PREGRADO

Unidad 4 | Trending Topics

RELATIONAL / NON-RELATIONAL DATABASE DESIGN REVIEW

SI720| Diseño y Patrones de Software

AGENDA

INTRO

RELATIONAL DATABASE DESIGN

NON-RELATIONAL APPROACH

Database

Una colección de información coherente, siendo información una forma procesada de datos.

Las bases de datos se asocian normalmente con datos estructurados: datos que se almacenan según una clase de patrón regular, permitiendo que una computadora pueda procesarlos y filtrarlos, así como inferir respuestas a preguntas que no están establecidas como entradas originales en los datos.

Data Model

Un conjunto de reglas, supuestos y convenciones que sirven para transformar conceptos abstractos en el mundo real, en un modelo finito de objetos en la computadora por medio de un número fijo de conceptos reutilizables.

El modelo relacional es uno de los más conocidos y utilizados comercialmente, aunque no es el único.

AGENDA

INTRO

RELATIONAL DATABASE DESIGN

NON-RELATIONAL APPROACH

Relational Database

Una base de datos relacional es aquella que organiza sus datos en colecciones de Tablas, Filas, Atributos y Dominios.

También conocida como SQL Database, en ella se utiliza la lógica de predicados para describir la información contenida en la base de datos y para realizar consultas sobre ella.

Relational Database Design

El diseño de bases de datos SQL se apoya a menudo en techniques como normalization, cuyo objetivo es reducir o eliminar datos duplicados en una base de datos para reducir errores en los datos almacenados.

Cada tabla en una base de datos relacional contiene datos de un mismo tipo o concepto, por ejemplo, addresses.

Relational databases utilizan un conjunto de reglas conocidas como ACID (Atomicity, Consistency, Integrity, Durability) que guían el database design.

Relational Database Design

Pasos clave:

- Definir el propósito de la base de datos.
- Investigar y recolectar toda la información sobre los datos a incluirse y el propósito de la base de datos en el tiempo.
- Dividir los datos a incluirse en subjects o types, por ejemplo, información de user account. Cada uno de ellos se convertirá (o debería convertirse) en tablas de base de datos.
- Para cada tabla de base de datos, identificar los data point a incluir. Cada data point se convertirá en una columna en la tabla de base de datos.
- Para cada tabla de base de datos, identificar la óptima primary key para identificar de manera única a cada fila de la tabla.
- (continua...)

Relational Database Design

Pasos clave (continuación):

- Comparar y evaluar cómo se relacionan entre sí los datos en las tablas. Adicionar campos o tablas, para aclarar las relaciones de datos en cada tabla. Por ejemplo, una base de datos con información de contactos para compañías podría necesitar incluir múltiples direcciones, números de teléfono entre otros datos para cada compañía.
- Probar el diseño de la base de datos en papel, escribiendo consultas para las tareas más recurrentes. Refinar el diseño de tablas según se requiera.
- Normalizar el diseño de base de datos para asegurarse que cada tabla represente una cosa o concepto, con referencias y relaciones hacia otras tablas según se necesite.

Entity Relationship Diagram

Entity Relational (ER) ffiodel es un diagrama conceptual de alto nivel de un modelo de datos. Representa entidades del mundo real y sus relaciones entre ellas.

Un Entity-Relationship Diagram (ERD), es una herramienta visual de utilidad para representar un Entity Relationship ffiodel.

Entity Relationship Diagram

Componentes:

Entities

Attributes

Relationships

Elementos

Entity. Concepto del mundo real.

Attribute. Características o propiedades de un entity.

Relationship. Dependencia o asociación entre entities.

Ejemplos:

Customer y *Product* son entidades.

Customer number y *name* son atributos de *Customer*.

Product name y *price* son atributos de *Product*.

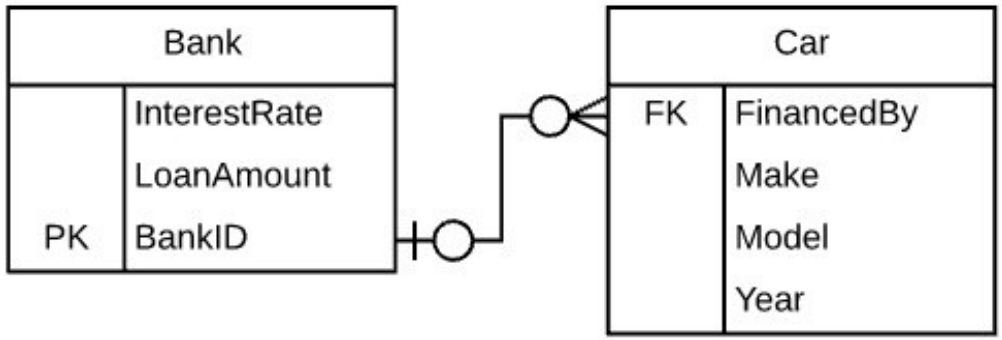
Sale es la relación entre *Customer* y *Product*.

Levels

Conceptual. Entities y Relationships

Logical. Entities, Attributes y Relationships.

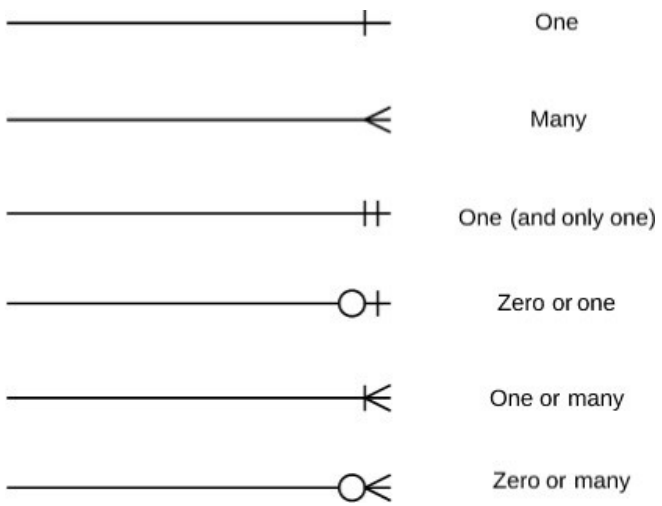
Physical. Todas las estructuras de tables, incluyendo column name, column data type, column constraints, primary key, foreign key y relationships entre tables.



Cardinality & Ordinality

Cardinality. ffiáximo número de veces que una instancia de entidad se puede relacionar con instancias de otra entidad.

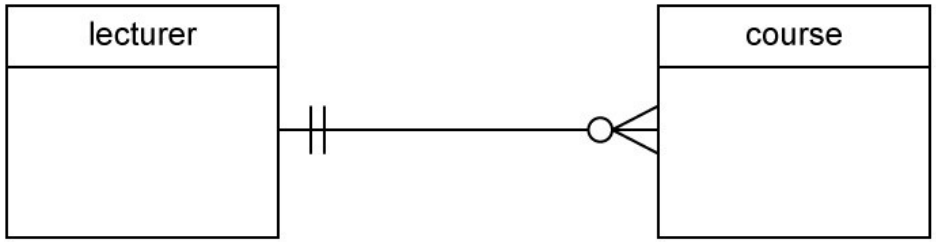
Ordinality. ffiínimo número de veces que una instancia de una entidad puede asociarse con una instancia de la entidad relacionada.



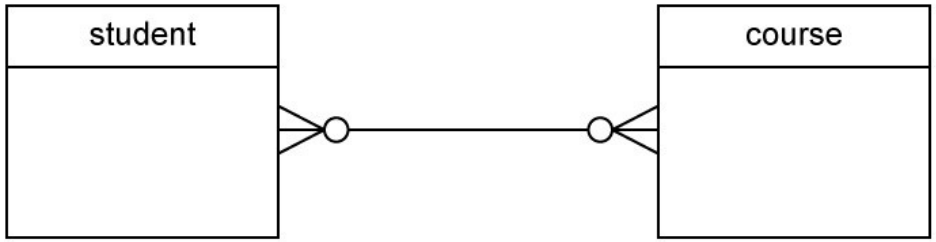
Cardinality & Ordinality

One-to-one

One-to-many



ffiany-to-many



ERD Logical Naming conventions

Elemento	Convención	Ejemplo
Entity	Noun, Upper Camel Case, Singular	Person, Customer
Attribute	Noun, Lower Camel Case, Singular	name, starsCount
Relationship	Verb, Present Tense, Third-Person, Upper Camel Case	Sales, Produces, Holds

ERD Physical Naming conventions

Elemento	Convención	Ejemplo
Table	Noun, Snake Case, Plural	people, customers, order_items
Column	Noun, Snake Case Singular	name, stars_count
Primary Key Column	id	id
Foreign Key Column	<entity_name>_id	customer_id, product_id

Audit Trail Fields Naming conventions

Elemento	Logical	Physical
Creation Date	createdAt	created_at
Creation User ID	createdBy	created_by
Last Update Date	updatedAt	updated_at
Last Update User ID	updatedBy	updated_by

Many-to-Many Relationship Table Naming conventions

Recomendable. Identificar un término que represente el concepto de la relación.

Válido. Término compuesto por nombres de ambas entidades que se relacionan.

From Entity	To Entity	From Table	To Table	Join Table
Warehouse	Product	warehouses	products	inventories
Student	Course	students	courses	course_enrolments
User	Plan	users	plans	subscriptions
Agent	Property	agents	properties	agent_assignments

AGENDA

INTRO

RELATIONAL DATABASE DESIGN

NON-RELATIONAL APPROACH

Non-relational Database

Una *non-relational database*, almaceda los datos en formato no tabular. A menudo llamadas *NoSQL Databases*, no siguen el modelo relacional. Las non-relational databases se pueden basar en estructuras tales como *documents*.

Un *document* puede ser altamente detallado a la vez que puede contener una variedad de diferentes tipos de información en diversos formatos.

Esta habilidad para alojar y organizar varios tipos de información coexistiendo en un mismo lugar es lo que hace que las non-relational databases sean más flexibles que las relational databases.

NoSQL Database Design

El diseño en NoSQL Databases depende del tipo de database, llamadas *stores*:

Document Stores. Asocian cada key identifier con un document, que puede ser propiamente un document, key-value pairs, o key-value arrays.

Graph Stores. Diseñada para almacenar datos que se representan mejor con grafos, datos interconectados con un número indeterminado de relaciones entre datos (por ejemplo, social networks) o mapas de rutas.

Key-value Stores. Es el tipo más simple, donde cada bit de datos almacenados (como keys) y sus datos (values).

Wide Column Stores. Optimizadas para consultas sobre grandes data sets.

NoSQL Database Data Model Techniques

Normalmente los datos se duplican en muchos lugares distintos de la database para ayudar a responder preguntas con menos esfuerzo.

NoSQL Database Design aplica un conjunto de reglas conocidas como BASE (Basically available, Soft-state, Eventually consistent) como guía para el diseño.

NoSQL Database Data Model Techniques

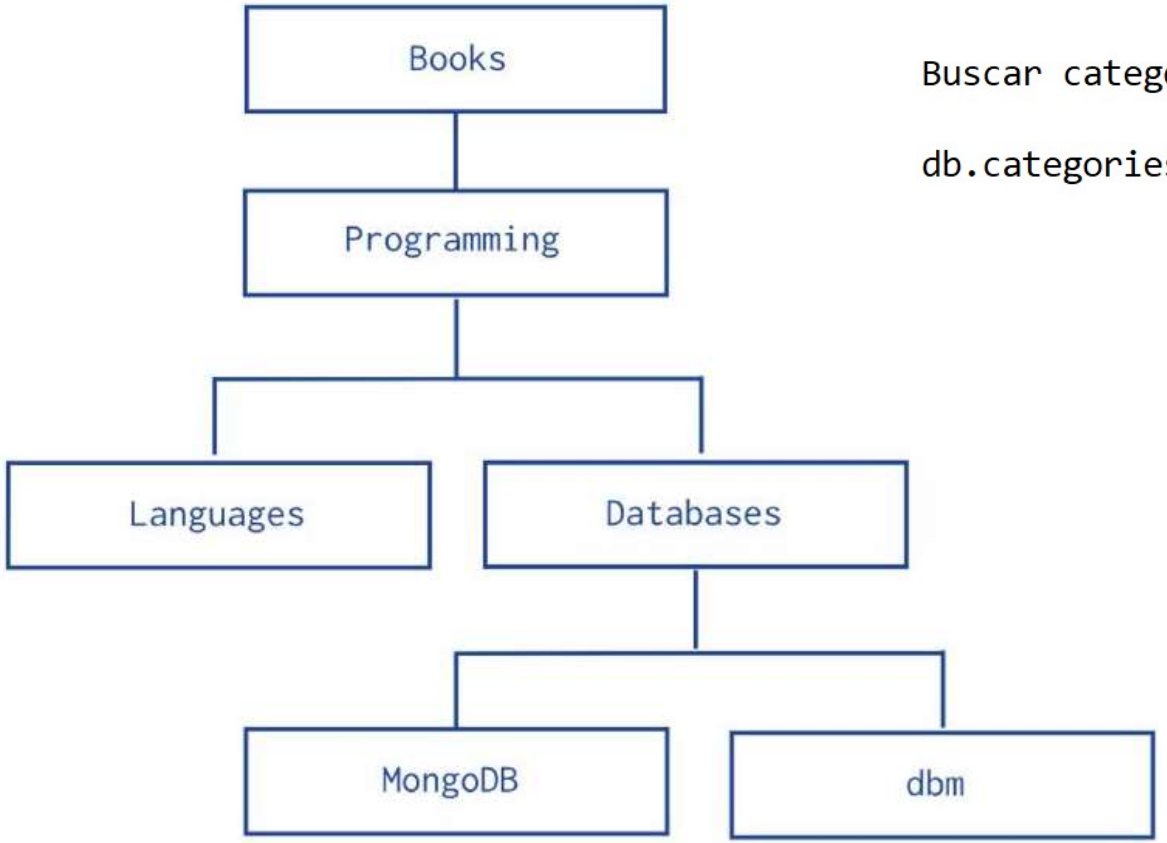
Techniques:

La denormalization, que coloca todos los datos necesarios para responder una consulta, en un solo lugar, normalmente una sola tabla de base de datos, en vez partir los datos en múltiples tablas.

Los aggregates, usan poca o ninguna validación de los tipos de datos, por ejemplo strings o integers.

Los joins se realizan a nivel de aplicación, no como parte de una consulta de base de datos. Esto requiere mayor planificación.

Los indexes y key tables para identificar y ordenar datos de forma rápida para recuperación.
Tree structures modeladas como una sola entidad de datos, por ejemplo, *comment* con todos sus *responses*.



Buscar categoria

```
db.categories.findOne( { _id: "MongoDB" } ).parent
```

```
db.categories.insert( { _id: "MongoDB", parent: "Databases" } )
db.categories.insert( { _id: "dbm", parent: "Databases" } )
db.categories.insert( { _id: "Databases", parent: "Programming" } )
db.categories.insert( { _id: "Languages", parent: "Programming" } )
db.categories.insert( { _id: "Programming", parent: "Books" } )
db.categories.insert( { _id: "Books", parent: null } )
```

relacion 1 a 1

```
{
   _id: "joe",
   name: "Joe Bookreader",
   address: {
              street: "123 Fake Street",
              city: "Faketon",
              state: "MA",
              zip: "12345"
            }
}
```

relacion 1 a muchos

```
{
   _id: "joe",
   name: "Joe Bookreader",
   addresses: [
                {
                  street: "123 Fake Street",
                  city: "Faketon",
                  state: "MA",
                  zip: "12345"
                },
                {
                  street: "1 Some Other Street",
                  city: "Boston",
                  state: "MA",
                  zip: "12345"
                }
              ]
 }
```

Uso de referencias a varios

```
{
   name: "O'Reilly Media",
   founded: 1980,
   location: "CA",
   books: [123456789, 234567890, ...]
}
```

```
{
    _id: 123456789,
    title: "MongoDB: The Definitive Guide",
    author: [ "Kristina Chodorow", "Mike Dirolf" ],
    published_date: ISODate("2010-09-24"),
    pages: 216,
    language: "English"
}
```

```
{
   _id: 234567890,
   title: "50 Tips and Tricks for MongoDB Developer",
   author: "Kristina Chodorow",
   published_date: ISODate("2011-05-06"),
   pages: 68,
   language: "English"
}
```


```
{
   _id: 234567890,
   title: "50 Tips and Tricks for MongoDB Developer",
   author: "Kristina Chodorow",
   published_date: ISODate("2011-05-06"),
   pages: 68,
   language: "English"
}
```

RESUMEN

Recordemos

Relational (SQL) Database Design

Non relational (NoSQL) Database Design

REFERENCIAS

Para profundizar

https://www.vivekmchawla.com/erd-crows-foot-relationship-symbols-cheat-sheet/

PREGRADO

Ingeniería de Software
Escuela de Ingeniería de Sistemas y Computación | Facultad de Ingeniería

UPC
Universidad Peruana
de Ciencias Aplicadas

Prolongación Primavera 2390,
ffionterrico, Santiago de Surco
Lima 33 - Perú
T 511313 3333
https://www.upc.edu.pe

exígete, innova

LAUREATE INTERNATIONAL UNIVERSITIES