# Evolution of Databases

An easy explanation of how databases work, and terminology used by database developers

Presented by
Dignity Computer Solutions
October 2017

## Introduction

This presentation looks at how databases have evolved from 1978 to 2018.  It takes you from how data used to be stored/accessed, to the present day methods of allowing for seamless data exchange and interoperability

## A Child’s Game

In 1978, when I was 7 years old, I remember finding out that I could store information on my computer.

My dad said that my report card was printed by the school computer, so I decided to make a program which would store not only my grades, but the grades that I thought all of my stuffed toys would get.

## Flat Files

- In the 70s and 80s, data used to be stored as text files (the kind of file you can open up in Notepad)
- Each piece of data would be separated by commas and quotation marks. These kinds of files were called Comma Separated Value Files (CSV Files)
- Here's an example of how my first database might have looked like:

# Example of a CSV File

```
“NAME”; “ADDRESS”;“SUBJECT”; “GRADE”
“Raja Lamba”; “123 Main Street”;“Math”; “A”
“Raja Lamba”; “123 Main Street”;“Language Arts”; “A”
“Raja Lamba”; “123 Main Street”;“Social Studies”; “A”
“Gerald Giraffe”; “777 Too Tall Ave”;“Math”; “D”
“Gerald Giraffe”; “777 Too Tall Ave”;“Language Arts”; “C”
“Gerald Giraffe”; “777 Too Tall Ave”;“Social Studies”; “A”
“Franklin Turtle”; “420 Fast Lane”;“Math”; “A”
“Franklin Turtle”; “420 Fast Lane”;“Language Arts”; “B”
“Franklin Turtle”; “420 Fast Lane”;“Social Studies”; “C”
```

# Tables, Records, and Fields

- The previous slide shows an example of a **TABLE** of data. A table has rows (each line), and columns (each value).
- Each row is called a **RECORD**. Here are 2 records from the previous slide

  "Raja Lamba"; "123 Main Street";"Social Studies"; "A"
  "Gerald Giraffe"; "777 Too Tall Ave";"Math"; "D"

- Each column is called a **FIELD**. The fields in the previous slide are Name, Address, Subject, and Grade

## Back End and Front End

- Here's some terminology which us database guys always use, but it's actually really simple
- In our example, the table of name, address, subject, and grade would be our **BACK END**. The Back End just <u>stores information</u>
- The <u>program</u> that I wrote which would ask how you wanted to sort it, and displayed it sorted by your choice (something 7 year old me was very proud of) would be our **FRONT END**

## What's Wrong With Flat Files?

- So in our Example of a CSV File, you'll notice how much information is duplicated. What if we wanted to change Franklin Turtle's address to 999 McDavid Street? We would have to change it in 3 places.
- The Front End was hard to program because many lines of code were required to sort Flat Files (nothing was indexed)
- The Back End and Front End were both stored on the same computer, and could only be accessed by 1 person at one time

# Relational Databases

- In the 1990s, a better way of storing data was engineered. Instead of having 1 big file, with many many columns, Relational Databases would group **relevant information together into separate tables**. Each table could then relate to each other through identity values.
- Here's an example of how my CSV File would look like as a Relational Database split into 2 tables.

# Relational Database: Example

STUDENT TABLE:

“StudentID”; “Name”; “Address”
“1”; “Raja Lamba”; “123 Main Street”
“2”; “Gerald Giraffe”; “777 Too Tall Ave”
“3”; “Franklin Turtle”; “999 McDavid Street”

GRADES TABLE:

“StudentID”; “Subject”; “Grade”
“1”; “Math”; “A”
“1”; “Language Arts”; “A”
“1”; “Social Studies”; “A”
“2”; “Math”; “D”
“2”; “Language Arts”; “C”
“2”; “Social Studies”; “A”
“3”; “Math”; “A”
“3”; “Language Arts”; “B”
“3”; “Social Studies”; “C”

Notice how I only had to change Franklins address in 1 spot!

The Student Table and Grades Table are related to each other by the StudentID **KEY**.

StudentID is the primary identifier, or **PRIMARY KEY** in the Student Table

## Relational Database Revolution

- The Relational Database Back End architecture was so successful that it is still being used today. No more text files or CSV files except for data exchange.
- 2 Common Relational Database Back Ends include **SQL Server** and **Oracle**
- With this new revolution in how the Back End was handled, a new way of setting up Front Ends (the user interface) was also developed

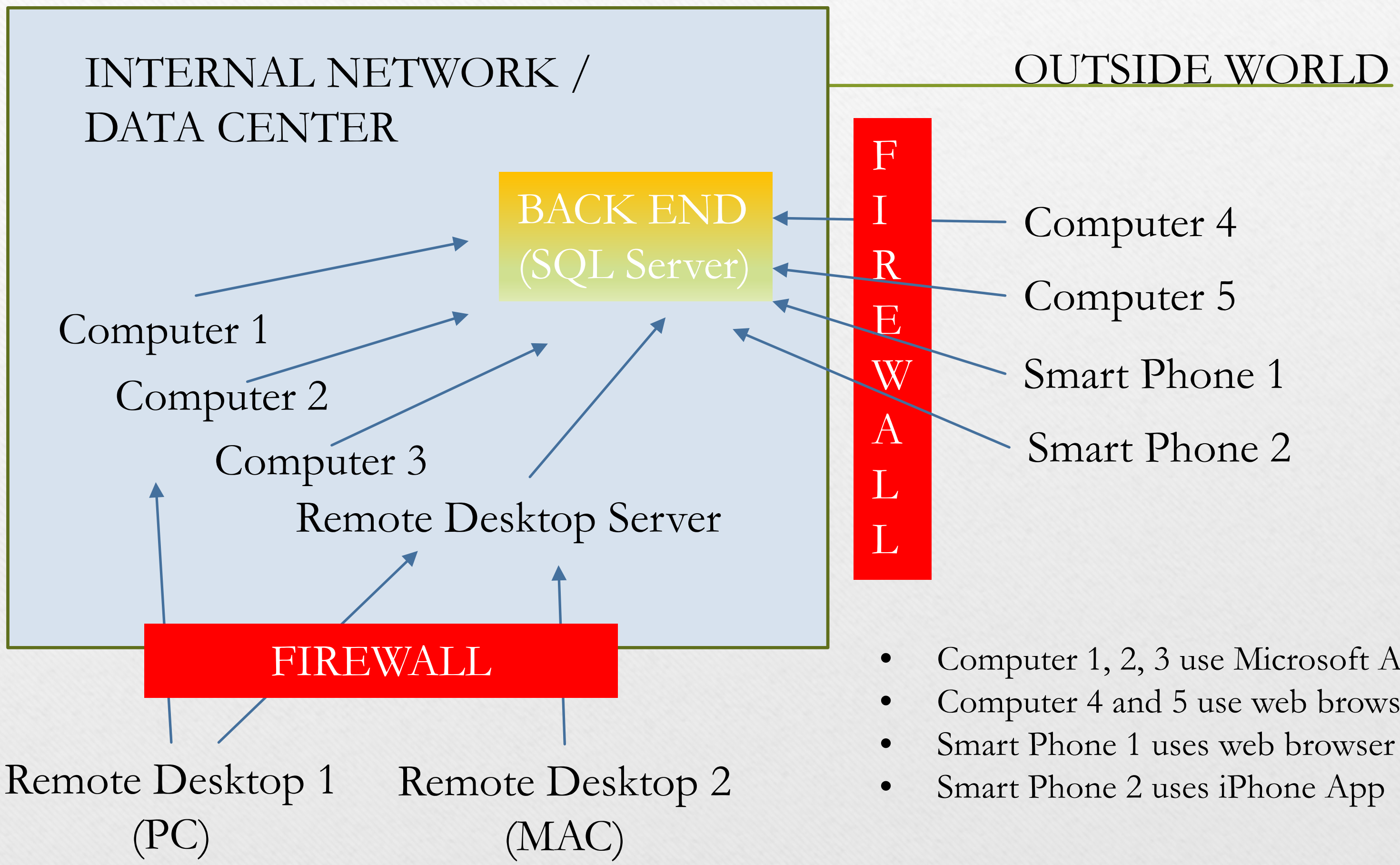
## Client / Server Databases

- So far, the example that we were using had the Back End (data) and Front End (user interface) on one computer which meant that only 1 person could access the data at one time.
- A better way to set up databases was to have 1 central place for the data (usually on a network server) and many different computers accessing the same data at the same time.  This is called a **Client / Server Database**
- An example of this would be a **Microsoft Access Front End Client** with a **SQL Back End Server**.

# Sharing Your Data Off-Site

- Client Server databases work great if you only have 1 network.  But what if you need to access the data off-site?
- One way is to set up a **Web Based Front End** which uses your browser (Internet Explorer, Google Chrome etc) which can also access your SQL Server Back End.
- This method of setting up 1 Back End, and different types of Front Ends (example Microsoft Access AND Web Browser) to access the same data is called an **N-Tier Database**
- Another way to access your database from off site is by using **Remote Desktop Connection**.  This allows you to see your computer off-site and no data is ever transmitted through your web browser.  Many feel that this is more easy to use, and that web based front ends should only be used for e-commerce

# N-Tier Databases



- Computer 1, 2, 3 use Microsoft Access
- Computer 4 and 5 use web browser
- Smart Phone 1 uses web browser
- Smart Phone 2 uses iPhone App

# Data Exchange

- So far our examples have always been using 1 database. What if you want to exchange data between databases? Data can be exported as a CSV File and imported into another database, but there must be a better way.
- **XML** is a technology standard which allows for data mapping between 2 databases, as well as scheduled synchronization between 2 databases.
- XML dealt with data exchange, but couldn't access things like database validation rules etc. **Web Service Technology** allows for seamless data integration between multiple back end databases.

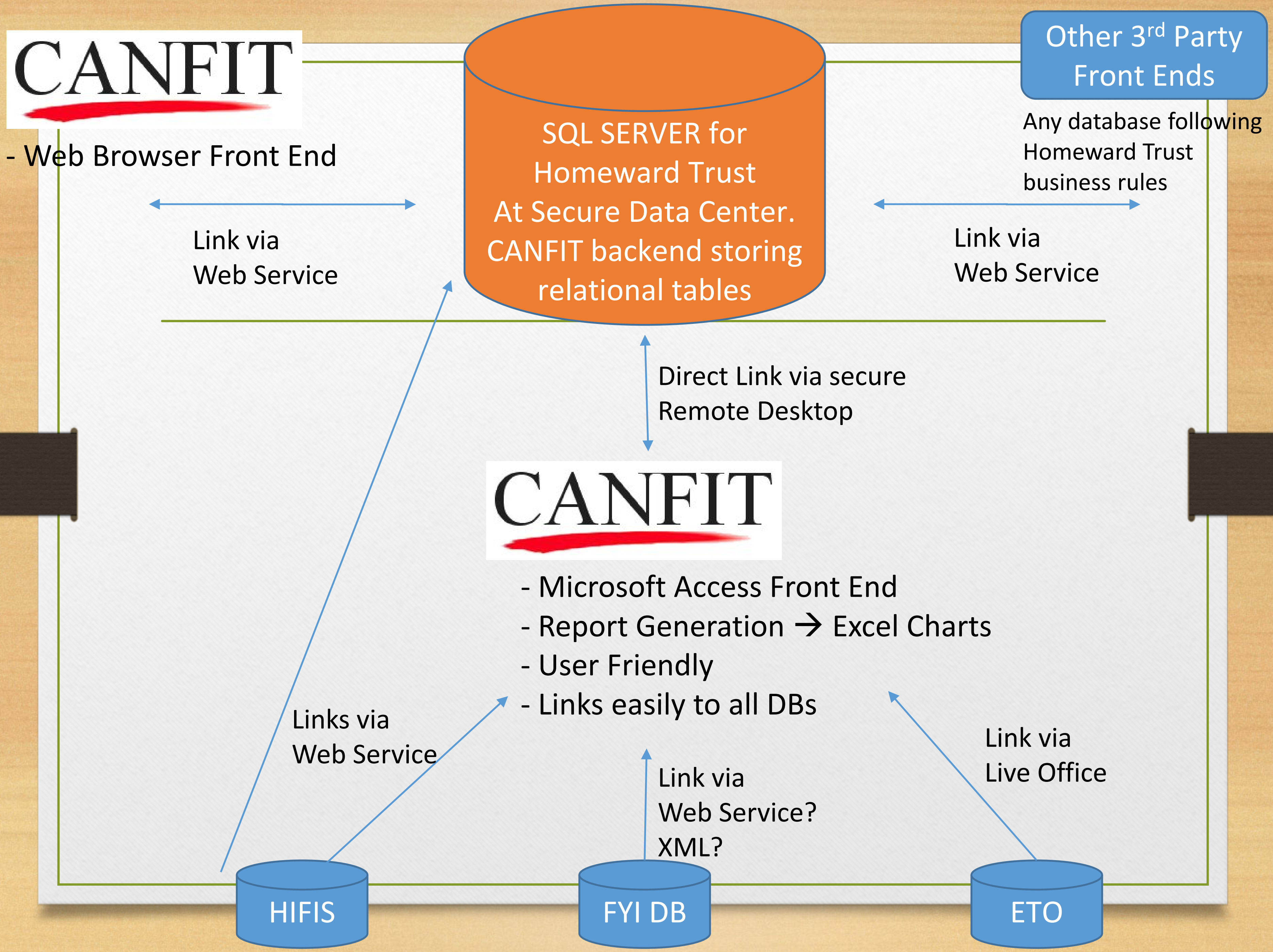
## Web Service Technology

- If a Back End uses Web Service Technology, it allows different Front Ends to access it by allowing the Front End to access it's functions via a secure certificate.
- An example would be Alberta Education who provides 1 central Web Service Back End (Their system is called PASI) and different databases can access it, add records, edit records etc. Whether it's being used by a Head Start Program, or an Elementary, or Post Secondary institution etc, **each institution can use their own database to access the PASI core** without data export or import.
- Web Service Technology uses Secure Certificates which are not issued to the general public, thus making it more secure.

## CANFIT

(Customized Agency Networked Family Information Tracker)

- CANFIT was developed in 2001 by Dignity Computer Solutions as an N-Tier Relational Database, and is now being set up as a master database using Web Service Technology.
- CANFIT is hosted at a secure data center facility in Edmonton. Features of the data center include redundant points of failure, off-site backups, and 99% up time with redundant Shaw, Telus, and Satellite connections. Local Network versions of CANFIT are also in use.
- Over 500 users currently use CANFIT across Canada.
- Millions of records have been entered into CANFIT
- Dignity Computer Solutions has passed Federal Security Clearance for some CANFIT Implementations.
- Here's an example of what we are currently setting up for Homeward Trust


CANFIT
- Web Browser Front End
SQL SERVER for
Homeward Trust
At Secure Data Center.
CANFIT backend storing
relational tables
Other 3rd Party
Front Ends
Any database following
Homeward Trust
business rules
Link via
Web Service
Link via
Web Service
Direct Link via secure
Remote Desktop
CANFIT
- Microsoft Access Front End
- Report Generation → Excel Charts
- User Friendly
- Links easily to all DBs
Links via
Web Service
Link via
Web Service?
XML?
Link via
Live Office
HIFIS
FYI DB
ETO

## Customized Solution

- So what's the most important thing that I've learned since the age of 7? COMMUNICATION. None of this technology will make a difference without good designers and project management to handle expectations.
- What can't CANFIT do? Nothing. If we haven't developed it, we can. The better question is When can CANFIT do it?

## Thank You!

Thank you for listening to my story. I hope that me and my team can have the opportunity to help tell your unique story over the coming years.

DCS

Dignity Computer Solutions

*Special Services for non-profit organizations*

Address: 3400, Manulife Place
City: Edmonton, Alberta
Phone: (780) 429-4525
Toll Free: 1-800-979-4525