DATA WAREHOUSE

SCALABILITY and TUNING

Data Warehouse Summit
Phoenix
Thursday, December 10, 1998
8:30 A.M. - 9:50 A.M.

David McGoveran
Alternative Technologies
13150 Highway 9, Suite 123
Boulder Creek, CA 95006
Telephone: 408/338-4621
www.AlternativeTech.com

BEFORE YOU LEAVE…

PLEASE FILL OUT YOUR EVALUATIONS.

Thank you!

OUTLINE

- **Scalability**
  - DEFINITION
- **Key Ways Data Warehouse Scalability Fails**
  - DEADLY ARCHITECTURAL DECISIONS
- **Difficulties in Tuning**
  - WHY ITS DIFFERENT FROM TUNING FOR OLTP
- **Tuning Options**
  - DATA PLACEMENT
  - TABLE PARTITIONING
  - QUERY AND LOAD PARALLELISM
- **The Secret(s) of Success**

SCALABILITY
Definition

SCALABILITY IS:

- *SCALEUP* or *SPEEDUP* (see slides which follow)
- WITH RESPECT TO A *SPECIFIC RESOURCE MIX*
  - » *AMOUNT OF MEMORY, NUMBER / SIZE OF STORAGE UNITS, NUMBER OF CPUs, NUMBER OF NODES, et cetera.*
- OVER A *SPECIFIED RANGE*
- FOR A *PARTICULAR WORKLOAD*
  - » *NUMBER OF USERS, DB SIZE, TRANSACTION RATE, TRANSACTION COMPLEXITY or PROFILE*

- **Conceptual Definition of Speed Up**

MORE RESOURCES ☑ BETTER PERFORMANCE, SAME WORKLOAD

- **Conceptual Definition of Scale Up**

MORE RESOURCES ☑ SAME PERFORMANCE, BIGGER WORKLOAD

SCALABILITY
GENERAL GOALS

The Essence of Scalability is Independence

of. . .

- **COMPONENTS BY FUNCTION AND TASK INSTANCE**
- **RESOURCES ASSIGNED TO INDEPENDENT COMPONENTS**

- **Non-Independence Manifests As. . .**
  - **RESOURCE CONTENTION (WAIT TIME)**
  - **PROCESSING ANOMALIES AND MAINTENANCE SIDE EFFECTS**
  - **INABILITY TO EXPLAIN THE ARCHITECTURE**
  - **INABILITY TO EXPLAIN THE CAUSE OF SYMPTONS**

Avoid These By Building-in Independence

HOW DW SCALABILITY FAILS

- **Physical Schema Rigidity**
  - **THE HIGH COST OF CHANGES**
- **Load Interferes with Query**
  - **QUERY ACCESS LIMITED DURING LOAD, REFRESH, INDEX BUILD**
- **Administrative Complexity**
  - **BACKUP, RECOVERY AREN'T REALLY ONLINE**
  - **REDISTRIBUTING DATA ON NEW DRIVES**

HOW DW SCALABILITY FAILS

- **Loss of Resource Control**
  - USERS MODIFY SCHEMA
  - USERS ISSUE ARBITRARY QUERIES
  - NO CONTROL OVER GENERATED SQL
  - NO KNOWLEDGE OF LOAD
  - NO MEANS TO MONITOR AND CONTROL LOAD
- **Poor Table Design**
  - COMPLEX PRIMARY KEYS
    - » IN AN ATTEMPT TO AVOID TOO MANY TABLES
  - NO PRIMARY KEYS, CHARACTER STORAGE, REDUNDANT DATA
  - RESULT: WASTED STORAGE AND EXECESSIVE I/O

HOW DW SCALABILITY FAILS

- **Denormalization Without Discipline (Potentially Bad)**
  - JOINED TABLES
  - PARTITIONED AND REPLICATED TABLES
  - REDUNDANT COLUMNS
  - DERIVED COLUMNS
  - EMBEDDED FOREIGN KEYS
  - UNIONED ENTITIES (LEADS TO NULLS!)
  - various other reasons....
- **Why is this done?**
  - ASSUMED TO OPTIMIZE STORAGE ALLOCATION
  - ASSUMED TO MINIMIZE I/O COSTS, INCLUDING JOIN I/O
  - MAKING IT “EASIER” TO ACCESS RELATED INFORMATION

HOW DW SCALABILITY FAILS

- **With VLDB, Physical Design Rules Change**

  ***EXAMPLE*:**

  - » **COMPOUND KEYS IN VERY LARGE TABLES ARE OFTEN REDUNDANT, WASTING LOTS OF SPACE**

  ***SOLUTION*:**

  - » **REPLACE WITH SURROGATE KEYS AND A LOOKUP TABLE**

  ***EXAMPLE*:**

  - » **"FACT" TABLES OFTEN CONTAIN MULTIPLE ENTITIES WITH NULLABLE ATTRIBUTES**
  - » **CAUSES CONDITIONAL PROCESSING**

  ***SOLUTION*:**

  - » **NORMALIZE AND ELIMINATE NULLS**

DEADLY ARCHITECTURAL DECISIONS

- **Mixing Workloads**
  - SYNCHRONIZING OPERATIONAL SOURCES
  - TRANSFORMATION AND CLEANSING
  - EXTRACT PROCESSING
    - » *MOLAP TOOLS*
    - » *BATCH REPORTING*
  - AD-HOC QUERY
- **Confused Design**
  - BY MIMICRY (OFTEN “FLAKEY”)
  - BY QUERY OR BI TOOL, OR BY USER
    - » *THE “TOO MANY DATA MARTS” TRAP*

DEADLY ARCHITECTURAL DECISIONS

- **Selecting the Wrong DBMS**
  - **LIMITATIONS**
    - » ***QUERY COMPLEXITY***
    - » ***TABLE SIZE***
    - » ***INDEX CHOICE AND SIZE***
- **Selecting the Wrong Hardware**
  - **LIMITATIONS:**
    - » ***NUMBER OF FILES***
    - » ***FILE SIZE***
    - » ***NUMBER OF CONTROLLERS***
    - » ***AMOUNT OF MEMORY***
    - » ***NUMBER OF CPUs***

ARE YOUR QUERIES OUT OF TUNE?
(-: *again?* :-)

- **Operational Query Tuning**
  - CAPACITY AND LOAD ANALYSIS
  - TUNE AND DEPLOY: DESIGN SEPARATE FROM OPERATIONS
  - RELATIVELY EASY TO LOCALIZE TUNING EFFECT
  - WELL DEFINED PROCESSING PRIORITIES
  - KEY PROBLEM IS CONCURRENCY AND CONTENTION
- **DW Query Tuning Is An On-Going Process**
  - STABLE LOAD PROFILES ARE RARE
  - RAPID GROWTH MAKES I/O DIFFICULT TO MODEL
  - HIGHLY INTEGRATED AND MULTIPLE PRIORITIES
  - KEY PROBLEM IS CHANGE

QUERY PRINCIPLES

- **Make Each Query Smart!**
- **Minimize Amount of Data**
  - **STORED AND ACCESSED**
  - **RETURNED OR UPDATED**
- **Divide and Conquer As Necessary**
  - **ASK FOR WHAT YOU NEED IN ONE QUERY**
    - » ***PROVIDE ALL KNOWN COLUMN RELATIONSHIPS***
  - **FLATTEN SUBQUERIES**
  - **AVOID AGGREGATE FUNCTIONS WHEN REASONABLE**
  - **BREAK INTO ADDITIONAL QUERIES *ONLY AS NECESSARY***
  - **USE TEMPORARY DATA WORK TABLES ONLY IF NECESSARY**

KEY TUNING OPTIONS

- **Indexes**
  - **AVOID TABLE SCANS!**
    - » ***EXCEPT FOR "SMALL" TABLES***
  - **INDEX TYPE**
    - » ***B-TREE, HASH, BIT-MAPPED, HYBRID, EXPRESSION, MULTI-TABLE, SPECIALTY (e.g., R-TREE) TABLE AND COLUMN SELECTION***

  REQUIREMENTS:
    - » ***LOAD PROFILES, PRIORITIES, INDEX OPTIONS, DATA INDEPENDENCE***

  METHOD:
    - » ***OPTIMIZATION VIA SEARCH ARGUMENTS***

KEY TUNING OPTIONS

- **Data and Index Placement**
  - **NODE, CONTROLLER, DISK DRIVE**
  - **RELATIVE PLACEMENT**
    - » ***AVOID CONTENTION***
    - » ***MAXIMIZE PARALLELISM***

REQUIREMENTS:

- » ***I/O DISTRIBUTION, CONTENTION, LOAD PROFILES, RESOURCES, DATA INDEPENDENCE***

METHOD:

- » ***CALCULATION BY REFINEMENT, CONFLICT ANALYSIS***

KEY TUNING OPTIONS

- **Table Partitioning**
  - PARTITION TYPE: KEY RANGE, EXPRESSION, HASH, ROUND ROBIN, SCHEMA
  - PARTITION SIZE
  - REQUIREMENTS: LOAD PROFILES, RESOURCES, DATA INDEPENDENCE
  - METHOD: CALCULATION BY REFINEMENT
- **Replication**
  - REPLICATION MECHANISM AND TIMING
  - TABLE (AND PARTITION) SELECTION
  - REQUIREMENTS: REFRESH COST, LOCALIZED LOADS
  - METHOD: SIMULTANEOUS GOAL OPTIMIZATION

KEY TUNING OPTIONS

- **Parallelism**
  - LOAD AND EXTRACT
    - » AVOID CONTENTION
  - QUERY
    - » THE RIGHT DEGREE OF PARALLELISM IS ESSENTIAL
    - » DIFFICULT TO CONTROL IN SOME PRODUCTS
  - INDEX AND TABLE BUILD
    - » AVOID ALLOCATION ERRORS
  - BACKUP AND RECOVERY
    - » PARTIAL DATABASE RECOVERY MAY SUFFER

THE DW TUNING DILEMMA

All Tuning Techniques Depend On . . .

KNOWLEDGE

and

INDEPENDENCE

The Two Things You Have The Least Of With Most Data Warehouses!

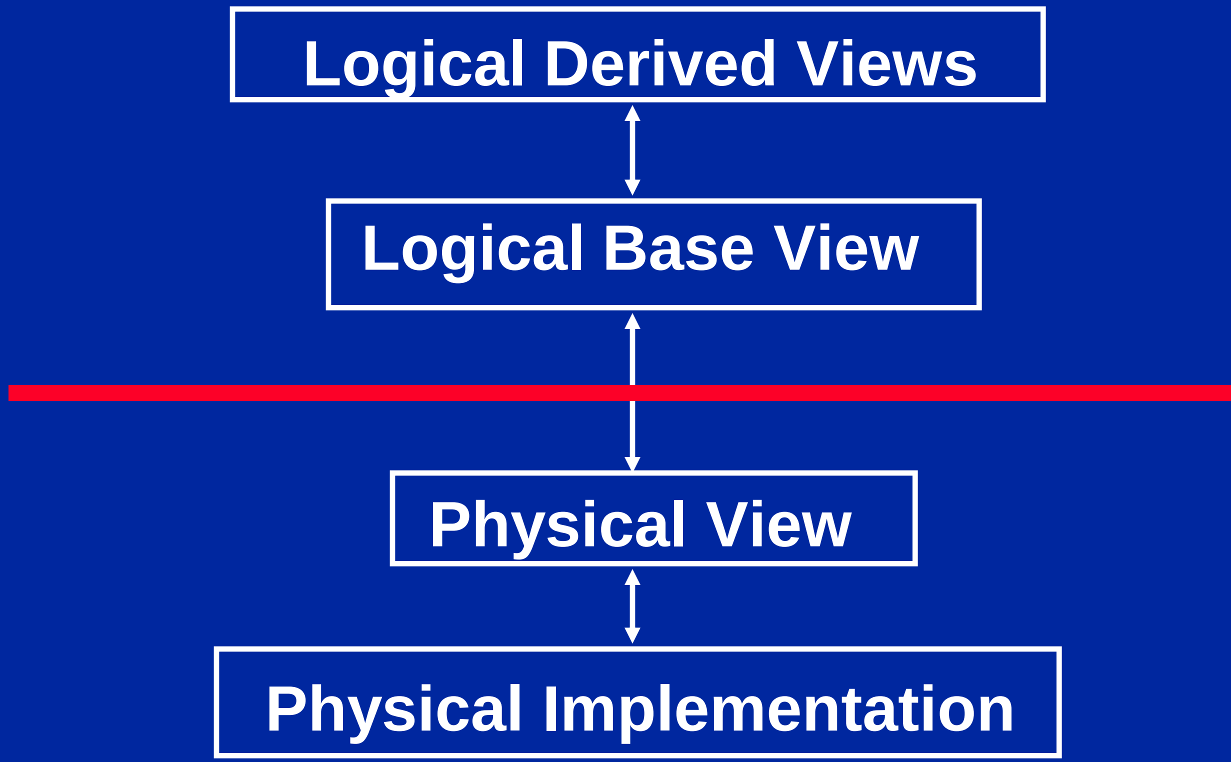
THE SECRETS TO SUCCESS

- **You Must Understand Logical Design**
  - DEPENDENCIES
  - NORMALIZATION
  - THE DATABASE DESIGN PRINCIPLES
    - » THE DATABASE DESIGN PRINCIPLE OF ORTHOGONALITY (MCGOVERAN-DATE)
    - » THE DATABASE DESIGN PRINCIPLE OF COMPLETENESS (MCGOVERAN)
    - » THE DATABASE DESIGN PRINCIPLE OF MINIMALITY (MCGOVERAN)
  - IDENTIFYING PROPER COLLECTIONS OF TABLES
  - GUARANTEEING VIEW UPDATABILITY

THE SECRETS TO SUCCESS

- **Logical**
  - *GUARANTEES ACCESS (RELATIONAL CORRECTNESS AND COMPLETENESS)*
    - » BOTH PROCESS (PERMISSIBLE STATE TRANSITIONS) AND DATA
    - » A SUCCESSFUL TRANSACTION IS A PERMISSIBLE STATE TRANSITION (TAKES DATABASE FROM ONE CONSISTENT STATE TO ANOTHER)
- **Physical**
  - *ADDRESSES EFFICIENCY (PERFORMANCE AND STORAGE)*
    - » BOTH PROCESS (ACCESS METHODS) AND DATA
  - MUST BE A VIEW OF THE LOGICAL MODEL (WHY?)

THE SECRETS TO SUCCESS
LAYERED DESIGN



PHYSICAL DATABASE DESIGN

- **The Design of Storage Structures**
  - FOR PERFORMANCE
  - WITHOUT SUBVERTING RELATIONAL CORRECTNESS!
  - DON'T CONFUSE WITH DESIGN OF THE LOGICAL VIEW!
- **Need Not Be Normalized If. . .**
  - CAN HIDE PHYSICAL DEVIATIONS FROM FROM ALL USERS
  - ALL OPERATIONS MANIPULATE ONLY THAT LOGICAL VIEW
  - PHYSICAL SCHEMA UPDATES NEVER INDUCE LOGICAL ANOMALIES

PHYSICAL DATABASE DESIGN

- **Method**
  - TREAT PHYSICAL SCHEMA AS A SET OF UPDATABLE VIEWS DEFINED FROM THE LOGICAL SCHEMA
    - » ***NOT*** THE REVERSE METHOD (AS IS MORE COMMON)!
  - ENFORCE PHYSICAL MULTI-TABLE CONSTRAINTS VIA TRIGGERS AND INTEGRITY CONSTRAINTS

Remember . . .

The Golden Guarantee of Data Independence

"ALL PHYSICAL COMPLEXITY CAN BE CONCEALED VIA ACCESS THROUGH THE LOGICAL SCHEMA"

PHYSICAL DATABASE DESIGN

- **What is Legitimate?**
  - **A SINGLE LOGICAL RELATION CAN BE REPRESENTED BY TWO OR MORE PHYSICAL TABLES**
    - » **JOIN, UNION, DIFFERENCE**
  - **MULTIPLE LOGICAL RELATIONS CAN BE REPRESENTED BY A SINGLE PHYSICAL TABLE**
    - » **PROJECTION, RESTRICTION**
    - » **REDUNDANT, PRECOMPUTED, AND ALTERNATE COLUMN FORMATS**

DIMENSIONAL SCHEMAS
THE RIGHT WAY

- ***Get the Benefits Without Abandoning Reason!***
  - FULLY NORMALIZE THE LOGICAL DESIGN
  - USE ONLY THE DEPENDENCIES THAT MATTER TO THE APPLICATION - *RELATIVE NORMALIZATION*
    - » *MANY DEPENDENCIES ARE NEVER SEEN BY THE APPLICATION*
    - » *ATTRIBUTES MAY BE COMPLEX (A SET FOR A REPEATING GROUP) - BE CAREFUL!*
  - OPTIMIZE THE PHYSICAL FOR MINIMUM STORAGE
    - » *HIGH SCAN COST OFTEN OUTWEIGHS JOIN COST*
  - MAKE CERTAIN THE PHYSICAL IS COMPATIBLE WITH THE LOGICAL

DATA INDEPENDENCE

THE SECRET TO SCALABLE DESIGN

- **Logical Mostly Independent of Physical**
  - **CAN HIDE STORAGE ALLOCATION AND PERFORMANCE**
  - **PHYSICAL PLATFORM ISSUES NEED BE KNOWN ONLY TO DBMS**
  - **SQL ENTANGLES THESE, ESPECIALLY AT TABLE CREATION**
- **Applications Access *Only* the Conceptual or Logical Schemas**

Result?

A SCALABLE DESIGN!

  - **CAN CHANGE THE APPLICATION CODE AND THE PHYSICAL SCHEMA INDEPENDENTLY!**
  - **ADDRESS INVARIANT AND VARIABLE REQUIREMENTS INDEPENDENTLY**
  - ***ENABLES SCALABLE PLATFORM ARCHITECTURE CHANGES***

Questions?

BIOGRAPHY

David McGoveran is an industry analyst, and an international management and technology consultant . He is president of Alternative Technologies (Boulder Creek, CA), specialists in solving difficult relational applications problems since 1981. Having authored numerous technical articles and co-authored several books (including those with Chris Date), his newest book is <u>A Zero Management: Business Success in the New Millenium.</u>

This seminar is based on his workshops: <u>The Client/Server University:</u> <u>Designing Effective Databases</u>, and <u>Achieving Scalability.</u>

PLEASE FILL OUT YOUR EVALUATIONS...

Thank you!