Explicaciones prácticas

MODELO CONCEPTUAL

el diseño conceptual parte de la especificación de requerimientos y su resultado es el esquema conceptual de la BD. Un esquema conceptual es una descripción de alto nivel de la estructura de la BD. El propósito del diseño conceptual es describir el contenido de información de la BD. La herramienta usada es el modelo entidad-relación

- **entidad:** representación de un elemento u objeto del mundo real con identidad
- **relación:** las relaciones representan agregaciones entre dos o más entidades Describen las dependencias o asociaciones entre dichas entidades
    - relación recursiva: relación que uno dos entidades particulares del mismo conjunto

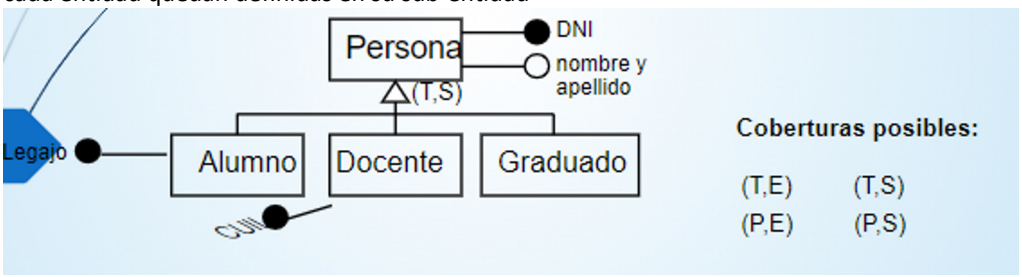
CARDINALIDAD EN RELACIONES	nivel de correspondencia entre las entidades que se relacionan. Se debe definir el nivel máximo de correspondencia (cardinalidad máxima) y el nivel mínimo de correspondencia (cardinalidad mínima)

- **atributo:** un atributo representa una propiedad básica de una entidad o relación. Un atributo es el equivalente a un campo de un registro
    - compuesto: un atributo compuesto representa un atributo generado a partir de la combinación de varios atributos simples
    - identificador: un identificador es un atributo o un conjunto de atributos que permite reconocer o distinguir a una entidad de manera unívoca dentro del conjunto de entidades

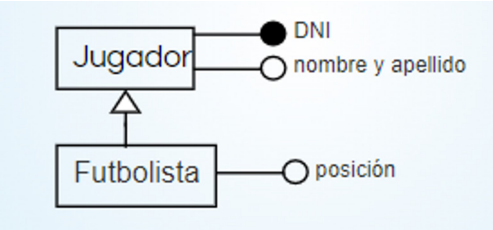
CARDINALIDAD DE ATRIBUTOS	los atributos, tienen asociado el concepto de cardinalidad. Cuando se define un atributo se debe indicar si es o no obligatorio y si puede tomar más de un valor (polivalente)
MONOVALENTE OBLIGATORIO (1,1)	la cardinalidad existe y está presente pero solamente en este caso no se indica de forma explícita
MONOVALENTE NO OBLIGATORIO	(0,1)
POLIVALENTE NO OBLIGATORIO	(0,N)
POLIVALENTE OBLIGATORIO	(1,N)

identificador compuesto: el identificador está conformado por más de un atributo
identificador externo: identificador conformado por atributos que pertenecen a otra entidad

jerarquías de generalización: permiten extraer propiedades comúnes de varias entidades o relaciones y generar con ellas una super-entidad que las aglutine. Así, las características compartidas son expresadas una única vez en el modelo y los rasgos específicos de cada entidad quedan definidas en su sub-entidad



subconjuntos: caso especial de las jerarquías de generalización donde se tiene una generalización de la que se desprende solamente una especialización. No es necesario indicar la cobertura para los subconjuntos



MODELO LÓGICO

el propósito de la generalización de un modelo ER lógico es convertir el esquema conceptual en un modelo más cercano a la representación entendible por el SGBD. Recordemos que el diseño conceptual busca representar, de la forma más clara posible, las necesidades del usuario. Una vez cumplido este paso, el diseño lógico busca representar un esquema equivalente, que sea más eficiente para su utilización
las decisiones sobre el diseño lógico están vinculadas, básicamente, con cuestiones generales de rendimiento y con un conjunto de reglas que actúan sobre características del esquema conceptual que no están presentes en los SGBD relacionales
las decisiones a tomar en el diseño lógico son con respecto a

- jerarquías, según la que sea

TIPO	SOLUCIÓN
TOTAL EXCLUSIVA (T,E)	• dejar todo • dejar solo los hijos • dejar solo el padre
TOTAL SUPERPUESTA (T,S)	• dejar todo • dejar solo al padre (nunca puedo eliminarlo)
PARCIAL SUPERPUESTA (P,S)	• dejar todo • dejar solo al padre (no puedo eliminarlo)

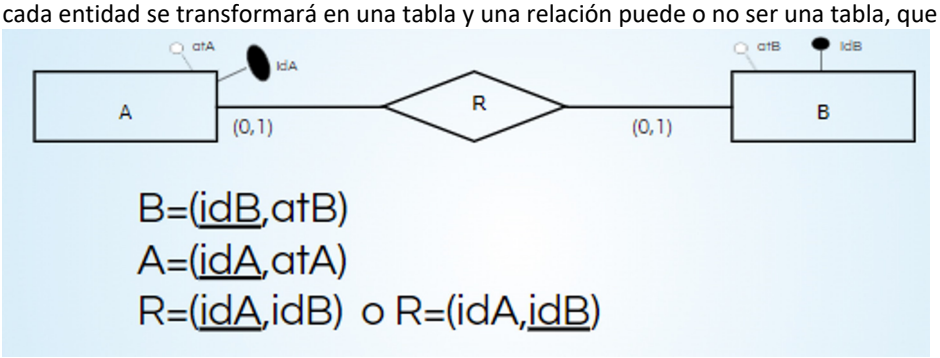
- atributos compuestos, puedo

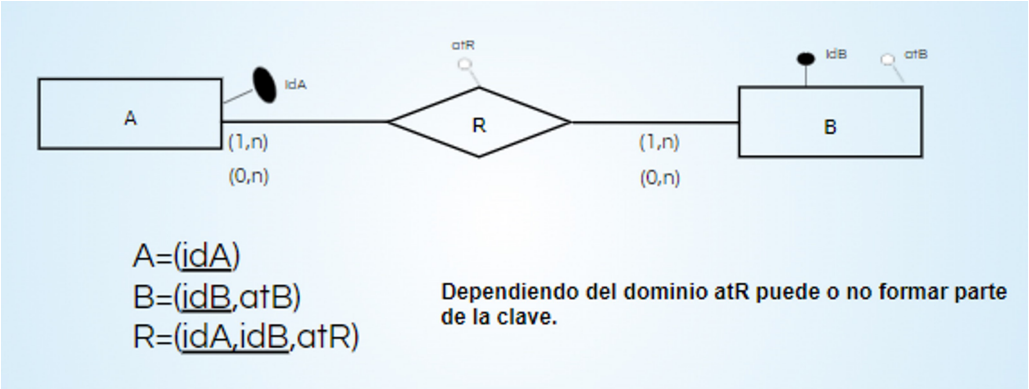
CONSIDARAR SOLO ATRIBUTOS INDIVIDUALES
CONSIDERAR TODO EN UN SOLO ATRIBUTO

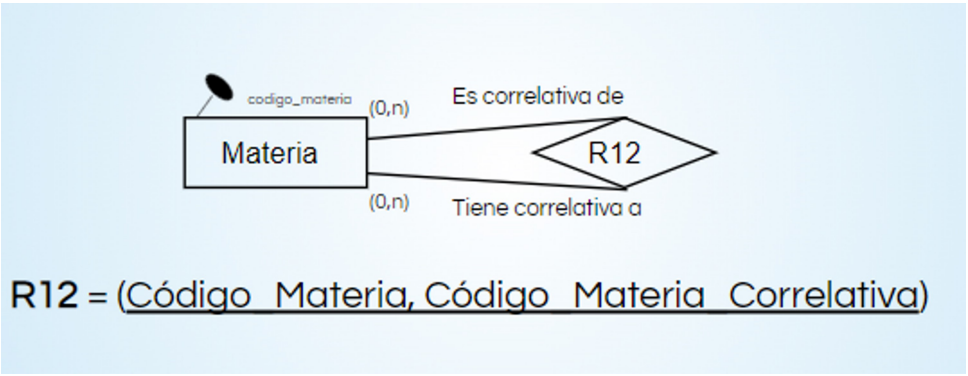
- atributos polivalentes: para resolver los atributos polivalentes se debe agregar una entidad y una interrelación

MODELO FÍSICO

el modelo relacional representa a una BD como una colección de archivos denominados tablas. Cada tabla se denomina relación y está integrada por filas y columnas. Cada fila se denomina *tupla* y cada columna representa un atributo
cada entidad se transformará en una tabla y una relación puede o no ser una tabla, que lo sea dependerá de su cardinalidad







ÁLGEBRA RELACIONAL

se denomina AR a un conjunto de operaciones simples sobre tablas, a partir de las cuales se definen operaciones más complejas mediante composición. Define, por tanto, un lenguaje de manipulación de datos

OPERACIÓN	ACCIÓN
selección	produce una tabla que contiene únicamente aquellas tuplas de T que satisfacen el predicado p
proyección	produce una tabla que contiene un subconjunto de atributos de T eliminando tuplas duplicadas
unión	produce una tabla que contiene todas las tuplas de T1 más todas las de T2 eliminando las tuplas duplicadas. T1 y T2 deben ser compatibles
intersección	produce una tabla que contiene todas las tuplas que se encuentran tanto en T1 como en T2. T1 y T2 deben tener esquemas compatibles
producto cartesiano	produce una tabla concatenando cada tupla de T1 con todas las tuplas de T2
producto natural	produce una tabla concatenando tuplas de ambas tablas que tengan valores iguales en atributos con igual nombre. Se elimina uno de los ejemplares de cada atributo común
diferencia	produce una tabla que contiene todas las tuplas de T1 que no se encuentran en T2. T1 y T2 deben tener esquemas compatibles
división	produce una tabla con los campos T1-T2 donde los valores en esos campos de T1 se corresponden con TODAS LAS TUPLAS DE T2. El esquema de T2 debe estar incluido en T1
remombre	renombra la tabla
asignación	vuelva los resultados de una consulta

SQL (STRUCTURED QUERY LANGUAGE)

SQL es un lenguaje de consultas de BD compuesto por dos submódulos

- módulo para definición del modelo de datos, denominado DDL(Data definition languaje)
- módulo para la operatoria normal de la BD, denominado DML (Data manipulation languaje)

DEFINICIÓN DE DATOS

- de base de datos: CREATE | DROP SCHEMA

- de dominios: CREATE | ALTER | DROP DOMAIN
- de tablas: CREATE | ALTER | DROP TABLE
- de visitas: CREATE | DROP VIEW
- de índices: CREATE | DROP INDEX

MANIPULACIÓN DE DATOS

- SELECT: lista contenido de una o varias tablas
  SELECT campo FROM tabla
  puedo usar * que me mostrará todos los campos de la tablas involucradas sin necesidad de escribir uno por uno
  DISTINCT se aplica a campos del select y elimina tuplas repetidas
  WHERE va luego del FROM y se usa para filtrar filas que cumplan determinada condición
  SELECT campo FROM tabla WHERE condición

OPERADOR	SIGNIFICADO
=	es igual a
>	es mayor a
<	es menor a
>=	mayor o igual a
<=	menor o igual a
<>	distinto a
BETWEEN	entre (se incluyen extremos)
LIKE	como

LIKE brinda gran potencia para aquellas consultas que requiere manejo de Strings. Se puede combinar con:
% que representa cualquier cadena inclusive la cadena vacía. Ejemplo: SELECT nombre, apellido FROM alumno WHERE (apellido LIKE '%ez')
_ (guión bajo) sustituye sólo el carácter del lugar donde aparece. Ejemplo: SELECT nombre, apellido FROM alumno WHERE (nombre LIKE '__ciana')

IS NULL verifica si un atributo contiene el valor de NULL, valor que se almacena por defecto si el usuario no define otro. Ejemplo: SELECT nombre, apellido FROM alumno WHERE (nombre IS NOT NULL)

PRODUCTO CARTESIANO (,) para realizar un producto cartesiano, basta con poner en la cláusula FROM dos o más tablas separadas por coma. Ejemplo: SELECT * FROM alumno,materia

INNER JOIN producto natural, reúne las tuplas de las relaciones que tienen sentido. El producto natural se realiza en la cláusula FROM indicando las tablas involucradas en dicho producto y luego de la sentencia ON la condición que debe cumplirse. Ejemplo: SELECT a.nombre,a.apellido, e.nota FROM alumno a INNER JOIN examen e ON (a.dni=e.dni)

LEFT JOIN contiene todos los registros de la tabla de la izquierda, aún cuando no exista un registro correspondiente en la tabla de la derecha, para uno de la izquierda. Retorna un valor nulo en caso de no correspondencia

RIGHT JOIN es la inversa de LEFT JOIN. Ejemplo: SELECT * FROM alumno a LEFT JOIN examen e ON (a.dni=e.dni)

OPERADOR	SIGNIFICADO
UNION	misma interpretación que en AR. No retorna tuplas duplicadas. Las consultas a unir deben tener esquemas compatibles
UNION ALL	misma interpretación que la UNION pero retorna las tuplas duplicadas. Ejemplo: SELECT nombre FROM alumno WHERE (dni > 25000000) UNION SELECT nombre FROM materia
EXCEPT	cláusula definida para la diferencia de conjuntos. Ambas consultas deben tener esquemas compatibles. Ejemplo: SELECT dni FROM alumno WHERE (dni > 25000000) EXCEPT (SELECT dni FROM examen
INTERSECT	clausula para la operación de intersección
ORDER BY	permite ordenar las tuplas resultantes por el atributo que se le indique. Por defecto ordena de menor a mayor (operador ASC). Si se desea ordenar de mayor a menor se usa el operador DESC. Ejemplo: SELECT nombre, apellido,dni FROM alumno WHERE (dni>23000000) and (nombre='Luciana')

	ORDER BY apellido, dni DESC. dentro de la cláusula ORDER BY se pueden indicar más de un criterio de ordenación. El segundo criterio se aplica en caso de que empate en el primero y así sucesivamente

FUNCIONES DE AGREGACIÓN	usadas en el select y operan sobre un conjunto de tuplas de entrada produciendo un único valor de salida
AVG	promedio del atributo indicado para todas las tuplas del conjunto
COUNT	cantidad de tuplas involucradas en el conjunto de salida
MAX	valor más grande dentro del conjunto de tuplas para el atributo indicado
MIN	valor más pequeño del conjunto de tuplas para el atributo indicado
SUM	suma del valor del atributo indicado para todas las tuplas del conjunto
GROUP BY	agrupa las tuplas de una consulta por algún criterio con el objetivo de aplicar alguna función de agregación. Por ejemplo: SELECT nombre, apellido, AVG(nota) as promedio FROM alumno a INNER JOIN examen e ON (a.dni=e.dni) GROUP BY e.dni, nombre, apellido
HAVING	se usa con la cláusula GROUP BY para restringir los grupos que aparecen en la tabla de resultados mediante alguna condición que deben cumplir los grupos. Ejemplo: SELECT nombre, apellido, AVG(nota) as promedio FROM alumno a INNER JOIN examen e ON (a.dni=e.dni) GROUP BY e.dni, nombre, apellido HAVING promedio > 5

SUBCONSULTAS	consiste en ubicar una consulta SQL dentro de otra. SQL define operadores de comparación para subconsultas
=	cuando una subconsulta retorna un único resultado, es posible compararlo contra un valor simple
IN	comprueba si un elemento es parte o no de un conjunto. Negación es NOT IN
=SOME	igual a alguno
>ALL	mayor que todos
<=SOME	menor o igual que alguno
EXIST	permite comprobar si una subconsulta generó o no alguna tupla como respuesta. El resultado de la cláusula EXIST es verdadero si la subconsulta tiene al menos una tupla y falso en caso contrario. La negación es NOT EXIST. Ejemplo: SELECT nombre, apellido FROM alumno a WHERE EXISTS (SELECT * FROM examen e WHERE a.dni=e.dni and e.nota<=4)

- INSERT: inserta contenido en una tabla
- UPDATE: actualiza contenido de una tabla
- DELETE: elimina contenido de una tabla

EQUIVALENCIA AR

T1 |X| T2	SELECT * FROM T1 NATURAL JOIN T2
T1 |X| T2	SELECT * FROM T1 INNER JOIN T2 ON (…)
T1 |X| T2	SELECT * FROM T1 NATUAL JOIN T2 WHERE
T1 X| T2	SELECT * FROM T1 LEFT JOIN T2 ON (…)
T1- T2	T1 EXCEPT T2

OPERACIONES ABM

INSERT INTO	agrega tuplas a una tabla. Ejemplo: INSERT INTO alumno (dni, nombre, apellido) VALUES (22670845, Juan,'Del Piero');
DELETE FROM	borra una tupla o un conjunto de tuplas de una tabla. Ejemplo: DELETE FROM alumno WHERE dni=28670845;
UPDATE … SET	modifica el contenido de uno o varios atributos de una tabla. Ejemplo: UPDATE alumno SET nombre='Luciana' WHERE dni=24564321