Integrate.io

6 Database Schema Designs and How to Use Them

September 03, 2021

By Mark Smallcombe

In this guide, we'll discuss what a database schema is, six database schema designs, and how and why they are used.

We know a lot of thought goes into database construction. Before creating any database, developers plan what it should include and how the different aspects work together. This planning ensures a database has the necessary design for its intended use. Coders then use the schema to implement the database's design.

Database schemas are the blueprints that help developers visualize how databases should be built. They provide a reference point that indicates what fields of information the project contains. If there is any issue, the developer can simply refer to the schema.

Integrate.io

ensures that all the stakeholders have fully considered every aspect of the project, which reduces the need for change.

Consider the following six database schema examples.

The six database schema designs covered in this article are:

1. The flat model is for small, simple applications.

2. The hierarchical model is for nested data, like XML or JSON.

3. The network model is useful in mapping and spatial data, also for depicting workflows.

4. The relational model best reflects Object-Oriented Programming applications.

5. The star schema and snowflake schema are for analyzing large datasets.

Choosing the correct database schema can ease a lot of anguish and heartache throughout the life of a software project. An incorrect schema design can lead to debilitating bottlenecks in an application and can be costly to refactor. For example, if you didn't realize early on that your application would rely on several table JOINs, your service will eventually grind to a halt when you reach a certain number of users and data.

To resolve this, data will likely have to move to new tables, code will have to point to those new tables, and then those tables will need the proper JOINs. This means that you will need a very strong test environment (database *and* source code) to test your changes, a plan to manage data integrity, and a plan for updating your

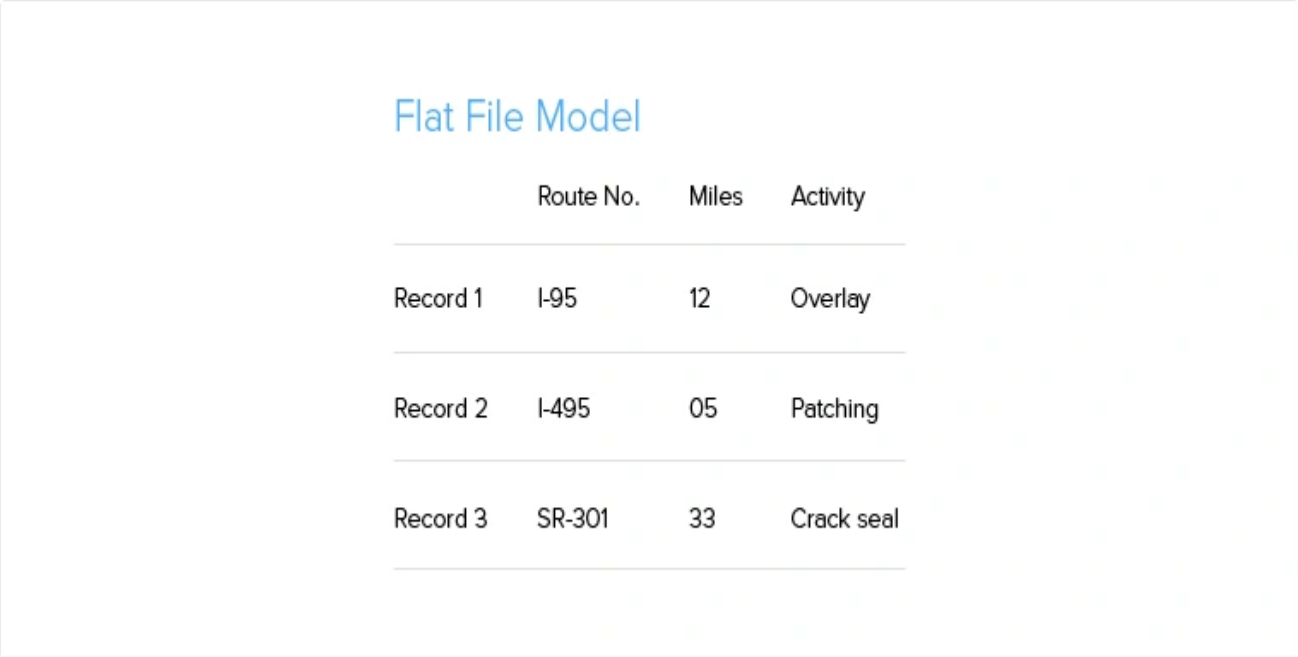
Integrate.io

Table of Contents:

Flat Model

Integrate.io

Think of this as a single, unrelated database table, like in an Excel spreadsheet. If you run a small business with a handful of employees and you want to store only their salary information, then a single, flat data model will suffice. This model abides by the KISS principle.

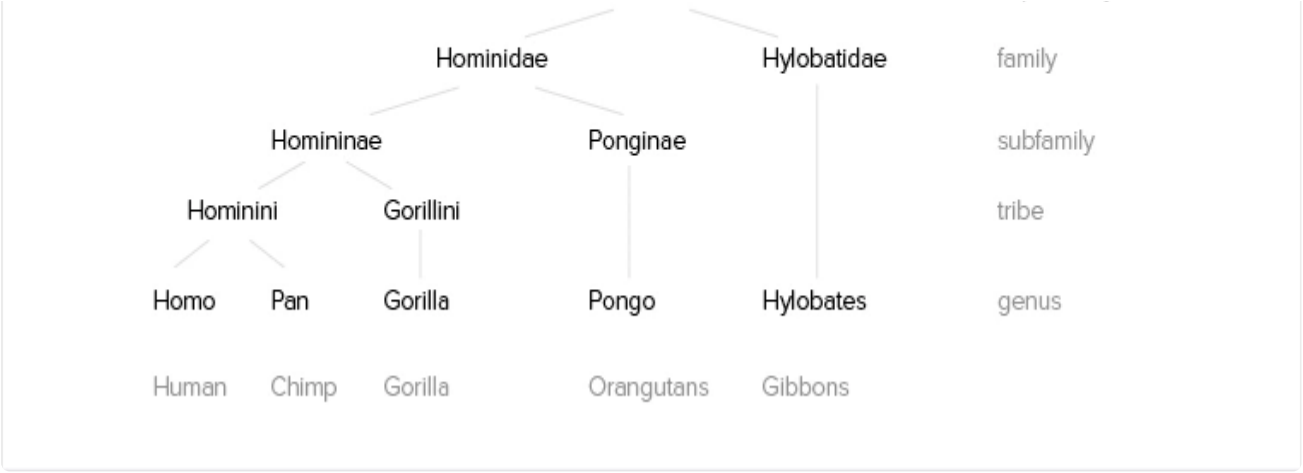


Hierarchical Model

Hierarchical models have a tree-like structure, with a "root" node of data and child nodes that branch out from that root. There is a one-to-many relationship between parent and child nodes. This type of data schema is best reflected in XML or JSON files, where an entity can have sub-entities that are not shared with other entities.

Hierarchical models are great for storing nested data, such as the study of taxonomy.

Integrate.io



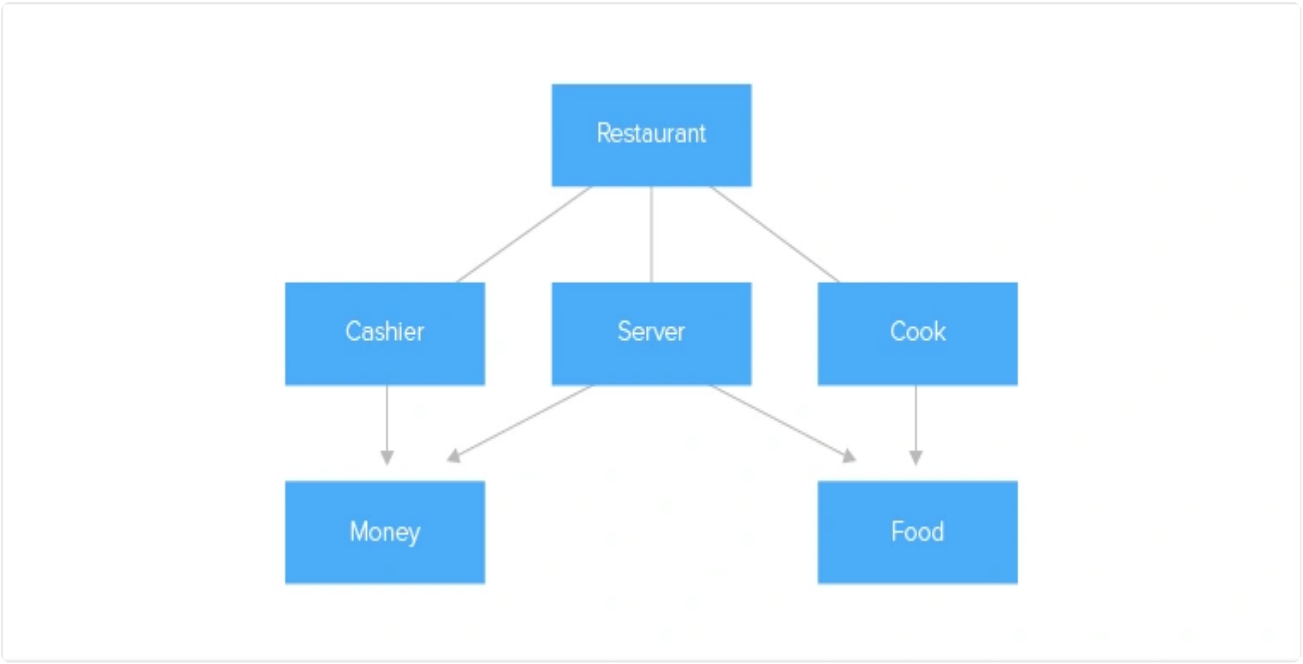
Network Model

The network model is like the hierarchical model in that it represents a series of nodes and vertices; however, unlike the hierarchical model, it allows for many-to-many relationships. From a theoretical standpoint, this means that the graph can have cycles. A cycle in the graph indicates that there is a path of vertices in which you can start and end at the same node.

Billions of dollars lie in a company's ability to efficiently move its goods from point A to point B, and thus, a deep understanding of how to apply the network model is vital. Most applications that need spatial calculations would likely benefit from having data stored inside of a network-modeled database. GIS, Geographic Information Systems, is software that enables users to efficiently store and analyze mapping data.

A network model is also useful when depicting workflows, especially when there are multiple paths to the same result. Take a restaurant chain for example, where a typical workflow is a server telling the cook what to make - let's say a burger with fries. The cook will whip up a delicious hamburger with all the fixings, fry up a salty batch of french fries, put it all on the plate, slap it on the counter, and announce "Order up!" The server will grab the plate and do a final quality assurance test to

Integrate.io



Relational Model

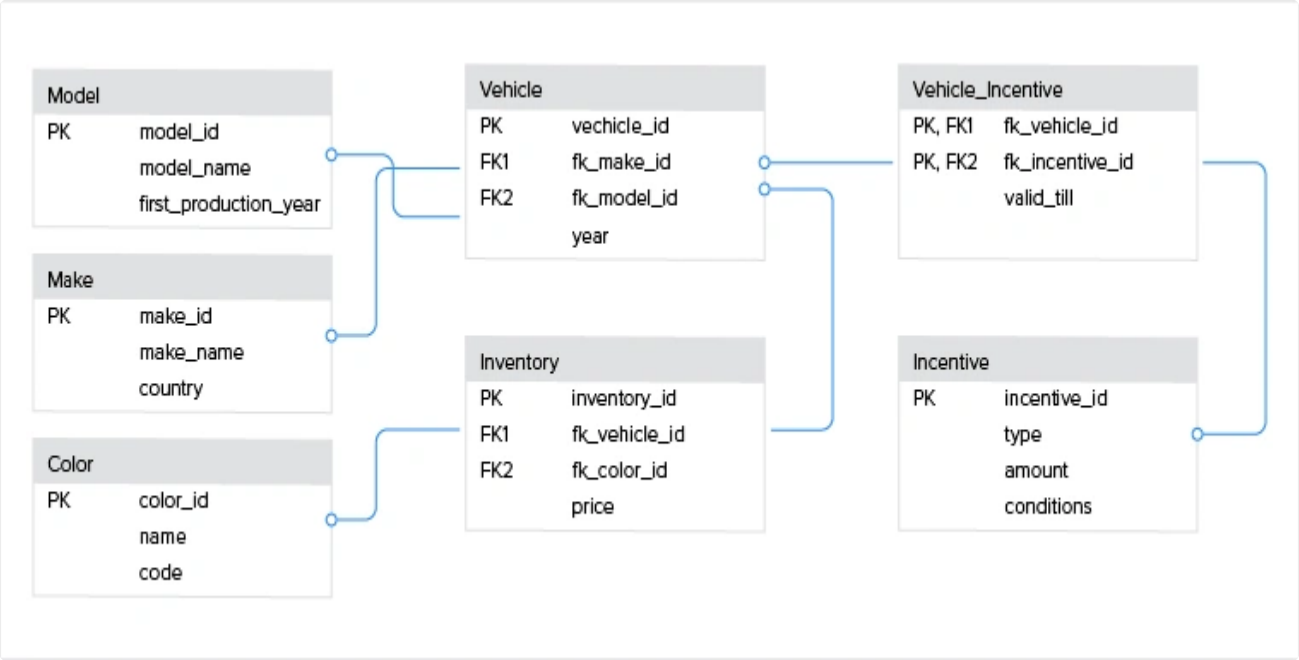
Relational databases are best thought of as a series of entities, some of which relate to each other in certain ways. It's important to think of them as their own, concrete beings.

If you are building out a piece of software that is following the Object-Oriented Programming approach, it would be best to store each object's data as its own table with the database. For example, if you're programming a car, you might have an Object for the tires, axles, engine, seats, paint, etc. The tires attach to the axles which spin because of the engine, and so on. Representing each of these objects as their own table, with a link between the appropriate entities (tire to axle, axle to engine, etc) would be an optimal way to neatly store data and understand how the car works.

The introduction of the relational database model ushered in a new era of data processing. Interestingly, the inventor of the relational database, Edgar Codd from

Integrate.io

more universal understanding of what a relational database is. That is, we store data as relations (i.e. tables), and there are relational operators that we perform on the data to manipulate and calculate things from it. People use Relational Database Management Systems to manage their relational databases. See our deep dive on RDBMSs here.



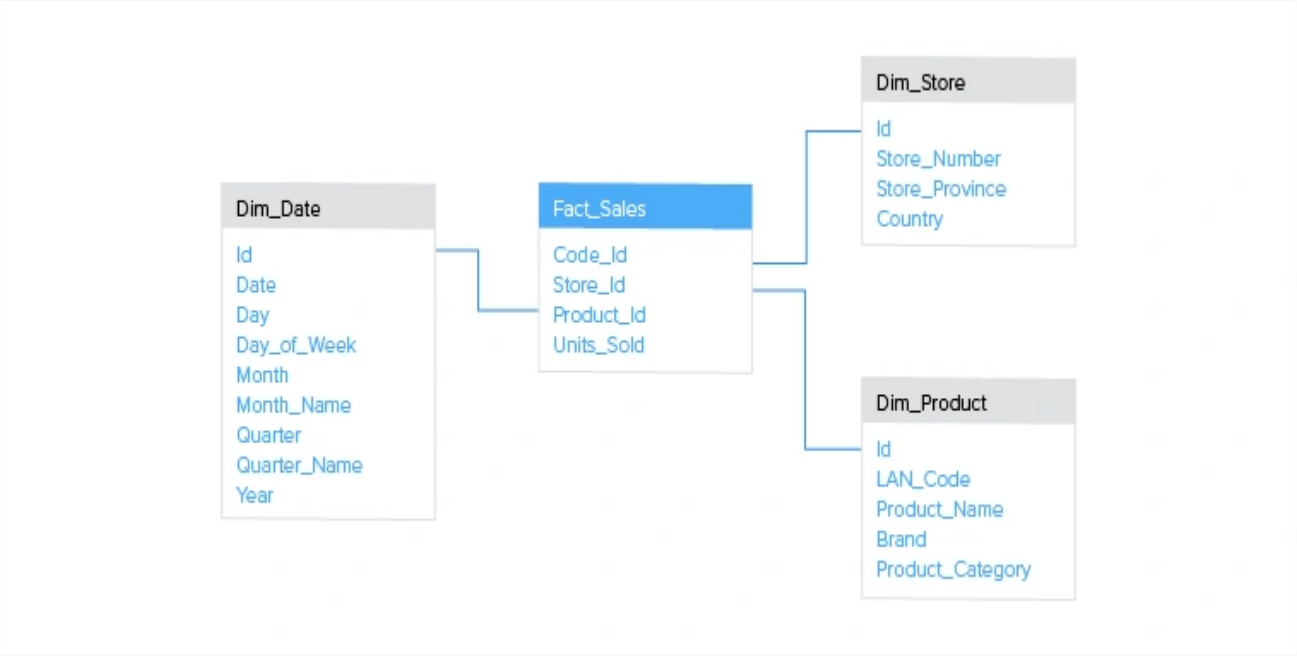
Star Schema

The star schema is a different way of organizing your data. It is an excellent design approach for storing and analyzing massive amounts of data, and it relies on the usage of "facts" and "dimensions."

A "fact" is a numerical data point that drives business processes, and a "dimension" is a description of that fact. Using car sales numbers, for example, the "fact" table would contain information about the number of units sold, and a corresponding "dimensional" table would have the colors of those cars.

Related Reading: Snowflake Schemas vs. Star Schemas

The cool thing about star schemas is that they're simply abstractions on top of traditional relational databases. That is, if you have an RDBMS, you can use it to structure your data into a star schema.



Snowflake Schema

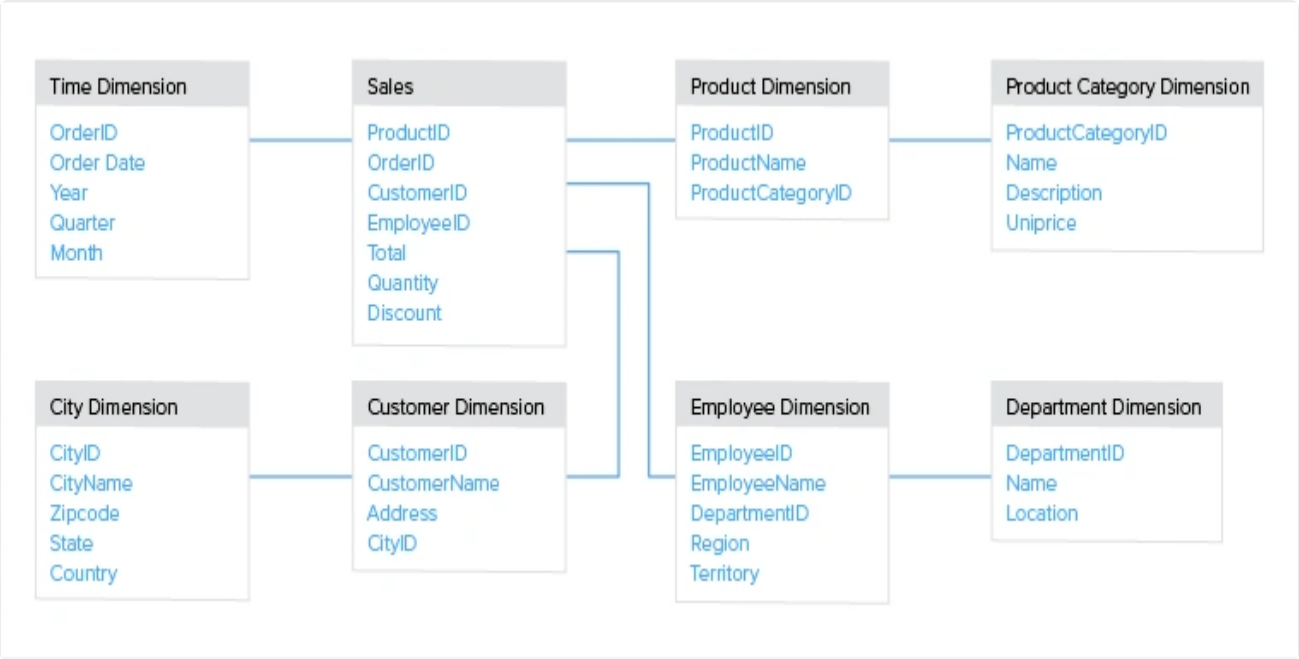
As the star schema is an adaptation of the relational database model, the snowflake schema is an adaptation of the star schema. Its name derives from

Integrate.io

As with the star schema, the snowflake schema has a central fact table that stores the main data points and references to its dimensional tables. Unlike the star schema, the snowflake schema dimensional tables can have their own dimensional tables, thus expanding how descriptive a dimension can be.

Using our car database design example, let's say the operations department needs to forecast which resources they'll need for building their cars. Like the sales department, they will want to know which cars have been selling, and how many. In the star-schema example above, we had a dimensional table indicating the color of the cars sold. The operations department might want to know more about the paint other than color: brand, cost, number of coats, and so on. In this scenario, a snowflake schema would be useful because the color dimensional table requires its own dimensional tables (paint brand, cost, number of coats, etc).

You can choose among any of these database schema examples to structure your data design project. Your database schema's size and complexity depend largely on the size of your project. The database's visual style will allow programmers to structure the information and their relationships properly before they dive into the code.



Integrate.io

complex applications to processing massive datasets, we've got you covered. Let us help you avoid stressful and costly database migrations. Design and execute data pipelines on the most advanced SecurETL Platform. Schedule a demo today.

Big Data

You might also like our other posts...

Big Data October 28, 2022

Data Science Maturity and Understanding Data Architecture/Warehousing

Bill Inmon, father of the data warehouse, discusses data science and the lack of understanding when it comes to data architecture and data warehousing.

Bill Inmon