

Pontificia Universidad Católica de Chile
Departamento de Ciencia de la Computación
IIC2413 - Bases de Datos

Pauta Examen

Fecha: 12 de diciembre de 2025

2º semestre 2025 - Profesores: Eduardo Bustos - Christian Álvarez

---

INSTRUCCIONES

a) La prueba tiene 3 preguntas, con un total de 60 puntos y una pregunta Bonus de 6 puntos. En total 66 puntos y con 60 puntos se obtiene un 7. **Duración 2,5 horas**.

b) **Lea toda la prueba primero**, las preguntas no están en orden de dificultad

c) Al finalizar la prueba debe digitalizar cada pregunta en formato PDF VERTICAL y subirlas a Canvas **correctamente**.

d) Cada plana debe contener su Nombre Completo y Número de Lista en la parte superior.

e) No mezcle respuestas de diferentes preguntas en la misma plana. Sí puede responder una pregunta en una plana y otra en la otra plana de la misma hoja.

f) Debe firmar la lista de asistencia.

g) No seguir esta instrucción dificulta el trabajo de evaluación y por ende el tiempo de entrega de las notas, además los expone a un descuento de hasta 5 décimas de la nota final.

Pregunta 1 - Evaluación de Consultas y No SQL (15 pts)

a) ¿Qué es `Nested Loop Join`? y explique su funcionamiento paso a paso. (5 pts)

Es un algoritmo para resolver `Joins` entre dos tablas basado en un loop. Funciona iterando (en un "bucle externo") por cada tupla de una tabla A y, para cada una, recorriendo (en un "bucle interno") todas las tuplas de una tabla B y entregando las que cumplen la condición del `Join`.

b) ¿Qué es `Block Nested Loop Join`? Explique la diferencia con `Nested Loop Join` (5 pts)

Es una optimización de Nested Loop Join para reducir el costo de E/S (lectura/escritura en disco) al realizar un `Join`. En lugar de procesar una fila externa a la vez, el motor lee un bloque completo (varias páginas) de la tabla externa y lo compara con la tabla interna.

c) Asocie cada una de los elementos que se listan más abajo con los siguientes tipos de Bases de Datos (5 pts):

- SQL (Transaccionales)
- Documentales
- De Grafos
- Columnar

a) Especializadas en guardar relaciones como redes sociales → `De Grafos`

a) La información cambia mucho → `SQL`

a) Búsquedas simples → `Documentales`

a) Reduce I/O seleccionando sólo las columnas necesarias. → `Columnar`

a) MySQL → `SQL`.

a) Búsquedas complejas → `SQL`

a) Blazegraph → `De Grafos`

a) Despliegue de información estática. → `Documentales`

a) MongoDB → `Documentales`

a) JSON → `Documentales`

Pregunta 2 - Text search and retrieval y Privacidad (15 pts)

a) ¿Que és `TF - IDF` y cuál es su objetivo en el análisis de texto? (3 pts)

Es una técnica que mide la importancia de una palabra en un documento en comparación con un conjunto de documentos.

Su objetivo es realizar indexación y búsqueda en documentos, buscar palabras muy frecuentes y poco relevantes.

b) ¿Por qué las palabras como `el, la, de, un` suelen tener un bajo valor en `TF - IDF`? Justifique su respuesta. (2 pts)

Porque al ser palabras muy comunes tiene un IDF muy bajo, cercano a 0 al estar presente en casi todos los docuemntos. Por lo tanto el valor de $TF * IDF$ es cercano a 0.

c) Defina los siguientes conceptos que son parte de la Ley de Privacidad: (5 pts)

a) Derecho de oposición: Negarse a nuestros datos sean tratados para ciertos fines u objetivos.

b) Derecho de supresión: Pedir la eliminación de nuestros datos cuando ya no sean necesarios o sea ilegal.

c) Derecho de bloquedo: Suspender temporalmente el uso de nuestros datos cuando se esté analizando la legalidad o exactitud.

d) Licitud y transparencia: El tratamiento o uso de los datos debe tener una base legal y comunicado claramente.

e) Finalidad específica: Los datos sólo pueden ser usados para los própositos que fueron informados al momento de ser autorizado.

d) Defina cada uno de los siguientes conceptos y cuál es su rol para mantener la privacidad: (5 pts)

a) k-Anonimato: es una técnica para mantener la privacidad de los datos a través de exigir que los cuasi identificadores estén al menos `k` veces repetidos en un conjunto de datos.

b) l-Diversidad: es una técnic para mantener la privacidad de los datos sensibles en un conjunto de dados anonimizados. Exige que por cada conjunto de cuasi identificadores, existan al menos `l` valores distintos para los datos sensibles.

c) Privacidad diferencial: La privacidad diferencial es un mecanismo que garantiza que un algoritmo aleatorio que opera sobre un conjunto de datos no revele información específica de ningún individuo.

Pregunta 3 - Modelo ER + Diseño de BD + FN + SQL + Lógica en la BD (30 pts)

Los profesores del curso de bases de datos le han encargado un sistema para llevar la gestión del curso IIC2413 para **todos** los períodos académicos (pasados y futuros). Ud. deberá modelar el problema en un MER, luego convertirlo en un esquema relacional y finalmente realizar algunas queries sobre él.
Reglas de negocio:

- Los periodos académicos donde se dicta el curso tienen el siguiente formato AAAA2S donde AAAA es el año y S es el semestre (1 para 1°, 2 para 2°, 3 para TAV). Tambien tienen un nombre y fechas de inicio, término y vigencia. Solo el periodo actual tiene vigencia=verdadero. Todos los anteriores son falsos.
- cada periodo académico puede tener un número diferente de secciones.
- cada sección tiene un profesor diferente en el mismo periodo.
- cada sección tiene un número diferente de ayudantes que son exclusivos de ella.
- se registran los datos de la persona(estudiante, ayudante, profesor) ID(número de alumno), RUN(no es llave), nombre, email(único), teléfono
- tanto de los ayudantes como los profesores pueden haber sido estudiantes en algún período anterior.
- Cada evaluación tiene un nombre y una ponderación. Algunas evaluaciones son con escala de nota (1-7) y otras con puntaje máximo (que para el cálculo de nota final se convierte a la escala normal Pje*6+1). Las evaluaciones(tipo, nombre, nota, etc.) pueden variar de un periodo a otro.
- La nota final se calcula como la suma ponderada de cada evaluación y su porcentaje

I. Modelo ER y Esquema relacional

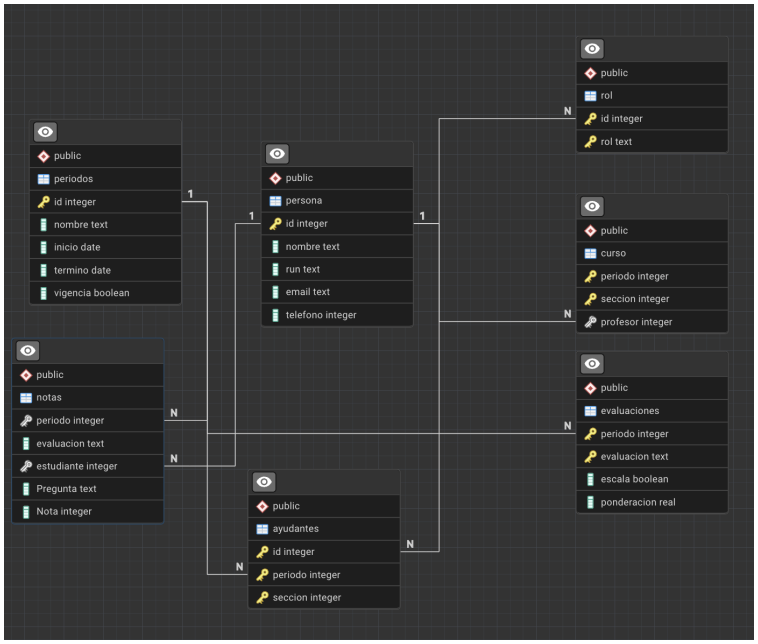
a) Construya un modelo Entidad Relación que represente el problema antes presentado.(3 pts.)

b) Transforme el modelo a un esquema relacional en 3NF especificando las relaciones, atributos, dominios, restricciones de integridad, llaves primarias y foráneas. (2 pts.)

c) Justifique si su modelo está en BCNF. (1 pt.)

pauta
Aquí un posible modelo, este se usa más adelante en las queries en la parte a) se debe mostrar como un modelo entidad relación y en la parte b) el esquema como lo visto en clases, es decir relación(atributo:dominio,...)

en la parte c) se debe demostrar que si hay una relacion a-¿b en todas las tablas, a es llave candidata eso es BCNF o indicar lo contrario y mostrar dónde esta la dependencia que hace que esté fuera. No basta decir "todas lñas relaciones cumplen.º algo similar.



II. Carga de datos en la instancia

Ud. cuenta un archivos excel con lengüetas-tabs (o un .csv por tab) de todos los semestres pasados conteniendo la siguiente información:

- El nombre del archivo excel (o prefijo del csv) es IIC2413-seccion-periodo-RUN del profesor. Ej. IIC2413-1-202522-141237741
- Un tab de excel (o archivo csv) llamado **evaluación** por cada evaluación (y periodo)conteniendo el ID de la persona, evaluación, pregunta y la nota (o puntaje) de cada una de las preguntas. Ej. tab evaluación o IIC2413-1-202522evaluacion.csv
- Un tab de excel (o archivo csv) llamado **lista** conteniendo la lista de todos los estudiantes y sus datos. Ej. tab lista o IIC2413-1-202522lista.csv
- Un tab de excel (o archivo csv) llamado ayudantes conteniendo la lista de los ayudantes Ej. tab lista o IIC2413-1-202522ayudantes.csv

a) Indique en detalle las acciones para la validación de los datos (limpieza) usando PHP (2 pts.)
pauta respuesta: Deben especificar todas las acciones de filtraje de datos para eliminar los datos sucios, entre ello: Verificaciones de formato (pej. RUT, teléfono, mail), valores nulos, valores inválidos(pej. texto en lugar de numeros), encoding, valores no normalizados (medico vs MÉDICO).

b) Indique cómo, a partir de datos limpios, se cargan en las tablas de la base de datos. (2 pts.)
pauta crear las tablas con todos los atributos y restricciones de integridad, para cargar hay dos formas posibles: 1.- hacer un ciclo while not EOF de cada archivo, leer cada linea y hacer insert de los campus en la tabla y atributos correspondientes. En un mismo archivo pueden haber campos correspondientes a varias tablas. 2.- crear una tabla T intermedia con todos los campos del archivo pero sin **restricciones de integridad** y cargar con COPY el archivo TODOS los valores de los campos en esta tabla intermedia. Luego mediante INSERT INTO TABLE A (at1, at2, ..) SELECT At1, At2, .. FROM T

III. Lógica en la BD

a) Cree una vista llamada **Resumen** que contenga la lista de todos los estudiantes (ID y nombre) y sección con los respectivos promedios de las evaluaciones con nota y la suma de los puntajes en caso contrario para un periodo dado 202522. Asuma que el número máximo de evaluaciones es 10 por período y los atributos se la tabla se llaman E1 a E10).(3 pts.)
pauta La vista debe contener la sentencia Create View Resumen 1.- Select Persona.ID, persona.nombre curso.seccion Nota.evaluacion
2.- Join de Persona, curso y notas para obtener las notas de la persona 3.- where periodo=202522 selecciona el periodo entrega ambas secciones

```
CREATE  VIEW Resumen AS
SELECT
p.id       AS id_estudiante,
p.nombre   AS nombre,
c.seccion  AS seccion,
-- Se repite 10 veces E1 a E10 con case o if
CASE -- Selecciona si es nota o puntaje, puede ser IF THEN
    WHEN e1.escala = TRUE
        THEN AVG(n1."Nota"::numeric)
    ELSE SUM(n1."Nota"::numeric)
END AS E1,
FROM persona p, curso c
Where c.estudiante = p.id AND c.periodo    = 202522
-- Se repite 10 veces
LEFT JOIN notas n1
ON n1.estudiante = p.id
AND n1.periodo    = 202522
AND n1.evaluacion = 'E1'
LEFT JOIN evaluaciones e1
```

```
    ON e1.periodo    = n1.periodo
    AND e1.evaluacion = n1.evaluacion

    GROUP BY
        p.id,
        p.nombre,
        c.seccion;
```

b) Cree un Store procedure que calcule la nota final del curso para cada estudiante. (3 pts.) pauta

```
CREATE PROCEDURE nota_final(p integer ---p periodo)
LANGUAGE plpgsql
AS $$
DECLARE
    est record;
    v_nota_final numeric;
BEGIN
    -- Recorre las notas de  cada estudiante
    FOR est IN
        SELECT DISTINCT estudiante
        FROM notas
        WHERE periodo = p
    LOOP
        *   - si escala = TRUE:
         *   - si escala = FALSE: usar suma de "Nota" * ponderacion
         * Luego sumar todos esos aportes.
         */
        SELECT
            SUM(
                CASE -- Selecciona su es Nota o Puntaje
                    WHEN e.escala = TRUE  -- es NOTA, ELSE PUNTAJE
                        THEN AVG(n."Nota") * e.ponderacion
                    ELSE SUM(n."Nota") * e.ponderacion
                END
            )
        INTO v_nota_final
        FROM notas n
        JOIN evaluaciones e
          ON e.periodo    = n.periodo
         AND e.evaluacion = n.evaluacion
```

```
        WHERE n.periodo    = p_periodo
          AND n.estudiante = r_est.estudiante
        GROUP BY n.estudiante;

        -- Insertar los resultados en una tabla notas_finales
        INSERT INTO notas_finales(periodo, estudiante, nota_final)
        VALUES (p_periodo, r_est.estudiante, COALESCE(v_nota_final, 0));
    END LOOP;
END;
$$;
```

c) Cree un Trigger que llame al SP cada vez que se ingresa un promedio de evaluación en la tabla temporal. (2 pts.) pauta

```
CREATE TRIGGER insert_after
AFTER INSERT ON tabla temporal
FOR EACH ROW
EXECUTE FUNCTION  nota_final();
}
```

IV. Consultas

a) Cree una consulta que entregue el promedio del curso (sección) en un periodo P dado y para una sección dada S. (2 pts.) Pauta

```
    SELECT
    c.periodo,
    c.seccion,
    AVG(n."Nota") AS promedio_curso
FROM curso c
JOIN notas n
    ON c.periodo = n.periodo
JOIN evaluaciones e
    ON n.periodo = e.periodo
   AND n.evaluacion = e.evaluacion
JOIN rol r
    ON n.estudiante = r.id
WHERE r.rol = 'estudiante'
  AND c.periodo = P
  AND c.seccion = S
  AND e.escala = true          -- solo evaluaciones con nota
GROUP BY
    c.periodo,
    c.seccion;
```

b) Cree una consulta que entregue los estudiantes que han cursado el ramo más de una vez. (2 pts.) pauta

```
SELECT
    p.id,
    p.nombre
FROM curso c
JOIN persona p
  ON p.id = c.estudiante
GROUP BY
    p.id,
    p.nombre
HAVING
    COUNT(DISTINCT c.periodo) > 1;
```

c) Cree una consulta que entregue los nombres de todos los profesores y ayudantes que fueron alumnos del curso; primero liste los profesores, luego los ayudantes. (2 pts.) pauta

```
    SELECT DISTINCT
    p.nombre,
    r2.rol
FROM persona p
JOIN rol r2
    ON p.id = r2.id
WHERE r2.rol IN ('profesor', 'ayudante')
  AND p.id IN (
        SELECT r1.id
        FROM rol r1
        JOIN notas n
            ON r1.id = n.estudiante
        WHERE r1.rol = 'estudiante'
    )
ORDER BY
    CASE
        WHEN r2.rol = 'profesor' THEN 1
        WHEN r2.rol = 'ayudante' THEN 2
    END,
    p.nombre;
```

d) Cree una consulta que entregue la nota final de cada estudiante de un curso sección y periodo, asumiendo que todas las notas están puestas.(3 pts.) pauta

```
    SELECT
    p.id     AS estudiante_id,
    p.nombre AS estudiante,
    c.periodo,
    c.seccion,
    SUM(n."Nota" * e.ponderacion) AS nota_final
FROM curso c
JOIN notas n
    ON c.periodo = n.periodo
JOIN evaluaciones e
    ON n.periodo = e.periodo
   AND n.evaluacion = e.evaluacion
JOIN persona p
    ON n.estudiante = p.id
JOIN rol r
    ON p.id = r.id
WHERE r.rol = 'estudiante'
  AND c.periodo = P
  AND c.seccion = S
  AND e.escala = true        -- solo evaluaciones con nota
GROUP BY
    p.id,
    p.nombre,
    c.periodo,
    c.seccion;
```

e) Dado el siguiente código en PHP que lee el archivo iic2413_1_202522_I1.csv por líneas y extrae de cada una el RUN, nombre y nota del estudiante. Modifique el código para que con sentencias sql embebidas en PHP cree una transacción (Begin Transaction, Commit) inserte los datos de la evaluación I1 del periodo 202522 en la tabla que corresponda de su esquema. Nota: Suponga que el archivo está abierto y el programa está correctamente conectado a la base de datos. hint(se puede usar pseudocódigo). (3 pts.)

```
// Usando Pseudo PHPy SQL
// Lee la 1a linea de encabezados
\$encabezados = fgetcsv($handle, 1000, ',');
// mientras no termine el archivo \$archivo lee los datos y almacena en \$fila
$result = pg_query($conn, "BEGIN TRANSACTION"); // $conn tiene la conexión a la
while ((\$fila = fgetcsv(\$archivo, 1000, ',')) !== false) {
    $registro = array_combine($encabezados, $fila);
    // guarda los campos run, nombre, nota en las variables homónimas
    $run  = $registro['RUN'];
```

```
        $nombre = $registro['Nombre'];
        $nota = $registro['nota'];
        $result = pg_query($conn, "select ID from Persona where RUN=$run") transfor
        $result = pg_query($conn, "INSERT INTO TABLE notas values (202522,I1,$ID,NU
    }
        $result = pg_query($conn, "COMMIT");
```

Si lo considera pertinente puede hacer supuestos dejándolos claramente indicados

USE LA HOJA DE RESPUESTA CON EL CÓDIGO INCLUIDO

Bonus Full Text Search & retrieval - 6 puntos

a) ¿Qué son las formas normales y cuál es su objetivo? (2 pts)

Las formas normales son un conjunto de reglas para estructurar la información en una base de datos, con el objetivo de reducir la redundancia de datos, evitar anomalías de actualización y mejorar la integridad y eficiencia de la base de datos.

b) La transacción `T1` actualiza el precio de un producto y la transacción `T2` calcula el total de las ventas del producto usando el precio. ¿Qué propiedad ACID se debe cumplir para que la ejecución anterior sea correcta? (2 pts)

La propiedad ACID que se debe cumplir es el `Aislamiento (isolation)`, ya que esta propiedad exige que las transacciones se ejecuten como si fueran secuenciales, es decir `T2` debe leer el valor del precio anterior o posterior a la ejecución de `T1`.

c) ¿Qué problema ocurre en un índice `Hash` cuando hay muchas colisiones? ¿Cómo se soluciona? Justifique su respuesta. (2 pts)

Se produce una colisión de tuplas en un bucket, originando encadenamiento de ellas y haciendo más lento el proceso de búsqueda.

Se soluciona realizando un proceso de `rehashing`, para distribuir en un espacio más grande las tuplas indexadas.

Torpedo SQL

Sintaxis consulta SQL

```
SELECT [ALL | DISTINCT] * | [COUNT, MAX, MIN, SUM, AVG](col1), col2, ...
       [AS alias_column]
FROM table1
     [JOIN table2 ON table1.common_field = table2.common_field]
WHERE condition1 [AND | OR condition2]
      [BETWEEN value1 AND value2]
      [IN (value1, value2, ...)]
      [LIKE pattern]
      [= | <> | <= | >= | < | >] value
GROUP BY column1, column2, ...
HAVING aggregate_condition [COUNT, MAX, MIN, SUM, AVG]
ORDER BY column1 [ASC | DESC]
UNION | INTERSECT | EXCEPT
[SELECT column1 FROM another_table]
LIMIT row_count;
```

Sintaxis Stored Procedure

```
CREATE OR REPLACE FUNCTION <nombre_función> (<argumentos>)
RETURNS [VOID | TABLE(...) ] AS $$
DECLARE
    <declaración_variables>
BEGIN
    <sentencias_SQL>
END; $$ language plpgsql
```

Sintaxis PL/pgSQL

```
BEGIN
    IF <condición> THEN
        <cuerpo_if>
    ELSE
        <cuerpo_else>
    END IF;
    INSERT INTO <tabla> VALUES (<tupla_valores>);
    FOR <variable> IN <iterable>
    LOOP
        <cuerpo_loop>
    END LOOP;
    RETURN QUERY <consulta_SQL>;
END;
```

Sintaxis pg_query

```
    $result = pg_query($conn, $sql);
```