Folien aus VLDB'04-Tutorial:
Self-Managing Technology in Database Management Systems,
S. Chaudri, Microsoft; B. Dageville, Oracle; G. Lohman, IBM

Autonome DatenBankSysteme

Norbert Ritter
Verteilte Systeme und Informationssysteme (VSIS)
Datenbanken und Informationssysteme

Outline

- Motivation
  - TCO and Complexity
  - Self-Management
  - Self-Managing DBMSs
- Self-Managing DBMS-Features
  - IBM DB2
  - MS SQL Server
  - Oracle
- Conclusion (our Seminar)

Motivation - TCO and Complexity

Today

- Disks on laptops have more capacity than most need
  - 1 Terabyte for $1199: http://www.lacie.com/products/product.htm?id=10118
- CPUs cost less than a good meal
  - Complete "bare bones" machines for $200 (retail)
  - Example: http://shop1.outpost.com/product/3847537
- Network capacity glut permits streaming voice and video
- But people, …

Motivation - TCO and Complexity

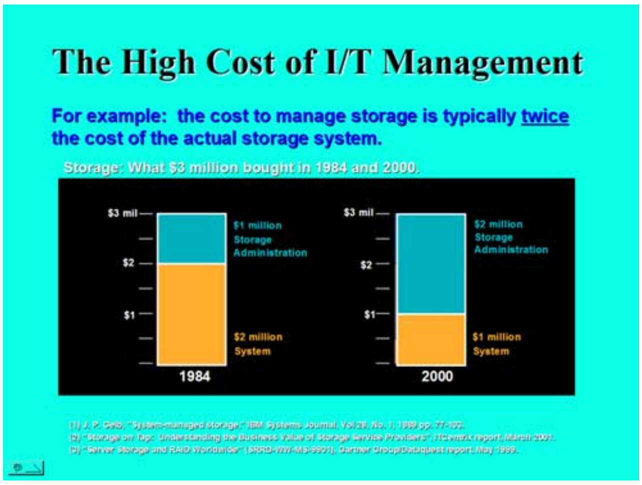
Cost of Labor Rarely Decreases (Despite Outsourcing)

Employment Cost Index (1989 = 100):

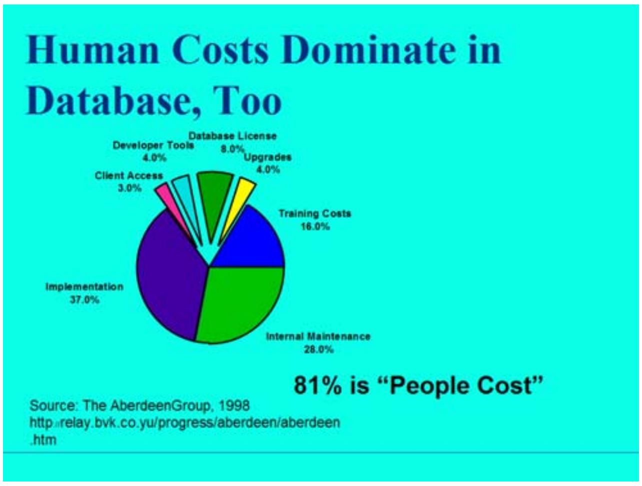
Total comp., Professional, Specialty, & Technical Occupations (all civilian)

- **Source:** U.S. Bureau of Labor Statistics (http://data.bls.gov/servlet/SurveyOutputServlet) **Series Id:** ECU11121I

Motivation - TCO and Complexity



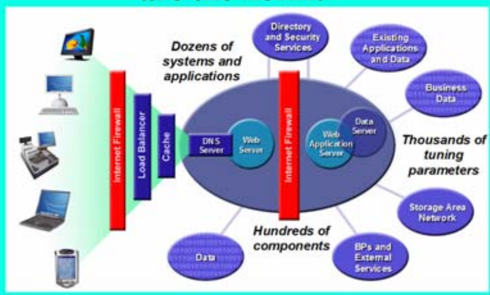
Motivation - TCO and Complexity



Motivation - TCO and Complexity

Houston, we have a problem …

Complex heterogeneous infrastructures are the norm!



Motivation - TCO and Complexity

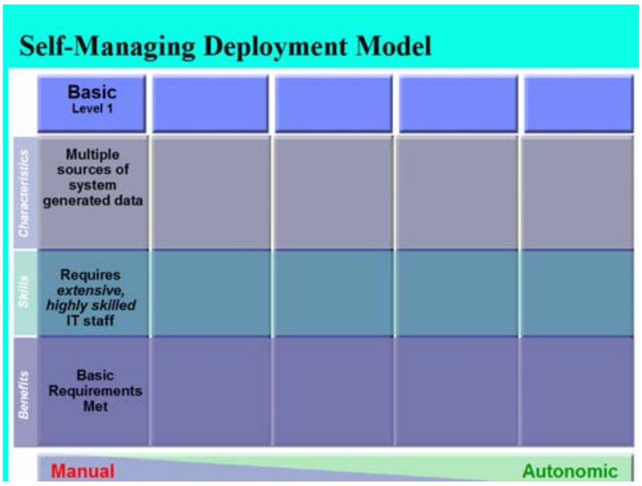
Managing Increasing Complexity

- Increase in Complexity & Size of Applications
  - Database workloads are more mixed (e.g. OLTP and complex reporting).
  - Database workloads are more dynamic.
  - Data size is growing rapidly
    - Multi-terabytes are no longer the exception!
- DBMS vendors have responded to these challenges by
  - Enlarging the scope of existing features
    - New access structures, complex optimizations
    - Complex hardware architectures like clusters or MPPs
  - Adding new features in the server
    - Objects, XML, OLAP, data mining, ETL
    - Replication, high-availability, …

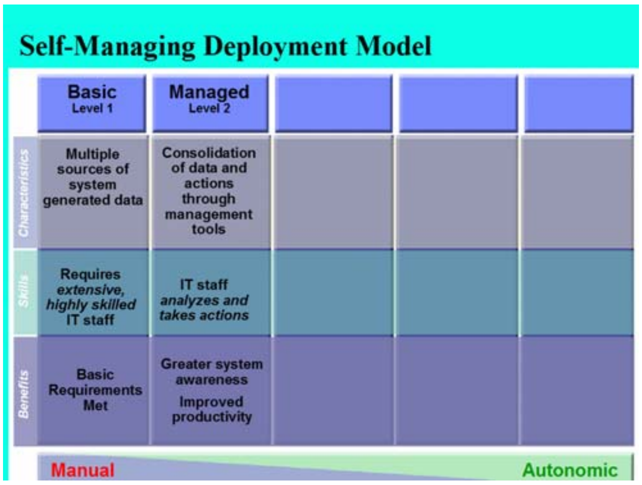
Managing/tuning a modern database system requires a very high degree of expertise!

Motivation – Self-Management

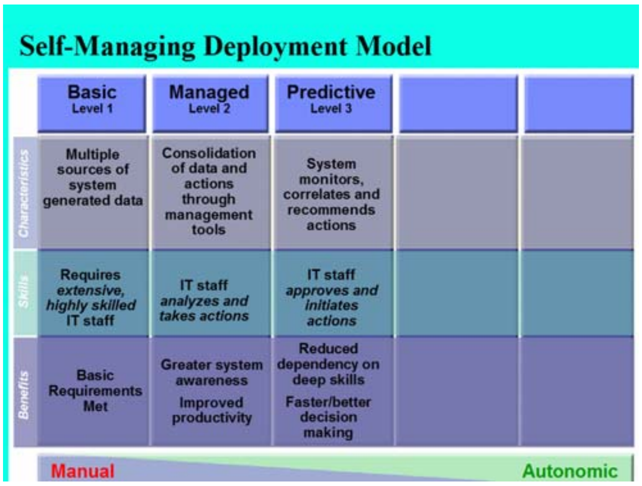
Motivation – Self-Management



Motivation – Self-Management



Motivation – Self-Management



Motivation – Self-Management

Self-Managing Deployment Model

	Basic Level 1	Managed Level 2	Predictive Level 3	Adaptive Level 4	
Characteristics	Multiple sources of system generated data	Consolidation of data and actions through management tools	System monitors, correlates and recommends actions	System monitors, correlates and takes action	
Skills	Requires *extensive, highly skilled* IT staff	IT staff *analyzes and takes actions*	IT staff *approves and initiates actions*	IT staff *manages performance* against SLAs	
Benefits	Basic Requirements Met	Greater system awareness Improved productivity	Reduced dependency on deep skills Faster/better decision making	Balanced human/system interaction IT agility and resiliency	

Manual — Autonomic

N. Ritter, ADBS, WS06/07 ***slide: http://www.cs.toronto.edu/vldb04/protected/Tutorial3-Intro.pps*** 13

Motivation – Self-Management

Self-Managing Deployment Model

Evolution, not revolution

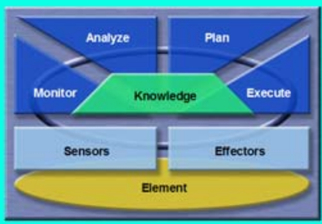
	Basic Level 1	Managed Level 2	Predictive Level 3	Adaptive Level 4	Autonomic Level 5
Characteristics	Multiple sources of system generated data	Consolidation of data and actions through management tools	System monitors, correlates and recommends actions	System monitors, correlates and takes action	Integrated components dynamically managed by business rules/policies
Skills	Requires *extensive, highly skilled* IT staff	IT staff *analyzes and takes actions*	IT staff *approves and initiates actions*	IT staff *manages performance* against SLAs	IT staff *focuses* on enabling business needs
Benefits	Basic Requirements Met	Greater system awareness Improved productivity	Reduced dependency on deep skills Faster/better decision making	Balanced human/system interaction IT agility and resiliency	Business policy drives IT management Business agility and resiliency

Manual — Autonomic

N. Ritter, ADBS, WS06/07 ***slide: http://www.cs.toronto.edu/vldb04/protected/Tutorial3-Intro.pps*** 14

Motivation – Self-Management

Core Building Blocks for an open architecture



- **An autonomic element contains a continuous control loop that monitors activities and takes actions to adjust the system to meet business objectives**
- **Autonomic elements learn from past experience to build action plans**
- **Managed elements need to be instrumented consistently**

Motivation – Self-Management

Architectural Trade-Offs (1 of 2)

- What granularity for such "autonomic elements"?
  - – Per database?
  - – Per CPU?
  - – Per component (e.g., DBMS, App Server, …)?
  - – Per complete system?
- Distributed?
  - + Local control
  - + Simpler
  - + Scalable
  - - Don't have the "big picture"
  - - Unstable "Tug of war" with other components possible
- Centralized?
  - + Have broader view of cause & impact
  - - Won't scale well
  - - Relies on communication speed, availability, & standards

Motivation – Self-Management

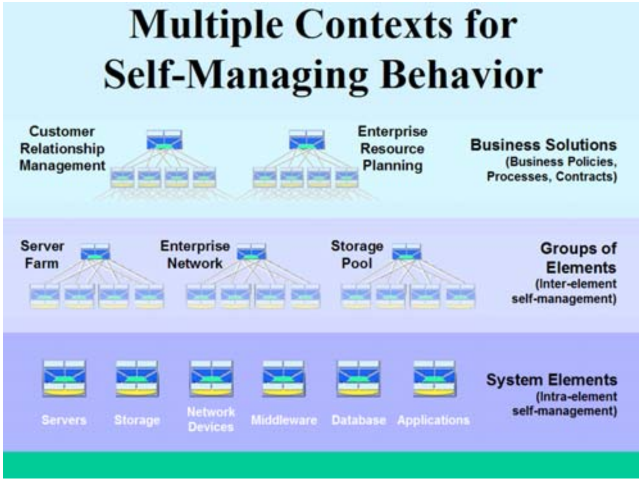
Architectural Trade-Offs (2 of 2)

- Hybrid (hierarchical)?
  - Blend: both distributed & centralized control elements
  - Communicate only necessary info to
    - Other components
    - Central controller

  + Have broad view as well as local control
  + Scalable
  - Relies on communication speed, availability, & standards
  - Complex interactions between controllers
  - Can still have unstable conflicts

N. Ritter, ADBS, WS06/07 ***slide: http://www.cs.toronto.edu/vldb04/protected/Tutorial3-Intro.pps*** 17

Motivation – Self-Management



N. Ritter, ADBS, WS06/07 ***slide: http://www.cs.toronto.edu/vldb04/protected/Tutorial3-Intro.pps*** 18

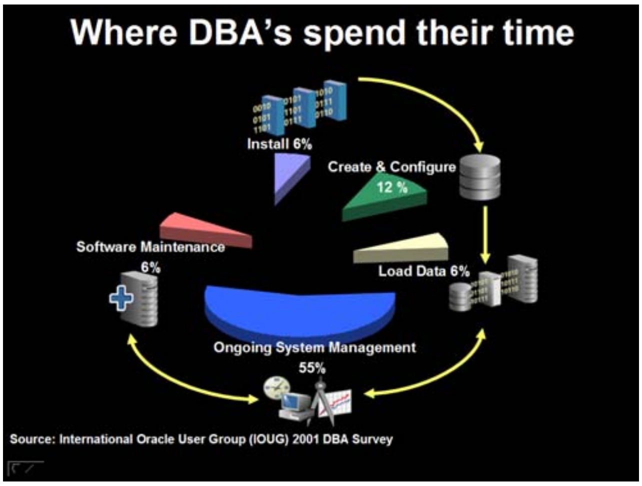
Motivation – Self-Managing DBMSs

Huge Scope of DBA Responsibilities

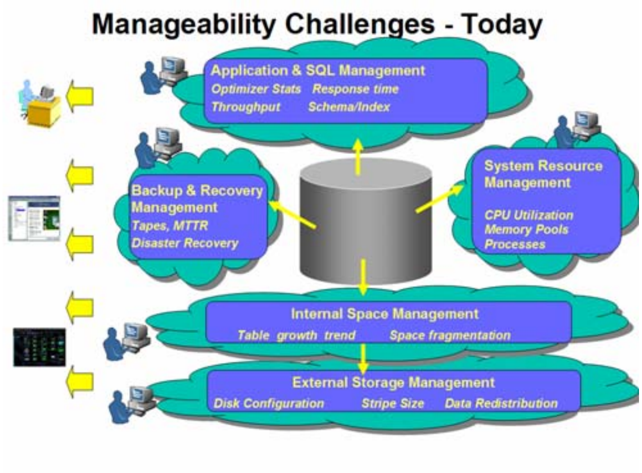
- Initial Design & Layout
  - Hardware configuration
  - Logical database design
  - Physical data layout (partitioning, allocation to nodegroups, clustering)
  - Auxiliary data structures (indexes, view materializations)
  - Configuration parameters (hundreds!)
  - Security policies, groups, userids

- Dynamic Monitoring & Adjustment
  - Database statistics to collect and when
  - Clustering and Reorganization
  - Memory allocation, esp. buffer pool sizes
  - System / query status
  - Problem determination (deadlocks, bad plans, ...)
  - Visualization of all the above

Motivation – Self-Managing DBMSs



Motivation – Self-Managing DBMSs



N. Ritter, ADBS, WS06/07 ***slide: http://www.cs.toronto.edu/vldb04/protected/Tutorial3-Intro.pps*** 21

Motivation – Self-Managing DBMSs

Research Topics / Issues (1 of 2)

- Capacity planning (modeling & estimation)
  - How model systems with limited specification?
  - How maintain model with evolving HW & SW?
- Installation
  - Dependency graph of prerequisite versions, configurations
- Database Design
  - Logical Design (application design, normalization)
  - Physical Design – how to decide:
    - Selection of indexes, materialized views, etc.
    - Data placement (clustering, partitioning, etc.)
    - Dynamic storage provisioning
- Performance tuning
  - How automate determination of poor performers?
  - How dynamically re-configure system in response to load changes?
- Maintenance – when / how to perform
  - Backups?
  - Reorganizations?
  - Statistics collection?
  - Upgrades?

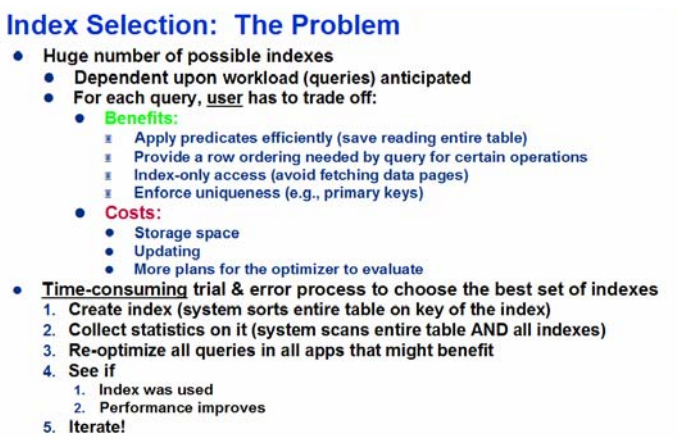
N. Ritter, ADBS, WS06/07 ***slide: http://www.cs.toronto.edu/vldb04/protected/Tutorial3-Intro.pps*** 22

Motivation – Self-Managing DBMSs

Research Topics / Issues (2 of 2)

- Self-Healing
  - How much monitoring data to collect?
  - How do you know if your system is "firing on all cylinders"?
  - How do you isolate problems from noise of diagnostics?
  - How do you correlate logs from components on different machines w/ diff. clocks?
  - How do you isolate root cause from cascading error messages?
  - Fuzzy searching of symptom databases
  - How do you automatically generate diagnostics to resolve ambiguous problems?
  - How do you model and determine the cause & repair for problems never before seen?
  - How do you determine the best fix for a problem, even if the cause is known?
  - How do you build repair rules automatically from past successes & failures?
- System Control
  - Scheduling & prioritization of tasks
  - How do you resolve conflicting rules & priorities?
  - How do you make progress on maintenance without impacting production workload?
  - How do you avoid instability and "thrashing" (control theory)?
  - How much monitoring is enough to resolve problems but not impact production?
  - How do you learn from past successes & failures?

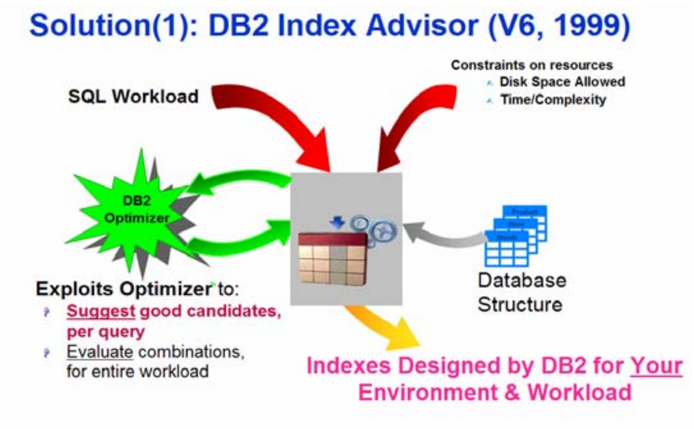
Self-Managing DBMS-Features
IBM: Index Advisor



Index Selection: The Problem

- Huge number of possible indexes
  - Dependent upon workload (queries) anticipated
  - For each query, user has to trade off:
    - Benefits:
      - Apply predicates efficiently (save reading entire table)
      - Provide a row ordering needed by query for certain operations
      - Index-only access (avoid fetching data pages)
      - Enforce uniqueness (e.g., primary keys)
    - Costs:
      - Storage space
      - Updating
      - More plans for the optimizer to evaluate
- Time-consuming trial & error process to choose the best set of indexes
  1. Create index (system sorts entire table on key of the index)
  2. Collect statistics on it (system scans entire table AND all indexes)
  3. Re-optimize all queries in all apps that might benefit
  4. See if
     1. Index was used
     2. Performance improves
  5. Iterate!

Self-Managing DBMS-Features
IBM: Index Advisor



Self-Managing DBMS-Features
IBM: Index Advisor

Index Advisor (DB2 V6) – The Math

- Variant of well-known "Knapsack" Problem
- Greedy "bang-for-buck" solution is optimal, when integrality of objects (indexes) is relaxed

- For each query Q:
  - Baseline: Explain each query w/ existing indexes, to get cost E(Q)
  - Unconstrained: Explain each query in RECOMMEND INDEXES mode, to get cost U(Q)
  - Improvement ("benefit") B(Q) = E(Q) - U(Q)
- For each index I used by one or more queries:
  - If query Q used index I, assign "benefit" B(Q) to index I: B(I) = B(I) + B(Q)
  - Assign "cost" C(I) = size of index in bytes
  - Order indexes by decreasing B(I) / C(I) ("bang for buck")
  - Cut off where cumulative C(I) exceeds disk budget
- Iterative improvement: exchange handfuls of "winners" with "losers"

- **REFN: "DB2 Advisor: An Optimizer Smart Enough to Recommend its Own Indexes", ICDE 2000 (San Diego), Valentin, Zuliani, Zilio, Lohman, et al.**

Self-Managing DBMS-Features
IBM: Configuration Advisor

Configuration Parameters

- **The Problem:**
  - Almost 150 configuration parameters in DB2 UDB
  - Users didn't know:
    - How to choose the right values
    - Possible interactions between them
  - Had to stop and restart DB2 to have them go into effect
    - Bad for availability, too!

- **Solution(1):**
  - Make many configuration parameters dynamic!
  - No need to stop and restart DB2 to change them
  - Not easy to implement, e.g. shrinking buffer pool
  - Shipped in DB2 UDB V8.1 (2002)
  - Prerequisite to automatically tuning them

Self-Managing DBMS-Features
IBM: Configuration Advisor

Solution(2): Configuration Advisor (V8.1, 2002)

- **What is it?**
  - Sets ~36 configuration parameters key to performance, including:
    - Memory heaps (buffer pool, sort heap, statement cache)
    - Connections (max and average, remote/local)
  - Based upon answers to 7 high-level questions
  - Equations from performance experts relate parameters
- **Enhanced in V8.1:**
  - Available in V7 as "Performance Configuration Wizard"
  - More sophisticated model in V8.1
  - Easier to invoke via:
    - CREATE DATABASE command extension
    - AUTOCONFIGURE command
  - Better decisions for OLTP and DSS workloads
  - Surprising benchmark results
    (well-known, industry-standard OLTP workload)

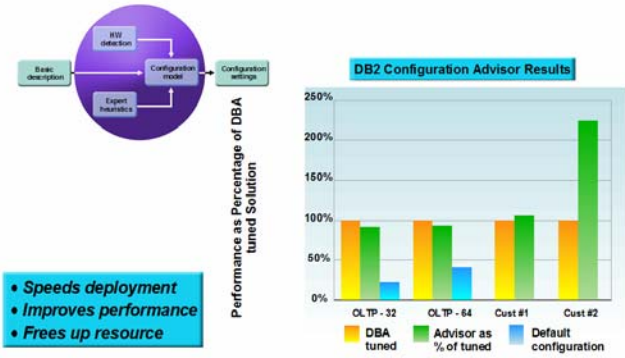
Self-Managing DBMS-Features
IBM: Configuration Advisor

Configuration Advisor: The Questions

- Percentage of Real Memory to dedicate to DBMS
- OLTP vs. Complex query vs. Mixed
- Length of Transaction (typical # of SQL queries per transaction)
- Relative priority of Recovery vs. Query speed
- Number of Local and Remote Connections
- Whether the database is populated or not
- Isolation Level

Self-Managing DBMS-Features
IBM: Configuration Advisor

DB2 Configuration Advisor vs. Human Experts



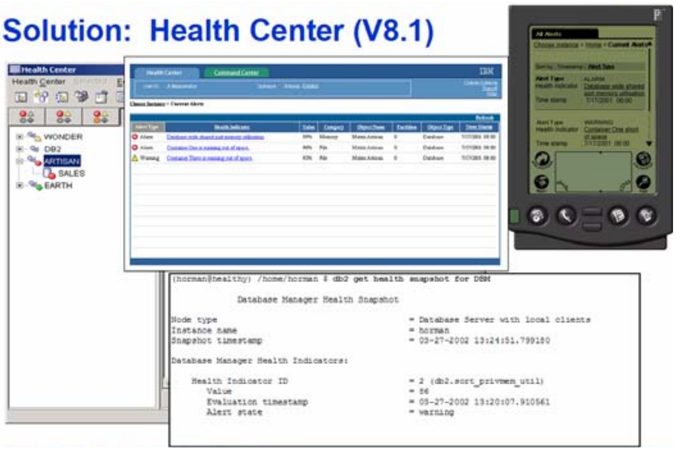
Self-Managing DBMS-Features
IBM: Health Advisor

Health Monitoring

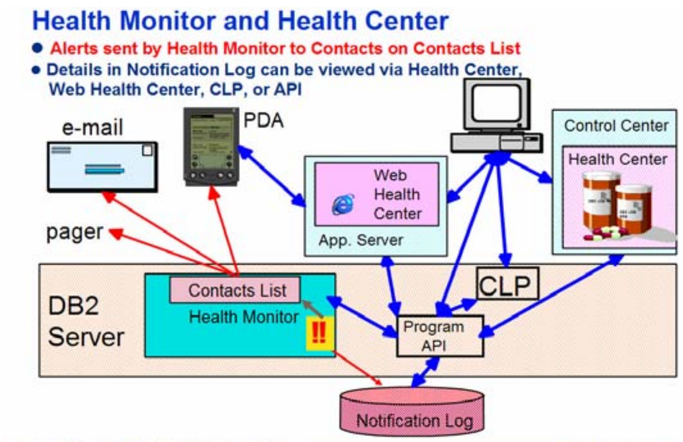
- **The Problem:**
  - How do you know if DB2 is running okay, performing well?
  - What do you do if you do manage to figure out it's "unhealthy"?
  - Too difficult to determine what to monitor and when to monitor it
  - Need to set up monitors, notification & resolution mechanisms
- **The Solution: Health Center**
  - DB2 monitors its own health right out of the box
  - Notifies user upon encountering unhealthy conditions
  - Advises on severity of condition, and suggests resolutions
  - Initiates corrective action if required, requested
  - Easy installation: just provide an e-mail or pager address
  - User can modify thresholds for notification

Self-Managing DBMS-Features
IBM: Health Advisor

Solution: Health Center (V8.1)

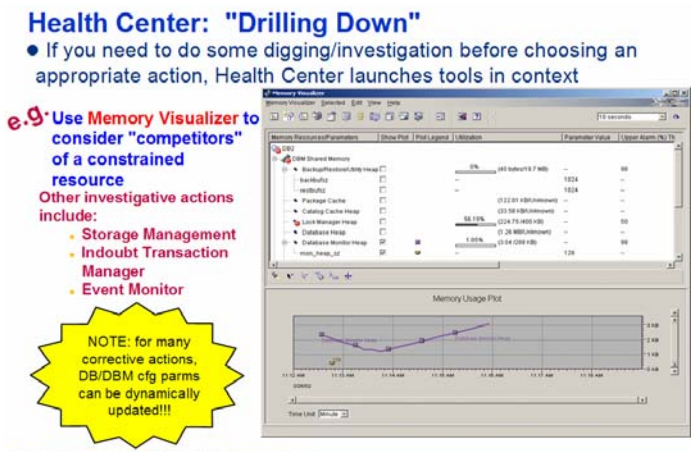


Self-Managing DBMS-Features
IBM: Health Advisor



N. Ritter, ADBS, WS06/07 ***slide: http://www.cs.toronto.edu/vldb04/protected/Tutorial3-DB2.pps*** 33

Self-Managing DBMS-Features
IBM: Health Advisor



N. Ritter, ADBS, WS06/07 ***slide: http://www.cs.toronto.edu/vldb04/protected/Tutorial3-DB2.pps*** 34

Self-Managing DBMS-Features
IBM: Design Advisor

Design Advisor ("Stinger")

- An extension of existing Index Advisor (V6)
- Headquarters for all physical database design
- Recommends any combination of:
  - **Indexes**
  - **Materialized Views (Materialized Query Tables (MQTs))**
    - **Called Automatic Summary Tables (ASTs) before V8.1**
  - **Partitioning of tables (in partitioned environment)**
  - **Multi-Dimensional Clustering (MDC) storage method (New in V8.1)**
- Takes interactions of these into consideration
- Status:
  - **Coming soon ("Stinger")!**
  - **Beta testing on customer databases now!**

- **REFNS:**
  - "DB2 Design Advisor: Integrated Automatic Physical Database Design", **VLDB 2004**
  - "Recommending Materialized Views and Indexes with IBM's DB2 Design Advisor", **IEEE Intl. Conf. on Autonomic Computing (ICAC 2004)**
  - "Trends in Automating Database Physical Design", **IEEE 2003 Workshop on Autonomic Computing Principles and Architectures**. Banff, Alberta, August 2003

Self-Managing DBMS-Features
IBM: Design Advisor

Design Advisor: Partition Advisor

- **Scope:**
  - DB2 "partitioned environment" (was called EEE prior to V8.1)
  - "Shared-nothing" parallelism
  - Data stored horizontally partitioned
    - **In a partition group, spread across specified partitions**
    - **Based upon hashing of partitioning column(s)**
    - **May be replicated across all partitions of partition group**
  - Need to co-locate similar values for joins, aggregation in queries
  - Partitioning required for a given table may be different
    - **Between queries**
    - **Even within a query (joined on different columns)!**

- **Problem:** What is optimal partitioning for each table, given:
  - **Workload of queries**
  - **Schema, including set of partition groups & tablespaces**
  - **Statistics on database**

Reference: "Automating Physical Database Design in a Parallel Database", ACM SIGMOD 2002 (Madison, WI, June 2002)

Self-Managing DBMS-Features
IBM: Automated Statistics Collection

Automating Statistics Collection:

- **Problem:**
  - Optimizer requires that statistics on database be
    - Up to date (after updates)
    - Complete (multi-column)
  - User must invoke RUNSTATS
- **Solution**: Automate RUNSTATS
  - ***Invocation*** scheduled and prioritized
  - ***Run silently*** as a background daemon
    - Throttled based upon workload
  - **LEO** the **LE**arning **O**ptimizer determines which ***statistics needed***
    - Based upon learning from past queries
    - Groups of columns
      - Enables correlation detection
    - Communicated to RUNSTATS via statistical "profiles"
- Shipping in DB2 "Stinger"
- **Refn**: "Automated Statistics Collection in DB2 Stinger", **VLDB 2004**

Self-Managing DBMS-Features
IBM: Automated Statistics Collection

Automating Statistics Collection:
LEO the LEarning Optimizer Determines Statistics Profiles

Refn: "LEO -- DB2's LEarning Optimizer", Intl. Conf. on Very Large Data Bases 2001 (Rome, Sept. 2001)

Self-Managing DBMS-Features
IBM: Automated Statistics Collection

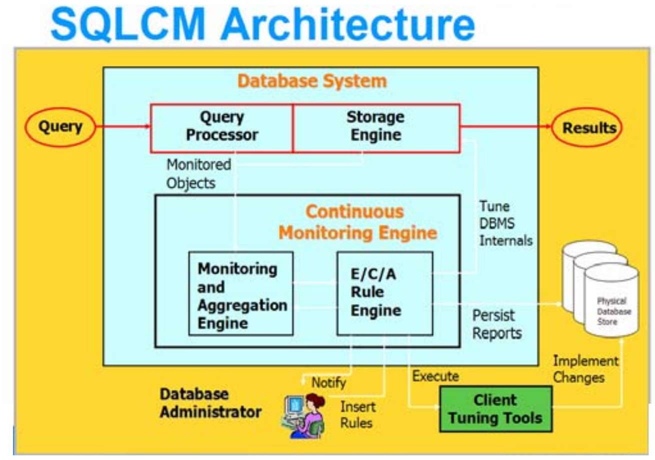
LEO Motivation

- Cost depends heavily on number of rows processed (cardinality)
- Optimizer's model limited by simplifying assumptions
  - Especially due to statistical correlation between columns
  - EXAMPLE: WHERE Make = 'Honda' AND Model = 'Accord'
  - Impossible to know a priori which columns are correlated!
- Why not use actual results from executed queries to
  - Validate statistics and assumptions
  - Advise when/how to run expensive statistics collection
  - Gather statistics that reflect the workload
  - Repair the model for optimizing "similar" future queries
- Could achieve automatically
  - Better quality plans
  - Reduced customer tuning & administration time
  - Reduced IBM support time
- Part of Automated RUNSTATS in "Stinger"

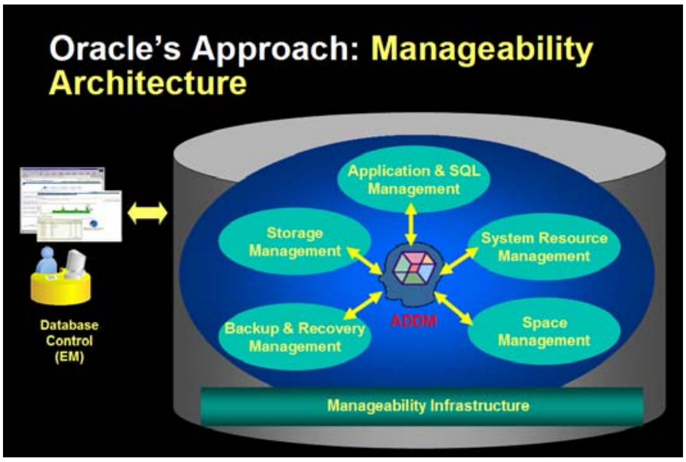
Self-Managing DBMS-Features
MS SQL Server

- Self-Tuning Physical Design
  - Automated index selection
  - 'What-if' index analysis
  - Automated selection of materialized views
  - Automation of horizontal and vertical partitioning
  - Tuning advisor
- Statistics Management
  - Automated statistics selection
  - Efficient statistics creation
  - Automated statistics exploitation for optimization
  - Statistics estimation
- Monitoring, workload analysis and Management
  - Continuous monitoring (SQLCM)
  - Workload compression
  - Workload summarization

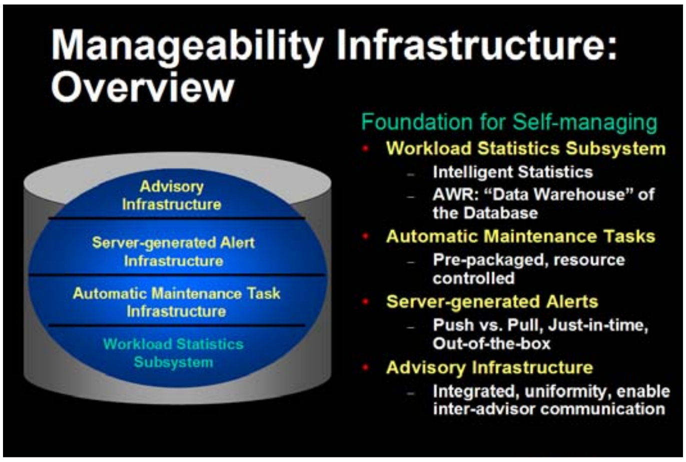
Self-Managing DBMS-Features
MS SQL Server



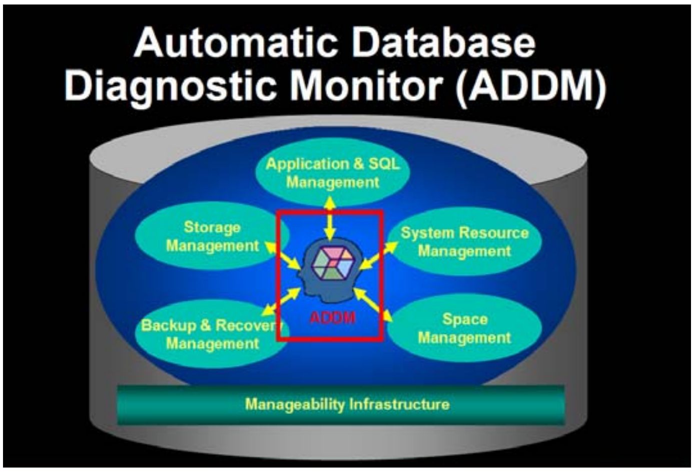
Self-Managing DBMS-Features
Oracle



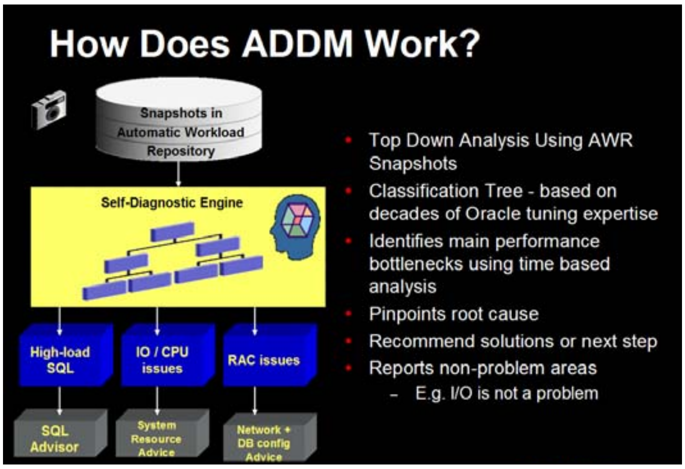
Self-Managing DBMS-Features Oracle



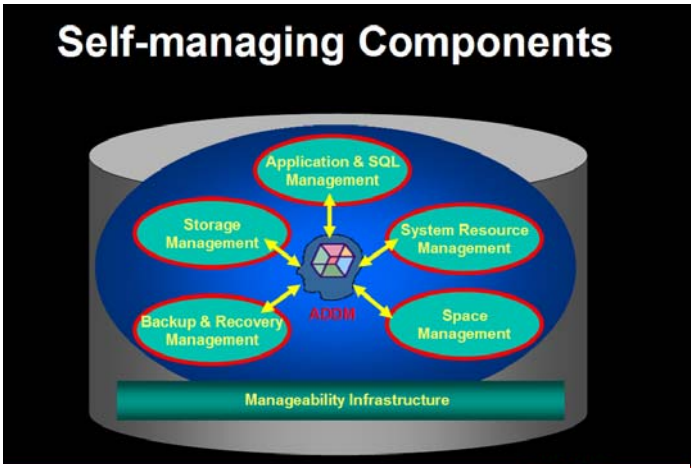
Self-Managing DBMS-Features Oracle



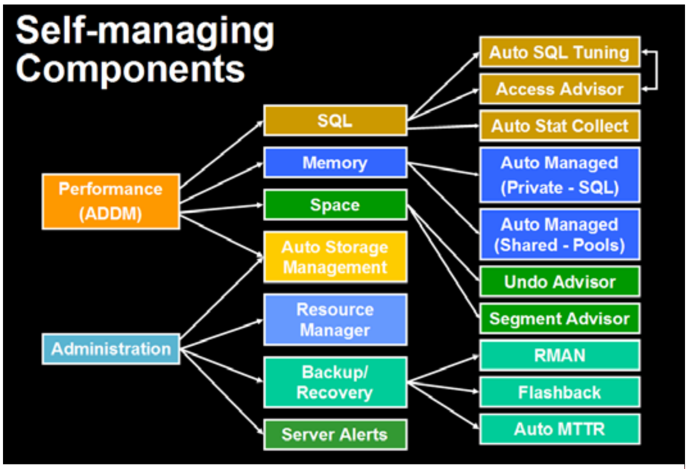
Self-Managing DBMS-Features Oracle



Self-Managing DBMS-Features Oracle



Self-Managing DBMS-Features Oracle



Conclusion

- Areas of DBMS self-management (orthogonality?)
  - Optimization and statistics
  - Physical design: indexes, materialized views, clustering, partitioning
  - Configuration and 'health'
  - (Further) maintenance issues: backup(/recovery), reorganization installation, upgrade
  - ...
- Our seminar
  - Each group
    - Complete overview of autonomic features of corresponding product as well as corresponding research activities **(what?)**
    - Detailed view to basic mechanisms, algorithms, approaches **(how?)**