

Data Warehousing Overview

This Presentation will leave you with a good understanding of Data Warehousing technologies, from basic relational through ROLAP to MOLAP and Hybrid Analysis.

However it is necessary to understand how transactional data is stored to fully appreciate the requirement for Data Warehousing, and that's how this presentation begins.

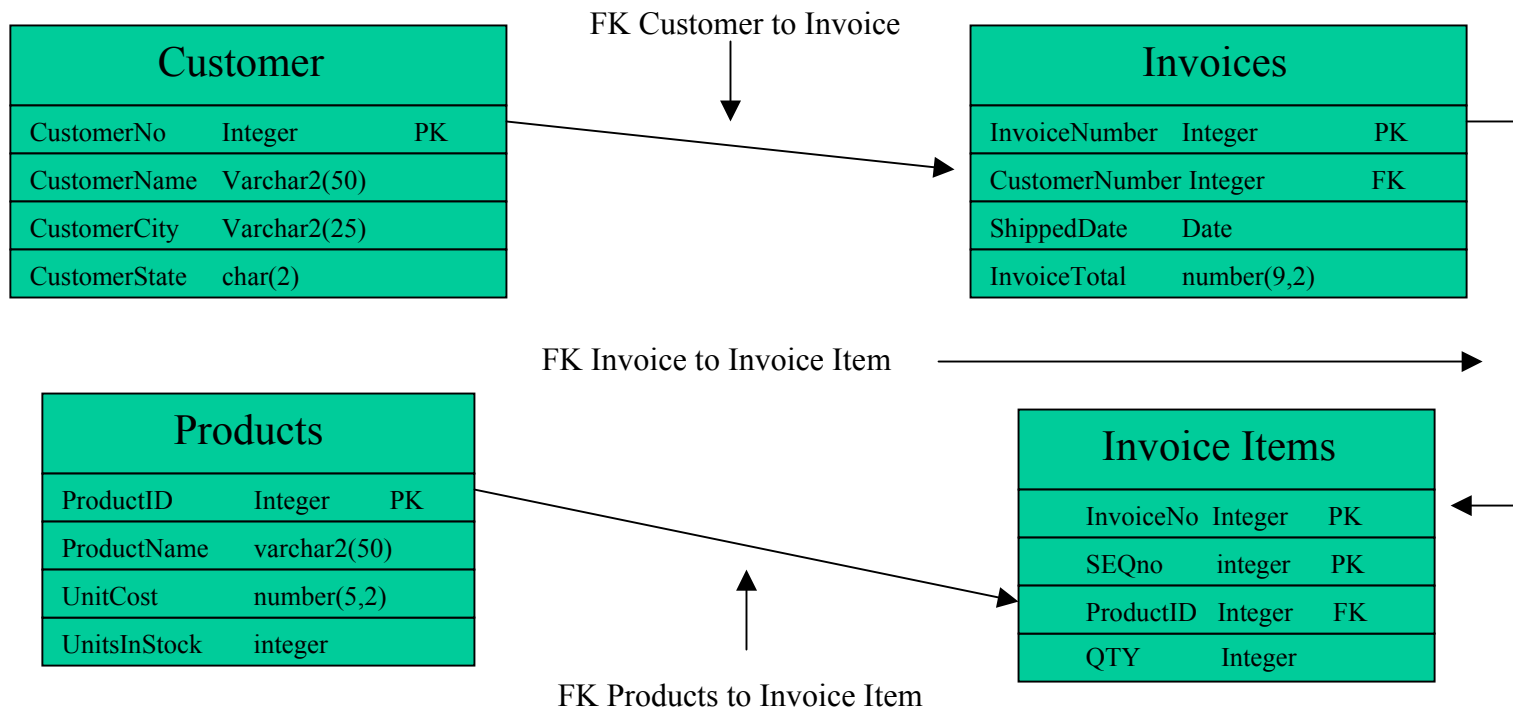
The Following topics will be covered in this presentation.

1. Transactional Databases (OLTP)
2. Data Warehousing
3. Relational On-Line Analytical Processing (ROLAP)
4. Multi-Dimensional On-Line Analytical Processing (MOLAP)
5. Hybrid Analysis
6. EssBase specifics

Transactional Databases (1)

What is Online Transaction Processing (OLTP)

A database designed for OLTP is highly normalized. It is geared towards avoidance of duplication of data and protecting of the integrity of that data. For example, an OLTP database will probably not let you record an invoice for a customer that is not in your customer table.



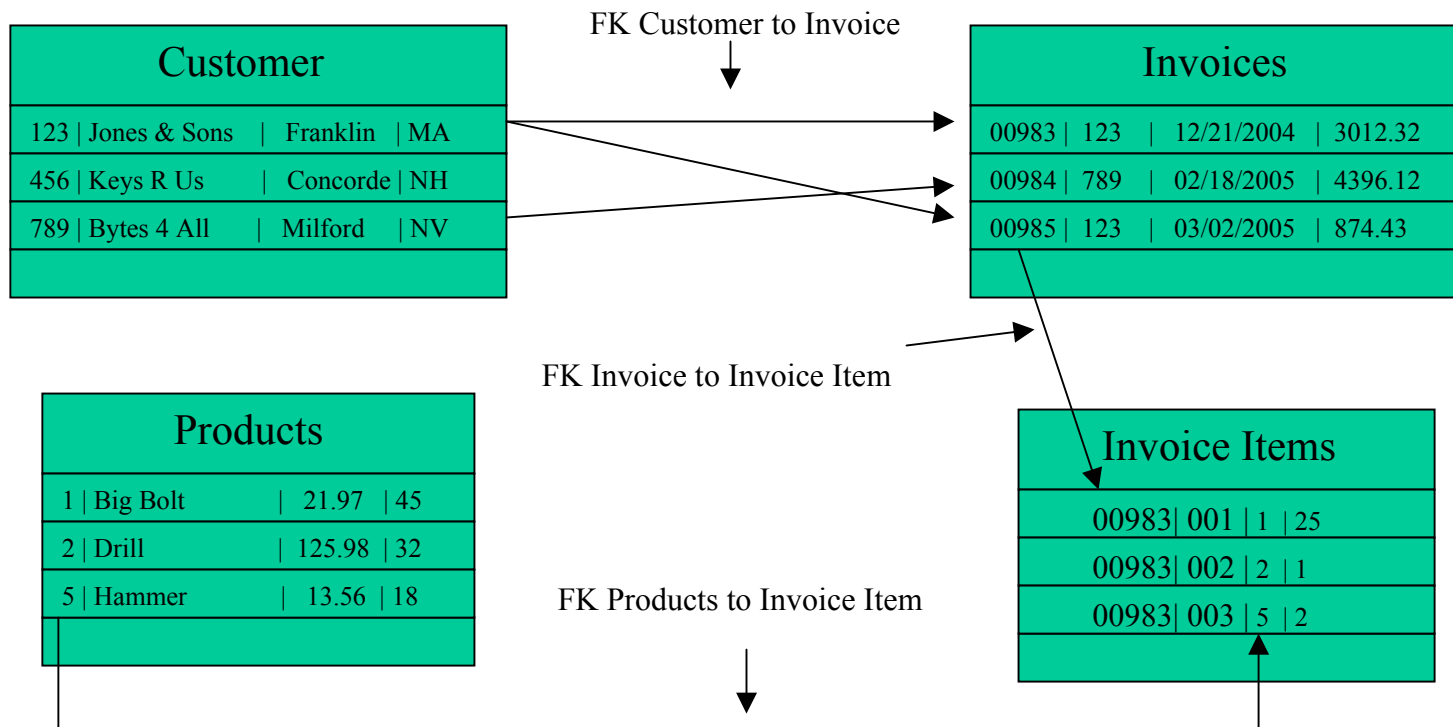
The tables are designed to record a fact, such as a customer name, only once and to relate that fact to other tables that require it, such as the invoice table. This is usually done through a constraint known as a foreign key.

Transactional Databases (2)

OLTP Database with Data

Here is how the Invoice / Customer / Product table structure looks when data is inserted.

Every invoice must have a valid customer, Every Invoice line item must be for a valid Invoice and every product on an invoice line item must be in the product table.



This is highly normalized and strongly protects the integrity of the transactional data that the business relies on to invoice customers.

Transactional Databases (3)

What is the Problem with OLTP

The structure described in the previous two slides is perfect for recording business transactions, invoicing, etc. So what then is the issue?

Well, what if you want to extract knowledge from that data so that key business decisions can be made and you had to address the following business intelligence scenario:

Show me Product Name, Quantity ordered, Customer Name and Shipped date for all invoices with a total value of greater than \$3,800 for customers located in Massachusetts or Rhode Island

Does this knowledge exist in our transactional database? It does, but it is fragmented and requires a complex SQL Statement to extract it. This will take time to run and is unlikely to be able to be created by the person requesting the knowledge. The SQL statement looks something like the following

```
SELECT p.ProductName, ii.QTY, c.CustomerName, i.ShippedDate
FROM Products p INNER JOIN ((Customers c INNER JOIN Invoices i ON C.CustomerNo = i.CustomerNumber)
INNER JOIN invoiceItems ii ON i.InvoiceNumber = ii.InvoiceNo) ON p.ProductID = ii.ProductID
WHERE (i.invoiceTotal >3800) AND (c.CustomerState In ("MA","RI"));
```

You should get the right answer BUT imagine this query running over millions of rows joining four tables. The answer will not come quickly and the query will take resource away from the database server, probably at a time when we need the maximum efficiency possible to record transactions.

This lack of efficiency and degradation of performance on the transactional database will not encourage analysts to request such knowledge from the database

Analytical Databases

How OLAP can help us with this problem

The term OLAP stands for Online Analytical Processing. OLAP can be enacted in a number of ways

Pure Data Warehouse Solution

Achieves its goal mostly by de-normalizing the transactional data

Star or Snowflake Schema in a relational Database (ROLAP)

Organizes the transactional data into a fact table of values to be measured and surrounds this with information about the dimensions that makes each of those facts unique. The data in such a schema is highly grouped at the lowest level of detail.

Data Cubes (MOLAP)

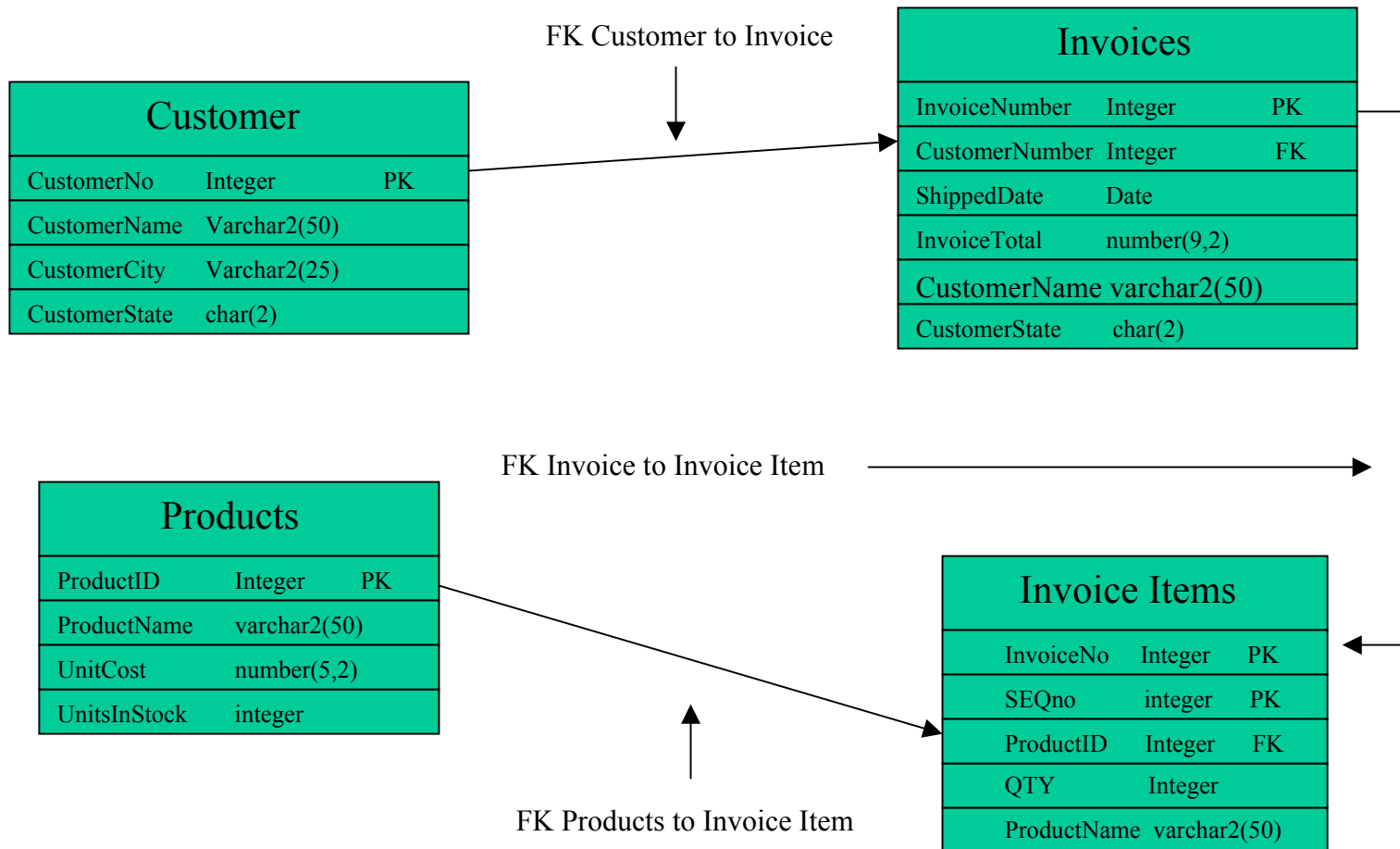
This is the ultimate. Usually based off a star or snowflake schema, a cube contains pre-compiled answers to ad-hoc queries.

We will now look at each of these in more detail.

Data Warehousing (1)

How would a Data Warehouse be structured

In a typical data warehouse usage, the previous customer / invoice situation may be represented thus:



Data Warehousing (2)

So. What is Different ?

Three Crucial changes have been made here in comparison with the OLTP version of these tables.

While we have retained the customer and product tables, we have duplicated some of their data in the invoice and InvoiceItem tables. Note that customer name and customer state now appear in the Invoice table and productName also appears in the InvoiceItem table. This de-normalization would never be allowed in an OLTP database.

Now the SQL Statement to answer the same Business Intelligence question:

Show me Product Name, Quantity ordered, Customer Name and Shipped date for all invoices with a total value of greater than \$3,800 for customers located in Massachusetts or Rhode Island

Can be written much simpler as:

```
SELECT ii.ProductName, ii.Quantity, i.customerName, i.ShippedDate
FROM Invoices i INNER JOIN invoiceItems ii ON i.InvoiceNumber = ii.invoiceNo
WHERE (i.invoiceTotal >3800) AND (i.CustomerState In ("MA","RI"));
```

Compare that with the OLTP version

```
SELECT p.ProductName, ii.QTY, c.CustomerName, i.ShippedDate
FROM Products p INNER JOIN ((Customers c INNER JOIN Invoices i ON C.CustomerNo = i.CustomerNumber)
INNER JOIN invoiceItems ii ON i.InvoiceNumber = ii.InvoiceNo) ON p.ProductID = ii.ProductID
WHERE (i.invoiceTotal >3800) AND (c.CustomerState In ("MA","RI"));
```

Data Warehousing (3)

Much Simpler

Clearly this is much simpler than trying to extract the same knowledge from OLTP

However, there is a price. How do we get the data duplicated in this fashion?

Most data warehouse databases reside on dedicated servers and typically are refreshed with that day's transactions, every night. Usually there will be a process on the data warehouse that will truncate a series of temporary tables and then repopulate them with that day's transactions by downloading them from the OLTP database, or combination of OLTP databases. The process would then continue to take the data in the temporary tables and update the Data Warehouse's tables, again via stored procedures. This last stage will be written to duplicate data into tables as felt necessary by the Data Warehouse designer.

The above nightly feed is no mean feat and is usually a complex set of stored procedures and other technologies, such as Informatica for mapping and Control-M for scheduling, as examples. There are many places this process can introduce bugs and itself needs maintenance. If the OLTP database changes, then it is likely the data warehouse and related procedures will have to change also.

While the data warehouse is designed to support ad-hoc queries to the best of its abilities, there is still an SQL query that has to be processed to return the required answer. The goal of the Data Warehouse is to have a design that keeps these queries to their simplest forms possible with as few joins and aggregations as possible.

However, at run time, the query will be taking resource and possibly processing millions of rows to obtain an aggregated answer. Definitely it will be a huge improvement over OLTP, but we are still at the mercy of how well the queries are written and how well designed the database was to support these ad-hoc queries.

One advantage of a data warehouse over a MOLAP cube is that the DW can support and report on virtually any type of data that is in the DW and that you can write an SQL query for. OLAP cubes usually deal only with aggregated numerical data known as measures, ie Bookings, Billings, Costs, etc.

ROLAP and MOLAP

What are these technologies used for

ROLAP (Relational OLAP) and MOLAP (Multi-Dimensional OLAP) are good at answering questions relating to measures. A question such as this

What was the value of my total billings of product X to customers in Canada in quarter 3 of 2004?

The measure in this case is BILLINGS and Products, Customers, Geography and Time would be relate to the billings number as Dimensions.

ROLAP is usually enacted as a Star and/or Snowflake schema with a FACT table containing the summarized numbers that you want to measure. The summarization is controlled by foreign keys in the FACT table pointing to other tables surrounding the fact in a star like fashion. These tables are known as Dimensions

MOLAP is usually based off a star schema but goes one step further in pre-compiling all the numbers associated with a hierarchy. This means that any drill down and drill out functionality will usually be dealing with values that are already calculated and is therefore very fast.

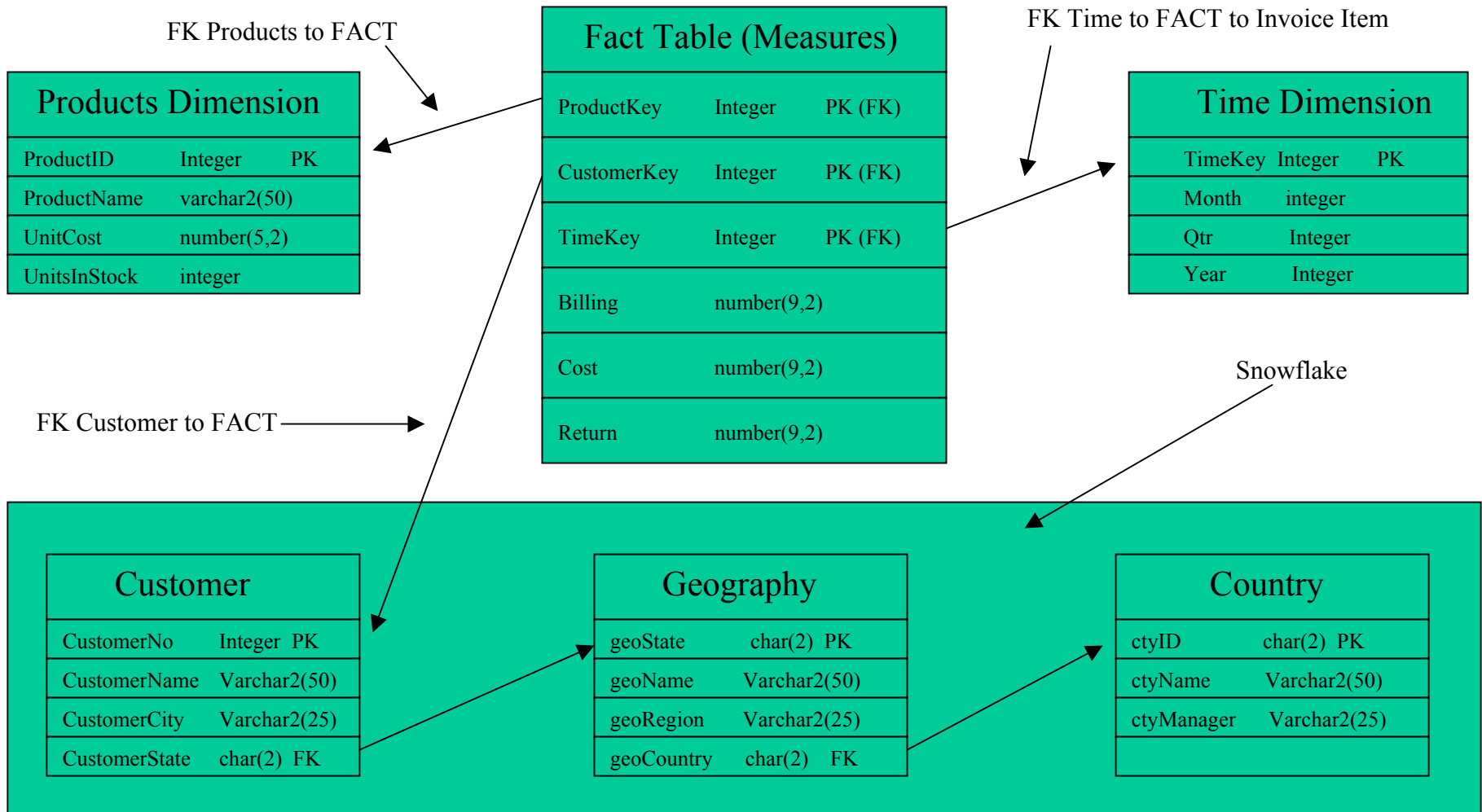
In some MOLAP solutions there is a combination of data that is accessed frequently and will be in a cube format and pre-compiled and data that is less frequently accessed and may be stored relationally. The term that describes this structure is **Hybrid Analysis**

We will now look at these in more detail

R(elational)OLAP (1)

Star/Snowflake Schema

Such a schema is based on a table of facts with foreign keys pointing to dimensions. The fact table is highly grouped, holding one number for each unique combination of dimension keys



R(elational)OLAP (2)

What is the benefit of the Star/Snowflake schema

The fact table consists entirely of foreign keys pointing to dimension tables and values for the relevant measures. Each row in the fact table is unique for the combination of foreign keys. This effectively means that the statistics in the fact table are highly grouped. One row in the fact table may be a summarization of hundreds of rows from the source data.

This grouping of results into a fact table typifies this kind of schema.

Each row of data in a dimension table will typically exist to represent the lowest level of a hierarchy, such as a product number. However, there will usually be parent levels indicated in that same record. This means that the fact table contains precompiled measures at the leaf level, but work has to be done to obtain a measure for a parent level. This would involve further grouping by parent level in the SQL statement.

This means that answers to leaf level queries will be quickly achieved, but parent level queries will be slower and will require knowledge in how to construct the relevant query

The snowflake schema indicates that there are parent levels associated with the dimension, in this case Customer, but that these are obtained from separate tables, rather than all the hierarchy information in one dimension table as in the star schema.

Star/Snowflake schemas usually get their data from a data warehouse, rather than a transactional database. In fact such schemas are often looked upon as being views of the data warehouse data.

R(relational)OLAP (3)

Tools associated with ROLAP

There are tools associated with ROLAP technology that will help you analyze the data in the Star Schema.

One example of this is SQL/Server's Analysis Services.

This tool lets you virtualized the star Schema relational database as if it were a cube with all the parent levels also pre-compiled.

However, the data is not physically stored as a cube. The technology uses a series of algorithms to quickly and efficiently calculate parent levels at request time.

However, there is no getting away from the fact that there is work going on at run time to calculate the parent level values.

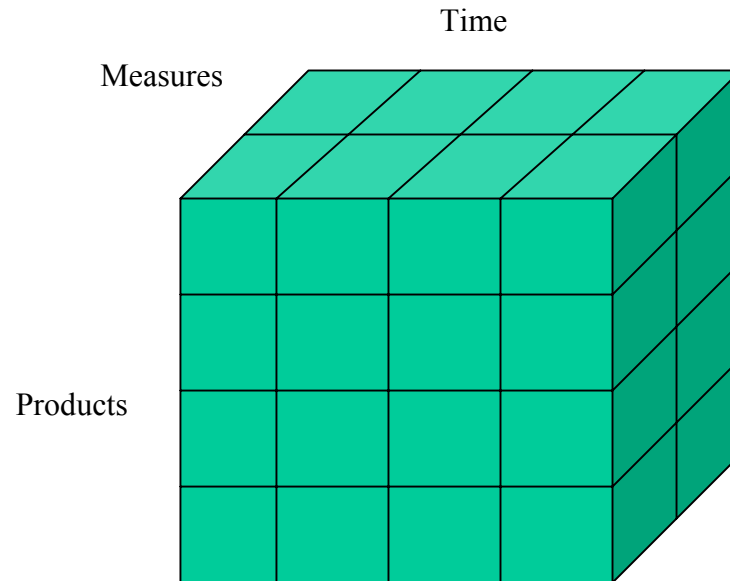
One huge benefit that this tool gives the users is that they do not have to concern themselves with HOW to summarize the data. That is the purpose of the cube algorithms. Client tools are provided to help the user retrieve, pivot and analyze the data with very little effort on their part.

Often, tools that are familiar to finance organizations, such as Excel, can be used to retrieve the Virtual Cube data.

M(ulti-dimensional)OLAP (1)

What is MOLAP

Bordering on the ultimate, is MOLAP technology. Here you build a physical cube, usually from a Star Relational Schema. A data cube physically stores the aggregations, not just for **leaf** but also **parent** levels.



M(ulti-dimensional)OLAP (2)

MOLAP Pros

Aggregations are stored at virtually all levels. Therefore no additional work is required to answer a business intelligence question

Very quick access

No complex knowledge required on the part of the user

Easy to use tools to access and report from the data.

M(ulti-dimensional)OLAP (3)

MOLAP Cons

Very expensive solution

High storage requirement

Time to build cube

Requires technical expertise to build and maintain the cube

Small changes can have dramatic effect

Complex tools required to build and maintain the cube

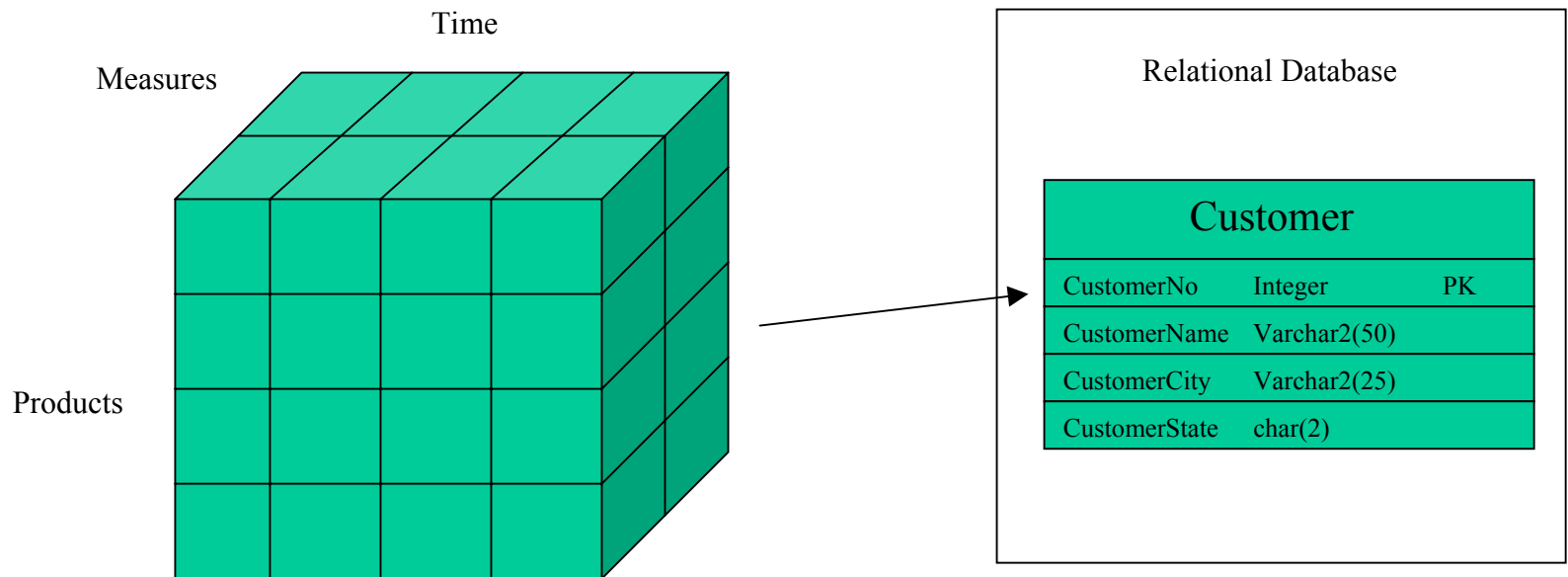
Reliance on source data

Hybrid Analysis

What is Hybrid Analysis

Due to the high storage requirements of a MOLAP cube, it may not make sense to represent every dimension in the cube. For example, if you had a data requirement to analyze Bookings and Billing for Products and Product Families over time, but only occasional need to report on the associated customer data, it may make sense to leave the customer dimension out of the cube and link to it via Hybrid Analysis when needed.

The customer data will take longer to extract than if it were in the cube, but this inefficiency is balanced by the much smaller and more efficient cube with one less dimension.



MOLAP Solutions

EssBase

EssBase is currently the leading MOLAP engine for Enterprise MOLAP Analysis

There are a number of features supported by EssBase to provide this high-end functionality

Dense / Sparse dimension

Outlines

Load Rules

Calculations

Reports

Integration Server

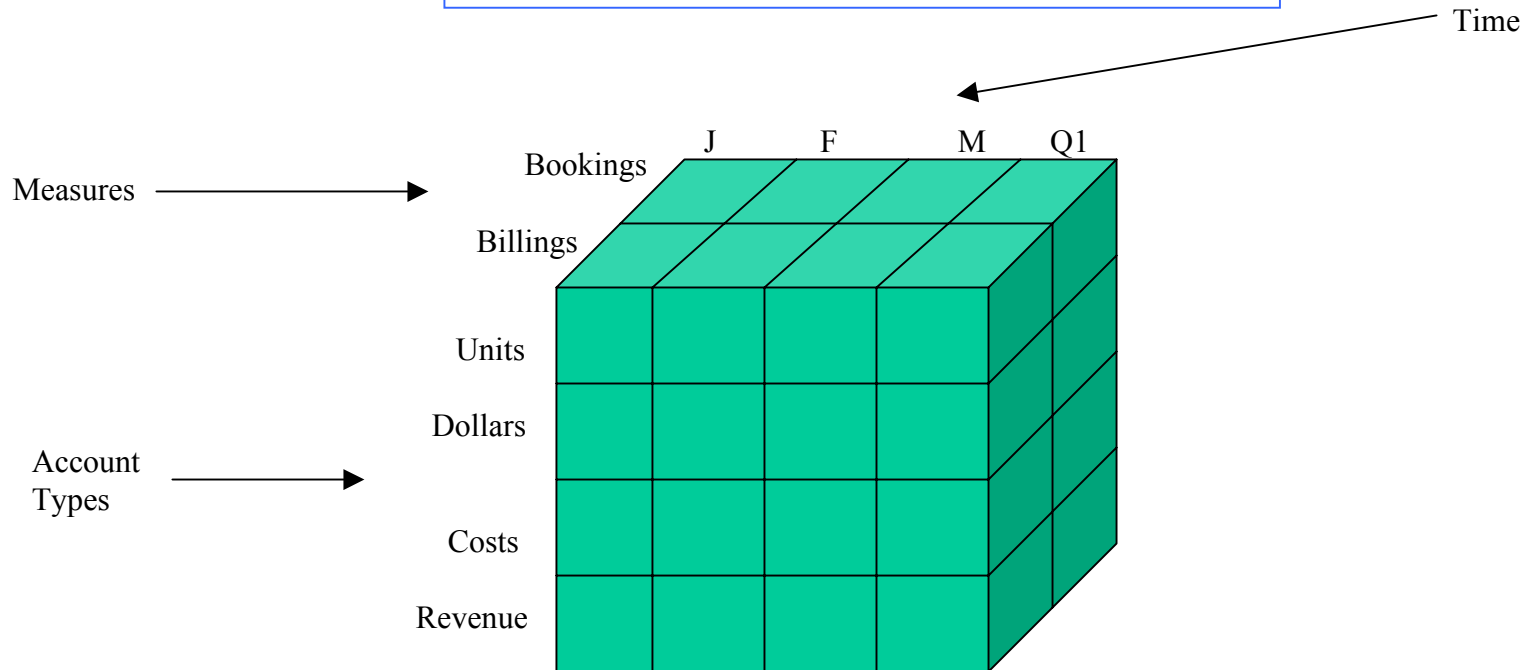
Drill-thru

Hybrid Analysis

The remaining slides in this presentation talk about specific essBase functionality

Essbase (1)

All Data blocks are of exactly the same construction
They are made up of all combinations of dense dimensions

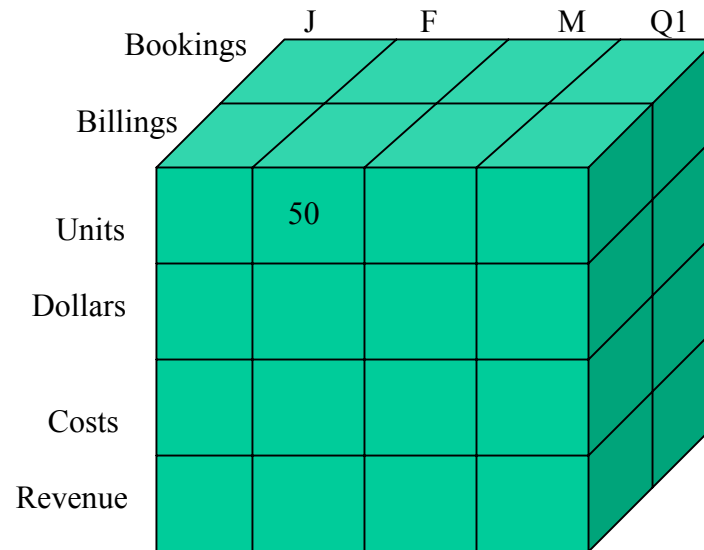


In this example cube, Measures, Time and Account Types are our dense dimensions

Essbase (2)

Blocks are created for each unique combination of sparse dimensions

These unique combinations are the index to the block

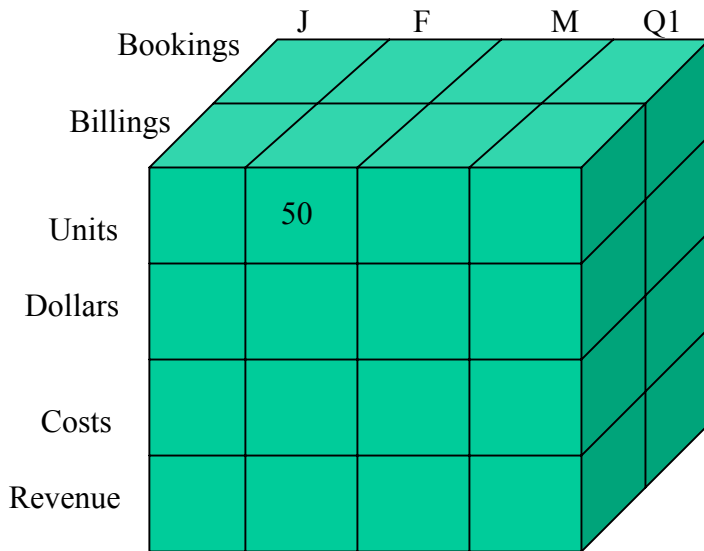


If we wanted to record the below data in essBase then a block would be created indexed by “IBM,RedHats” (the combination of sparse dimensions that we want to record data for)

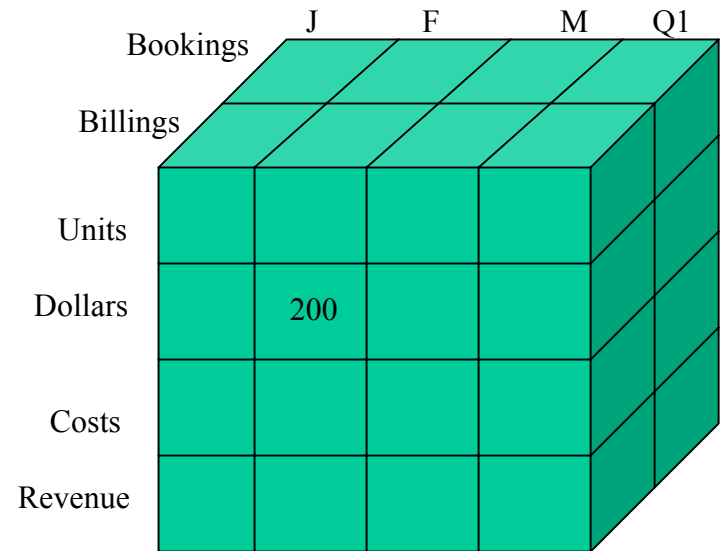
<u>Customer (sparse)</u>	<u>Product (sparse)</u>	<u>Feb Units Billings</u>	<u>Feb Dollars Billings</u>
IBM	Red Hats	50	

Essbase (3)

Now we have another piece of data to record. This is for Cisco,Cables. This is a new Combination of sparse dimensions and therefore a new block will be created



IBM,Red Hats



Cisco,Cables

Customer (sparse)

Product (sparse)

IBM
Cisco

Red Hats
Cables

Feb
Units
Billings

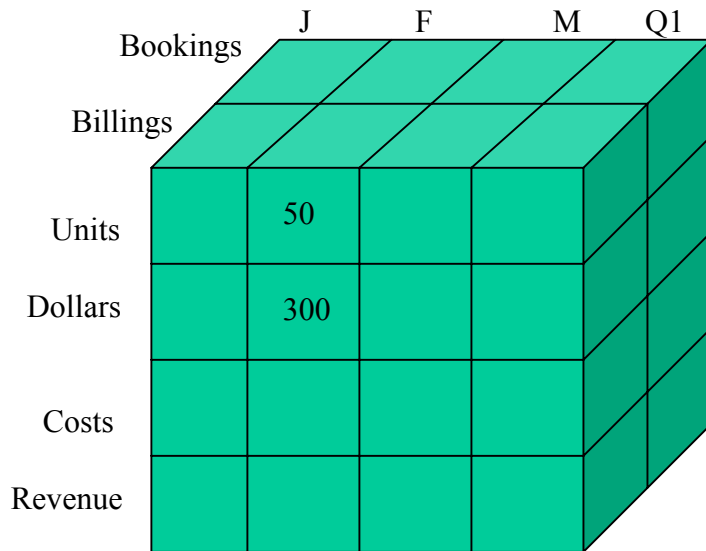
Feb
Dollars
Billings

50

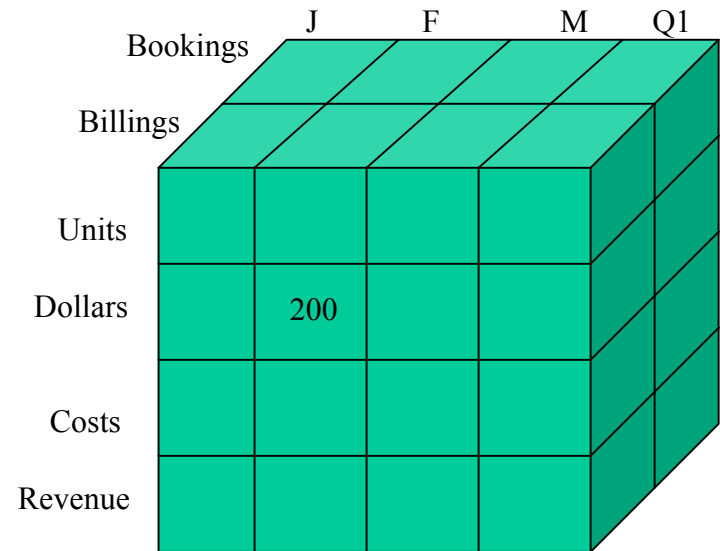
200

Essbase (4)

Yet another piece of data to record. These are dollars for IBM,Red Hats. This is an existing combination of sparse dimensions and therefore an existing block will be retrieved and updated.



IBM,Red Hats



Cisco,Cables

Customer (sparse)

IBM
Cisco
IBM

Product (sparse)

Red Hats
Cables
Red Hats

Feb
Units
Billings

50

Feb
Dollars
Billings

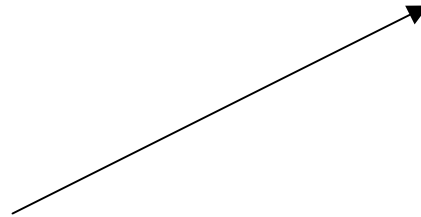
200
300

Essbase (5)

Some statistics for you

The BRM essBase cube Has

Dimensions	-	9
Block Size	-	18,480 bytes
Block Density	-	46.450%
Number of Blocks	-	14,859,468
Cube size	-	8 gig (approx)
Potential Blocks	-	281,140,587,000



This is a very interesting number and tells you a lot about OLAP.

This is saying that, if there were data for all possible combinations, then this is how many blocks there would be. Look at the huge difference between this number and the actual number of blocks. Clearly we only pre-compile a very small percentage of the possible sparse combinations in the outline, ie the combinations that the source system provides data for.