Clase 3

Modelos Relacionales

Basado el algebra booleana y algebra de predicados. Aparece a resolver problemas de consistencia y de redundancia. Existen el modelo lógico y el modelo físico. El modelo lógico es el der y la definición de atributos. Cuando lo implemento, se vuelve modelo físico. Ahora los elementos son tablas, los atributos son columnas y las filas son los valores que toman esos atributos

**Relación:** Conjunto de tuplas sin importar su orden

**Tuplas:** Conjunto de atributos no vacío de los elementos. Una dupla son 2 elementos

**Atributos:** característica de interés y tiene un dominio

**Dominio:** Valores posibles de un atributo acotado por el contexto

**Valor nulo**: definido con valor **desconocido**, que permite agregar un nuevo valor a la tabla de valores de V o F. No es un valor ni true ni false

A screenshot of a computer

AI-generated content may be incorrect.

**Primary key**: Cada tupla tiene un atributo o conjunto de atributos (donde ninguno es nulo), que conociendo su o sus valores se puede reconocer unívocamente a la tupla

**Foreign key**: Es un atributo o conjunto de atributos que es **primary key** en otra relación. Implementa el concepto de integridad referencial. Hay 2 tipos

* Identificativa -> en su tabla forma parte de la primary key
* No Identificativa -> en su tabla NO forma parte de la primary key

**Integridad referencial**: una tabla de referencia siempre debe aludir a una fila válida le la tabla a la que se le haga referencia. Garantiza que la relación entre 2 tablas permanezca sincronizada durante las operaciones de inserción, actualización y eliminación

**Independiente de la implementación**: el modelo es independiente de la implementación física, es abstraída. Mi tabla o elemento funciona igual sin importar la implementación

**Independiente del orden**: Los conjuntos o elementos de la relación no tienen un elemento relacionado

Las tablas deben estar normalizadas

Es un proceso repetitivo que estandariza para que todos sigan el mismo patron que saca redundancias y deja el modelo acorde a lo relacional. Los 3 pasos o formas normales son:

* 1FN: Tiene primary key, no tiene atributos calculables (obtenible a partir de otro atributo o columna en mi tabla), atributos repetitivos (Se va a repetir varias veces en la misma instancia) ni compuestos (en su interior se compone por varios atributos, como el *struct* de C).
* 2FN: Es 1FN y todos los atributos no clave dependen de la totalidad de la clave (Primary key), es decir, que si la primary key son 2 atributos, cada atributo no clave debe depender de ambos.
* 3FN: está en 2FN y todos los atributos no clave no dependen de algún atributo no clave

Ventajas y desventajas de la normalización

* Es útil para quitar todo tipo de redundancia.
* Es útil para dejar un modelo estable y estandarizado
* No es útil, si tengo que calcular siempre los valores, por ejemplo, para saber el precio total de la factura tendría que sumar siempre que lo necesite, por lo cual incrementa el costo computacional, versus es costo espacial
* En algunas casos, la redundancia puede ser de utilidad.