Learn to build cloud apps with iCloud and Core Data

Beginning iOS Cloud and Database Development

Nathan Ooley | Nick Tichawa | Brian Miller

Apress®

For your convenience Apress has placed some of the front matter material after the index. Please use the Bookmarks and Contents at a Glance links to access them.

Apress®

Contents at a Glance

Introduction

We are really excited for the future of cloud development for programmers. There is no question that the majority of applications we will develop over the next few years will have some type of connection to the cloud.

In this book, we cover each of the basic saving methods in detail. We start with creating container objects and change notifications, and then we cover the basics of data connections. We also discuss Key Values, best user practices, and how to get this lightweight data quickly to your applications. Next, we dive in a little deeper and discuss in detail how to add Documents and Core Data.

Building great iCloud applications also takes a vision of what is possible with early technology, such as iCloud and other cloud services. As we develop with new phases of technology we should always reach for what is next out there and develop those technologies further to see if they fit into our vision and style. We investigate new trends in the industry and what the iOS engineers offer developers to take that technology further.

This book hopefully opens up possibilities that you as a developer can look at to get started developing data-driven applications with iCloud and further develop your skills for adding iCloud to your application.

We also cover issues that network connections have when trying to sync your data. Connectivity is always going to be a variable, never a constant. Connectivity is fluid, as mobile users may experience a slow network, a public network, and an LTE network all within the same public area. These variables have an impact on how you design your methods and code architecture. Before you write that first line of code, you learn how to create your user experience wireframe with every scenario in mind. If there is anything we want you to take from this book, it is design and user experience must be considered on the frontend rather than the backend. It's easy to say that Apple has all these situations figured out, but it's important for you as a developer to differentiate your end-user's experience. We will cover some pitfalls and how to avoid them.

iCloud is relatively simple to integrate. You have great APIs built in that are easy to integrate into the workflow. Once you have the essentials covered in this book and develop your first application, you'll be well on your way to understanding what database integration can bring to your application. I hope that you enjoy the step-by-step approach here. This book can help mold your baseline foundation for connecting your local data to iCloud using Key Values, Documents and Core Data.

Let's get started…

Chapter 1

Cloud Database Development — The Basics

Just so we can start at the same place, I want to cover a quick overview of database development. Since database development can be a book of its own, understanding it is key, as it is the backbone of the journey that will take place in this book. I won't go too extensively into the topic but want to at least cover the basics for getting us started on the path of developing an iCloud application. I expect people reading this book to be somewhat knowledgeable of databases and how they work. This chapter serves as a refresher on databases and sheds some light on the back-end systems we often take for granted. Remote and cloud databases are still a relatively new subject, especially when relating to how they can be used with mobile applications. Understanding how connections are made and the ways they are being used in the modern world are important when it comes time to think through your application and begin coding. While this book focuses primarily on iCloud development, it's also important to be aware of other database platforms such as Azure and MySQL. Most likely, you won't be able avoid these other platforms for long, and cloud databases are becoming so necessary in today's world, having a general understanding of other systems is crucial.

Anyone who's used a program such as Excel will have some experience with data tables. Having data broken down into characters and fields and realizing how those interact with one another gives us a deeper perspective than the average user receives. Also, many of you reading this will also have experience with some database management system or another, but may not be aware of the many functions and capabilities of these systems. By gaining a better understanding of cloud services, as well as the types of databases, you'll be able gain greater clarity on how users will interact with your database, and how it should be structured. Now let's dig a bit deeper into the cloud.

Explanation of a Remote or Cloud Database

Whether or not you like it, many of the mundane actions you perform throughout the day turn into data that is stored and used by various organizations. Whether using an automated calculator for filling out a form on your iPad or ordering a new book on Amazon, all this data gets compiled and

sorted into various databases. As a mobile developer, you will need to consider the architecture of the database you're connecting to and how to access that data remotely.

Businesses store data for primarily financial and legal reasons. Companies and organizations need to collect and store vast amounts of data about their employees' and customers' finances, various habits, and so on. In addition to legal reasons for keeping this information, data can be used to see trends over time, manage inventory, compare with competitors, and so forth. Data allows monitoring and acting on an individual's personal buying habits. The data that is collected is becoming increasingly important as corporations become more global. Companies can capture large amounts of data from all over the world and use it to recognize and act on the trends they see.

This information collected in databases not only needs to be viewed on computer applications, but also has to be accessible on mobile devices. This means data input on a mobile device needs to be continually reconciled with the application server that's hosting the data. An example would be a company that needs to keep track of all their projects and the materials needed to produce those projects. In most cases, there are several devices accessing this data. The database is keeping track of several data sets. For example, an administrator adjusts a data set, which, depending on the variables involved, will adjust other data sets. Because the availability and costs of individual parts to make an end-product are always changing, the administrator can set individual costs, operating costs, and other data that needs to be captured into a working database. They could then potentially see tables, charts, graphs, projection estimates, price suggestions, and so on, all based on this data. This access to the company numbers could be important to a manager. But the information, how it's displayed, and the types of comparisons, graphs, or predictions all depends on the parameters defined when laying out the database. The true value of this data cannot be appropriately valued without making sure that this data is organized and is accessible to the right users. The latter thought, accessibility to the right users, is where the cloud comes in.

A *remote cloud database* is a server that sits on a rack somewhere in the world and is connected to the Internet. It has a server application running and has extra hardware. It is connected to high-speed Internet. So when I talk about data services and about "connecting to the cloud," I am talking about connecting to the server. The critical task of every server is to create a connectable directory that has different types of data stored. The cloud distributes that data quickly. Running a powerful server with the latest hardware has the best capabilities for manipulating, querying, and extracting data and metadata.

Now that we've covered a little bit about cloud databases, let's tie this concept into other kinds of database development.

Types of Database Development

One of the newer cloud services is iCloud from Apple. This book primarily discusses iCloud and the features that iCloud has built into it. Apple has established a very user-friendly system for developers' connections with their service. For example, I mentioned a cloud's ability to quickly distribute data and hardware that is capable of manipulating, querying, and extracting data and metadata. One of iCloud's nicest features is how it handles your file's metadata. This document service makes sure to always push metadata to the cloud ahead of the data that is actually changed in the file. This means that your application is aware of the files that are available to it before the file data has been completely pushed to iCloud or downloaded to a device.

We discuss database development, but more specifically we'll be talking about how that development is best worked into an online and data-driven application. When I talk about applications in this book, I will be primarily speaking of mobile applications. That's not to say that you couldn't develop a desktop or web application to access the database, it's just not something I will be covering in this book.

Database Platforms and Services

I spoke earlier of data management platforms used with businesses. There is a variety of these database management system (DBMS) companies that are available for businesses to work with, including SAP, Oracle, MySQL, and Apple. These management systems need to be defined differently from what resides on the actual databases. Data that is managed is always referred to as "actual data." Many of these companies have very different ways of managing their online database, comprising the "services" part of the DBMSs.

Within a database, raw data alone is not useful. Suppose we did a market research study where we interviewed 50 employees and asked each a series of 25 questions. Once each person has answered the questions, you want to compare the results. Say you want to find all the employees who have become dissatisfied about certain issues. How would you know how to make this comparison?

You have to ask yourself how important each piece of data is before putting filters in the database to decipher it. This helps us structure conclusions based on that data. Some information captured and imported into the database is going to be more important than other information. Not only do you need to decide which pieces of data are included, but also which pieces that are included are the most important, and then structure filters and hierarchies accordingly. This is the key to database management.

With iCloud, the data within your document exists in iCloud until you explicitly request that it be downloaded. Then, once the file is downloaded, iCloud propagates any changes to the document down to your device. This function is another great benefit of iCloud. It is handled this way in order to conserve the storage space on your mobile device. So when you edit a document on your device, the document service only pulls down the individual file that is needed for editing, leaving the rest of it on the server. The application is still aware of these other files because of the metadata that is being persisted across devices. This makes iCloud an efficient data management platform that happens to be perfect for mobile devices, which tend to not have as much storage space as a general computer.

In many ways, iCloud extends beyond what is normally considered a data management platform. Not only does iCloud have a cloud server similar to some of the data management platforms listed previously, but it can update documents other ways as well, such as peer-to-peer.

Characters and Fields

Characters are the basic element of data for the purpose of this book. Non-textual data is also considered data but not in the way I will be describing. For purposes of this book, we will be discussing only textual data. Those characters are the building blocks of the field, which are the building blocks of the record and the table.

A *field* contains an item of data—a character, or group of characters that are related. For example, a grouping of related text characters such as “Erin” makes up a first name in the name field in Figure 1-1.

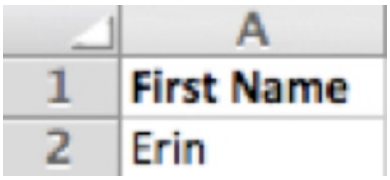

Figure 1-1. *Example of a field (First Name) and four characters (Erin)*

For each person in our example, we must identify the name, address, city, state, zip code, and telephone number. A field is established for each type of information in the list. The First Name field contains all the letters of the first name. The zip code field holds all the digits of a person’s zip code, and so on. In summary, a field may contain an attribute (e.g., employee salary) or the name of an entity (e.g., person, place, or event).

Records and Tables

A *record* is composed of a group of related fields. It contains a collection of attributes related to an entity, such as a person or product. When all gathered, it contains the most important information for you to be accessing.

As shown in Figure 1-2, you have the name, address, zip code, and telephone number of a single individual that would constitute a record. This group of five records becomes a table of the database.

A	B	C	D	E
First Name	**Last Name**	**DOB**	**Address**	**Social Security**
Erin	Gibb	3/2/54	1201 Westmore Dr	664-20-2345
David	Olsen	3/2/58	33002 Addison Lane	433-41-4978
Jack	Olson	2/3/89	3044 Berrymore Dr	446-59-1102
Mary	Stein	12/21/80	1025 Oregon trail	568-43-2322
Ronald	Gross	1/3/92	22 West 1st Street	460-43-5605

Figure 1-2. *An example of 5 records making up a table*

The Database File

A *database file* is defined as a collection of related data. A database contains tables with each family of data. A database file may be composed of a complete list of individuals on a mailing list, including their addresses and telephone numbers. Files are frequently categorized by the purpose or application for which they are intended. Some common examples include mailing lists, customer files, inventory files, or document files.

Organizations and individuals use databases to bring independent sources of data together and store them electronically. Thus, a database is composed of related files that are consolidated, organized, and stored together. One collection of related files might pertain to employee information. Another collection of related files might contain the items in the inventory. As you'll see in the next section, the database model consists of multiple tables.

The Relational Database

Relational databases work on the principle that each table has a key field that uniquely identifies each row, and that these key fields can be used to connect one table of data to another. Thus, one table might have a row consisting of a customer account number as the key field along with address and telephone number. The customer account number in this table could be linked to another table of data that also includes customer account number (a key field), but in this case, contains information about product returns, including an item number (another key field). This key field can be linked to another table that contains item numbers and other product information, such as production location, color, quality control person, and other data. Therefore, using this database, customer information can be linked to specific product information.

The relational database has become quite popular for two reasons. First, relational databases can be used with little or no training. Second, database entries can be modified without redefining the entire structure. The downside of using a relational database is that searching for data can take more time than if other methods are used. Figure 1-3 shows an example relational design.

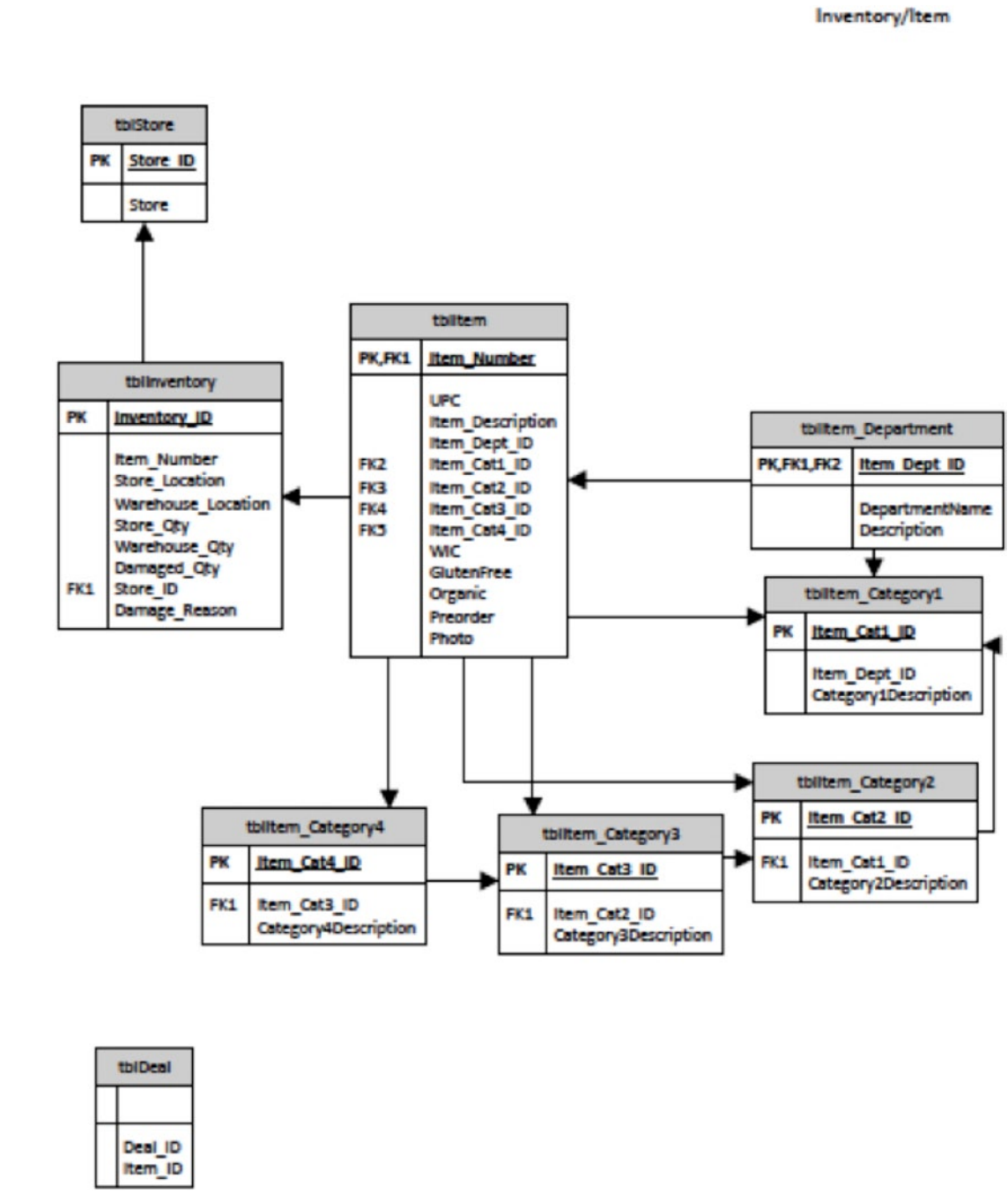

Figure 1-3. *An example of a database design wireframe*

Organizations and individuals may use many different databases depending on the nature of the work involved. For example, a library database might consist of several related, but separate databases, including book titles and author names, book description, books on order, books checked out, and similar sets of information.

Database Management System

A database management system is used to access and manipulate data in a database. On a basic level, it's a software system that enables users to edit, link, and update files as needs dictate. It is a very customizable approach to managing databases and suits a lot of companies because of that reason alone. With this managed approach, a company would need to spend human resources and consulting time to make sure they have support if the software has a bug. This is another good reason to look at some of the newer systems like iCloud. That being said, while it is not as customizable as the other systems, you can create an application that manages some of this for you. That's why you are reading this book, right?

At the core of every good database is organization. As we saw earlier, in order to track and analyze data effectively, each record requires a unique identifier or a "key." The key must be completely unique to a particular record just as each individual has a unique social security number assigned to them (see Figure 1-4).

A	B	C	D	E
First Name	**Last Name**	**DOB**	**Address**	**Social Security**
Erin	Gibb	3/2/54	1201 Westmore Dr	664-20-2345
David	Olsen	3/2/58	33002 Addison Lane	433-41-4978
Jack	Olson	2/3/89	3044 Berrymore Dr	446-59-1102
Mary	Stein	12/21/80	1025 Oregon trail	568-43-2322
Ronald	Gross	1/3/92	22 West 1st Street	460-43-5605

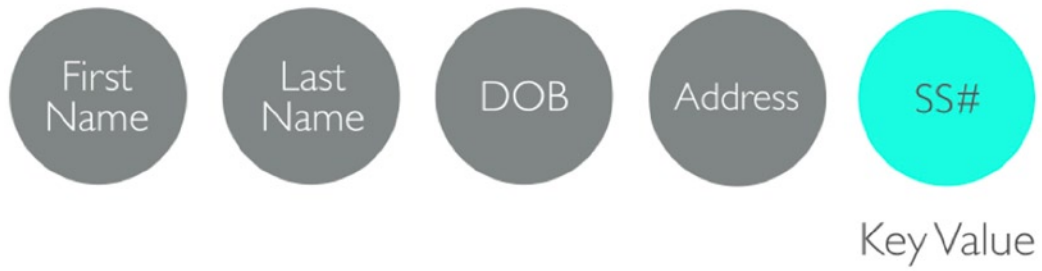

Figure 1-4. *Each circle in this figure represents a field in the table and the SS# is established as the key value*

In fact, social security numbers are often used as keys in large databases. You might think that the name field would be a good choice for a key in a mailing list; however, this would not be a good choice because people might have the same name. A key must be identified or assigned to each record for computerized information processing to function correctly. An existing field may be used if the entries are entirely unique, such as a social security number or telephone number. In most cases, a new field will be developed to hold a key, such as a customer number or product number.

iCloud uses a Ubiquity Identity Token for this function. This is a token that is provided to you by the iCloud Storage API, and it references a particular user on a device within your app. This means that if you are signed in to multiple devices, the Ubiquity Token is different for each device. You can use the Ubiquity Identity Token to first confirm that the user has iCloud set up for their account. It can also be used to determine whether a different user is now using your app on the same device.

Multiple Sources

A database is more useful if there is little redundancy between the files it contains. In other words, it would be inefficient and a waste of human and computer resources to have the same information repeated over and over again in different files. Some companies maintain databases with very similar information. Sometimes there are good reasons for this; e.g. for security purposes. However, it's simply more costly to maintain accurate information in multiple locations. In addition, there would also be a need to resolve discrepancies occurring between the same information in multiple files.

iCloud has an automatic conflict resolution. This means that if you edit the same file on two different devices at the same time iCloud will see that these edits are in conflict. It will resolve the conflict automatically by selecting what it believes is the most up to date file. However, you are also notified of this conflict. The iCloud Storage API also lets you traverse through file versions and resolve the conflict manually if needed.

One of the beauties of cloud databases is linking together data from multiple sources to accomplish a specific task. This can be accomplished on mobile devices, and users can interact with the cloud like they have never done before. Writing applications that utilize these services will become more of the norm going forward in a connected world. Users demand the need to access their data and share with others. Now many users need to have the same experience on their tablet or phone and have confidence that their data is secure and accessible.

The Advantages of a Cloud Database Management System

One of the principle advantages of a cloud management system is that the same information can be made available to different users.

The data in the cloud is more concise because, as a general rule, the information in it appears just once. This reduces data redundancy, or in other words, the need to repeat the same data over and over again. Minimizing redundancy can therefore significantly reduce the cost of storing information on hard drives and other storage devices. In contrast, data fields are commonly repeated in multiple files when a file management system is used.

Accurate, consistent, and up-to-date data is a sign of data integrity. Cloud database management systems maintain data integrity because updates and changes to the data only have to be made in one place. The chances of making a mistake are much higher if you are required to change the same data in several different places than if you only have to make the change in one place.

When using a cloud database management system, file formats and system programs are standardized. This makes the data files easier to maintain because the same rules and guidelines apply across all types of data. The level of consistency across files and programs also makes it easier to manage data when multiple programmers are involved and are working on the same project.

Data is easier to access and manipulate with a cloud solution than without one. As we are usually not far from Wi-Fi and cellular LTE networks, we have the ability to be online at almost all times. In most cases, cloud services also reduce the reliance of individual users on computer specialists to meet their data needs.

As stated earlier, cloud database management services allow multiple users to access the same data resources. This capability is generally viewed as a benefit, but there are potential risks for the organization. Some sources of information should be protected or secured and only viewed by select individuals. Through the use of passwords and transport security, database management systems can be used to restrict data access to only those who are authorized to see it.

The Disadvantages of a Cloud Database Management System

There are two major downsides to using cloud database management systems. One of these is cost, and the other is threat to data security.

Implementing a cloud database management system can be expensive and time-consuming, especially in large organizations. Training requirements alone can be quite costly. This is where I believe application developers like us can create unique experiences for the user to limit this learning curve.

Even with the best safeguards in place, it may be possible for some unauthorized users to access your database. In general, database access is an all or nothing proposition. Once an unauthorized user gets into your database, he has access to all the files, not just a few. Depending on the nature of the data involved, these breaches in security can also pose a threat to individual privacy. Steps should always be taken to regularly make backup copies of the database files and store them because of the possibility of fires and natural disasters that might destroy the system. This is another reason to use a company such as Apple or Microsoft to manage these services so you can concentrate on developing great applications and leave the management of the data to someone else.

Summary

This book is going to really need to be tied in to how iCloud comes out of the box and serves three main features: consumer experience, backup, and availability.

The consumer is very used to having a good experience. By this I mean not replying to several alerts and notifications that are just annoying for most. iCloud doesn't have to ask for passwords, it stores them automatically. This happens in the background, so it is never an issue for the user.

Second, out of the box, we want to make sure that the backup of the data is always available and done automatically because most users don't have that capability or know where to back up the data. iCloud is a seamless integration with backups of user data.

Third,—the last and most important thing—whether it's on my iPad at a meeting, my iPhone on the way home in my car, or later on my laptop, iCloud makes it easy for me to only have the music and videos that I want at any given time. I can download the music and videos from the cloud. This is a data-driven application. When you're creating your application you want to always make sure that you can access that database with all iOS and OS devices. You can manipulate your data on your devices and let iCloud manipulate, store, and compare the data on a server, not your mobile application.

Chapter 2

Introduction to the Cloud

In this chapter, I talk about dynamic data services, specifically iCloud's ability to automatically sync documents across devices, giving users a simple, streamlined, and seamless experience. Because iCloud is simply a server performing a web service, it is certainly possible to have devices continually sync with the cloud using a web service and a SQL database. However, this is much more complicated than in iCloud, as iCloud has all these features built directly in to it, right out of the box. When developing apps natively for iOS, it requires much undue time and effort to work outside the system that has been set up for you. That being said, if an app is not being developed natively, and is instead a hybrid app that contains native and web content, iCloud may no longer be an option. A database such as Azure SQL Database or MySQL may be necessary. Assuming you will be coding natively, this chapter focuses on iCloud and has you syncing documents promptly.

> **Note** The documents I talk about in this chapter should be thought of as captured, compiled data and include things such as set images and graphics. This is unlike Core Data, which can be thought of as raw, uncompiled data. I will cover Core Data in a later chapter.

The Movement to the Cloud

The cloud is a way for us to move data off our local storage system. What was once a novel practice is now becoming common. Having your data stored outside your building or company walls allows for more connection services, as well as some specialized services. For example, iCloud and certain other cloud services enable developers to ensure that once all the data is saved and sustained, it is the same across all devices. One of the best features of iCloud is the simplicity of adding devices and syncing across them.

We live in the second decade of the 21st-century. It feels appropriate that we have reached our capacity for making bigger hard drives and computers. They are becoming less and less common. Instead, companies such as Amazon have started to create server farms where it can keep adding servers or clouds, as companies require their services. At the same time, mobile devices are becoming faster, cheaper, and smaller. Consumption from these devices is increasing dramatically with more and more data being sent across all our mobile devices or desktops daily. Especially within businesses, this is becoming more critical every day. The trend appears to be moving toward online and cloud services because it makes sense with the world in which we now live. We are trying to make cheaper, faster, smaller devices that don't hold a lot of data, but still have an enormous capacity for data consumption and processes that are much faster than the processors that we carry around in our pockets. Even our latest smartphone is faster than a desktop computer from 10 years ago. It has become more and more evident that we need services that store large amounts of data but also give us access to it quickly.

Table 2-1 shows the top ten applications and the amount of usage compared to others. Five of the ten are data-driven applications. The other five are applications for downloading from these and other sites.

***Table 2-1.** Mobile streaming data comparison (Source: Sandvine Network Demographics)*

	Upstream		Downstream		Aggregate	
Rank	Application	Share	Application	Share	Application	Share
1	Facebook	15.43%	YouTube	30.97%	YouTube	28.03%
2	HTTP	13.60%	HTTP	14.37%	HTTP	14.28%
3	SSL	13.26%	SSL	8.92%	SSL	9.49%
4	YouTube	7.91%	MPEG	8.90%	Facebook	7.95%
5	Google Talk	2.23%	Facebook	6.83%	MPEG	7.93%
6	MPEG	1.92%	Pandora Radio	5.15%	Pandora Radio	4.74%
7	Pandora Radio	1.90%	Google Play	3.27%	Google Play	2.96%
8	Skype	1.56%	Netflix	2.69%	Netflix	2.42%
9	SMTP	1.52%	iTunes	1.46%	iTunes	1.34%
10	Yahoo! Mail	1.49%	Flash Video	1.18%	Flash Video	1.05%
	Top 10	60.82%	Top 10	83.74%	Top 10	80.19%

In Table 2-2, you see the trend is the same for all fixed access or desktop users.