Creating a database design often comes with a lot of questions. Some of them are “what are the information should be stored?”, “how often a particular data gets taken?”, and “how data should be fetched?”. These questions are related to the context of data storing and fetching which helps to design the database according to expected query result.  If you are an SQL guy/gal, you automatically respond to use ACID principle, well yes it is valid, the architecture of well-known RDBMS is different from DynamoDB (NoSQL). Unlike RDBMS, you can create tables that often represent as entities, and different tables can have relationships. On the other hand, DynamoDB doesn’t support table relations, and well yes, we can create tables for each entity, querying different tables to get the actual data is cumbersome and limits the advantage of DynamoDB architecture.

Problem:

* DynamoDB closely guards against any operations that won't scale, and there's not a great way to make relational joins scale. Rather than working to make joins scale better, DynamoDB sidesteps the problem by removing the ability to use joins at all.
* N + 1 query problem: Example: we want to get both a Customer record and all Orders for the customer. Many developers apply relational design patterns with DynamoDB even though they don't have the relational tools like the join operation. This means they put their items into different tables according to their type. However, since there are no joins in DynamoDB, they'll need to make multiple, serial requests to fetch both the Orders and the Customer record.

The solution: pre-join your data into item collections. Now when we want to fetch the User and Orders, we can do it in a single request without needing a costly join operation

**Reasons to use multiple tables in DynamoDB**

You have multiple needs for DynamoDB Streams

You want an easier export for analytics

**When not to use single-table design**

* in new applications where developer agility is more important than application performance;
* in applications using GraphQL.

<https://www.alexdebrie.com/posts/dynamodb-costs>

https://nordcloud.com/tech-community/problems-with-dynamodb-single-table-design/