UNIVERSIDAD NACIONAL DE INGENIERÍA

SÍLABO

INFORMACIÓN GENERAL

ASIGNATURA	:BASES DE DATOS
DOCENTE	:GERSON GARRIDO
HORAS POR SEMANA	:06 (TEORÍA: 04, LABORATORIO: 02)

SUMILLA

El curso es de naturaleza teórico-práctica y estudia las bases de datos desde el lado de quien las usa y desde el lado de quien las construye. Se organiza en cuatro partes: **modelo relacional y SQL** (relaciones, claves, álgebra relacional, SQL y su uso desde un programa); **diseño de bases de datos** (modelo entidad-relación, dependencias funcionales y formas normales); **almacenamiento, índices y consultas** (discos y páginas, buffer pool, B+ trees y hashing, algoritmos de join y optimización de consultas); y **transacciones y sistemas distribuidos** (propiedades ACID, serializabilidad, control de concurrencia con locks y MVCC, recuperación con log, particionamiento, replicación y commit distribuido). Se usa PostgreSQL 18 y Node.js 26. El laboratorio y un proyecto que crece durante todo el semestre (la base de datos de una empresa de encomiendas y, en la segunda mitad, un motor de almacenamiento propio) se desarrollan en un contenedor común para todos los estudiantes.

COMPETENCIAS

- Explica el modelo relacional y escribe consultas en álgebra relacional y en SQL, incluidas joins, agregación, subconsultas, vistas y funciones de ventana.
- Diseña una base de datos a partir de requisitos con el modelo E-R, la traduce a tablas y la normaliza con dependencias funcionales (*functional dependencies*) hasta BCNF o 3NF.
- Escribe aplicaciones que usan una base de datos de forma segura, con consultas parametrizadas y transacciones, y reconoce una inyección SQL (*SQL injection*).
- Describe cómo un sistema de gestión de bases de datos organiza los datos en disco y en memoria, e implementa estructuras de índice como B+ trees y tablas hash.
- Estima el costo de los algoritmos de procesamiento de consultas, lee un plan de ejecución y elige índices para una carga de trabajo.
- Explica las propiedades ACID, analiza la serializabilidad de un schedule, compara 2PL, MVCC y los niveles de aislamiento (*isolation levels*), y explica cómo el log recupera la base de datos después de una caída.

UNIDADES DE APRENDIZAJE

Unidad	Contenido	Semanas	Silberschatz
I	Modelo relacional y SQL	1–5	Cap. 1–5
II	Diseño de bases de datos	6–8	Cap. 6, 7, 8.1
III	Almacenamiento, índices y consultas	9–12	Cap. 12–16
IV	Transacciones y sistemas distribuidos	13–16	Cap. 17–19 y otros

La unidad IV toma además partes de los caps. 10, 20, 21 y 23. Cada semana tiene, además de las secciones del libro, lecturas gratuitas en línea que cubren los mismos temas: las notas del curso CS186 de la Universidad de California en Berkeley, las clases del curso 15-445/645 de Carnegie Mellon University y la documentación oficial de PostgreSQL.

PROGRAMACIÓN SEMANAL

Cada semana comprende dos sesiones de teoría de 2 horas (A y B) y una sesión de laboratorio de 2 horas. Cada capítulo de los apuntes empieza con una ruta de lectura: una lectura obligatoria corta antes de cada sesión (15 a 20 minutos), lo que se explica en clase y lo que queda para después, incluidas las secciones indicadas del libro de Silberschatz. Las sesiones marcadas con (*) pueden pasar a lectura autónoma si se pierden clases por feriados.

Sem.	Sesión A	Sesión B	Laboratorio
1	Presentación del curso. Introducción a los sistemas de bases de datos (cap. 1)	El modelo relacional: relaciones, esquemas y claves (sec. 2.1–2.4)	Entorno de trabajo: Docker, `psql`, Node.js. *P0 publicado*
2	Álgebra relacional (sec. 2.5–2.6)	SQL I: definición de datos y consultas básicas (sec. 3.1–3.4)	Primeras consultas con `psql`
3	SQL II: conjuntos, valores nulos y agregación (sec. 3.5–3.7)	SQL III: subconsultas y modificación de datos (sec. 3.8–3.9)	Consultas con agregación y subconsultas. *Entrega P0, publicado P1*
4	Joins y vistas (sec. 4.1–4.2)	Restricciones de integridad, tipos y autorización (sec. 4.4–4.7)	Un esquema que se defiende solo
5	SQL desde un programa: Node.js y `pg`, inyección SQL (sec. 5.1, 9.8)	Funciones, triggers, consultas recursivas y funciones de ventana (sec. 5.2–5.5)	Una aplicación pequeña con `pg`
6	El modelo entidad-relación (sec. 6.1–6.6)	De E-R a tablas. E-R extendido (*) (sec. 6.7–6.10)	Modelar un dominio. *Entrega P1, publicado P2*
7	Dependencias funcionales y BCNF (sec. 7.1–7.3)	Teoría de dependencias: clausura, cubierta canónica, 3NF. 4NF (*) (sec. 7.4–7.6)	Normalizar un esquema
8	Datos semiestructurados: JSON en PostgreSQL (*) (sec. 8.1). Repaso	**Examen parcial**	Repaso con ejercicios
9	Almacenamiento físico: discos, SSD y bloques (cap. 12)	Archivos, páginas y registros. Buffer pool (sec. 13.1–13.5)	Las páginas de PostgreSQL por dentro. *Entrega P2, publicado P3*
10	Índices: conceptos y B+ trees (sec. 14.1–14.3)	Índices hash. Índices en la práctica. LSM trees (*) (sec. 14.4–14.8)	Un B+ tree en Node.js. Índices en PostgreSQL

Sem.	Sesión A	Sesión B	Laboratorio
11	Procesamiento de consultas: costo, selección y ordenamiento externo (sec. 15.1–15.4)	Algoritmos de join. Evaluación con iteradores (sec. 15.5–15.7)	Joins a mano y en el motor
12	Optimización: equivalencias y estimación de cardinalidades (sec. 16.1–16.3)	Elección de planes. `EXPLAIN ANALYZE` (sec. 16.4)	Leer y mejorar planes de ejecución. *Entrega P3, publicado P4*
13	Transacciones: ACID y serializabilidad (sec. 17.1–17.6)	Recuperabilidad y niveles de aislamiento en SQL (sec. 17.7–17.10)	Anomalías con dos sesiones
14	Control de concurrencia: 2PL, deadlocks y granularidad (sec. 18.1–18.4)	MVCC y snapshot isolation. Protocolos con timestamps (*) (sec. 18.5–18.8)	Locks y MVCC en PostgreSQL. *Entrega P4, publicado P5*
15	Recuperación: fallos, log y WAL (sec. 19.1–19.5)	ARIES. Checkpoints (sec. 19.6–19.9)	El WAL de PostgreSQL
16	Bases de datos distribuidas: particionamiento, replicación y 2PC (sec. 20.1–20.5, 21.2–21.4, 23.1–23.2)	NoSQL y MapReduce (*) (sec. 10.2–10.3). Cierre y repaso	Presentación de proyectos. *Entrega P5*
17	**Examen final**		

METODOLOGÍA

- **Teoría:** clases expositivas con discusión de casos, ejercicios resueltos en clase (consultas, diseño, cálculos de costo y análisis de schedules) y demostraciones en vivo sobre PostgreSQL. El estudiante hace la lectura obligatoria de la ruta de lectura antes de cada sesión.
- **Laboratorio:** prácticas guiadas en un contenedor común para todos los estudiantes, con PostgreSQL 18 y Node.js 26, que funciona igual en Linux, Windows y macOS.
- **Proyecto:** un solo proyecto que crece durante el semestre en seis entregas, cada una con pruebas automáticas que deben seguir pasando en las entregas siguientes.
- Los términos técnicos se usan en inglés cuando así se dicen y así aparecen en la bibliografía y en la documentación (join, buffer pool, commit, lock). Los que de verdad se usan en español (tabla, consulta, índice, transacción) se presentan en español, con su equivalente en inglés la primera vez.

Proyecto: Chasqui, base de datos de una empresa de encomiendas

N.º	Entrega	Lo que agrega	Semanas
P0	Calentamiento	Consultas SQL sobre los datos del proyecto; un primer programa en Node.js	1–3
P1	Esquema y consultas	Restricciones de integridad, joins, agregación, subconsultas y vistas	3–6
P2	Diseño y aplicación	Modelo E-R normalizado, migración del esquema y una aplicación con transacciones	6–9
P3	Motor: almacenamiento	Páginas con slots, heap file, buffer pool e índice B+ tree	9–12
P4	Motor: consultas	Operadores con iteradores, joins, ordenamiento externo y agregación	12–14
P5	Motor: transacciones	Lock manager con 2PL estricto, detección de deadlocks y recuperación con log	14–16

USO DE HERRAMIENTAS E INTELIGENCIA ARTIFICIAL

La inteligencia artificial (IA) generativa no se permite en los proyectos ni en los laboratorios: entregar código, consultas o textos producidos con ella cuenta como plagio. Para el resto de herramientas:

Herramienta	En este curso
Mensajes del intérprete y del motor: errores y warnings de Node.js y de PostgreSQL, `node -check`, `EXPLAIN`, el runner de pruebas	Permitido
Documentación oficial (PostgreSQL, Node.js, `pg`) y búsqueda en la web	Permitido. Copiar una solución publicada cuenta como plagio
Autocompletado del editor que completa nombres o una línea, sin generar funciones ni consultas enteras	Permitido
Corrector ortográfico, traductores y herramientas de accesibilidad	Permitido
Formateadores y linters de código y de SQL	Permitido
IA generativa para explicaciones, sin código ni consultas generados	Solo para estudiar la teoría; no mientras se trabaja en un proyecto o un laboratorio
IA generativa que escribe código, consultas o texto	No permitido

El `README.md` de cada entrega declara qué herramientas de la tabla se usaron, aparte del editor y del contenedor del curso, y para qué. Cualquier integrante puede tener que explicar de forma oral cualquier consulta o parte del código entregado.

BIBLIOGRAFÍA

Texto principal

1. Silberschatz, A., Korth, H. F. y Sudarshan, S. (2020). *Database System Concepts* (7.ª ed.). McGraw-Hill. Sitio del libro: `https://www.db-book.com/`

Textos complementarios

2. Ramakrishnan, R. y Gehrke, J. (2003). *Database Management Systems* (3.ª ed.). McGraw-Hill.
3. Garcia-Molina, H., Ullman, J. D. y Widom, J. (2008). *Database Systems: The Complete Book* (2.ª ed.). Pearson.
4. Kleppmann, M. (2017). *Designing Data-Intensive Applications*. O'Reilly.
5. Winand, M. (2012). *SQL Performance Explained*. Disponible gratuitamente en línea como *Use The Index, Luke!*: `https://use-the-index-luke.com/`

Recursos en línea

- Notas del curso CS186, Universidad de California en Berkeley: `https://cs186berkeley.net/notes/`
- Clases del curso 15-445/645, Carnegie Mellon University: `https://15445.courses.cs.cmu.edu/fall2026/schedule.html`
- Documentación de PostgreSQL: `https://www.postgresql.org/docs/current/`
- node-postgres, el driver de PostgreSQL para Node.js: `https://node-postgres.com/`
- Sitio del curso: guías semanales, apuntes, diapositivas, código y enunciados.