Trabajo Final Integrador (TFI) - Bases de Datos I

Introducción

Este Trabajo Final Integrador tiene como finalidad consolidar los contenidos de la asignatura **Bases de Datos I** a través de un proyecto único y progresivo, articulado con el desarrollo de un **CRUD en Java** que los estudiantes realizan en la materia Programación.

Se busca simular un entorno real de gestión de datos, se incorporan actividades que simulan el trabajo con bases de datos con volúmenes importantes de datos (miles de registros), consultas optimizadas, seguridad aplicada y manejo de concurrencia, integrando aspectos de diseño, carga de información, consultas, seguridad, integridad y concurrencia.

Se promueve además el uso responsable de herramientas de **Inteligencia Artificial (IA)** como apoyo en el proceso de aprendizaje. Los estudiantes podrán valerse de IA generativa y asistentes inteligentes como **ChatGPT (OpenAI)**, **Gemini (Google DeepMind)**, **Claude (Anthropic)** o **GitHub Copilot (Microsoft)**, entre otras. El rol de estas herramientas será de **tutoría y acompañamiento**, ayudando a formular consultas, corregir razonamientos y explorar alternativas de solución, sin reemplazar el trabajo crítico del estudiante.

Se organiza en **dos partes complementarias**:

1. **Uso pedagógico de la Inteligencia Artificial (IA):** se presenta una reflexión y guía de trabajo sobre cómo integrar herramientas de IA como recurso de apoyo al aprendizaje, entendiendo su rol actual en el mundo profesional de la programación y su potencial pedagógico en la formación académica.
2. **Desarrollo del Trabajo Final Integrador:** se detallan las consignas que los estudiantes deberán realizar, organizadas en cinco etapas progresivas que abarcan el modelado, la implementación, la carga masiva de datos, las consultas avanzadas, la seguridad y la concurrencia en bases de datos.

Uso pedagógico de la Inteligencia Artificial (IA)

Introducción

Hoy en día, la **Inteligencia Artificial (IA)** está presente de forma ubicua en el trabajo cotidiano de los programadores. Desde asistentes integrados en los entornos de desarrollo hasta sistemas capaces de generar código, la IA se ha convertido en una herramienta habitual de apoyo técnico y creativo.

Frente a este panorama, resulta fundamental integrarla también en el **ámbito pedagógico**, no como un atajo que entregue soluciones finales, sino como un **aliado en el proceso de aprendizaje**. El objetivo es que los estudiantes aprendan a **dialogar con la IA a través de prompts bien diseñados**, obteniendo pistas, explicaciones y guías que los ayuden a construir por sí mismos las soluciones, en lugar de recibirlas terminadas.

De esta manera, se fomenta la autonomía, el pensamiento crítico y la capacidad de utilizar la IA como un recurso formativo en lugar de depender de ella de manera pasiva.

Objetivos

- Guiar al alumno en la resolución asistida y documentada.
- Evaluar el proceso además del resultado.
- Reducir la carga de corrección de los entregables relacionados con IA (naturalmente extensos) usando IA también para resumir y extraer evidencias.

Metodología

1. El alumno copia y pega un prompt inicial con la consigna.
2. Trabaja de forma iterativa con la IA, haciendo preguntas, corrigiendo y refinando soluciones mientras hace pruebas en los entornos correspondientes.
3. Entrega como evidencia el link o PDF del chat completo.

Ejemplo breve de prompt:

"Estoy desarrollando un script SQL para generar 100.000 registros ficticios en la tabla ventas. Estoy trabajando en MySQL. Necesito que me guíes paso a paso, sin darme todo el código de golpe."

*(El prompt completo inicial sugerido figura en el **Anexo I** al final de este documento.)*

Entregables

- Link o PDF del chat como registro del proceso.
- Producto final funcional (consulta, script, código Java, etc.).

Nota sobre el uso de IA en las actividades mínimas

Las actividades mínimas de cada etapa incluyen instancias explícitas de interacción con herramientas de Inteligencia Artificial, en coherencia con el *Prompt General* detallado en el Anexo I. Estas interacciones no reemplazan el trabajo del estudiante, sino que funcionan como tutoría pedagógica: la IA debe guiar con pistas, sugerencias y correcciones parciales, mientras el estudiante mantiene un rol activo en el diseño, la prueba y la validación de las soluciones. La entrega final debe incorporar como evidencia los fragmentos de chat (link o PDF) que muestren este proceso reflexivo de acompañamiento.

Beneficios pedagógicos

- Potencia el aprendizaje guiado y fomenta la autonomía.
- Documenta decisiones y correcciones realizadas durante el proceso.
- Facilita la evaluación basada en evidencias reales.
- Prepara al estudiante para un uso profesional y crítico de la IA.

Desarrollo del Trabajo Final Integrador

Resumen Etapas

Etapa 1 – Modelado y Definición de Constraints: Elaborar el DER y el modelo relacional con PRIMARY KEY, FOREIGN KEY, UNIQUE, CHECK y restricciones de dominio.

Etapa 2 – Implementación y Carga Masiva de Datos: Generación de datos masivos (10.000 a 100.000 registros) con SQL puro.

Etapa 3 – Consultas Avanzadas y Reportes: Desarrollo de consultas complejas con JOIN, GROUP BY, HAVING, subconsultas y vistas.

Etapa 4 – Seguridad e Integridad: Creación de usuarios con privilegios mínimos, vistas para ocultar información sensible, validación de restricciones y consultas seguras en Java con PreparedStatement.

Etapa 5 – Concurrencia y Transacciones: Simulación de bloqueos y deadlocks, comparación de niveles de aislamiento, manejo de transacciones en Java.

Beneficios pedagógicos

- Integración transversal entre Programación y Bases de Datos.
- Consolidación de SQL avanzado y constraints.
- Exposición a escenarios de seguridad, integridad y concurrencia.
- Desarrollo de competencias críticas en entornos realistas.
- Fomento del uso responsable y reflexivo de IA como tutoría de aprendizaje.

ETAPAS DETALLADAS

Etapa 1 – Modelado y Definición de Constraints

Descripción

Partiendo del enunciado del CRUD en Java (materia Programación), se debe realizar el modelo entidad–relación y el correspondiente modelo relacional, incorporando todas las restricciones de integridad necesarias:

- Primary Key (PK)
- Foreign Key (FK)
- UNIQUE
- CHECK
- Restricciones de dominio (tipos de datos, longitudes, obligatoriedad de nulos, rangos de valores).

Esto asegura que la base de datos no solo esté correctamente diseñada en lo conceptual, sino también validada en lo físico antes de comenzar con la carga masiva de datos.

Objetivos

- Comprender el vínculo entre el modelado conceptual y la implementación física.
- Aplicar constraints para garantizar la integridad de los datos.
- Asegurar la consistencia en la base de datos antes de la carga masiva.

Metodología

1. El alumno parte del modelo de dominio definido en Programación.
2. Identifica las entidades y relaciones principales.
3. Define para cada tabla los atributos y sus restricciones de dominio (tipo, tamaño, obligatoriedad).
4. Incorpora constraints (PK, FK, UNIQUE, CHECK).
5. Prueba la robustez del diseño intentando **inserciones válidas e inválidas**, analizando los mensajes de error.
6. Interactúa con la IA como tutor pedagógico, pidiendo pistas para:
    - Validar que las claves elegidas sean correctas.

- Revisar si hay redundancias o inconsistencias.
- Chequear si los dominios de atributos son apropiados.

Actividades mínimas

- El DER podrá tener 2 entidades (las del TFI de PROG II en relación 1→1 unidireccional) o más si el equipo incorpora una tercera entidad auxiliar coherente con el dominio (catálogos/maestras), sin alterar la 1→1.
- Construir el modelo relacional en SQL con PK, FK, UNIQUE, CHECK y restricciones de dominio.
- Validación práctica: ejecutar 2 inserciones correctas y 2 erróneas mostrando mensajes de error distintos (ej. violación de UNIQUE y CHECK o FK).
- Interacción con IA: consultar a la IA sobre la elección de claves y restricciones (ej. "¿mi elección de PK es correcta o convendría otra?") y adjuntar la captura/resumen de esa interacción como evidencia.

Entregables

- DER en PDF o imagen.
- Script SQL con creación de tablas y constraints.
- Evidencia de IA: link al chat completo o PDF con capturas de la interacción.

Beneficios pedagógicos

- Permite reflexionar sobre cada decisión de modelado y su impacto en la integridad.
- Refuerza la capacidad de abstraer un problema del dominio y llevarlo a un modelo entidad–relación.
- Desarrolla la habilidad de definir claves y restricciones correctas, esenciales en cualquier proyecto de bases de datos.
- Ayuda a comprender el impacto de las restricciones en la consistencia y robustez de un sistema real.
- Fomenta el hábito de validar modelos con ejemplos prácticos, una práctica profesional crítica.

Etapa 2. Generación de datos masivos con SQL puro

Descripción

Ampliar el proyecto con datos ficticios en gran volumen (miles o cientos de miles de registros) generados usando únicamente SQL, sin recurrir a otros lenguajes. Esto permitirá realizar pruebas de rendimiento, evaluar índices y trabajar con escenarios más cercanos a la realidad, ya que la mayoría de los CRUD académicos se prueban con muy pocos registros.

Objetivos

- Comprender cómo poblar bases de datos solo con SQL.
- Evaluar el impacto del volumen de datos en tiempos de respuesta.
- Experimentar con la creación y el uso de índices.

Metodología

1. Uso de funciones de MySQL (RAND(), CONCAT(), fechas aleatorias) y sentencias INSERT ... SELECT.
2. Generación escalonada: empezar con 10.000 y llegar a >100.000 registros.
3. Medición de tiempos con y sin índices.

Actividades mínimas

- Generar con SQL puro al menos 10.000 registros distribuidos en 2 o más tablas, usando funciones nativas (RAND(), CONCAT(), etc.).
- Crear y probar al menos 1 índice adicional, midiendo la diferencia de tiempos de ejecución con y sin índice.
- Entregar capturas de resultados y una breve conclusión (5–10 líneas).
- Interacción con IA: pedir apoyo para diseñar la estrategia de generación masiva de datos y documentar la respuesta usada como guía.

Entregables

- Script SQL para la generación masiva.
- Resultados de las mediciones con/sin índices.
- Evidencia del proceso (chat + producto final).

Beneficios pedagógicos

- Escenarios más realistas para pruebas.
- Comprensión del papel de los índices.
- Entrenamiento en generación de datos para pruebas y QA.

Etapa 3. Consultas complejas y útiles a partir del CRUD inicial

Descripción

Partiendo del CRUD desarrollado en Programación con JDBC, en Base de Datos I los alumnos deberán diseñar consultas SQL complejas y útiles que agreguen valor al sistema. Por ejemplo: reportes agrupados, análisis temporales, filtros avanzados o combinaciones de tablas mediante JOIN.

Objetivos

- Pasar de consultas simples a consultas multi-tabla y agregadas.
- Diseñar consultas que tengan utilidad práctica en el sistema.

- Entender el impacto de la estructura de datos y los índices en consultas complejas.

Metodología

1. Identificar áreas del CRUD donde un reporte agregue valor.
2. Diseñar consultas con JOIN, GROUP BY, HAVING, subconsultas y vistas.
3. Probar con grandes volúmenes de datos.
4. Opcional: optimizar y medir tiempos con/sin índices.

Actividades mínimas

- Diseñar 4 consultas: al menos 2 con JOIN (entre A↔B y/o con tablas auxiliares del dominio), 1 con GROUP BY + HAVING y 1 con subconsulta. En consultas de 'producción', filtrar eliminado = false.
- Crear al menos 1 vista útil en el contexto del sistema.
- Documentar brevemente (3–5 líneas) la utilidad práctica de cada consulta.
- Interacción con IA: plantear una consulta compleja o a optimizar y mostrar cómo la IA ayudó a mejorarla o a validarla.

Entregables

- Scripts SQL de las consultas.
- Capturas de resultados.
- Breve explicación de utilidad y lógica de cada consulta.

Beneficios pedagógicos

- Enfoque en resolución de problemas reales.
- Mejora de la capacidad de pensar en términos de datos.
- Consolidación de conocimientos de SQL intermedio/avanzado.

Etapa 4. Seguridad e integridad

Descripción

Implementar medidas de seguridad en el acceso a datos y garantizar la integridad de la base. Esto implica aplicar mínimos privilegios, usar vistas para exponer datos, evitar SQL injection mediante consultas parametrizadas y demostrar que las restricciones de integridad se cumplen.

Objetivos

- Aplicar la regla de usuario con mínimos privilegios.
- Usar vistas para filtrar y simplificar acceso.
- Respetar integridad referencial y unicidad.
- Aplicar seguridad en el código Java.

Metodología

1. Creación de un usuario con permisos acotados.

2. Vistas que oculten información sensible.
3. Pruebas de restricciones PK, FK, UNIQUE, CHECK.
4. En Java, uso de PreparedStatement y validaciones.

Actividades mínimas

- Crear un usuario con privilegios mínimos y mostrar pruebas de acceso restringido.
- Diseñar 2 vistas que oculten información sensible.
- Ejecutar al menos 2 pruebas de integridad (ej. duplicación de PK, inserción fuera de rango, violación de FK).
- Implementar en Java al menos 1 consulta parametrizada con PreparedStatement para prevenir SQL Injection.
- Interacción con IA: consultar sobre riesgos de seguridad o buenas prácticas y adjuntar evidencia de cómo se aplicaron las sugerencias.

Entregables

- Script SQL con creación de usuario, permisos y vistas.
- Pruebas de integridad.
- Código Java seguro.
- Evidencia del proceso de interacción con la IA (link o pdf)

Beneficios pedagógicos

- Comprensión práctica de la seguridad en bases de datos.
- Aprendizaje del uso real de vistas.
- Reforzamiento de buenas prácticas en desarrollo seguro.

Etapa 5. Concurrencia y transacciones

Descripción

Diseñar y ejecutar pruebas que muestren el comportamiento de la base ante accesos simultáneos, implementando transacciones y evaluando bloqueos, deadlocks y niveles de aislamiento.

Objetivos

- Implementar transacciones en operaciones críticas.
- Comparar niveles de aislamiento en MySQL.
- Diagnosticar y manejar bloqueos y deadlocks.
- Medir impacto de índices en entornos concurrentes.

Metodología

1. Simulación con dos sesiones MySQL para generar bloqueos y deadlocks.
2. Comparación de READ COMMITTED y REPEATABLE READ.
3. Uso opcional de mysqlslap para carga concurrente.

4. Implementación de retry ante deadlock en Java.

Actividades mínimas

- Simular al menos 1 deadlock en dos sesiones y documentar el error generado.
- Implementar en Java un ejemplo de transacciones con BEGIN, COMMIT y ROLLBACK.
- Comparar en la práctica 2 niveles de aislamiento (ej. READ COMMITTED y REPEATABLE READ) mostrando diferencias con ejemplos simples.
- Elaborar un breve informe (5–10 líneas) con las observaciones sobre concurrencia y transacciones.
- Interacción con IA: usarla como apoyo para interpretar los resultados de las pruebas de concurrencia o sugerir pasos adicionales, dejando evidencia del intercambio.

Entregables

- Guion de pruebas SQL.
- Resultados de mediciones.
- Código Java con transacciones y manejo de errores.
- Evidencia del proceso.

Beneficios pedagógicos

- Vivencia directa de problemas de concurrencia.
- Comprensión de la importancia de índices y diseño.
- Mejora en el manejo de transacciones y recuperación de errores.

RUBRICA DE EVALUACIÓN

Etapa	Peso	Criterios de evaluación
Etapa 1: Modelado y Constraints	20%	- DER con al menos 3 entidades, atributos y relaciones bien cardinalizadas. - Modelo relacional en SQL con PK, FK, UNIQUE, CHECK y dominios. - Validación con 2 inserciones correctas y 2 erróneas que disparen errores distintos. - Evidencia de interacción con IA sobre claves y restricciones. - **IA (5%)**: uso reflexivo, no copia literal.
Etapa 2: Implementación y Carga Masiva	20%	- Generación de al menos 10.000 registros con SQL puro. - Creación y prueba de al menos 1 índice. - Medición comparativa con/sin índice + conclusión. - Evidencia de interacción con IA sobre estrategia de generación de datos. - **IA (5%)**: aplicación práctica de la sugerencia recibida.
Etapa 3: Consultas Avanzadas y Reportes	25%	- 5 consultas: 3 JOIN, 1 GROUP BY/HAVING, 1 subconsulta. - Creación de al menos 1 vista útil. - Documentación breve de la utilidad de cada consulta. - Evidencia de interacción con IA para optimización o diseño de consulta. - **IA (5%)**: calidad de la reflexión, no mera copia.
Etapa 4: Seguridad e Integridad	15%	- Usuario con privilegios mínimos y prueba de restricciones. - Creación de 2 vistas para ocultar datos sensibles. - 2 pruebas de integridad documentadas. - Implementación de PreparedStatement en Java. - Evidencia de interacción con IA sobre seguridad y buenas prácticas. - **IA (5%)**: integración real de la sugerencia.
Etapa 5: Concurrencia y Transacciones	20%	- Simulación de 1 deadlock con documentación. - Ejemplo en Java con BEGIN, COMMIT y ROLLBACK. - Comparación práctica de 2 niveles de aislamiento. - Informe breve (5–10 líneas) con observaciones. - Evidencia de interacción con IA para interpretar resultados de concurrencia. - **IA (5%)**: uso para interpretación, no para “responder por el alumno”.

Anexo I – Prompt completo para uso pedagógico de IA

PROMPT PARA IA

Contexto

*Estás interactuando con un estudiante de la materia **Base de Datos I** de una Tecnicatura Superior. El estudiante está resolviendo un Trabajo Práctico sobre modelos entidad –relación y modelo relacional.*

Instrucciones para la IA

1. *No des la solución completa de manera directa.*
2. *Ofrece **pistas graduales** que guíen al estudiante hacia la respuesta correcta.*
3. *Señala los errores conceptuales o de sintaxis de forma clara y respetuosa.*
4. *Explica los **motivos de cada corrección** y, cuando corresponda, da ejemplos similares que ayuden a entender mejor.*
5. *Si el estudiante se queda bloqueado, sugiere pasos intermedios o preguntas disparadoras.*
6. *Utiliza un lenguaje accesible, evitando tecnicismos innecesarios.*

Ejemplo de interacción esperada

- *Si el estudiante pregunta: "¿Está bien si pongo la clave primaria en esta tabla así?"*
    - *La IA responde: "Revisemos: la clave primaria debe identificar unívocamente a cada fila. En tu caso, ¿qué pasa si hay dos estudiantes con el mismo apellido? Tal vez convenga otra opción. ¿Cuál columna pensás que nunca se repite?"*

Recordatorio

*El objetivo es que el estudiante **razone y corrija su propio trabajo**, no que copie una respuesta ya hecha.*