Formación Laboral Básica
COBATAB

COMPETENCIA LABORAL BÁSICA EN

Tecnologías de la Información y la Comunicación

5.° Semestre

2026–2027A

MÓDULO III

Desarrollo de Sistemas

Submódulo I:
Sistemas de Información

Submódulo II:
Programación

Datos del Estudiante

Nombre: _____

Plantel: _____ Grupo: _____ Turno: _____

COLEGIO DE BACHILLERES DE TABASCO

Mtro. Fernando Yrys Hernández
Director General

Mtra. Ana Fanny Hernández Montero
Directora Académica

Lic. Celis del Carmen Ranero Lara
Subdirectora de Planeación Académica

Mtra. Natividad Vertiz Hernández
Jefa del Departamento Capacitación para el Trabajo

FORMACIÓN LABORAL BÁSICA: TECNOLOGÍAS DE LA INFORMACIÓN Y LA COMUNICACIÓN
MÓDULO III. Desarrollo de Sistemas.
SUBMÓDULO I: Sistemas de Información
SUBMÓDULO II: Programación

En la realización del presente material participaron:
Coordinador: Mtro. Jorge Alberto Almeida Alejandro.
Asesor: Mtro. Inés Filiberta Díaz Ramírez,

Participantes:

Dr. Milton Hernández García,
Mtra. Miriam Córdova Hernández,
Mtra. Patricia del Carmen Rabanales Cervantes,
Mtro. Gabriel Ramón Velueta
Lic. Avit Vázquez Gómez

Edición: 2025-2026A

Revisado por moderador a cargo del seguimiento académico
Lic. Martín Ricardo Beltrán Cháirez

Este material fue elaborado bajo la coordinación y supervisión del Departamento Capacitación para el Trabajo de la Dirección Académica del Colegio de Bachilleres de Tabasco y el contenido del mismo es gratuito, sin fines de lucro y con la responsabilidad de su buen uso para fines educativos www.cobatab.edu.mx

CONTENIDO

TABASCO
GOBIERNO DEL PUEBLO

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN

1976 · 2026
ANIVERSARIO

COBATAB
COLEGIO DE BACHILLERES DE TABASCO

GUÍA DE ESTUDIANTES

Mensaje del Director General

Estimada comunidad estudiantil del Colegio de Bachilleres de Tabasco: quiero hacer énfasis en nuestra Institución, la cual tiene 5 décadas forjando el aprendizaje de cada uno de los jóvenes, hoy muchos de los que han pasado por nuestras aulas de clases son profesionistas exitosos.

Dentro del marco del 50 aniversario COBATAB, me permito como Director General enviarles un cordial saludo y dirigirme a ustedes a través de esta guía de estudio diseñada para acompañarlos como una herramienta en su formación académica y personal.

En esta etapa de su vida, el aprendizaje autónomo es clave para descubrir sus talentos, fortalecer sus habilidades y construir un futuro con propósito. Cada actividad, cada lectura y cada reto que encuentren en esta guía de estudio está pensado para impulsar su Desarrollo integral.

Les invito a asumir con responsabilidad su proceso de aprendizaje, a colaborar con sus compañeros, docentes y a mantener siempre una actitud de respeto y apertura. En el Colegio de Bachilleres de Tabasco promovemos la equidad y creemos firmemente en el potencial de cada uno de ustedes.

Esta guía de estudio busca ser clara y accesible, con objetivos definidos y contenidos relevantes para su contexto. Les animo a aprovecharla al máximo, a preguntar, reflexionar y aplicar lo aprendido en su vida diaria.

Confío en su capacidad para superar desafíos, alcanzar metas y contribuir positivamente a su comunidad. Cuentan con el apoyo de sus docentes y de todos los que integramos esta institución.

Con entusiasmo y compromiso, sigamos construyendo juntos una educación de calidad, que genera cambio.

Con aprecio y confianza,

Fernando Yrys Hernández
Director General
Colegio de Bachilleres de Tabasco

Presentación Programas de Estudio

La Dirección General del Bachillerato (DGB) presenta las Competencias de las diversas Unidades de Aprendizaje Curricular del Competente de Formación Laboral, para el Plan de estudios propio de esta Dirección General. Estas tienen su sustento, teórica y conceptualmente, en el modelo educativo del Marco Curricular Común de la Educación Media Superior (MCCEMS)[1] , y dan cumplimiento a las atribuciones conferidas a esta Dirección por el Reglamento Interior de la Secretaría de Educación Pública (SEP), en el cual se establece, en el Artículo 19 Fracciones I y II la importancia de "proponer las normas pedagógicas, contenidos, planes y programas de estudio, métodos, materiales didácticos e instrumentos para la evaluación del aprendizaje del bachillerato general, en sus diferentes modalidades y enfoques, y difundir los vigentes"; además de "impulsar las reformas curriculares de los estudios de bachillerato que resulten necesarias para responder a los requerimientos de la sociedad del conocimiento y del desarrollo sustentable" (RISEP, 2020).

En este sentido, los planteamientos del MCCEMS buscan una formación integral en el estudiantado mediante el desarrollo de la capacidad creadora, productiva, libre y digna del ser humano, con amor al país, a su cultura e historia. Por ello, el Bachillerato General plantea las diversas Unidades de Aprendizaje Curricular (UAC) y Formaciones Laborales Básicas para que con sus estudiantes egresados y egresadas contribuya al logro de su objetivo específico, el cual radica en la "conformación de una ciudadanía reflexiva, con capacidad de formular y asumir responsabilidades de manera comunitaria, interactuar en contextos plurales y propositivos, trazarse metas y aprender de manera continua y colaborativa".

En este contexto, se presenta la Formación Laboral Básica en **Tecnologías de la Información y la Comunicación** específica del Bachillerato General, con objetivos delimitados acorde a las características del subsistema y de la población a la cual se dirige. El documento se encuentra conformado por apartados mediante los cuales se describe la justificación y los elementos claves para su implementación en el aula.

[1] El cual puede ser consultado a través del siguiente enlace: https://educacionmediasuperior.sep.gob.mx/work/models/sems/Resource/13634/1/images/Documento_Base_rediseno_MCCEMS_Seg__Ed__final.pdf

Programa de Estudios de la Formación Laboral Básica en Tecnologías de la Información y la Comunicación

Semestre	**Tercero, Cuarto, Quinto y Sexto**	
Créditos	**56 totales / 14 por semestre**	
Componente	**De formación laboral**	
Nivel de formación laboral	**Básica**	
Tiempo asignado	**Mediación docente**	**Estudio independiente**
	448 h. totales **112 h. por semestre**	**112 h. totales** **28 h. por semestre**
Sector productivo	**Tecnologías de la Información** [2]	

[2] Acorde al RENEC – Registro Nacional de Estándares de Competencia por Sector Productivo. Disponible en: https://conocer.gob.mx/renec-registro-nacional-de-estandares-de-competencia-por-sector-productivo/

Fundamentación

La Dirección General del Bachillerato (DGB), acorde a la Nueva Escuela Mexicana (NEM) y al Marco Curricular Común de la Educación Media Superior (MCCEMS), y en su responsabilidad de enfrentar tanto los nuevos retos como los objetivos que de estos se desprenden, actualiza el presente Programa de estudios, el cual responde a una visión de educación integral, pertinente, de calidad y excelencia.

Dicho programa forma parte del Componente de Formación Laboral Básico del Bachillerato General, el cual busca ser un espacio vinculado con el sector productivo, permitiendo al estudiantado no solo cumplir con su trayecto educativo, sino construir su proyecto de vida, con mayores posibilidades de inserción en el mercado de trabajo.

Esto atendiendo al mandato constitucional que en materia educativa, con base en la Reforma Constitucional a los artículos 3°, 31° y 73° de la Constitución Política de los Estados Unidos Mexicanos y de la emisión de la Legislación Secundaria, publicadas el 30 de septiembre de 2019 en el Diario Oficial de la Federación, determinan la reorientación del Sistema Educativo Nacional para "garantizar el derecho a la educación con un enfoque de derechos humanos y de igualdad sustantiva, para incidir en la cultura educativa mediante la corresponsabilidad y el impulso de transformaciones sociales dentro de la escuela y en la comunidad"

Bajo este contexto, es que el Componente de Formación Laboral Básico, adquiere mayor relevancia, pues tendrá como ejes rectores:

- **Enfoque en competencias,** que busca desarrollar las capacidades y habilidades necesarias para el desempeño laboral.
- **Enfoque humanista,** que valora y respeta la diversidad y la dignidad del estudiantado, su potencial creativo, su participación, su bienestar integral y su compromiso social.

Estos enfoques se orientan a promover una educación inclusiva, equitativa y de calidad, que favorezca el aprendizaje permanente y el máximo logro de los aprendizajes.

Así pues, el Componente de Formación Laboral Básico, busca desarrollar en el estudiantado competencias laborales básicas, que le permitan aplicar en forma integrada los conocimientos, destrezas, habilidades, actitudes y valores con responsabilidad y autonomía para desenvolverse en contextos específicos del desarrollo personal, académico, social y profesional en situaciones de la vida común, de estudio o trabajo a lo largo de la vida.[3]

[3] Subsecretaría de Educación Media Superior. (2023b). El currículum laboral en la educación media superior. SEP.

Para lograr dicho propósito, lo que a continuación se presenta es una serie de competencias laborales básicas las cuales permitirán al personal docente el abordaje de aprendizajes con la amplitud y profundidad acorde a los diversos contextos.

Es decir, a partir de estas competencias , se diseñarán estrategias de enseñanza aprendizaje que permita a las y los estudiantes ser capaces de conducir su vida hacia su futuro con bienestar y satisfacción, con sentido de pertenencia social, conscientes de los problemas sociales, económicos y políticos que aquejan al país, pero también de su entorno inmediato, dispuestos a participar de manera responsable y decidida en los procesos de democracia participativa y a comprometerse en las soluciones de las problemáticas que los aquejan y que tengan la capacidad de aprender a aprender en el trayecto de su vida.[4]

Para lo cual, es de primordial importancia visualizar que debe existir una articulación contextualizada del Componente Laboral Básico, con el Currículum Fundamental y el Ampliado, para garantizar así la transferencia de los conocimientos, experiencias, habilidades, capacidades, actitudes y valores.

Es decir, esta articulación permitirá:

- **La transversalidad del conocimiento adquirido en el Currículum Fundamental con las competencias laborales básicas requeridas en el mercado laboral.**
- **Fomentar el aprendizaje en contextos diversos, utilizando métodos y estrategias de aprendizaje.**
- **Centrar las necesidades del mercado laboral.**
- **Fomentar la interacción entre la vida educativa y la comunidad, a fin de propiciar la adquisición de conocimientos y competencias, de acuerdo con el desarrollo biopsicosociocultural del estudiantado.**
- **Aprovechar, mediante el Programa Aula, Escuela, Comunidad (PAEC), todas las oportunidades para poner en práctica lo que se ha aprendido; su aplicación no se limitará solo a los recursos sociocognitivos y a las áreas de conocimiento, sino que abarcará los aspectos funcionales (competencias laborales) y recursos socioemocionales.**
- **Mejorar la relación entre la escuela y los sectores productivos.[5]**

[4] Subsecretaría de Educación Media Superior. (2023b). El currículum laboral en la educación media superior. SEP.

[5] Subsecretaría de Educación Media Superior. (2023b). El currículum laboral en la educación media superior. SEP.

Justificación de la Formación Laboral Básica en Tecnologías de la Información y la Comunicación

La Formación Laboral Básica de Tecnologías de la Información y Comunicación pertenece al Recurso Sociocognitivo de Cultura Digital, el cual tiene como propósito promover en el estudiantado el pensar y reflexionar sobre las aplicaciones y los efectos de la tecnología, la capacidad de adaptarse a la diversidad y disponibilidad de los contextos y circunstancias de las y los estudiantes. El propósito es que puedan hacer uso de los recursos tecnológicos (TICCAD, entre otras) para seleccionar, procesar, analizar y sistematizar la información dentro de un marco normativo y de seguridad, y fomenten el uso de dichos recursos de forma responsable en el entorno que lo rodea. Por otra parte, las Tecnologías de la Información y Comunicación se vinculan de manera interdisciplinar tanto con Pensamiento Matemático como con Comunicación, ya que aportan los elementos para la resolución de problemas mediante los algoritmos y la programación.

El propósito general de Tecnologías de la Información y Comunicación es: Desarrollar la capacidad para proponer soluciones a problemas del contexto laboral y escolar, mediante su aplicación de forma creativa e innovadora, con una postura ética y responsable como ciudadana y ciudadano digital.

El uso de las Tecnologías de la Información y Comunicación, desde esta formación laboral básica, destaca el manejo avanzado de las aplicaciones de software y hardware para la resolución de problemas de los diferentes ámbitos de la vida cotidiana, desarrollando los aspectos metodológicos, creativos y comunicativos, sin olvidar un comportamiento propositivo en beneficio personal y dentro de la sociedad.

Tecnologías de la Información y Comunicación busca desarrollar en el estudiantado las competencias profesionales en las áreas de aplicaciones de oficina, los elementos del hardware, las comunicaciones mediante las redes informáticas, el desarrollo de sistemas y el software de diseño, sin olvidar la transversalidad, interdisciplinariedad, vinculación laboral, así como la continuación de sus estudios a nivel superior.

En el contexto curricular de la Formación Laboral Básica de Tecnologías de la Información y Comunicación, el contenido se divide en cuatro módulos que se imparten a partir del tercer semestre con una carga de 7 horas semanales, cada módulo se integra por dos submódulos en los que se busca desarrollar el manejo de aplicaciones de oficina que permiten elaborar documentos electrónicos con características avanzadas utilizando el procesador de textos y la hoja de cálculo, crear y participar en comunidades virtuales para el intercambio de información incluyendo el ámbito educativo, aplicar mantenimiento al equipo de cómputo; para el desarrollo de sistemas con fundamento en las bases de datos y la programación, mediante la creación de páginas web y el software de diseño lograr comunicar ideas e información, en el entorno laboral y escolar.

Todas estas competencias posibilitan a la persona egresada en su incorporación al mundo laboral o bien desarrollar procesos productivos independientes de acuerdo con sus intereses profesionales o las necesidades de su entorno social como asistente en las siguientes áreas: administrativas, soporte técnico, área de sistemas, publicidad y otras, en diferentes instituciones tanto públicas como privadas.

Ubicación de la Formación Laboral Básica Tecnologías de la Información y la Comunicación

Primer semestre	Segundo semestre	Tercer semestre	Cuarto semestre	Quinto semestre	Sexto semestre
Unidad de aprendizaje curricular	Unidad de aprendizaje curricular	Unidad de aprendizaje curricular	Unidad de aprendizaje curricular	Unidad de aprendizaje curricular	Unidad de aprendizaje curricular
La materia y sus interacciones	Conservación de la energía y sus interacciones con la materia	Ecosistemas: interacciones, energía y dinámica	Reacciones químicas: conservación de la materia en la formación de nuevas sustancias	La energía en los procesos de la vida diaria	Organismos: estructuras y procesos. Herencia y evolución biológica
Ciencias Sociales I	Ciencias sociales II		Conciencia histórica I. Perspectivas del México antiguo. Los contextos globales	Conciencia histórica II. México durante el expansionismo capitalista	Conciencia histórica III. La realidad actual en perspectiva histórica
Cultura digital I	Cultura digital II		* Taller de cultura digital		* Temas selectos de matemáticas II
Pensamiento matemático I	Pensamiento matemático II	Pensamiento matemático III	* Temas selectos de matemáticas I		
Lengua y comunicación I	Lengua y comunicación II	Lengua y comunicación III	* Pensamiento literario		
Inglés I	Inglés II	Inglés III	Inglés IV	** (UAC optativas)	** (UAC optativas)
Humanidades I	Humanidades II	Humanidades III	* Espacio y sociedad	** (UAC optativas)	** (UAC optativas)
* Laboratorio de investigación	* Taller de ciencias I	* Taller de ciencias II	Ciencias sociales III	** (UAC optativas)	** (UAC optativas)
* Orientación Educativa: Escolar	* Orientación Educativa: Personal y ocupacional	* Orientación Educativa: Psicosocial	* Orientación Educativa: Vocacional y Profesional	** (UAC optativas)	** (UAC optativas)
* Laboratorio de Ciencias Naturales y Experimentales I	* Laboratorio de Ciencias Naturales y Experimentales II	* Taller de Cultura y Recreación I	* Taller de Cultura y Recreación II	* Taller de Cultura y Recreación III	* Orientación Educativa: Proyecto de Vida y Profesional
- Currículum ampliado	- Currículum ampliado	* Laboratorio de Ciencias Naturales y Experimentales III	* Laboratorio de Ciencias Naturales y Experimentales IV	* Laboratorio de Ciencias Naturales y Experimentales V	* Laboratorio de Ciencias Naturales y Experimentales VI
		Gestión de archivos de texto.	Comunidades virtuales.	Sistemas de Información	Páginas web
		Hoja de cálculo aplicado.	Mantenimiento y redes de cómputo.	Programación.	Diseño Digital
		- Currículum ampliado	- Currículum ampliado	- Currículum ampliado	- Currículum ampliado

COMPONENTE DE FORMACIÓN FUNDAMENTAL
COMPONENTE DE FORMACIÓN FUNDAMENTAL EXTENDIDO OBLIGATORIO
COMPONENTE DE FORMACIÓN FUNDAMENTAL EXTENDIDO
COMPONENTE DE FORMACIÓN LABORAL BÁSICO
COMPONENTE DE FORMACIÓN AMPLIADA

HD: Horas con docente
HI: Horas de estudio independiente
HT: Horas totales
C: Créditos

TOTAL DE HORAS CON DOCENTE SEMANA MES: 187
TOTAL DE HORAS SEMANA MES: 233.75
TOTAL DE HORAS: 3740
TOTAL DE CRÉDITOS: 374

Mapa de la Formación Laboral Básica de Tecnologías de la Información y la Comunicación

Módulo I: Software de aplicación

Submódulo 1	Gestión de archivos de texto. 48 horas de mediación docente 12 horas de estudio independiente 6 créditos
Submódulo 2	Hoja de cálculo aplicado. 64 horas de mediación docente 16 horas de estudio independiente 8 créditos

Módulo II: Hardware y comunicaciones

Submódulo 1	**Comunidades virtuales.** **48 horas de mediación docente** **12 horas de estudio independiente** **6 créditos**
Submódulo 2	Mantenimiento y redes de. cómputo 64 horas de mediación docente 16 horas de estudio independiente 8 créditos

Módulo III: Desarrollo de sistemas.

Submódulo 1	**Sistemas de Información** **48 horas de mediación docente** **12 horas de estudio independiente** **6 créditos**
Submódulo 2	**Programación.** **64 horas de mediación docente** **16 horas de estudio independiente** **8 créditos**

Módulo IV: Software de diseño.

Submódulo 1	Páginas web. 48 horas de mediación docente 12 horas de estudio independiente 6 créditos
Submódulo 2	**Diseño Digital.** **64 horas de mediación docente** **16 horas de estudio independiente** **8 créditos**

Competencias Laborales Básicas

Hacen referencia a la capacidad para aplicar conocimientos, destrezas, habilidades, actitudes y valores en el desarrollo personal, académico, social y profesional en situaciones de la vida común, de estudio o trabajo para que el estudiantado desarrolle la formación fundamental o laboral básica, que les permite desempeñar funciones laborales de nivel dos de competencia, aplicando soluciones a problemas simples en contextos conocidos y específicos.

Tienen validez oficial dentro del Sistema Educativo Nacional (SEN), lo cual se expresa con la emisión del documento que acredita su formación.

Estas competencias se caracterizan por:

- **Pertinencia:** Atiende a las necesidades del sector productivo y son valoradas
- **Relevancia:** Favorece la empleabilidad y emprendimientos productivos, sin disparidades de género, étnicas o exclusión de grupos vulnerables.
- **Coherencia:** Acorde al tipo educativo de media superior.
- **Suficiencia:** Abarca un proceso completo e independiente a las demás funciones que desempeña quien egresa de la formación laboral básica, técnica o tecnológica en el sector productivo.
- **Autonomía:** Faculta para el análisis y toma de decisiones.
- **Responsabilidad:** Capacidad para asumir compromisos orientados al logro de objetivos y metas laborales.
- **Variedad:** Abarca la ejecución de actividades rutinarias – no rutinarias, predecibles – impredecibles – contextos diversos.
- **Complejidad:** Moviliza recursos cognitivos, procedimentales y actitudinales en diferentes niveles para la ejecución de actividades y funciones.

De igual forma es importante señalar, que, para el caso de la Formación Laboral Básica, como se señaló anteriormente, se logrará en el estudiantado el nivel de competencia dos, el cual se caracteriza por:

- Realización de actividades programadas.
- Aplicación de habilidades cognitivas y de comunicación para recibir, transmitir y recordar información.
- Utiliza técnicas, materiales, herramientas y equipamiento que no requieren un nivel de especialización para realizar actividades en contextos conocidos, además del uso de tecnologías de la información y comunicación básicas, actuando con ética, con un enfoque de sostenibilidad y responsabilidad sobre su entorno.[6]

[6] Subsecretaría de Educación Media Superior. (2023b). El currículum laboral en la educación media superior. SEP.

Así mismo, para el caso de la Formación Laboral Básica en Tecnologías de la Información y Comunicación, se considerará el siguiente el perfil de egreso:

1. Integra con la asistencia del personal docente, información digital mediante la creación de documentos electrónicos, empleando software de aplicación, como procesadores de textos y editor de imágenes de manera responsable y creativa en ámbitos laborales, escolares y de la vida cotidiana.
2. Prepara con la supervisión del personal docente, información a través de la manipulación de datos y fórmulas, elaborando gráficos en una aplicación de hoja de cálculo, resolviendo de manera creativa e innovadora, situaciones en diversos ambientes y contextos.
3. Plantea con la asistencia del personal docente el uso, creación y administración de plataformas electrónicas de consulta, comunicación y distribución de contenidos multimedia, proponiendo comunidades virtuales que le permitan comunicarse, favoreciendo su autoaprendizaje en un ambiente innovador en sus diferentes contextos.
1. Desarrolla con el acompañamiento del personal docente, acciones correctivas para los problemas de operación del equipo de cómputo, mediante la aplicación de mantenimiento preventivo y correctivo de acuerdo con las especificaciones del fabricante, prolongando la vida útil del equipo, mostrando responsabilidad e iniciativa en diversos ámbitos.
4. Propone el diseño de sistemas de información, a partir del análisis de las necesidades de las personas usuarias permitiendo junto con la supervisión del personal docente, la solución de problemas de manera responsable e innovadora en diferentes contextos.
5. Construye sistemas de información organizacionales asistido por el personal docente, mediante la codificación y compilación de instrucciones algorítmicas pertinentes utilizando lenguajes de programación y bases de datos para cumplir con los requerimientos de funcionalidad y rendimiento establecidos en el diseño de sistemas de información asumiendo la frustración como parte del proceso en ambientes laborales, educativos y de la vida cotidiana.
6. Construye con el acompañamiento del personal docente, sitios web creativos y funcionales mediante software de diseño web, para transmitir información electrónica diversa a gran escala de manera responsable y empática en contextos laborales, educativos y de la vida cotidiana.
7. Elabora con la asistencia del personal docente, diversos recursos gráficos publicitarios utilizando software de diseño, permitiendo su publicación en medios digitales e impresos para comunicar ideas o emociones aplicables a contextos laborales, escolares y de la vida cotidiana, en un ambiente ético e innovador, mostrando flexibilidad y apertura a diferentes puntos de vista.

Así mismo se señalan algunas normas de competencia laboral vigentes, avaladas por el Consejo Nacional de Normalización y Certificación de Competencias Laborales, que podrán servir de guía para el desarrollo de los módulos:

- CINF0376.01 Elaboración de documentos y comunicación mediante el empleo de las características avanzadas de aplicaciones de cómputo.
- UINF0650.01 Preservar el equipo de cómputo, insumos, información y el lugar de trabajo.

- UINF0947.01 Operar las Herramientas de Cómputo en ambiente de red.
- UINF0948.01 Elaborar documentos de texto mediante el empleo de las características avanzadas de la aplicación de cómputo.
- UINF0949.01 Elaborar hojas de cálculo mediante el empleo de las características avanzadas la aplicación de cómputo.

Proceso de evaluación bajo el enfoque en competencias

La evaluación por competencias es un proceso que permitirá mediante la obtención de evidencias conocer el dominio de conocimientos, habilidades y actitudes socioafectivas desarrolladas por el estudiantado.

Dicho proceso deberá ser formativo e integral, es decir, permitirá visualizar no solo el saber, saber hacer y saber ser, sino también el bagaje histórico y cultural lo cual permitirá al estudiantado la comprensión de la realidad social y laboral de los sectores y de la comunidad para a partir de ello lograr su intervención y aporte.

Para ello lo que a continuación se presentan son los principios que orientarán el proceso de evaluación:

1. **Validez:** debe existir correlación entre los resultados de la evaluación y los resultados esperados en situaciones laborales reales.
2. **Confiabilidad:** producir resultados consistentes al evaluar en momentos diferentes y en diversos contextos.
3. **Accesibilidad:** facilitar el acceso a cualquier persona que pueda ser capaz de demostrar el desarrollo de la competencia.
4. **Comunicación:** dar a conocer previamente las condiciones en que se va a evaluar, y posteriormente, los resultados mediante la retroalimentación.
5. **Equidad:** evitar cualquier práctica discriminatoria, es decir, el estudiantado será evaluado bajo los mismos criterios e indicadores.
6. **Flexibilidad:** adaptarse a diferentes modalidades y opciones de formación, así como a las características y necesidades del estudiantado.[7]

Así pues para poder llevar a cabo lo aquí planteado, se propone la utilización de una amplia gama de instrumentos que permitan visualizar tanto el proceso como el resultado final del aprendizaje del estudiantado, entre los que se encuentran: rúbricas, pruebas de ejecución, portafolios de evidencias, diario de campo o bitácora, organizadores gráficos, ensayos, resolución de ejercicios y problemas, exámenes o pruebas tipo saber, exposición, método de casos, proyectos y debates o discusiones dirigidas, entre otros.

[7] Subsecretaría de Educación Media Superior. (2023b). El currículum laboral en la educación media superior. SEP.

De igual forma, es necesario que la evaluación contemple:

- **Autoevaluación:** cuando el estudiantado valora su desarrollo y la forma en que aprendió.
- **Coevaluación:** a través de la retroalimentación entre pares, fomentando la cooperación, la colaboración, la empatía y la crítica constructiva.
- **Heteroevaluación:** emitida por el personal docente en función de los conocimientos, habilidades, actitudes y valores ponderados en los instrumentos de evaluación.

Finalmente, respecto a los pasos para evaluar las competencias laborales básicas, se presenta a continuación una propuesta elaborada por la Subsecretaría de Educación Media Superior (2023b), la cual ilustra la ruta a seguir y constituye una referencia para el personal docente.

Fuenterelaboracion propia

Rol del estudiantado

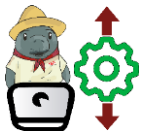
El rol del estudiantado en el proceso educativo no se limita simplemente a recibir información y repetirla, sino que debe ser un agente activo en la construcción de su propio conocimiento y de su identidad. En este sentido, no sólo se trata de aprender a leer y escribir; implica aprender a narrar y comprender su propia vida, tanto como autor o autora de su historia personal, como testigo de su contexto social y cultural. Este proceso es fundamental para que el estudiantado se convierta en un sujeto consciente y crítico de su realidad.

La educación es un motor de transformación social, pero también puede perpetuar las desigualdades existentes al tratar a todos y todas por igual sin considerar la diversidad inherente al estudiantado. La educación debe empoderarles, dándoles las condiciones necesarias para reconocer y cuestionar las desigualdades que les rodean.

Si las y los estudiantes son insertados en una educación que no considera su clase, sexo, género, etnia, lengua, cultura, capacidad, condición migratoria, religión o cualquier otro aspecto de su identidad, es muy probable que se apropien de la idea de que "la escuela no es para ellos y ellas", ya que se enfrentarían constantemente a comentarios o actitudes que les califican de incapaces, ignorantes, indolentes o inútiles terminando por creerlo y asumirlo como verdad. Esta autodesvalorización es una barrera significativa para su desarrollo ya que puede llevar a creer que el conocimiento y la sabiduría pertenecen únicamente a las y los "profesionales" y no reconocen el valor de su propio conocimiento y experiencia.

El rol de las y los estudiantes, entonces, debe ser el de un sujeto activo que desafía y transforma estas narrativas opresivas que fomentan las desigualdades. Debe aprender a valorar su propia voz y experiencia, y a reconocer su capacidad para conocer y transformar su realidad. La educación debe ser un proceso liberador que les permita verse a sí mismos o mismas como agentes de transformación social, capaces de escribir su propia historia y de participar activamente en la construcción de una sociedad más justa y humana.

Simbología

Ícono	Descripción	Ícono	Descripción
	**Lectura**: Indica que se realizará la lectura que contiene información de los conocimientos básicos del Programa de Estudio.		**Evaluación Diagnóstica**: Presenta el momento para llevar a cabo la evaluación diagnóstica del submódulo.
	**Material audio visual**: Representa recursos adicionales al contenido de los conocimientos básicos del Programa de Estudio a través de un video mostrado por el docente, indicado por QR.		**Instrumento de evaluación**: Documento en el que tanto el estudiante como el docente observan los criterios con los que se evaluará el producto a realizar.
	**Actividad**: Es momento de realizar una actividad teórica – escrita que contribuye a evidenciar el propósito del aprendizaje esperado de los conocimientos básicos del Programa de Estudio.		**Encuadre**: Presenta cada aspecto y el porcentaje de calificación con el que se evaluará el submódulo.
	**Práctica**: Hagamos la práctica guiada, que contribuye a la aplicación de los saberes obtenidos con relación al conocimiento básico del Programa de Estudio, se acompaña con el instrumento de evaluación correspondiente.		**Dosificación**: Muestra las fechas en que los conocimientos básicos del Programa de Estudio se van a desarrollar.
	**Docente explica**: Se sugiere que esta sección el docente profundice los conocimientos para adecuarlos a su contexto.		**PEC:** Proyecto Escolar Comunitario

Ícono	Descripción	Ícono	Descripción
	**Actividad SIGA**: Indica el producto que se contempla para la plataforma SIGA.		

TEMARIO

SUBMÓDULO 1

INTRODUCCIÓN A LOS SISTEMAS DE INFORMACIÓN

DISEÑO DE LOS SISTEMAS DE INFORMACIÓN

MODELO RELACIONAL DE BASE DE DATOS

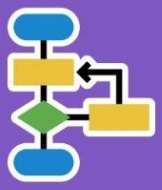
LENGUAJE SQL PARA GESTIÓN DE BASE DE DATOS

DISEÑO Y DESARROLLO DE BASE DE DATOS

SUBMÓDULO 2

LÓGICA DE PROGRAMACIÓN

Algoritmo
Diagramas de flujo
Pesudocódigo
Decisiones
Ciclos



LENGUAJE DE PROGRAMACIÓN

Tipos de lenguajes
Metodología de programación
Estructurado | Orientado a objetos

LENGUAJE DE ALTO NIVEL PARA PROGRAMACIÓN

Entorno de desarrollo
Variables
Operadores
Constantes
Palabras reservadas
Sentencias de decisión
Estructura de condición y repetición

LISTAS

FUNCIONES Y LIBRERÍAS

BASE DE DATOS

SUBMÓDULO I

Sistemas de Información

BASES DE DATOS

CLOUD COMPUTING

ANÁLISIS DE INFORMACIÓN

SEGURIDAD DE DATOS

REDES Y CONECTIVIDAD

Propósito del Módulo

Plantea soluciones críticas y responsables mediante la metodología de desarrollo de software para demostrar eficiencia en el manejo de base de datos y software de programación de alto nivel, que sean aplicables a necesidades de una empresa o institución para el tratamiento de información.

Sugerencia de evidencias de Evaluación

En equipos de tres integrantes, el estudiantado aplicará la metodología de cascada para diseñar un Sistema de Información para Reservas de Eventos, el cual permitirá a los asistentes:

- Registrar sus datos personales.
- Elegir asientos específicos.
- Seleccionar diferentes horarios y salas.
- Realizar pagos en distintos formatos.

Competencia Laboral Básica Para Desarrollar

Propone el diseño de sistemas de información, a partir del análisis de las necesidades de los usuarios, permitiendo la solución de problemas de manera responsable e innovadora en diferentes contextos.

Submódulo 1. Dosificación programática Sistemas de información

Formación Laboral Básica: Tecnologías de la información y la comunicación
Módulo III: Desarrollo de Sistemas **Submódulo I:** Sistemas de Información **Clave:** C5SI
Semestre: 5to. Turno: Matutino/Vespertino **Período:** 2026 – 2027A

Submódulo	Momento	Tiempo (minutos)	Conocimientos	Semana	Fecha inicio	Observaciones
SUBMÓDULO I. SISTEMAS DE INFORMACÓN	Apertura	50 min 300 min	Bienvenida Encuadre Evaluación Diagnóstica **Lectura 1. Del dato a la sesión. Fundamentos de los Sistemas de Información**	**1**	31 de agosto – 04 de septiembre de 2026	Inicio del semestre 31 de agosto
	Desarrollo	350 min	**Lectura 2. Diseño de Sistemas de Información del problema a la solución real.**	**2**	07 – 11 de septiembre de 2026	
		350 min	**Lectura 3. Modelo relacional de bases de datos.**	**3**	14 – 18 de septiembre de 2026	
		350 min	**Lectura 4. "SQL sin HACKS"**	**4**	21 – 25 de septiembre de 2026	
		350 min	**Lectura 5 Diseño y Desarrollo de Base de datos.**	**5**	28 de septiembre – 02 de octubre de 2026	
		350 min	**LECTURA 6. Implementación en un gestor de base de datos (MySQL, PostgreSQL, SQLite**	**6**	05 – 09 de octubre de 2026	
	Cierre	350 min	**Situación didáctica "Haz que tus ideas cuenten"**	**7**	12 – 16 de octubre de 2026	SIGA

Submódulo 1. Encuadre Sistemas de información

No.	Actividades	Ponderación
1	Actividad 1. Juego de Roles Metodología de cascada	5%
2	Actividad 2. Modelando mis datos (DER)	5%
3	Actividad 3.1. Ejercicio teórico Modelo relacional: Tablas, claves y relaciones	5%
4	Actividad 3.2. Ejercicio Práctico Construye un modelo de BD para la Reservación de	5%
5	Actividad 4. Gestión de Videojuegos	5%
6	Actividad 5.1. Infografía por etapas	5%
	TOTAL	30%

No.	Practicas	Ponderación
	Práctica 1. Diseña tu propia interfaz de usuario (Opción A o B)	5%
1	Práctica guiada 5.2. Mis relaciones y yo	5%
2	Práctica 6.2. Que SQL el código	10%
	TOTAL	20%

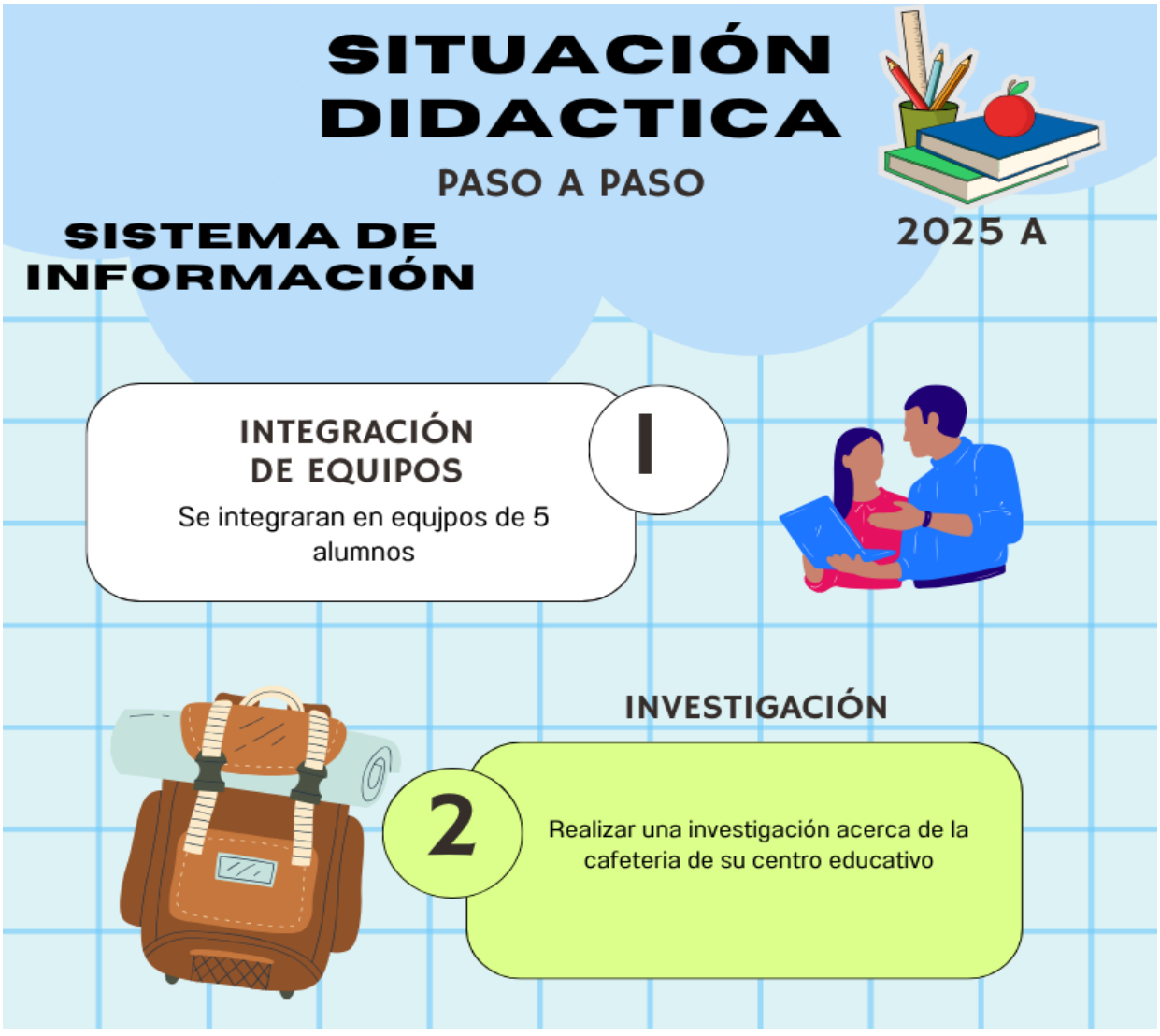
No.	Situación didáctica	Ponderación
1	Situación didáctica. ""Haz que tus ideas cuenten"	30%

No.	Evaluación escrita	Ponderación
1	Examen	20%

Submódulo 1. Situación didáctica "Haz que tus ideas cuenten"

Propósito de la Situación didáctica:	Desarrollar un sistema de información que permita la gestión de reservas de espacios en eventos escolares, mediante el análisis de necesidades, el diseño de soluciones y la implementación básica con herramientas de programación, favoreciendo el trabajo colaborativo y la resolución de problemas del contexto.
Sugerencia de Evidencia de Evaluación:	En equipos de cinco integrantes, el estudiantado aplicará la metodología de cascada para diseñar un Sistema de Información para Reservas de Eventos, el cual permitirá a los asistentes: • Registrar sus datos personales. • Elegir asientos específicos. • Seleccionar diferentes horarios y salas. • Realizar pagos en distintos formatos.
Problema de Contexto:	El estudiantado de quinto semestre, en coordinación con sus docentes, se encuentra organizando diversos eventos en el auditorio de su plantel. Ante esta situación, surge la necesidad de contar con un sistema que permita a las y los participantes seleccionar los espacios disponibles dentro de la sala, de manera similar a los sistemas utilizados en cines y eventos con control de acceso. Actualmente, la asignación de lugares se realiza de manera manual, lo que genera desorganización, duplicidad de espacios y dificultad para el control de asistentes. Por ello, el grupo solicita el apoyo del docente de Tecnologías de la Información y la Comunicación para adquirir los conocimientos necesarios que les permitan diseñar y desarrollar un sistema de información que facilite la gestión, organización y control de eventos masivos.
Conflicto Cognitivo:	¿Cómo se puede diseñar un sistema que permita a los usuarios seleccionar su lugar de manera organizada y eficiente? ¿Qué elementos son necesarios para construir un sistema de información funcional? ¿Cómo se puede almacenar y consultar la información de los asistentes? ¿Qué herramientas tecnológicas permiten desarrollar este tipo de soluciones?

Propósito: Elaborar en equipos de 5 integrantes a través de la técnica de Aprendizaje Basado en Problemas (ABP) un *Sistema de gestión informático (Base de datos)* tomando en cuenta las necesidades de la microempresa abordar, aplicando la metodología en cascada de desarrollo de software para la creación de una base de datos, considerando el instrumento de evaluación propuesto y socializarlo ante el grupo.



ORGANIZA LA INFORMACIÓN

3

·Recopilación de datos para análisis, preferencias y tendencias de consumo de productos por parte de los estudiantes.
·Optimización de la producción y el inventario.
·Elaboración de informes de ventas.

CREANDO MATERIAL

4

Con el modelado de datos, modelo entidad relación y diccionario de datos, realizado de la microempresa seleccionada vamos a crear una base de datos, de acuerdo a la lista de cotejo

INTEGRACIÓN

5

Se creara una base de datos, siguiendo la lista de cotejo para su elaboración, en donde debe contener:

- Tablas
- Relaciones y Cardinalidad de las tablas
- Consultas

Submódulo 1. Evaluación diagnóstica

Instrucciones: *Lea las siguientes preguntas y subraye la respuesta correcta.*

NOMBRE: _____ **GRUPO:** _____

1. ¿Entre los componentes de un sistema es aquel que hace referencia a hechos e información recopilados y utilizados por el sistema?

a. Software b. Datos c. Procedimientos

2. ¿Es una de las aplicaciones del mundo real, donde las empresas que adoptan un sistema de información pueden responder rápidamente a cambios del mercado, mejorar su relación con los clientes y mantenerse competitivos?

a. Eficiencia operativa b. Toma de decisiones c. Ventaja competitiva

3. ¿Cuál es el propósito principal de un diagrama Entidad-Relación (ER) en el modelado de datos?

a. Representar la estructura de una base relacional b. Modelar el comportamiento de un sistema c. Describir la interfaz de usuario de una aplicación

4. ¿Cuál es el objetivo principal de la usabilidad en el diseño de interfaz de usuario?

a. Hacer que la interfaz sea lo más atractivo posible b. Reducir el tiempo de desarrollo de la aplicación c. Facilitar la interacción del usuario con el sistema de manera eficiente y efectiva

5. ¿Qué representa una tabla en el modelo Relacional?

a. Una lista de comandos SQL b. Una colección de datos organizados en filas y columnas c. El tipo de relación entre tablas

6. ¿Cuál es la característica principal de una clave Primaria en una base de datos relacional?

a. Tener valores duplicados b. Puede ser nula c. Debe ser única y no nula

7. ¿Es un lenguaje estándar para la creación y manipulación de base de datos Relacionales?

a. MySQL b. HTML c. Python

8. ¿Qué sentencia se utilizar para obtener resultados de consultas con datos específico, que se especifica con una condición y el uso de operadores lógicos y de comparación?

a. SELECT b. FROM c. WHERE

9. ¿En esta fase o etapa del diseño de las Base de datos se describen los datos que se almacenarán en la base de datos y las restricciones a las que se someterán?

a. Recopilación y análisis de datos b. Diseño conceptual c. Diseño lógico

10. ¿es una forma de representar la BD en el modelado de Base de Datos, que representa visualmente los objetos o entidades importantes de nuestra BD, así como sus atributos, las relaciones entre entidades y su cardinalidad?

a. Diagrama Entidad-Relación b. Diagrama de flujo de datos c. Diagrama de procesos

Lectura 1. Del dato a la decisión Fundamentos de los Sistemas de Información

Instrucciones: *Realiza el análisis de la siguiente información, posteriormente, completa la Actividad 1. Juego de roles: Metodología en cascada.*

Concepto y componentes de un sistema de información.

Un sistema se entiende como un grupo de elementos que trabajan juntos, interactuando entre sí para alcanzar un objetivo compartido según Senn (2001). Siguiendo esta idea, podemos definir un sistema como la organización de partes que se influyen mutuamente y dependen unas de otras, unidas y relacionadas para formar una entidad compleja. Entonces, los sistemas son conjuntos organizados de elementos interrelacionados que trabajan en conjunto para alcanzar un objetivo común. En el contexto de los sistemas de información, estos combinan tecnología, personas y procesos para recopilar, procesar, almacenar y distribuir información, (Classter, 2005). Su propósito principal es apoyar la toma de decisiones, coordinar actividades y mejorar la eficiencia en organizaciones o instituciones. Por ejemplo, los sistemas de información de control escolar que usan en tu escuela y que ayudan a gestionar datos académicos y administrativos, optimizando recursos y mejorando la experiencia educativa. Los sistemas de información como se mencionó con anterioridad involucran a las personas y la forma en que se utilizan, es por ello por lo que en la vida cotidiana los sistemas de información están presentes en situaciones como:

- **Control de calificaciones de una escuela**
- **Sistema para reserva de eventos**
- **Los sistemas de registro en hospitales**

En todos estos casos el objetivo primordial es transformar datos en información útil.

De acuerdo con (Kendall y Kendall, 2011), los sistemas de información están diseñados para facilitar las operaciones, la gestión y la toma de decisiones dentro de una organización. Incluyen no solo la tecnología de la información y la comunicación (TIC) que una organización utiliza, sino también la forma en que las personas interactúan con esta tecnología para apoyar los procesos de negocio. Ahora dentro de las organizaciones el análisis y diseño de sistemas se refiere al "proceso de examinar la situación de una empresa con el propósito de mejorarla con métodos y procedimientos más adecuados", (Senn, 2001, p. 11). Por ejemplo, en una tienda de ropa, un sistema de información es fundamental para gestionar eficientemente las operaciones del almacén. Este sistema permite un control preciso del inventario, lo que significa que se pueden manejar fácilmente tallas y colores ilimitados para cada modelo de prenda. Además, proporciona información en tiempo real sobre el stock disponible, lo que facilita la toma de decisiones sobre reposiciones y pedidos. Por ejemplo, el sistema puede configurarse para generar pedidos automáticos cuando el stock de un artículo cae por debajo de un nivel mínimo, asegurando que siempre haya suficiente mercancía para satisfacer la demanda.

De acuerdo con Wallace (2025), entre los componentes de un sistema de información encontramos:

Figura 1: Elaboración propia

Importancia y aplicaciones en el mundo actual.

Los sistemas de información (SI) son herramientas esenciales en el mundo moderno, ya que permiten a las organizaciones recopilar, procesar y analizar datos para tomar decisiones informadas. Su importancia radica en varios aspectos claves, tal y como lo señala (Shai, 2024):

Eficiencia operativa

- Los SI optimizan procesos internos, reducen errores y automatizan tareas rutinarias, lo que mejora la productividad y permite a las empresas centrarse

Toma de decisiones

- Proveen datos precisos y en tiempo real que ayudan a identificar tendencias, evaluar alternativas y prever resultados, facilitando decisiones más acertadas

Ventaja Competitiva

- Empresas que adoptan SI pueden responder rápidamente a cambios del mercado, mejorar su relación con los clientes y mantenerse competitivas

En el mismo sentido, los sistemas de información tienen aplicaciones prácticas en diversos sectores como, por ejemplo:

Negocios y comercio

- Gestión de clientes: Los sistemas CRM (Customer Relationship Management) ayudan a rastrear interacciones con clientes, mejorar el servicio al cliente y personalizar estrategias de marketing
- Gestión financiera: Los SI facilitan la contabilidad, el control de inventarios y la planificación financiera

Educación

- Plataformas como Moodle o Blackboard permiten la creación de aulas virtuales, haciendo posible el aprendizaje remoto y flexible

Salud

- Los registros médicos electrónicos (EHR) centralizan la información del paciente, mejorando la atención médica y permitiendo consultas remotas mediante telemedicina

Es por ello por lo que antes de proponer el diseño de un sistema de información conozcamos el ciclo de vida del desarrollo de sistemas que de acuerdo con (Kendall y Kendall, 2011), está formado por las siguientes etapas:

Figura 2: Elaboración propia

A continuación, describiremos de manera breve las etapas del ciclo de vida del desarrollo de sistemas a fin de saber, que esta permite diseñar soluciones tecnológicas de manera organizada con el propósito de reducir errores y mejorar la toma de decisiones.

Identificación de problemas, oportunidades y objetivos

Las actividades de esta fase consisten en entrevistar a los encargados de coordinar a los usuarios, sintetizar el conocimiento obtenido, estimar el alcance del proyecto y documentar los resultados. El resultado de esta fase es un informe de viabilidad que incluye una definición del problema y un resumen de los objetivos.

Determinación de los requerimientos de información

Los implicados en esta fase son el analista y los usuarios, por lo general trabajadores y gerentes del área de operaciones. El analista de sistemas necesita conocer los detalles de las funciones del sistema actual: el quién (la gente involucrada), el qué (la actividad del nego cio), el dónde (el entorno donde se desarrollan las actividades), el cuándo (el momento oportuno) y el cómo (la manera en que se realizan los procedimientos actuales)

Análisis de las necesidades del sistema

Durante esta fase el analista de sistemas analiza también las decisiones estructuradas que se hayan tomado. Las decisiones estructuradas son aquellas en las cuales se pueden de terminar las condiciones, las alternativas de condición, las acciones y las reglas de acción. Existen tres métodos principales para el análisis de decisiones estructuradas: español estructurado, tablas y árboles de decisión.

Diseño del sistema recomendado

En esta fase el analista utiliza la información recopilada en las primeras fases para realizar el diseño lógico del sistema de información. El analista diseña procedimientos precisos para la captura de datos que aseguran que ios datos que ingresen al sistema de información sean correctos. Además, el analista facilita la entra da eficiente de datos al sistema de información mediante técnicas adecuadas de diseño de formularios y pantallas.

Desarrollo y documentación del software

En la quinta fase del ciclo de vida del desarrollo de sistemas, el analista trabaja de manera conjunta con los programadores para desarrollar cualquier software original necesario. Durante esta fase el analista también trabaja con los usuarios para desarrollar documentación efectiva para el software, como manuales de procedimientos, ayuda en línea y sitios Web que incluyan respuestas a preguntas frecuentes

Pruebas y mantenimiento del sistema

Antes de poner el sistema en funcionamiento es necesario probarlo. Es mucho menos cos toso encontrar los problemas antes que el sistema se entregue a los usuarios. Una parte de las pruebas las realizan los programadores solos, y otra la llevan a cabo de manera conjunta con los analistas de sistemas.

Implementación y evaluación del sistema

En esta fase se capacita a los usuarios en el manejo del sistema. Parte de la capacitación la imparten los fabricantes, pero la supervisión de ésta es responsabilidad del analista de sistemas. Además, el analista tiene que planear una conversión gradual del sistema anterior al actual. Este proceso incluye la conversión de archivos de for matos anteriores a los nuevos, o la construcción de una base de datos, la instalación de equipo y la puesta en producción del nuevo sistema.

Metodología de cascada

Es importante destacar en el mismo sentido que también existe una metodología de desarrollo secuencial y lineal llamada: Metodología de cascada, aquí cada fase debe terminarse antes de pasar a la siguiente y normalmente se trabaja con una documentación detallada desde el inicio. El desarrollo del modelo se atribuye al teórico de la informática Winston W. Royce. Sin embargo, Royce no es el inventor de este modelo.

En la práctica, se aplican diversas versiones del modelo. Los más habituales son los modelos que dividen los procesos de desarrollo en cinco fases. En ocasiones, las fases 1, 2 y 3 definidas por Royce se integran en una sola fase de proyecto a modo de análisis de los requisitos.

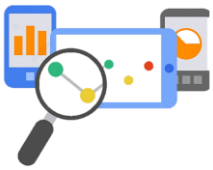
1. **Análisis:** planificación, análisis y especificación de los requisitos.
2. **Diseño:** diseño y especificación del sistema.
3. **Implementación:** programación y pruebas unitarias.
4. **Verificación:** integración de sistemas, pruebas de sistema y de integración.
5. **Mantenimiento:** entrega, mantenimiento y mejora.

La siguiente imagen explica por qué el procedimiento lineal se denomina metodología en cascada.

Figura 3: Tomada de IONOS Digital Guide (2019, March 21)
(https://www.ionos.mx/digitalguide/fileadmin/DigitalGuide/Screenshots_2019/wasserfallmodell-ES-1.jpg)

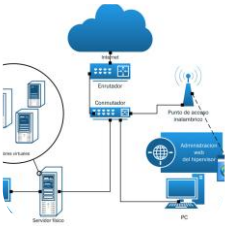
De acuerdo con el sitio: Equipo editorial de IONOS (2019), el modelo en cascada de cinco niveles, basado en las propuestas de Winston W. Royce, divide los procesos de desarrollo en las siguientes fases de proyecto: análisis, diseño, implementación, verificación y mantenimiento.

Metodología de cascada (Waterfall)

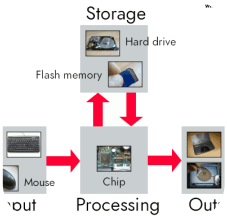


Análisis: Todo proyecto de software comienza con una fase de análisis que incluye un estudio de viabilidad y una definición de los requisitos.

Diseño: sirve para formular una solución específica en base a las exigencias, tareas y estrategias definidas en la fase anterior. En esta fase, los desarrolladores de software se encargan de diseñar la **arquitectura de software**, así como un **plan de diseño detallado del mismo**,



En la fase de **implementación**, el proyecto de software se traduce al correspondiente lenguaje de programación. Los diversos componentes se desarrollan por separado, se comprueban a través de las **pruebas unitarias** y se integran poco a poco en el producto final

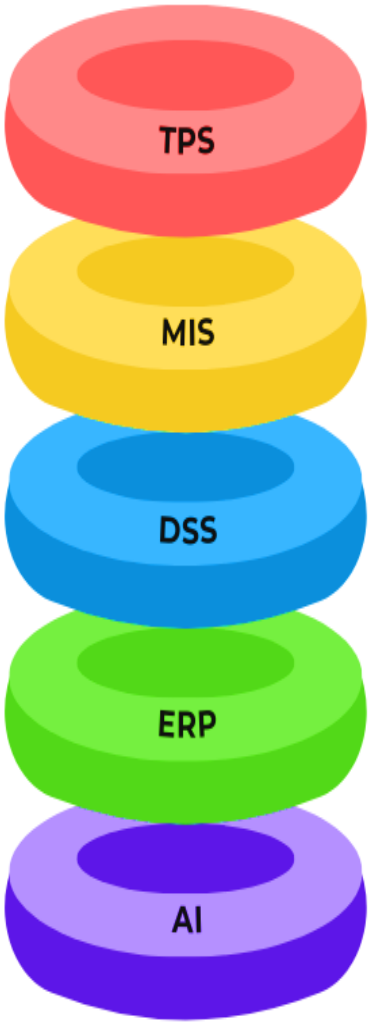


Las **pruebas de aceptación/Verificación** desarrolladas en la fase de análisis permiten determinar si el software cumple con las exigencias definidas con anterioridad. Aquellos productos de software que superan con éxito las pruebas beta están listos para su lanzamiento.

Una vez que la fase de **prueba/Mantenimiento** ha concluido con éxito, se autoriza la **aplicación productiva** del software. La última fase del modelo en cascada incluye la **entrega**, el **mantenimiento** y la **mejora del software**.

Figura 4: Elaboración propia

Tipos de SI



SISTEMA DE PROCESAMIENTO DE TRANSACCIONES

Los sistemas de procesamiento de transacciones son sistemas de información computarizada creados para procesar grandes cantidades de datos relacionadas con transacciones rutinarias de negocios, como las nóminas y los inventarios.

SISTEMA DE INFORMACIÓN GERENCIAL

Los sistemas de información gerencial, Los MIS son sistemas de información computarizados cuyo propósito es contribuir a la correcta interacción entre los usuarios y las computadoras

SISTEMA DE APOYO A LA TOMA DE DECISIONES

Los sistemas de apoyo a la toma de decisiones, constituyen una clase de alto nivel de sistemas de información computarizada. Los DSS coinciden con los sistemas de información gerencial en que ambos dependen de una base de datos para abastecerse de datos.

SISTEMA DE PLANEACIÓN DE RECURSOS EMPRESARIALES

Muchas organizaciones consideran los beneficios potenciales que se derivan de la integración de los diversos sistemas de información que existen en los diferentes niveles administrativos, con funciones dispares. Esta integración es precisamente el propósito de los sistemas de planeación de recursos empresariales.

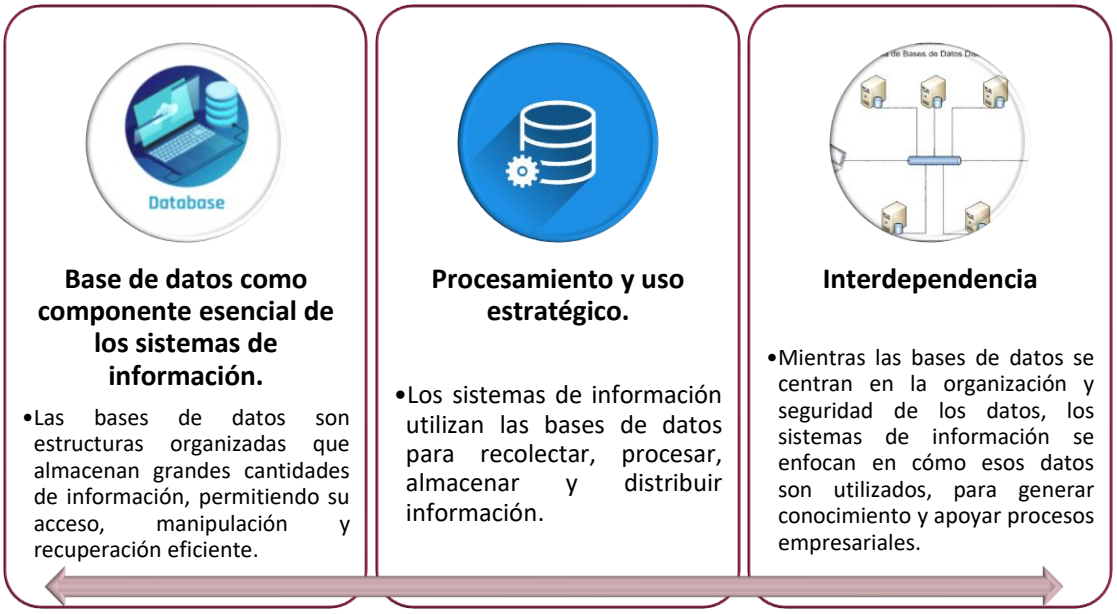
SISTEMAS EXPERTOS E INTELIGENCIA ARTIFICIAL

Los sistemas expertos utilizan las técnicas de razonamiento de la AI para solucionar los problemas que les plantean los usuarios de negocios (y de otras áreas).

Figura 5: Elaboración propia

Relación entre sistemas de información y bases de datos.

Los sistemas de información y las bases de datos están muy vinculados, ya que ambos son esenciales para manejar los datos de manera eficiente dentro de una organización. Según Kendall y Kendall (2011), una base de datos es un repositorio central de información que puede ser utilizada por distintos usuarios y aplicaciones. El componente principal de una base de datos es el sistema de gestión de bases de datos (DBMS), que facilita tareas como la creación, modificación, actualización, consulta y presentación de los datos. El administrador de bases de datos se encarga de asegurar que la base de datos cumpla sus propósitos, como permitir el acceso compartido de los datos para diferentes aplicaciones, mantener la precisión y coherencia de la información, y adaptarse a las nuevas necesidades de los usuarios a medida que evolucionan.



Estos componentes incluyen la entrada de datos, el procesamiento de la información, el almacenamiento de la información y la salida de datos. En este contexto, las bases de datos juegan un papel crucial en el almacenamiento de la información, permitiendo que los sistemas de información accedan y manipulen los datos necesarios para apoyar las operaciones y la toma de decisiones en una organización, Senn (2001). Por ejemplo, en el sector financiero, las bases de datos almacenan registros de transacciones y cuentas, mientras que el sistema de información analiza estos datos para generar reportes financieros o detectar fraudes, (SISE, 2024).

Actividad 1. Juego de Roles Metodología de cascada

Instrucciones: *De manera individual, después de realizar la lectura anterior, da respuesta a las etapas de la metodología de cascada que se solicitan, a partir del siguiente planteamiento:*

En la comunidad donde vives, se ha inaugurado un nuevo centro de entretenimiento llamado "Eventos Tabasco", el cual ofrece diferentes actividades como conciertos, obras de teatro, conferencias y proyecciones de cine.

Actualmente, la administración del lugar enfrenta diversos problemas, ya que todo el proceso de reservación se realiza de manera manual. Los asistentes deben acudir personalmente para registrarse, elegir sus asientos y pagar su entrada. Esto ha generado situaciones como:

- **Duplicidad en la asignación de asientos**
- **Pérdida de información de los asistentes**
- **Largas filas y tiempos de espera**
- **Errores en los registros de pago**
- **Dificultad para organizar eventos en diferentes horarios y salas**

Ante esta problemática, los encargados del centro han decidido implementar un **sistema de información digital** que permita mejorar la organización y brindar un mejor servicio a los usuarios.

Considerando la información anterior, en equipos simulen las primeras etapas del modelo de cascada y empieza a desarrollar una propuesta para solucionar dicha problemática.

- **Fase de análisis (planificación, análisis y especificación de los requisitos)**

Responde:

1. **¿Cuál es el problema principal?**
2. **¿Qué situaciones afectan a los usuarios?**
3. **¿Quiénes son los usuarios del sistema?**

- **Fase de diseño: (diseño y especificación del sistema)**

Responde:

1. **¿Qué necesita hacer el sistema?**
2. **¿Qué información debe registrar?**
3. **¿Qué funciones debe tener?**

- **Fase de implementación (programación, pruebas unitarias y componentes de un sistema)**

Identifica:

Componentes	Ejemplos
Hardware	
Software	
Personas	
Datos	
Procedimientos	

Referencias

AI Journals. (2025). Understanding information systems for business success. https://aijourn.com/understanding-information-systems-for-business-success/

Classter. (2025). Cómo funcionan los sistemas de información estudiantil y cómo aprovechar su potencial. Recuperado de https://www.classter.com/es/blog/edtech-es/como-funcionan-los-sistemas-de-informacion-estudiantil-y-como-aprovechar-su-potencial/

Ecosistemas.win. (2024). Qué son las bases de datos y los sistemas de información. https://ecosistemas.win/que-son-las-bases-de-datos-y-los-sistemas-de-informacion/

ECPI University. (n.d.). How do information systems help organizations thrive? Rhttps://www.ecpi.edu/blog/how-do-information-systems-help-organizations-thrive edición. PEARSON EDUCACIÓN, México

El modelo en cascada: desarrollo secuencial de software. (2019, March 21). IONOS Digital Guide. https://www.ionos.mx/digitalguide/paginas-web/desarrollo-web/el-modelo-en-cascada/

Instituto SISE. (2024, mayo 13). ¿Para qué sirve una base de datos? Recuperado de https://www.sise.edu.pe/blog/utilidad-bases-datos-importancia/

Kendall, K. E. y Kendall, J. E. (2011). Análisis y diseño de sistemas. ISBN: 978-607-32-0577-1. 8va

Senn James A. (2001) Análisis y diseño de sistemas de información. Segunda edición. ISBN:978968422914.

Shai, S. (2024). The role of information technology in modern society. LinkedIn. Recuperado el 22 de marzo de 2025, de https://www.linkedin.com/pulse/role-information-technology-modern-society-sidharth-shai-1bb2e

Lectura 2. Diseño de sistemas de información Del problema a la solución real

Instrucciones: *Realiza la lectura correspondiente a diseño de sistemas de información, posteriormente, completa la* ***Actividad 2 Diseña tu propia interfaz de usuario, para un sistema escolar.***

Principios de diseño de sistemas.

Como lo menciona Collado & Vila (2024), el diseño de sistemas de información se basa en varios principios fundamentales que aseguran la creación de sistemas informáticos eficientes, efectivos y adaptables. Es como construir una casa: necesitas buenos planos y materiales para que todo funcione bien.

Algunos principios importantes:

Simplicidad:	Mantén las cosas simples. Un sistema sencillo es más fácil de entender y usar.
Modularidad:	Divide el sistema en partes pequeñas que puedan funcionar por sí solas. Es como tener diferentes habitaciones en una casa.
Escalabilidad:	Asegúrate de que el sistema pueda crecer sin problemas. Imagina que tu casa necesita más habitaciones porque tu familia crece.
Seguridad:	Protege la información para que nadie pueda acceder sin permiso. Es como tener cerraduras en las puertas.
Usabilidad:	Haz que el sistema sea fácil de usar. Piensa en cómo hacer que las habitaciones de tu casa sean cómodas y accesibles.

Modelado de datos (diagramas ER, UML)

El **modelado de datos** es como crear un mapa detallado que muestra cómo se relacionan y conectan las diferentes partes de un sistema, ya sea una aplicación, una base de datos o cualquier herramienta que organice información. En palabras de Silberschatz, et al. (2020), es el proceso de representar y organizar la información de un sistema, definiendo sus elementos, características y relaciones, con el fin de diseñar bases de datos eficientes.

Imagina que estás diseñando un videojuego o un proyecto escolar; Necesitas saber cómo se enlazan los personajes, los niveles, los puntajes y las configuraciones. En lugar de hacerlo todo al azar, haces un plan visual y estructurado que te ayuda a entender cómo funciona todo junto.

Por ejemplo, piensa en una red social. Un modelo de datos mostraría cómo los usuarios están conectados a sus publicaciones, comentarios y amigos. Este modelo responde preguntas como: ¿Cómo sabe la red social qué publicaciones pertenecen a quién? ¿Cómo conecta a los usuarios con sus amigos? Es como hacer un esquema o un dibujo que organiza toda esta información para que sea más fácil de entender y manejar.

Proceso de modelado de datos

El modelado de datos anima a los interesados a revisar de manera detallada el manejo y la conservación de la información. Las metodologías de modelado de datos tienen distintas normativas que determinan los iconos utilizados para representar información, la presentación de los modelos y la forma en que se comunican los objetivos empresariales. Todos los métodos ofrecen flujos de trabajo estructurados que incluyen una serie de tareas a realizar de forma cíclica. Usualmente, estos flujos de trabajo son de esta manera:

Identificar las entidades.	La etapa inicial del modelado de datos empieza con la identificación de los elementos, eventos o ideas que están representados en los datos que se desean modelar. Cada entidad necesita ser coherente y diferenciada lógicamente de las demás.
Identificar propiedades clave de cada entidad.	Cada clase de entidad se puede distinguir de las demás ya que posee una o más características únicas, conocidas como atributos. Por ejemplo, una entidad denominada "cliente" podría tener atributos como nombre, apellido, número de teléfono y saludo, mientras que una entidad llamada "dirección" podría tener el nombre y número de una calle, así como la ciudad, estado, país y código postal.
Identificar relaciones entre entidades.	La versión inicial de un modelo de datos describirá la naturaleza de las conexiones que cada entidad tiene con las demás. En el caso anterior, cada cliente "reside en" una dirección. Si se amplía ese modelo para incluir una entidad llamada "pedidos", cada pedido también se enviaría y facturaría a una dirección. Estas interrelaciones generalmente se documentan usando el lenguaje de modelado unificado (UML).

Asignar atributos a entidades por completo.	Esto asegurará que el modelo refleje cómo la organización utilizará la información. Existen varios patrones formales de modelado de datos que son ampliamente utilizados. Los desarrolladores enfocados en objetos a menudo utilizan patrones de análisis o diseño, mientras que los interesados de otros sectores comerciales pueden recurrir a otros modelos.
Asignar claves según sea necesario y decidir un grado de normalización que equilibre la necesidad de reducir la redundancia con los requisitos de rendimiento.	La normalización es un método para organizar modelos de datos (y las bases de datos que representan) en el cual se asignan identificadores numéricos, conocidos como claves, a grupos de datos para mostrar las relaciones entre ellos sin repetir la información. Así, si a un cliente se le asigna una clave, esa clave puede relacionarse tanto con su dirección como con su historial de pedidos, evitando la repetición de dicha información en la tabla de clientes. La normalización suele disminuir la cantidad de espacio de almacenamiento que una base de datos necesitará, pero puede afectar el rendimiento de las consultas.
Finalizar y validar el modelo de datos.	La creación de modelos de datos es un proceso cíclico que necesita repetirse y mejorarse conforme evolucionan los requerimientos del negocio.

Tipos de modelado de datos.

El modelado de datos ha transformado en coordinación con los sistemas de gestión de bases de datos y los tipos de modelos aumentan en complejidad a medida que las necesidades de almacenamiento de datos de cada empresa vayan evolucionando, en los siguientes párrafos se muestran varios tipos de modelos:

Modelos de datos jerárquicos:	Representan relaciones de uno a muchos en un formato arbóreo. En este tipo de modelo, cada registro tiene una única raíz.

Modelos de datos relacionales:	Este modelo los segmentos de datos se unen explícitamente mediante el uso de tablas.

Las bases de datos relacionales suelen utilizar el lenguaje SQL para manejar la información. Estas bases de datos son efectivas para garantizar la integridad de los datos y reducir la duplicación. Se emplean frecuentemente en sistemas de venta y en otros tipos de procesamiento de transacciones. He de recordar que el modelado de datos jerárquico y el relacional son dos formas clásicas de organizar la información en una base de datos. El primero la estructura como un árbol padre-hijo, mientras que el segundo la organiza en tablas que se relacionan mediante claves de acuerdo con (Pressman, 2020)

Los modelos entidad-relación (ER) usan diagramas formales para mostrar las conexiones entre diferentes entidades dentro de una base de datos. Los diseñadores de datos aplican diversas herramientas de modelado ER para elaborar representaciones visuales que comunican los objetivos de diseño de la base de datos.

Beneficios del modelado de datos.

- Reducir errores en el desarrollo de software y bases de datos.
- Aumentar la consistencia en la documentación y el diseño del sistema en toda la empresa.
- Mejorar el rendimiento de la aplicación y la base de datos.
- Facilitar el mapeo de datos en toda la organización.
- Mejorar la comunicación entre desarrolladores y equipos de inteligencia empresarial.
- Facilitar y acelerar el proceso de diseño de bases de datos a nivel conceptual, lógico y físico.

Elaboración Propia

Modelo Entidad Relación.

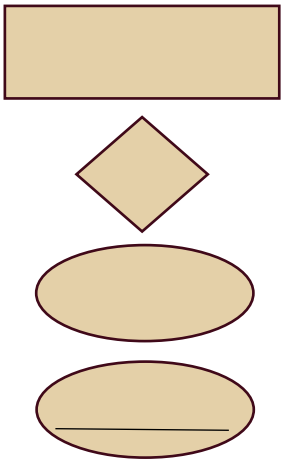
Un diagrama de relación entre entidades (diagrama ER o ERD) es una representación visual de cómo los elementos de una base de datos se relacionan entre sí. Es una técnica de modelado de datos que de acuerdo con Silberschatz, et al. (2020), permite representar de manera gráfica la estructura de la información de un sistema, mostrando:

Entidades es algo específico, como un individuo, una función, un suceso, una idea o un objeto, que puede tener datos sobre sí mismo guardados en una base de datos relacional.

Atributos: describen las propiedades que posee cada entidad. En el ejemplo anterior, atributos de la entidad "Cliente" podrían ser "Nombre", "Primer Apellido", "Fecha de nacimiento", etc.

Relaciones: sirven para crear vínculos entre parejas de entidades. Siguiendo con el ejemplo, cada "Cliente" está vinculado a una "Dirección", a uno o varios "Productos", etc.

El diagrama entidad-relación corresponde a la expresión gráfica del modelo entidad relación. Para ello, se utilizan los siguientes símbolos:

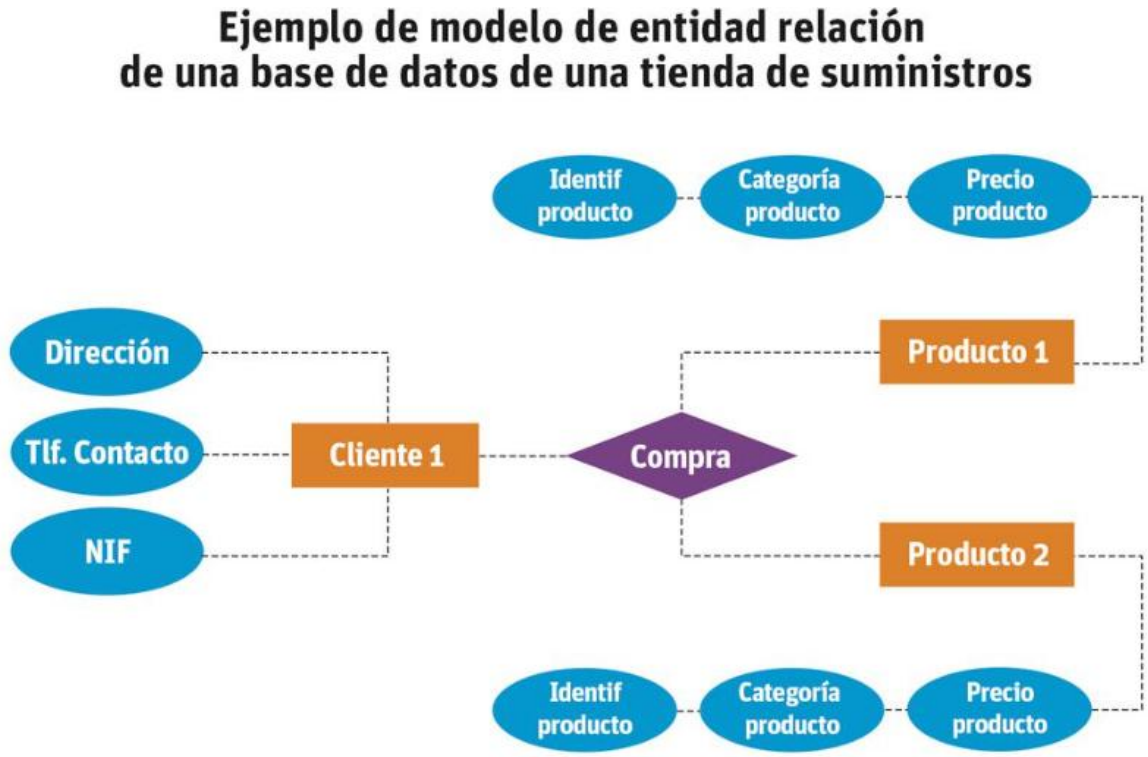
Símbolo	**Significado**
	Entidad
	Vínculo o relación
	Atributo
	Atributo clave

Las líneas de conexión suelen tener una apariencia gráfica diferente en función del tipo de relación que exista entre las entidades que conectan (puede ser una línea continua o discontinua, por ejemplo).

El modelo entidad relación pretende ser un reflejo de la estructura gramatical y, por ello, utilizan:

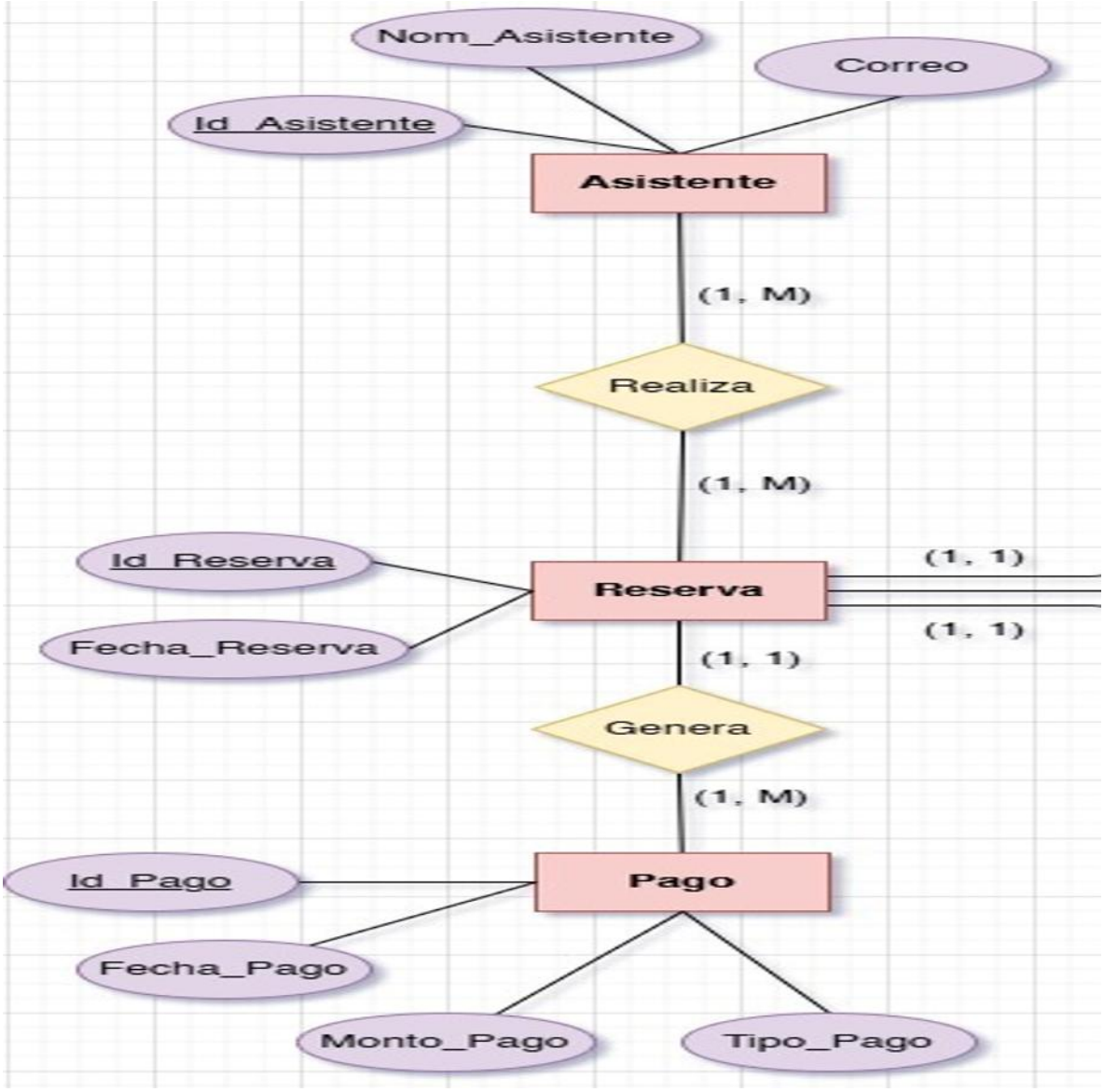
- Sustantivos, comunes o propios, para definir tipos de entidades y entidades.
- Verbos, para definir tipos de relación.
- Adjetivos, como atributos de una entidad.
- Adverbios, como atributos de una relación.

A continuación, verás un ejemplo de un diagrama entidad relación donde podrás observar algunas entidades que son relevantes para la construcción de una base de datos basados en el diseño de dicho modelo



Elaboración propia

Para efectos de nuestra asignatura y partiendo del análisis de las necesidades construiremos parte de nuestro diagrama Entidad-Relación:



Elaboración propia

Diseño de interfaces de usuario.

La interfaz que utilizan los usuarios es el enlace entre las personas y los equipos digitales. Todo dispositivo tecnológico que uses en tu rol como usuario forma parte de esta interfaz. Por ejemplo, los monitores, los efectos de sonido, el diseño estético y la interacción son aspectos de la interfaz de usuario. Una interfaz de usuario consiste en los siguientes cuatro elementos:

1. **Elementos de navegación.** Los elementos de navegación ayudan a los usuarios a navegar por una interfaz. Algunos ejemplos de elementos de navegación en la interfaz de usuario son barras deslizantes, campos de búsqueda y flechas de retroceso.

2. **Controles de entrada.** Los elementos en la página que permiten a los usuarios ingresar información son controles de entrada. Botones, casillas de verificación y campos de texto son ejemplos de controles de entrada.

3. **Componentes informativos.** Los componentes informativos se utilizan para comunicar información al usuario. Una barra de progreso debajo de un video o tutorial es un ejemplo de un componente informativo.

4. **Contenedores.** Los contenedores organizan el contenido en secciones fáciles de digerir. En lugar de listar cada subtítulo debajo de una pestaña, se puede utilizar un elemento contenedor como un menú de acordeón para ocultar o mostrar contenido

Principios clave del diseño de la interfaz de usuario.

Una manera fácil de retener los conceptos clave del diseño de interfaces es recordar estos cuatro puntos importantes:

Control.	Es esencial que los usuarios manejen la interfaz a su antojo.
Uniformidad.	Emplea componentes familiares para que tu interfaz sea intuitiva y simple de explorar, incluso para quienes son nuevos.
Flexibilidad.	La conexión con un producto debe ser una experiencia agradable y sin complicaciones.
Esfuerzo mental.	Es crucial no saturar a los usuarios con demasiada información. Mantén la comunicación lo más clara y sucinta que pueda
Usabilidad:	Haz que el sistema sea fácil de usar. Piensa en cómo hacer que las habitaciones de tu casa sean cómodas y accesibles.

Función del diseño de interfaz de usuario.

Un diseño de interfaz cuidadosamente elaborado puede impactar notablemente la vivencia del usuario. Una interfaz que sea clara y visualmente atractiva puede llevar a los usuarios a sentirse más a gusto y conectados con una aplicación o página web. Analizaremos el proceso de desarrollo:

1.Investigación y Análisis de Usuarios.

Antes de iniciar el proceso de diseño, es fundamental entender a los usuarios y sus requerimientos. Realizar investigación y análisis de usuarios facilita la identificación de metas y dificultades que deben ser resueltas por el diseño.

2.Prototipado y Wireframing.

Para visualizar la organización y la interacción de la interfaz previa a su ejecución, los diseñadores generalmente producen prototipos y wireframes. Estos prototipos son versiones simplificadas que permiten ensayar ideas y obtener opiniones.

3.Testing y Mejora Iterativa.

Tras la puesta en marcha de la interfaz, se llevan a cabo pruebas con usuarios reales para detectar inconvenientes y oportunidades de mejora. El diseño de la interfaz es un ciclo continuo donde se ajustan y perfeccionan constantemente los componentes y la interacción con el fin de ofrecer la mejor experiencia posible.

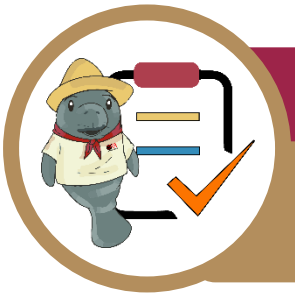
Ejemplos de interfaz de usuario:

Figura 6: Pantalla interfaz de usuario de texto. Tomado de Redd.It(2026, may 28)

Figura 7. Pantalla interfaz de usuario gráfica. Tomado de controlescolarmexico.com. (s/f).

Actividad 2. Modelando mis datos (DER)

Instrucciones: En equipo, diseñen un modelo de datos para el SI que se está desarrollando como solución a la situación didáctica. Representen mediante un DER. Asegúrense de que refleje los procesos y necesidades clave de la situación didáctica que se busca resolver.



LISTA DE COTEJO

Actividad 2.1 Modelando mis datos.

DATOS GENERALES		
Nombre(s) del alumno (s)		**Matriculas(s)**
Producto: Caso práctico		**Fecha**
Submódulo: Sistemas de información		**Periodo:2025-2026A**
Nombre del docente:		**Firma del docente**

CRITERIOS	INDICADORES	VALOR OBTENIDO		PONDERACIÓN	CAL	OBSERVACIONES Y/O SUGERENCIAS
		SI	NO			
1	Utiliza adecuadamente los símbolos para la elaboración de diagramas de flujo.			2		
2	Participa y colabora de forma colaborativa, mostrando interés en su realización.			2		
3	El diseño diagrama de flujo, no presenta errores ortográficos y de estética en su elaboración.			2		
4	Entrega en tiempo y forma la elaboración de los Modelos de datos.			2		
5	Identifica claramente un atributo, entidad y relaciones.			2		
				CALIFICACIÓN		
Logros				Aspectos de Mejora		

Práctica 1. Diseña tu propia interfaz de usuario (Opción A)

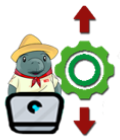
Opción A **Instrucciones:** *Con tu equipo de trabajo, crea una interfaz sencilla y funcional, para un sistema de información.*

Diseño de pantallas en Canva
https://www.canva.com/

En equipos, con la ayuda de la aplicación canva, figma, PowerPoint o. crea una pantalla de inicio para un sistema de información. Al finalizar la actividad comparte los enlaces de tus creaciones con tus compañeros para socializar los diseños de las pantallas. Usa los siguientes videos de apoyo:

Recurso didáctico sugerido

Crear un diseño de interfaz de usuario con canva
https://www.youtube.com/watch?v=_j9DzsQPrjY



Práctica 1. Diseña tu propia interfaz de usuario (Opción B)

Opción B

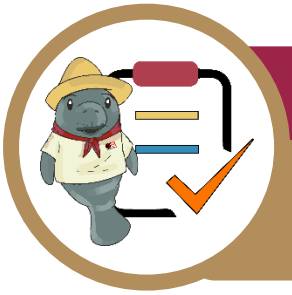
Instrucciones: *Con tu equipo de trabajo, crea una interfaz sencilla y funcional, para un sistema de información.*

Materiales necesarios:

- Hoja de papel cuadriculado o software de diseño simple (puede ser PowerPoint, Canva o Paint).
- Lápices, marcadores, o herramientas digitales según tu preferencia.

Diseña las pantallas principales:

- Dibuja cómo se verán las principales pantallas del sistema.
- Incluye botones, menús, y campos de texto.
- Asegúrate de que las opciones sean fáciles de encontrar.
- Usa colores que hagan que tu diseño sea claro y fácil de usar.
- Asegúrate de que el texto sea legible y las opciones sean lo suficientemente grandes para seleccionarlas.



LISTA DE COTEJO

Actividad 2.2: Diseña tu propia interfaz de usuario.

DATOS GENERALES		
Nombre(s) del alumno (s)		**Matriculas(s)**
Producto: Caso práctico		**Fecha**
Submodulo: Sistemas de información		**Periodo:2025-2026A**
Nombre del docente:		**Firma del docente**

CRITERIOS	INDICADORES	VALOR OBTENIDO SI	VALOR OBTENIDO NO	PONDERACIÓN	CAL	OBSERVACIONES Y/O SUGERENCIAS
1	Utiliza elementos de navegación			2		
2	Las pantallas son fáciles de entender			2		
3	Uso de colores, íconos, y diseño atractivo			2		
4	Usa controles que permiten el acceso a la información			2		
5	Posee uniformidad			1		
6	Tiene buena ortografía			1		
				CALIFICACIÓN		

Logros		Aspectos de Mejora	

Referencias

¿Qué es el modelado de datos? (2023, 3 de julio). *IBM.com*. https://www.ibm.com/mx-es/topics/data-modeling

Cooper, A., Reimann, R., & Cronin, D. (2014). *About Face: The Essentials of Interaction Design*. Wiley.

Norman, D. A. (2013). *The Design of Everyday Things*. Basic Books.

Pastor Collado, J. A., & Elias Vila, E. (2024)) *Diseño y Arquitectura de Sistemas de Información*

Pressman, R. S., & Maxim, B. R. (2020). Software Engineering: A Practitioner's Approach (9th ed.). McGraw-Hill.

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). Database system concepts (7th ed.). McGraw-Hill.

Lectura 3. Modelo relacional de bases de datos

¿Has pensado cómo Netflix sabe qué serie estás viendo?

Piensa cómo la plataforma sabe qué serie estás viendo, qué capítulo sigue y que usuario lo está reproduciendo y en qué minutos pausaste. No es magia, todo está guardado en tablas.

El modelo relacional de base de datos partió de la idea que inventó un matemático Edgar F. Codd en 1970, donde la información se almacena en tablas formadas por filas y columnas, permitiendo que los datos se puedan consultar, combinar y actualizar fácilmente. El conjunto de tablas interrelacionadas da origen a una base de datos, la cual es la parte central de un sistema de gestión de información, y estos a través de un Sistema Gestor de Base de Datos (SGBD) nos permite realizar consultas, informes y formularios.

3.1 Conceptos clave: tablas, registros, campos

En la actividad anterior trabajaste con el diagrama entidad-relación; esas entidades forman las tablas y cada atributo corresponde a los campos de la tabla, de acuerdo con la figura 5.1

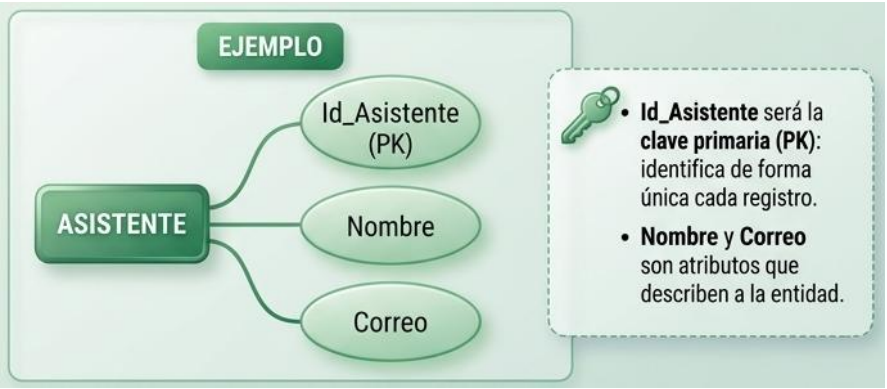


Ilustración 5. 1 Transición de una entidad a una tabla relacional. Elaborado mediante el modelo de Inteligencia Artificial Google Nano Banana (2026).

Tabla

La **tabla** es una estructura organizada que almacena información sobre un tema o una entidad específica (como alumnos, películas, usuario).

Elementos de la tabla:

- **Nombre:** Debe ser único y descriptivo (ej. PRODUCTOS, CALIFICACIONES).
- **Campos / Atributos:** Definen **qué tipo** de datos se van a guardar (ej. Nombre, Precio, Fecha de nacimiento).
- **Registros / Tuplas:** Son los **datos reales** de cada objeto o persona individual. Es el conjunto de datos que pertenece al mismo objeto, persona o evento dentro de una tabla. Es una fila completa de la tabla

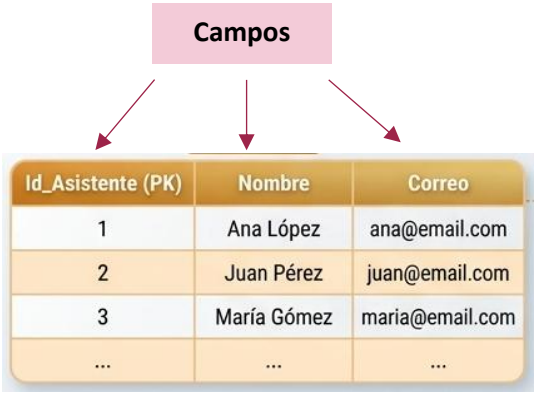


Ilustración 5. 2 Campos

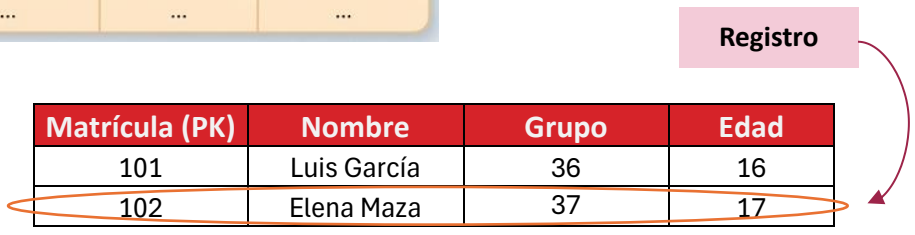


Ilustración 5. 3 Registro

Características de los registros:

1. Se lee de izquierda a derecha

2. Toda la fila es un bloque de información

3. No puede haber dos registros exactamente iguales en la misma tabla.

Campos (Atributos)

Los campos son importantes en la estructuración de la información, porque determina el tipo de dato a almacenar, él análisis de la información y las relaciones con otras tablas. A continuación, la descripción de las características:

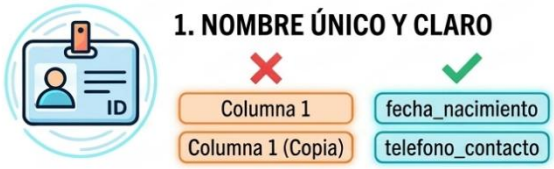
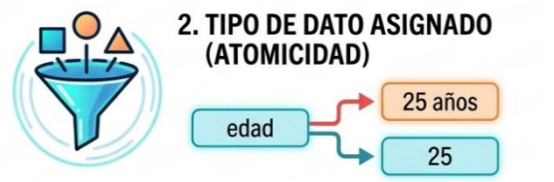
Característica de Campo	Descripción
	Las columnas deben tener un nombre descriptivo corto con el fin de que cualquier persona pueda entender el concepto del sistema. Dos columnas no pueden llamarse igual en la misma tabla
	Numéricos: INT, DECIMAL, REAL Cadena (Texto): CHAR(n), VARCHAR(n), CLOB Tipos de Fecha: DATE, TIME, TIMESTAMP Lógicos: BOOLEAN
	Evita lista o datos combinados porque violan la **Primera Forma Normal (1FN)** del diseño de bases de datos. Por ello es que dirección se divide en: calle, colonia y ciudad.
	Son reglas del negocio para asegurar la **integridad, calidad y exactitud** de los datos. Restricciones: **De Identificación:** PRIMARY KEY, FOREIGN KEY **De Validación:** NOT NULL, UNIQUE, CHECK **Por Defecto:** DEFAULT

Ilustración 5. 4 Características de los campos

3.2 Claves Primarias (PK) y Claves Foráneas (FK)

Ilustración 5. 5 Clave primaria generada con IA Nano banana 2 (2026)

Clave Primaria (PK)

Según Microsoft, una **clave principal o primaria (Primary Key)** es:
Un campo o conjunto de campos (ID Alumno + ID Materia) que identifica de manera **única** cada registro en una tabla y que **no permite valores nulos**.

a) *No se repite (es única).*

ID (PK)	Nombre	Edad
1	Ana	20
1	Luis	22

ID (PK)	Nombre	Edad
1	Ana	20
2	Luis	22

b) *No puede estar vacía (NOT NULL). Siempre debe tener un valor*

ID (PK)	Nombre	Edad
	Ana	20
2	Luis	22

ID (PK)	Nombre	Edad
1	Ana	20
2	Luis	22

Clave foránea (FK)

Es un campo (o conjunto de campos) en una tabla que hace referencia a la clave primaria de otra tabla. Es un vínculo que conecta dos tablas, indicando que un valor en una tabla depende de otro valor existente en otra tabla, permitiendo relacionar datos de forma segura y organizada.

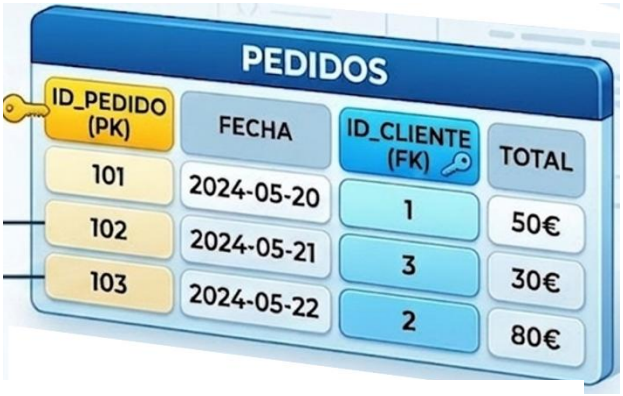


Ilustración 5. 6 Ilustración 5. 7 Clave primaria y foránea generada con IA Nano banana 2 (2026)

Ventajas:

- Evitar errores
- Reducir redundancia y
- Mantener la integridad de la base de datos.
- Facilitar consultas
- Permite hacer consultas combinadas (*JOINs, en SQL*),

Ejemplo:

Imagina que tienes una tienda con entrega a domicilio y necesitas organizar la información de tus **Clientes** y los **Pedido.** Entonces tenemos una **Tabla Clientes** en el cual se almacenará los datos de los compradores y una **Tabla Pedidos,** esta tabla almacena los datos de los compradores. **ID_CLIENTE** en **CLIENTES** es llave primaria o llave principal mientras que **ID_CLIENTE** en **Pedidos** es una **clave foránea.**

Ilustración 5. 7 clave primaria y foránea

En este caso se evita errores como:

- Registrar un pedido de un cliente que no existe.

- Eliminar un cliente que aún tiene pedidos asociados.

3.3 Relaciones

El modelo relacional desarrolla un esquema de base de datos (data base schema) a partir del cual se podrá realizar el modelo físico o de implementación en el SMBD

En esta parte las tablas las sustituiremos por las cajitas, es decir tablas relacionales.

Tabla: Estudiante

ID_Estudiante (PK)	Nombre_Estudiante	Dirección	Teléfono
12	Alex García	C23	9875641236

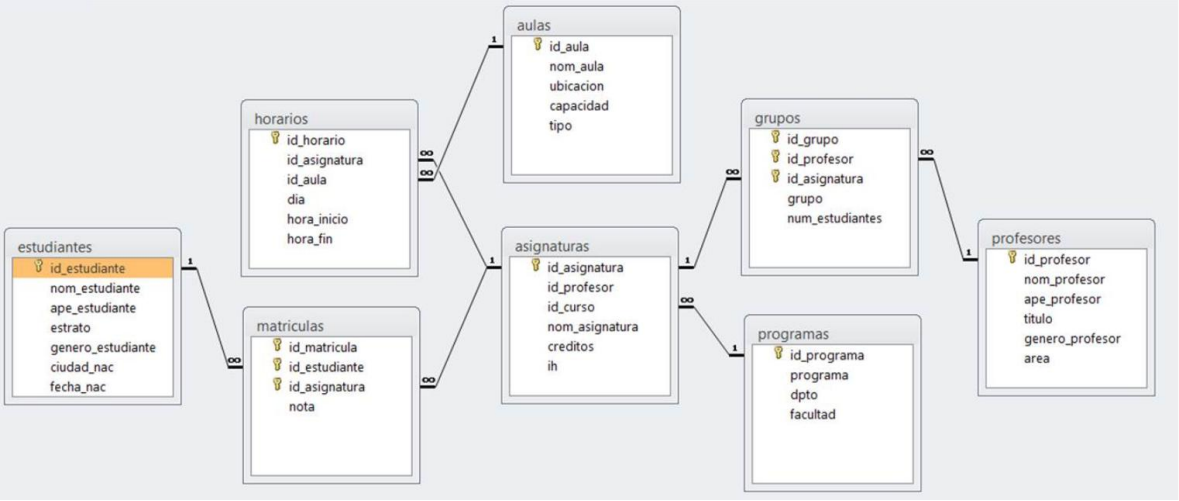


Ilustración 5. 8 Ejemplo de Relación de una base de datos de horario de clase

Un modelo relacional se representa a través de un **Diagrama Relacional**, donde cada tabla se muestra como un rectángulo con su nombre arriba y la lista de columnas dentro. Las **relaciones** entre tablas se representan mediante líneas que conectan la Clave Primaria de una tabla con la Clave Foránea de otras

Ventajas:

1. Evita la Redundancia de Datos.
    - Registras al cliente una sola vez en la tabla Clientes. En la tabla Ventas solo guardas su número de ID, ahorrando espacio de almacenamiento
2. Previene Errores de Actualización (Anomalías)
    - Modificas el teléfono una sola vez en la tabla Proveedores y el cambio se refleja instantáneamente en todo el sistema.
3. Garantiza la Integridad Referencial
    - El sistema te obliga a elegir un país que ya exista en la tabla Países, impidiendo la introducción de datos basura o inválidos.

Cardinalidad de las Relaciones

Retomemos el caso de Netflix. Tenemos las reglas del negocio (Restricciones):

1. Un (1) usuario (el titular de la cuenta que paga) puede tener **muchos (N)** perfiles dentro de Netflix (el perfil del papá, de la mamá, del hijo). Pero cada perfil individual pertenece estrictamente a **un** solo usuario.
2. **Un (1)** perfil de usuario puede reproducir contenido **muchas (N)** veces, generando múltiples filas en el historial. Sin embargo, cada registro individual de reproducción en el historial le pertenece a un único perfil.
3. Una (1) película o serie específica puede ser vista **muchas (N)** veces por diferentes personas, quedando registrada repetidamente en el historial de la plataforma.
    - Nota: Los motores de bases de datos no pueden procesar relaciones Muchos a Muchos de forma directa, para solucionar esa complejidad se recurre a crear dos relaciones sencillas de **1 a Muchos**.

Como habrás notado estas reglas del negocio dan lugar a la cardinalidad de las relaciones.

Los conceptos teóricos estandarizados en la ingeniería de software y el modelado de datos definen la **cardinalidades** de acuerdo con cuántos registros de una tabla se pueden asociar con otros registros de otra tabla. Existen tres tipos principales de relaciones:

1. Relación Uno a Uno (1:1)
Un registro de la Tabla A se relaciona con **un solo** registro de la Tabla B, y viceversa.
- **Ejemplo:** CLIENTE y DATOS_FISCALES. Un cliente solo tiene un RFC/número fiscal, y ese número fiscal pertenece a un solo cliente.

2. Relación Uno a Muchos (1:N)

Un registro de la Tabla A se puede relacionar con **muchos** registros de la Tabla B, pero un registro de la Tabla B se relaciona con **un solo** registro de la Tabla A. Es la más común.

- **Ejemplo:** CLIENTE y PEDIDOS. Un cliente puede realizar muchos pedidos, pero cada pedido pertenece a un único cliente.

3. Relación Muchos a Muchos (N:M)

Muchos registros de la Tabla A se pueden relacionar con **muchos** registros de la Tabla B. En las bases de datos relacionales, esta relación se debe transformar creando una tabla intermedia.

- **Ejemplo:** PEDIDOS y PRODUCTOS. Un pedido puede contener muchos productos, y un producto puede estar presente en muchos pedidos diferentes.

La relación de las tablas de Netflix quedaría de la siguiente manera:

Ilustración 5. 9 Cardinalidad en Tablas de Netflix

Donde:

Tabla: USUARIOS	Guarda los datos de las personas que pagan la suscripción.
Tabla: PERFILES	Una cuenta tiene varios perfiles (el tuyo, el de tu mamá, tu hermano, etc.).
Tabla: CONTENIDO	Guarda las películas o series disponibles.
Tabla: HISTORIAL	Esta tabla une a los **Perfiles** con el **Contenido** para registrar qué vio cada quién y en qué minuto se quedó.
Plan:	Premium, Estándar o Estándar con anuncios.
Genero:	Ciencia Ficción, Drama, Comedia.

Normalización de Base de Datos

Imagínate esto: lunes por la mañana, vas tardísimo a la escuela, abres el clóset y lo único que ves es un cerro gigante de ropa amontonada. Hay pantalones, camisas, playeras hechas bola y los calcetines flotando por todos lados.

Para colmo, necesitas a fuerzas ese calcetín azul marino que combina con el uniforme. Te metes a buscar en esa montaña de tela, avientas todo al piso y te empiezas a estresar porque el tiempo corre y el bendito calcetín no aparece por ningún lado. Un caos por culpa del desorden.

Bueno, pues a una computadora le pasa exactamente lo mismo cuando guardamos los datos todos amontonados en una sola tabla de Excel o de una base de datos. El sistema se vuelve lentísimo buscando un solo dato entre millones, se traba y, si borras o modificas algo mal, te terminas trayendo entre las patas a toda la demás información. Por eso existe la normalización: para pasar del cerro de ropa a un clóset bien organizado donde encuentras todo a la primera.

Concepto de Normalización:

Es un proceso que consiste en organizar los datos de una base de datos para evitar que la información se duplique innecesariamente y para asegurar que todo esté guardado en el lugar correcto. Con la normalización, reducimos la repetición de datos, ahorramos espacio en el disco duro del servidor, el mantenimiento de la base es más simple, se protegen mejor los datos, las consultas con SQL tienen mejor rendimiento, se obtiene flexibilidad y escalabilidad en el crecimiento a futuro de la base de datos.

Existen 6 formas normales para aplicar normalización a una base de datos, recordemos que una normalización son reglas que el programador asigna para poder organizar la base de datos, estas nos van a permitir identificar y eliminar errores en nuestra base.

De estas seis realmente se aplican las primeras tres, con ellas aseguramos una óptima base de datos y las consiguientes se trabajan cuando hay algún requerimiento especial

- Primera forma normal (1FN)
- Segunda forma normal (2FN)
- Tercera forma normal (3FN)
- Forma normal Boyce Codd (FNBC)
- Cuarta forma normal (4FN)
- Quinta forma normal (5FN)

Analizaremos las 3 primeras formas según Microsoft:

1ª Forma Normal (1NF): El fin del amontonamiento

- **La regla:** Eliminar los grupos repetidos y asegurar que cada celda tenga un solo valor indivisible (atómico).
- No se permiten listas dentro de una sola casilla. Si un cliente tiene tres números de teléfono, no los escribas separados por comas en la misma celda ni crees columnas llamadas Teléfono1, Teléfono2 y Teléfono3

2ª Forma Normal (2NF): Todo con su dueño [1]

- **La regla:** Cumplir la 1NF y hacer que todos los datos de la tabla dependan completamente de la llave primaria.
- Se enfoca en separar los temas. Si tienes una tabla de compras donde la llave es el ID_Pedido, no metas ahí el nombre del producto ni el precio del proveedor. Esos datos no dependen del pedido, pertenecen al producto en sí.

3ª Forma Normal (3NF): Eliminar los intermediarios [1]

- **La regla:** Cumplir la 2NF y eliminar las dependencias transitivas (datos que dependen de otros datos que no son la llave primaria)
- **Explicación humana:** "Los amigos de mis amigos no tienen que vivir en mi casa". Si en tu tabla de CLIENTES guardas el Código Postal y el nombre de la Ciudad, la ciudad realmente depende del código postal, no directamente del ID del cliente. Si el cliente se muda, tendrías que cambiar la ciudad de forma redundante.

El siguiente paso con el caso de Netflix, es normalizar nuestras tablas. La información que tenemos en el campo **Planes** en **USUARIOS** y los **Géneros** en **CONTENIDO** dcausan redundancia de datos (repetición innecesaria).

Diseño normalizado (en Tercera Forma Normal),

1. Modificación en USUARIOS y Nueva Tabla PLANES (Evita errores de dedo)

Sí en planes por error se escribiese "Premium", "premium" o "Premuim", la base de datos se corrompe. Lo que tenemos que hacer es separar **Planes** a su propia tabla.

- **Tabla: PLANES (Nueva)**
  - **ID_Plan [PK]**: Código del plan (ej. PLN-1, PLN-2).
  - **Nombre_Plan**: Texto (Estándar con anuncios, Estándar, Premium).

- **Precio**: Costo mensual (ej. 139.00, 269.00, 369.00).

- **Tabla: USUARIOS (Modificada)**
  - **ID_Usuario [PK]**: Código único del cliente.
  - **Nombre**: Nombre del titular.
  - **Correo**: Dirección de email.
  - **ID_Plan [FK]**: Conecta con la tabla PLANES. *(Eliminamos el campo de texto "Plan")*.

2. Tabla PERFILES: Esta tabla ya cumple perfectamente con la 3FN. No requiere modificaciones.

- **Tabla: PERFILES**
  - **ID_Perfil [PK]**: Código único del perfil.
  - **Nombre_Perfil**: Nombre en pantalla.
  - **Tipo**: Infantil o Adulto.
  - **ID_Usuario [FK]**: Conecta con el usuario dueño de la cuenta.
  -

3. CONTENIDO y Nueva Tabla GENEROS (Relación de catálogo)

Una película o serie puede tener varios géneros, o bien, el mismo género ("Ciencia Ficción") se va a repetir miles de veces escritos como texto. Lo ideal es independizarlo.

- **Tabla: GENEROS (Nueva)**
  - **ID_Genero [PK]**: Código del género (ej. GEN-01, GEN-02).
  - **Nombre_Genero**: Texto (Drama, Comedia, Ciencia Ficción).

- **Tabla: CONTENIDO (Modificada)**
  - **ID_Contenido [PK]**: Código único del video.
  - **Título**: Nombre de la película o serie.
  - **Año**: Año de estreno.

- **ID_Genero [FK]**: Conecta con la tabla GENEROS. *(Eliminamos el campo de texto "Género")*

4. Tabla HISTORIAL: Es una tabla de hechos, sus campos son eficientes y cumplen las 3NF.

Tabla: HISTORIAL

- **ID_Historial [PK]**: Registro único de reproducción.
- **ID_Perfil [FK]**: ¿Quién lo vio?
- **ID_Contenido [FK]**: ¿Qué vio?
- **Minuto_Exacto**: Minuto de la pausa.

En la normalización pasamos **4 tablas a 6 tablas**. Al eliminar las depencias transitivas llegamos a la **Tercera Forma Normal (3FN)**. Si el precio del "Plan Premium" cambia, solo lo editas en **una sola fila** de la tabla PLANES, en lugar de modificar millones de registros de usuarios uno por uno.

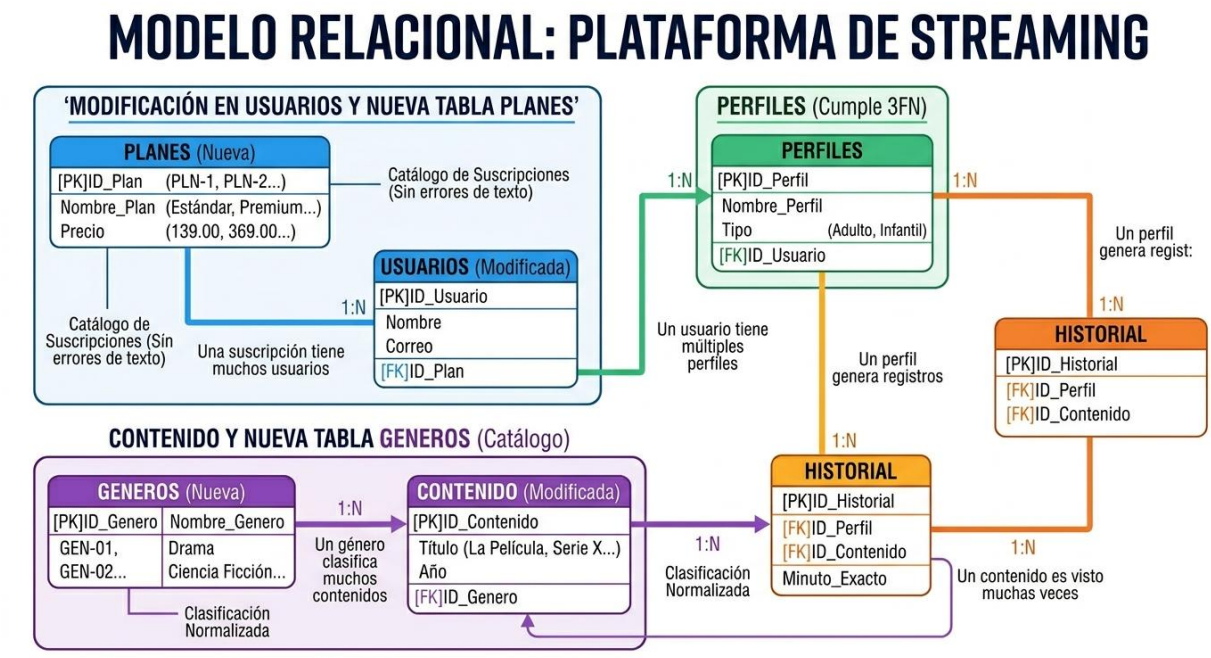


Ilustración 5. 10 Modelo Relacional de Base de Datos. Generado por Nano Banana (2026)

Actividad 3.1. Ejercicio teórico: Modelo relacional: Tablas, claves y relaciones

Instrucciones: *Lee la información proporcionada sobre el modelo de base de datos relacional, como los conceptos fundamentales, las características principales, los tipos de relaciones y los roles de las claves en una base de datos.*

Sección 1: Conceptos Básicos

1. ¿Qué es una base de datos relacional?

2. Dibuje lo que a continuación se solicita:
   a. Tabla

   b. Registro

   c. Campo

3. Explica los tres tipos de relaciones (1:1, 1: N, N:M) y proporcione un ejemplo de cada uno.

4. ¿Qué es una clave primaria?

5. ¿Cuál es la diferencia entre clave primaria y clave foránea?

6. ¿Qué es una restricción en el modelo relacional?

Sección 2: Análisis Práctico

Observa las siguientes tablas de una base de datos de una tienda y responde las preguntas:

Tabla de Productos

Producto_ID	Nombre	Categoría	Precio
1	Laptop	Electrónica	15000
2	Teléfono	Electrónica	8000

Tabla de Clientes

Cliente_ID	Nombre	Correo electrónico
101	Juan Pérez	juan@gmail.com
102	María López	maria@hotmail.com

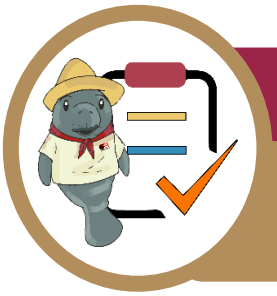
- Determine las llaves primarias y las llaves foráneas de las tablas anteriores.

- Si aplicaras normalización, ¿qué tabla se modificaría, y cuál campo generaría otra tabla?

- ¿Puede un cliente comprar múltiples productos? Justifique su respuesta.

Sección 3: Reflexión Final

- Explica la importancia del modelo relacional para el diseño y gestión de bases de datos.



INSTRUMENTO DE EVALUACIÓN - LISTA DE COTEJO

Actividad 3.1. Ejercicio Teórico:

"Modelo relacional: Tablas, Claves y Relaciones"

DATOS GENERALES		
Nombre(s) del alumno (s)		**Matriculas(s)**
Producto: Caso práctico		**Fecha**
Submodulo: Sistemas de información		**Periodo:2025-2026A**
Nombre del docente:		**Firma del docente**

	INDICADORES	VALOR OBTENIDO SI	VALOR OBTENIDO NO	PONDERACIÓN	CAL	OBSERVACIONES Y/O SUGERENCIAS
1	Reconoce adecuadamente los conceptos claves.			2		
2	Distingue correctamente el tipo de relaciones.			2		
3	Diferencia entre claves foráneas y primarias.			1		
4	Comprende el uso de restricciones en las bases de datos.			1		
5	Analiza las tablas y relaciones en el ejercicio			1		
6	Identifica adecuadamente las relaciones entre tablas.			1		
7	Entrega en tiempo y forma			2		
				CALIFICACIÓN		

Logros		**Aspectos de Mejora**	

Actividad 3.2. Ejercicio Práctico: Construye un modelo de BD para la Reservación de Eventos

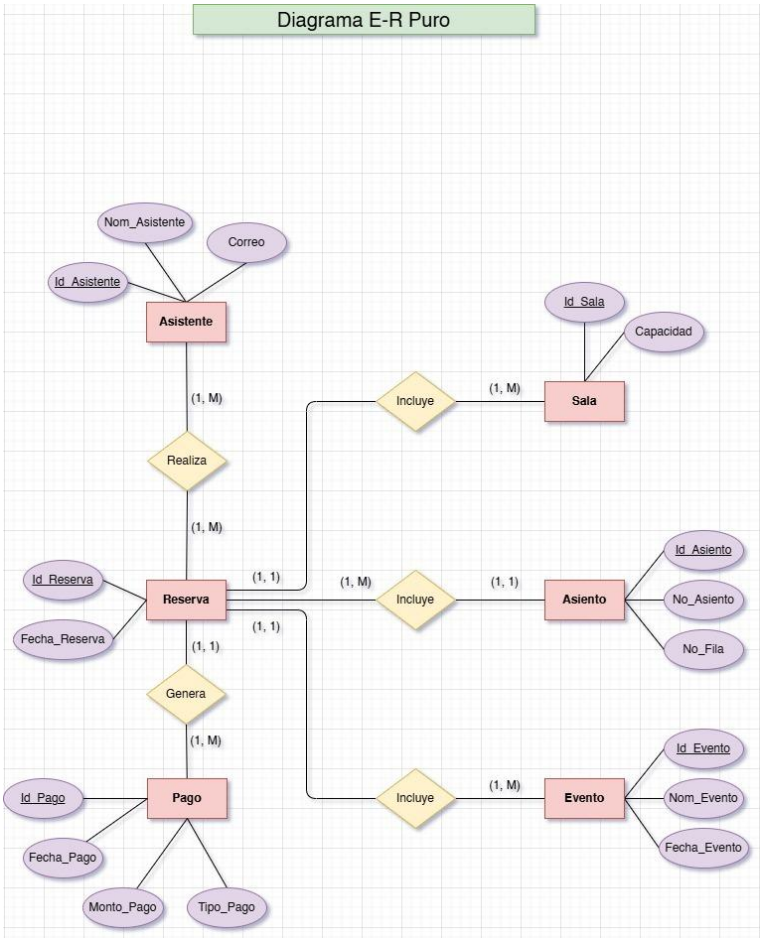
Instrucciones: A partir del diagrama entidad-relación de la reservación de eventos: Genera las tablas relacionales y la normalización de las tablas.

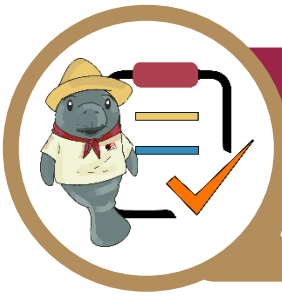
Método de Desarrollo de Software: Metodología de cascada

Objetivo: Diseñar un Sistema de Información para Reservas de Eventos, el cual permitirá a los asistentes:

- Registrar sus datos personales.
- Elegir asientos específicos.
- Seleccionar diferentes horarios y salas.
- Seleccionar medio de pago.

1. Dibuja o crea las tablas clave del sistema y realice 3 registros en cada tabla. Entidad; Atributos; Clave primaria
2. Realice el diagrama relacional y especifica cómo las entidades están relacionadas entre sí. Relaciones 1:1, 1:N o N:M. Indica las claves foráneas que son necesarias.
3. Analice cada tabla y justifique en qué tipo de normalización se encuentra cada tabla.
4. Preguntas de reflexión:
   ¿Por qué es bueno determinar claves primarias y foráneas?





INSTRUMENTO DE EVALUACIÓN - LISTA DE COTEJO

Actividad 3.2. Ejercicio práctico:

"Construye un modelo de BD para Reservación de Eventos"

DATOS GENERALES		
Nombre(s) del alumno (s)		**Matriculas(s)**
Producto: Caso práctico		**Fecha**
Submodulo: Sistemas de información		**Periodo:2025-2026A**
Nombre del docente:		**Firma del docente**

No.	INDICADORES	VALOR OBTENIDO SI	VALOR OBTENIDO NO	PONDERACIÓN	CAL	OBSERVACIONES Y/O SUGERENCIAS
1	Identificó correctamente las entidades			1		
2	Asignó al menos 3 registros por cada entidad			1		
3	Identificó claramente las claves primarias por entidad			1		
4	Indicó adecuadamente las relaciones entre entidades (1: N, N:M)			2		
5	Especificó correctamente las claves foráneas			1		
6	Construyó una tabla resumen clara (Entidad, Atributos, Clave primaria)			2		
7	La información del modelo ER está bien estructurada, clara y sin errores conceptuales			1		
8	Respondió de manera reflexiva la pregunta final de análisis			1		
				CALIFICACIÓN		

Logros		Aspectos de Mejora	

Referencias

Escuela británica de artes creativas y tecnológicas. (13 de 04 de 2025). Obtenido de ebac: https://ebac.mx/blog/normalizacion-de-bases-de-datos

Capacho, J. R., & Nieto Bernal, W. (2017). Diseño de bases de datos. Universidad del Norte.

Costa, D. C. (2002). El modelo relacional y el álgebra relacional. UOC, la universidad virtual.

Quiroz, J. (2003). El modelo relacional de bases de datos. Boletín de Política Informática, 6, 53–61.

Rocío, P. L. (s.f.). Apuntes de base de datos relacionales.

Sánchez, J. (2004). Principios sobre bases de datos relacionales. Informe, Creative Commons, 11, 20.

Silberschatz, A., Korth, H. F., Sudarshan, S., Pérez, F. S., Santiago, A. I., & Sánchez, A. V. (2002). Fundamentos de bases de datos (Vol. 11). McGraw-Hill.

Tecnología, M. (13 de 04 de 2025). Obtenido de Youtube: https://www.youtube.com/watch?v=m7kpSO6kqY8

Lectura 4. SQL sin HACKS

Instrucciones: *Realiza el análisis de la siguiente información, posteriormente, completa la Actividad 4. Gestión de Videojuegos.*

Diccionario de Datos

Jugar con hacks (modificaciones no autorizadas o exploits) en un videojuego te puede dar una victoria rápida, pero en el mundo real, meterle 'hacks' a una base de datos de SQL es la forma más fácil de que te despidan por tirar el sistema. En esta lectura veras cómo trabajar con orden y sin trucos raros, usando nuestra mejor herramienta: el Diccionario de Datos.

El diccionario de datos es el mapa y el manual de instrucciones de una base de datos. Es un contenedor que explica detalladamente qué significa cada tabla, qué tipo de información guarda cada columna, y qué reglas debe cumplir un dato para ser válido.

Ejemplo:

"Imaginen que Spotify los contrata para diseñar la base de datos de una nueva función: **Las Listas de Reproducción Compartidas**. El jefe les da el código de la tabla, pero se le olvidó poner la documentación. Tu trabajo es crear el **Diccionario de Datos** para que los nuevos programadores entiendan qué significa cada campo."

Código de la tabla.

```sql
CREATE TABLE playlist_compartida (
    id_play INT,
    nom_p VARCHAR(50),
    user_crea VARCHAR(30),
    f_alta DATE,
    tipo_acc INT
);
```

Intenta traducir el código a "idioma humano" usando tu lógica:

Nombre del Campo (Código)	¿Qué tipo de dato es?	Descripción en español simple (¿Para qué sirve?)	Ejemplo de un dato real
`id_play`	Número entero		
`nom_p`	Texto		
`user_crea`	Texto		
`f_alta`	Fecha		
`tipo_acc`	Número entero		

¿Lo lograste?

Compara tus respuestas

- **`id_play`** : Número único para identificar la playlist. (Ejemplo: `1054`).
- **`nom_p`** : Nombre de la playlist. (Ejemplo: `'Éxitos para estudiar'`).
- **`user_crea`** : El nombre de usuario de la persona que inventó la playlist. (Ejemplo: `'juan_perez20'`).
- **`f_alta`** : El día exacto en que se creó la playlist. (Ejemplo: `'2026-06-09'`).
- **`tipo_acc`** : El tipo de privacidad. **Regla del negocio:** `1` = Pública, `2` = Privada, `3` = Solo amigos.

El diccionario de datos definitivo **se crea y se completa después de la normalización**. Esto permite documentar al detalle cada columna, tipo de dato, claves (PK/FK) y restricciones de las tablas ya normalizadas.

Ejemplo de una estructura:

Tabla:	Nombre de la entidad o tabla		
Nombre	**Tipo de dato**	**Longitud**	**Descripción**

Tenemos el diccionario de datos de Cliente

Tabla:	Cliente		
Nombre del campo	**Tipo de dato**	**Longitud**	**Descripción**
Id_cliente	Texto	6	Código del cliente, clave principal
Nombre	Texto	30	Nombre del cliente
Correo	Texto	30	Correo electrónico del cliente

Tabla:	Sala		
Nombre del campo	**Tipo de dato**	**Longitud**	**Descripción**
Id_sala	Entero	2	Código de la sala, clave principal
Número de sala	Texto	30	Nombre de la sala (1, 2, 3)
Asiento	Texto	30	Número de asiento

Y así sucesivamente podrías ir registrando los datos de las tablas resultantes de la normalización.

Implementación en SQL

SQL (Structured Query Language) es un lenguaje de programación utilizado en la mayoría de las aplicaciones actuales, ya que este permite crear bases de datos relacionales. Este permite realizar operaciones como **insertar, actualizar, consultar y eliminar** datos que se requieren para obtener información estructurada garantizando la **integridad, seguridad y coherencia** de estos (Núñez, 2025).

Existen diversos tipos de programas para crear bases de datos relacionales como: MySQL, Microsoft SQL Server, PostgreSQL y Oracle, estos se conocen como sistemas de gestión de bases de datos (DBMS), estos permiten a los usuarios manipular eficientemente con grandes cantidades de datos (Piñeiro, 2015)

Sentencias Básicas: SELEC, INSERT, UPDATE, DELETE

SQL utiliza las siguientes cuatro sentencias básicas para realizar operaciones en bases de datos:

Sentencia	Descripción	Sintaxis	Ejemplo
SELECT	Consulta datos en una base de datos. Esta sentencia permite especificar qué columnas y filas se desean recuperar de una o varias tablas	SELECT columna1, columna2 FROM nombre_tabla WHERE condición;	**SELECT** nombre, edad **FROM** alumnos **WHERE** edad > 12;
INSERT	Inserta nuevos registros en una tabla.	INSERT INTO nombre_tabla (columna1, columna2) VALUES (valor1, valor2);	**INSERT INTO** alumnos (nombre, edad) **VALUES** ('Karla', 15);
UPDATE:	Modifica los datos existentes en una tabla	UPDATE nombre_tabla SET columna1 = valor1, columna2 = valor2 WHERE condición;	**UPDATE** alumnos **SET** edad = 15 **WHERE** nombre = 'Karla';
DELETE:	Elimina registros de una tabla	DELETE FROM nombre_tabla WHERE condición;	**DELETE FROM** alumnos **WHERE** nombre = 'Karla';

Estas sentencias forman la base de toda interacción con bases de datos relacionales, siendo universales en distintos sistemas como MySQL, PostgreSQL o SQL Server (Mosquera et al., 2018).

WHERE

Es una cláusula que se utiliza para mostrar solo aquellos conjuntos de datos de las filas de la tabla al cual se refiera, y que se especifica con una condición que debe cumplir antes de mostrar el resultado, estas condiciones utilizan operadores lógicos y de comparación

FROM

AS

- **FROM Especifica la(s) tabla(s) que se van a utilizar al momento de hacer la consulta**
- **AS Permite renombrar el campo (columna) de una tabla al crearse la consulta, sin afectar la tabla original.**

Operadores en SQL

Los operadores en SQL son símbolos o palabras claves que son utilizados en consultas para **combinar condiciones**, **comparar datos** o realizar **operaciones**

lógicas/matemáticas. Al evaluar una expresión con operadores, el resultado será un valor booleano (true o false) que determina si una fila cumple o no con los criterios especificados.

Operadores relaciones

Operador	Descripción	Ejemplo: **condición**
=	Verdadero si es Igual a	**Where** campo = "valor"
<>, !=	Verdadero si es Diferente	**Where** campo <> "valor"
>	Verdadero si es Mayor que	**Where** campo > "valor"
<	Verdadero si es Menor que	**Where** campo < "valor"
>=	Verdadero si es Mayor o igual	**Where** campo >= "valor"
<=	Verdadero si es Menor o igual	**Where** campo <= "valor"

Operadores lógicos

Operador	Descripción
AND	Verdadero (true) si dos o más condiciones son verdaderas (true)
OR	Verdadero (true) si al menos una de las condiciones es verdadera (true)
NOT	Verdadero si no se cumple la condición

Operadores matemáticos

Operador	Descripción	Ejemplo en SQL
+	Suma	precio + impuesto
-	Resta	precio - descuento
*	Multiplicación	cantidad * precio_unitario
/	División	total / cantidad
%	Modulo (Residuo)	Cantidad % 2

Los operadores matemáticos se usan cuando se necesitan:

- **Calcular totales o subtotales.**
- **Aplicar descuentos**
- **Realizar estadísticas simples**
- **Mostrar resultados matemáticos directamente en los reportes**

Ejemplo (emulador https://sqliteonline.com/):

Tabla: libros_vendidos

Consulta: ¿Cuál fue el total pagado por cada libro?

Run | biblioteca.db

```
SELECT * FROM libros_vendidos
```

id	titulo	precio_unitario	cantida
1	Álgebra Moderna	150	2
2	Química en Acción	200	1
3	Introducción a S…	100	3

Run | biblioteca.db

```
SELECT titulo,
       precio_unitario,
       cantidad,
       precio_unitario * cantidad AS total_pagado
FROM libros_vendidos
```

titulo	precio_unitario	cantidad	total_paga
Álgebra Moderna	150	2	300
Química en Acción	200	1	200
Introducción a SQL	100	3	300

Ordenamiento y Límites.

Cuando se trabaja con grandes volúmenes de datos se puede hacer uso de comandos para organizar resultados y limitar cuando aparecen.

Commando SQL	Descripción
ORDER BY:	Ordena los resultados por una o más columnas.
LIMIT	Restringe el número de registros devueltos

Ejemplo:

Tabla: libros_vendidos

```
SELECT * FROM libros_vendidos
```

id	titulo	precio_unitario	cantida
1	Algebra Moderna	150	2
2	Química en Acción	200	1
3	Introducción a S…	100	3
4	Física I	200	4
5	Python	230	3
6	Algebra de Baldor	500	2

Consulta: ¿Ordena los títulos Ascendentemente y muestra los primeros 3 libros vendidos?

```
SELECT * FROM libros_vendidos ORDER BY titulo ASC LIMIT
```

id	titulo	precio_unitario	cantidad
1	Algebra Moderna	150	2
6	Algebra de Baldor	500	2
4	Física I	200	4

El **ORDER BY** puede hacer uso de la sintaxis **DESC** para ordenar de forma descendente y **ASC** para ordenar Ascendentemente.

Funciones de agregación (COUNT, SUM, AVG, MAX, MIN)

Las funciones de agregación permiten calcular valores sobre un conjunto de datos que permite resumir información numérica.

FUNCIÓN	QUÉ HACE	Ejemplo
COUNT()	Cuenta el número de registros	SELECT COUNT(*) FROM libros_vendidos
SUM()	Suma valores de una columna	SELECT SUM(precio_unitario) FROM libros_vendidos
AVG()	Calcula el promedio	SELECT AVG(precio_unitario) FROM libros_vendidos
MAX()	Encuentra el valor máximo	SELECT MAX(precio_unitario) FROM libros_vendidos
MIN()	Encuentra el valor mínimo	SELECT MIN(precio_unitario) FROM libros_vendidos

Ejemplo utilizando la **Tabla**: **libros_vendidos:**

Consulta: ¿Libro con más ventas?

```
SELECT titulo, max(cantidad) AS más_ventas
FROM libros_vendidos
```

titulo	más_ventas
Introducción a Física	4

Consulta: ¿Cuántos libros en total se vendieron?

```
SELECT SUM(cantidad) AS total_libros_vendidos
FROM libros_vendidos
```

total_libros_vendidos
12

Agrupamiento con GROUP BY y Filtrado con HAVING

GROUP BY: Esta sentencia al realizar una consulta se encarga de agrupar los registros por un criterio.

Consulta

```
SELECT fecha_venta, SUM(cantidad) AS libros_vendidos
FROM libros_vendidos
GROUP BY fecha_venta;
```

Resultado

fecha_venta	libros_vendidos
2025-04-01	2
2025-04-02	3
2025-04-03	5
2025-04-04	2

HAVING: Filtra después del agrupamiento.

SELECT ciudad, COUNT(*) FROM clientes GROUP BY ciudad HAVING COUNT(*) > 10

UNION ALL: operador que combina los resultados de dos o más consultas **SELECT** y muestra todos los registros incluidos los duplicados.

Ejemplo utilizando Group BY

Tabla: libros_vendidos

```
SELECT *FROM libros_vendidos
```

id	titulo	autor	precio_unitario	cantidad	fecha
1	Álgebra Moderna	Mario López	150	2	2025-0
7	SQL Básico	Ana Torres	120	1	2025-0
8	Química en Acción	Carlos Pérez	200	2	2025-0
2	SQL Básico	Ana Torres	120	3	2025-0
9	Introducción a Física	Laura Gómez	180	2	2025-0
3	Química en Acción	Carlos Pérez	200	1	2025-0
4	Introducción a Física	Laura Gómez	180	4	2025-0
6	Álgebra Moderna	Mario López	150	1	2025-0
10	Bases de Datos Avanzadas	Sofía Ramírez	250	3	2025-0
5	Bases de Datos Avanzadas	Sofía Ramírez	250	2	2025-0

Consulta ¿Agrupar los libros vendidos por fecha de venta?

```
SELECT
    fecha_venta,
    SUM(cantidad) AS total_vendido
FROM libros_vendidos
GROUP BY fecha_venta
```

fecha_venta	total_vendidos
2025-04-01	5
2025-04-02	5
2025-04-03	9
2025-04-04	2

Consulta con JOIN (INNER, LEFT, RIGHT, FULL)

Los **JOINS** permiten combinar datos de múltiples tablas.

- **INNER JOIN: Devuelve registros que coinciden en ambas tablas.**

  SELECT clientes.nombre, pedidos.fecha FROM clientes

  INNER JOIN pedidos ON clientes.id = pedidos.id_cliente;

- **LEFT JOIN: Devuelve todos los registros de la primera tabla y los coincidentes de la segunda.**

  SELECT clientes.nombre, pedidos.fecha FROM clientes

  LEFT JOIN pedidos ON clientes.id = pedidos.id_cliente;

- **RIGHT JOIN: Devuelve todos los registros de la segunda tabla y los coincidentes de la primera.**

  SELECT clientes.nombre, pedidos.fecha FROM clientes

  RIGHT JOIN pedidos ON clientes.id = pedidos.id_cliente;

Ejemplo:

Tabla: libros

Biblioteca.db

```
1 SELECT * FROM libros
```

id	titulo	autor_id	disponible
1	Cien años de soledad	1	1
2	La casa de los espíritus	2	1
3	Viaje al centro de la Tierra	3	0
4	El amor en los tiempos del cólera	1	0

Tabla: Ventas

Biblioteca.db

```
1 SELECT * FROM ventas
```

id_venta	id_libro	cantidad	fecha
1	1	3	2025-04-10
2	1	5	2025-04-11
3	2	2	2025-04-12
4	2	1	2025-04-13
5	2	3	2025-04-10
6	3	5	2025-04-11
7	4	2	2025-04-12
8	1	1	2025-04-13

Consulta para mostrar solos los libros que se han vendido

Biblioteca.db

```
SELECT libros.titulo, ventas.cantidad, ventas.fecha
FROM libros
INNER JOIN ventas ON libros.id = ventas.id_libro;
```

titulo	cantidad	fecha
Cien años de soledad	3	2025-04-10
Cien años de soledad	5	2025-04-11
La casa de los espíritus	2	2025-04-12
La casa de los espíritus	1	2025-04-13
La casa de los espíritus	3	2025-04-10
Viaje al centro de la Tierra	5	2025-04-11
El amor en los tiempos del cólera	2	2025-04-12
Cien años de soledad	1	2025-04-13

Uniones Múltiples (3+ Tablas)

Podemos hacer **JOINS** entre más de dos tablas.

- **SELECT clientes.nombre, pedidos.fecha, productos.nombre FROM clientes**
- **INNER JOIN pedidos ON clientes.id = pedidos.id_cliente**
- **INNER JOIN productos ON pedidos.id_producto = productos.id;**

Subconsultas

Las subconsultas permiten hacer consultas dentro de otras consultas.

```
SELECT nombre FROM clientes WHERE id IN (
    SELECT id_cliente FROM pedidos WHERE total > 100
);
```

Actividad 4. Gestión de Videojuegos

Instrucciones: *Dado la tabla de videojuegos, analizar y escribir lo que realiza cada sentencia SQL*

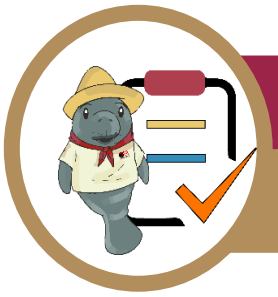
📋 El Escenario: La Tabla `videojuegos`

Imagina que tenemos una tabla llamada `videojuegos` con los siguientes datos iniciales:

id_juego	titulo	plataforma	precio	stock
1	Minecraft	PC	300	15
2	FIFA 26	PS5	1400	8
3	Zelda: TotK	Switch	1200	5

Identifica que hace los siguientes códigos:

Sentencias	¿Qué realiza?
`INSERT INTO videojuegos (titulo, plataforma, precio, stock) VALUES ('Elden Ring', 'Xbox', 1100, 10);`	
`SELECT titulo, precio, stock FROM videojuegos WHERE plataforma = 'PS5';`	
`UPDATE videojuegos SET precio = 900 WHERE titulo = 'Zelda: TotK';`	
`DELETE FROM videojuegos WHERE titulo = 'Minecraft';`	



INSTRUMENTO DE EVALUACIÓN - RÚBRICA

Actividad 4: Gestión de Videojuegos

DATOS GENERALES	
Nombre(s) del alumno (s)	**Matricula(s)**
Producto: Interpretación de sentencias	**Fecha**
Formación laboral básica: TIC	**Periodo:2026-2027A**
Nombre del docente:	**Firma del docente**

Criterio	SI (2.5)	NO	Observaciones
Determina con precisión la función del comando **INSERT**			
Determina con precisión la función del comando **SELECT**			
Determina con precisión la función del comando **UPDATE**			
Determina con precisión la función del comando **DELETE**			
TOTAL			

Logros:	**Aspectos de Mejora:**

Referencias

Rosas Hernández, O. A. (2018). *Base de datos, programación*.

Gómez Fuentes, M. del C. (2013). *Notas del curso Bases de datos*. Universidad Autónoma Metropolitana, Unidad Cuajimalpa.

Marqués, M. (2011). *Base de datos* (1.ª ed.). Universitat Jaume I.

Galvis Lista, E., & Bustamante Martínez, A. (2023). *Base de datos relacionales: Con un enfoque aplicado y orientado a resultados de aprendizaje*.

SQLiteOnline. (s.f.). *SQLiteOnline*. https://sqliteonline.com/

LECTURA 5. Diseño y Desarrollo de Base de datos

Instrucciones: *Realiza el análisis de la siguiente información, posteriormente, completa la Actividad 5.1. Infografía por etapas.*

5.1 Proceso de diseño de bases de datos

> El diseño de una base de datos consiste en definir la estructura de los datos que debe tener la base de datos de un sistema de información determinado. En el caso relacional, esta estructura será un conjunto de esquemas de relación con sus atributos, dominios de atributos, claves primarias, claves foráneas, etc. (Casas Roma, 2019)

De acuerdo con Costal Costa (2002), El diseño de una base de datos "es el proceso en el que se define la estructura de los datos que debe tener la base de datos de un sistema de información determinado."

El proceso de diseño de la base de datos consiste en **identificar** y **especificar** los objetos que la conformarán y las propiedades de estos objetos que necesitaremos para estructurar los datos.

Sin embargo, el diseño de base de datos no es una tarea sencilla, debido a los requerimientos del sistema de información y así como la misma complejidad de la información que hacen de la tarea de diseñar una base de datos sea por sí mismo un proceso complicado, para esto usamos un método ampliamente usado en la programación estructurada u orientada a objetos, que conocemos como "Divide y vencerás", que es dividir el problema a resolver en partes más pequeñas, que se hacen más sencillas de resolver, pero que su fin es el de solucionar el problema inicial.

Es al aplicar este concepto donde podemos obtener diferentes fases o etapas para abordar el diseño de la base de datos, las secuenciales o procesos que inician al terminar la anterior, así como que la última etapa se convierte en el diseño final o el resultado de todas las etapas en conjunto.

Obtenemos las siguientes etapas:

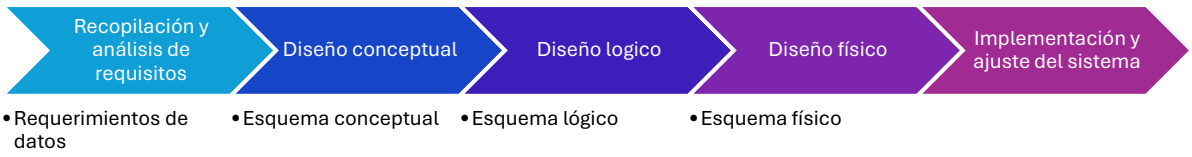


Fig. 1 Etapas de diseño de base de datos

1. **Recopilación y análisis de datos:**

En esta etapa **se cuenta con diferentes técnicas de recopilación de datos** (encuestas, revisión documental, entrevistas, reingeniería de sistemas de información anteriores, observación, entre otros) **con los usuarios potenciales de la base de datos, y de los programas y aplicaciones que interactuarán con ella.** Esto permitirá visualizar y obtener una perspectiva completa de los requerimientos, así como las restricciones a las que estará sujeta la base de datos a diseñar, así como comprender y determinar qué datos se almacenan en la base de datos, así como el procesamiento más frecuente y los resultados requeridos para una correcta toma de decisiones.

2. **Diseño conceptual:**

En esta fase o etapa del diseño **se describen los datos que se almacenarán en la base de datos y las restricciones a las que se someterán.** Siendo una forma de las más usadas el **Diagrama Entidad-Relación (DER**), ya que debido a su sencillez facilita el diseño de una estructura lógica, que posteriormente, pasaremos a una Base de Datos Relacional implementada en algunos de los Sistemas Gestores de Base de Datos (SGBD), como MySQL o SQLite.

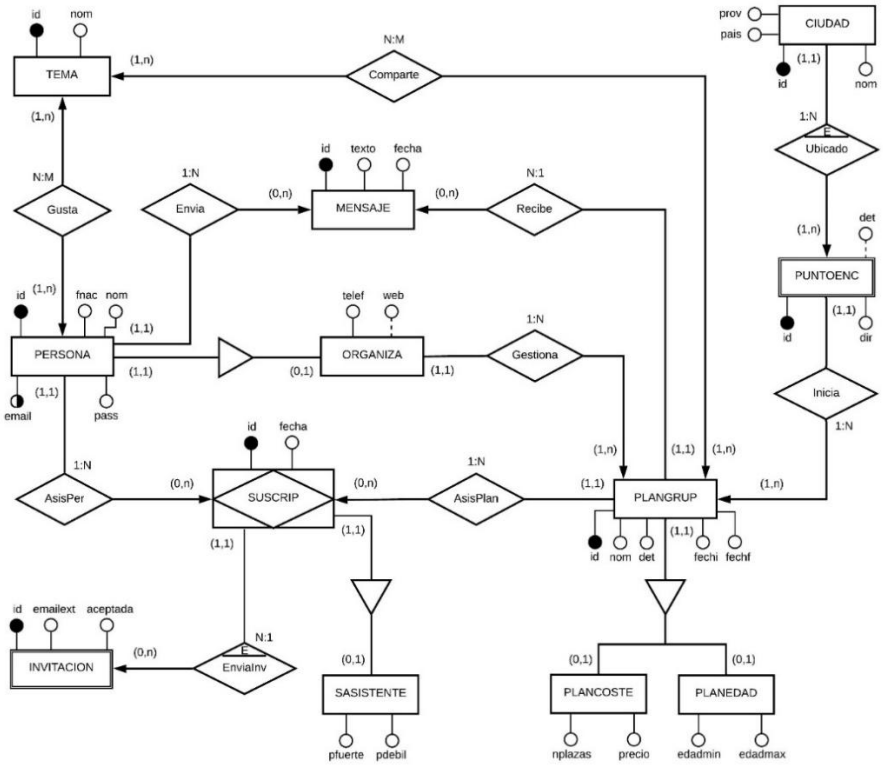


Imagen 1 – Modelo Entidad-Relación. León Romero, (2021)

3. **Diseño lógico:**

Con el esquema conceptual, que se ha visto será el **Modelo Entidad-Relación** (MER) o **Diagrama Entidad-Relación** (DER) se **busca un esquema lógico que se adapte al modelo de datos en él está basado en el SGBD, que crearemos las tablas que se generarán a partir de las entidades y relaciones del DER y especificar sus identificadores o claves primarias o principales y claves secundarias o foráneas.** Ya en lecturas anteriores se ha propuesto, para su proyecto el **Modelo Relacional**, que, en otras palabras, representará las entidades y algunas relaciones en tablas y los atributos en campos o columnas de estas tablas.

También hay que establecer las reglas para el comportamiento de las claves, a fin de asegurar la integridad referencia entre nuestras tablas, quiere decir que, todas las claves secundarias de una tabla deben corresponder a la clave primaria de la tabla con la que se relaciona y si esta clave primaria se elimina, especificando si queremos que la eliminación se de en cascada, significa, que al eliminar la clave primaria, también lo hagan las claves secundarias que depende de la misma o restringirla, de tal manera que si se intenta eliminar una clave primaria que haga referencias a claves secundarias, no permita la eliminación, no así para las claves primarias que no tengan referencia a claves secundarias; definir si la clave secundaria puede aceptar nulos o claves primarias no existentes; así como el permitir las modificaciones a la clave secundaria (en cascada), en caso de que la clave primaria de la que depende se modifique.

Siendo uno de los más utilizados actualmente, se optó por el modelo relacional, sin embargo, no es el único, teniendo, según León Romero (2021), además:

- **Modelo jerárquico**
- **Modelo de red**
- **Modelo orientado a objetos**

La página de diseño de diagramas Lucidchart agrega:

- **Modelo Entidad-Relación**
- **Modelo de documentos**
- **Modelo Entidad-Atributo-Valor**
- **Esquema estrella**
- **Modelo relacional de objetos**

Esta etapa del diseño puede dividirse en dos fases:

- **Diseño lógico estándar:** el diseño lógico que se obtiene es mediante el uso de un lenguaje de definición de datos estándar, SQL, por ejemplo y no depende de un SGBD comercial.

- **Diseño lógico específico:** a diferencia del diseño lógico estándar, en esta fase se hace uso de una SGBD comercial, por ejemplo, MySQL, Microsoft SQL Server, Oracle, entre otros. También se conoce como etapa de implementación de la BD.

Para obtener un diseño adecuado, es recomendable pasarlo por un proceso de normalización para obtener datos atómicos, evitar la duplicidad de datos y eliminar la redundancia de datos que podrían provocar inconsistencias cuando se realicen operaciones con los datos contenidos.

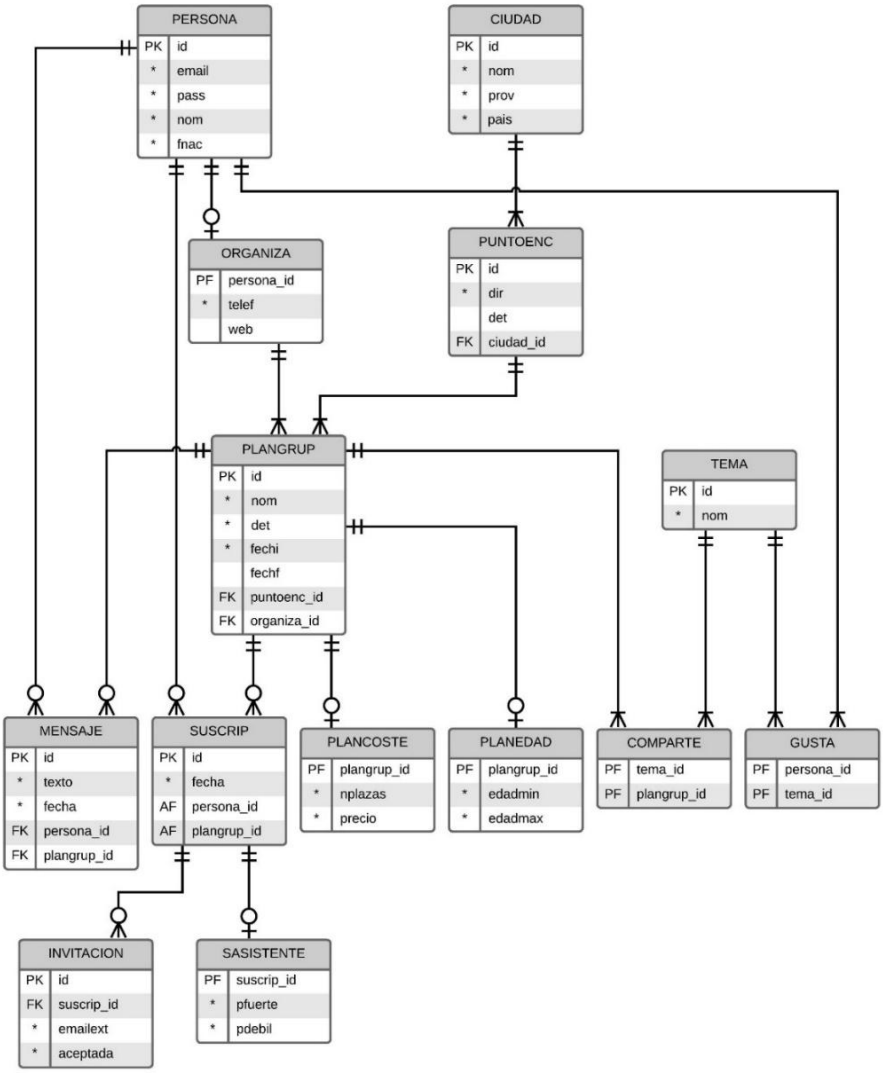


Imagen 2 – Modelo Relacional. León Romero, (2021)

Aquí podemos utilizar una manera de documentar los datos, a fin de que se pueda dar un uso correcto de estos, esto es un **diccionario de datos**. Este método se realiza en la búsqueda de poder

especificar los metadatos de los atributos de una tabla o entidad, este podemos hacerlo mediante una tabla:

Tabla: Nombre de la tabla					
Nombre del Campo	**Tipo de Dato**	**Longitud**	**Clave**	**Permite Nulos**	**Descripción / Reglas de Negocio**

4. **Diseño físico:**

Siendo la última fase del diseño de base de datos, pasará del diseño lógico a las tablas de nuestra BD, con una vista gráfica de diseño, como en el caso de Microsoft Access, y mediante el lenguaje SQL, a través de un SGBD.

5. **Implementación y ajuste del sistema:**

Esta fase busca instalar la BD, que va desde la elección de un motor de base de datos o un SGBD, por ejemplo, MySQL, Oracle, SQL Server, entre otros, así como los parámetros que tienen que ver con la configuración básica, es decir, donde se ubicarán los archivos de la BD, la definición de los usuarios, sus permisos y privilegios de acuerdo con los roles que tomen en el manejo de la BD y si es una BD distribuida, tenemos que hacer la configuración de la red.

Implica también la instalación de los archivos y datos iniciales para pruebas por parte de los usuarios.

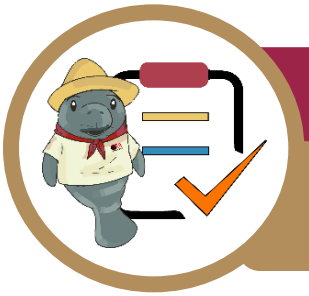
En cuanto al ajuste del sistema, los analistas y usuarios del sistema de BD evalúan su funcionamiento, integridad, consultas, el uso de recursos del equipo de cómputo y red para determinar y corregir los fallos.

En caso de que se ajusten los procesos de captura, el procesamiento y salida de los datos, y se verificarán que las consultas sean correctas, eficaces y eficientes, considerando que pueden llevarse modificaciones en la estructura de estas, hasta eliminar consultas que puedan resultar repetitivas o innecesarias.

Además de lo antes dicho, nuestro trabajo implica el monitoreo frecuente del sistema, para recibir retroalimentación de los usuarios en cuanto a posibles fallos que pudieran presentarse.

Actividad 5.1. Infografía por etapas

Instrucciones: *Integrados en equipos de 5 estudiantes, diseña una infografía en una herramienta de diseño (Canva, Genial.ly, etc.) o de forma manual donde explique brevemente cada una de las etapas del proceso de diseño de la BD y lo expone en plenaria para su evaluación*



INSTRUMENTO DE EVALUACIÓN

Rubrica

Actividad X - Infografía por etapas

DATOS GENERALES	
Nombre(s) del alumno (s)	**Matriculas(s)**
Producto: Infografía en formato tabloide	**Fecha**
Materia:	**Periodo:2025-2026A**
Nombre del docente:	**Firma del docente**

CRITERIO	INSUFICIENTE	SUFICIENTE	BUENO	EXCELENTE	TOTAL
Contenido: Etapas del diseño de BD	Presenta menos de 1 a 2 etapas, incompletas o incorrectas.	Presenta 3 etapas con información parcial o superficial.	Presenta 4 etapas con información clara, aunque algo limitada.	Presenta las 5 etapas completas, explicadas con claridad y precisión técnica.	
25 pts.	*10 pts.*	*15 pts.*	*20 pts.*	*25 pts.*	
Ortografía y redacción	Múltiples errores ortográficos o gramaticales que dificultan la comprensión.	Algunos errores, pero la idea general se entiende.	Pocos errores menores que no afectan el entendimiento.	Sin errores ortográficos ni gramaticales, redacción clara y coherente.	
15 pts.	*4 pts.*	*8 pts.*	*12 pts.*	*15 pts.*	
Diseño y creatividad	Diseño desorganizado, colores mal combinados, sin originalidad.	Diseño básico con poco atractivo visual.	Diseño atractivo, uso adecuado del color y organización.	Diseño llamativo, original, equilibrado y profesional.	
15 pts.	*4 pts.*	*8 pts.*	*12 pts.*	*15 pts.*	
Imágenes y recursos visuales	No se incluyen imágenes o son irrelevantes/confusas.	Pocas imágenes, algunas relevantes.	Buen uso de imágenes relacionadas con el contenido.	Imágenes claras, bien seleccionadas y complementan perfectamente el contenido.	
10 pts.	*2.5 pts.*	*5 pts.*	*7.5 pts.*	*10 pts.*	
Trabajo colaborativo	Poco o ningún trabajo en equipo, no todos	Participación desigual;	La mayoría participó	Todos los integrantes	

	participaron.	algunos miembros hicieron más que otros.	activamente, con buena coordinación.	participaron equitativamente con excelente coordinación.	
15 pts.	*4 pts.*	*8 pts.*	*12 pts.*	*15 pts.*	
Presentación oral	No se comprende, lectura mecánica, sin seguridad.	Explican lo básico, con poca fluidez y seguridad.	Explicación clara, aunque con uso de apoyo escrito o nerviosismo.	Presentación clara, segura, fluida y bien distribuida entre todos los integrantes.	
10 pts.	*2.5 pts.*	*5 pts.*	*7.5 pts.*	*10 pts.*	
Uso de herramienta o material	No utilizaron herramientas adecuadas o el cartel está desorganizado.	Uso básico de herramienta o cartel funcional, pero sin impacto visual.	Herramienta bien usada o cartel organizado y presentable.	Uso creativo y profesional de herramienta o cartel, con alto impacto visual.	
10 pts.	*2.5 pts.*	*5 pts.*	*7.5 pts.*	*10 pts.*	

5.2 Creación de diagramas entidad-relación (DER)

El Diagrama Entidad – Relación (DER) **es una forma de representar la BD en el modelado de Base de Datos, que representa visualmente los objetos o entidades importantes de nuestra BD, así como sus atributos, las relaciones entre entidades y su cardinalidad**. Al realizar nuestro diseño, es fundamental.

Recordemos los pasos para el diseño de la BD:

1. **Identificar las entidades:** este paso consiste en determinar qué elementos intervienen dentro de un sistema y que son necesarias para proporcionar información.
2. **Añadir sus atributos:** ahora identifica los detalles o propiedades de cada una de las entidades, este paso es importante para que podamos dar la información necesaria de cada entidad, sin exagerar en los detalles, sino buscar ser concretos.
3. **Identificar las relaciones entre las entidades:** con una palabra que colocaremos dentro de la forma de la relación escribiremos el tipo de interacción que existe entre dos o más entidades relacionadas.
4. **Definir la cardinalidad:** ahora determina el número de veces que una entidad se relaciona con la otra entidad.
5. **Organizar y visualizar:** para este paso puedes usar aplicaciones donde puedas dar un mejor orden a tu DER, así como mejorar su diseño (colores, tipografía, orden) para que pueda ser entendible.

Existen un sinfín de aplicaciones de las que podemos hacer uso el diseño de nuestro DER de forma sencilla, ya sea para nuestros dispositivos móviles o para PC en forma Online, sin embargo, en este caso, cuentan con las desventajas de no tener funciones avanzadas en el modo gratuito o no tener modo gratuito, sin embargo, en la paquetería ofimática algunos programas, como procesadores de texto y/o presentadores electrónicos cuentan con la opción de insertar formas y podemos apoyarnos en ellos.

Algunas herramientas online son:

Boardmix.com **Lucidchart.com** **Canva.com** **Miro.com**

Algunas herramientas para el SO Android son:

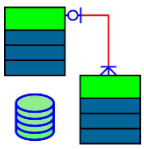
Flowdia

Lucidchart

EdrawMax

Canva.com

Algunas herramientas para el iOS son:



SQL-ER-Diagram

Lucidchart

Canva.com

2.

6. **Actualizar: un DER no es constante, puede sufrir cambios o modificaciones durante el proceso, por lo que se sujetará a estos cambios, siendo estos pasos iterativos, que pueda darse como un ciclo.**

Database designer:

Es una App distribuida a través de la tienda de aplicaciones de Google (PlayStore) y creada por KLIM, bajo la bandera de ser una App gratuita de por vida, además que tiene buena puntuación de 4.8 / 5 estrellas y leyéndose comentarios sobre la sencillez de la aplicación, así como tener lo básicamente esencial para su uso, haciendo que no sea una App complicada.

Esta App permite el diseño lógico de la BD y permite que podamos exportarlo a SQL, ya sea al portapapeles, como a un archivo *.txt

Práctica guiada 5.2. Mis relaciones y yo

Instrucciones: *El estudiante en forma individual y apoyándose en el DER realizado en la Actividad 2 y el diccionario de datos elaborado en la lectura anterior elabora un DER en esquema lógico que plasme las entidades, atributos, claves primarias, secundarias y relaciones de las entidades de su BD, haciendo uso de la App Database Designer y exportarlo al diseño físico mediante el lenguaje SQL siguiendo las instrucciones de esta práctica.*

1. **Abre la App y en el botón Añadir (+) crea una nueva BD, para este caso le pondremos el nombre Reservas, colocaremos una pequeña descripción y definiremos el SGBD al que lo exportaremos al finalizarlo, para este ejemplo, usaremos elige SQLite y presiona Mantener, observa el ejemplo:**

2. **Ahora abre el proyecto dando un Tap sobre su nombre para abrirla y comenzar a diseñar la estructura.**

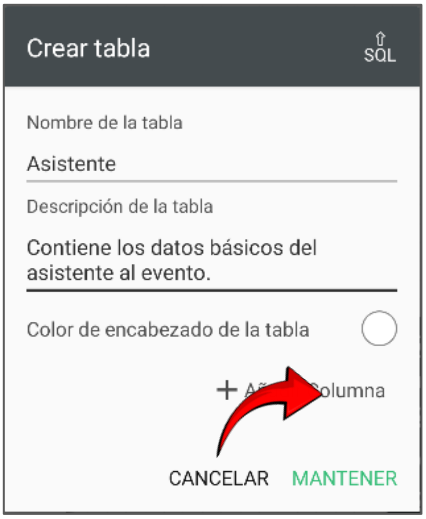
3. **Presiona el botón Crear tabla**  **y define el nombre de la tabla (Asistente) de acuerdo con nuestro DER además de su descripción y define un color para el encabezado (opcional)**

Para sus atributos nos vamos a basar en el diccionario de datos diseñado en la lectura anterior:

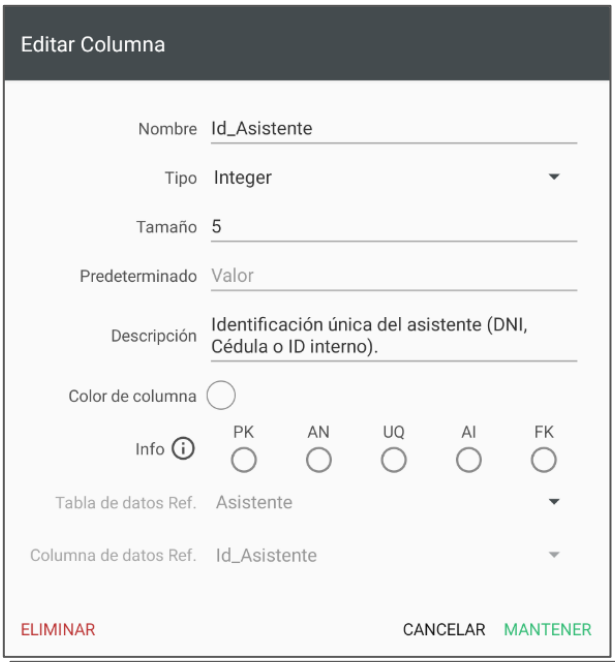
Tabla: Asistentes

Nombre del Campo	Tipo de Dato	Longitud	Clave	Permite Nulos	Descripción / Reglas de Negocio
Id_Asistente	VARCHAR / INT	5	**PK**	NO	Identificación única del asistente (DNI, Cédula o ID interno).
Nombre	VARCHAR	40	-	NO	Primer y segundo nombre del asistente.
Correo	VARCHAR	40	-	NO	Correo electrónico. Debe ser único para evitar duplicados.

4. **Presiona la opción +Añadir Columna para crear las columnas o campos**



De acuerdo con los atributos propuestos en el diccionario para cada tabla (entidad), define su nombre, tipo de dato, tamaño, valor predeterminado (opcional), descripción, color de columna (opcional)

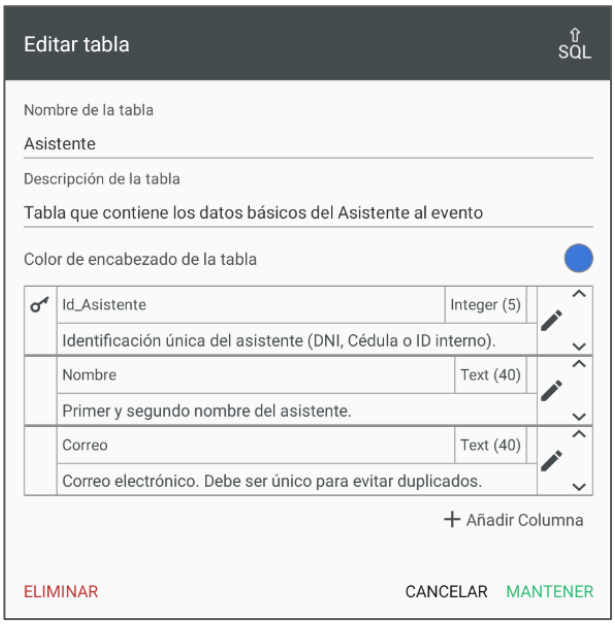


En este caso es una **clave primaria** así que selecciona la casilla **PK**

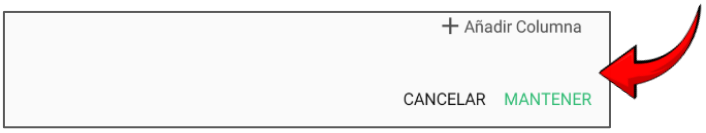
Notarás que al atributo o campo que definiste como llave primaria aparecerá

con un icono de una llave .

5. **Captura los demás campos, para esto presiona en la opción +Añadir Columna y escribe los datos que se te solicitan, recuerda apoyarte en tu diccionario de datos y repite el proceso para cada uno de los atributos, resultando en esto.**



3. **Al finalizar solo deberás presionar Mantener y lo verás como una tabla.**



Asistente			
Tabla que contiene los datos básicos del Asistente al evento			
🔑	Id_Asistente	Integer(5)	Identificación única del asistente (DNI, Cédula o ID interno).
	Nombre	Text(40)	Primer y segundo nombre del asistente.
	Correo	Text(40)	Correo electrónico. Debe ser único para evitar duplicados.

6. **Ahora repite los pasos 4 y 5 para la tabla Salas.**

Al finalizar quedarán de la siguiente forma, solo deberás presionar **Mantener**

Sala			
Salas en el edificio destinadas a eventos.			
🔑	Id_Sala	Integer(6)	Identificador de la sala, clave principal, Autoincremental.
	Capacidad	Integer(2)	Número del asiento en cada sala, del 1 al 99.

Para mover la tabla, solo presionamos la tabla y la movemos al sitio en el que queremos que quede.

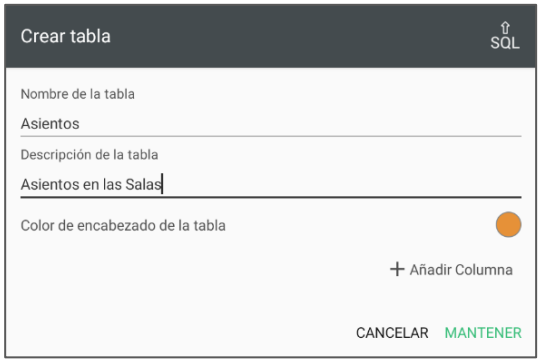
Sala			
Salas en el edificio destinadas a eventos.			
🔑	Id_Sala	Integer(6)	Identificador de la sala, clave principal, Autoincremental.
	Capacidad	Integer(2)	Número del asiento en cada sala, del 1 al 99.

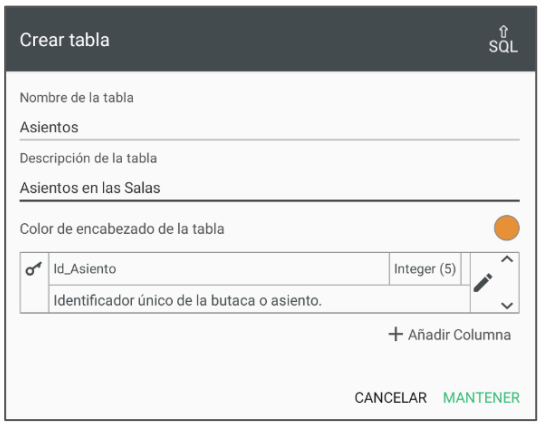
7. **Ya aprendiste a capturar atributos y clave primaria, ahora vamos a aprender a capturar una clave secundaria, así como a crear la relación entre dos tablas.**

Vamos a capturar la tabla Asientos que tiene una llave secundaria relacionada con la tabla Salas.

Presiona el botón **Crear tabla** y define el nombre de la tabla (Asientos) y escribe su descripción, puedes elegir un color.

Presiona la opción **+Añadir Columna** y captura el primer atributo que es una llave primaria siguiendo los pasos para esto.

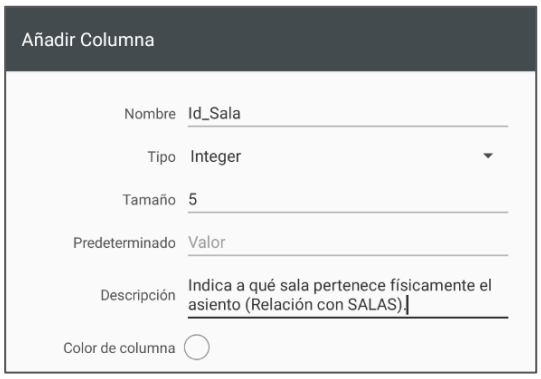


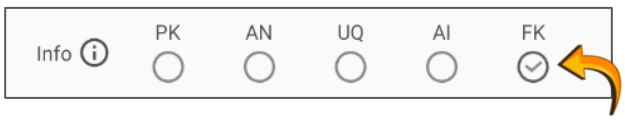


Ahora vamos a capturar la primera llave secundaria de la Tabla Asiento que es Id_Sala que se relaciona con la tabla Sala.

7.1 Después de presionar la opción +Añadir Columna captura el nombre del atributo, su tipo y tamaño (debe ser el mismo tipo que en su tabla principal), así como su descripción.

7.2 En la sección Info selecciona la opción FK (Foreign Key o llave foránea /

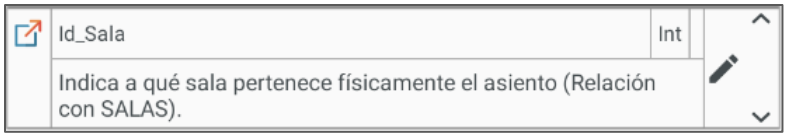




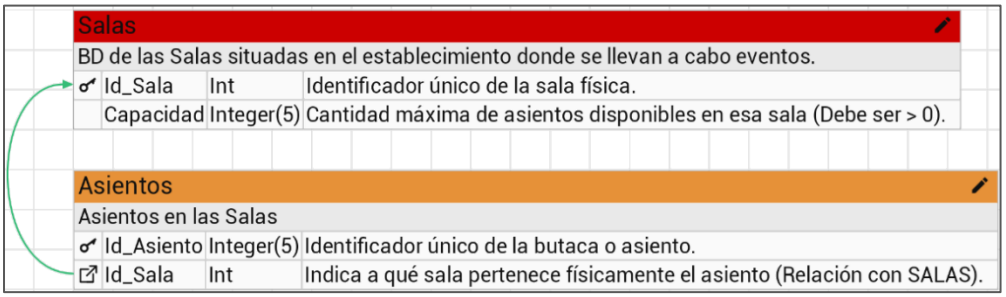
secundaria)

7.3 Ahora define a que tabla y campo hace referencia, así como sus restricciones al momento de actualizar y eliminar.

7.4 Si ahora presionas Mantener verás que la llave secundaria llevará el icono , quedando



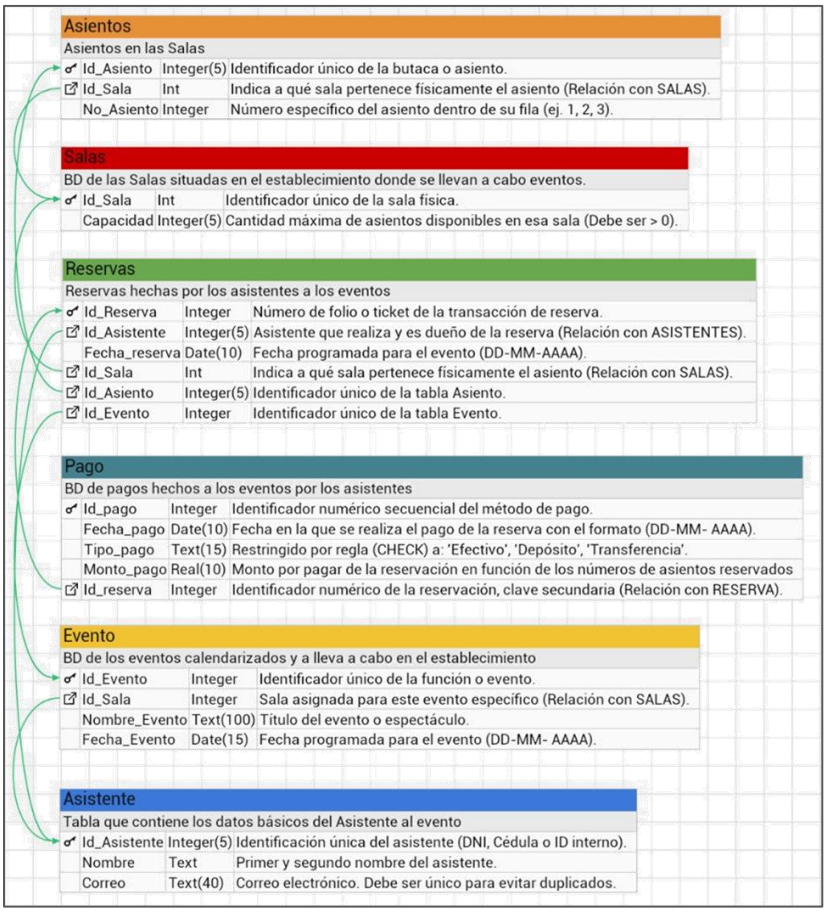
7.5 Si presionamos de nuevo Mantener veremos que las tablas aparecerán enlazadas con una flecha



Termina de capturar los demás atributos de la misma forma, quedando:

Termina de capturar las demás tablas atributos de la misma forma, quedando:

Cuando tengamos todas las entidades capturadas y las relaciones que hay entre tablas podremos ver lo siguiente.



Asientos
Asientos en las Salas

Id_Asiento	Integer(5)	Identificador único de la butaca o asiento.
Id_Sala	Int	Indica a qué sala pertenece físicamente el asiento (Relación con SALAS).
No_Asiento	Integer	Número específico del asiento dentro de su fila (ej. 1, 2, 3).

Salas
BD de las Salas situadas en el establecimiento donde se llevan a cabo eventos.

Id_Sala	Int	Identificador único de la sala física.
Capacidad	Integer(5)	Cantidad máxima de asientos disponibles en esa sala (Debe ser > 0).

Reservas
Reservas hechas por los asistentes a los eventos

Id_Reserva	Integer	Número de folio o ticket de la transacción de reserva.
Id_Asistente	Integer(5)	Asistente que realiza y es dueño de la reserva (Relación con ASISTENTES).
Fecha_reserva	Date(10)	Fecha programada para el evento (DD-MM-AAAA).
Id_Sala	Int	Indica a qué sala pertenece físicamente el asiento (Relación con SALAS).
Id_Asiento	Integer(5)	Identificador único de la tabla Asiento.
Id_Evento	Integer	Identificador único de la tabla Evento.

Pago
BD de pagos hechos a los eventos por los asistentes

Id_pago	Integer	Identificador numérico secuencial del método de pago.
Fecha_pago	Date(10)	Fecha en la que se realiza el pago de la reserva con el formato (DD-MM- AAAA).
Tipo_pago	Text(15)	Restringido por regla (CHECK) a: 'Efectivo', 'Depósito', 'Transferencia'.
Monto_pago	Real(10)	Monto por pagar de la reservación en función de los números de asientos reservados
Id_reserva	Integer	Identificador numérico de la reservación, clave secundaria (Relación con RESERVA).

Evento
BD de los eventos calendarizados y a lleva a cabo en el establecimiento

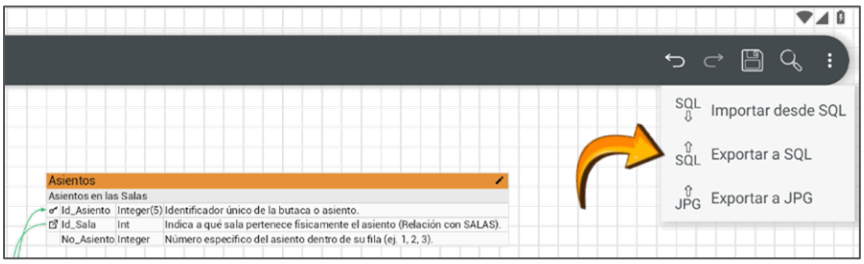
Id_Evento	Integer	Identificador único de la función o evento.
Id_Sala	Integer	Sala asignada para este evento específico (Relación con SALAS).
Nombre_Evento	Text(100)	Título del evento o espectáculo.
Fecha_Evento	Date(15)	Fecha programada para el evento (DD-MM- AAAA).

Asistente
Tabla que contiene los datos básicos del Asistente al evento

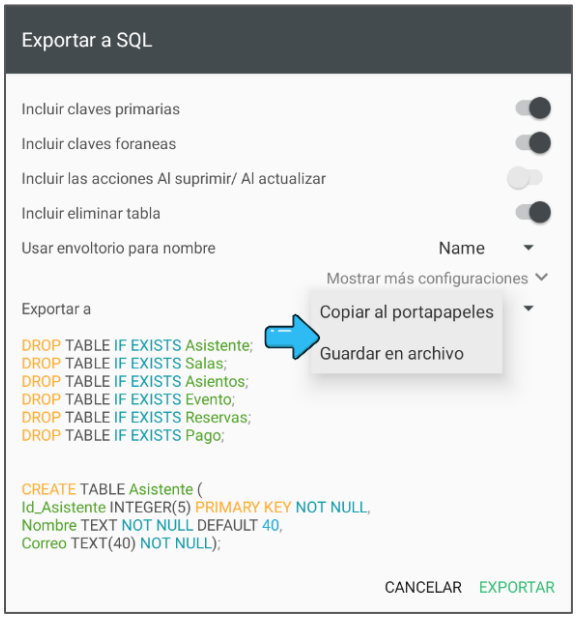
Id_Asistente	Integer(5)	Identificación única del asistente (DNI, Cédula o ID interno).
Nombre	Text	Primer y segundo nombre del asistente.
Correo	Text(40)	Correo electrónico. Debe ser único para evitar duplicados.

Ahora exporta tu BD a SQL.

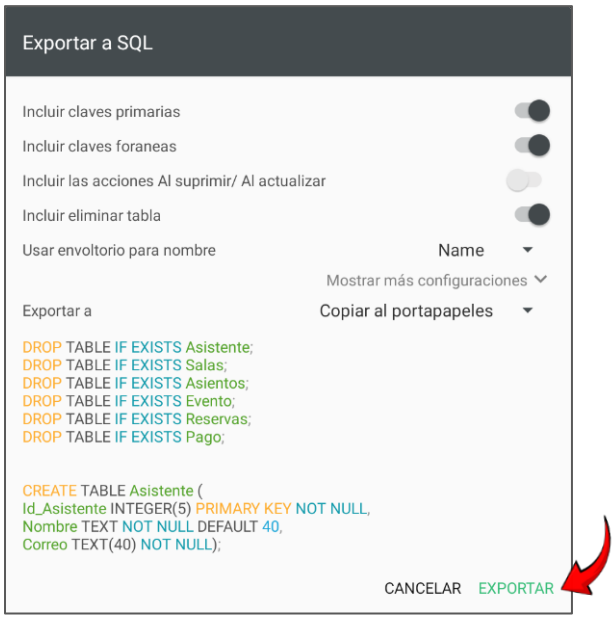
1. **Ve al Menú Exportar a SQL y activa los selectores que se muestran**



2. **Elige la forma en que quieres exportarlo, podemos hacerlo en cualquiera de las siguientes dos formas:**



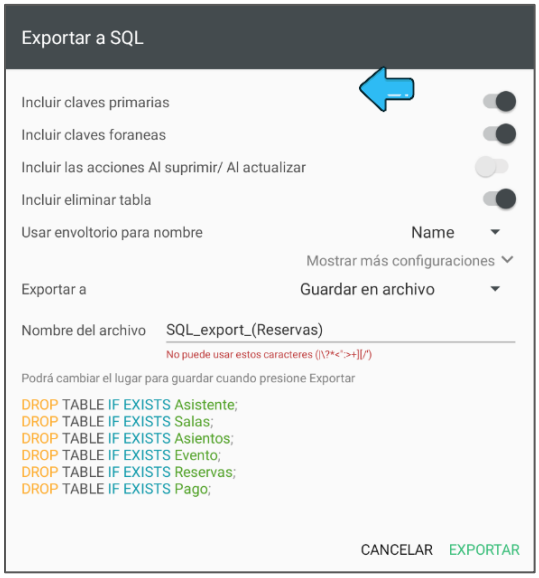
a. **Portapapeles:**
   i. **Presiona Exportar lo que copiará el código SQL a tu portapapeles para que puedas pegarlo donde desees, ya sea un documento o para compartirlo en cualquier red social o mensajería.**



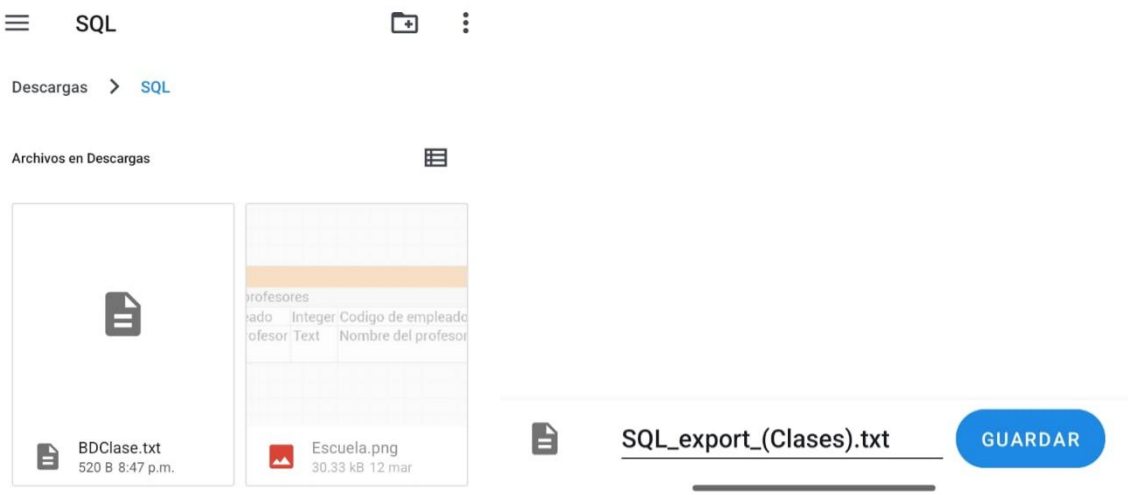
b. Guardar en archivo

i. Elige un nombre para tu documento

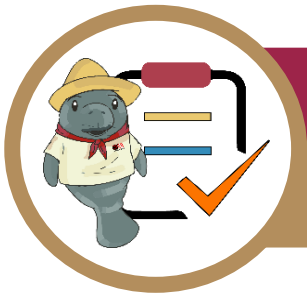
ii. Presiona Exportar



iii. Elige la carpeta donde desees que se guarde el documento, aquí también puedes modificar el nombre del documento que se guardará con la extensión .txt



Ahora puedes hacer uso de tu código SQL para colocarlo en cualquier SGBD



INSTRUMENTO DE EVALUACIÓN

Rubrica
Actividad X – Mis relaciones y yo

DATOS GENERALES	
Nombre(s) del alumno (s)	**Matriculas(s)**
Producto: Diagrama Entidad - Relación	**Fecha**
Materia:	**Periodo:2025-2026A**
Nombre del docente:	**Firma del docente**

CRITERIO	INSUFICIENTE	SUFICIENTE	BUENO	EXCELENTE	TOTAL
Entidades y relaciones	Faltan varias entidades o relaciones clave, están mal definidas o confusas.	Incluye algunas entidades y relaciones, pero con errores de estructura o relación.	Incluye la mayoría de las entidades y relaciones correctamente definidas.	Todas las entidades y relaciones están completas, claras y correctamente estructuradas.	
20 pts.	*5 pts.*	*10 pts.*	*15 pts.*	*20 pts.*	
Atributos (simples, compuestos, etc.)	Atributos incompletos, mal ubicados o incorrectamente representados.	La mayoría de los atributos están presentes, aunque con algunos errores.	Atributos bien definidos y ubicados, con pequeños detalles por corregir.	Todos los atributos son pertinentes, correctamente nombrados y representados.	
15 pts.	*4 pts.*	*8 pts.*	*12 pts.*	*15 pts.*	
Cardinalidades incorrectas o ausentes.	Cardinalidades incorrectas o ausentes.	Algunas cardinalidades están correctas, otras no se entienden o faltan.	La mayoría de las cardinalidades están correctamente representadas.	Todas las cardinalidades son correctas, claras y coherentes con el modelo.	
15 pts.	*4 pts.*	*8 pts.*	*12 pts.*	*15 pts.*	
Claves (primarias, candidatas, foráneas)	No se identifican correctamente o están ausentes.	Se identifican parcialmente, con errores de concepto o representación.	Identificación adecuada de claves primarias, foráneas y candidatas con	Todas las claves están correctamente representadas y diferenciadas.	

			mínimos errores.		
20 pts.	*5 pts.*	*10 pts.*	*15 pts.*	*20 pts.*	
Presentación del DER	Diagrama desorganizado, ilegible o con símbolos incorrectos.	Presentación básica, pero entendible; falta organización o claridad.	Presentación clara, con símbolos adecuados y organización comprensible.	Presentación impecable, profesional, limpia, con todos los símbolos estandarizados.	
10 pts.	*2.5 pts.*	*5 pts.*	*7.5 pts.*	*10 pts.*	
Trabajo colaborativo	No trabajaron en equipo, participación desequilibrada.	Participación limitada o desigual entre los miembros.	Buena colaboración, aunque algunos miembros aportaron más que otros.	Todos los miembros participaron de forma activa y equilibrada durante el trabajo.	
10 pts.	*2.5 pts.*	*5 pts.*	*7.5 pts.*	*10 pts.*	
Explicación oral del DER	No se entiende, no hay claridad ni dominio del tema.	Se explican los elementos básicos, pero con dudas y poca claridad.	Explicación clara, con conocimiento general del DER.	Explicación clara, detallada y segura, evidenciando comprensión total del modelo.	
10 pts.	*2.5 pts.*	*5 pts.*	*7.5 pts.*	*10 pts.*	

Referencias

Casas Roma, J. (2019). Introducción a la base de datos. Universitat Oberta de Catalunya.

Costal Costa, D. (2002). Introducción al diseño de base de datos. Universidad virtual.

Erickson, J. (29 de agosto de 2024). MySQL: qué es y cómo se usa. https://www.oracle.com/: https://www.oracle.com/mx/mysql/what-is-mysql/

León Romero, B. (2021). El libro práctico de Base de datos. (B. L. Rmero, Ed.) España: Blas León Rmero. https://libropracticodebasesdedatos.com/descargas/primeraspaginas.pdf

Microsoft. (2010). Tareas básicas en Access 2010. https://support.microsoft.com/: https://support.microsoft.com/es-es/office/tareas-b%C3%A1sicas-en-access-2010-268acfed-2484-4822-acb3-c30e58045588#:~:text=Access%20es%20una%20herramienta%20que,ayudan%20a%20administrar%20la%20informaci%C3%B3n.

LECTURA 6. Implementación en un gestor de base de datos (MySQL, PostgreSQL, SQLite)

> Las BD son desarrolladas a través de Sistemas Gestores de Base de Datos (SGBD) entiéndase por estos: a la aplicación que permite a los usuarios definir, crear y mantener la BD, además de proporcionar un acceso controlado a la misma y precisamente es la herramienta seleccionada por el presente estudio.
>
> (Araujo-Inastrilla et al., 2023)

Podemos encontrar distintos SGBD tanto comerciales como libres. Como, por ejemplo:

MySQL Online:

> MySQL es un sistema de gestión de bases de datos relacionales (RDBMS) de código abierto que se utiliza para almacenar y gestionar datos. Su fiabilidad, rendimiento, escalabilidad y facilidad de uso hacen de MySQL una opción popular para los desarrolladores. De hecho, lo encontrarás en la base de aplicaciones exigentes y de alto tráfico como Facebook, Netflix, Uber, Airbnb, Shopify y Booking.com.
>
> Erickson (2024)

De acuerdo con Erickson (2024), "es un sistema de gestión de bases de datos de código abierto más popular del mundo.", cuando realizamos búsquedas en internet, muchas de las veces MySQL interviene en estas, para ello usa el **Lenguaje de Consulta Estructurado**, que por sus siglas en inglés conocemos con **SQL (Structured Query Languaje)**. Aunque muchas veces pronunciamos "my sequel", su pronunciación correcta es "My es-kiu-el", no obstante la anterior es una variación bastante común. MySQL ha estado vigente por casi 30 años y los desarrolladores confían en dos capacidades, que "son el soporte de transacciones ACID que significa (Atomicity, Consistency, Isolation and Durability) Atomicidad, Consistencia, Aislamiento y Durabilidad y su capacidad de escalado." Erickson (2024)

Además Erickson (2024), pone en claro que MySQL conlleva las siguientes ventajas:

- **El rendimiento de MySQL**
- **Su facilidad de uso**
- **Bajo costo, combinado con su capacidad de escalar de manera confiable a medida que un negocio crece.**

Lo que ha logrado que sea la BD de código abierto más popular en el mundo.

Hay que entender además que MySQL utiliza SQL como lenguaje de programación para poder manipular los datos dentro de una BD, es decir, no es un lenguaje de programación, sino un programa que utiliza a un lenguaje de programación, es a saber SQL.

Microsoft Access:

Es una herramienta de implementación y diseño de aplicaciones de BD que puede usar para realizar un seguimiento de información importante, mediante la implementación de BD relacionales, para crear inventarios, contactos profesionales o procesos empresariales, entre otros.

Lanzado en 1992, forma parte de la paquetería ofimática de Microsoft Office, teniendo las siguientes versiones:

1. **Access 1.0**
2. **Access 2.0**
3. **Access 95**
4. **Access 97**
5. **Access 2000**
6. **Access 2003**
7. **Access 2007**
8. **Access 2010**
9. **Access 2013**
10. **Access 2013**
11. **Access 2016**
12. **Access 2019**

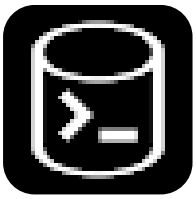
Y actualmente en la paquetería de Office 365

Además, incluye algunas plantillas, que además podemos modificar a fin de adaptarlas a nuestras necesidades. (Microsoft, 2010)

Este software por su simple forma y curva de aprendizaje es óptimo para quienes comienzan el aprendizaje de creación de las BD, ya que permite crear tablas desde la vista de datos o diseño; consultas desde la vista de diseño o mediante asistente; formularios, desde un formulario en blanco o con ayuda de un asistente; informes, desde la vista de diseño de formulario, en blanco o con ayuda de un asistente; informes, desde la vista gráfica que facilita su comprensión.

Creación de tablas y relaciones en SQL

Como vimos en al subtema anterior, un SGBD es aquel programa que nos permite la definición y creación de un BD, así como su manipulación.



En el caso del SO Android tenemos la App SQLPhone : SQL Interpreter en una aplicación educativa que nos ayudará a practicar y sobre todo a comprender el funcionamiento de las sentencias SQL sin la necesidad de un servidor, su entorno de desarrollo, aunque básico nos otorga una experiencia similar a la que tendríamos en una PC, permitiendo dos modos de trabajo, línea de comandos o con el editor.

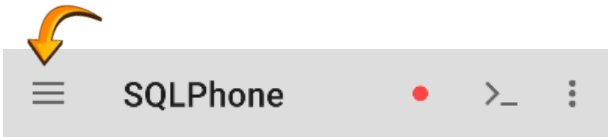
Práctica guiada 6.1. Mis relaciones y yo

Instrucciones: *El estudiante en forma individual creará parte de la BD diseñada, siguiendo las instrucciones de su docente y haciendo uso de la App SQL Phone ejecutará las sentencias SQL solicitadas.*

Usando esta aplicación vamos a crear una BD con una tabla usando lo aprendido en la lectura 4 sobre las sentencias básicas, para el ejemplo crearemos la BD Reservas y la primera tabla Asistente

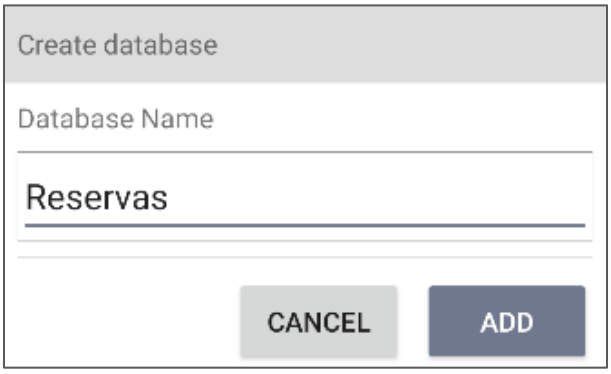
1. **Crear la BD**

Para crear la BD presiona el **Menú**



Botón **Añadir**

Escribe el nombre de la BD (Reservas) y acepta en el botón **ADD**



Ahora aparecerá el nombre de la BD y a su izquierda un punto verde que significa que la BD está conectada.

En el caso que no se vea un punto verde (BD desconectada), vamos al menú ⋮ y presiona la opción **Connect** y veremos entonces el punto verde, para desconectarla, haremos los mismo, solo que veremos la opción **Disconnect**.

2. **Crear la Tabla usando la sentencia CREATE TABLE**

Sintaxis para crear una tabla.

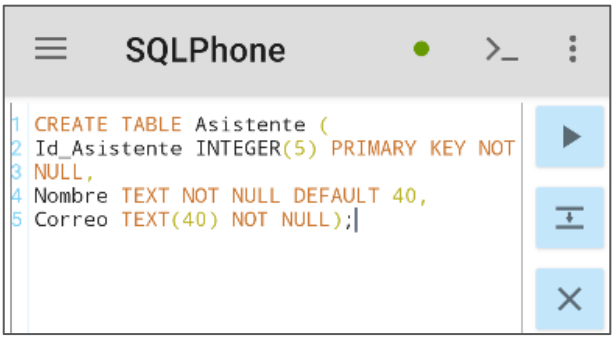
CREATE TABLE Nombre_de_la_tabla (Columna1 Tipo de dato1 (Longitud1) [Restricciones1], (Columna2 Tipo de dato2 (Longitud2) [Restricciones]);

Usaremos nuestro diccionario de datos para crear nuestra primer tabla.

Tabla: ASISTENTE

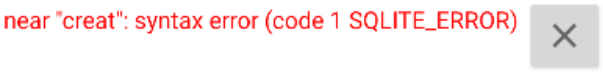
Nombre del Campo	Tipo de Dato	Longitud	Clave	Permite Nulos	Descripción / Reglas de Negocio
Id_Asistente	VARCHAR / INT	5	**PK**	NO	Identificación única del asistente (DNI, Cédula o ID interno).
Nombre	VARCHAR	40	-	NO	Primer y segundo nombre del asistente.
Correo	VARCHAR	40	-	NO	Correo electrónico. Debe ser único para evitar duplicados.

Crear la tabla Asistentes, definiendo su tipo de dato, longitud, restricciones y si es una llave primaria, quedando:



Ahora presionaremos el botón **Ejecutar**

Si aparece un texto en rojo, significará que hay un error, ya sea en el nombre de la sentencia, como es este caso o de dato, restricción, etc.



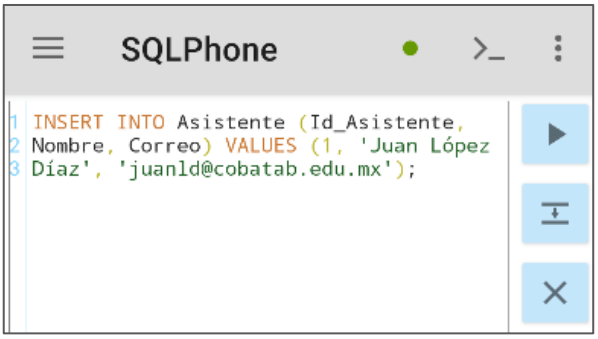
Si nuestra sentencia y sintaxis está correcta veremos el siguiente mensaje en verde:

3. **Insertar datos en la tabla**

Sintaxis para insertar datos en una tabla.

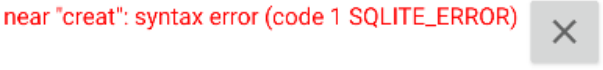
INSERT INTO nombre_tabla (columna1, columna2) **VALUES** (valor1, valor2);

Inserta los datos a la tabla Asistentes, respetando los tipos de datos y las restricciones, si el tipo de dato es texto debemos colocarlo entre comillas



Ahora presionaremos el botón **Ejecutar**

Si aparece un texto en rojo, significará que hay un error, ya sea en el nombre de la sentencia, tipo de dato u otro.

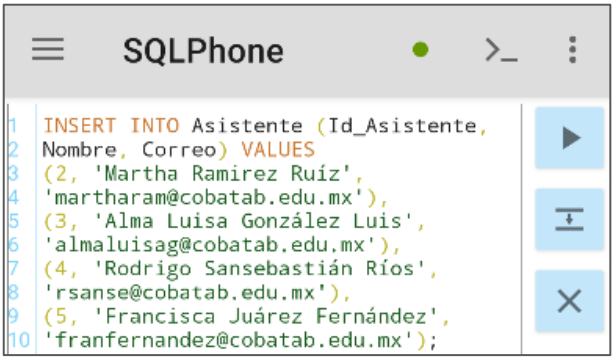


Si nuestra sentencia y sintaxis está correcta veremos el siguiente mensaje en verde:

Vamos a insertar 5 nuevos registros a nuestra tabla:

Recuerda su sintaxis para insertar datos en una tabla.
INSERT INTO nombre_tabla (columna1, columna2) **VALUES** (valor1, valor2);

Podemos insertar más de un registro con la sentencia **INSERT INTO**, solo debemos separar los registros por comas



4. **Consultar los datos de la tabla**

Sintaxis para consultar datos en una tabla.
SELECT columna1, columna2 **FROM** nombre_tabla **WHERE** condición.

Escribe la sentencia **SELECT** respetando su sintaxis, como en este caso solo no usaremos **WHERE.**

Primero solo seleccionaremos el campo Nombre.

Ahora presionaremos el botón **Ejecutar**

Resultando en esto:

nombre
juan lopez diaz
martha ramirez ruíz
alma luisa gonzález luis
rodrigo sansebastián ríos
francisca juárez fernández

Ahora vamos a seleccionar dos campos, por ejemplo, Id_Asistente y Nombre

Resultando en esto:

id_asistente	nombre
1	juan lopez diaz
2	martha ramirez ruíz
3	alma luisa gonzález luis
4	rodrigo sansebastián ríos
5	francisca juárez fernández

Si queremos seleccionar todas las columnas debemos usar * que significa "Todo"

Resultando en esto:

id_asistente	nombre	cor
1	juan lopez diaz	juar
2	martha ramirez ruíz	mart
3	alma luisa gonzález luis	alma
4	rodrigo sansebastián ríos	rsar
5	francisca juárez fernández	frar

5. **Actualizar datos en la tabla**

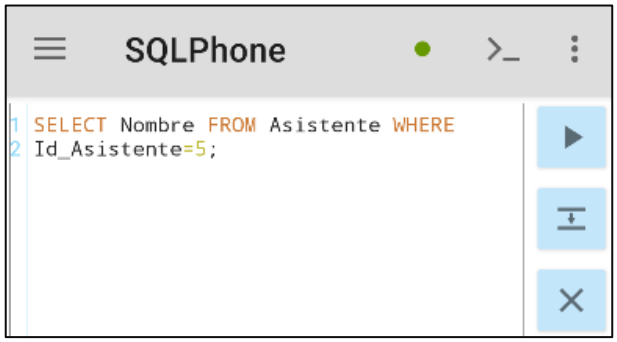
Sintaxis para insertar datos en una tabla.

UPDATE nombre_tabla **SET** columna1 = nuevo valor1, columna2 = nuevo "'valor2 **WHERE** condición;

Primero debemos detectar que registro necesita ser actualizado, esto lo podemos hacer mediante una consulta con la sentencia **SELECT**.

Una vez detectada y respetando la sintaxis de la sentencia **UPDATE** modificaremos el registro, ya sea en todos los campos o solo uno, como será nuestro caso, que actualizaremos el campo nombre del registro 5, en este caso usaremos **WHERE** para poner una condición de selección.

Ahora ejecutaremos una consulta para ver la actualización.



Resultando

6. **Eliminar datos en la tabla**

Sintaxis para insertar datos en una tabla.

DELETE FROM nombre_tabla **WHERE** condición;

Primero debemos detectar que registro necesita ser actualizado, esto lo podemos hacer mediante una consulta con la sentencia **SELECT**.

id_asistente	nombre	cor
1	juan lopez diaz	juar
2	martha ramirez ruíz	mart
3	alma luisa gonzález luis	alma
4	rodrigo sansebastián ríos	rsar
5	francisca juárez fernández	frar

Una vez detectada y respetando la sintaxis de la sentencia **DELETE** y usando **WHERE** con una condición, debemos tener cuidado, ya que si no usamos **WHERE** eliminaremos todos los registros.

Ahora ejecutaremos una consulta para ver los cambios en nuestra tabla.

Resultando

id_asistente	nombre	corre
1	juan lopez diaz	juan
2	martha ramirez ruíz	marth
3	alma luisa gonzález luis	alma
4	rodrigo sansebastián ríos	rsans

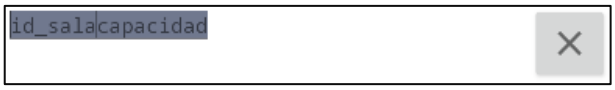
Ahora crearemos las demás tablas de otra manera, es decir, copiando el código generado y exportado en Database Designer y pegándolo en el editor de SQL Phone.

1. **Ubica el documento de texto generado en Database Designer y ábrelo.**

2. **Copia la sección de código SQL para la tabla Salas.**

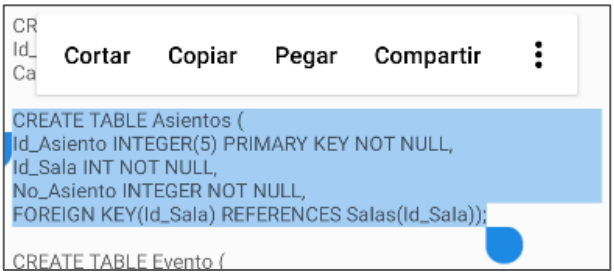
3. **Pégalo en el editor de SQL Phone y presiona ejecutar**

4. **Usa la sentencia SELECT para ver el esquema de la tabla, aunque como no tiene registros, aparecerá limpia.**

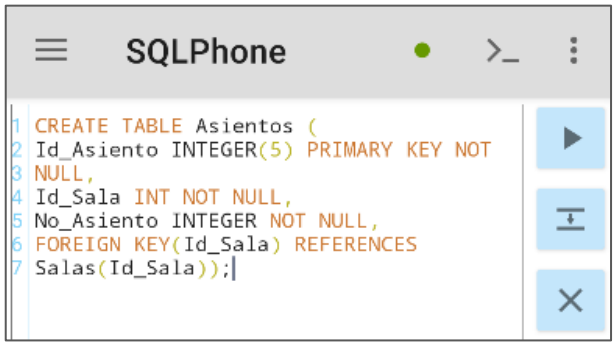


Ahora veremos el mismo caso, pero con una tabla que tiene una llave secundaria (Id_Sala) relacionada hacia otra tabla (Sala)

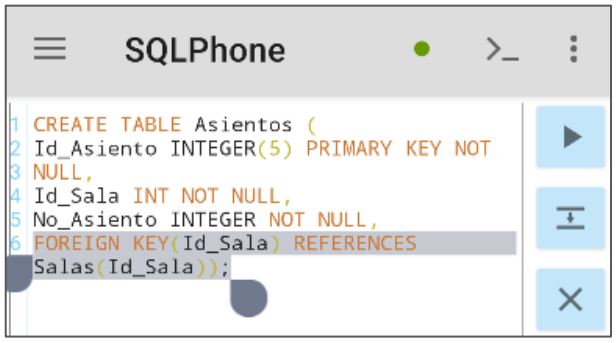
1. **Copia la sección de código SQL para la tabla Asientos.**



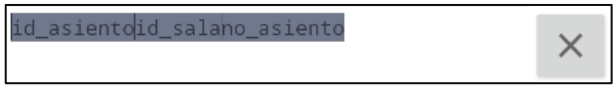
2. **Pégalo en el editor de SQL Phone y presiona ejecutar**



La línea FOREIGN KEY(Id_Sala) REFERENCES Salas(Id_Sala)); define la relación señalando el atributo como una llave secundaria que va relacionada hacia otra tabla.

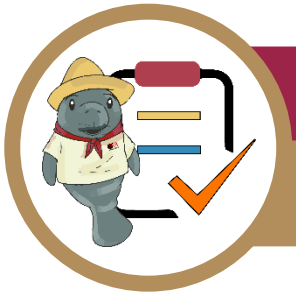


3. **Usa la sentencia SELECT para ver el esquema de la tabla, aunque como no tiene registros, aparecerá limpia.**



Práctica 6.2. Que SQL el código

Instrucciones: *El estudiante en equipo completará la creación de la BD haciendo uso de la App SQL Phone ejecutando las sentencias SQL generadas en Database Designer.*



INSTRUMENTO DE EVALUACIÓN

Guía de observación

Práctica 6 – Que SQL el código

DATOS GENERALES	
Nombre(s) del alumno (s)	**Matriculas(s)**
Producto: BD generada en SQL Phone	**Fecha:**
Materia: Sistemas de información	**Periodo:2025-2026A**
Nombre del docente:	**Firma del docente**

Etapa / Criterio	Insuficiente	Suficiente	Bueno	Excelente
1. Configuración y acceso a la aplicación SQL Phone	Presenta dificultades para acceder o utilizar la aplicación y requiere ayuda constante.	Accede a la aplicación y realiza las acciones básicas con apoyo ocasional.	Utiliza la aplicación de forma adecuada y casi sin ayuda.	Maneja la aplicación con seguridad, autonomía y aprovecha correctamente sus herramientas.
20%	5%	10%	15%	20%
2. Creación de la base de datos y tablas	No logra crear la base de datos o las tablas requeridas.	**Crea parcialmente la base de datos o algunas tablas con errores.**	**Crea correctamente la mayoría de las tablas requeridas.**	**Crea correctamente la base de datos y todas las tablas solicitadas sin errores.**
25%	6%	12%	18%	25%
3. Definición de claves primarias y foráneas	**No identifica ni implementa las claves requeridas.**	**Implementa algunas claves, pero presenta errores de relación.**	Implementa correctamente la mayoría de las claves y relaciones.	Implementa correctamente todas las claves primarias y foráneas, garantizando la integridad de la base de datos.
20%	5%	10%	15%	20%
4. Uso de sentencias SQL (CREATE, INSERT,	Presenta errores frecuentes y no logra ejecutar	**Ejecuta algunas sentencias con apoyo y corrige**	**Ejecuta correctamente la mayoría de las**	**Ejecuta todas las sentencias de manera correcta,**

UPDATE, DELETE, SELECT)	correctamente las sentencias.	**errores básicos.**	**sentencias requeridas.**	**demostrando dominio de la sintaxis SQL.**
25%	6%	12%	18%	25%
5. Verificación y organización del trabajo realizado	No verifica resultados ni identifica errores en la base de datos.	Verifica parcialmente los resultados y corrige algunos errores.	Verifica los resultados y corrige adecuadamente los errores encontrados.	Verifica sistemáticamente la información, corrige errores y mantiene un trabajo ordenado y documentado.
10%	2.5%	5%	7.5%	10%
Total				

Referencias

Araujo-Inastrilla, C. R., Roche-Madrigal, M. d., & García-Savón, Y. (2023). Diseño de base de datos para el departamento de Sistemas de Información en Salud, La Habana 2021. Revista Información Científica, 102.

Casas Roma, J. (2019). Introducción a la base de datos. Universitat Oberta de Catalunya.

Costal Costa, D. (2002). Introducción al diseño de base de datos. Universidad virtual.

Erickson, J. (29 de agosto de 2024). MySQL: qué es y cómo se usa. https://www.oracle.com/: https://www.oracle.com/mx/mysql/what-is-mysql/

León Romero, B. (2021). El libro práctico de Base de datos. (B. L. Rmero, Ed.) España: Blas León Rmero. https://libropracticodebasesdedatos.com/descargas/primeraspaginas.pdf

Microsoft. (2010). Tareas básicas en Access 2010. https://support.microsoft.com/: https://support.microsoft.com/es-es/office/tareas-b%C3%A1sicas-en-access-2010-268acfed-2484-4822-acb3-c30e58045588#:~:text=Access%20es%20una%20herramienta%20que,ayudan%20a%20administrar%20la%20informaci%C3%B3n.

Submódulo 1. Situación didáctica "Haz que tus ideas cuenten"

INSTRUCCIONES

Organizados en **equipos colaborativos de 5 integrantes**, diseñen e implementen una base de datos sencilla que dé solución a la situación didáctica planteada.

Como evidencia del proyecto, elaboren las **sentencias SQL** necesarias para crear y administrar la base de datos del sistema, considerando al menos la creación de tablas, relaciones, restricciones e inserción de datos de prueba que permitan comprobar su funcionamiento.

PROPÓSITO DEL PROYECTO

Desarrollar un sistema de información que permita la **gestión de reservas de espacios en eventos escolares**, mediante el análisis de necesidades, el diseño de soluciones y la implementación básica con herramientas de programación, favoreciendo el trabajo colaborativo y la resolución de problemas del contexto escolar.

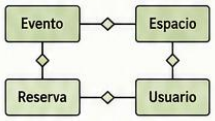
DESARROLLEN EL PROYECTO RESPETANDO LAS FASES DEL MODELO EN CASCADA

1 ANÁLISIS DE REQUERIMIENTOS

- Identifiquen las necesidades del sistema.
- Determinen los usuarios, funciones y reglas del negocio.
- Establezcan los requerimientos funcionales y no funcionales.

Producto de la fase: Documento de análisis de requerimientos.

2 DISEÑO DEL SISTEMA Y DE LA BASE DE DATOS



- Diseñen el modelo de datos (Entidad–Relación o modelo relacional).
- Definan las tablas, atributos, claves primarias, foráneas y restricciones.
- Elaboren el diccionario de datos.

Producto de la fase: Diagrama de la base de datos y diccionario de datos.

3 IMPLEMENTACIÓN MEDIANTE SENTENCIAS SQL

- Creen la base de datos y sus objetos con sentencias SQL.
- Implementen las tablas con sus relaciones y restricciones.
- Inserten datos de prueba.

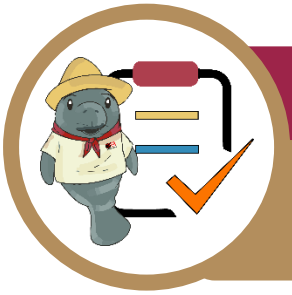
Producto de la fase: Script SQL con sentencias (CREATE, INSERT, UPDATE, DELETE y SELECT).

PRODUCTO ESPERADO DEL PROYECTO

- Análisis de requerimientos del sistema.
- Diagrama de la base de datos (Entidad–Relación o modelo relacional).
- Diccionario de datos.
- Sentencias SQL (CREATE, INSERT, UPDATE, DELETE y SELECT) que demuestren el funcionamiento.
- Evidencias de pruebas realizadas y conclusiones del equipo.

La comunicación, la colaboración y la organización son clave para el éxito de su proyecto.
¡ÉXITO, EQUIPO!

Trabajen con responsabilidad, respeto y compromiso para construir soluciones que mejoren nuestro entorno escolar.



RÚBRICA

Situación Didáctica. "Haz que tus ideas cuenten"

DATOS GENERALES	
Nombre(s) del alumno (s)	**Matriculas(s)**
Producto: Base de Datos	**Fecha**
UAC: Sistemas de información	**Periodo:2025-2026A**
Nombre del docente:	**Firma del docente**

Etapa / Criterio	Insuficiente	Suficiente	Bueno	Excelente
1. Recopilación y análisis de datos (¿Qué información necesita registrar la cafetería? ¿Qué procesos debe controlar?)	• No se identifican los datos necesarios (productos, precios, ventas, etc.). • No hay relación entre los datos propuestos. • No se entiende cómo se usará la información.	• Se identifican algunos datos importantes, pero de forma desorganizada. • Hay confusión sobre los procesos a controlar. • Faltan elementos clave como clientes, tipo de producto, fechas, etc.	• Se identifican bien los datos principales (productos, precios, ventas, cantidades, fechas, etc.). • Se explican de forma clara, aunque con algunos detalles menores faltantes.	• Todos los datos importantes están correctamente definidos y organizados. • Se explica qué procesos se deben controlar (ventas, productos, stock) y por qué. • Se presentan ejemplos reales o supuestos claros.
20%	5%	10%	15%	20%
2. Diseño conceptual (Diagrama Entidad-Relación - DER) (¿Qué	• El diagrama está incomplet o o tiene errores	• Las entidades están presentes, pero con errores en	• El DER incluye entidades y relaciones bien definidas,	• El DER está completo, con todas las entidades necesarias

entidades existen y cómo se relacionan entre sí?)	graves (entidades mal definidas, relaciones que no se entienden). • Faltan las llaves primarias o hay relaciones incorrectas.	los nombres, relaciones o cardinalidades. • Faltan llaves primarias o relaciones necesarias.	con nombres claros. • Se utilizan correctamente llaves primarias y la mayoría de las cardinalidades son correctas. • El diseño es entendible.	(como Producto, Venta, Cliente, Categoría). • Las relaciones están bien definidas, las cardinalidades son correctas. • El diseño está limpio, ordenado y bien presentado visualmente.
25%	6%	12%	18%	25%
3. Diseño lógico (Esquema relacional) (¿Cómo se convierten las entidades y relaciones en tablas?)	• Las tablas no coinciden con el DER o hay errores en nombres, campos y relaciones. • No se definen correctamente las llaves primarias o foráneas. • No se puede entender la estructura.	• Las tablas reflejan el DER pero con errores (nombres mal puestos, campos faltantes, relaciones incompletas). • Llaves mal asignadas o mal uso de tipos de datos.	• Las tablas están bien organizadas y reflejan correctamente el DER. • Llaves primarias y foráneas bien definidas. • Tipos de datos adecuados en la mayoría de los campos.	• Las tablas reflejan fielmente el DER y siguen buenas prácticas de normalización. • Se usan nombres descriptivos, tipos de datos precisos, llaves primarias y foráneas correctas. • Se documenta claramente cada tabla.
20%	5%	10%	15%	20%
4. Diseño físico (Código SQL para SQLite)	• El código está incompleto o	• El código funciona parcialmente,	• El código crea correctamente	• El código SQL está bien estructurado,

(¿El código SQL crea correctamente las tablas con sus relaciones?)	tiene errores que impiden ejecutarlo. • Faltan muchas tablas o no se definen bien los campos ni las relaciones.	pero tiene errores de sintaxis o campos mal definidos. • Algunas relaciones están incompletas. • Falta indentación o está desorganizado.	las tablas, llaves primarias y foráneas. • Puede tener pequeños errores, pero funciona en general. • Está organizado de forma clara.	completo y se puede ejecutar sin errores. • Incluye todas las tablas, campos, relaciones y restricciones necesarias. • Se usa buena indentación, nombres descriptivos y comentarios si es necesario.
25%	6%	12%	18%	25%
5. Trabajo colaborativo y presentación del proyecto (¿Todos participaron y presentaron de forma clara el trabajo?)	• Solo una o dos personas trabajaron. • No hay evidencia de colaboración. • La presentación fue desorganizada, sin claridad o incompleta.	• Participaron algunos integrantes, pero no de forma equilibrada. • La presentación fue rápida o poco clara. • Faltaron partes por explicar.	• Todos los integrantes participaron. • La presentación fue clara y estructurada. • Se explicó el proceso completo, aunque sin mucha profundidad.	• Todos participaron activamente. • La presentación fue clara, completa y con seguridad. • Usaron ejemplos, explicaron decisiones técnicas y demostraron dominio del tema.
10%	2.5%	5%	7.5%	10%
Total				

Referencias

Casas Roma, J. (2019). Introducción a la base de datos. Universitat Oberta de Catalunya.

Costal Costa, D. (2002). *Introducción al diseño de base de datos.* Universidad virtual.

Erickson, J. (29 de agosto de 2024). *MySQL: qué es y cómo se usa*. https://www.oracle.com/: https://www.oracle.com/mx/mysql/what-is-mysql/

León Romero, B. (2021). *El libro práctico de Base de datos.* (B. L. Rmero, Ed.) España: Blas León Rmero. https://libropracticodebasesdedatos.com/descargas/primeraspaginas.pdf

Microsoft. (2010). *Tareas básicas en Access 2010*. https://support.microsoft.com/: https://support.microsoft.com/es-es/office/tareas-b%C3%A1sicas-en-access-2010-268acfed-2484-4822-acb3-c30e58045588#:~:text=Access%20es%20una%20herramienta%20que,ayudan%20a%20administrar%20la%20informaci%C3%B3n.

SUBMÓDULO II

Programación

Algoritmos | Código | Lógica | Desarrollo | Soluciones

Propósito del Módulo

Plantea soluciones críticas y responsables mediante la metodología de desarrollo de software para demostrar eficiencia en el manejo de base de datos y software de programación de alto nivel, que sean aplicables a necesidades de una empresa o institución para el tratamiento de información.

Sugerencia de evidencias de Evaluación

En equipos de cinco integrantes, el estudiantado aplicará la metodología de cascada para diseñar un Sistema de Información para Reservas de Eventos, el cual permitirá a los asistentes:

- Registrar sus datos personales.
- Elegir asientos específicos.
- Seleccionar diferentes horarios y salas.
- Realizar pagos en distintos formatos.

Roles dentro del equipo:

Cada integrante asumirá un rol específico en el desarrollo del sistema:

- Programador de la Base de Datos: Diseña y desarrolla la base de datos utilizando MySQL.
- Desarrollador de la Interfaz Gráfica: Crea la interfaz utilizando Python o Visual Studio.
- Coordinador del Proyecto: Supervisa la integración de la base de datos y la interfaz, asegurando el cumplimiento de la metodología.

Competencia Laboral Básica Para Desarrollar

Identifica sistemas de información organizacionales mediante la codificación y compilación de instrucciones algorítmicas pertinentes utilizando lenguajes de programación y bases de datos para cumplir con los requerimientos de funcionalidad y rendimiento establecidos en el diseño de sistemas de información asumiendo la frustración como parte del proceso en ambientes laborales, educativos y de la vida cotidiana

SEPT-ENE 26-27A

Submódulo 2. Dosificación programática Programación

Formación Laboral Básica: Tecnologias de la Información y la comunicación
Módulo II: Desarrollo de sistemas
Submódulo II: Programación Clave: C5PO
Semestre: 5to. **Turno:** Matutino/Vespertino **Periodo:** 2026 – 2027A

Submódulo	Momento	Tiempo (minutos)	Conocimientos	Semana	Fecha inicio	Observaciones
SUBMÓDULO II. PROGRAMACIÓN	Apertura	50 min 50 min 250 min	Bienvenida Encuadre Reglamento Instrumentos de evaluación. Evaluación diagnóstica. Lectura 1. Estructuras de datos y algoritmos	**8**	19 – 23 de octubre de 2026	Evaluación extraordinaria intrasemestral Reunión de discusión del Marco Curricular Común de la Educación Media Superior y Evaluación de los aprendizajes alcanzados
	Desarrollo	100 min 250 min	Lectura 2. Lenguajes de programación de alto nivel	**9**	26 – 30 de octubre de 2026	Reunión de academias
		350 min	Lectura 3. Python • **Estructuras básicas, variables, contantes** • **Operadores**	**10**	02 – 06 de noviembre de 2026	2 de noviembre Festivo
		350 min	Lectura 4. Funciones	**11**	09 – 13 de noviembre de 2026	

Submódulo	Momento	Tiempo (minutos)	Conocimientos	Semana	Fecha inicio	Observaciones
SUBMÓDULO II. PROGRAMACIÓN	Desarrollo	250 min 100 min	Lectura 5. Todo se ve mejor en ventanas • **Tkinter**	**12**	16 – 20 de noviembre de 2026	17 de noviembre suspensión de labores
		100 min 250 min	Lectura 5. Todo se ve mejor en ventanas • **Tkinter**	**13**	23 – 27 de noviembre de 2026	
		350 min	Lectura 6. Python y las bases de datos. • **Creación de Base de datos con SQlite** • **Consultas**	**14**	30 de noviembre - 04 de diciembre de 2026	
		350 min	Lectura 6. Python y las bases de datos. • **Creación de Base de datos con SQlite** • **Consultas**	**15**	07 - 12 de diciembre de 2026	
	Cierre	350 min	Situación didáctica **12. "Haz que tu código cuente**	**16**	14 - 18 de diciembre de 2026	
				17	06 - 08 de enero de 2027	Reforzamiento académico Reunión académica

Submódulo	Momento	Tiempo (minutos)	Conocimientos	Semana	Fecha inicio	Observaciones
SUBMÓDULO II. PROGRAMACIÓN				**18**	11 - 15 de enero de 2027	Reforzamiento académico.
				19	18 - 22 de enero de 2027	Evaluación Extraordinaria Intersemestral.
				20	25 - 29 de enero de 2027	Jornada de capacitación
				21 y 22		Fin de periodo Receso escolar
	Total: Tiempo en minutos sumando los dos submódulos		5,600 min			

Submódulo 2. Encuadre Programación

Formación Laboral Básica: Tecnologías de la información y la comunicación
Módulo III: Desarrollo de Sistemas
Submódulo I: Sistemas de Información **Clave:** C5SI

No.	Actividades	Ponderación
1	Actividad 1. Relacionan los conceptos de estructuras.	5%
2	Actividad 2. ¡Falso o Verdadero!	5%
3	Actividad 3. ¿Qué tipo de variables tengo?	5%
4	Ejercicio teórico. Mapa conceptual uso de Tkinter	5%
	TOTAL	20%

No.	Practicas	Ponderación
1	Práctica 1. Funciones y parámetros	15%
2	Práctica 3. Calculadora básica	15%
	TOTAL	30%

No.	Situación didáctica	Ponderación
1	Situación didáctica. ""Haz que tu código cuente"	30%

No.	Evaluación escrita	Ponderación
1	Examen	20%

Submódulo 2. Situación didáctica "Haz que tu código cuente"

Propósito de la Situación didáctica:	Desarrollar un sistema de información que permita la gestión de reservas de espacios en eventos escolares, mediante el análisis de necesidades, el diseño de soluciones y la implementación básica con herramientas de programación, favoreciendo el trabajo colaborativo y la resolución de problemas del contexto.
Sugerencia de Evidencia de Evaluación:	En equipos de cinco integrantes, el estudiantado aplicará la metodología de cascada para diseñar un Sistema de Información para Reservas de Eventos, el cual permitirá a los asistentes: ●Registrar sus datos personales. ●Elegir asientos específicos. ●Seleccionar diferentes horarios y salas. ●Realizar pagos en distintos formatos. Roles dentro del equipo: Cada integrante asumirá un rol específico en el desarrollo del sistema: ●Programador de la Base de Datos: Diseña y desarrolla la base de datos utilizando MySQL. ●Desarrollador de la Interfaz Gráfica: Crea la interfaz utilizando Python o Visual Studio. ●Coordinador del Proyecto: Supervisa la integración de la base de datos y la interfaz, asegurando el cumplimiento de la metodología.
Problema de Contexto:	El estudiantado de quinto semestre, en coordinación con sus docentes, se encuentra organizando diversos eventos en el auditorio de su plantel. Ante esta situación, surge la necesidad de contar con un sistema que permita a las y los participantes seleccionar los espacios disponibles dentro de la sala, de manera similar a los sistemas utilizados en cines y eventos con control de acceso. Actualmente, la asignación de lugares se realiza de manera manual, lo que genera desorganización, duplicidad de espacios y dificultad para el control de asistentes. Por ello, el grupo solicita el apoyo del docente de Tecnologías de la Información y la Comunicación para adquirir los conocimientos necesarios que les permitan diseñar y desarrollar un sistema de información que facilite la gestión, organización y control de eventos masivos.
Conflicto Cognitivo:	¿Cómo se puede diseñar un sistema que permita a los usuarios seleccionar su lugar de manera organizada y eficiente? ¿Qué elementos son necesarios para construir un sistema de información funcional? ¿Cómo se puede almacenar y consultar la información de los asistentes? ¿Qué herramientas tecnológicas permiten desarrollar este tipo de soluciones?

Submódulo 2. Evaluación diagnóstica

Instrucciones: *Lea las siguientes preguntas y subraye la respuesta correcta.*

NOMBRE		GRUPO:	

1. ¿En cuál opción se describe un algoritmo?

a. Una lista de instrucciones sin orden específico.
b. Un conjunto de pasos ordenados que resuelven un problema.
c. Un conjunto de datos sin procesar

2. ¿Qué inciso describe, de manera correcta, las formas de representación de los algoritmos?

a. Texto narrativo, diagramas de flujo y código de máquina.
b. Diagramas de flujo, ecuaciones y tablas de verdad.
c. Diagramas de flujo, pseudocódigo y descripción, narrativa.

3. ¿Se utilizan para tomar decisiones lógicas; de ahí que se suelan denominar también estructuras de decisión o alternativas?

a. Estructura doble
b. Estructura selectiva
c. Estructura selectiva múltiple

4. ¿Permite comparar una variable con múltiples valores definidos y ejecutar una serie de instrucciones en cada caso?

a. Estructura selectiva múltiple
b. Estructura secuencial
c. Estructura doble

5. Es la estructura secuencial que se da naturalmente en el lenguaje.

a. Control
b. Secuencia
c. Bucle

6. Se basa en dividir el código en partes más pequeñas y seguir una estructura clara.

a. Objetos
b. Funciones
c. Estructurado

7. ¿Qué nivel de programación permite codificar las instrucciones de un computador utilizando una sintaxis y una semántica similar al lenguaje natural o lenguaje humano?

a. Programación a bajo nivel
b. Programación a alto nivel
c. Ensamblador

8. Un programa realizado en lenguaje de alto nivel su código se traduce al código máquina, mediante:

a. Signos y letras
b. 0 y 1
c. Compiladores e intérpretes

9. Es un software que nos permite la construcción o elaboración de otros programas informáticos. Está formado por un conjunto de símbolos y reglas y palabras especiales que utilizamos para comunicarnos con las computadoras y otros dispositivos electrónicos

a. Algoritmo
b. Lenguaje Maquina
c. Lenguaje de Programación

10. Son como contenedores donde se pueden almacenar y manipular datos, como números, texto o valores booleanos (verdadero/falso). Tiene un nombre que la identifica.

a. Variables
b. Estructura de datos
c. Operadores

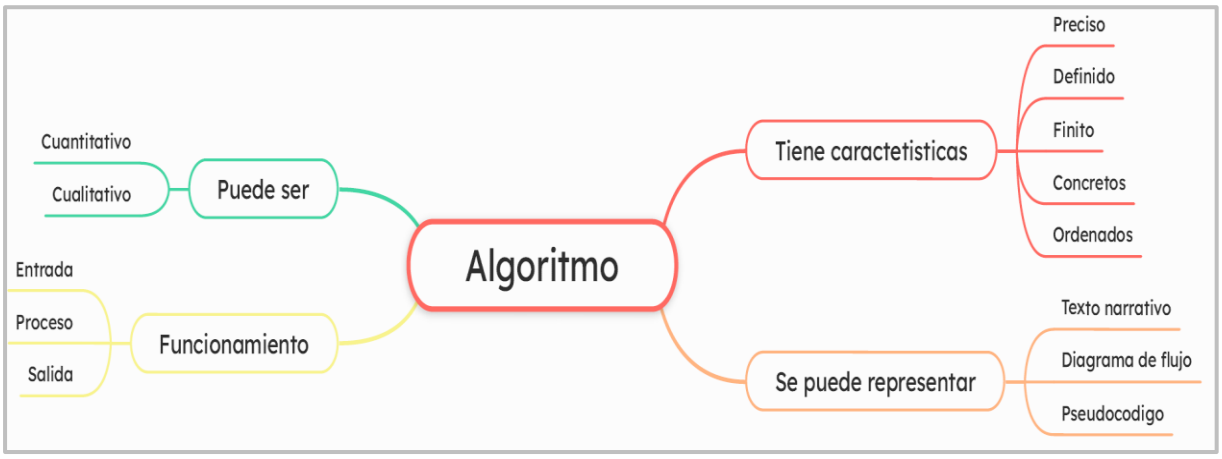
Lectura 1. Estructuras de datos y algoritmos

Instrucciones: *Realiza el análisis de la siguiente información, posteriormente, completa la* ***Actividad 2. Relacionan los conceptos de estructuras y resuelve el ejercicio planteado al final de la lectura.***

Algoritmos

Un algoritmo se puede definir como una secuencia de instrucciones o pasos ordenados que llevan a la solución de un problema. Elizondo y Rubio Sosa (2023, p. 111) mencionan que "Un algoritmo es una especie de receta de instrucciones o rutinas para resolver un problema. Los algoritmos son estrategias para la solución de problemas no solo aplicables en actividades académicas o profesionales, también a todo tipo de problemas relacionados con actividades cotidianas".

Así que los algoritmos no solo te ayudan a programar, sino a resolver situaciones cotidianas de una manera más eficiente y exitosa. Por esta razón, resulta fundamental, comprender los elementos conceptuales que lo conforman. En el siguiente esquema se pueden apreciar su clasificación, características y formas de representación.

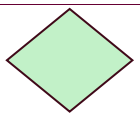
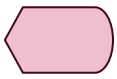


En las tablas se describen de manera detallada cada uno de sus elementos.

Funcionamiento	Tipos	Características Maluenda (2025)	Formas de Representación
• **Entrada:** Son los datos necesarios o datos de entrada para resolver el problema.	• **Cuantitativo.** Es aquel algoritmo que resuelve problemas que requieren algún calculo.	• **Precisos.** Son objetivos, sin ambigüedades.	• **Texto narrativo.** Se usa el lenguaje natural para indicar las acciones a realizar.
Proceso: Es el procedimiento realizado para solucionar el problema. • **Salida:** Refiere al resultado obtenido y/o esperado.	• **Cualitativo.** Son los algoritmos que no solicitan la realización de cálculos para llevarse a cabo.	• **Definidos.** El mismo algoritmo debe dar el mismo resultado al recibir la misma entrada. • **Finitos.** Contienen un número determinado de pasos. • **Ordenados.** Presentan una secuencia clara y precisa para poder llegar a la solución. • **Concretos.** Ofrecen una solución determinada para la situación o problema planteados.	• **Diagrama de flujo.** Es la representación gráfica de un algoritmo. • **Pseudocódigo.** Es el uso de un lenguaje no formal, para describir la secuencia de acciones a realizar.

Símbolos usados en los diagramas de flujo

Los Diagramas de flujo son una forma de representar gráficamente un algoritmo, usando, estrictamente, los símbolos estandarizados, por el Instituto Nacional Americano de Estándares (ANSI), que se enlistan a continuación (*Significado de los 23 Símbolos de Diagrama de Flujo de Procesos*, s. f.):

Símbolo	Nombre	Descripción
	Inicio/Fin	Indica el comienzo y el final del diagrama de flujo.
	Entrada/Salida	Representa el ingreso de datos y también la salida de un resultado.
	Proceso	Representa la realización de una actividad, operación o proceso.
	Decisión	Indica que en la resolución del problema debe tomarse una decisión y que el flujo del diagrama continua de acuerdo con esta.
	Conector dentro de la página	Enlaza el diagrama en otro espacio dentro de la misma hoja.
	Conector fuera de la página	Enlaza el diagrama en otro espacio en otra página.
	Líneas de flujo	Son las líneas de flujo del diagrama, solo deben ir de arriba-abajo y de izquierda-derecha.
	Documento	Envía el resultado a documento.
	Pantalla	Muestra un resultado en pantalla.
	Y	Se usa para indicar la Y lógica.
	O	Se usa para indicar la O lógica.

Los símbolos anotados aquí no son exhaustivos, por lo cual puedes consultar en el siguiente LINK o QR para más información.

Recurso didáctico sugerido	
EL SIGNIFICADO DE 23 SÍMBOLOS DE DIAGRAMA DE FLUJO https://www.heflo.com/es/blog/significado-simbolos-diagrama-flujo	

Un programa propio puede ser escrito utilizando solamente tres tipos de estructuras de control:

Secuencial:	En el mismo orden en forma imperativa, no depende del contenido.
Repetición:	Llevar a cabo la ejecución de un conjunto de funciones una determinada cantidad de veces.
Decisión:	Mediante estructura lógica >> saber si se realiza una función o no.

Un programa se define como propio si cumple las siguientes características:

- Posee un solo punto de entrada y uno de salida, o fin, para control del programa;
- Existen caminos desde la entrada hasta la salida que se pueden recorrer y que pasan por todas las partes del programa.
- Todas las instrucciones son ejecutables y no existen lazos o bucles infinitos.

Estructura secuencial

Es aquella en la que una acción sigue a otra en secuencia. Las tareas se suceden de tal modo que la salida de una es la entrada de la siguiente y así sucesivamente hasta el final del proceso. La estructura secuencial tiene una entrada y una salida. Su representación gráfica se muestra en la siguiente figura.

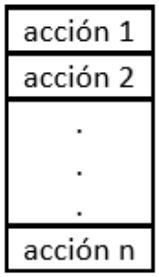


Figura 2. Diagrama N-S de una estructura secuencial

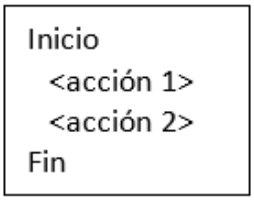


Figura 3. Pseudocódigo de Una estructura secuencial

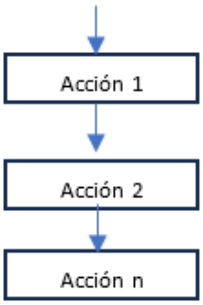
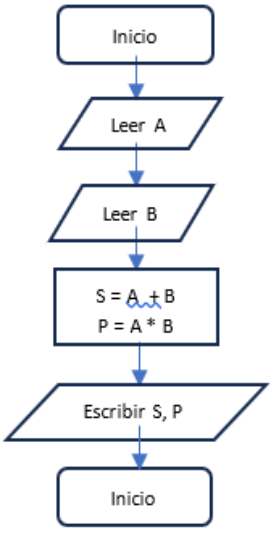


Figura 1. Algoritmo de Estructura secuencial

Ejemplo 1
Calcular la suma y producto de dos números.

Diagrama de flujo



Pseudocódigo

Inicio
Leer (A)
Leer (B)
S = A + B
P = A * B
escribir (S, P)
Fin

Ejemplo 2.
Calcular el salario neto de un trabajador en función de horas trabajadas, precio de la hora de trabajo y, considerando unos descuentos fijos, el suelo bruto en concepto de impuesto (20 por 100).

Diagrama de flujo

Pseudocódigo

```
Inicio
  //Calculo salario neto
  leer (nombre, horas, precio_hora)
  salario_bruto = horas * precio_hora
  impuestos = 0.20 * salario_bruto
  salario_neto = salario_bruto – impuestos
  escribir (nombre, salario_bruto, salario_neto)
fin
```

Estructura selectiva

La estructura lógica selectivas se encuentran en la solución algorítmica de casi todo tipo de problemas. Se utilizan para tomar decisiones lógicas; de ahí que se suelan denominar también *estructuras de decisión o alternativas.*

Hay situaciones en las que la toma de decisiones se realiza en cascada. Para alcanzar la solución de un problema o subproblema debemos aplicar prácticamente un árbol de decisión. La representación de una estructura selectiva se hace con palabras en pseudocódigo (if, then, else o bien en español si, entonces, si_no), con una figura geométrica en forma de rombo. Las estructuras selectivas o alternativas pueden ser:

Si entonces	(if_then)	(Estructura selectiva simple)
Si entonces/sino	(else)	(Estructura selectiva doble)
Si multiple	(switch)	(Estructura selectiva multiple)

Estructura selectiva simple (SI – ENTONCES / IF – THEN)

Permite que el flujo del diagrama siga por un camino especifico si se cumple una condición o conjunto de condiciones. Si al evaluar la condición el resultado es verdadero, entonces se ejecuta cierta operación.

Pseudocódigo

si <condición>
entonces
<acción SI>
fin_si

Diagrama de flujo

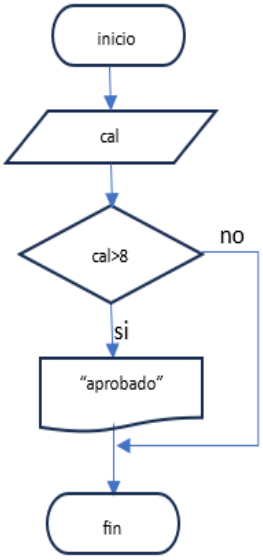
Donde:
Condición expresa la condición o conjunto de condición a evaluar.
Operación expresa la operación o conjunto de operaciones que se van a realizar si la condición resulta verdadera.

Ejemplo 1

Construye un diagrama de flujo, dada la calificación de un alumno en un examen, escriba "aprobado" en caso de que esa calificación sea mayor a 8.

Dato: cal (variable de tipo real que representa la calificación del alumno)

Diagrama de flujo



Pseudocódigo

{cal es una variable de tipo real}

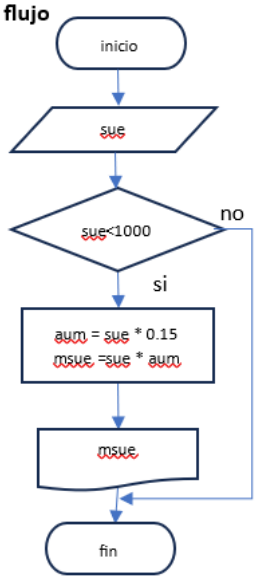
1. Leer cal
2. Si cal>8 entonces
   Escribir "Aprobado"
3. {fin del condicional del paso 2}

Ejemplo 2

Dado como dato el sueldo de un trabajador, aplíquele un aumento del 15% si su sueldo es inferior a $ 1000. Imprima en este caso el nuevo sueldo del trabajador.

Dato: sue (variable de tipo real que representa el sueldo del trabajador)

Diagrama de flujo



Pseudocódigo

{sum, aum y mseu son variables de tipo real}

1 Leer sue
2 Si sue < 1000 entonces
  Hacer aum = sue * 0.15 y
    msue = sue + aum
  Escribir msue

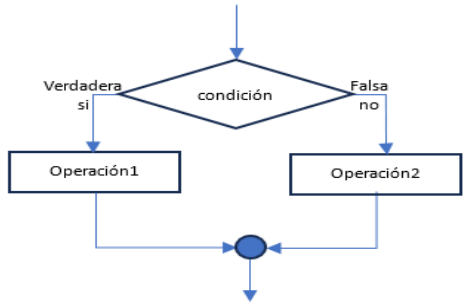
3. {fin del condicional del paso 2}

Estructura selectiva doble (SI – ENTONCES – SINO / IF – THEN – ELSE)

Permite que el flujo del diagrama se bifurque por las 2 ramas diferentes en el punto de la toma de decisión. Si al evaluar la condición el resultado es verdadero, entonces se sigue por un camino especifico y se ejecuta cierta operación. Por otra parte, si el resultado es falso entonces se sigue por otro camino y se

ejecuta otra operación. En ambos casos luego de ejecutarse la operación indicada, se continua con la secuencia normal del diagrama.

Diagrama de flujo



Donde:

Condición expresa la condición o conjunto de condiciones a evaluarse

operación1 expresa la operación o conjunto de operaciones que se van a realizar si la condición resulta verdadera

operacion2 expresa la operación o conjunto de operaciones que se van a realizar si la condición resulta falsa

Pseudocódigo

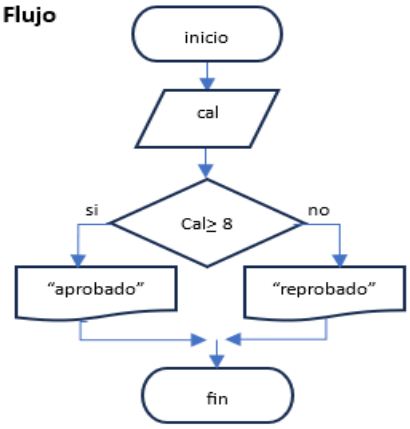
```
Si condición
  entonces
    hacer operación1
 sino
    hacer operacion2
{fin del condicional}
```

Ejemplo

Construye un diagrama de flujo tal que, dada la calificación de un alumno en un examen, escriba "aprobado" si su calificación es mayor o igual que 8 y "reprobado" en caso contrario.

Dato: cal (variable de tipo real que expresa la calificación del alumno)

Diagrama de Flujo



Pseudocódigo

1. Leer cal
2. Si cal ≥ 8
   Entonces
   Escribir "aprobado"
   Sino
   Escribir "reprobado"
3. {Fin del condicional del paso 2}

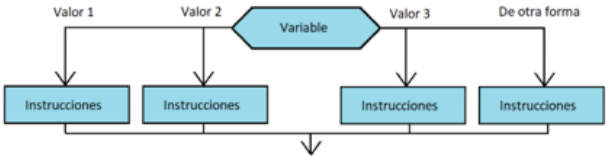
Estructura selectiva múltiple.

Una estructura selectiva múltiple permite comparar una variable con múltiples valores definidos y ejecutar una serie de instrucciones en cada caso

Pseudocódigo

Según variable Hacer
  Opción_1:
    Instrucciones
  Opción_2:
    Instrucciones
  Opción_3:
    Instrucciones
De otro modo
    Instrucciones
Fin según

Diagrama de flujo



Ejemplo

Realizar un pseudocódigo y un diagrama de flujo que dependiendo de la elección del usuario realicen la suma, resta, multiplicación o división de dos números.

Pseudocódigo

```
Algoritmo calculadora
    Escribir "Selecciona una operación"
    Escribir "Suma=1 Resta=2 Multiplicación=3 División=4"
    Leer opc
    Escribir "Ingresa dos números"
    Leer num1,num2
    Segun opc Hacer
        1:
            res<-num1 + num2
        2:
            res<-num1 - num2
        3:
            res<-num1 * num2
        4:
            res<-num1 / num2
        De Otro Modo:
            Escribir "La operación no es válida"
    Fin Segun
    Escribir "El resultado es: " res
FinAlgoritmo
```

Actividad 1. Relacionan los conceptos de estructuras.

Instrucciones: *Relaciona las columnas de forma correcta según corresponda los conceptos al finalizar diseña el algoritmo y diagrama de flujo del ejercicio.*

Las tareas se suceden de tal modo que la salida de una es la entrada de la siguiente y así sucesivamente hasta el final del proceso	**Estructura selectiva**
Si al evaluar la condición el resultado es verdadero, entonces se ejecuta cierta operación.	**Estructura selectiva doble**
Se encuentran en la solución algorítmica de casi todo tipo de problemas	**Estructura Secuencial**
Permite que el flujo del diagrama se bifurque por las 2 ramas diferentes en el punto de la toma de decisión	**Estructura selectiva simple**
Permite comparar una variable con múltiples valores definidos y ejecutar una serie de instrucciones en cada caso	**Estructura selectiva múltiple**

Ejercicio 1.

Se quiere diseñar el algoritmo de un programa que:

1. Pida por teclado la calificación (real) de una asignatura.
2. Muestre la salida según sea el caso:
    a. "Aprobado", en el caso de que la nota sea mayor o igual que 5 y menor o igual que 10.
    b. "No aprobado", en el caso de que la nota sea mayor o igual que 0 y menor que 5.
    c. "ERROR: Nota incorrecta.", en el caso de que la nota sea menor que 0 o mayor que 10.

Referencias

Aguilar, L. J. (2003). Fundamentos de programación. En L. J. Aguilar, *Fundamentos de programación* (págs. 132, 133, 134). Madrid: Mc Graw Hill.

Battistutti, D. O. (2003). Metodología de la programación. En D. O. Battistutti, *Metodología de la programación* (págs. 52, 53, 54, 55). Mexico: Alfaomega.

Camilo Zapata, M. P. (enero de 2025). *Cienciayt*. Obtenido de Cienciayt: https://cienciayt.com/programacion/logica-de-programacion/estructura-selectiva-multiple/

Lectura 2. Lenguaje de programación de alto nivel

Instrucciones: *Realiza el análisis de la siguiente información, posteriormente, completa la* ***Actividad 1. ¡Falso o Verdadero!***

Uno de los aspectos importantes que todo programador debe conocer son los conceptos esenciales de la programación y los tipos de lenguajes de programación que facilitan la tarea de dar las instrucciones a una computadora para darle solución a los problemas planteados.

Un **Lenguaje de Programación** es un software que nos permite la construcción o elaboración de otros programas informáticos. Está formado por un conjunto de símbolos y reglas y palabras especiales que utilizamos para comunicarnos con las computadoras y otros dispositivos electrónicos. Estos lenguajes nos permiten escribir **código**, que son las instrucciones paso a paso que le indican a la máquina qué hacer, cómo hacerlo y cuándo hacerlo (Sebesta, 2012).

La **Programación** es el proceso de análisis, diseño, implementación, prueba y depuración de un algoritmo a partir de un lenguaje que compila y genera un código fuente (programa) ejecutado en la computadora.

La función principal de los lenguajes de programación es escribir programas que permiten la comunicación usuario-máquina. Para ello necesita de unos programas especiales que convierten las instrucciones escritas en código fuente a instrucciones escritas en lenguaje máquina (0 y 1). A estos programas se les conoce como compiladores e intérpretes (Aho et al., 2006).

La diferencia que existe entre estos es que el compilador recibe todo el código fuente, lo analiza, optimiza y traduce a lenguaje máquina dejando un programa completo listo para su ejecución mientras que los intérpretes leen la instrucción línea por línea, o sea, a medida que el programa se va ejecutando traduce las instrucciones y obtienen el código máquina al finalizar la ejecución.

Clasificación de los lenguajes de programación.

Existe una gran categorización de los lenguajes de programación que permiten a los desarrolladores poder entender las características y capacidades de cada uno de ellos para elegir de forma correcta la herramienta adecuada de acuerdo con la necesidad de su proyecto (Tucker & Noonan, 2006). Algunos de los criterios considerados para esta clasificación son los siguientes:

1. **Según su nivel de abstracción**
2. **Por paradigma de programación**
3. **Por generación**

1.- Nivel de abstracción

Lenguaje de bajo nivel: Son aquellos que están más cerca del lenguaje de máquina, es decir, de las instrucciones que el hardware puede ejecutar directamente. Ofrecen un control muy preciso sobre el hardware, pero son más difíciles de leer y escribir para los humanos. Entre estos se encuentra, lenguaje máquina (0,1) y el lenguaje ensamblador (Brookshear & Brylow, 2019).

Lenguaje de alto nivel: son aquellos que utilizan una sintaxis cercana al lenguaje humano, lo que facilita su lectura, escritura y mantenimiento (Sebesta, 2012). Estos lenguajes permiten a los programadores concentrarse en la lógica del problema sin preocuparse por los detalles específicos del hardware de la computadora. A diferencia de los lenguajes de bajo nivel, que están más relacionados con el funcionamiento interno del equipo, los lenguajes de alto nivel son más abstractos y fáciles de comprender.

Características de los lenguajes de alto nivel

- **Fácil comprensión**
- **Portabilidad**
- **Mayor productividad**
- **Mantenimiento sencillo**
- **Independencia del hardware**

Ejemplos de lenguajes de programación de alto nivel

- **Python: Es uno de los lenguajes más populares debido a su simplicidad y versatilidad. Se utiliza en desarrollo web, inteligencia artificial, análisis de datos y automatización.**
- **Java: Es ampliamente utilizado para desarrollar aplicaciones empresariales, móviles y sistemas distribuidos.**
- **C++: Combina características de alto y bajo nivel, siendo utilizado en videojuegos, sistemas operativos y aplicaciones de alto rendimiento.**
- **JavaScript: Es el lenguaje principal para el desarrollo de páginas web interactivas y aplicaciones web modernas.**
- **C#: Se utiliza principalmente para el desarrollo de aplicaciones de escritorio, videojuegos y soluciones empresariales.**

Ventajas de los lenguajes de alto nivel

- **Reducen el tiempo de desarrollo.**

- **Facilitan el aprendizaje de la programación.**
- **Permiten escribir menos líneas de código.**
- **Favorecen el trabajo colaborativo entre programadores.**
- **Ofrecen una gran cantidad de bibliotecas y herramientas.**
- **Son ideales para proyectos educativos y profesionales.**

2.- Paradigma de programación

Tabla 1

Clasificación de los lenguajes según su paradigma de programación

Paradigma	Descripción	Ejemplos
Imperativo	Se basa en instrucciones secuenciales que modifican el estado del programa.	C, Pascal, Fortran, Cobol
Declarativo	Describe el problema y no el proceso para resolverlo, enfocándose en el qué en lugar del cómo, es decir, se especifica el resultado deseado, no cómo lograrlo	SQL, Prolog, Haskell, Lisp, Miranda
Orientado a objetos	Organiza el código en objetos que encapsulan datos y comportamiento en lugar de funciones y lógicas. El objeto es el campo de datos que tiene atributos y comportamientos únicos.	Java, JavaScrpt, C++, Python, Ruby
Orientado a eventos	La estructura y la ejecución de los programas se determinan por las acciones que ocurren en el sistema, definidas por el usuario o por el propio sistema	Java, JavaScript, C++
Orientado a problemas	Las instrucciones están diseñadas para hacer frente a un problema específico	Prolog
Naturales	Pretenden aproximar el diseño y la construcción de programas al lenguaje de las personas. El procesamiento del lenguaje natural (PNL) es el campo de conocimiento de la inteligencia artificial (IA) que investiga la forma de comunicar las maquinas con las personas mediante el uso de lenguas naturales	Python, Lenguaje en R, Java, PHP

Nota. Adaptado de Programming Languages: Principles and Paradigms, por A. B. Tucker y R. E. Noonan, 2006, McGraw-Hill.

3.- Por generación

Tabla 2

Evolución Cronológica y Clasificación por Generaciones

Generación	Descripción	Ejemplos
Primera generación	Cada computadora tiene sólo un lenguaje de programación que su procesador puede ejecutar. Las instrucciones se codifican en código binario (0,1)	Lenguaje maquina y ensambladores
Segunda generación	Primeros lenguajes de alto nivel imperativo	Fortran, Cobol

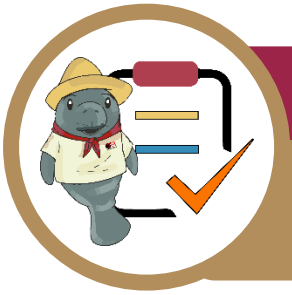
Tercera generación	Lenguaje de alto nivel imperativo. Son parecidos al inglés. Actualmente son muy utilizados	Algol, Pascal, Ada, Lisp, Prolog
Cuarta generación	Orientados básicamente a las aplicaciones de gestión y manejo de base de datos	SQL, Natural
Quinta generación	Orientados a la inteligencia artificial y al procesamiento de los lenguajes naturales	LISP, Prolog

Nota. Basado en la segmentación histórica del desarrollo de software de ingeniería de sistemas (Brookshear & Brylow, 2019).

Actividad 2. ¡Falso o Verdadero!

Instrucciones: E*ncontrarás una serie de afirmaciones relacionadas con la lectura. Lee cada afirmación y determinar si es VERDADERA (V) o FALSA (F) escribiendo en el espacio proporcionado.*

Respuesta	Justificación
	Un lenguaje de programación es un conjunto de símbolos, reglas y palabras especiales que permiten comunicarse con una computadora.
	La programación consiste únicamente en escribir código sin necesidad de analizar o diseñar soluciones.
	Los compiladores traducen el código fuente completo a lenguaje máquina antes de ejecutar el programa.
	Los intérpretes traducen todo el programa al mismo tiempo antes de iniciar su ejecución.
	Los lenguajes de bajo nivel son más fáciles de leer y escribir para los humanos que los lenguajes de alto nivel.
	Los lenguajes de alto nivel permiten a los programadores enfocarse en la lógica del problema sin preocuparse demasiado por el hardware.
	Python es un ejemplo de lenguaje de programación de alto nivel utilizado en inteligencia artificial y análisis de datos.
	La portabilidad y la independencia del hardware son características de los lenguajes de alto nivel.
	El paradigma de programación orientado a objetos organiza el código en objetos que encapsulan datos y comportamientos.
	Los lenguajes de quinta generación están orientados principalmente a aplicaciones de gestión y bases de datos.



INSTRUMENTO DE EVALUACIÓN - RUBRICA

Actividad 3: ¡Falso o Verdadero!

DATOS GENERALES	
Nombre(s) del alumno (s)	**Matriculas(s)**
Producto: Tabla de Falso o Verdadero	**Fecha**
UAC: Programación	**Periodo:2026-2027A**
Nombre del docente:	**Firma del docente**

Criterio	Excelente (3 puntos)	Satisfactorio (2 puntos)	Necesita Mejorar (1 punto)	No Cumple (0 puntos)
Exactitud de las Respuestas	Responde correctamente a 9-10 de las afirmaciones.	Responde correctamente a 7-8 de las afirmaciones.	Responde correctamente a 6 o menos de las afirmaciones.	|No responde correctamente a ninguna o casi ninguna afirmación.
Comprensión Conceptual	Demuestra una comprensión clara de los conceptos al responder correctamente y explica la falsedad de la afirmación de manera precisa (si lo realiza).	Demuestra una comprensión básica de los conceptos al responder correctamente. La explicación de la justificación podría ser incompleta o ligeramente imprecisa (si lo realiza).	Muestra una comprensión limitada de los conceptos, evidenciado por varias respuestas incorrectas. La explicación de la justificación es vaga o incorrecta (si lo realiza).	No demuestra comprensión de los conceptos clave de la lectura y no explica la falsedad de las afirmaciones
Participación y Cumplimiento	Completa la actividad dentro del tiempo asignado y responde a todas las afirmaciones. Realiza las 3 justificaciones de las respuestas falsas	Completa la actividad dentro del tiempo asignado y responde a la mayoría de las afirmaciones. Intenta el bono. Realiza al menos 2 justificaciones de las respuestas falsa	Completa la actividad fuera del tiempo asignado o deja varias afirmaciones sin responder. Realiza al menos 1 justificación de las respuestas falsas	No completa la actividad o responde muy pocas afirmaciones. No realiza la justificación de las respuestas falsas.
Logros:		**Aspectos de Mejora:**		

Referencias

Aho, A. V., Lam, M. S., Sethi, R., & Ullman, J. D. (2007). Compilers: Principles, Techniques, & Tools with Gradiance.

Brookshear, G., & Brylow, D. (2024). Computer science: An overview.

Sebesta, R. W. (2016). Concepts of programming languages. Pearson Education India.

Tucker, A. B., & Noonan, R. E. (2006). Programming languages: Principles and paradigms (2nd ed.). McGraw-Hill.

Lectura 3. Python

Instrucciones: *Realiza el análisis de la siguiente información, posteriormente, completa la* ***Actividad 3. ¿Qué tipo de variables tengo?***

Los lenguajes de programación de alto nivel han revolucionado la forma en que las personas desarrollan software, ya que permiten crear programas utilizando instrucciones fáciles de leer y comprender. Entre todos ellos, Python destaca por su simplicidad, versatilidad y amplia aplicación en áreas como la inteligencia artificial, el análisis de datos, la automatización, el desarrollo web y la ciberseguridad (Python, 2021).

Python es un lenguaje de programación de alto nivel creado por Guido van Rossum en 1991. Su principal característica es que utiliza una sintaxis clara y sencilla, lo que facilita el aprendizaje para principiantes y permite desarrollar proyectos complejos con menos líneas de código que otros lenguajes. Como señala Chazallet (2016), la legibilidad del código es un pilar fundamental en la filosofía de este entorno. Dentro de las aplicaciones de Python están:

- **Desarrollo Web:** Permite crear sitios y aplicaciones web dinámicas.
- **Inteligencia Artificial:** Es ampliamente utilizado en el desarrollo de sistemas inteligentes y aprendizaje automático.
- **Ciencia de Datos:** Facilita el análisis y procesamiento de grandes cantidades de información.
- **Automatización:** Permite automatizar tareas repetitivas y ahorrar tiempo.
- **Desarrollo de Aplicaciones:** Puede utilizarse para crear programas de escritorio y aplicaciones móviles.

Así mismo sus ventajas son:

- Fácil de aprender.
- Sintaxis sencilla y legible.
- Gran comunidad de usuarios.
- Amplia documentación.
- Compatible con Windows, Linux, macOS y Android.
- Gratuito y de código abierto.

Los archivos de python tienen la extensión .py. Son archivos de texto que son interpretados por el compilador. Para poder ejecutar programas en Python se necesita en primer lugar el intérprete de python, y en segundo lugar el código que queremos ejecutar.

Instalación de Python

Para empezar nuestra travesía en programación, debemos descargar un IDE o un editor de Python. De acuerdo con Python (2021), contar con un entorno interactivo adecuado reduce significativamente la fricción inicial en el aprendizaje de la sintaxis. En este caso te presentamos dos opciones:

1. Ir al sitio web oficial de Python: https://www.python.org/downloads/

   - **Dar clic en Descargar el instalador de Python correspondiente a tu sistema operativo.**

Una vez descargado, ejecuta el instalador. Marcar la opción "Add Python to PATH" durante el proceso de instalación en Windows. Esto permitirá ejecutar Python desde la línea de comandos.

- **Sigue las instrucciones del instalador y espera a que se complete la instalación.**

Ilustración 2.1. Página principal para descargar Python

En tu computadora puedes instalar el editor Mu, que puedes obtener en: https://codewith.mu/en/download

Ilustración 2.2. Pantalla Principal del Editor Mu

2. Para tu móvil Android: descarga en Play store la app Pydroid3

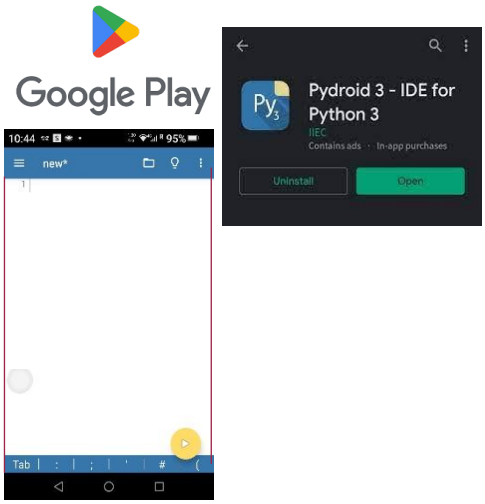


Ilustración 2.3. Pantalla Principal de Pydroid 3

Estructura básica

Dentro de la estructura básica de Python desarrollaremos los siguientes elementos:

- **Los comentarios**
- **Las variables**
- **Entrada y salida estándar: print, input.**
- **Tipo de Datos**

Los comentarios

Su propósito es detallar las secciones del programa, permitiendo identificar que realiza la línea o

bloque de código, lo cual en caso de error facilita la búsqueda. No se visualizan en la ejecución del programa. Los comentarios lo podemos realizar en línea y múltiples líneas (bloques).

a) # En línea, con el símbolo numeral **#.** Ejemplo:

```
1. 1 # Este programa calcula el área del círculo
```

b) """ **De bloque o múltiples líneas,** con comillas triple (""") de apertura y cierre**.** Ejemplo:

```
1 """
2 este programa solicita al usuario un número,
3 lo verifica, y muestra si es par o impar.
4 """
```

Entrada y salida Estándar: input, print

input ( ): Esta función permite al usuario introducir mediante el teclado un valor o cadena de caracteres, que finaliza al pulsar la tecla Enter. En Python, la función input() siempre devuelve los datos ingresados como una cadena de texto (str) (Python Software Foundation, 2026), es por ello por lo que según sea el caso el valor obtenido pueda convertirse a entero (int) o flotante (float)

print ( ): Esta función manda el valor o mensaje que se visualizará en pantalla. El prefijo f en los mensajes dentro de print se usa para crear **f-strings**, una forma eficiente y legible de formatear cadenas en Python. Permite incluir variables dentro de una cadena de texto de manera directa, sin necesidad de concatenaciones o funciones adicionales (Python Software Foundation, 2026).

Ejemplo:

Código realizado en Pydroid3

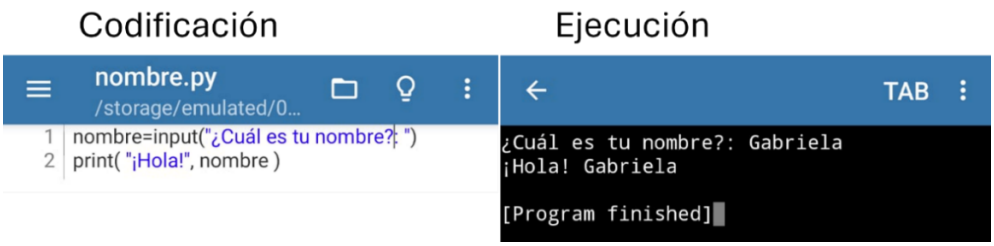


Ilustración 2.4. Código y ejecución de un programa con input y print

Las variables

Resguarda un espacio reservado en la memoria mediante un identificador (nombre) en donde se puede almacenar un valor; este valor puede cambiar a lo largo de la ejecución de un programa. Si se asigna un valor inicial a la variable, éste se coloca a continuación con el símbolo del "=". **El nombre de las variables debe tener un nombre significativo** de acuerdo con el programa que se esté realizando (Mirjalili y Raschka, 2020).

Características de las variables en Python:

- **No necesitan declaración previa: Se crean en el momento en que les asignas un valor. Ejemplo:**

```
nombre = "Enrique"
```

- **Son dinámicas: Pueden cambiar su tipo de dato en cualquier momento.**

Ejemplo:

```
nombre=input(¿Cuál es tu nombre?)
```

- **Siguen reglas de nomenclatura: No pueden empezar con un número ni contener espacios o caracteres especiales (excepto _).**
- **Son sensibles a mayúsculas y minúsculas (nombre y Nombre son distintos)**

Tipo de Datos

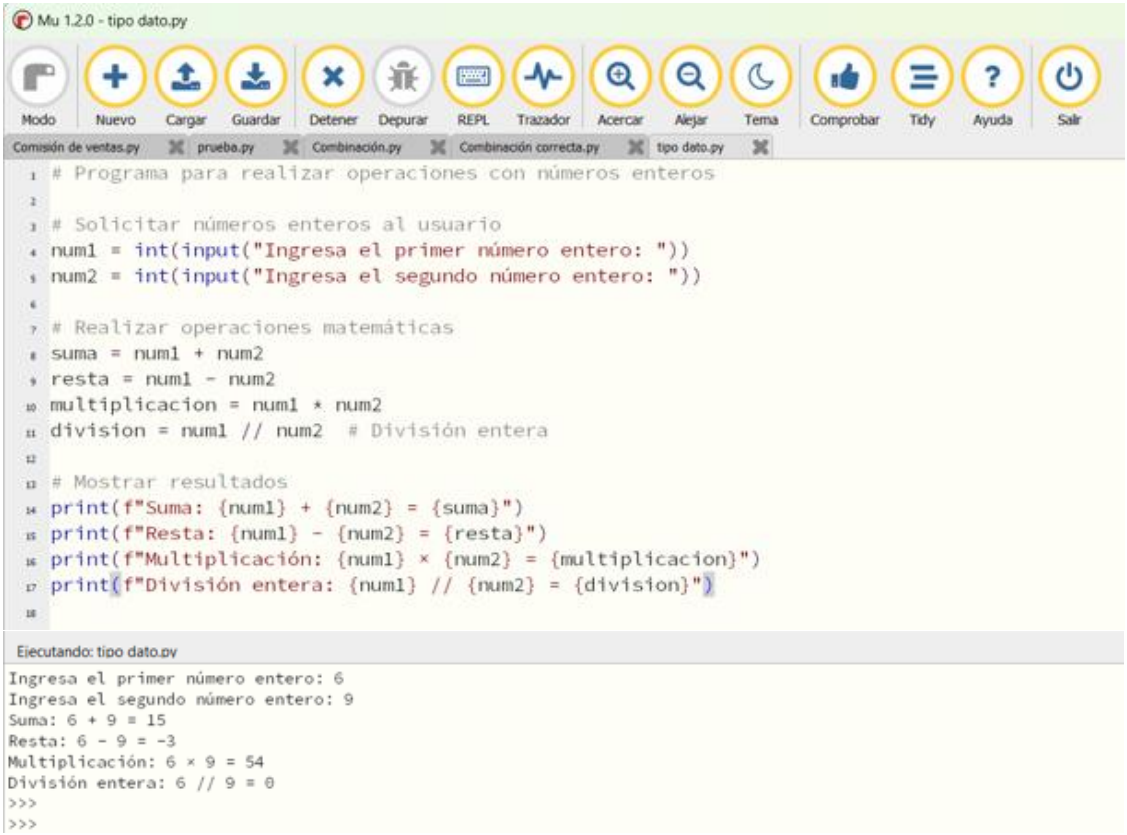
En Python, las variables pueden almacenar diferentes tipos de datos, se dividen en varias categorías, cada una con características específicas (Python Software Foundation, 2026):

```
nombre = "Patricia"  # Variable tipo string
edad = 30  # Variable tipo entero
altura = 1.65  # Variable tipo flotante
es_profesora = True  # Variable tipo booleanc

print(nombre, edad, altura, es_profesora)
```

A continuación, veremos algunos ejemplos de tipos de datos usados en Python, desarrollados con el editor MU, también puedes realizarlos en tu celular con la app Pydroid3.

- **Enteros (int):** Representa los números enteros. Ejemplo:

  Código realizado en MU Editor



- **Flotantes (Float):** Representa números con punto decimal y negativos. Ejemplo:

Ilustración 2.5. Programa con datos tipo flotante

- **Cadena (Str):** Se puedes usar str() para convertir valores de otro tipo de dato a cadenas. Cuando se asigna algo entre comillas ("texto", 'texto'), Python ya lo reconoce como una cadena sin necesidad de usar str(), según indica Van Rossum y Drake (2020).

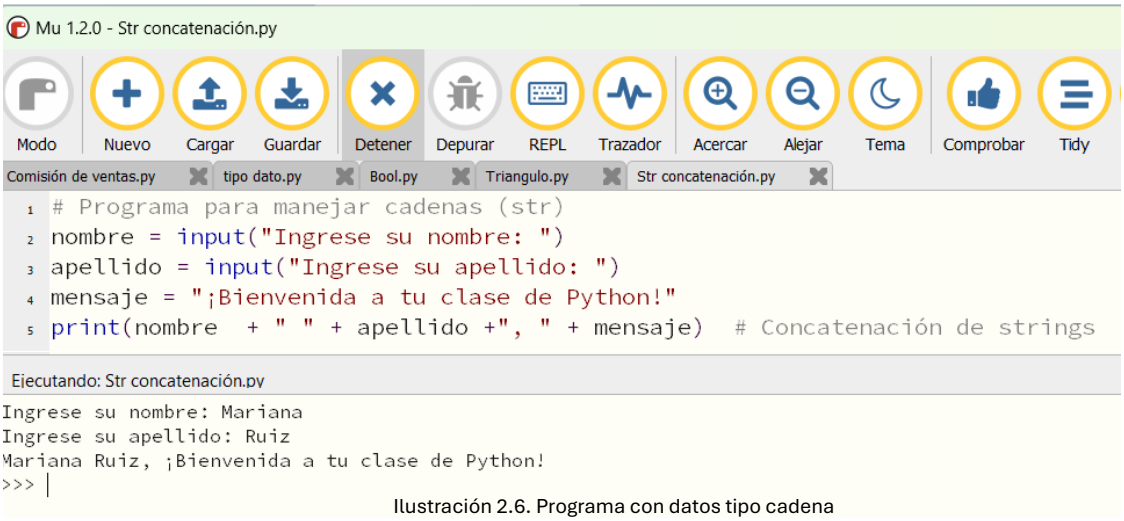


Ilustración 2.6. Programa con datos tipo cadena

Las operaciones básicas que se pueden hacer con los datos string son:

- **Concatenación (+):** Unir cadenas de texto.
- **Formato con f-strings (f""):** Incluir variables dentro de una cadena.
- **Manipulación (upper(), lower(), replace()):** Transformar texto.
- **División (split()):** Separar palabras o frases en listas.
- **Verificación (in):** Buscar si una palabra está dentro del texto.
- **Booleanos (bool):** Representa valores de lógica binaria: 0 (false) y 1 (true). Y sus operadores son:
    - **And:** Devuelve True si ambas condiciones son verdaderas.
    - **Or:** Devuelve True si al menos una condición es verdadera.
    - **No:** Invierte el valor lógico

Ilustración 2.7. Programa con resultados booleanos.

Operadores

Los operadores son símbolos que permiten realizar diferentes operaciones sobre valores y variables. Se pueden clasificar en varias categorías:

1. **Operadores Aritméticos**
2. **Operadores relacionales o de comparación**
3. **Operadores lógicos**
4. **Operadores de Asignación**
5. **Operadores de Identidad y Pertenencia**

Operadores Aritméticos: Realizan cálculos matemáticos

Tabla 1
Operadores aritméticos

1. Operador	2. Descripción	3. Ejemplo
4. +	5. Suma	6. a + b
7. -	8. Resta	9. a - b
10. *	11. Multiplicación	12. a * b
/	División	a / b
%	Módulo (resto de la división)	a % b

Nota. Adaptado de Python 3.12.x documentation, por Python Software Foundation (2026)

Operadores relacionales o de Comparación: Comparan valores y devuelven True o False

Tabla 2

Operadores relacionales o de comparación

Operador	Descripción	Ejemplo (Cuando a=5, b=3)	Resultado
==	Igualdad	a == b	False
!=	Diferente o distinto	a != b	True
>	Mayor que	a > b	True
<	Menor que	a < b	False
>=	Mayor o igual	a >= b	True
<=	Menor o igual	a <= b	False

Nota. Adaptado de Python 3.12.x documentation, por Python Software Foundation (2026)

Operadores Lógicos: Combinan condiciones booleanas

Tabla 3

Operadores lógicos

Operador	Descripción	Ejemplo Ejemplo (Cuando a=5, b=3)
and	Verdadero si ambas condiciones se cumplen (True)	a > 2 and b < 4
Or	Verdadero si al menos una condición se cumple (True)	a > 2 or b > 4
not	Niega el valor o resultado de la condición	not(a > b)

Nota. Adaptado de Python 3.12.x documentation, por Python Software Foundation (2026)

Operadores de Asignación *(Asignan valores a variables)*

Tabla 4

Operadores de asignación

Operador	Descripción	Ejemplo
=	Asignación	x = 10
+=	Suma y asigna	x += 5 (equivale a x = x + 5)
-=	Resta y asigna	x -= 2

*=	Multiplica y asigna	x *= 3
/=	Divide y asigna	x /= 2
%=	Módulo y asigna	x %= 3
**=	Exponente y asigna	x **= 2
//=	División entera y asigna	x //= 3

Nota. Adaptado de Python 3.12.x documentation, por Python Software Foundation (2026)

Operadores de Identidad y Pertenencia

Tabla 5
Operadores de identidad y pertinencia

Operador	Descripción	Ejemplo
Is	¿Apuntan a la misma referencia en memoria?	a is b
is not	¿No apuntan a la misma referencia?	a is not b
in	¿Está un elemento en una secuencia?	"a" in "apple"
not in	¿No está un elemento en una secuencia?	"x" not in "apple"

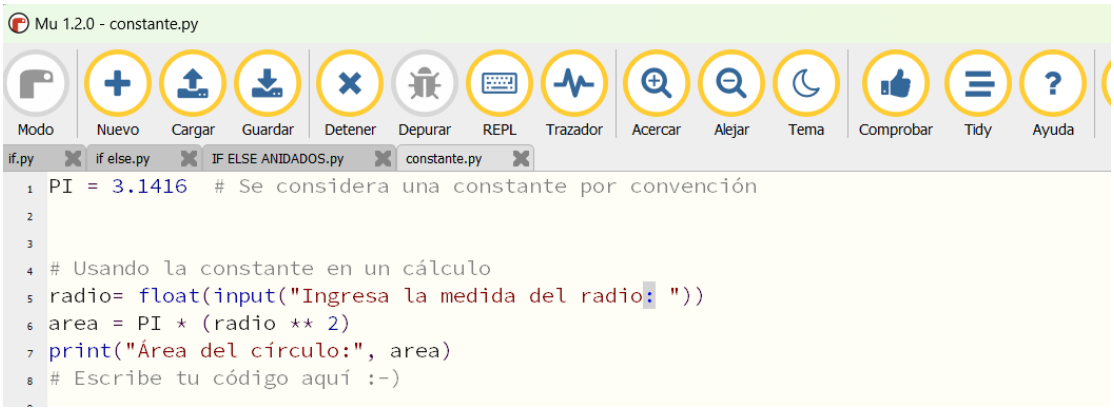
Nota. Adaptado de Python 3.12.x documentation, por Python Software Foundation (2026)

Constantes

En Python, una **constante** es una variable cuyo valor no cambia a lo largo de la ejecución del programa. Aunque Python no tiene una sintaxis específica para definir constantes (Chazallet, 2016), se sigue una convención para nombrarlas:

1. **Usar nombres en mayúsculas** para indicar que una variable debe tratarse como constante.
2. **Evitar modificar su valor** después de asignarlo.

Ejemplo de constantes en Python:



```
Ejecutando: constante.py
Ingresa la medida del radio: 7
Área del círculo: 153.9384
>>>
```

Ilustración 2.8. Programa con declaración de variables

Palabras Reservadas

Las **palabras reservadas** en Python son identificadores que tienen un significado especial en el lenguaje y **no pueden ser usados como nombres de variables, funciones o identificadores** (Van Rossum y Drake, 2017). Estas palabras forman parte de la sintaxis y estructura de Python. Para encontrar las palabras reservadas en Python, escribimos el siguiente código:

```
1 # Programa que importa las palabras claves
2 import keyword
3 print (keyword.kwlist)
4
```

Ilustración 2.8. Programa que enlista las palabras reservadas.

['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']

Actividad 3. ¿Qué tipo de variables tengo?

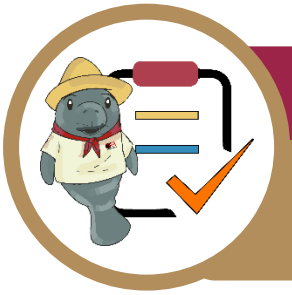
Ejercicio 1: Clasificación de tipos de datos

Instrucciones: *Revisa la columna de las variables y escribe en la columna siguiente el tipo de datos a la que pertenece:*

Variables	*Tipo de Dato*
Edad = 25	
Entrada = "Hola, mundo"	
Suma = 3.1416	
Estudiante= True	
Dinero = [10, 20, 30]	
Tabla = {"nombre": "Mauricio", "edad": 25}	
Valores = (5, 8)	

Ejercicio 3: Conversión de tipo de datos

Instrucciones: *Realizar un programa en Python que convierta pesos a dólar, solicitando al usuario la cantidad en pesos a convertir y mostrando su equivalente en pantalla.*



INSTRUMENTO DE EVALUACIÓN - LISTA DE COTEJO

Actividad 2: ¿Qué tipo de variable tengo?

DATOS GENERALES	
Nombre(s) del alumno (s):	**Matriculas(s)**
Producto: Programa que convierte pesos a dólar	**Fecha:**
UAC: Programación	**Periodo:2026-2027A**
Nombre del docente:	**Firma del docente**

CRITERIO	VALOR OBTENIDO		PONDERACIÓN		OBSERVACIONES Y/O SUGERENCIAS
	SI	NO			
El programa se ejecuta correctamente			4		
Utiliza nombre de variables significativos y descriptivos			2		
Incluye comentarios claros y descriptivos			1		
La estructura del código es clara y bien organizada.			1		
Implementa validación de entrada para evitar errores.			2		
			CALIFICACIÓN		
Logros			Aspectos de Mejora		

Referencias

Chazallet, S. (2016). *Python 3: los fundamentos del lenguaje*.

Mirjalili, V. y Raschka, S. (2020). *Aprendizaje automático de Python*. Marcombo.

Python Software Foundation. (2026, 3 de junio). Python 3.12.x documentation. https://docs.python.org/3/

Python, W. (2021). Python. *Versiones de Python para Windows*, *24*.

Van Rossum, G., y Drake, F. L. (2017). El tutorial de Python. Python Software Foundation.

Van Rossum, G., y Drake, F.L. (2020). *Introducción a Python* (p. 115). Bristol: Network Theory Ltd.

Lectura 4. Funciones

Instrucciones: *Realiza el análisis de la siguiente información, con sus ejercicios de ejemplo para practicar, posteriormente, completa la* ***Actividad sugerida***

En la programación, una función es un bloque de código reutilizable que realiza una tarea específica. En Python, las funciones permiten organizar el código, mejorar su legibilidad y evitar la repetición, lo que facilita su mantenimiento y escalabilidad.

Importancia de las funciones en Python

Las funciones son esenciales en la programación porque permiten dividir un problema grande en partes más pequeñas y manejables. Además, promueven la reutilización del código, lo que ahorra tiempo y reduce errores. Gracias a ellas, los programas se vuelven más modulares, estructurados y fáciles de entender.

Declaración de funciones en Python

Para definir una función en Python, se usa la palabra clave def, seguida del nombre de la función y paréntesis que pueden contener parámetros opcionales. Luego, el bloque de código de la función se escribe con identación. Una función puede devolver un valor utilizando la palabra clave return.

La estructura básica de una función en Python es la siguiente:

```
# Definición de una función
def nombre_de_la_funcion(parámetros):
    """Descripción de la función"""
    # Cuerpo de la función
    return resultado
```

Ejemplos de funciones en Python

1. Función sin parámetros y sin valor de retorno

```
def saludar():

    print("¡Hola, bienvenido a Python!")

saludar()
```

Nombre de la función Saludar

Print imprime el mensaje

Llamada de la función saludar

Imagen1 función sin argumentos [creación propia]

2. Función con parámetros y sin valor de retorno

```
# Definimos una función con parámetros
def saludar(nombre, edad):
    print(f"Hola {nombre}, tienes {edad} años.")

# Llamamos a la función pasando valores a los paráme
saludar("Ana", 25)
saludar("Luis", 30)
```

Imagen2: función con parámetros y sin valor de retorno [creación propia]

1. def saludar (nombre, edad): → Aquí se define la función llamada saludar.

   - def indica que estamos creando una función.
   - saludar es el nombre de la función.
   - (nombre, edad) son los parámetros que recibirá la función cuando la llamemos.

2. print (f"Hola {nombre}, tienes {edad} años.") → Dentro de la función, usamos print para mostrar un mensaje en pantalla.

   - La f delante de las comillas indica que es una f-string (cadena formateada).
   - {nombre} y {edad} se reemplazan por los valores que pasemos al llamar la función.

3. saludar("Ana", 25) → Llamamos a la función saludar y le pasamos "Ana" como nombre y 25 como edad.

   - El resultado será: Hola Ana, tienes 25 años.

4. saludar("Luis", 30) → Otra llamada a la función, esta vez con "Luis" y 30.

   - El resultado será: Hola Luis, tienes 30 años.

3. Función con parámetros y con valor de retorno

```
def multiplicar(a, b):
   c=a * b
   return c

# Programa principal
num1 = float(input("Ingresa el primer número de la multiplicacio: "))
num2 = float(input("Ingresa el segundo número a multiplicar: "))

resultado = multiplicar(num1, num2)

print("El resultado de la multiplicación es:", resultado)
```

Imagen3: función con parámetros y con valor de retorno [creación propia]

1. def multiplicar(a, b): → Se define una función llamada multiplicar que recibe dos parámetros: a y b.
2. c = a * b → Dentro de la función, se multiplica a por b y se guarda el resultado en la variable c.
3. return c → La función devuelve el valor de c al lugar donde fue llamada.
4. num1 = float(input("Ingresa el primer número de la multiplicacio: ")) → Se pide al usuario que ingrese un número.
   - input() recibe el texto escrito por el usuario.
   - float() convierte ese texto en un número decimal (flotante).
   - El valor se guarda en la variable num1.

5. num2 = float(input("Ingresa el segundo número a multiplicar: ")) → Igual que la línea anterior, pero ahora se pide el segundo número y se guarda en num2.
6. resultado = multiplicar(num1, num2) → Se llama a la función multiplicar pasando como argumentos num1 y num2.
   - El resultado de la multiplicación se guarda en la variable resultado.
7. print("El resultado de la multiplicación es:", resultado) → Se muestra en pantalla el mensaje junto con el valor de resultado.

4. Función con valores por defecto

```
def saludar(nombre="Invitado", mensaje="Bienvenido"):
    print(f"Hola {nombre}, {mensaje}!")

# Ejemplos de uso
saludar()                    # Usa ambos valores por defecto
saludar("Carlos")            # Cambia solo el nombre
saludar("Ana", "gracias por venir") # Cambia ambos valores
```

Imagen 4: función con valores por defecto [creación propia]

1. def saludar(nombre="Invitado", mensaje="Bienvenido"): → Se define una función llamada saludar con dos **parámetros**:

- nombre, cuyo valor por defecto es "Invitado".
- mensaje, cuyo valor por defecto es "Bienvenido". Si el usuario no proporciona valores al llamar la función, se usarán estos por defecto.

2. print(f"Hola {nombre}, {mensaje}!") → Dentro de la función se imprime un saludo usando una **f-string**, que permite insertar variables dentro del texto. Por ejemplo, si nombre = "Ana" y mensaje = "gracias por venir", el resultado será: **Hola Ana, gracias por venir!**
3. # Ejemplos de uso → Comentario que indica que las siguientes líneas muestran cómo se puede usar la función.

4. saludar() → Llamada sin argumentos. Usa los valores por defecto: nombre = "Invitado" y mensaje = "Bienvenido". Resultado: **Hola Invitado, Bienvenido!**
5. saludar("Carlos") → Se pasa solo el primer argumento (nombre = "Carlos"). El segundo (mensaje) conserva su valor por defecto "Bienvenido". Resultado: **Hola Carlos, Bienvenido!**
6. saludar("Ana", "gracias por venir") → Se pasan ambos argumentos, reemplazando los valores por defecto. Resultado: **Hola Ana, gracias por venir!**

Las funciones en Python son una herramienta fundamental para estructurar mejor los programas y hacerlos más eficientes. Al dominar su uso, los programadores pueden escribir código más limpio, reutilizable y fácil de mantener, lo que mejora la calidad y productividad en el desarrollo de software.

Referencias

Funciones en Python. (s/f). El Libro De Python. Recuperado el 27 de abril de 2026, de https://ellibrodepython.com/funciones-en-python

Funciones incorporadas. (s/f). Python documentation. Recuperado el 27 de abril de 2026, de https://docs.python.org/es/3.13/library/functions.html

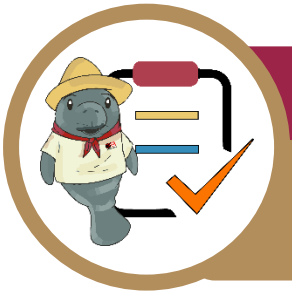
W3Schools. (s.f.). *Python functions*. W3Schools. Recuperado el 29 de abril de 2026, de https://www.w3schools.com/python/python_functions.asp

Práctica 1. Funciones y parámetros

Instrucciones: *Creen un programa que realice las cuatro operaciones matemáticas básicas (suma, resta, multiplicación y división) usando funciones. Esto les ayudará a practicar la definición de funciones, el manejo de parámetros y el control de errores (como la división entre cero).*

```python
def sumar(a, b):
    # Aquí va la función de suma

def restar(a, b):
    # Aquí va la función de resta

def multiplicar(a, b):
    # Aquí va la función de multiplicación

def dividir(a, b):
    # Aquí va la función de división, con manejo de división por cero

# Pedimos los números al usuario
num1 = float(input("Introduce el primer número: "))
num2 = float(input("Introduce el segundo número: "))

# Aquí van las llamadas a las funciones y la impresión de los resultados
```

Imagen5: practica de funciones [creación propia]



Rubrica

Practica 1 – Funciones y parámetros

DATOS GENERALES		
Nombre(s) del alumno (s)		**Matriculas(s)**
Producto: programa		**Fecha**
UAC: Programación		**Periodo:2025-2026A**
Nombre del docente:		**Firma del docente**

Criterio	Excelente (10)	Satisfactorio (7-9)	Insuficiente (0-6)
Definición de Funciones	Todas las funciones (suma, resta, multiplicación, división) están definidas y funcionan correctamente.	La mayoría de las funciones están definidas y funcionan, pero hay algún detalle menor por corregir.	Faltan una o más funciones, o varias no funcionan correctamente.
Manejo de Errores	La función de división maneja correctamente la división por cero, mostrando un mensaje adecuado.	La función de división maneja el error, pero el mensaje podría ser más claro.	No hay manejo de división por cero, o el programa se detiene sin un mensaje adecuado.
Interacción con el Usuario	El programa solicita los números de forma clara y muestra resultados comprensibles.	El programa interactúa con el usuario, pero los mensajes pueden mejorarse.	La interacción con el usuario es confusa o los resultados no se muestran claramente.
Total			

Ejercicio teórico. Tipo de Funciones

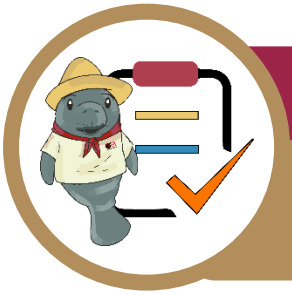
Instrucciones: *En el siguiente programa en Python dónde se utilicen Los cuatro tipos de funciones qué revisaste en la lectura, identifica cada una de las funciones que utilizaste*

```
1
2  def saludar():
3      print("¡Hola a todos!")
4  def saludar_persona(nombre):
5      print(f"¡Hola, {nombre}!")
6  def obtener_saludo():
7      return "¡Hola desde la función!"
8  def saludar_y_devolver(nombre):
9      return f"¡Hola, {nombre}! Encantado de saludarte."
10
11 saludar()
12 saludar_persona("esther")
13 print(obtener_saludo())
14 print(saludar_y_devolver("maria"))
15
```

```
¡Hola a todos!
¡Hola, esther!
¡Hola desde la función!
¡Hola, maria! Encantado de saludarte.
```

Imagen6: Tipo funciones [creación propia]

Función	Tipo
Obtener saludo	
Saludar y devolver	
Saludar	
Saludar persona	



Rubrica

Ejercicio teórico – Funciones y parámetros

DATOS GENERALES			
Nombre(s) del alumno (s)			**Matriculas(s)**
Producto: Ejercicio			**Fecha**
UAC: Programación			**Periodo:2025-2026A**
Nombre del docente:			**Firma del docente**

Criterio	Excelente (10)	Satisfactorio (7-9)	Insuficiente (0-6)
Identificación de tipos de funciones	Reconoce correctamente los 4 tipos de funciones (sin parámetros, con parámetros, con retorno, con parámetros y retorno).	Identifica 2 o 3 tipos correctamente.	Identifica 1 o ningún tipo correctamente.
Justificación del tipo	Explica con claridad y precisión por qué cada función pertenece a su tipo, usando vocabulario técnico adecuado.	Explica parcialmente o con errores menores en la terminología.	No justifica o lo hace de forma incorrecta.
Comprensión del código	Demuestra comprensión del flujo del programa y del uso de cada función.	Comprende parcialmente el funcionamiento del código.	No demuestra comprensión del código o presenta confusión evidente.
Presentación y organización	Entrega ordenada, con tabla completa y formato claro.	Presentación aceptable, aunque con detalles menores de formato o incompletitud.	Entrega desorganizada o sin completar la tabla.

Lectura 5. Todo se ve mejor en ventanas

Instrucciones: *Realiza el análisis de la siguiente información, con sus ejercicios de ejemplo para practicar, posteriormente, completa la* ***Actividad sugerida.***

Introducción a Tkinter en Python

Tkinter es la biblioteca estándar de Python para crear interfaces gráficas de usuario (GUI). Es una envoltura (wrapper) de la biblioteca Tcl/Tk que permite crear ventanas, botones, cuadros de texto, menús, entre otros elementos gráficos, de forma sencilla y multiplataforma.

Conceptos Importantes

1. Widget

Un widget es cualquier elemento de la interfaz de usuario: botón, etiqueta, cuadro de texto, etc.

2. Ventana raíz (root)

Es la ventana principal de la aplicación. Todos los demás widgets se colocan sobre ella o sobre otros contenedores.

3. Loop principal (mainloop)

Es el bucle que mantiene activa la ventana y escucha eventos como clics o entradas de teclado.

¿Cómo funciona Tkinter?

1. **Se importa la librería.**
2. **Se crea la ventana principal (Tk()).**
3. **Se agregan los widgets (botones, etiquetas, etc.).**
4. **Se empaquetan los widgets (pack, grid o place).**
5. **Se ejecuta el mainloop() para iniciar la app.**

Declaración y Estructura Básica

```
import tkinter as tk
root = tk.Tk()
root.title(" HOLA MUNDO")
root.geometry("300x200")
etiqueta = tk.Label(root, text="¡Hola, Tkinter!")
etiqueta.pack()
root.mainloop()
```

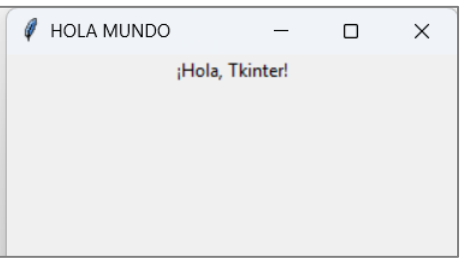


Imagen 1 ventana [fuente: creación propia]

Descripción de código

1. **import tkinter as tk** – Importa la librería Tkinter y la renombra como tk. Esto te permite usar sus funciones para crear interfaces gráficas.
2. **ventana = tk.Tk()** – Crea la ventana principal de la aplicación y la guarda en la variable ventana.
3. **ventana.title("Mi primera practica de tkinter")** – Le asigna un título a la ventana: "Mi primera práctica de tkinter".
4. **ventana.geometry("300x200")** – Define el tamaño de la ventana: 300 píxeles de ancho por 200 de alto.
5. **etiqueta = tk.Label(ventana, text="¡Hola, Tkinter!")** – Crea una etiqueta dentro de la ventana que mostrará el texto "¡Hola, Tkinter!".
6. **etiqueta.pack()** – Coloca la etiqueta en la ventana usando el administrador de geometría pack(), que la organiza automáticamente.
7. **ventana.mainloop()** – Inicia el bucle principal de la aplicación. Esto mantiene la ventana abierta y lista para interactuar hasta que el usuario la cierre.

Tipos de datos de opciones de widget

Los widgets de Tkinter aceptan diversas opciones configurables:

Tipo de oración	Ejemplo	Descripción
Text	"Aceptar"	Texto que se muestra en el widget
bg o background	"Blue"	Color de fondo
Fg o foreground	"White"	Color de texto
font	("Arial", 12)	Fuente de texto

command	Funccion()	Accion al hacer clic (en botones)

Asociación de variables de widget

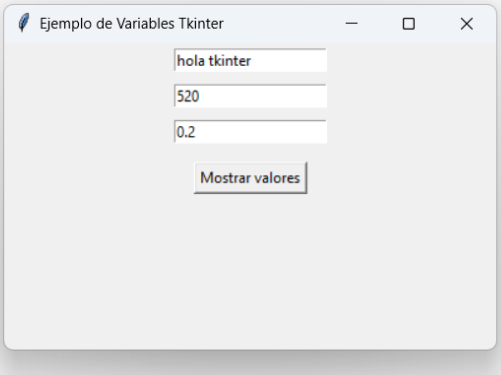
Tkinter permite asociar widgets con variables especiales llamadas variables de control:

- **StringVar() para texto**
- **IntVar() para enteros**
- **DoubleVar() para decimales**
- **BooleanVar() para valores lógicos**

Ejemplo de usos de datos

imcm.py | ventana.py | calculadora.py | imc.py * | CODIGO 1 ´1.py | tipos de datos para entrada.py

```
1  import tkinter as tk
2  root = tk.Tk()
3  root.title("Ejemplo de Variables Tkinter")
4  root.geometry("400x250")
5
6  def mostrar():
7      print("Texto:", texto.get())
8      print("Entero:", entero.get())
9      print("Decimal:", decimal.get())
10
11 texto = tk.StringVar()
12 entero = tk.IntVar()
13 decimal = tk.DoubleVar()
14
15 entrada_texto = tk.Entry(root, textvariable=texto)
16 entrada_texto.pack(pady=5)
17 entrada_entero = tk.Entry(root, textvariable=entero)
18 entrada_entero.pack(pady=5)
19 entrada_decimal = tk.Entry(root, textvariable=decimal)
20 entrada_decimal.pack(pady=5)
21
22 boton = tk.Button(root, text="Mostrar valores", command=mostrar)
23 boton.pack(pady=10)
24
25 root.mainloop()
```



Consola

```
Texto: hola tkinter
Entero: 520
Decimal: 0.2
```

Imagen 2: ventana mostrar [creación propia]

Resultado: al dar clic en ejecutar aparece la ventana de la derecha a la que proporcionamos datos y al dar clic en mostrar valores da los resultados en la parte de debajo de la consola. En este ejemplo declaramos variables como texto, entero y decimal para su uso en los widgets de cuadro de textos por ejemplo para hablarlas y asignarle en "textvariable = text y esta sea tomada como texto o entero o decimal dependiendo del caso.

Diálogos de entrada estándar

Tkinter ofrece funciones modales para interacción con el usuario:

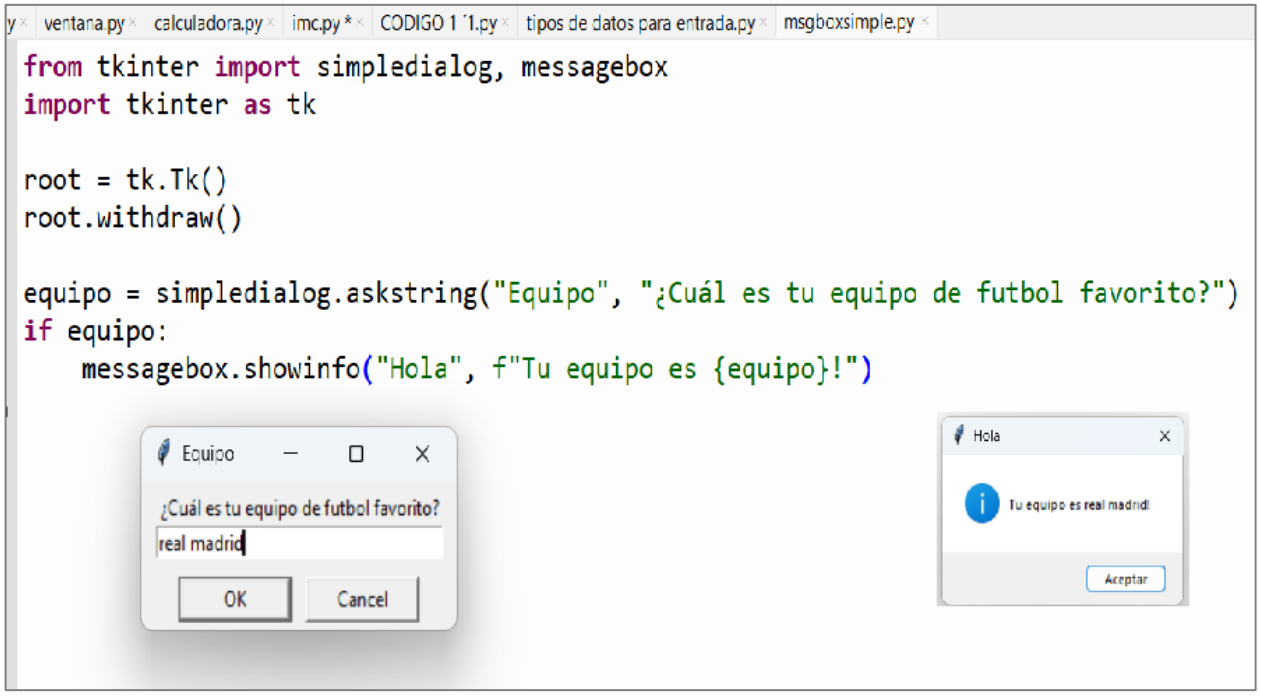


Imagen 3: cuadro de diálogos [creación propia]

Interpretación del código:

1. **from tkinter import simpledialog, messagebox**

Aquí le decimos a Python que queremos usar dos herramientas de la librería **Tkinter**, simpledialog: sirve para mostrar una ventana emergente donde el usuario puede escribir algo. messagebox: sirve para mostrar mensajes en una ventana emergente.

2. **import tkinter as tk**

Importamos la librería principal **Tkinter** con el alias tk. Esto nos permite crear la ventana principal de la aplicación.

3. **root = tk.Tk()**

Creamos la **ventana principal** de la aplicación. Aunque no la vamos a mostrar, es necesaria para que Tkinter funcione.

4. **root.withdraw()**

Ocultamos la ventana principal. De esta forma, solo veremos los cuadros de diálogo y no una ventana vacía.

5. **equipo = simpledialog.askstring("Equipo", "¿Cuál es tu equipo de futbol favorito?")**

Se abre un cuadro de diálogo que pregunta: *"¿Cuál es tu equipo de futbol favorito?"*. Lo que el usuario escriba se guarda en la variable equipo.

6. **if equipo:**

Esta línea significa: "Si el usuario escribió algo (es decir, si la variable equipo no está vacía)".

7. **messagebox.showinfo("Hola", f"Tu equipo es {equipo}!")**

   Si el usuario respondió, se abre una ventana emergente que muestra el mensaje: *"Tu equipo es [nombre del equipo]!"*. El texto dentro de {equipo} se reemplaza automáticamente por lo que el usuario escribió.

Resultado: ¡se puede apreciar al ejecutar la aplicación que debe aparecer en la ventana o formulario principal pidiendo ingresemos nuestro equipo favorito para el ejemplo "el super Real Madrid!" y demos clic en el botón aceptar aparecerá el siguiente cuadro de dialogo creado y volverá invisible la ventana principal.

Resultado: Una pequeña app donde el usuario escribe su nombre y al hacer clic en "Saludar", aparece una ventana emergente con un saludo personalizado.

Tkinter es una poderosa herramienta para desarrollar interfaces gráficas en Python de forma rápida y sencilla. Entender cómo se declaran los widgets, se asocian a variables, y cómo crear ventanas personalizadas te da una base sólida para construir aplicaciones interactivas.

Ejercicio para practicar

1. **Importar el módulo tkinter y le definimos un alias luego de la palabra clave as. Es más fácil luego escribir 'tk' en lugar de 'tkinter':**

   import tkinter as tk

2. **La librería tkinter está codificada con la metodología de programación orientada a objetos. El módulo 'tkinter' tiene una clase llamada 'Tk' que representa una ventana. Debemos crear un objeto de dicha clase:**

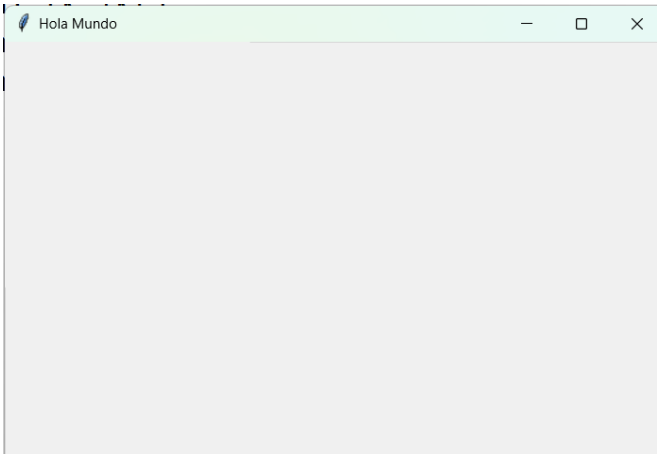
   13. **ventana1=tk.Tk()**

3. **Seguidamente llamamos al método 'title' y le pasamos un string con el mensaje que queremos que aparezca en la barra del título de la ventana:**

   14. **ventana1.title("Hola Mundo")**

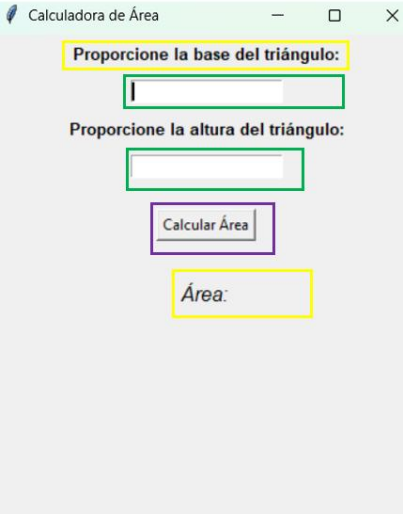
4. **Para que se muestre la ventana en el monitor debemos llamar por último al método 'mainloop' que pertenece a la clase Tk:**

   15. **ventana1.mainloop()**



lo cual quedara de la siguiente manera.

5. **Como siguiente paso sería agregar los elementos que utilizaremos como control como ejemplo utilizaremos una aplicación del área de un triángulo por lo cual necesitamos los siguientes elementos**



6. **Donde aquí utilizaremos tres elementos los cuales son los label o etiquetas (color amarillo) los entry o cuadros de textos(oro) y button (azul) para realizar los cálculos.**
7. **Aquí logramos la primera etiqueta en amarillo utilizando una etiqueta la base del triángulo donde también le asignamos el tipo de letra Arial, tamaño 10 en negritas**

```
# Base del triángulo
tk.Label(ventana1, text="Proporcione la base del triángulo:",
        font=("Arial", 10, "bold")).pack(pady=5)
```

Imagen 4area del triangulo [creación propia]

8. **El siguiente paso es agregar el cuadro de texto o entry:**

```
entry_base = tk.Entry(ventana1)
entry_base.pack(pady=5)
```

9. **Ahora haremos lo mismo, pero para la etiqueta altura y su cuadro de texto:**

```
# Altura del triángulo
tk.Label(ventana1, text="Proporcione la altura del triángulo:",
         font=("Arial", 10, "bold")).pack(pady=5)
entry_altura = tk.Entry(ventana1)
entry_altura.pack(pady=5)
```

10. **Agregamos el botón que al darle click realizara el cálculo del área del triángulo aunque ahorita solo creara el botón sin función alguna**

```
# Botón para calcular
tk.Button(ventana1, text="Calcular Área",
          command=calcular_area).pack(pady=20)
```

11. Por último, la etiqueta que nos dará el resultado

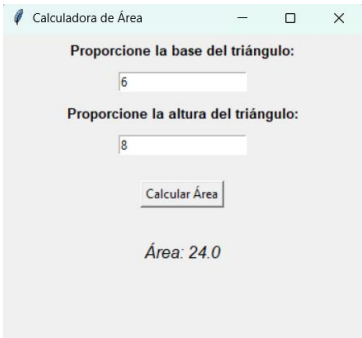
```
# Etiqueta para mostrar el resultado
label_resultado = tk.Label(ventana1, text="Área: ",
            font=("Arial", 12, "italic"))
label_resultado.pack(pady=10)
```

12. Así de esta manera obtendremos la ventana requerida, ahora tenemos que crear la función para el botón calcular, que llevara el nombre como dice en command calcular_area, pera antes debemos colocar en la segunda línea un código que invoque la librería que me permita enviar un cuadro de dialogo simple, si el usuario por error digita una tecla que no sea número, lo que ocasiona que la operación matemática no funcione:

```
from tkinter import messagebox
```

13. Ahora con la librería de cuadro de dialogo podemos crear nuestra función comenzando con un try, que permitirá evitar que por un error se detenga nuestra aplicación y un except enviar el cuadro de dialogo donde indique que solo se aceptan números quedando de la siguiente manera:

```
def calcular_area():
    try:
        # Usamos .get() para extraer el texto y luego lo convertimos
        b = float(entry_base.get())
        h = float(entry_altura.get())
        area = (b * h) / 2
        label_resultado.config(text=f"Área: {area}")
    except ValueError:
        messagebox.showerror("Error", "Por favor ingresa solo números")
```



La variable b y h almacenan lo que se encuentre en los cuadros de dialogo base y altura no sin antes convertirlo a un dato real utilizando la función predeterminada **float**, ahora si podemos ejecutar la aplicación quedando de la siguiente forma:

Imagen 5: ventana triangulo con resultados [creación propia]

Ejercicio de aprendizaje. Funciones y parámetros

Instrucciones: *Con la guía del profesor realizar la aplicación del cálculo del IMC en la cual seguiremos guiando y explicando los elementos del programa*

Ejercicio 1 cálculo del índice de masa corporal(imc)

1. **Con la guía del profesor realizar la aplicación del cálculo del IMC en la cual seguiremos guiando y explicando los elementos del programa. Comenzaremos mostrando la ventana para analizar los elementos:**

```
# Creamos la ventana principal
ventana = tk.Tk()
# Le damos un título a la ventana
ventana.title("Calculadora de IMC Profesional")
# Definimos el tamaño de la ventana
ventana.geometry("350x400")
```

2. **se importa la librería ktinker y se asigna a la constante tk para su manejo dentro de la aplicación también la librería messagebox para los cuadros de diálogos que se utilizaran posteriormente.**

```
# Importamos la librería para la interfaz gráfica
import tkinter as tk
# Importamos una versión moderna de las ventanas de mensajes
from tkinter import messagebox
# Importamos sqlite3 para manejar la base de datos
```

Imagen5 códigos [creación propia ventana]

3. Activamos la ventana asignada la función TK() de la librería, también se le asigna un título el cual es "Calculadora de IMC Profesional". Por último, asignamos el tamaño de la ventana con la

imagen 6: ventana imc[image

4. función geometryEn esta aplicación podemos observar 4 etiquetas que corresponde al nombre del estudiante, peso(kg), estatura (metros) y IMC:-- (color amarillo) que corresponde a dar el resultado, las cuales

```
tk.Label(ventana, text="Nombre del Estudiante:",
         font=("Arial", 10, "bold")).pack(pady=5)
```

están marcadas en amarillo los cuales tienen el siguiente código:

5. **Para cada una de las etiquetas solo se le cambiaria el text=”peso (kg) y así por cada una de las etiquetas.**
6. **Para los cuadro de textos o entry (morado)se utiliza:**

16.
```
entrada_nombre = tk.Entry(ventana)
entrada_nombre.pack(pady=5)
```

7. **Para estos elementos de cuadro de textos si se les debe colocar el nombre del elemento en este caso el elemento se llama entrada_nombre, aunque se podría colocar cualquier nombre a la variable para no confundirnos se utiliza de esa forma. Recuerden que para respetar la forma primero va la etiqueta y luego va el cuadro de texto o entry, es decir que seguiría la etiqueta peso (kg), seguida del cuadro de texto entrada_pesos, la etiqueta estatura y sus entry o cuadro de texto con nombre entrada_estatura.**
8. **A continuación, es el turno del boton calcular y guardar en la BD que su código quedaría**

17.
```
# Botón para ejecutar la lógica (Módulo 2)
boton_calcular = tk.Button(ventana, text="Calcular y Guardar en BD", command=calcular_y_guardar, bg="#4CAF50", fg="white")
boton_calcular.pack(pady=20)
```

9. **Como se puede ver el botón se llama boton_calcular, y lleva como texto dentro del botón “calcular y guardar en la BD”, y cuando le demos clic al botón estaremos ejecutando la**

función calcular_y_guardar, siempre el nombre de la función debe estar precedida por "command=", el color del boton está dado por bg y el de la letra por fg

10. **Cerramos con el código ventana.mainloop() que lo que hace es impedir que al finalizar se cierre la ventana y se mantenga en un ciclo. Ahora el código completo para que pruebes su funcionamiento}**

```
# Creamos la ventana principal
ventana = tk.Tk()
# Le damos un título a la ventana
ventana.title("Calculadora de IMC Profesional")
# Definimos el tamaño de la ventana
ventana.geometry("350x400")
```

```
# Importamos la librería para la interfaz gráfica
import tkinter as tk
# Importamos una versión moderna de las ventanas de mensajes
from tkinter import messagebox
```

o

```
# --- FUNCIÓN PARA CALCULAR Y GUARDAR ---
def calcular_y_guardar():
    # Obtenemos los datos de las cajas de texto
    nombre = entrada_nombre.get()

    try:
        # Convertimos el texto ingresado a números decimales (float)
        peso = float(entrada_peso.get())
        estatura = float(entrada_estatura.get())

        # Fórmula del IMC: peso / (estatura al cuadrado)
        imc = peso / (estatura ** 2)
        # Redondeamos el resultado a 2 decimales
        imc_redondeado = round(imc, 2)

        # Mostramos el resultado en la interfaz y un mensaje de éxito
        label_resultado.config(text=f"IMC: {imc_redondeado}", fg="green")
        messagebox.showinfo("Éxito", f"Registro de {nombre} guardado correctamente.")

    except ValueError:
        # Si el usuario no ingresa números válidos, mostramos un error
        messagebox.showerror("Error", "Por favor, ingresa valores numéricos válidos en peso y estatura.")
    except ZeroDivisionError:
        # Evitamos que el programa falle si la estatura es 0
        messagebox.showerror("Error", "La estatura no puede ser cero.")
```

```
# Creamos la ventana principal
ventana = tk.Tk()
# Le damos un título a la ventana
ventana.title("Calculadora de IMC Profesional")
# Definimos el tamaño de la ventana
ventana.geometry("350x400")

# Etiqueta y entrada para el NOMBRE
tk.Label(ventana, text="Nombre del Estudiante:",
         font=("Arial", 10, "bold")).pack(pady=5)
entrada_nombre = tk.Entry(ventana)
entrada_nombre.pack(pady=5)

# Etiqueta y entrada para el PESO
tk.Label(ventana, text="Peso (kg):").pack(pady=5)
entrada_peso = tk.Entry(ventana)
entrada_peso.pack(pady=5)

# Etiqueta y entrada para la ESTATURA
tk.Label(ventana, text="Estatura (metros):").pack(pady=5)
entrada_estatura = tk.Entry(ventana)
entrada_estatura.pack(pady=5)

# Botón para ejecutar la lógica (Módulo 2)
boton_calcular = tk.Button(ventana, text="Calcular y Guardar en BD",
                           command=calcular_y_guardar, bg="#4CAF50", fg="white")
boton_calcular.pack(pady=20)

# Etiqueta donde se mostrará el resultado final (Módulo 3)
label_resultado = tk.Label(ventana, text="IMC: --", font=("Arial", 14, "bold"))
label_resultado.pack(pady=10)

# Iniciamos el bucle de la aplicación
ventana.mainloop()
```

Imagen 7 [creación propia ventana]

Referencias

Python Software Foundation. (2024). *tkinter.messagebox — Indicadores de mensajes de Tkinter*. En *Documentación de Python 3.10.20* (versión en español). Recuperado el 13 de mayo de 2026, de https://docs.python.org/es/3.10/library/tkinter.messagebox.html

Python, R. (2018, agosto 14). Cuadros de diálogo (messagebox) en Tcl/Tk (tkinter). Recursos Python. https://recursospython.com/guias-y-manuales/cuadros-de-dialogo-messagebox-en-tkinter/

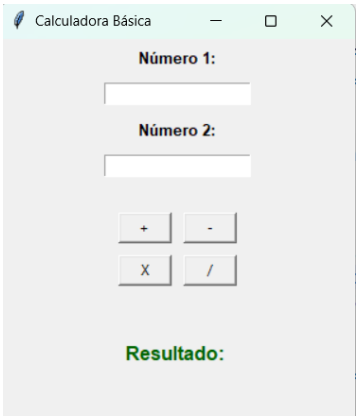
tkinter — Python interface to Tcl/Tk. (s/f). Python documentation. Recuperado el 29 de abril de 2026, de https://docs.python.org/es/3.13/library/tkinter.html

W3Resource. (s.f.). *Python tkinter events and event handling exercise 1*. W3Resource. Recuperado el 13 de mayo de 2026, de https://www.w3resource.com/python-exercises/tkinter/python-tkinter-events-and-event-handling-exercise-1.php

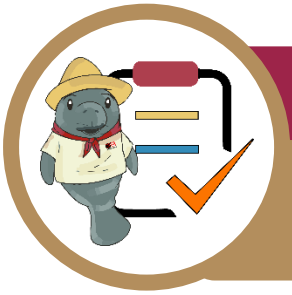
Práctica 3. Calculadora básica

Instrucciones: *Realizar la aplicación de la calculadora básica en la cual seguiremos guiando y explicando los elementos del programa por último el estudiante realizara las funciones faltantes de suma y resta*

Tomando como base la aplicación anterior debe realizar una aplicación donde se puedan realizar las operaciones básicas multiplicación, división, suma y resta donde utilizaran 4 botones que tengan como etiqueta los símbolos X, /, + y – y cada vez que el usuario proporcione los dos números puedan realizar estas operaciones dando el resultado, para que el estudiante se dé una idea se mostrara la pantalla de la aplicación y el código con dos funciones de los botones multiplicación y división, sabemos que en caso de la división el segundo número debe ser diferente de cero porque si no provocaría error de división por cero por eso se aplica un if para evitarlo:



Sugerencia de interfaz



Rubrica

Practica 2 - Calculadora básica

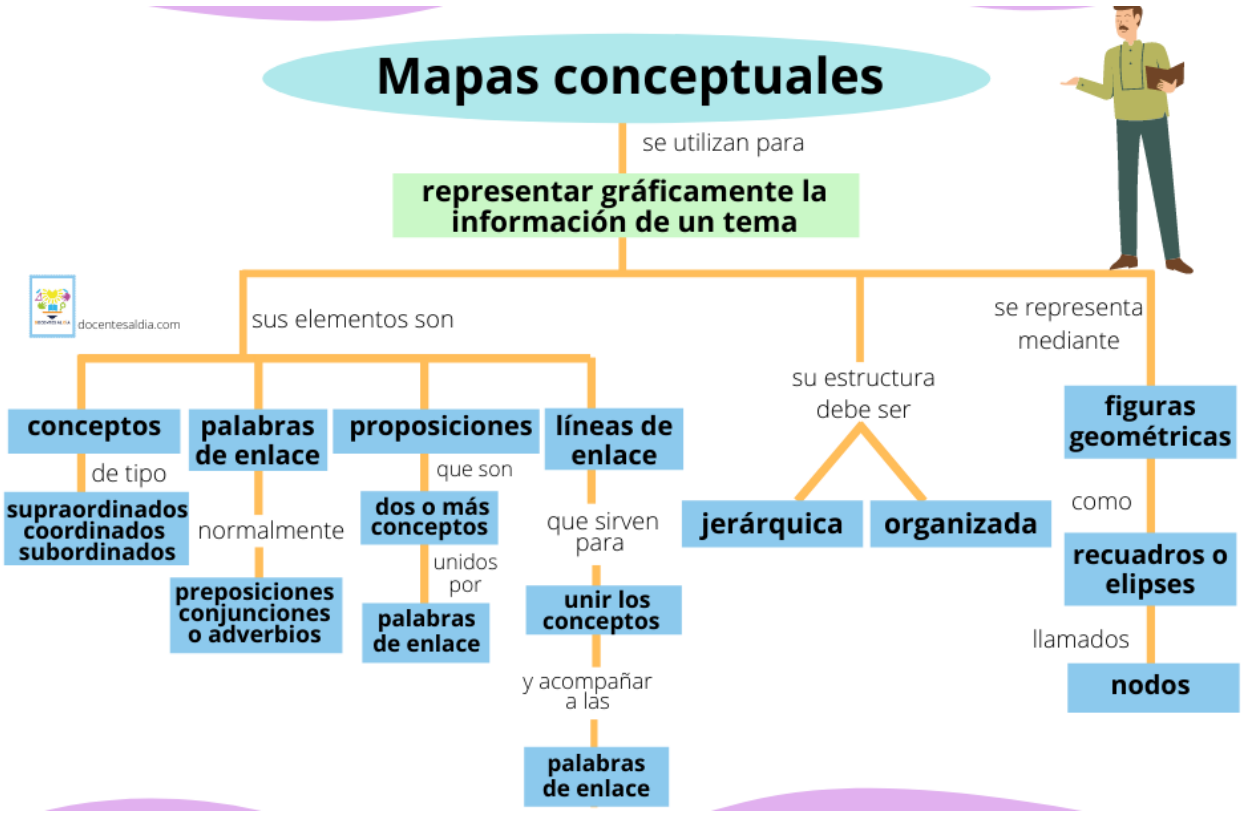
DATOS GENERALES		
Nombre(s) del alumno (s)		**Matriculas(s)**
Producto: Programa Calculadora básica		**Fecha**
UAC: Programación		**Periodo:2025-2026A**
Nombre del docente:		**Firma del docente**

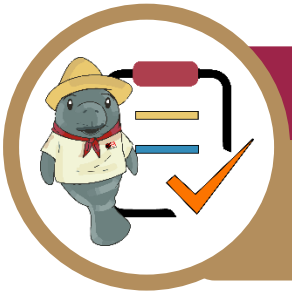
Criterio	Excelente (10)	Satisfactorio (7-9)	Insuficiente (0-6)
Funcionamiento del programa	La calculadora ejecuta correctamente las cuatro operaciones incluyendo manejo de errores	Ejecuta la mayoría de las operaciones correctamente, pero presenta errores menores o falta manejo de excepciones.	El programa no funciona adecuadamente o no realiza las operaciones básicas.
Estructura del código	El código está ordenado, indentado correctamente y usa funciones bien definidas con nombres claros.	El código tiene buena estructura general, aunque con algunos detalles de organización o nombres poco descriptivos.	El código está desordenado, sin funciones claras o con errores de sintaxis.
Interfaz gráfica (Tkinter)	La interfaz es funcional, clara y bien organizada; los botones y etiquetas están correctamente ubicados y etiquetados.	La interfaz funciona, pero presenta problemas menores de diseño o distribución.	La interfaz no se muestra correctamente o no permite realizar las operaciones.

Ejercicio teórico. Mapa conceptual uso de Tkinter

Instrucciones: *De la lectura anterior realice un mapa* conceptual *que capturen los conceptos básicos de modo gráfico en Python*

Ejemplo de mapa conceptual:





Rubrica

Practica 2 - Mensajes gráficos

DATOS GENERALES		
Nombre(s) del alumno (s)		**Matriculas(s)**
Producto: Infografía en formato tabloide		**Fecha**
UAC: Programación		**Periodo:2025-2026A**
Nombre del docente:		**Firma del docente**

Criterio	Excelente (10)	Satisfactorio (7-9)	Insuficiente (0-6)
Selección de conceptos	Incluye todos los conceptos clave de la lectura (widgets, ventana raíz, mainloop, variables de control, diálogos, ejemplos).	Incluye la mayoría de los conceptos, aunque falta alguno importante o está poco desarrollado.	Incluye pocos conceptos o irrelevantes, sin relación clara con la lectura.
Organización y jerarquía	Los conceptos están organizados de forma clara, con relaciones jerárquicas y conexiones lógicas bien definidas.	La organización es aceptable, aunque algunas relaciones son poco claras o confusas.	No hay organización ni jerarquía; los conceptos aparecen desordenados.
Claridad visual	El mapa es legible, con uso adecuado de colores, formas y conectores que facilitan la comprensión.	El mapa es entendible, pero con problemas menores de legibilidad o diseño.	El mapa es difícil de leer, sin conectores claros ni estructura visual.

Explicación y síntesis	Los conceptos están acompañados de definiciones o ejemplos breves que muestran comprensión.	Algunos conceptos tienen explicación, pero incompleta o superficial.	No hay explicaciones o son incorrectas.
Presentación final	Entrega ordenada, limpia y completa, lista para exposición o revisión.	Entrega aceptable, aunque con detalles menores de formato o incompletitud.	Entrega incompleta, desorganizada o sin acabado.

Lectura 6. Python y las bases de datos

Instrucciones: *realiza la siguiente lectura, no olvides identificar los puntos clave y repasar tantas veces consideres los ejercicios hasta que logres dominar la conexión a base de datos desde Python.*

Como habrás notado, las librerías de Python son potentes módulos que facilitan múltiples tareas, en esta ocasión veremos la implementación de SQLite con el módulo **sqlite3** que viene incluido en las librerías de Python. Esta librería permite trabajar con base de datos locales a través de un archivo de base de datos que se almacena en tu dispositivo, por lo que se sugiere tener un directorio en él, donde puedas guardarlo. Guarda tu trabajo antes de ejecutarlo para que se almacene en el directorio correspondiente y no en la carpeta temporal, ya que en donde se guarde el archivo fuente ahí se creará la base de datos, a menos que le des una ruta especifica.

Para entender de una forma clara el proceso lo haremos por serie de pasos a continuación

Paso 1: importar librería

Para hacer uso de esta librería iniciaremos con su importación, para ello usaremos la instrucción **import** de Python de la siguiente forma:

```
import sqlite3
```

Paso 2: conexión a la base de datos

La conexión a la base de datos la haremos de la siguiente forma:

```
# Conectar a la base de datos (o crearla si no existe)
conn = sqlite3.connect('mi_base_de_datos.db')
```

Donde:

conn: es el objeto que representa la conexión a la base de datos

sqlite.conect(): crea una conexión a la base de datos que se especifica, en caso de no existir, creará uno nuevo. Dentro del paréntesis va el **nombre de la base de datos** en comillas simples o dobles con la extensión **".db"**.

Paso 3: creación de tablas

Creada nuestra base de datos procederemos a la creación de las tablas mediante a función "cursor" de SQLite3 en Python, de esta forma podrás invias las instrucciones desde **cursor** a **SQL** para ello basta con hacer lo siguiente:

```
cursor =conn.cursor()
```

A partir de este momento con **cursor** enviaremos las instrucciones SQL, para ello usaremos la función **excecute**. El código Python para crear una **tabla** con el nombre "ejemplo" y las columnas "id", "nombre" y "edad" es el siguiente:

```
cursor.execute("CREATE TABLE IF NOT EXISTS ejemplo (id INTEGER, nombre TEXT, edad INTEGER)")
```

1 2 3 4

donde:

1. **cursor.execute():** función de Python que envía instrucciones SQL a la base de datos
2. **CREATE TABLE IF NOT EXISTS:** instrucción SQL que crea una tabla siempre y cuando esta no exista en la base de datos.
3. Ejemplo: nombre que se le da a la tabla de la base de datos
4. id INTEGER, nombre TEXT, edad INTEGER: nombres de las columnas de la tabla y su tipo de dato que almacena.

Siguiendo este ejemplo puedes construir dentro de tu base de datos las tablas que requieras, con sus columnas y atributos correspondientes.

Paso 4: añadir información

Otra función que puedes hacer con execute es la de añadir información a cada una de las tablas, para ello debes estructurar la sentencia SQL y enviarla con dicha función. Ejemplo:

```
cursor.execute("INSERT INTO ejemplo VALUES (1, 'Jorge', 20)")

cursor.execute("INSERT INTO ejemplo VALUES (2, 'Inés', 30)")

cursor.execute("INSERT INTO ejemplo VALUES (3, 'Carlos', 40)")
```

Si ejecutas este código ingresaras a la base de datos en la tabla **ejemplo,** recordaras que los datos tipo texto se deben colocar entre comillas ya sea simples o dobles, para que los datos queden guardados de actualiza la base de datos con la siguiente instrucción:

```
conn.commit()
```

Paso 5: cerrar a conexión

Una vez realizado todos los cambios en tu base de datos deberás cerrar la conexión para ello debes llamar la función desde tu objeto de conexión en este caso conn de la siguiente forma:

```
conn.closed()
```

Con estos simples pasos queda lista la creación de la base de datos con sus tablas y con información añadida y cierre de conexión.

Lectura de datos

Sqlite3 puede leer y extraer datos de otras bases de datos, primero debes conectarte a la base de datos que quieres y, como te hemos mostrado anteriormente, crear un objeto **conn** (conexión) y un **cursor**. Una vez establecida la conexión puedes realizar tu consulta SQL, enviar el resultado a tu base de datos con "execute" y mostrar las filas obtenidas con la función "fetchall":

```
#ejecuta la consulta SQL seleccionando todo (*) de la tabla ejemplo

cursor.execute("SELECT * FROM ejemplo")

rows = cursor.fetchall()

for row in rows:

    print(row)
```

"La función "fetchall" te muestra una lista con las filas que coinciden con tu consulta. Puedes combinar un bucle for en Python con una sentencia print para que aparezcan todas las líneas en la consola." IONOS(2023)

Eliminar datos

Para eliminar datos sigue el mismo proceso de conexión a la base de datos, una vez realizada envía la instrucción SQL con la función execute identificando los datos a eliminar, en este caso eliminará el registro que coincida con la id=1.

```
cursor.execute("DELETE FROM ejemplo WHERE id = 1")
```

Modificar datos

Utiliza el siguiente comando para cambiar la edad en la línea con ID 2:

```
cursor.execute("UPDATE ejemplo SET edad = 31 WHERE id = 2")
```

Marcadores de posición

Python puede utilizar valores y marcadores de posición para consultas dinámicas, por lo que por cuestiones de seguridad se recomienda usar marcadores de posición.

https://youtu.be/pESDsngc8as?si=UvL1tKRl9keTOiAW

Práctica guiada 1. Reserva de eventos

Instrucciones: *Usaremos lo aprendido en este curso para simular la creación de un pequeño sistema de reservas de espacios tipo sala de cines, para ello trabajaremos con conexión a base de datos. Usaremos la librería Tkinter para generar la ventana y sus opciones correspondientes*

Idea general:

Diccionario de Datos

Base de Datos

Nombre: reservaciones.db

Tabla: reservas

Campo	Tipo de dato	Longitud	Llave	Nulo	Descripción
id	INTEGER	-	Primaria	No	Identificador único de la reserva. Se incrementa automáticamente.
nombre	TEXT	100	No	No	Nombre de la persona que realiza la reservación.
evento	TEXT	100	No	No	Nombre del evento seleccionado.

Campo	Tipo de dato	Longitud	Llave	Nulo	Descripción
asiento	TEXT	5	No	No	Código del asiento reservado (Ejemplo: A1, B5, E8).

Descripción de los campos

id

- **Tipo:** INTEGER
- **Restricción:** PRIMARY KEY AUTOINCREMENT
- **Descripción:** Identificador único de cada registro.

nombre

- **Tipo:** TEXT
- **Descripción:** Nombre del cliente que realiza la reservación.

evento

- **Tipo:** TEXT
- **Descripción:** Evento para el cual se realiza la reserva. En el sistema los eventos provienen del Combobox (Avengers, Minecraft Movie, Mario Bros, Concierto y Obra teatral).

asiento

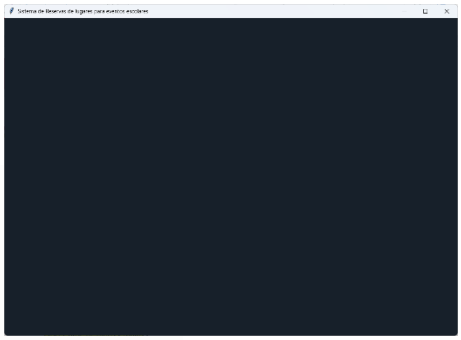
- **Tipo:** TEXT
- **Descripción:** Lugar reservado dentro de la sala.

Paso 1. Diseño del sistema

Iniciemos por crear la ventana del sistema con sus opciones:

Visualizaremos la creación de esta ventana paso a paso utilizando Tkinter en Python desde tu dispositivo móvil o computadora.

a.- Ventana general



```
import tkinter as tk
from tkinter import ttk
from tkinter import messagebox
import sqlite3


ventana = tk.Tk()
ventana.title("Sistema de Reservas de lugares para eventos escolares")
ventana.geometry("1000x700") #no funciona en celulares, por lo que se debe omitir
ventana.config(bg="#17202A")

ventana.mainloop()
```

b.- Encabezados de la ventana


```
label_titulo = tk.Label(
    ventana,
    text="SISTEMA DE RESERVAS",
    bg="#1B2631",
    fg="white",
    font=("Arial", 24, "bold"),
    pady=15
)

label_titulo.pack(fill=tk.X)
```

c.- anexamos un frame para delimitar la zona de los datos del cliente

```
frame_superior = tk.Frame(
    ventana,
    bg="#17202A"
)

frame_superior.pack(pady=10)
```

d.- Datos del cliente

creamos los elementos para solicitar los datos del cliente y opciones a elegir

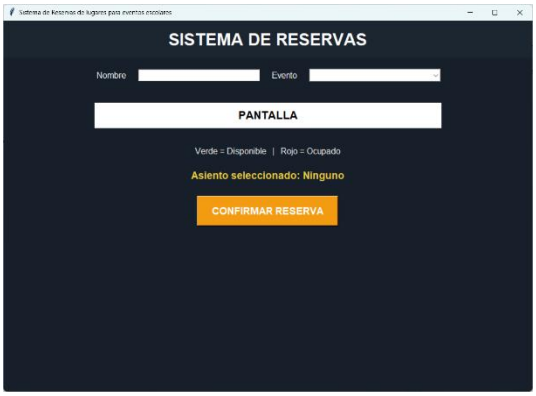
```
label_nombre = tk.Label(
    frame_superior,
    text="Nombre",
    bg="#17202A",
    fg="white",
    font=("Arial", 12)
)

label_nombre.grid(row=0, column=0, padx=10, pady=10)

entry_nombre = tk.Entry(
    frame_superior,
    width=25,
    font=("Arial", 12)
)

entry_nombre.grid(row=0, column=1, padx=10)

label_evento = tk.Label(
    frame_superior,
    text="Evento",
    bg="#17202A",
    fg="white",
    font=("Arial", 12)
)

label_evento.grid(row=0, column=2, padx=10)

combo_evento = ttk.Combobox(
    frame_superior,
    values=[
        "Avengers",
        "Minecraft Movie",
        "Mario Bros",
        "Concierto",
        "Obra teatral"
    ],
    width=25,
    font=("Arial", 12)
)
```

e.- agregamos los elementos para visualizar la sala y sus asientos

```
label_pantalla = tk.Label(
    ventana,
    text="PANTALLA",
    bg="white",
    fg="black",
    width=50,
    font=("Arial", 16, "bold"),
    pady=10
)

label_pantalla.pack(pady=20)

# ====================================
# ASIENTOS
# ====================================

frame_asientos = tk.Frame(
    ventana,
    bg="#17202A"
)

frame_asientos.pack()
```

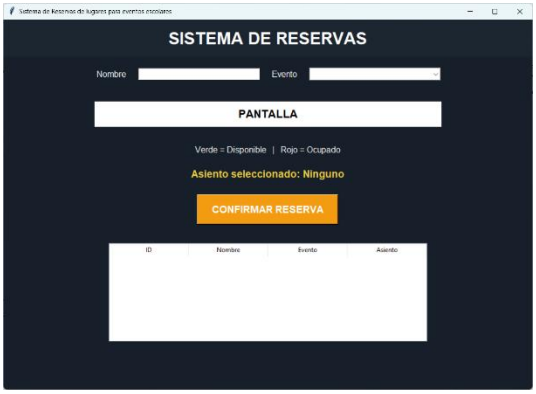
f.- creamos los elementos para mostrar la información y el asiento seleccionado

```
# ====================================
# INFORMACIÓN
# ====================================

label_info = tk.Label(
    ventana,
    text="Verde = Disponible   |   Rojo = Ocupado",
    bg="#17202A",
    fg="white",
    font=("Arial", 12)
)

label_info.pack(pady=10)

label_asiento = tk.Label(
    ventana,
    text="Asiento seleccionado: Ninguno",
    bg="#17202A",
    fg="#F4D03F",
    font=("Arial", 14, "bold")
)

label_asiento.pack(pady=10)

# ====================================
# BOTÓN RESERVAR
# ====================================

boton_reservar = tk.Button(
    ventana,
    text="CONFIRMAR RESERVA",
    bg="#F39C12",
    fg="white",
    font=("Arial", 14, "bold"),
    padx=20,
    pady=10,
    command=""
)

boton_reservar.pack(pady=15)
```



g.- agregamos una tabla (donde se mostrarán gráficamente los asientos de la sala)

```
# ====================================
# TABLA
# ====================================

columnas = (
    "ID",
    "Nombre",
    "Evento",
    "Asiento"
)

tabla = ttk.Treeview(
    ventana,
    columns=columnas,
    show="headings",
    height=8
)

for col in columnas:
    tabla.heading(col, text=col)

    tabla.column(col, width=150)


tabla.pack(pady=20)
```



Paso 2 Base de datos y tablas

Importa la librería sqlite3 y crea la base de datos "reservaciones.db**"** y su tabla "**reservas**", con sus correspondientes campos y tipos de datos:

```
# ====================================
# BASE DE DATOS
# ====================================

conexion = sqlite3.connect("reservaciones.db")
cursor = conexion.cursor()

cursor.execute('''
CREATE TABLE IF NOT EXISTS reservas(
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    nombre TEXT,
    evento TEXT,
    asiento TEXT
)
''')

conexion.commit()

# ====================================
# VENTANA PRINCIPAL
# ====================================

ventana = tk.Tk()
ventana.title("Sistema de Reservas de lugares para eventos escolares")
ventana.geometry("1000x700")
ventana.config(bg="#17202A")

# ====================================
# VARIABLES
# ====================================

asiento_seleccionado = tk.StringVar()
```

Paso 3 agrega las funciones de cada botón inmediatamente después de crear la base de datos.

```
def obtener_asientos_ocupados():
    cursor.execute("SELECT asiento FROM reservas")
    registros = cursor.fetchall()

    ocupados = []

    for fila in registros:
        ocupados.append(fila[0])

    return ocupados



def seleccionar_asiento(asiento):
    asiento_seleccionado.set(asiento)
    label_asiento.config(text=f"Asiento seleccionado: {asiento}")
```

```
def guardar_reserva():
    nombre = entry_nombre.get()
    evento = combo_evento.get()
    asiento = asiento_seleccionado.get()

    if nombre == "" or evento == "" or asiento == "":
        messagebox.showerror(
            "Error",
            "Complete todos los datos"
        )
        return

    cursor.execute(
        "SELECT * FROM reservas WHERE asiento=? AND evento=?",
        (asiento, evento)
    )

    ocupado = cursor.fetchone()

    if ocupado:
        messagebox.showwarning(
            "Ocupado",
            "Ese asiento ya está ocupado"
        )
        return

    cursor.execute(
        "INSERT INTO reservas(nombre, evento, asiento) VALUES(?,?,?)",
        (nombre, evento, asiento)
    )

    conexion.commit()

    messagebox.showinfo(
        "Correcto",
        "Reserva realizada"
    )

    pintar_asientos()
    mostrar_reservas()
```

```python
def pintar_asientos():

    ocupados = obtener_asientos_ocupados()

    for widget in frame_asientos.winfo_children():
        widget.destroy()

    filas = ["A", "B", "C", "D", "E"]

    for i, fila in enumerate(filas):

        for numero in range(1, 9):

            codigo = f"{fila}{numero}"

            if codigo in ocupados:
                color = "red"
                estado = tk.DISABLED
            else:
                color = "green"
                estado = tk.NORMAL

            boton = tk.Button(
                frame_asientos,
                text=codigo,
                width=6,
                height=2,
                bg=color,
                fg="white",
                font=("Arial", 10, "bold"),
                state=estado,
                command=lambda c=codigo: seleccionar_asiento(c)
            )

            boton.grid(
                row=i,
                column=numero,
                padx=5,
                pady=5
            )
```

```python
def mostrar_reservas():

    tabla.delete(*tabla.get_children())

    cursor.execute("SELECT * FROM reservas")

    registros = cursor.fetchall()

    for fila in registros:
        tabla.insert("", tk.END, values=fila)
```

Paso 4 vincular el botón con la función correspondiente

command="", cambia las comillas por los nombres de cada función de la siguiente forma:

```
# ====================================
# BOTÓN RESERVAR
# ====================================

boton_reservar = tk.Button(
    ventana,
    text="CONFIRMAR RESERVA",
    bg="#F39C12",
    fg="white",
    font=("Arial", 14, "bold"),
    padx=20,
    pady=10,
    command=guardar_reserva
)

boton_reservar.pack(pady=15)
```

Al finalizar ejecuta la función pintar_asientos() y mostrar_reservas(), para leer la base de datos y mostrar su contenido en tu programa.

```
# ====================================
# CARGA INICIAL
# ====================================

pintar_asientos()
mostrar_reservas()
```

Paso 5 cierra la base de datos

No olvides que al finalizar tu programa debes cerrar la conexión a la base de datos en este caso después de **ventana.mainloop()** usa la instrucción **conn.close()** de esta forma cierras la conexión a tu base de datos

La información proporcionada es una sugerencia para comprender cada uno de los pasos, puedes hacer los cambios que consideres para lograr percibir y aplicar cada método disponible para manejo de base de datos en Python.

La práctica constante te ayudará a mejorar en tus códigos y formar nuevas soluciones creativas.

Referencias

Funciones en Python. (s/f). El Libro De Python. Recuperado el 29 de julio de 2025, de https://ellibrodepython.com/funciones-en-python

Funciones incorporadas. (s/f). Python documentation. Recuperado el 29 de julio de 2025, de https://docs.python.org/es/3.13/library/functions.html

Python, R. (2018, agosto 14). Cuadros de diálogo (messagebox) en Tcl/Tk (tkinter). Recursos Python. https://recursospython.com/guias-y-manuales/cuadros-de-dialogo-messagebox-en-tkinter/

tkinter — Python interface to Tcl/Tk. (s/f). Python documentation. Recuperado el 29 de julio de 2025, de https://docs.python.org/es/3.13/library/tkinter.html

Submódulo 2. Situación didáctica "Haz que tu código cuente" |

Instrucciones: *analiza la información proporcionada en cada lectura, ordena tus actividades del submódulo 1 y 2 para que con ella realices una aplicación en Python que dé solución a la situación "Haz que tu código cuente" prevista al inicio de este submódulo*

Resumen

En esta tarea, trabajará en colaboración con su equipo para crear una aplicación de Python que aborde la situación "Haga que su código cuente" presentada al principio de este submódulo. Este proyecto requerirá que analice lecturas, organice sus actividades y codifique una solución basada en la información proporcionada.

Instrucciones

Paso 1: Forma tu equipo

- **Composición del equipo**: Organícense en un equipo de 5 miembros. Asegúrese de elegir miembros del equipo que estén dispuestos a colaborar y contribuir por igual al proyecto.
- **Roles y responsabilidades**: Discuta y asigne roles dentro de su equipo. Considere roles como:
    - Gerente de proyecto: supervisa el cronograma del proyecto y se asegura de que todos estén encaminados.
    - Investigador: Analiza las lecturas y recopila la información necesaria.
    - Desarrollador: Se centra en codificar la aplicación.
    - Tester: Responsable de probar la aplicación en busca de errores y funcionalidad.
    - Documentador: toma notas y prepara la presentación final.

Paso 2: Analice las lecturas

- **Revisar contenido**: Lea atentamente los materiales proporcionados. Preste atención a los conceptos clave, ejemplos y cualquier instrucción específica relacionada con la situación "Haga que su código cuente".

- **Tome notas**: Mientras lee, anote los puntos, preguntas e ideas importantes que surjan. Esto te ayudará durante las discusiones con tu equipo.
- **Discuta los hallazgos**: Realice una reunión de equipo para compartir sus ideas. Discuta cómo se puede aplicar la información a su proyecto y qué soluciones podrían ser efectivas.

Paso 3: Organizar actividades

- **Actividades del submódulo**: Revise las actividades del submódulo 1 y 2. Cree una lista de tareas que deben completarse para desarrollar la aplicación.
- **Priorizar tareas**: Ordena las actividades en función de su importancia y la secuencia en la que deben completarse. Considere las dependencias entre tareas (por ejemplo, es posible que deba completar la investigación antes de codificar).
- **Crea una línea de tiempo**: Desarrolla una línea de tiempo para tu proyecto. Establezca plazos para cada tarea para asegurarse de que su equipo se mantenga encaminado.

Paso 4: Revisar el diccionario de datos

- **Comprender las relaciones**: Examine el diccionario de datos proporcionado. Concéntrese en las relaciones que se establecen dentro de él, ya que serán cruciales para su aplicación.
- **Identificar elementos clave**: Resalte los elementos de datos clave que su aplicación necesitará para funcionar de manera efectiva. Discuta cómo estos elementos se relacionan con la situación "Haga que su código cuente".

Paso 5: Codificar la solución

- **Codificación colaborativa**: Comience a codificar su aplicación como equipo. Usa una plataforma colaborativa (como GitHub) para compartir tu código y trabajar juntos en tiempo real.
- **Siga las mejores prácticas**: Asegúrese de que su código esté bien organizado y siga las mejores prácticas de Python. Esto incluye el uso de nombres de variables significativos, la escritura de comentarios y la estructuración lógica del código.
- **Iterar y mejorar**: A medida que codifica, pruebe continuamente su aplicación. Realice mejoras basadas en los comentarios de los miembros del equipo y cualquier problema que surja.

Paso 6: Pruebas

- **Realice pruebas exhaustivas**: Una vez que su aplicación esté codificada, realice pruebas exhaustivas para identificar errores o problemas. Cada miembro del equipo debe probar diferentes partes de la aplicación.

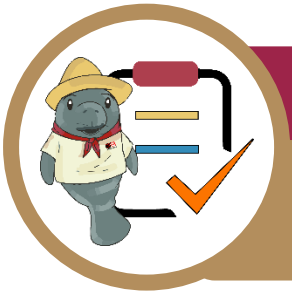
- **Documentar problemas**: Mantenga un registro de los problemas encontrados y cómo se resolvieron. Esta documentación será útil para su envío final.

Paso 7: Preparar la presentación

- **Requisitos de formato**: Siga el formato de envío indicado por su maestro. Asegúrese de que todos los componentes de su proyecto estén incluidos.
- **Revisión final**: Antes de enviar, revise su solicitud y documentación en equipo. Asegúrese de que todo sea claro, completo y bien presentado.
- **Enviar a tiempo**: Asegúrese de que su envío esté completo y enviado antes de la fecha límite establecida por su maestro.

Conclusión

Siguiendo estos pasos detallados, usted y su equipo podrán crear una aplicación Python integral que aborde de manera efectiva la situación "Haz que tu código cuente". Recuerde comunicarse abiertamente con los miembros de su equipo y apoyarse mutuamente durante todo el proceso. ¡Buena suerte!



INSTRUMENTO DE EVALUACIÓN -

Rubrica

Situación Didáctica II "Has que tu código cuente"

DATOS GENERALES		
Nombre(s) del alumno (s)		**Matriculas(s)**
Producto: Aplicación		**Fecha**
UAC: Programación		**Periodo:2025-2026A**
Nombre del docente:		**Firma del docente**

Criterio	Excelente (10)	Bueno (8)	Satisfactorio (6)	Insuficiente (5 o menos)
Diseño de interfaz con Tkinter	La interfaz es clara, intuitiva, organizada y visualmente atractiva. Usa adecuadamente botones, menús y cuadros de texto.	La interfaz es funcional y entendible, aunque con algunos detalles visuales o de distribución.	La interfaz cumple con lo básico, pero es poco estética o difícil de navegar.	La interfaz es confusa, poco funcional o presenta errores.
Conexión y uso de base de datos SQLite3	Implementa correctamente la creación, conexión y manipulación de la base de datos (CRUD).	Presenta una conexión funcional y realiza operaciones básicas (inserción y consulta).	Usa la base de datos, pero con errores o sin todas las funciones requeridas.	No implementa o la base de datos no funciona correctamente.
Registro y análisis de ventas	El sistema permite registrar ventas, genera reportes claros y muestra	Permite registrar ventas y muestra información básica de	El registro funciona, pero no genera reportes o estadísticas.	No se implementa correctamente el registro de ventas.

Criterio	Excelente (10)	Bueno (8)	Satisfactorio (6)	Insuficiente (5 o menos)
	productos más vendidos.	productos vendidos.		
Organización del código y uso de funciones	El código está modularizado (uso correcto de funciones), comentado y organizado.	El código está bien estructurado, pero con pocos comentarios o funciones.	El código es entendible, pero presenta desorden o exceso de código repetido.	El código es confuso, sin funciones ni comentarios claros.
Cumplimiento de requerimientos	Cumple con todos los puntos: análisis de preferencias, informes, inventario y toma de decisiones.	Cumple con la mayoría de los requerimientos funcionales del sistema.	Solo cumple parcialmente con los requisitos propuestos.	No cumple con los requerimientos mínimos del sistema.
Documentación del proyecto	Incluye una guía o informe detallado del funcionamiento, estructura del sistema y análisis de factibilidad.	Entrega un informe con los puntos básicos del sistema.	Solo presenta una descripción breve o poco clara del proyecto.	No entrega documentación o es incompleta.
Presentación del proyecto	Explica con claridad, demuestra dominio del proyecto y responde preguntas con seguridad.	Explica el funcionamiento y responde a la mayoría de las preguntas.	La explicación es confusa o incompleta.	No explica adecuadamente o no demuestra conocimiento del sistema.

**Puntaje total: ** _____ /70

**Nivel de desempeño: **
- 61–70 = Excelente
- 51–60 = Bueno
- 41–50 = Satisfactorio
- 40 o menos = Insuficiente

**Observaciones del docente: ** _____

Anexos

Actividad de reforzamiento 1: "El Reto del Sistema de Información"

Tiempo 50 min.

Objetivo:
Que los estudiantes identifiquen, comprendan y relacionen los conceptos, componentes e importancia de los sistemas de información mediante un juego colaborativo.

Desarrollo del juego

1. Formación de equipos
- Dividir a la clase en 3 o 4 equipos.
- Cada equipo recibe 1 juego de tarjetas mezcladas (conceptos y descripciones).

2. Desafío de encontrar pareja (10 min)
- Los equipos deben **emparejar correctamente** cada concepto con su definición o ejemplo (Se anexa sugerencias de conceptos y descripciones de conceptos básicos de base de datos)
- Gana puntos el equipo que termine primero y tenga más aciertos.

3. Ronda de preguntas rápidas (10 min)
- La maestra lanza **preguntas tipo reto** relacionadas con la teoría, por ejemplo:
  - "Menciona un ejemplo de sistema de información en educación"
  - "¿Qué componente corresponde a los usuarios y operadores?"
  - "¿Qué es un CRM y para qué sirve?"
- Cada respuesta correcta = 1 punto extra.

4. Reto final: "Construye tu propio SI" (10 min)
- Cada equipo inventa un ejemplo de sistema de información para una situación específica (puede ser una tienda, hospital, escuela, etc.).
- Deben explicar:
  - Su **objetivo**
  - **Componentes** que incluiría
  - **Tipo de datos** que manejaría
  - **Beneficio principal**
- Presentan su propuesta en 1 minuto.

• Concepto: Sistema de Información	• Conjunto de tecnología, personas y procesos para recopilar, procesar, almacenar y distribuir información.
Concepto: Hardware	Computadoras, servidores, dispositivos de red.
• Concepto: Software	• Programas y aplicaciones que procesan datos.
Concepto: Datos	• Información y hechos que el sistema recopila y utiliza.

• Concepto: Procedimientos	• Normas y métodos que regulan el uso del sistema.
Concepto: Personas	• Usuarios, administradores y personal técnico.
• Concepto: Entrada de datos	• Captura de datos desde diferentes fuentes.
Concepto: Procesamiento	• Transformación de datos en información útil.

• Concepto: Salida de información	• Distribución de resultados o reportes generados.
Concepto: Eficiencia operativa	• Automatizar tareas y reducir errores para mejorar productividad.
•	
• Concepto: Toma de decisiones	• Uso de información en tiempo real para decisiones acertadas.

Capacidad de responder rápido al mercado y mejorar la relación con clientes.	• Concepto: Ventaja competitiva
• Concepto: CRM	• Sistema para gestionar relaciones con clientes y personalizar marketing.
Moodle o Blackboard para clases virtuales.	• Concepto: Plataforma educativa
• Concepto: Registro médico electrónico	• Base de datos centralizada de pacientes para telemedicina.

• Repositorio central de información que puede ser usada por distintos usuarios.	Concepto: Base de datos
• Concepto: DBMS	• Sistema para crear, modificar, consultar y mantener datos.
La BD organiza datos y el SI los usa para generar conocimiento.	• Concepto: Interdependencia SI-BD

Actividad de reforzamiento 2 "Diseño de ventanas Tkinter"

Tiempo 50 minutos

Objetivo de aprendizaje

Que el alumno comprenda y aplique los conceptos básicos para crear y configurar ventanas en Tkinter, incluyendo:
- Creación de una ventana principal.
- Configuración de tamaño, título y color de fondo.
- Uso de Label, Button y Entry.
- Uso de ventanas secundarias (Toplevel) para mostrar información.

Instrucciones para el alumno

1. Crea un nuevo archivo en Python llamado ventanas_tkinter.py.

2. Importa la librería Tkinter: import tkinter as tk y from tkinter import messagebox.

3. Crea la ventana principal con título 'Mi primera ventana en Tkinter', tamaño 400x300 y fondo a elección.

4. Agrega un Label con tu nombre completo.

5. Agrega un Entry para que el usuario escriba un mensaje.

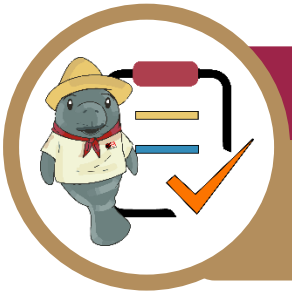
6. Agrega un Button llamado 'Mostrar mensaje' que abra una ventana secundaria (Toplevel) con el mensaje del Entry y un botón para cerrarla.

7. Guarda y ejecuta el programa para verificar su funcionamiento.

Ejemplo de referencia (no copies tal cual, captura y realiza los cambios que consideres pertinentes para hacer un trabajo original)

Nota. No olvides comentar tu código para recordar la funcionalidad de cada segmento.

```
import tkinter as tk
from tkinter import messagebox

root = tk.Tk()
root.title("Mi primera ventana en Tkinter")
root.geometry("400x300")
root.configure(bg="#cce7ff")

lbl_nombre = tk.Label(root, text="Juan Pérez", font=("Arial", 14), bg="#cce7ff")
lbl_nombre.pack(pady=10)

entrada = tk.Entry(root, width=30)
entrada.pack(pady=10)

def mostrar_mensaje():
    mensaje = entrada.get()
    if mensaje.strip() == "":
        messagebox.showwarning("Aviso", "Por favor escribe un mensaje.")
    else:
        ventana_secundaria = tk.Toplevel(root)
        ventana_secundaria.title("Mensaje")
        ventana_secundaria.geometry("300x200")
        tk.Label(ventana_secundaria, text=mensaje, font=("Arial", 12)).pack(pady=20)
        tk.Button(ventana_secundaria, text="Cerrar", command=ventana_secundaria.destroy).pack(pady=10)

btn_mostrar = tk.Button(root, text="Mostrar mensaje", command=mostrar_mensaje)
btn_mostrar.pack(pady=10)

root.mainloop()
```



INSTRUMENTO DE EVALUACIÓN -

Rubrica

"Diseño de ventanas Tkinter"

DATOS GENERALES				
Nombre(s) del alumno (s)			**Matriculas(s)**	
Producto: Aplicación			**Fecha**	
UAC: Programación			**Periodo:2025-2026A**	
Nombre del docente:			**Firma del docente**	

Criterio	Excelente (4 pts)	Bueno (3 pts)	Regular (2 pts)	Deficiente (1 pt)
Ventana principal	Tamaño, título y color configurados correctamente	2 de 3 elementos configurados	1 de 3 elementos configurados	No configuró correctamente la ventana
Label con nombre	Nombre completo bien colocado y visible	Nombre incompleto o con errores menores	Nombre poco legible o mal ubicado	No agregó el Label
Entry funcional	Entrada permite capturar texto correctamente	Entrada presente, pero con problemas menores	Entrada con fallos de funcionamiento	No agregó Entry
Botón funcional	Botón abre ventana secundaria y muestra mensaje correctamente	Botón abre ventana, pero con errores menores	Botón abre ventana pero no muestra mensaje	No agregó botón funcional
Ventana secundaria	Correcta creación, muestra mensaje y permite cerrar	Muestra mensaje, pero no cierra correctamente	Cierra pero no muestra mensaje	No abre ventana secundaria

Puntaje máximo: 20 puntos
Escala:
- 18–20: Excelente
- 14–17: Bueno
- 10–13: Regular
- <10: Deficiente

HIMNO DEL COBATAB

¡Oh!, Colegio de Bachilleres, impetuosa y querida
institución casa fiel del conocimiento,
hoy te canto este himno con amor,
eres rayo de esperanza del mañana,
eres la voz de la verdad.

¡Oh!, Colegio de Bachilleres, eres
luz en medio de la oscuridad.
//Colegio de bachilleres conducta
clara y firme decisión
Colegio de bachilleres tu misión
para siempre es ser mejor.//

En Tabasco se ha sembrado
la semilla que un día germinara,
el impulso de la vida modernista
en progreso de toda la sociedad,
es tu memorable historia gran
orgullo para toda la región,
educación que genera cambio,
ejemplo digno en cada generación.

PORRA INSTITUCIONAL

¡Somos!
¡Somos!

Jóvenes Bachilleres
Jóvenes Bachilleres

Con Valor y Lealtad
De Norte a Sur
De Este a Oeste

Somos líderes Bachilleres del Sureste
COBATAB Unido, COBATAB Fortalecido.
Este encuentro lo gano porque lo gano
Como dijo el Peje, me canso ganso.

¡Somos!
¡Somos!

Jóvenes Bachilleres
Jóvenes Bachilleres

¡Somos!
¡Somos!

Jóvenes Bachilleres
Jóvenes Bachilleres

COBATAB Unido, COBATAB Fortalecido

COBACHITO

Colegio de Bachilleres,
Está de fiesta señores
Pues todos sus estudiantes
Hoy celebran con honores.

Que ya llegó la alegría
Es hora de motivar
Bailemos con algarabía
Cobachito nos guiará.

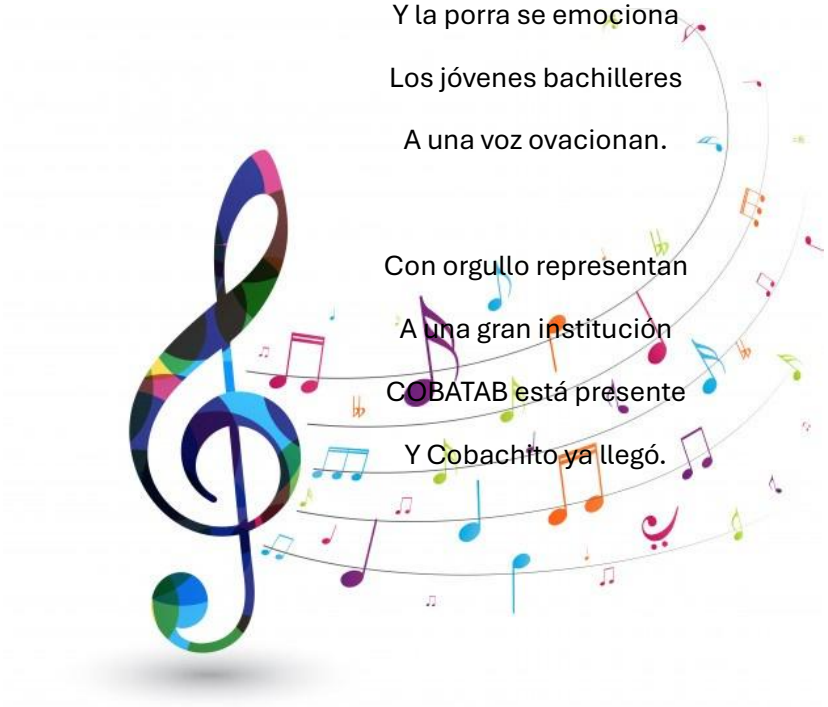
Allá por el acahual
En los ríos de Tabasco
Aconchado en unas ramas
O nadando sin parar.

Un manatí se ha ganado
El cariño de la gente
Cobachito le han llamado
Y no para de bailar.

Cobachito, con él vamos a ganar
Cobachito, eres espectacular
Cobachito, respetamos tu hábitat

Cobachito, mascota del COBATAB.

Mientras la orquesta se escucha
Y la porra se emociona
Los jóvenes bachilleres
A una voz ovacionan.



Con orgullo representan
A una gran institución
COBATAB está presente
Y Cobachito ya llegó.