Data Modeling

Exploring best techniques and best practices for turning data into something that represents the data.

Data models help us to organize our app’s data.

1. Five steps to creating a data model:
   1. Identify entities
   2. Identify relationships
   3. Identify attributes
   4. Assign keys
   5. Normalization
2. Identifying Entities
   1. This is what we care about storing in our database – basically, think about our tables in SQL or something like that.
   2. There are 4 types of entities.
      1. People
      2. Things
      3. Events
      4. Locations
   3. Something that is not one of these might be an attribute, not an entity.
3. Example – if we are creating a database for a chain of stores, what would we have to keep track of?
   1. People
      1. Customers
      2. Employees
      3. Suppliers/Vendors
      4. Partners
   2. Things
      1. Products
      2. Equipment
      3. Advertisements
   3. Events
      1. Sales (discount days)
      2. Purchases and Sales
      3. Returns/Refunds
   4. Locations
      1. Shops
4. Identify our relationships.
   1. High-level, not specific talk about keys.
   2. Is it one to one? One to many? Many to many?
5. Cardinality:
   1. Is it one to one? One to many?
   2. Shows how much of one side of the relationship belongs to the other side of the relationship.
      1. Customer->sales
         1. 1 customer can make many sales. (one to many)
         2. 1 sale can only belong to one customer. (one to one)
         3. This relationship is one to many.
      2. Customer -> products
         1. One customer should be able to buy many products.
         2. One product should be available to many customers.
         3. This relationship is many to many.
      3. Customers->shops
         1. 1 customer can go to different locations
         2. One location can serve many customers
         3. This relationship is many to many.
      4. Etc etc etc….
   3. After identifying the relationships, we can start drawing up a high level diagram of what our database can look like.
      1. Information Engineering diagram (using an ER diagram)
      2. +-- represents “one”
      3. O(triangle) represents “many”
   4. The relationship to “customer” to “products” is made twice, because a **customer only has a relationship to a product through a sale.**
      1. So the sale itself tracks the customer’s relationship to the product.
   5. Many to Many relationships
      1. Golden Rule: a many to many relationship often calls for a join table.
         1. Product 🡨🡪 shops
            1. Join table of “stock” (what products are at which shop)
         2. Products 🡨 🡪 sales
            1. Join table of sales\_item
6. Identifying Attributes
   1. Let this brainstorm process be free-flowing.
   2. Throw out attributes as you imagine them necessary.
   3. It’s going to be quite important to be thorough, because we will need to be considerate of our potential needs.
   4. Identify redundant data
      1. Suppose we have 3 items all manufactured by the same manufacturer.
      2. We can eliminate this possibility for redundant data.
      3. Well… **the manufacturer should be its own entity.**
7. Assigning Keys
   1. Select the attributes that uniquely identify each record.
   2. Anything can be using keys
   3. Sometimes, this can be used by using multiple attributes as a key (a composite key)
      1. Composite keys would be most common in join tables
      2. And composite keys would have be set up within our schema to use particular columns as a composite key pair.
      3. If we have a composite key as a primary key,
         1. The foreign keys must also be identified as a pair.
   4. Foreign Keys
      1. Used to create relationships between our tables
      2. And used to enforce referential integrity
         1. All foreign keys **must** point to a valid primary key.
         2. You **cannot** **delete any record that has a foreign key pointing to it.**
8. Normalization
   1. Making our model flexible and reliable.
   2. Reduce, possibly eliminate redundancy.
      1. This is our key goal.
   3. Three forms:
      1. First form of normalization –
         1. **There may be no repeating groups of columns in an entity.**
         2. If you wish that you could use an array, create a new table and establish a link via a one-to-many relationship.
      2. Second Form of normalization –
         1. **All attributes of an entity should be fully dependent on the whole Primary Key**
         2. The date of the **sale** is more important than the **date that each item was sold.**
         3. Rather than the date that the product was sold,
         4. Move the date to mark the date for a particular sale.
      3. Third Form of Normalization –
         1. **All attributes need to be directly dependent on the primary key, and not on other attributes.**