Part III - Derived Data

In this final part of the book, we will examine the issues around integrating multiple different data systems, potentially with different data models and optimized for different access patterns, into one coherent application architecture. This aspect of system-building is often overlooked by vendors who claim that their product can satisfy all your needs. In reality, integrating disparate systems is one of the most important things that needs to be done in a nontrivial application.

Systems of Record and Derived Data

On a high level, systems that store and process data can be grouped into two broad categories:

1. Systems of record
   A.k.a source of truth, holds the authoritative version of your data. When new data comes in, it is first written here. Each fact is represented exactly once (typically normalized). If there is any discrepancy between another system and the system of record, the value in the system of record is the correct one.

2. Derived data systems
   Data in a derived system is the result of taking some existing data and transforming it in some way. If you lose this data, you can recreate it from the source. An example is a cache. Denormalized values, indexes, and materialized views fall into this category.

Derived data is often essential for getting good performance on read queries. It is commonly denormalized. Not all systems make a clear distinction between systems of record and derived data in their architecture, but it is a helpful distinction to make, because it clarifies the dataflow in your system.

The distinction between system of record and derived data system depends on how you use your tool in your application.

We will start in **Chapter 10** by examining batch-oriented dataflow systems such as MapReduce, and see how they give us good tools and principles for building largescale data systems. In **Chapter 11** we will take those ideas and apply them to data streams, which allow us to do the same kinds of things with lower delays. **Chapter 12** concludes the book by exploring ideas about how we might use these tools to build reliable, scalable, and maintainable applications in the future.