(Approx. 1227 words)

Personal Computer Databases  
An Overview  
by Tom Burt, Vice President, Sun City Summerlin Computer Club  
<https://www.scscc.club>

tomburt89134 (at) cox.net

## Introduction

I’ve spent much of my career in software development, working with various database programs, including a stint at Ashton-Tate / Borland as Director of the dBase development team. For this month’s Tech Notes discussion, we’ll do an overview of what database technology is available for regular personal computer users. Interestingly, some of the best database programs are free. The tradeoff is that you have to invest some time learning how to use them.

The basic idea of a database is that it’s a way to save and organize information so that specific pieces can be quickly retrieved on demand for either display or computational analysis. The most common way this is done is to impose some tabular structure (rows and columns) on the data so that a standardized set of retrieval interfaces can be used. Such databases are referred to as “relational” databases.

A relational database is comprised of one or more data **tables**.

* Each table contains one or more **rows** of data.
* Each row includes one or more **columns**.
* Individual columns have a column name, a size, and a data type such as text, integer, float, date, or Boolean.
* One column is commonly designated the **primary key** and contains a unique value that can be used to identify and refer to that row in the table.

It’s possible to link data from one column of a set of rows in a table to the “key” column of a row in another table, creating a relationship between the “master” row in one table and a set of “detail” rows in another. For example: think of the bill for a meal at a restaurant (a row in the Chits table) with the individual itemized entrees, sides, and beverages as details (one or more rows in the Chit\_Items table).

To support data retrieval and other operations (e.g., Add, Update, Delete), relational databases use a special language: **Structured Query Language** (SQL for short, often pronounced “sequel”). Many database programs also have a visual front end that makes it easy to design SQL queries that will retrieve specific subsets of the data.

## Spreadsheets

The most straightforward database programs are traditional spreadsheet tools like **MS Excel** or the free **LibreOffice Calc**. These have a natural rows and columns structure. It’s easy to sort the rows based on values in various columns. You can set filters on multiple columns to display only a subset of the data. A search can also look for specific values in a column or anywhere in the sheet. You can also include computational columns and rows that aren’t part of the actual data but “sum up,” count, average, or perhaps show minimum and maximum values.

Spreadsheets are great for contact lists, asset tracking, inventories, and catalogs. However, they don’t easily support relationships between records in different tables, and there’s no SQL support.

## “Flat File” Database Programs

Commercial database programs like **MS Access**, Word Perfect **Paradox** (formerly a Borland product), and **Apple FileMaker** are more full-featured database tools with true relational support and SQL support. These tools can build relatively sophisticated applications with complex relational data structures, input forms, result viewers, and printable reports. However, these commercial tools aren’t free and take some serious time to learn.

These commercial tools support multiple concurrent users via “file server” locking technology on a local area network. Each user runs their copy of the database “engine” to do reads, inserts, updates, and deletes, which can lead to access contention. Performance can become an issue if more than a few concurrent users access the database file(s).

The screenshot following shows an example of the MS Access visual query designer setting up a query to retrieve a list of club members who attended at least one session of the Internet Investing SIG since January 2017.

Graphical user interface, application, table

Description automatically generated

**MS Access Visual Query Designer**

Once the query has been set up, you can click the View button in the ribbon to see the results as a data table. Note that various selection criteria are defined so that we only get the set of rows we’re interested in.

The equivalent Structured Query Language (SQL) statement would look like this:

**SELECT DISTINCT SCSCA\_Number, Last\_Name, First\_Name FROM ClassSignInList**

**WHERE ((Last\_Name<>"Guest") AND (Valid=Yes) AND (Event\_ID="SI0BC097B") AND (Signin\_Date>=#1/1/2017#)) ORDER BY Last\_Name, First\_Name;**

The SELECT DISTINCT tells the database engine only to return each unique attendee once, no matter how many times that person attended the SIG. Here’s a display of the results of running the SELECT query. I’ve intentionally blurred the results.

Graphical user interface, application, table, Excel

Description automatically generated

**MS Access Results Display  
Members who attended a session of the Investing SIG since 1-1-2017**

I use MS Access and its databases (along with the VB.Net programming system) for the Computer Club’s membership roster and the event sign-in systems, which has worked very well in those contexts.

A good **free alternative** to commercial programs is **LibreOffice Base**. It also has a nice visual front end. However, the native database engine in Base is fairly limited, so it’s best to use Base as a visual front-end to a more sophisticated database engine/server, such as MySQL or PostgreSQL (see below).

## Full-Featured FREE Relational Database Servers

I’ve recently looked at a few free full-featured database server products running on Windows. Of these, Microsoft’s **SQL Server Express**, **Oracle’s MySQL,** and the open-source **PostgreSQL** all combine high-end server functionality and performance with the ultimate low price (FREE). You can run these on the same PC as your end-user applications, or you can run them on another PC acting as a network database server.

At this point, I’m most familiar with Oracle’s **MySQL**, which runs on Windows, Linux, and macOS. I’ve built an experimental version of the club’s sign-in system that uses MySQL databases instead of MS Access. As a true database server engine, its performance on a network is much better than that of MS Access. MySQL is often used as the database engine to support websites. It can scale to retrieve thousands of records per second. Despite this, MySQL was very quick to download and install on my Windows desktop at home. MySQL is accompanied by the **MySQL Workbench** visual front-end, which isn’t quite as nice as MS Access but does a decent job. You can get MySQL at: <https://www.mysql.com>.

**SQL Server Express** is a free download from [www.Microsoft.com](http://www.Microsoft.com). It is a “light” version of Microsoft’s very high-end SQL Server and only runs on Windows. SQL Server Express is high-performance and has an excellent visual front-end for designing and managing databases.

**PostgreSQL** is a free download from [www.PostgreSQL.org](http://www.PostgreSQL.org). It runs on Windows, Linux, macOS, and Unix. In addition, it has a very rich browser-based visual front-end, **pgAdmin**, for designing and managing databases.

## Summary

Depending on your needs and skills, a simple MS Excel or LibreOffice Calc spreadsheet may be the perfect solution for managing your various data collections. Spreadsheets are also easy to back up and share with others.

For more sophisticated users, the flat file database tools may be the way to go – especially if you’re running a small business or supporting a non-profit or charity.

The full-featured database server products may be overkill for a home user, but it’s hard to beat the price. To use them, you will need to learn their features, including a working knowledge of the Structured Query Language.