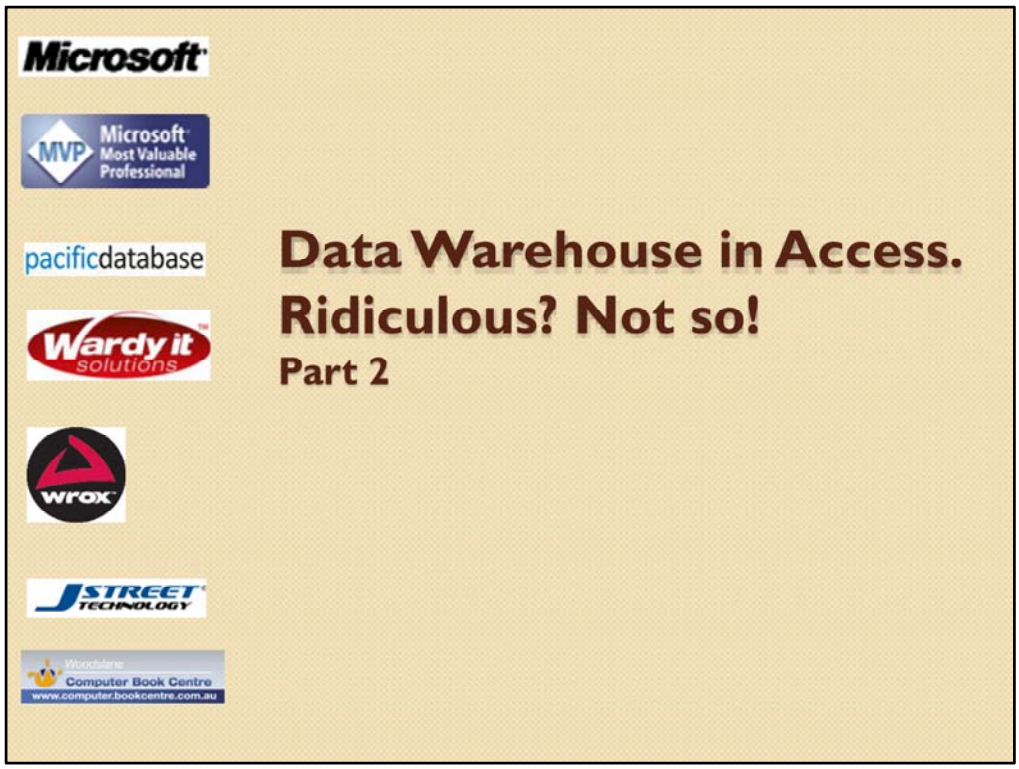

Microsoft
Microsoft Most Valuable Professional
pacificdatabase
Wardy it solutions
wrox
STREET TECHNOLOGY
Woodslane Computer Book Centre
www.computer.bookcentre.com.au
Data Warehouse in Access.
Ridiculous? Not so!
Part 2

Who Am I?

- Roger Carlson
- Access MVP 2006 – 2010
- Senior Analyst for System Quality Analytics at Spectrum Health
- Roger's Access Library
  - www.rogersaccesslibrary.com
  - http://rogersaccessblog.blogspot.com/

Who are you?

- How many here were not in the last session?

I just want a feel for how much I need to recap.

Recap:

- What is a data warehouse?
- Most people think of a data warehouse in terms of storage space and processing power.
- But a data warehouse is really defined by function and topology.

Function and Topology

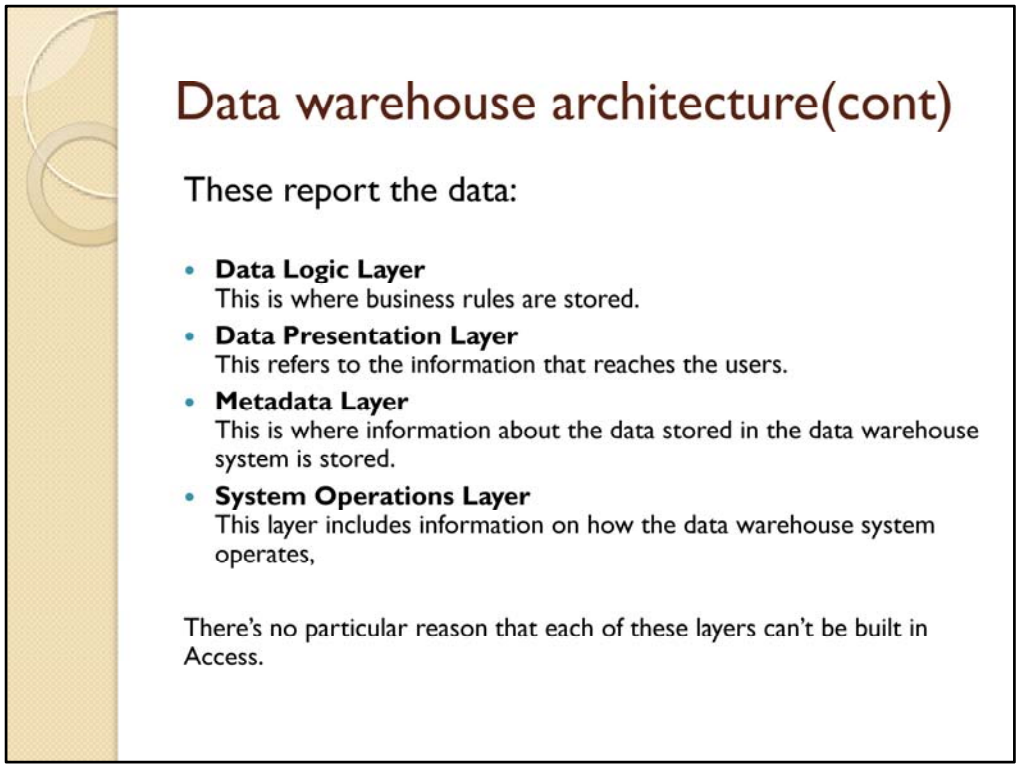
- **Bill Inmon,** a pioneer in data warehousing theory, defined a data warehouse as:
- "A data warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process."
- **Ralph Kimball,** another data warehouse guru, provided a more concise definition of a data warehouse:
- "A data warehouse is a copy of transaction data specifically structured for query and analysis."

Nothing in either definition mentions size or speed. There's nothing that excludes Access.

Data warehouse architecture

These store the data (discussed last time):

- **Data Source Layer**
  This represents the different data sources that feed data into the data warehouse. **Data Extraction Layer**
  Data gets pulled from the data source into the data warehouse system.
- **Staging Area**
  This is where data sits prior to being scrubbed and transformed into a data warehouse / data mart.
- **ETL Layer**
  This is where data gains its "intelligence", as logic is applied to transform the data from a transactional nature to an analytical nature.
- **Data Storage Layer**
  This is where the transformed and cleansed data sit



These layers are the subject of this session:

Data Logic Layer
This is where business rules are stored. Business rules stored here don't affect the underlying data transformation rules, but do affect what the report looks like.

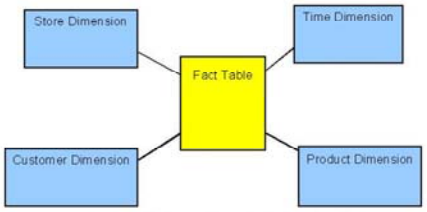
Data Presentation Layer
This refers to the information that reaches the users. This can be in a form of a **tabular /** graphical report in a browser, an emailed report that gets automatically generated and sent every day, or an alert that warns users of exceptions, among others.

Metadata Layer
This is where information about the data stored in the data warehouse system is stored. A logical data model would be an example of something that's in the metadata layer.

System Operations Layer
This layer includes information on how the data warehouse system operates, such as ETL job status, system performance, and user access history.

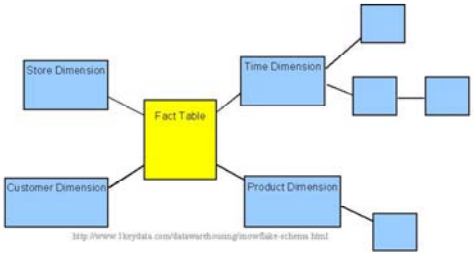
Topology

- ***Star Schema***



http://www.1keydata.com/datawarehousing/star-schema.html

- ***Snowflake Schema***



http://www.1keydata.com/datawarehousing/snowflake-schema.html

Data Warehousing Paradigms

- **Bill Inmon's paradigm**:
  Data warehouse is one part of the overall business intelligence system. An enterprise has one data warehouse, and data marts source their information from the data warehouse. In the data warehouse, information is stored in 3rd normal form.

- **Ralph Kimball's paradigm:**
  Data warehouse is the conglomerate of all data marts within the enterprise. Information is always stored in the dimensional model.

Most data warehouses fall under the Kimball paradigm simply because they started as departmental efforts which were conglomerated into an overall data warehouse at some time.

Implementation

- There is nothing inherent in any of the above definitions, functions, topologies, or paradigms that precludes Access as an implementation platform.

My conclusion from the first session is that there is nothing inherent in any of the above definitions, functions, topologies, or paradigms that precludes Access as an implementation platform. Granted most data warehouses will hold more data than Access is capable of, but that is a limit of scale not of kind. Inmon would call it a data mart rather than a data warehouse, so if you're more comfortable thinking of what I'm doing here as a data mart, that's fine with me.

The Problem

- Every database, whether a data warehouse or not, is created to solve a problem.
- Healthcare Quality Improvement
  - You cannot improve healthcare quality until you can measure it.

one creates a database to answer all questions possible about a problem. As with any other database project, a data warehouse goes through a data gathering and requirements phase.

So before I can talk about a specific data warehouse project, I have to discuss the problem it's meant to solve. For this, I've chosen to discuss a health care related problem, something I deal with every day, but the principles can be applied to any report-intensive application.

I work at the second largest hospital system in Michigan. Despite what you've heard, health care quality improvement is a major initiative. Our system has put a lot of time, effort, and money behind it. We've targeted 40 major conditions that represent 70% of all activity in the hospitals.

Conditions and Indicators

- **Conditions**
  - A condition is a disease, procedures, or surgery.
    - Examples: hip replacement, vaginal delivery, pneumonia, AMI (acute myocardio infarction).
- **Indicators**
  - A measure related to the condition
    - Length Of Stay
    - Age
    - Mortality
    - Aspirin on Arrival
  - Indicator Types
    - Scalar
    - Boolean

Conditions

A condition is a disease, procedures, or surgery.

Condition Examples: AMI, hip replacement, vaginal delivery, heart failure.

Each Condition has a team of doctors, nurses, and other specialists who meet periodically to drive improvement in that condition.

Indicators

But before you can improve something, you have to measure it – know where you are, where you want to go, if or when you ever get there. So each team decides on measurement indicators – things that can be measured – to show whether patient outcomes are improving.

Indicator Examples:
Length of Stay: patient outcomes have been tied to appropriate length of stay. We want patients out of the hospital as soon as possible, but not too soon.
Mortality: obviously we want to know if patients die in our care.
Aspirin on Arrival: research has shown that heart attack patients that receive an aspirin within 30 minutes of arrival at the hospital have better outcomes.
Age: age is an important factor when evaluating the other indicators.

Indicator Types
Boolean values – measures things that either happened or they didn't. A patient died or he didn't. A patient received and aspirin or she didn't.
Scalar values – measure things that have a range of values, like length of stay, age, costs

Reporting

- But within a Condition, we might want to aggregate the data in different ways.
  - Mortality Rate (average) for 2009.
  - Mortality Rate by month for 2009
  - Mortality Rate by race for 2009
  - Aspirin Rate by race by month for 2008-2009
  - Aspirin Rate by physician for 2009
  - Patient list of people who died in March 2009

Another important consideration in a data warehouse system is what you want to get out of it. In almost all cases, we want to see the data aggregated at various levels. The vast majority will be averages (or means), but occasionally medians, counts, or even patient lists.

In all cases, however, we are interested at aggregating data by Condition because the improvement teams are built around conditions. A Condition is also considered a Population, that is, the basic grouping for whom the indicators apply. (This distinction will become important later.) We're not particularly interested in the mortality rate for the hospital over all, but we are interested in the mortality rates for heart attack (AMI) patients because the AMI team can do something about that.

Dashboard: Aspirin, Mortality, Length Of Stay, Beta Blocker

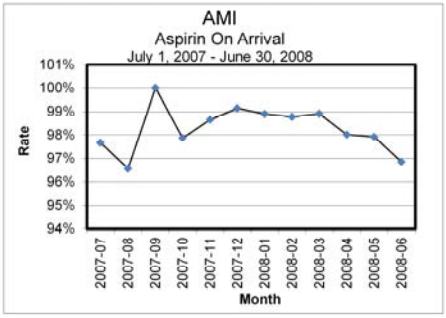
AMI

July 1, 2007 – June 30, 2007

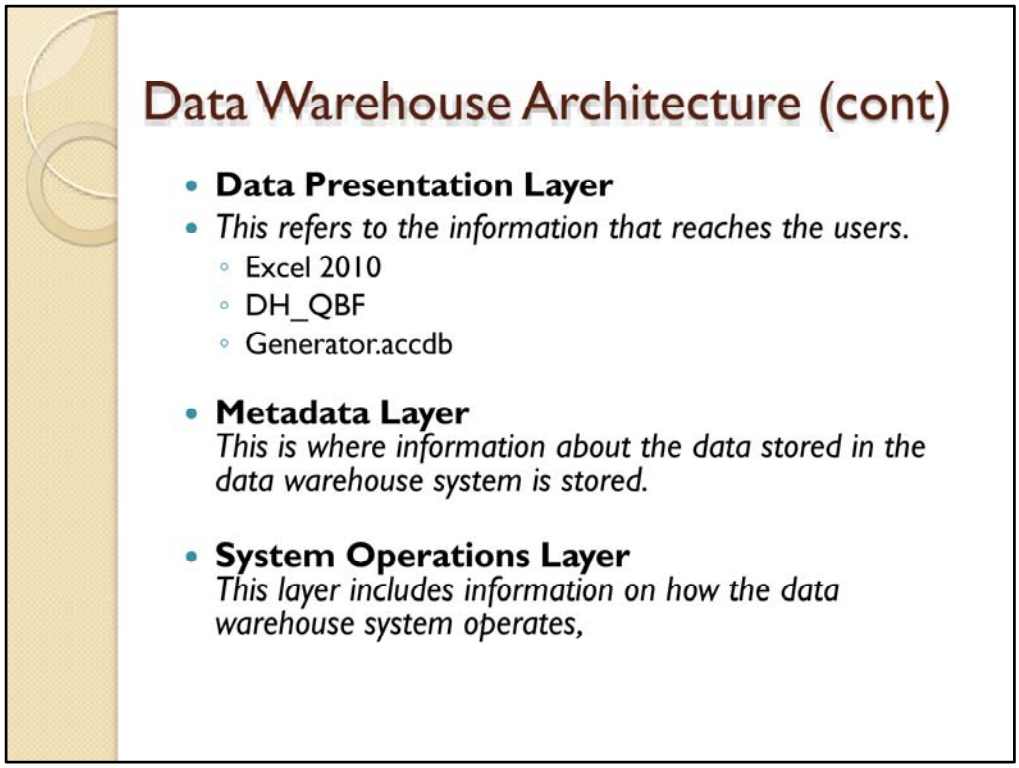
	Aspirin On Arrival	Mortality	LOS	Beta Blocker
Hospital A	53%	16%	3.7	3%
Hospital B	92%	4%	2.7	3%

Chart: Aspirin On Arrival by Month

Month	Rate
2007-07	97.67%
2007-08	96.55%
2007-09	100.00%
2007-10	97.87%
2007-11	98.67%
2007-12	99.15%
2008-01	98.90%
2008-02	98.78%
2008-03	98.91%
2008-04	98.02%
2008-05	97.92%
2008-06	96.84%



Now, this information is surprisingly difficult and time consuming to produce from the Transaction database. First of all, I need to know which patients had an AMI (heart attack), then of those, I need to know which of those had an aspirin given to them and how many did not. Lastly, I have to aggregate the data (average by month).



This time I want to talk about reporting the information I've stored in my warehouse

Data Presentation Layer
This refers to the information that reaches the users.

I'm going to spend the majority of my talk on this layer, so I'll skip it for now

Excel 2010
DH_QBF
Generator.accdb

Metadata Layer
This is where information about the data stored in the data warehouse system is stored.

In this case, the metadata is stored throughout the system. Each database stores information about its layer. The relationships window in DW_Stage is where the data model for the ETL Stage. The individual queries in DW_Stage hold the logic for calculating the fields in the Fact Table (Patient_Indicator).

System Operations Layer
This layer includes information on how the data warehouse system operates,

Again, this information is scattered throughout the various programs. While it is certainly possible to completely automate this process, in practice, I find it useful to run each stage manually, although each stage is as automated as possible. Mostly, this is because my actual system is fed from multiple, external data sources over which I have no control. Sometimes human judgment is the best solution to a problem.

Data Presentation Layer

- ***Three Solutions***
  - ***Duane Hookom's Query By Form*** http://www.rogersaccesslibrary.com/forum/hookom-duane_forum4.html
- ***Excel 2010: Pivot Tables / Pivot Charts***
- ***Custom Report Solution with Office Automation***

Okay, let's return to the presentation layer. Obviously, this is the whole purpose of the data warehouse. It's the part that is often referred to as Business Intelligence or the Decision Support system.

A decision support system will usually two parts: Standard Reports and Ad-hoc Reports. Standard reports can be run by decision makers and they usually give a 30,000 foot view. These are often created by analysts and placed in a library for the decision maker to run. Ad-hoc reports are often run by analysts who understand the system better to answer questions that may not be on the standard reports. Useful ad-hoc reports often become standard reports.

Another key feature of decision support systems is the ability to drill down into progressively finer granularity. For instance, my report may show the Aspirin on Arrival rate per quarter. I may want drill down to the month level. I may even want to drill down to the actual patient identifiers to investigate why the failure occurred.

Three Solutions:
My intent here is not to give you comprehensive solutions or a working utility. My intent is to show several approaches that you can use to create flexible and easy to use reporting services for your data warehouse users.

Duane Hookom's Query By Form

- http://www.rogersaccesslibrary.com/forum/hookom-duane_forum4.html
- DEMO

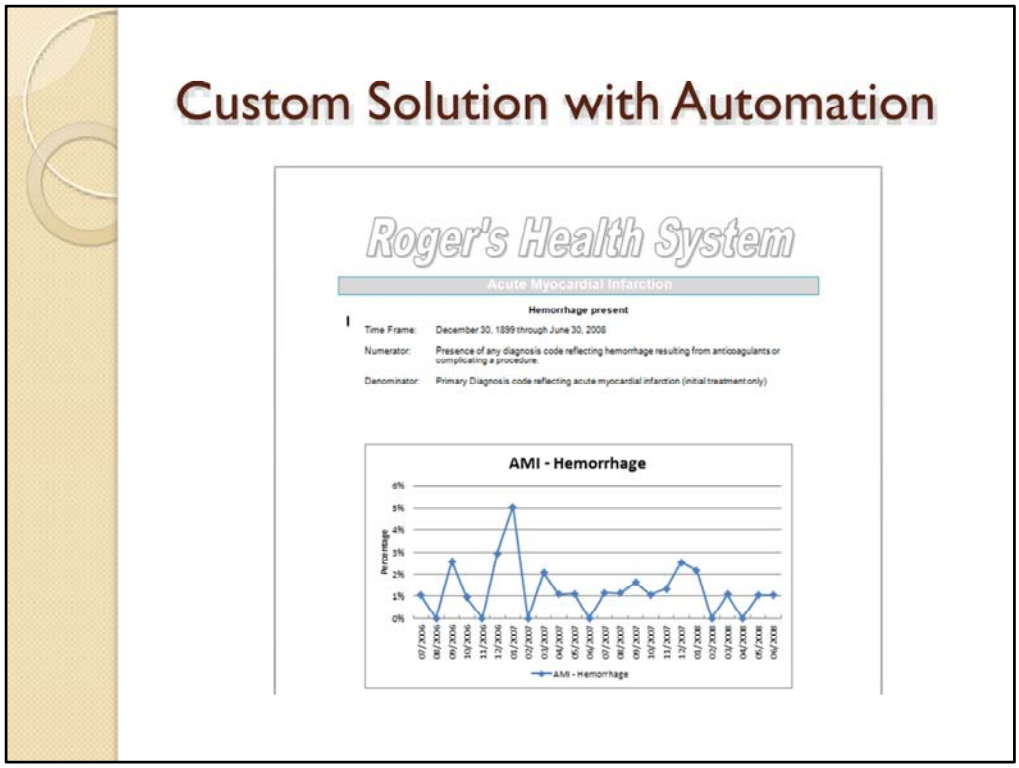
Query By Form was written by Duane Hookom as a simple query creation form. I've modified it slightly in the way that it interacts with Excel. In the original, it just exports the query to a new spreadsheet named "MyQuery.xls". I modified it to name the file after the report and used the TransferSpreadsheet method instead of OutputTo.

For years, I've been using a trick of the TransferSpreadsheet method in Access to do some simple Office Automation. When you export a query with TransferSpreadsheet , it will create a sheet with the name of the query on the tab and a named range on the sheet with the same name. If you export the query again to the same Excel file, it will over-write the existing tab with data from the new export.

So what I do, is export the query once, then create a new sheet (Sheet1) and link the cells from the new sheet back to the query sheet (MyQuery). I can format the cells in Sheet1 any way I like, and the next time I export, the data on MyQuery sheet is replaced, and the linked cells on Sheet1 are updated.

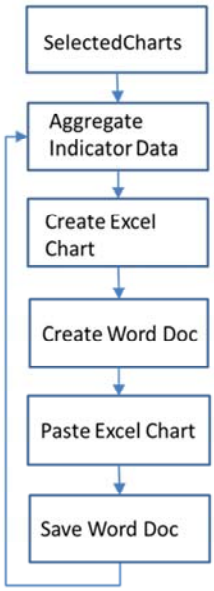
Excel 2010 Business Intelligence

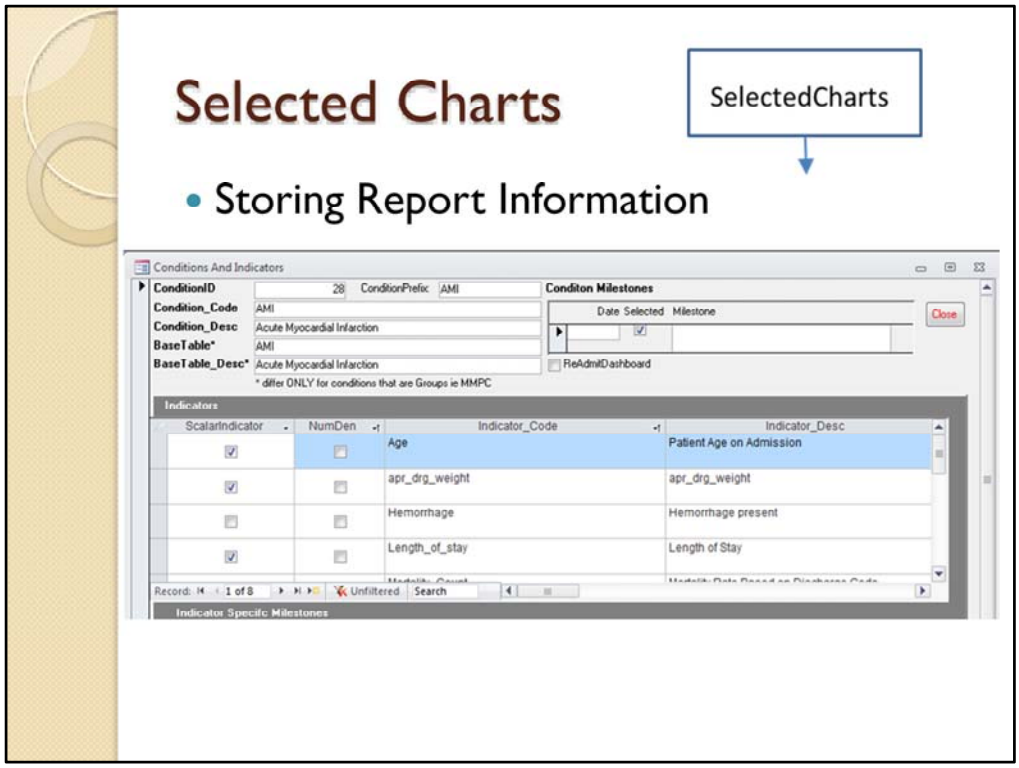
- Pulling Data
  - Pivot Tables & Pivot Charts
- Pushing Data
  - QBF & Excel BI
- DEMO



What we've seen so far is great for ad-hoc or periodic reporting by individual decision makers. But in my situation, I'm required to create a 1300 page document with text and charts every month, reporting every condition and indicator. This is too much to do individually with any method I've shown so far.
My solution was to create a custom report generator.

Overall process

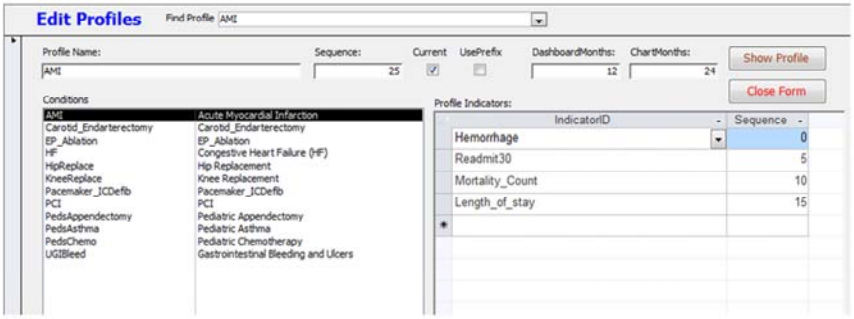




Naturally, if I'm going to automate the process, I have to store the parameters necessary to create the report. For instance, the text for the Numerator and Denominator definitions or which indicators belong to which conditions. The Conditions And Indicators form allows me to easily create new conditions and corresponding indicators or modify or delete old ones:

Selected Charts (cont)

- Storing Profile Information



Similarly, if I'm going to create many indicators at once by condition, I need to store information about which indicators belong to which conditions. I call this a Profile and you can see that AMI has for conditions: Hemorrhage, Readmit30, Mortality, and LOS. With the **Edit Profiles** form, it is easy to add or remove indicators from a profile.

Selected Chart (final)

- Combining the information into the SelectedIndicators query

Condition_Code	Condition_Desc	Indicator_Code	Indicator_Desc	Numerator	Denominator
AMI	Acute Myocardial Infarction	Hemorrhage	Hemorrhage present	Presence of any diagnosis code reflecting hemorrhage resulting from anticoagulants or complicating a	Primary Diagnosis code reflecting acute myocardial infarction (initial treatment
AMI	Acute Myocardial Infarction	Length_of_stay	Length of Stay	A numerical figure representing the Length of Stay as calculated by Spectrum Health. A patient admitted	Primary Diagnosis code reflecting acute myocardial infarction (initial treatment
AMI	Acute Myocardial Infarction	Mortality_Count	Mortality Rate Based on Discharge Code	Presence of a match in Discharge Status which relates to Death Discharge Codes (20 through 29 and 40	Primary Diagnosis code reflecting acute myocardial infarction (initial treatment
AMI	Acute Myocardial Infarction	Readmit30	Readmission within 30 Days for Any Reason	Patients readmitted within 30 days regardless of the APR-DRG.	Primary Diagnosis code reflecting acute myocardial infarction (initial treatment

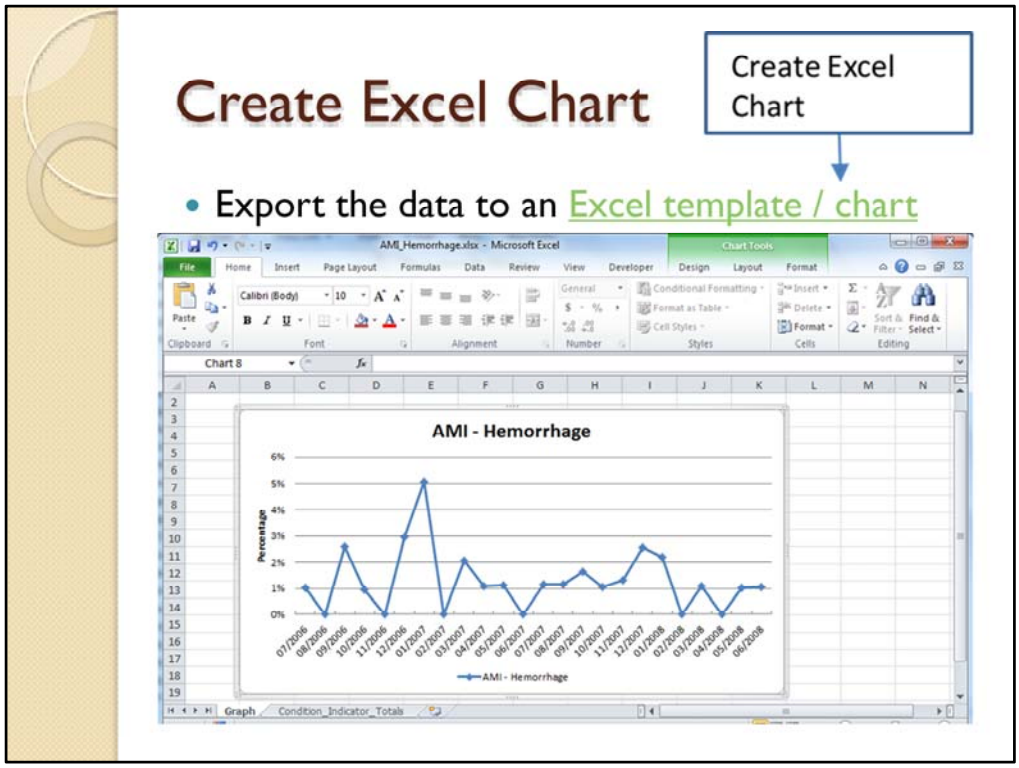
All of this information goes to build the **SelectedIndicators** query, the information of which is used to create the Source Data, Excel Chart, and Word Document.

Aggregate Indicator Data

Aggregate Indicator Data

- Starting with first record in SelectedIndicator query, programmatically create the totals query
-
```
SELECT [month] & "/20" & [year] AS
MoYear, avg(Hemorrhage) AS [AMI -
Hemorrhage], NULL AS Percentage
FROM AMI
GROUP BY Year, Month, [month] & "/20"
& [year]
ORDER BY Year, Month;
```

Start with the first indicator information and programmatically create the totals query that aggregates the data (Average by Year/Month) using the information in qrySelectedIndicators.



Export the data to an Excel template (as we did before) which will feed a chart(AMI_Hemorrhage.xlsx).
Since I'm creating the chart, I might as well save it

Create Word Doc

Create Word Doc

- Open a Word template and fill in appropriate information from qrySelectedIndicators

Roger's Health System

Acute Myocardial Infarction

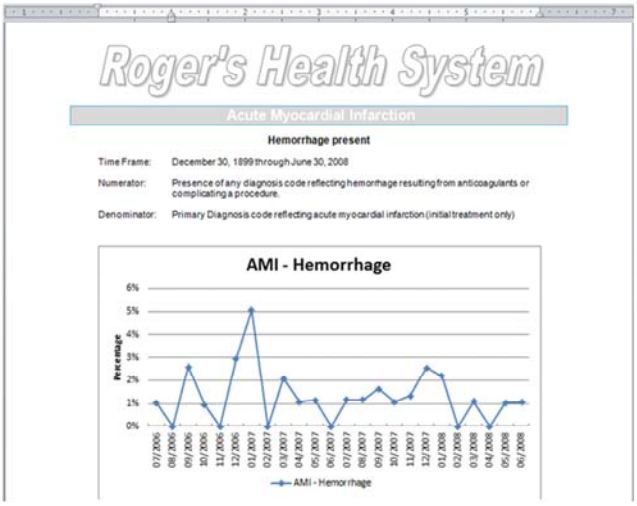
Hemorrhage present

Time Frame:	December 30, 1899 through June 30, 2008
Numerator:	Presence of any diagnosis code reflecting hemorrhage resulting from anticoagulants or complicating a procedure.
Denominator:	Primary Diagnosis code reflecting acute myocardial infarction (initial treatment only)

Paste Excel Chart

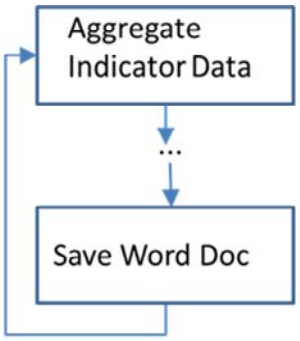
Paste Excel Chart

- Copy and paste Chart into document



Save Word Doc

- Save document
- Rinse and Repeat for all records in SelectedIndicator query



Dashboard

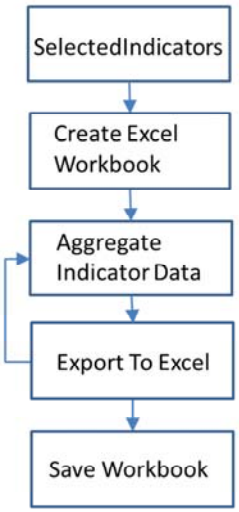
Clinical Outcome Report

AMI

July 1, 2007 to June 30, 2008

Administrative Data		Process	Outcome		
No. of Patients	Average Age	Hemorrhage	Unplanned Readmit 30 Days	Mortality	LOS
1077	63.2	1.2%	8.1%	4.5%	4.40

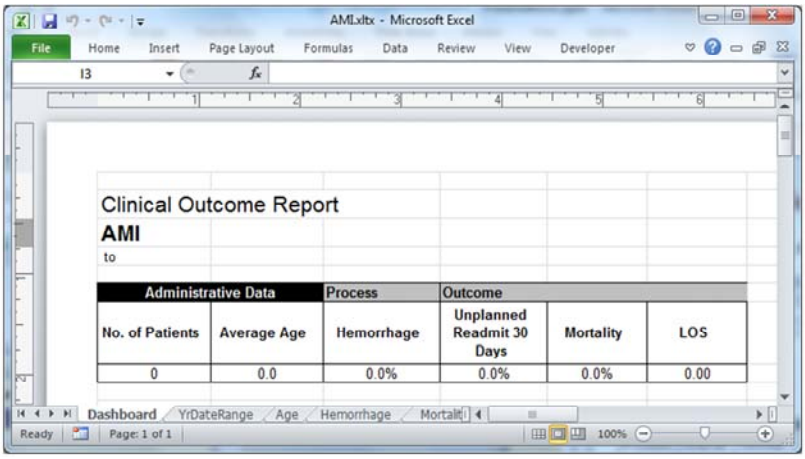
Overall process



Create Workbook

Create Excel Workbook

- Open an Excel template and save



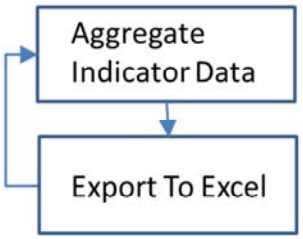
Aggregate Indicator Data

Aggregate Indicator Data

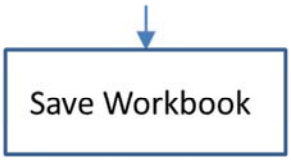
- Starting with first record in SelectedIndicator query, programmatically create the totals query
-
```
SELECT Sum(Hemorrhage) AS SumOfHemorrhage,
    Count(Hemorrhage) AS CountOfHemorrhage,
    IIf (Count(Hemorrhage)=0,Null,
          Sum(Hemorrhage)/Count(Hemorrhage))
          AS AvgOfHemorrhage
FROM AMI
WHERE[Discharge_Date]>=#7/1/2007#;
```

Start with the first indicator information and programmatically create the totals query that aggregates the data (Across 1 year) using the information in qrySelectedIndicators.

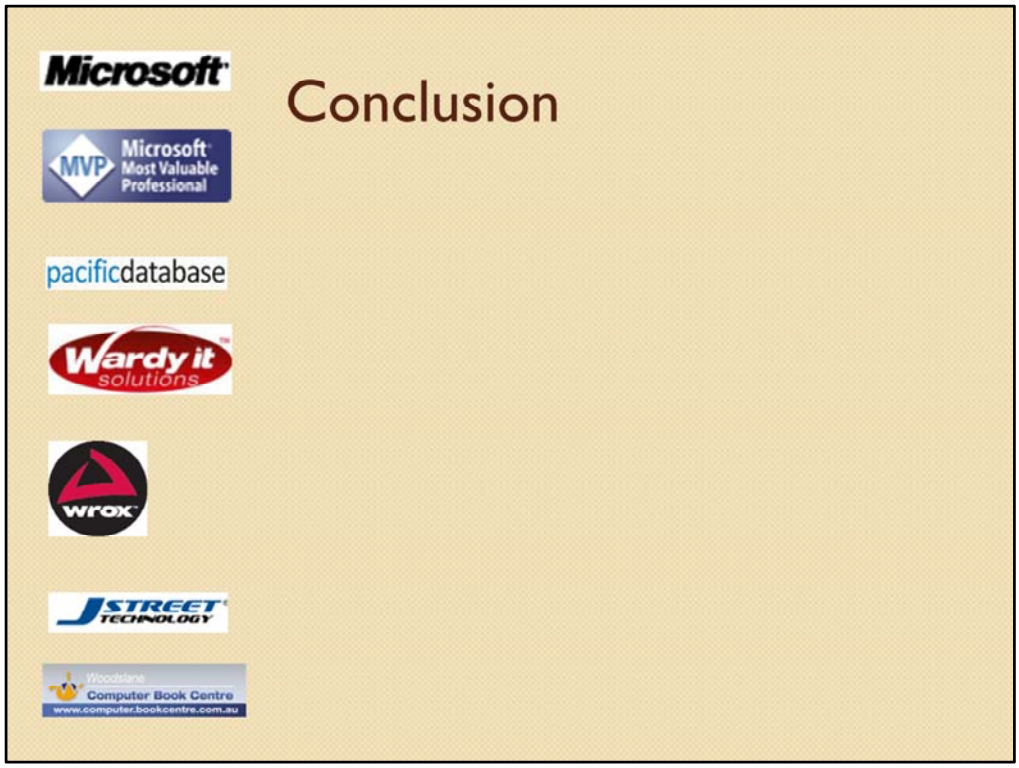
Export To Excel



- Use TransferSpreadsheet to export the query to the saved Workbook
- Rinse and Repeat for all indicators



- Save the Workbook



Now the purpose of this is obviously not to demonstrate how to create a reporting system for your local healthcare provider. This isn't a healthcare conference. However, I hope I've demonstrated how it is possible and even practical to develop a reporting system for your transactional system --based on data warehouse principles -- that will give you improved performance and flexibility for you and your users.