CREATE OR REPLACE TRIGGER trg_validar_edad
BEFORE INSERT OR UPDATE ON Estudiante
FOR EACH ROW
BEGIN
    -- Validar que la edad sea al menos 16 años
    IF MONTHS_BETWEEN(SYSDATE, :NEW.fecha_nacimiento) / 12 < 16 THEN
        RAISE_APPLICATION_ERROR(-20001, 'El estudiante debe tener al menos 16 años.');
    END IF;
END;
/
```

```
SQL> CREATE OR REPLACE TRIGGER trg_validar_edad
  2  BEFORE INSERT OR UPDATE ON Estudiante
  3  FOR EACH ROW
  4  BEGIN
  5      -- Validar que la edad sea al menos 16 años
  6      IF MONTHS_BETWEEN(SYSDATE, :NEW.fecha_nacimiento) / 12 <
  16 THEN
  7          RAISE_APPLICATION_ERROR(-20001, 'El estudiante debe ten
er al menos 16 años.');
  8      END IF;
  9  END;
10  /
```

Disparador creado.



-- Verificar creación

DESC Estudiante;

Nombre	Null?	Tipo
CEDULA	NOT NULL	VARCHAR2(10)
NOMBRES	NOT NULL	VARCHAR2(50)
APELLIDOS	NOT NULL	VARCHAR2(50)
EMAIL	NOT NULL	VARCHAR2(100)
TELEFONO		VARCHAR2(15)
FECHA_NACIMIENTO	NOT NULL	DATE
GENERO	NOT NULL	CHAR(1)
CODIGO_CARRERA	NOT NULL	VARCHAR2(10)
ESTADO	NOT NULL	VARCHAR2(20)
CREDITOS_APROBADOS	NOT NULL	NUMBER(3)
FECHA_INGRESO	NOT NULL	DATE

SQL> |

PASO 4: Crear Tabla ASIGNATURA

sql

```
-- =====
-- TABLA ASIGNATURA - ORACLE
-- =====

-- Eliminar tabla si existe
BEGIN
    EXECUTE IMMEDIATE 'DROP TABLE Asignatura CASCADE CONSTRAINTS';
EXCEPTION
    WHEN OTHERS THEN
        IF SQLCODE != -942 THEN RAISE; END IF;
END;
/
```



```
SQL> BEGIN
 2 EXECUTE IMMEDIATE 'DROP TABLE Asignatura CASCADE CONSTRAINTS'; EXCEPTION
 3 WHEN OTHERS THEN
 4 IF SQLCODE != -942 THEN RAISE; END IF; END;
5 /
```

Procedimiento PL/SQL terminado correctamente.

-- Crear tabla ASIGNATURA

```
CREATE TABLE Asignatura (
    codigo VARCHAR2(10) NOT NULL,
    nombre VARCHAR2(100) NOT NULL,
    creditos NUMBER(1) NOT NULL,
    nivel NUMBER(2) NOT NULL,
    codigo_carrera VARCHAR2(10) NOT NULL,
    horas_teoria NUMBER(2) DEFAULT 0 NOT NULL,
    horas_practica NUMBER(2) DEFAULT 0 NOT NULL,
    descripcion CLOB,
    -- Clave primaria
    CONSTRAINT pk_asignatura PRIMARY KEY (codigo),
    -- Unicidad de nombre por carrera
    CONSTRAINT uk_asignatura_nombre_carrera
        UNIQUE (nombre, codigo_carrera),
    -- Validación de créditos
    CONSTRAINT chk_asignatura_creditos
        CHECK (creditos BETWEEN 1 AND 8),
    -- Validación de nivel
    CONSTRAINT chk_asignatura_nivel
        CHECK (nivel BETWEEN 1 AND 10),
    -- Validación de horas
    CONSTRAINT chk_asignatura_horas
        CHECK (horas_teoria + horas_practica > 0),
    -- Clave foránea a Carrera con CASCADE
    CONSTRAINT fk_asignatura_carrera
        FOREIGN KEY (codigo_carrera)
        REFERENCES Carrera(codigo)
        ON DELETE CASCADE
);
```



```
SQL> CREATE TABLE Asignatura (
  2  codigo VARCHAR2(10) NOT NULL, nombre VARCHAR2(100) NOT NULL
  '
  3  creditos NUMBER(1) NOT NULL, nivel NUMBER(2) NOT NULL,
  4  codigo_carrera VARCHAR2(10) NOT NULL, horas_teoria NUMBER(2
) DEFAULT 0 NOT NULL, horas_practica NUMBER(2) DEFAULT 0 NOT NUL
L, descripcion CLOB,
  5  -- Clave primaria
  6  CONSTRAINT pk_asignatura PRIMARY KEY (codigo),
  7  -- Unicidad de nombre por carrera
  8  CONSTRAINT uk_asignatura_nombre_carrera UNIQUE (nombre, cod
igo_carrera),
  9  -- Validación de créditos
 10 CONSTRAINT chk_asignatura_creditos CHECK (creditos BETWEEN
1 AND 8),
 11 -- Validación de nivel
 12 CONSTRAINT chk_asignatura_nivel CHECK (nivel BETWEEN 1 AND
10),
 13 -- Validación de horas
 14 CONSTRAINT chk_asignatura_horas
 15 CHECK (horas_teoria + horas_practica > 0),
 16 -- Clave foránea a Carrera con CASCADE
 17 CONSTRAINT fk_asignatura_carrera FOREIGN KEY (codigo_carrer
a) REFERENCES Carrera(codigo) ON DELETE CASCADE
 18 );
```

Tabla creada.

-- Índices

CREATE INDEX idx_asignatura_carrera ON Asignatura(codigo_carrera);

CREATE INDEX idx_asignatura_nivel ON Asignatura(nivel);

CREATE INDEX idx_asignatura_carrera_nivel ON Asignatura(codigo_carrera, nivel);

-- Comentarios

COMMENT ON TABLE Asignatura IS

'Catálogo de asignaturas organizadas por carrera y nivel';

COMMENT ON COLUMN Asignatura.codigo IS

'Código único de asignatura (ej: BD-101, MAT-201)';

COMMENT ON COLUMN Asignatura.creditos IS

'Número de créditos académicos que otorga la asignatura';



COMMENT ON COLUMN Asignatura.nivel **IS**

'Nivel o semestre en el que se cursa normalmente';

```
SQL> COMMENT ON TABLE Asignatura IS
  2  'Catálogo de asignaturas organizadas por carrera y nivel';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Asignatura.codigo IS 'Código único de asignatura (ej: BD-101, MAT-201)';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Asignatura.creditos IS
  2  'Número de créditos académicos que otorga la asignatura';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Asignatura.nivel IS 'Nivel o semestre en el que se cursa normalmente';
```

Comentario creado.

-- Verificar creación

DESC Asignatura;

```
SQL> DESC Asignatura
```

Nombre	Nulo?	Tipo
CODIGO	NOT NULL	VARCHAR2(10)
NOMBRE	NOT NULL	VARCHAR2(100)
CREDITOS	NOT NULL	NUMBER(1)
NIVEL	NOT NULL	NUMBER(2)
CODIGO_CARRERA	NOT NULL	VARCHAR2(10)
HORAS_TEORIA	NOT NULL	NUMBER(2)
HORAS_PRACTICA	NOT NULL	NUMBER(2)
DESCRIPCION		CLOB



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

PASO 5: Crear Tabla DOCENTE

sql



```
-- =====
-- TABLA DOCENTE - ORACLE
-- =====

-- Eliminar tabla si existe
BEGIN
    EXECUTE IMMEDIATE 'DROP TABLE Docente CASCADE CONSTRAINTS';
EXCEPTION
    WHEN OTHERS THEN
        IF SQLCODE != -942 THEN RAISE; END IF;
END;
/


-- Crear tabla DOCENTE
CREATE TABLE Docente (
    cedula VARCHAR2(10) NOT NULL,
    nombres VARCHAR2(50) NOT NULL,
    apellidos VARCHAR2(50) NOT NULL,
    email VARCHAR2(100) NOT NULL,
    telefono VARCHAR2(15),
    titulo VARCHAR2(50) NOT NULL,
    especialidad VARCHAR2(100),
    tipo_contrato VARCHAR2(20) DEFAULT 'TIEMPO_COMPLETO' NOT NULL,
    fecha_ingreso DATE DEFAULT SYSDATE NOT NULL,
    -- Clave primaria
    CONSTRAINT pk_docente PRIMARY KEY (cedula),
    -- Unicidad de email
    CONSTRAINT uk_docente_email UNIQUE (email),
    -- Validación de cédula
    CONSTRAINT chk_docente_cedula
        CHECK (REGEXP_LIKE(cedula, '^\\d{10}$')),
    -- Validación de tipo de contrato
    CONSTRAINT chk_docente_tipo_contrato
        CHECK (tipo_contrato IN ('TIEMPO_COMPLETO', 'MEDIO_TIEMPO', 'HORA_CLASE'))
);
```



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

```
SQL> CREATE TABLE Docente (
 2  cedula VARCHAR2(10) NOT NULL, nombres VARCHAR2(50) NOT NULL,
 3  apellidos VARCHAR2(50) NOT NULL, email VARCHAR2(100) NOT NULL,
 4  telefono VARCHAR2(15),
 5  titulo VARCHAR2(50) NOT NULL,
 6  especialidad VARCHAR2(100),
 7  tipo_contrato VARCHAR2(20) DEFAULT 'TIEMPO_COMPLETO' NOT NULL,
 8  fecha_ingreso DATE DEFAULT SYSDATE NOT NULL,
 9  -- Clave primaria
10 CONSTRAINT pk_docente PRIMARY KEY (cedula),
11 -- Unicidad de email
12 CONSTRAINT uk_docente_email UNIQUE (email),
13 -- Validación de cédula
14 CONSTRAINT chk_docente_cedula
15 CHECK (REGEXP_LIKE(cedula, '^\\d{10}$')),
16 -- Validación de tipo de contrato
17 CONSTRAINT chk_docente_tipo_contrato
18 CHECK (tipo_contrato IN ('TIEMPO_COMPLETO', 'MEDIO_TIEMPO', 'HORA_CLASE'))
19 );
```

Tabla creada.

-- Índices

CREATE INDEX idx_docente_apellidos **ON** Docente(apellidos);



```
CREATE INDEX idx_docente_especialidad ON Docente(especialidad);
CREATE INDEX idx_docente_tipo_contrato ON Docente(tipo_contrato);
```

```
SQL>
SQL> CREATE INDEX idx_docente_especialidad ON Docente(especialidad);
=ndice creado.

SQL>
SQL> CREATE INDEX idx_docente_tipo_contrato ON Docente(tipo_contrato);
=ndice creado.
```

-- Comentarios

```
COMMENT ON TABLE Docente IS
```

'Información de docentes de la institución';

```
COMMENT ON COLUMN Docente.titulo IS
```

'Título académico máximo obtenido (Licenciado, Magíster, PhD, etc)';

```
COMMENT ON COLUMN Docente.tipo_contrato IS
```

'Tipo de contrato: TIEMPO_COMPLETO, MEDIO_TIEMPO, HORA_CLASE';

```
SQL> COMMENT ON TABLE Docente IS
2  'Información de docentes de la institución';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Docente.titulo IS
2  'Título académico máximo obtenido (Licenciado, Magíster, PhD, etc)';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Docente.tipo_contrato IS
2  'Tipo de contrato: TIEMPO_COMPLETO, MEDIO_TIEMPO, HORA_CLASE';
```

Comentario creado.

-- Verificar creación

```
DESC Docente;
```



SQL> DESC Docente	Nulo?	Tipo
Nombre		
CEDULA	NOT NULL	VARCHAR2(10)
NOMBRES	NOT NULL	VARCHAR2(50)
APELLIDOS	NOT NULL	VARCHAR2(50)
EMAIL	NOT NULL	VARCHAR2(100)
TELEFONO		VARCHAR2(15)
TITULO	NOT NULL	VARCHAR2(50)
ESPECIALIDAD		VARCHAR2(100)
TIPO_CONTRATO	NOT NULL	VARCHAR2(20)
FECHA_INGRESO	NOT NULL	DATE

PASO 6: Crear Tabla PRERREQUISITO

```
sql
-- =====
-- TABLA PRERREQUISITO - ORACLE
-- =====

-- Eliminar tabla si existe
BEGIN
    EXECUTE IMMEDIATE 'DROP TABLE Prerrequisito CASCADE CONSTRAINTS';
EXCEPTION
    WHEN OTHERS THEN
        IF SQLCODE != -942 THEN RAISE; END IF;
END;
/

-- Crear tabla PRERREQUISITO
CREATE TABLE Prerrequisito (
    codigo_asignatura VARCHAR2(10) NOT NULL,
    codigo_prerrequisito VARCHAR2(10) NOT NULL,
    -- Clave primaria compuesta
    CONSTRAINT pk_prerrequisito
        PRIMARY KEY (codigo_asignatura, codigo_prerrequisito),
    -- Claves foráneas con CASCADE
    CONSTRAINT fk_prereq_asignatura
        FOREIGN KEY (codigo_asignatura)
```



```
REFERENCES Asignatura(codigo)
ON DELETE CASCADE,
CONSTRAINT fk_prereq_prerrequisito
FOREIGN KEY (codigo_prerrequisito)
REFERENCES Asignatura(codigo)
ON DELETE CASCADE,
-- Una asignatura no puede ser prerequisito de sí misma
CONSTRAINT chk_prereq_no_circular
CHECK (codigo_asignatura != codigo_prerrequisito)
);
```

```
SQL> CREATE TABLE Prerrequisito (
  2  codigo_asignatura VARCHAR2(10) NOT NULL, codigo_prerrequisito VARCHAR2(10) NOT NULL,
  3  -- Clave primaria compuesta
  4  CONSTRAINT pk_prerrequisito
  5  PRIMARY KEY (codigo_asignatura, codigo_prerrequisito),
  6  -- Claves foráneas con CASCADE
  7  CONSTRAINT fk_prereq_asignatura FOREIGN KEY (codigo_asignatura) REFERENCES Asignatura(codigo) ON DELETE CASCADE,
  8  CONSTRAINT fk_prereq_prerrequisito FOREIGN KEY (codigo_prerrequisito) REFERENCES Asignatura(codigo) ON DELETE CASCADE,
  9  -- Una asignatura no puede ser prerequisito de sí misma
 10 CONSTRAINT chk_prereq_no_circular
 11 CHECK (codigo_asignatura != codigo_prerrequisito)
 12 );
```

Tabla creada.

-- Índice para búsquedas inversas (qué asignaturas tienen X como prerequisito)

```
CREATE INDEX idx_prereq_codigo_prereq
ON Prerrequisito(codigo_prerrequisito);
```

```
SQL> CREATE INDEX idx_prereq_codigo_prereq ON Prerrequisito(codigo_prerrequisito);

=ndice creado.
```



-- Comentarios

COMMENT ON TABLE Prerrequisito **IS**

'Relación de prerequisitos entre asignaturas';

COMMENT ON COLUMN Prerrequisito.codigo_asignatura **IS**

'Asignatura que requiere el prerequisito';

COMMENT ON COLUMN Prerrequisito.codigo_prerrequisito **IS**

'Asignatura que es prerequisito';

```
SQL> COMMENT ON TABLE Prerrequisito IS 'Relación de prerequisitos entre asignaturas';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Prerrequisito.codigo_asignatura IS 'Asignatura que requiere el prerequisito';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Prerrequisito.codigo_prerrequisito IS 'Asignatura que es prerequisito';
```

Comentario creado.

-- Verificar creación

DESC Prerrequisito;

```
SQL> DESC Prerrequisito
```

Nombre	Nulo?	Tipo
CODIGO_ASIGNATURA	NOT NULL	VARCHAR2(10)
CODIGO_PRERREQUISITO	NOT NULL	VARCHAR2(10)

PASO 7: Crear Tabla MATRICULA con Secuencia y Trigger

sql

```
-- =====  
-- TABLA MATRICULA CON SECUENCIA - ORACLE  
-- =====
```



-- Eliminar objetos si existen

```
BEGIN
    EXECUTE IMMEDIATE 'DROP SEQUENCE seq_matricula';
EXCEPTION
    WHEN OTHERS THEN NULL;
END;
/
```

BEGIN

```
    EXECUTE IMMEDIATE 'DROP TABLE Matricula CASCADE CONSTRAINTS';
EXCEPTION
    WHEN OTHERS THEN
        IF SQLCODE != -942 THEN RAISE; END IF;
END;
/
```

-- Paso 7.1: Crear secuencia para ID de matrícula

```
CREATE SEQUENCE seq_matricula
    START WITH 1
    INCREMENT BY 1
    MAXVALUE 9999999
    MINVALUE 1
    NOCACHE
    NOCYCLE
    ORDER;
```

```
SQL> CREATE SEQUENCE seq_matricula START WITH 1
  2  INCREMENT BY 1
  3  MAXVALUE 9999999
  4  MINVALUE 1 NOCACHE NOCYCLE ORDER;
```

Secuencia creada.

PROMPT Secuencia seq_matricula creada

-- Paso 7.2: Crear tabla MATRICULA

```
CREATE TABLE Matricula (
    id_matricula NUMBER(10) NOT NULL,
```



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

```
cedula_estudiante VARCHAR2(10) NOT NULL,  
codigo_asignatura VARCHAR2(10) NOT NULL,  
cedula_docente VARCHAR2(10) NOT NULL,  
periodo VARCHAR2(10) NOT NULL,  
paralelo CHAR(1) NOT NULL,  
fecha_matricula DATE DEFAULT SYSDATE NOT NULL,  
nota_parcial1 NUMBER(4,2),  
nota_parcial2 NUMBER(4,2),  
nota_final NUMBER(4,2),  
estado VARCHAR2(20) DEFAULT 'CURSANDO' NOT NULL,  
-- Clave primaria  
CONSTRAINT pk_matricula PRIMARY KEY (id_matricula),  
-- Unicidad: un estudiante no puede matricularse dos veces en la misma asignatura en el mismo periodo  
CONSTRAINT uk_matricula_estudiante_asig_periodo  
    UNIQUE (cedula_estudiante, codigo_asignatura, periodo),  
-- Claves foráneas  
CONSTRAINT fk_matricula_estudiante  
    FOREIGN KEY (cedula_estudiante)  
        REFERENCES Estudiante(cedula)  
        ON DELETE CASCADE,  
CONSTRAINT fk_matricula_asignatura  
    FOREIGN KEY (codigo_asignatura)  
        REFERENCES Asignatura(codigo) ,  
CONSTRAINT fk_matricula_docente  
    FOREIGN KEY (cedula_docente)  
        REFERENCES Docente(cedula)  
        ON DELETE SET NULL,  
-- Validación de formato de periodo (YYYY-NS: 2024-1S, 2024-2S)  
CONSTRAINT chk_matricula_periodo  
    CHECK (REGEXP_LIKE(periodo, '^\\d{4}-[12]S$')),  
-- Validación de paralelo (A-Z)  
CONSTRAINT chk_matricula_paralelo  
    CHECK (REGEXP_LIKE(paralelo, '^[A-Z]$')),  
-- Validación de notas (0-10 o NULL)  
CONSTRAINT chk_matricula_nota1  
    CHECK (nota_parcial1 IS NULL OR (nota_parcial1 BETWEEN 0 AND 10)),  
CONSTRAINT chk_matricula_nota2  
    CHECK (nota_parcial2 IS NULL OR (nota_parcial2 BETWEEN 0 AND 10)),  
CONSTRAINT chk_matricula_notafinal
```



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

```
CHECK (nota_final IS NULL OR (nota_final BETWEEN 0 AND 10)),
-- Validación de estado
CONSTRAINT chk_matricula_estado
    CHECK (estado IN ('CURSANDO', 'APROBADO', 'REPROBADO', 'RETIRADO')),
-- Restricción compleja: si está CURSANDO no debe tener nota final
-- Si está APROBADO o REPROBADO debe tener nota final
CONSTRAINT chk_matricula_estado_nota
    CHECK (
        (estado = 'CURSANDO' AND nota_final IS NULL) OR
        (estado IN ('APROBADO', 'REPROBADO') AND nota_final IS NOT NULL) OR
        (estado = 'RETIRADO')
    )
);
```



```
19 CONSTRAINT chk_matricula_paralelo
20 CHECK (REGEXP_LIKE(paralelo, '^[A-Z]$')),
21 -- Validación de notas (0-10 o NULL)
22 CONSTRAINT chk_matricula_nota1
23 CHECK (nota_parcial1 IS NULL OR (nota_parcial1 BETWEEN 0 AND 10)),
24 CONSTRAINT chk_matricula_nota2
25 CHECK (nota_parcial2 IS NULL OR (nota_parcial2 BETWEEN 0 AND 10)),
26 CONSTRAINT chk_matricula_notafinal
27 CHECK (nota_final IS NULL OR (nota_final BETWEEN 0 AND 10))
,
28 -- Validación de estado
29 CONSTRAINT chk_matricula_estado
30 CHECK (estado IN ('CURSANDO', 'APROBADO', 'REPROBADO', 'RETIRADO')),
31 -- Restricción compleja: si está CURSANDO no debe tener nota final
32 -- Si está APROBADO o REPROBADO debe tener nota final
33 CONSTRAINT chk_matricula_estado_nota CHECK (
34 (estado = 'CURSANDO' AND nota_final IS NULL) OR
35 (estado IN ('APROBADO', 'REPROBADO') AND nota_final IS NOT
NULL) OR
36 (estado = 'RETIRADO')
37 )
38 );
```

Tabla creada.

```
-- Paso 7.3: Crear trigger para auto-incremento
CREATE OR REPLACE TRIGGER trg_matricula_id
BEFORE INSERT ON Matricula
FOR EACH ROW
BEGIN
    -- Si no se proporciona id_matricula, usar la secuencia
    IF :NEW.id_matricula IS NULL THEN
        SELECT seq_matricula.NEXTVAL INTO :NEW.id_matricula FROM DUAL;
    END IF;
END;
/
```



PROMPT Trigger trg_matricula_id creado

```
SQL> CREATE OR REPLACE TRIGGER trg_matricula_id BEFORE INSERT ON
  Matricula
  2 FOR EACH ROW BEGIN
  3 -- Si no se proporciona id_matricula, usar la secuencia
  4 IF :NEW.id_matricula IS NULL THEN
  5 SELECT seq_matricula.NEXTVAL INTO :NEW.id_matricula FROM DU
  AL; END IF;
  6 END;
  7 /
```

Disparador creado.

```
SQL> PROMPT Trigger trg_matricula_id creado
Trigger trg_matricula_id creado
```

-- Paso 7.4: Crear índices para optimización

```
CREATE INDEX idx_matricula_estudiante
  ON Matricula(cedula_estudiante);
```

```
CREATE INDEX idx_matricula_asignatura
  ON Matricula(codigo_asignatura);
```

```
CREATE INDEX idx_matricula_periodo
  ON Matricula(periodo);
```

```
CREATE INDEX idx_matricula_docente
  ON Matricula(cedula_docente);
```

```
CREATE INDEX idx_matricula_estado
  ON Matricula(estado);
```

-- Índice compuesto para consultas frecuentes

```
CREATE INDEX idx_matricula_periodo_estado
  ON Matricula(periodo, estado);
```



```
SQL>
SQL> CREATE INDEX idx_matricula_estado ON Matricula(estado);
=ndice creado.

SQL>
SQL> -- Índice compuesto para consultas frecuentes
SQL> CREATE INDEX idx_matricula_periodo_estado ON Matricula(periodo, estado);

=ndice creado.
```

-- Comentarios

COMMENT ON TABLE Matricula **IS**

'Registro de matrículas de estudiantes en asignaturas por periodo académico';

COMMENT ON COLUMN Matricula.id_matricula **IS**

'Identificador único generado automáticamente por secuencia';

COMMENT ON COLUMN Matricula.periodo **IS**

'Periodo académico en formato YYYY-NS (ej: 2024-1S, 2024-2S)';

COMMENT ON COLUMN Matricula.paralelo **IS**

'Paralelo o grupo de la asignatura (A, B, C, etc)';

COMMENT ON COLUMN Matricula.estado **IS**

'Estado de la matrícula: CURSANDO, APROBADO, REPROBADO, RETIRADO';



```
SQL> COMMENT ON TABLE Matricula IS
  2  'Registro de matrículas de estudiantes en asignaturas por período académico';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Matricula.id_matricula IS 'Identificador único generado automáticamente por secuencia';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Matricula.periodo IS
  2  'Período académico en formato YYYY-NS (ej: 2024-1S, 2024-2S)';

```

Comentario creado.

```
SQL> COMMENT ON COLUMN Matricula.paralelo IS 'Paralelo o grupo de la asignatura (A, B, C, etc)';
```

Comentario creado.

```
SQL> COMMENT ON COLUMN Matricula.estado IS
  2  'Estado de la matrícula: CURSANDO, APROBADO, REPROBADO, RETIRADO';
```

Comentario creado.

-- Verificar creación

DESC Matricula;



```
SQL> DESC Matricula
```

Nombre	Nulo?	Tipo
ID_MATRICULA	NOT NULL	NUMBER(10)
CEDULA_ESTUDIANTE	NOT NULL	VARCHAR2(10)
CODIGO_ASIGNATURA	NOT NULL	VARCHAR2(10)
CEDULA_DOCENTE	NOT NULL	VARCHAR2(10)
PERIODO	NOT NULL	VARCHAR2(10)
PARALELO	NOT NULL	CHAR(1)
FECHA_MATRICULA	NOT NULL	DATE
NOTA_PARCIAL1		NUMBER(4,2)
NOTA_PARCIAL2		NUMBER(4,2)
NOTA_FINAL		NUMBER(4,2)
ESTADO	NOT NULL	VARCHAR2(20)

-- Verificar secuencia

```
SELECT sequence_name, last_number, increment_by, cache_size
FROM user_sequences
WHERE sequence_name = 'SEQ_MATRICULA';
```

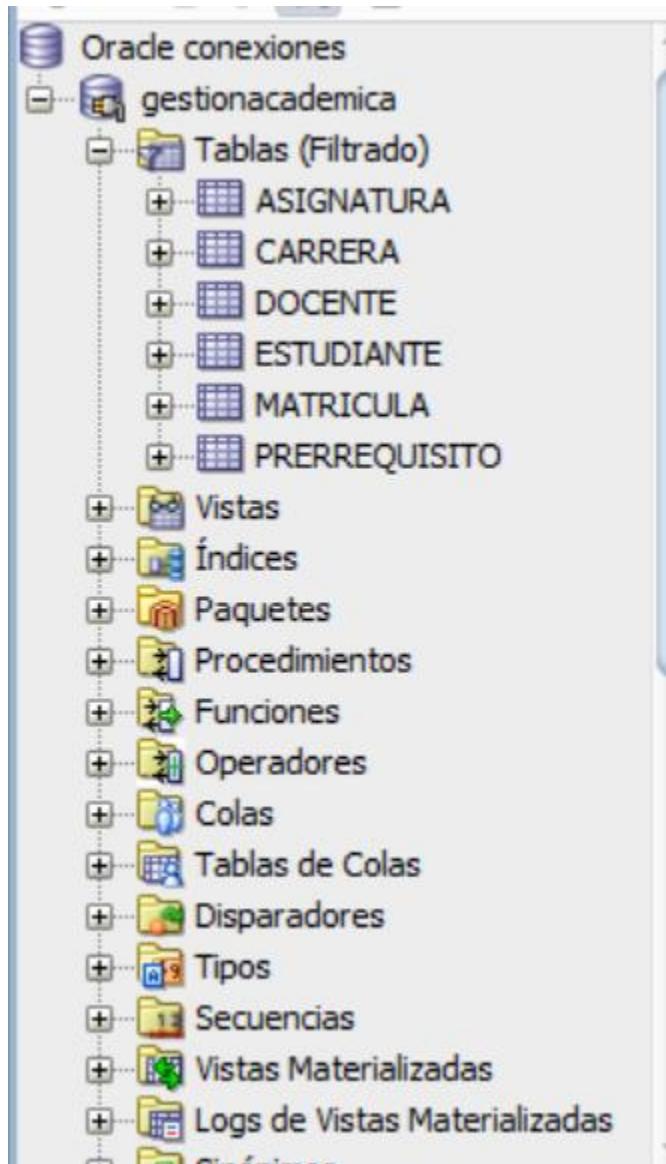
```
SQL> SELECT sequence_name, last_number, increment_by, cache_size FROM user_sequences
  2 WHERE sequence_name = 'SEQ_MATRICULA';
```

SEQUENCE_NAME
LAST_NUMBER INCREMENT_BY CACHE_SIZE
SEQ_MATRICULA 1 1 0

Trabajo hecho hasta ahora:



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS





```
SQL> desc tab
Nombre          Nullable?  Tipo
-----          -----
TNAME           NOT NULL  VARCHAR2(128)
TABTYPE         VARCHAR2(13)
CLUSTERID      NUMBER

SQL> select tname from tab;

TNAME
-----
CARRERA
ESTUDIANTE
ASIGNATURA
DOCENTE
PRERREQUISITO
MATRICULA

6 filas seleccionadas.
```

A	B	C
1 CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
2 SYS_C007671	C	"NOMBRE" IS NOT NULL
3 SYS_C007672	C	"DURACION_SEMESTRES" IS NOT NULL
4 SYS_C007673	C	"CREDITOS_TOTALES" IS NOT NULL
5 SYS_C007670	C	"CODIGO" IS NOT NULL
6 SYS_C007674	C	"FACULTAD" IS NOT NULL
7 CHK_CARRERA_CREDITOS	C	creditos_totales > 0
8 SYS_C007675	C	"FECHA_CREACION" IS NOT NULL
9 CHK_CARRERA_DURACION	C	duracion_semestres BETWEEN 8 AND 12
10 PK_CARRERA	P	
11 UK_CARRERA_NOMBRE	U	
12		
13		

Archivo Excel generado