Santiago Chiale

santiagochiale@abc.gob.ar

LOGICA COMPUTACIONAL

Contenido

[A) Fundamentación 2](#_Toc190531270)

[B) Expectativas de logro 3](#_Toc190531271)

[C) Contenidos académicos 3](#_Toc190531272)

[D) Intervención didáctica 4](#_Toc190531273)

[E) Atención al impacto de la propuesta en la Práctica Docente o Profesional 5](#_Toc190531274)

[F) Presupuesto de tiempo 5](#_Toc190531275)

[G) Recursos 5](#_Toc190531276)

[H) Bibliografía 6](#_Toc190531277)

[I) Propuesta de evaluación 7](#_Toc190531278)

[J) Propuesta de actividades de extensión e investigación 8](#_Toc190531279)

# Introducción

A lo largo de la historia, la humanidad ha buscado diversas formas de representar y procesar la información. ¿Cómo es posible que un dispositivo tan cotidiano como una calculadora “comprenda” que, al presionar los botones correspondientes a “5”, “+”, “3” y “=”, el resultado que debe mostrar es “8”? ¿De qué manera una computadora personal (PC) es capaz de ejecutar miles de instrucciones por segundo para gestionar tareas que van desde la edición de textos hasta la navegación por internet? Y, finalmente, ¿cómo se llega al desarrollo de la inteligencia artificial (IA), que puede procesar millones de datos, reconocer rostros o jugar partidas de ajedrez con gran maestría?

La clave común en todos estos procesos –ya sea en la calculadora, en la PC o en la IA– reside en los principios lógicos y en los circuitos electrónicos que administran señales binarias, es decir, secuencias de “0” y “1”. La lógica computacional constituye la base fundamental en la que se diseñan y organizan estos dígitos, transformándolos en operaciones aritméticas, lógicas y, en última instancia, en acciones concretas: la pantalla muestra un resultado, el procesador ejecuta un programa y una red neuronal lleva a cabo una fase de aprendizaje.

Imagina que partimos de una calculadora: cada vez que se introducen dos números y se selecciona un símbolo de operación (como PLUS, MINUS, MULTIPLY, o DIVIDE), la máquina debe convertir esa entrada en impulsos eléctricos compuestos por niveles “alto” (1) y “bajo” (0). Lo que para el usuario es una simple suma o resta se transforma internamente en una serie de operaciones lógicas y aritméticas elementales, organizadas en circuitos mediante puertas lógicas como AND, OR, NOT, entre otras. Sin la estructura lógica basada en la teoría booleana, la calculadora no podría interpretar correctamente los dígitos ingresados.

El mismo principio se extiende a una computadora personal. Aquí se pasa de manejar un par de señales a coordinar millones de ellas, utilizando un procesador que ejecuta instrucciones almacenadas en la memoria. Detrás de la carcasa de una PC, tanto la CPU como la memoria se basan en biestables, buses y en compuertas lógicas más complejas, como multiplexores y decodificadores. Comprender el funcionamiento de estos “ladrillos digitales” nos brinda una visión clara de lo que ocurre cuando se inicia un programa, se guarda un archivo o se establece una conexión a internet. Cada clic y cada instrucción se implementa, en última instancia, mediante las mismas señales binarizadas que se utilizan en una calculadora, pero en una escala muchísimo mayor.

Por último, la inteligencia artificial lleva esta lógica computacional a un nivel superior. En la IA se aplican redes neuronales, algoritmos de aprendizaje y se manejan grandes volúmenes de datos. Sin embargo, en el núcleo de todas estas operaciones complejas se encuentran las operaciones sobre bits –multiplicaciones de matrices, pasos de backpropagation y otros cálculos– que se implementan en hardware especializado (como GPUs, TPUs o ASICs) y que están diseñados con puertas lógicas, registros, sumadores y decodificadores.

¿Por qué es importante estudiar esta base de la lógica computacional?

Para desentrañar el funcionamiento interno de la tecnología: Lo que puede parecer un misterio o una “caja negra” se revela como un conjunto de principios claros basados en “0”, “1” y en el uso de puertas lógicas.

Para comprender las limitaciones inherentes a los sistemas digitales: Al dominar la representación de la información y las operaciones lógicas, se entiende por qué existen límites, como el desbordamiento numérico, la velocidad del reloj y la latencia de la memoria.

Para apreciar la continuidad conceptual entre dispositivos simples y sistemas avanzados: Desde una calculadora elemental hasta el “cerebro” de un sistema de IA, la base es la lógica digital que, ampliada con técnicas de programación y matemáticas avanzadas, permite alcanzar niveles de complejidad sorprendentes.

En este manual se explorarán:

La forma en que los enunciados lógicos y la teoría booleana se traducen en circuitos digitales mediante compuertas, flip-flops, decodificadores y multiplexores.

La importancia del diseño de estos circuitos en la evolución tecnológica, desde la calculadora hasta la computadora personal y, en última instancia, hasta la inteligencia artificial.

Cómo el razonamiento formal –tanto en la lógica proposicional como en la lógica de predicados– se utiliza para describir el comportamiento de sistemas y para verificar su correcta implementación en proyectos de ingeniería de software, data science e IA.

A medida que avances por cada capítulo, descubrirás progresivamente:

En primer lugar, los fundamentos de la lógica proposicional y la manipulación de circuitos básicos.

En segundo lugar, la lógica de predicados, que extiende la capacidad descriptiva mediante la introducción de cuantificadores.

Finalmente, cómo estas ideas abstractas se concretan en hardware real, abriendo el camino para comprender desde el funcionamiento de un microprocesador hasta cómo las redes neuronales pueden clasificar miles de imágenes por segundo o procesar lenguaje natural, gracias a la integración de lógica, electrónica y computación.

# 

# CAPÍTULO 1: LÓGICA PROPOSICIONAL

## Antecedentes Históricos y Relevancia Contemporánea

El estudio de la lógica proposicional tiene raíces que se remontan a la Antigua Grecia, donde Aristóteles sentó las bases de lo que hoy conocemos como lógica silogística. Sin embargo, la idea de tratar las proposiciones como entidades formales, que pueden combinarse mediante conectores lógicos, comenzó a consolidarse a partir de los trabajos de George Boole en el siglo XIX, quien desarrolló lo que ahora se conoce como álgebra booleana. Posteriormente, durante el siglo XX, figuras como Hilbert, Frege y Russell sistematizaron la lógica formal, lo que permitió establecer los fundamentos de la lógica computacional que subyace en la informática moderna.

Hoy en día, la lógica proposicional es considerada la puerta de entrada al razonamiento formal. No solo tiene un profundo valor filosófico, sino que también es de enorme relevancia en áreas como la ingeniería de software, la inteligencia artificial, la verificación de circuitos digitales y la teoría de lenguajes formales.

## Conceptos Fundamentales

### Proposiciones

Una proposición es un enunciado declarativo que puede evaluarse como verdadero (V) o falso (F). No se consideran proposiciones aquellas expresiones que no admiten una clasificación clara de verdad, como preguntas, órdenes o exclamaciones.

Ejemplos de proposiciones:

* "7 es un número primo" → Puede ser evaluada matemáticamente como verdadera.
* "Haré café mañana" → Aunque su evaluación depende de circunstancias futuras, sigue siendo una proposición porque en algún momento podrá clasificarse como V o F.

Ejemplos de expresiones que no son proposiciones:

* "¿Hace calor hoy?" → No es declarativa, es una pregunta.
* "¡Qué hermoso día!" → Es una exclamación sin valor de verdad.
* "Cierra la puerta." → Es una orden, no tiene valor de verdad.

### 1.2.2 Variables Proposicionales

Las variables proposicionales se utilizan para representar enunciados de forma simbólica. Se denotan con letras como P, Q, R, S, ... y pueden tomar solo dos valores: verdadero o falso.

**Ejemplo:**

* P = "Está lloviendo"

Si está lloviendo, P es verdadero; si no, es falso.

Las variables proposicionales permiten generalizar razonamientos y aplicar reglas lógicas de manera más eficiente.

### 1.2.3 Conectivos Lógicos

Los conectivos lógicos permiten combinar proposiciones para formar expresiones más complejas. Sus principales operadores son:

* Negación (NOT, !P) → Invierte el valor de la proposición.
* Conjunción (AND, P & Q) → Es verdadera solo si ambas proposiciones son verdaderas.
* Disyunción (OR, P || Q) → Es verdadera si al menos una proposición es verdadera.
* Implicación (IMPLIES, P → Q) → Solo es falsa cuando P es verdadera y Q es falsa.
* Bicondicional (IFF, P ↔ Q) → Es verdadera si ambas proposiciones tienen el mismo valor de verdad.

### 1.2.4 Tablas de verdad

Las tablas de verdad son herramientas fundamentales en lógica proposicional. Sirven para analizar el valor de verdad (Verdadero o Falso) de una fórmula lógica en función de todas las combinaciones posibles de valores de sus variables.

Se utilizan para:

* Verificar si una fórmula es siempre verdadera (tautología), siempre falsa (contradicción) o satisfacible.
* Estudiar la validez de argumentos.
* Analizar circuitos lógicos y condiciones en programación.

### 1.2.5 Como se construye una tabla de verdad

1. Identificar las variables proposicionales

Primero, contá cuántas variables contiene la fórmula (por ejemplo, P, Q, R).

* + Si hay n variables, la tabla tendrá 2n2^n2n filas, porque cada variable puede ser V (Verdadero) o F (Falso).
    - 1 variable → 2 combinaciones
    - 2 variables → 4 combinaciones
    - 3 variables → 8 combinaciones

1. Enumerar todas las combinaciones posibles

Listá todas las combinaciones de verdadero (V) y falso (F) para esas variables. Usá un patrón binario descendente.

Ejemplo con 2 variables:

|  |  |
| --- | --- |
| **P** | **Q** |
| F | F |
| F | V |
| V | F |
| V | V |

1. Evaluar la fórmula paso a paso

Agregar columnas intermedias si la expresión es compleja, para calcular por partes.

Ejemplo, para P && !Q, se puede construir así:

|  |  |  |  |
| --- | --- | --- | --- |
| **P** | **Q** | **!Q** | **P & !Q** |
| F | F | V | F |
| F | V | F | F |
| V | F | V | V |
| V | V | F | F |

1. Interpretar los resultados
   * Una vez completada la tabla:
   * Si la última columna es siempre V, la fórmula es una tautología.
   * Si es siempre F, es una contradicción.
   * Si tiene al menos una V, es satisfacible.

¿Para qué se usan?

* Verificación de circuitos digitales.
* Programación: análisis de condiciones lógicas en if, while, etc.
* Diseño lógico: ayuda a convertir expresiones en formas equivalentes o simplificadas.
* Lógica matemática: comprobar propiedades, leyes o validez de razonamientos.

Ejemplo para la conjunción (AND):

|  |  |  |
| --- | --- | --- |
| P | Q | P & Q |
| V | V | V |
| V | F | F |
| F | V | F |
| F | F | F |

### 1.2.4 Propiedades de los Conectivos Lógicos

Algunas propiedades fundamentales en la lógica proposicional son:

1. Conmutatividad

Estas leyes indican que el orden de los operandos no afecta el resultado:

P || Q === Q || P

P && Q === Q && P

1. Asociatividad

Estas propiedades dicen que la agrupación entre paréntesis no cambia el resultado:

(P || Q) || R === P || (Q || R)

(P && Q) && R === P && (Q && R)

1. Distributividad

Permite reorganizar expresiones combinando AND y OR:

P && (Q || R) === (P && Q) || (P && R)

P || (Q && R) === (P || Q) && (P || R)

1. Leyes de De Morgan

Muestran cómo distribuir la negación sobre una conjunción o disyunción:

!(P && Q) === !P || !Q

!(P || Q) === !P && !Q

### 1.2.5 Ejercicios Propuestos

#### Ejercicio 1: Identificación de Proposiciones

Dado el siguiente conjunto de enunciados, indicar cuáles son proposiciones y justificar la respuesta:

1. "El sol es una estrella."
2. "Apaga la luz."
3. "5 + 7 = 13."
4. "¿Vendrás mañana?"
5. "El cielo es azul y el agua es transparente."

#### Ejercicio 2: Evaluación de Expresiones Lógicas

Para cada una de las siguientes expresiones lógicas, construí la tabla de verdad completa considerando todas las combinaciones posibles de los valores de P, Q y (si corresponde) R.

1. P || Q

|  |  |  |
| --- | --- | --- |
| **P** | **Q** | **P || Q** |
| F | F | F |
| F | V | V |
| V | F | V |
| V | V | V |

2. !P && !Q

Tabla intermedia:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **P** | **Q** | **!P** | **!Q** | **!P & !Q** |
| F | F | V | V | V |
| F | V | V | F | F |
| V | F | F | V | F |
| V | V | F | F | F |

Resultado:

|  |  |  |
| --- | --- | --- |
| **P** | **Q** | **!P & !Q** |
| F | F | V |
| F | V | F |
| V | F | F |
| V | V | F |

3. (!P || Q) || R

Tabla intermedia:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **P** | **Q** | **R** | **!P** | **(!P || Q)** | **(!P || Q) || R** |
| F | F | F | V | V | V |
| F | F | V | V | V | V |
| F | V | F | V | V | V |
| F | V | V | V | V | V |
| V | F | F | F | F | F |
| V | F | V | F | F | V |
| V | V | F | F | V | V |
| V | V | V | F | V | V |

Resultado:

|  |  |  |  |
| --- | --- | --- | --- |
| **P** | **Q** | **R** | **(!P || Q) || R** |
| F | F | F | V |
| F | F | V | V |
| F | V | F | V |
| F | V | V | V |
| V | F | F | F |
| V | F | V | V |
| V | V | F | V |
| V | V | V | V |

4. !(P & Q) || R

Tabla intermedia:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **P** | **Q** | **R** | **P & Q** | **! (P & Q)** | **! (P & Q)|| R** |
| F | F | F | F | V | V |
| F | F | V | F | V | V |
| F | V | F | F | V | V |
| F | V | V | F | V | V |
| V | F | F | F | V | V |
| V | F | V | F | V | V |
| V | V | F | V | F | F |
| V | V | V | V | F | V |

Resultado:

|  |  |  |  |
| --- | --- | --- | --- |
| **P** | **Q** | **R** | **! (P & Q)|| R** |
| F | F | F | V |
| F | F | V | V |
| F | V | F | V |
| F | V | V | V |
| V | F | F | V |
| V | F | V | V |
| V | V | F | F |
| V | V | V | V |

Instrucciones para los alumnos:

* Enumerá todas las combinaciones posibles de valores de P, Q (y R cuando sea necesario).
* Evaluá paso a paso cada subexpresión si es necesario.
* Indicá si el resultado final de la expresión es V (verdadero) o F (falso) en cada caso.

#### Ejercicio 3: codepen

Escriba una sencilla aplicación web que simule los ejercicios anteriores.

Parámetros de diseño:

* Debe estar escrita en JavaScript
* En el front debe contener inputs para colocar los valores con las etiquetas correspondientes (P, Q y R)
* Estos inputs solo deben admitir valores V o F o bien 1 o 0
* El resultado de la proposición debe verse en otro input pero sobre el cual no se pueda ingresar valores

Aclaraciones

* Se puede realizar un grupo de inputs para cada ejercicio o se puede reutilizar los mismos inputs indicando el tipo de operación a realizar con una lista desplegable al lado de cada uno (Pensar en la escalabilidad)

# Contenidos académicos

LOGICA COMPUTACIONAL

*(Desarrollo durante el 1er cuatrimestre)*

**Unidad 1: Lógica Proposicional**

* Elementos de lógica, proposiciones y conectivos lógicos.
* Formas normales (conjuntiva y disyuntiva), validez y argumentación.
* Tablas de verdad: definición, construcción y uso en la verificación de validez.
* Conjuntos adecuados de conectivas, usos en aplicaciones computacionales.

**Unidad 2: Lógica de Predicados**

* Cuantificadores universal y existencial, interpretación en lenguajes de primer orden.
* Sistemas formales: corrección y completitud, modelos e interpretaciones.
* Límites de la lógica de predicados: intratabilidad e inexpresabilidad en ciertos problemas.
* Aplicación en la deducción natural, estrategias de formalización y deducción.

**Unidad 3: Lógica Digital**

* Introducción al álgebra de Boole y compuertas lógicas: NAND, NOR, OR exclusiva.
* Multiplexores, decodificadores, biestables, memorias, microcontroladores.
* Diseño y optimización de circuitos digitales.
* Relación con la computación y la ciencia de datos (nivel hardware y procesamiento básico).

ADMINISTRACIÓN Y GESTIÓN DE BASE DE DATOS

*(Desarrollo durante el 2do cuatrimestre)*

**Unidad 1: Organización de Datos**

* Modelos conceptuales: E/R, UML, orientado a objetos, semiestructurados (XML).
* Componentes y funciones de un SGBD (Sistema Gestor de Base de Datos).
* Definición de datos y álgebra relacional.
* Estructuras de almacenamiento: fundamentos para la manipulación de datos.

**Unidad 2: Diseño de Base de Datos**

* Diseño lógico y físico: dependencia funcional, normalización (formas normales).
* Descomposición de esquemas, claves primarias y foráneas.
* Procesamiento de transacciones (fallas, recuperación, control de concurrencia).
* Bases de datos distribuidas y problemas inherentes a su explotación.

**Unidad 3: Fundamentos de Administración y Gestión**

* Distintas categorías de SGBD: libres y propietarios (MySQL, PostgreSQL, Oracle, etc.).
* Actores y roles: administrador, desarrollador, analista.
* Operaciones: consultas, inserción, modificación, eliminación. Procedimientos almacenados y disparadores.
* Seguridad de la información: accesos, permisos, roles, control de transacciones.

1. Intervención didáctica

En este espacio curricular, se contemplan diversos estilos y estrategias de enseñanza y aprendizaje, atendiendo a la naturaleza teórico-práctica de los contenidos. Para favorecer el aprendizaje de las y los estudiantes, se proponen:

* Ejercicios de formalización (lógica proposicional, predicados), simuladores de circuitos digitales, construcción de tablas de verdad.
* Bases de Datos: Talleres para modelar esquemas (E/R), normalización, configuración de un SGBD, simulaciones de transacciones.
* Metodologías Teórico-Prácticas
* Clases expositivo-dialogadas para presentar los fundamentos y ejemplos de lógica y bases de datos.
* Aprendizaje basado en problemas (ABP): planteo de escenarios reales (control de stock, razonamiento lógico) que requieran el uso de lógica formal y el diseño de una base de datos.
* Casos simulados o reales en entornos de ciencia de datos e IA: configuración de microcontroladores y verificación de su lógica, o diseño de una BD para almacenar información masiva.
* Evaluación de la escalabilidad y la seguridad (por ejemplo, en la gestión de una BD distribuido).
* Implementación en el Entorno Virtual
* Uso de laboratorios informáticos o entornos virtuales (Moodle, Classroom) para compartir material teórico y prácticas.
* Repositorios en línea (GitHub, GitLab) para alojar scripts SQL, esquemas E/R, ejercicios de lógica proposicional/predicados.

# Atención al impacto de la propuesta en la Práctica Docente o Profesional

Esta propuesta está diseñada para conectar los aspectos teóricos de la programación y el rol laboral con situaciones reales de la industria, especialmente en el ámbito de la Ciencia de Datos e Inteligencia Artificial.

Se promueve que las y los estudiantes reconozcan el valor de la programación y el análisis de datos en diversos entornos (PyMEs, organizaciones grandes, sector público), articulando así con la práctica profesional.

Aportes al Perfil del Egresado

Gracias a las actividades de simulación (por ejemplo, proyectos colaborativos con un supuesto cliente), se desarrollan competencias clave para la inserción laboral exitosa: comunicación efectiva, resolución de problemas, uso de tecnologías emergentes y responsabilidad profesional (cumplimiento legal y ético).

Al concluir, el/la estudiante podrá integrarse rápidamente en equipos de trabajo orientados al desarrollo de software y la ciencia de datos, aportando soluciones técnicas y criterios de calidad, además de la capacidad para comunicarse con clientes y usuarios de manera clara y profesional.

# Presupuesto de tiempo

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  | | --- | --- | | Cuatrimestre / Mes | Contenidos / Actividades | | ***Primer Cuatrimestre*** | ***Lógica Computacional: Unidades 1 a 3*** | | Marzo | Unidad 1:Lógica Proposicional | | Abril | Unidad 1:Lógica Proposicional, Unidad 2:Lógica de Predicados | | Mayo | Unidad 3: Lógica Digital | | Junio/Julio (1er y 2da semana) | Unidad 3: Lógica Digital, Parcial (1ª y 2dainstancia) | | ***Segundo Cuatrimestre*** | ***Administración y Gestión de BD*** | | Agosto (3era y 4ta semana) /Septiembre | Unidad 1: Organización de Datos (modelos E/R, relacional).  Prácticas iniciales. | | Octubre | Unidad 2: Diseño de Base de Datos | | Noviembre | Unidad 3: Fundamentos de Administración y Gestión | |

# Recursos

Recursos Materiales

* Computadora y software de simulación de circuitos lógicos.
* **SGBD** instalados (MySQL, PostgreSQL, etc.) para la práctica de bases de datos.
* Herramientas de presentación (proyector, pizarrón, etc.).

Recursos Didácticos

* Apuntes de clase, simuladores (por ej. Logisim para lógica digital).
* Plataforma virtual (Moodle, Classroom) para compartir y recibir material y actividades.

# Bibliografía

Bibliografía para el Docente

* Enderton, H. (2001). A Mathematical Introduction to Logic.
* Rosen, K. (2012). Discrete Mathematics and its Applications.
* Elmasri, R., & Navathe, S. (2015). Fundamentals of Database Systems.Date, C. J. (2003). An Introduction to Database Systems.

Recursos Transversales

* Mendelson, E. (1987). Introduction to Mathematical Logic.
* Connolly, T., & Begg, C. (2015). Database Systems: A Practical Approach to Design,

BIBLIOGRAFÍA PARA EL ALUMNO

* Rojas, R. (1997). Teoría de Autómatas, Lenguajes y Computación (algunos capítulos)
* Manual de Lógica Digital (Logisim)
* Documentaciones y tutoriales (sitios web oficiales, guías de la herramienta) para la simulación de circuitos.
* Coronel, C., Morris, S., & Rob, P. (2012). Database Systems: Design, Implementation, & Management.
* Prácticas con MySQL o PostgreSQL (Documentación oficial y tutoriales en línea)

Recursos Adicionales

* OpenAI / ChatGPT: para consultas rápidas sobre definiciones lógicas o ejemplos de queries SQL.
* Simuladores de Lógica (ej. Logisim, Digital Works): para diseñar compuertas y circuitos combinacionales/secuenciales.

# Propuesta de evaluación

LOGICA COMPUTACIONAL

1. Examen parcial escrito teórico-práctico

Se realizará un (1) examen parcial que abarque conceptos fundamentales, estructuras de datos, algoritmos y desarrollo de programas. Este examen tendrá, como es habitual, una segunda instancia.

La calificación tendrá la siguiente escala de aprobación:

* + De 1 a 3: desaprobado
  + De 4 a 6: debe rendir examen final
  + De 7 a 10: promocionado

1. Proyectos grupales

Objetivo: Fomentar el trabajo colaborativo y la aplicación de metodologías ágiles.

Entrega: Repositorio en GitHub y presentación oral/codificada.

1. Participación y compromiso

Se dará gran importancia a la participación proactiva en cada una de las clases y los aportes al grupo en ese sentido.

1. Asistencia

El alumno deberá contar con un mínimo de 60%.

Indicadores de Evaluación

* Resolución de problemas vinculados a la lógica proposicional con valores de verdad.
* Estudio de las deducciones y razonamientos, justificando la validez del razonamiento mediante la justificación sintáctica.
* Producciones vinculadas a predicados de primer orden, alfabeto de símbolos, conjunto de cadenas finitas de los símbolos, axiomas y el conjunto finito

ADMINISTRACIÓN Y GESTIÓN DE BASES DE DATOS

1. Examen parcial escrito teórico-práctico

Se realizará un (1) examen parcial que abarque conceptos fundamentales, estructuras de datos, algoritmos y desarrollo de programas. Este examen tendrá, como es habitual, una segunda instancia.

La calificación tendrá la siguiente escala de aprobación:

* + De 1 a 3: desaprobado
  + De 4 a 6: debe rendir examen final
  + De 7 a 10: promocionado

1. Proyectos grupales

Objetivo: Fomentar el trabajo colaborativo y la aplicación de metodologías ágiles.

Entrega: Repositorio en GitHub y presentación oral/codificada.

1. Participación y compromiso

Se dará gran importancia a la participación proactiva en cada una de las clases y los aportes al grupo en ese sentido.

1. Asistencia

El alumno deberá contar con un mínimo de 60%.

Indicadores de Evaluación

* Manejar el motor de base de datos de acuerdo con la situación a resolver.
* Definir la estructura, índices y relaciones entre tablas de la base de datos.
* Optimizar bases de datos, mediante procedimientos de normalización.
* Elaboración de documentación acorde al problema

# Propuesta de actividades de extensión e investigación

Proyectos de corta duración

* Armar mini-herramientas que validen tablas de verdad o generen expresiones booleanas.
* Desarrollar una pequeña aplicación CRUD con un SGBD, analizando la concurrency.

Simulaciones de entornos laborales

Se realizará una articulación con el espacio Practica profesionalizante: Aproximación al campo laboral del mismo año donde, en ella se desarrollará una aplicación web para un cliente ficticio con una necesidad puntual. Aquí se implementará la gestión de los datos en la base de datos de dicha aplicación.

La pertinencia de estas actividades radica en fortalecer la conexión con la realidad productiva y la factibilidad se ajusta al entorno institucional (laboratorios, tiempos de cursada, plataformas de colaboración).