U21
Implementation Fundamentals for The Latest DB2 UDB Partitioned Database

Philip K. Gunning, DGI

Anaheim, CA **Sept 9 - 13, 2002**

Objectives

- Define New Terminology
- Planning
  - Partitioned Database Fundamentals
- Installation
  - Prerequisites, Requirements, Key Steps
- Implementation
  - Backup & Recovery
  - Performance
  - Monitoring and Tuning

TERMINOLOGY

Terminology

DB2 V7.2 and prior	DB2 V8.1
Node – used to refer to partition or physical server	PARTITION
Nodegroup	Database Partition Group
Enterprise Extended Edition (EEE)	Enterprise Server Edition (Database Partition Server, DB2 Partitioned Database, “Super Scalability Feature”, licensing feature)
Enterprise Edition (EE)	Enterprise Server Edition
Workgroup Edition	Workgroup Server Edition


MAPQUEST
0 100mi 100km
Chandigarh
Dehra Dun
Patiala
Karnal
Muzzaffarnagar
New Delhi
Jahngirabad
Aligarh
Mathura
Agra
India
Nepal
Kathmandu
Bhutan
Faizabad
Bettiah
Singari
Gwalior
Kanpur
Muzzaffarpur
Dhuburi
Jaunpur
Patna
Jhansi
Varanasi
Shillong
Allahabad
Bhagalpur
Mirzapur
Gaya
Rewa
Bangladesh
Sagar
Murwara
Dhanbad
Agartala
Bhopal
Jabalpur
Ranchi
Asansol
Kolkata
Jamshedpur
Bilaspur
Raurkela
Chittagagong
Khandwa
Nagpur
Durg
Sambalpur
PLANNING
© IBM Corporation 2002
IBM Data Management Technical Conference
©2002 MapQuest.com, Inc.; ©2002 AND Data Solutions B.V.

Planning

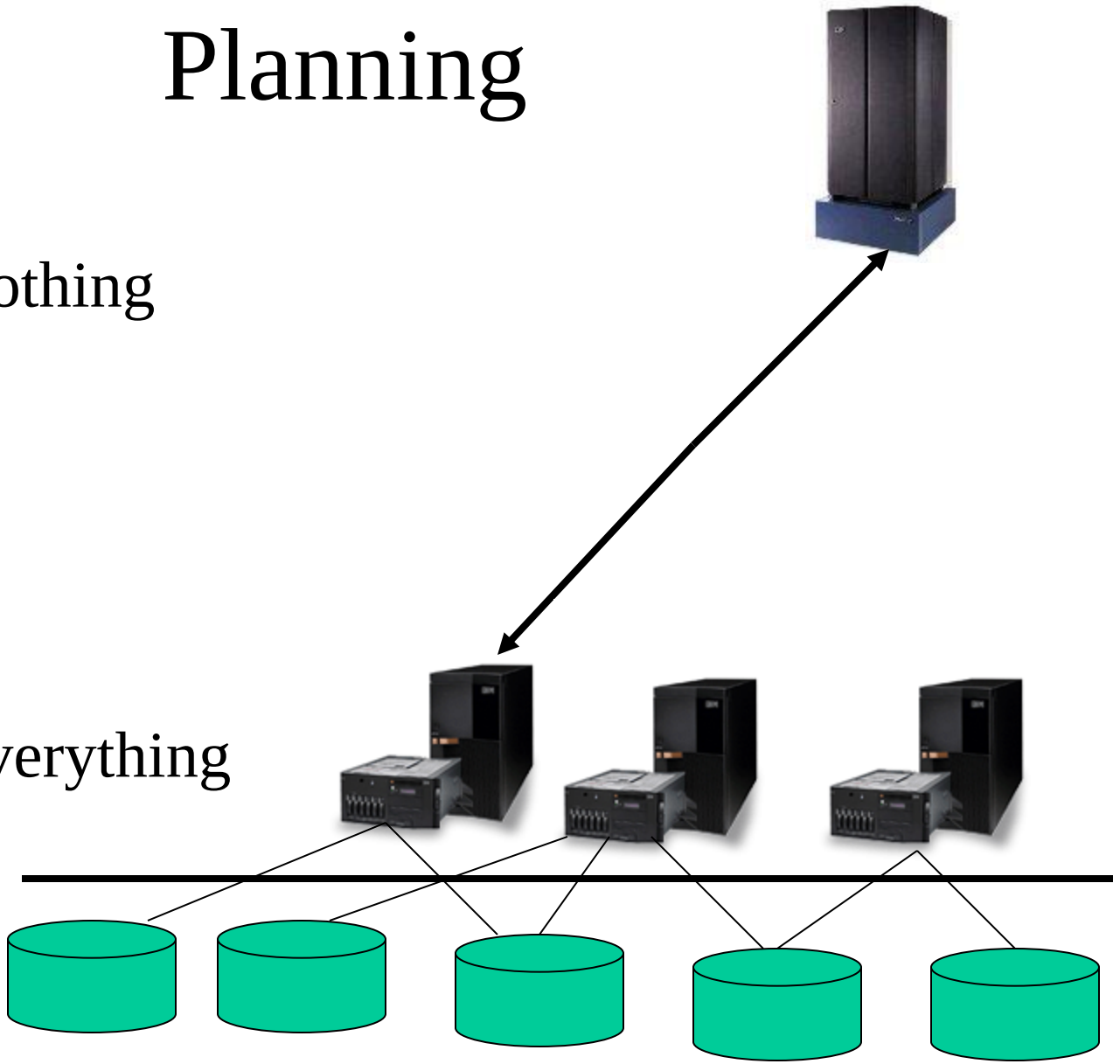
- New to Partitioned Database Environment
- New to Hardware
- New to Your Organization
- New Skills required
- Selection of Partitioning Key
- Need for new Monitoring and Tuning approach

Attendee Notes

- Planning must cover many different areas. If you or your organization is new to a DB2 partitioned database environment then you will have to decide how to partition your data, may have a new Operating System to support, new system administration requirements or environmental (Data Center) changes. New hardware may be required.
- You will need to update your skills regarding administering and tuning a DB2 partitioned databases
  - How do you Administer?
  - What is a catalog partition?
  - How do you add a partition?
  - How do you manage many more logs on more than one server? How do you recover the database?
- I can't over-emphasize the importance of training, you can do a much better job for your organization and customers with training prior to implementation
  - Sometimes it is not possible but you should still get it when the opportunity arises
- What columns are good candidates for partitioning keys?
- How do I optimally partition the data?
- How do you monitor and tune multiple partitions?

Planning

- Shared-Nothing
- Shared-Everything



Attendee Notes

- Shared-Everything architectures are typified by Symmetric Multi-processors (SMP) servers with multiple CPUs that share memory, bus, and disks
- Shared-Everything scalability though is limited due to:
  - Can scale by adding CPUs and memory Law of Diminishing Returns applies
  - Overhead in managing multiple CPUs and resources leads to scalability limitations
- ccNUMA architecture greatly improves on SMP limitations
- Shared-Nothing architecture is implemented in DB2 ESE with Partitioning option
  - ESE Partitioned Database spread over multiple physical servers with their own CPU, memory and Disk
  - No sharing so very little overhead involved
  - "Super Scalability"
    - Add database partitions
    - Add servers
    - Add CPUs
- Implemented with Scalar Power Parallel (SP) architecture
  - Frame houses servers (nodes)
  - Interconnected with High Speed Switch
  - Control Workstation
- Limitations of Shared-Nothing
  - Complexity
  - Shipment of data to partitions as a result of poor database design

Planning

- ESE Partitioned Database can support up to 1,000 partitions and 64GB of data (per partition)
- That's a lot of DATA!

Planning

- Partitioned data versus non-partitioned data
- A simple view:
  - Non-partitioned
    - Data fully contained within 1 table on 1 server
  - Partitioned
    - Data spread across multiple physical database partitions (usually on multiple servers)
    - A hashing algorithm is used to evenly partition the data

Planning

- Partitioning Keys
- Rules
- New V8.1 Support for Identity Columns

Planning

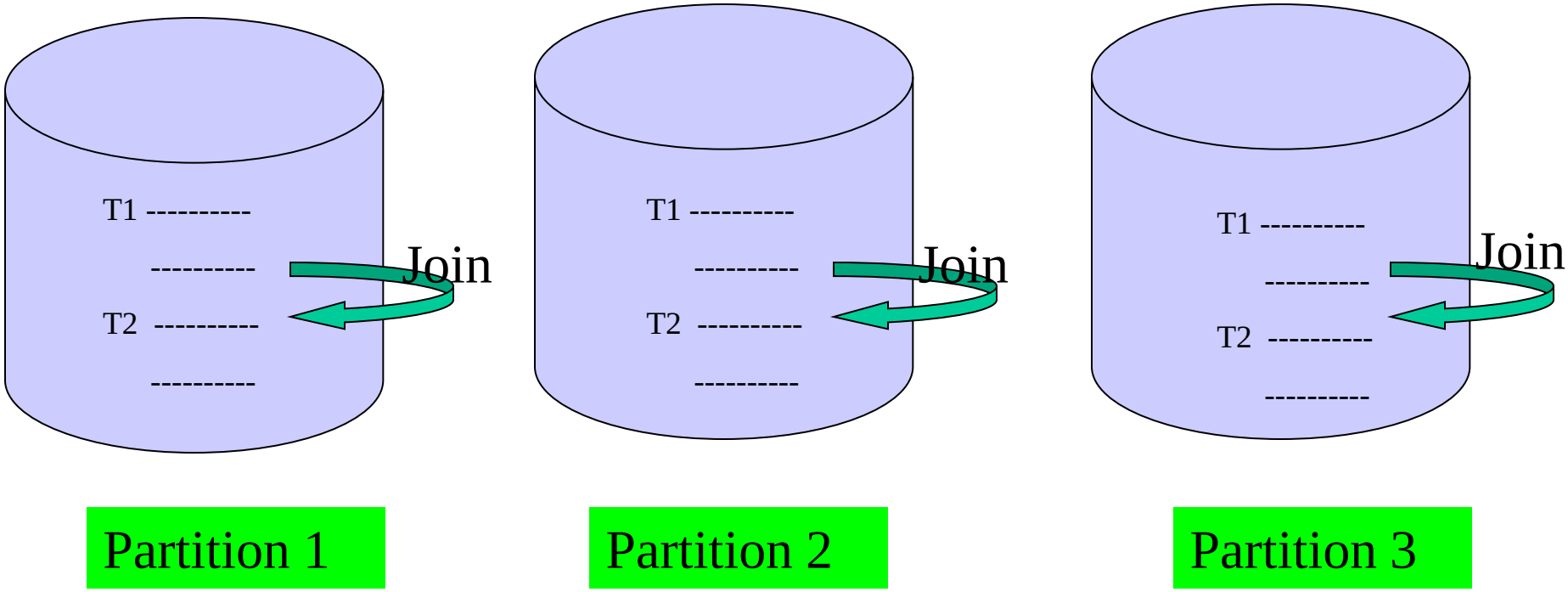
- Rules for selecting good candidate partitioning keys
  - Frequently used columns
  - Columns that have a high proportion of different values
  - Integer columns are more efficient than character columns which are more efficient than decimal
  - Equijoin columns
  - Use smallest number of columns possible

Attendee Notes

- A Partitioning Key is used to partition data over multiple database partitions with each partition containing a portion of the data
- The following rules govern the use of partitioning keys:
  - Primary key or unique index must include the partitioning key
  - Long fields cannot be part of the partitioning key
  - If no partitioning key is specified, the first column that is not a long field is used
  - Tables containing only long fields cannot have partitioning keys and can only be placed into single-partition database partition groups

Planning

- Rules regarding Collocated Joins



Attendee Notes

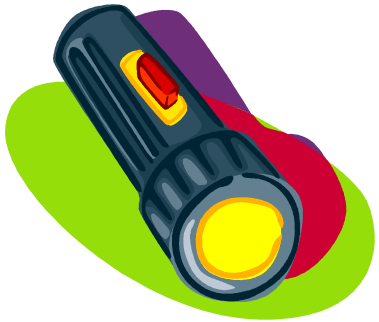
- Collocation between two joining tables means having to have the matching rows of the two tables always in the same database partitions
- Design your database to make maximum use of collocated joins as this avoids having the database manager ship data between partitions

Planning

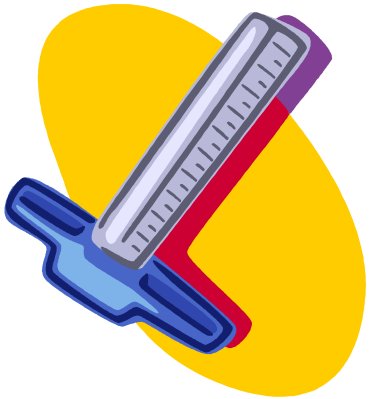
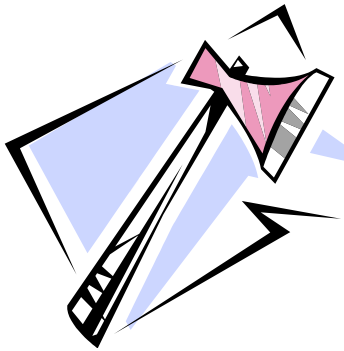
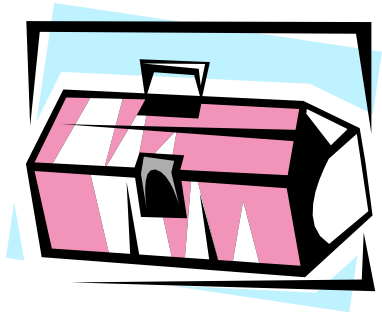
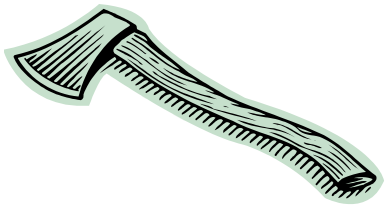
- Tables are collocated when they satisfy the following requirements:
  - Tables must be in same database partition group (Multi-partition database partition group)
  - The database partition group of both the tables must have the same partitioning map
  - The partitioning key from both tables must have the same number of columns

Planning

- Tables are collocated when they satisfy the following requirements (continued):
  - Corresponding partitioning key columns must be partition compatible (the same or similar data types)
- A collocated join will occur if two collocated tables are joined using all of the columns in the partitioning key



INSTALLATION

Installation

- Three userids required
  - Instance Owner
    - Must have same group id
  - UDF Owner
    - The userid used for external functions
  - DAS Owner
    - Userid for DAS instance owner

Installation

- Make sure some preliminary steps are performed
  - Make sure NFS is running
- Create and mount the file system in UNIX
  - Size should be xxxx in UNIX
  - Kernal settings in Solaris
  - NFS mount DB2 $HOME directory
  - Change ufiles, xxx,xxxx
- Ensure adequate disk space is available on Windows platforms
- Mount the file system

Installation

- Edit the DB2NODES.CFG file and make entries for partitions, ports, servers and switch names

DB2NODES.CFG

- File Entries:

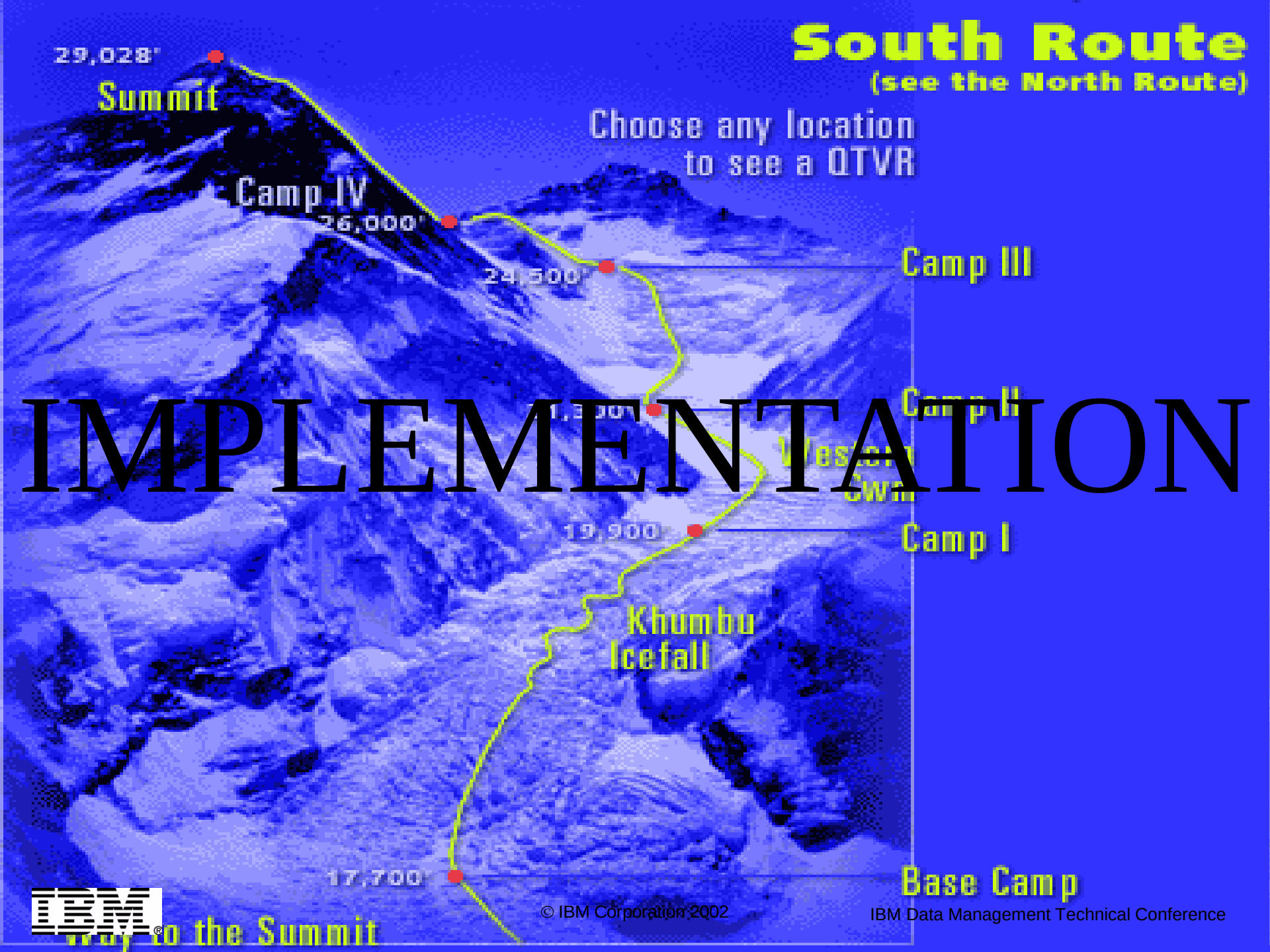
DB Partition Num	hostname	Logical Port	netname
1	host1	0	switch1
2	host1	1	switch1
3	host2	0	switch2
4	host2	1	switch2
5	host3	0	switch3

Attendee Notes

- The db2nodes.cfg file entries must be made before you can create a database.The following entries correspond to the previous chart:

Installation

- Install code on all servers in the configuration



IMPLEMENTATION

Implementation

- Catalog Partition
  - Created based on the contents of the DB2NODES Environmental Variable
    - DB2 Terminate
    - DB2SET DB2NODE=1
- Application Partition
  - Can have different categories of users connection to different partitions to balance out the workload
    - DB2SET CLIENT ATTACH-NODE=

Implementation

- Each Database Partition has its own:
  - Logs
  - Database Configuration File (DB CFG)
  - Buffer pools

Implementation

- Database Partition Group (formerly known as nodegroup)
  - Consists of database partitions

Implementation

- Create Table Wizard

Implementation

- V8.1 Global Snapshot Monitoring with tables

Implementation

- Instead of Triggers

- Informational Constraints

Implementation

- CALL statement now fully compiled statement
- INSPECT Command
- DB2TRACE Overhead reduction

Implementation

- New DB2SETUP Wizard (UNIX)

Implementation

- Redistribute Data Wizard
- Backup and Restore Wizard
- Configure Database Logging Wizard
- Add Partition Wizard
- Storage Management View
- Load Wizard
- Workload Performance Wizard

Implementation

- Data Warehouse Center and Warehouse Manager Enhancements
  - See What's New Guide for more Info

Implementation

- New Configuration Assistant (replaces CCA)

Implementation

- Memory Visualizer

Implementation

- Database Maintenance Mode
  - Quiesce

- Load Enhancements
- Multiple Service Level Install (UNIX)
- Flush Package Cache
- Improved Catalog/Authorization Caching for Partitioned Databases

Implementation

- Backup and Recovery Enhancements
- Health Monitor and Health Center
- Management by Exception Alert Capability
- Health Monitor
  - Server side agent that monitors the health of an instance
  - Alerts by email or pager
  - Preconfigured actions can be taken
  - Integrated into Control Center via Health Beacons
  - Tools catalog

Implementation

- Health indicator
  - Predefined threshold that the HM checks
- Health Center is used to configure the Health Monitor
- Health Monitor can be accessed via Health Center from the Web Browser or PDA

Implementation

- Health Monitor gathers information about the health of the system using new interfaces that do not impose a performance penalty
  - Does not turn on monitor switches
  - Enabled by default at instance creation

Implementation

- New Event Monitor Capability enables truly global event monitoring by inserting into the same table on each database partition.
- Improved DEADLOCK Event Monitoring
  - Now shows SQL involved in a deadlock and shows which locks are being held

Implementation

- Snapshots can now be taken via SQL through new functions
  - Example

Implementation

- Type-2 Indexes
  - Reduces next key locking to a minimum
  - Keys now marked deleted instead of physically removed from the index page
  - Required for online reorg and online table load
  - Required for MDC
  - Cannot mix type-1 and type-2 on a table
- Convert from type-1 to type-2 using the REORG INDEXES command

Implementation

- Autoconfigure Command
- Block-based buffer pools improve prefetching
- Page Cleaner I/O Improvements exploit asynchronous I/O facilities to improve performance (especially UNIX?)
- Resource sharing of JVM

Implementation

- User-Maintained Materialized Query Tables (MQTs)

Implementation

- Connection Concentrator

Implementation

- RENAME INDEX

Implementation

- Multidimension Data Clustering (MDC)

Implementation

- Online Reorg

Implementation

- In V8.1 Autloader functions now integrated into Load Utility

Implementation

- New exception monitoring via alerts from the new V8.1 Health Center

Implementation

- New V8.1 Declared Global Temporary Table (DGTT) enhancements
  - Can create indexes on DGTT using CREATE INDEX statement
  - Undo logging supports rollback of changes to DGTT
  - Can run RUNSTATS on table and indexes

Implementation

- Dynamic DB CFG parameters
  - Can now change over 50 parameters online
    - Can defer changes
    - Get DB CFG changes
      - Will show if changes deferred or not
  - Buffer pool changes take effect immediately

- Compression of Nulls and Defaults

- Insert through UNIONALL Views

Implementation

- Infinite Active Log
- Drop DMS Container
- Perform DMS container operations such that a rebalance does not occur

Implementation

- Development Center replaces Stored Procedure Builder

SUCCESS!

Thanks!

Implementation Fundamentals for The Latest DB2 UDB Partitioned Database

pgunning@breakthroughdb2.com

Anaheim, CA **Sept 9 - 13, 2002**