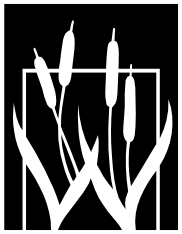


Whitemarsh
Information Systems Corporation

Database Management Systems: Understanding and Applying Database Technology
Chapters 6

System Control

Whitemarsh Information Systems Corporation
2008 Althea Lane
Bowie, Maryland 20716
Tele: 301-249-1142
Email: Whitemarsh@wiscorp.com
Web: www.wiscorp.com

Table of Contents

1.0 System Control

- System Control Introduction

- System Control Components
  - Audit Trails
  - Message Processing
  - Backup & Recovery
  - Reorganization
  - Concurrent Operations
  - Multiple Database Processing
  - Security & Privacy
  - Installation & Maintenance
  - Application Optimization

- System Control Summary

2.0 System Control Introduction

- Rationale for System Control
- Components of System Control
- Static/dynamic Differences
- Risk of Absence (What's better than no System Control?)

2.1 Rationale for System Control

- Basic Definition
- DBMS Implications (Static vs Dynamic)
- Why It's the Critical Success/failure Factor

Definition

- System Control: A Series of DBMS Vendor Supplied Capabilities, Processes, and Activities Necessary to Keep in Optimum condition:
  - The Database,
  - The DBMS and
  - The Application
- These Facilities Are Supplied Through
  - Natural Language Commands And/or
  - Specially Created Utilities, And/or
  - Specially Created Notes/course

2.2 System Control Components

- Audit Trails – The Capture of Database Update Information
- Message Processing – Exception Condition Processing
- Backup and Recovery – Copying & Restoring Databases
- Reorganization – Logical & Physical to Restore Optimum Organization
- Security and Privacy – Protecting Data from Theft
- Multiple Database Processing – Mechanisms to Access Several Physical Databases
- Concurrent Operations – The Specification of Conflicting Operations
- Application Optimization – The Discipline of Reducing Resource Consumption
- Installation and Maintenance – Procedures to Install, Validate, and Maintain DBMS Software

2.3 Static Vs. Dynamic

SYSTEM CONTROL FACILITY	DBMS TYPE	
	STATIC	DYNAMIC
Audit Trails	Poor to Good	Poor to Good
Message Processing	Poor to Good	Acceptable
Backup and Recovery	Good	Good
Reorganization	Acceptable	Acceptable to Good
Security and Privacy	Poor to Acceptable	Poor to Acceptable
Multi-Database Processing	Poor to Good	Good
Concurrent Operations	Acceptable to Good	Good
Application Optimization	Acceptable to Good	Poor to Acceptable
Installation and Maintenance	Poor to Acceptable	Acceptable to Good

Value of System Control

- **Audit Trails** Enable Knowing Database Activity, Critical for Optimization and Evolution
- **Message Processing** Provides Alerts for Training, Reduces End-user Frustrations.
- **Backup & Recovery** Enables Quick Restoration of Operational Database Environment upon Computer Or Software Failure.
- **Reorganization** Enables the Database to Evolve As Changes Arise, and Prevent "Freeze-ups" Due to Storage Structure Degradations.
- **Security and Privacy** Prevents Loss of Valuable Corporate Sensitive Data. Protection of Personnel Data Is the Law!

Value of System Control (cont,)

- **Multiple Database Processing** Is Critical as Different Databases Will Exist That Have to Be Integrated for Reporting, Updating, Etc.
- **Concurrent Operations** Guidelines Are Required to Effectively Schedule Programs/processes That Absolutely or Effectively Lockout Other Database Activities.
- **Application Optimization** Is Required To Reorient the Physical Database to Different Types of Processing.
- **Installation and Maintenance** Is Required to Manage DBMS Versions, Fix Bugs, Evolve the DBMS Software Through Different Releases.

Without High Quality System Control

All Logical Database Design

All Physical Database Design

All Quality Interrogation

Is Lost

3.0 Audit Trails

- Introduction
- Transaction Sources
- Capture Context
- Transaction Content
- Storage Media
- Reporting
- Static vs Dynamic
- Summary

3.1 Audit Trails Introduction

- Software to Capture Database Transactions
- These must Be Reportable

- Key Issues:
  - ♦ Transaction Sources
  - ♦ Transaction Content
  - ♦ Transaction Storage Media
  - ♦ Single User Environment
  - ♦ Multi User Environment

3.2 Transaction Sources

- Host Language Interface Programs
- Procedure Oriented Language Programs
- Query-update Language Transactions
- Report Writer Programs
- Different Capture Modes
  - Batch
  - Interactive
  - Exclusive or Shared

3.3 Transaction Content

- Date = Yyyymmddd (Julian Format)
- Time = Hh: Mm: Ss: 100
- Submitter Id
- Database Id
- DBMS Version Id
- Database Data Cycle Id
- Actual Command
- Table Name
- Before Image
- After Image
- Etc

3.4 Transaction Storage Media

- Tape -- Safe, but Consumes Tape Drive For Each DBMS Central Version
- Disk -- less Safe, but More Flexible For Allocation and Hardware Resources Consumed.

In Either Case, the "Write" Should Be Direct Not Buffered.

3.5 Audit Trail Reporting

- Standard Reports
  - User-id
  - Date
  - Database
  - Table
  - Run-unit-id
  - Language Type
  - Error Message Type

- Ad Hoc Reports
  - Procedure Oriented Language
  - Query Language
  - Report Writer

3.6 Static Vs. Dynamic

Audit Trail Aspect	DBMS Type	
	Static	**Dynamic**
Database Definition	Many interconnected tables	Single table per database
Database Sources	Well developed	Acceptable
Transaction Content	Poor	Poor
Transaction Storage Media	Ok	Ok
Reporting	Ok	Ok

3.7 Audit Trail Summary

- Critical Facility to Monitor Database Activity
- Especially Important for 'Young' Applications
- Critical to Determine Optimum Database Usage
- Enables Auditing, Eg. CPA Type
- Supports Legal Considerations (HR, Sarbanes-Oxley?)
- What to Do: Examine the DBMS Critically to Determine If Facility Is Available Include Within Run-units Otherwise

4.0 Message Processing

- Introduction
- Message Types
- Severity Levels
- Messages Database
- Summary

4.1 Message Processing Introduction

- Definition: Messages Issued to the Run-unit (User)
- Issues Covered
  - Typical Message Types and Sources
  - Effects of Messages
  - Message Effects
  - On-line Help
  - Static Vs. Dynamic DBMS Comparison

4.2 Typical Message Types

- Type = 1: DBMS Syntax Errors
- **Example:**

  Modify Employee-name Ware

  << Syntax Error >>

- Source
  - ♦ DDL Language
  - ♦ Interrogation Languages
  - ♦ System Control Languages

Typical Message Type (Cont.)

Type = 2 Application Generated

- Example:

  Print Employee-mame Where...

  << No Valid Field Employee-mame >>

- Sources:

  - DDL Languages
  - Duplicate Field Name
  - Duplicate Schema/subschema
  - Interrogation Languages
  - Illogical Request
  - Untrue Selection Clause
  - System Control Language
  - Illegal Password

Typical Message Types (Cont.)

- Type = 3: Data Generated
- Example:

  Modify Employee-name Eq Ted C0dd Where

  Employee-id Eq 485927143

  << Illegal Employee-name Value >>

- Sources:
  - ♦ Data Loading
  - ♦ Rows out of Sequence
  - ♦ Illegal Data Value
  - ♦ Interrogation
  - ♦ Insert rows or column values
  - ♦ Modify rows or column values

Typical Message Type (Cont.)

- Type = 4: Database Integrity

- Example:

  Open Sales (Mike)

  << Sales Database Damaged >>

- Sources:

  - Damaged Pointers
  - Unreadable Dictionary,
  - Index,
  - Relationships,
  - Or Data
  - Missing Storage Structure Components

Typical Message Type (Cont.)

- Type = 5: DBMS Integrity

- Example:

  Start DBMS
  <<Unrecoverable Error 123>>

- Sources:

  - Missing Subroutines
  - Time Bomb Expired
  - Trapped Logic Error

4.3 Severity Levels

- Severity = 1: Information Only

- Example:

  Delete Employee Where Ssn Eq '184-86-7787'

  <<Employee Deleted>>

Severity Levels Cont.

- Severity = 2: Correctable

- Example:

  Print <Employee Row> If Ssn Eq '34-134-8618'

  <<No Rows Found>>

Severity Levels Cont.

- Severity = 3: Run Unit Fatal

- Example:

  Modify Salary = 100000 If Employee-name Eq 'Me'

  <<Update Authority Not Allowed>>

  <<Run-unit Terminated>>

- Sources

  Request for Inactive Database
  Request for an Already Exclusively
  Controlled Function
  Security Violation

Severity Levels (Cont.)

Severity = 4: Database Fatal

Example:

- A Read to Index That Has Bad Track
- <<Database Damage>>
- <<Run-unit Terminated>>
- <<No Database Access Allowed>>
- Blue Screen of Death

- Sources:
  - I/O Errors
  - Memory Failure
  - DBMS Bug

MESSAGE TYPE AND SOURCE		MESSAGE SEVERITY				
		1 INFO	2 CORRECTABLE	3 RU FATAL	4 DB FATAL	5 DBMS FATAL
1	**Syntax Error**	N/A	Local CV	Local	N/A	N/A
2	**Application**	N/A	Local CV	Local N/A	N/A	N/A
3	**Data**	N/A	Local CV	Local	N/A	N/A
4	**DB Integrity**	N/A	N/A	N/A	Local CV	N/A
5	**DBMS integrity**	N/A	N/A	Local N/A	Local CV	Local CV

Legend:

- local = local mode (single user)
- CV = multiple users from batch or on-line
- N/A = not applicable

4.4 Messages Database

- A Table in the Data Dictionary System

- Access Keys
  - Message Number
  - Short Name
  - Long Name

- Message Entry Content
  - Keys above
  - Message and Explanation
  - Suggested Remedies

- Audit Trail Entry for Help Transaction
  - Message Types 2, 3, 4, 5
  - Message Severity 3, 4, 5

- Usages
  - Fundamental Component of Data Dictionary System
  - Transactions Stored on Audit Trail
  - Audit Trail Report Identifies Training Needs.

Static Vs. Dynamic DBMS Type Comparisons

- No Need for a Difference

- While All DBMS Provide Messages, Few Provide a Facility for On-line Help, or Central Logging and Review.

- Case in Point

  Helicopter is lost in fog in Seattle. Pilot sees a tall building. Hovers. Yells out the window, "Where am I?". The office occupant yells back, "23rd Floor, South-West Corner, across from Cubical 17c."

  At that point, the pilot says, "OK!" He turns redirects his helicopter and makes a safe landing.

  How did he know where he was? Simple... The Microsoft building. Absolutely accurate, precise, but completely useless information.

4.5 Messages Summary

- Five Classes

  1. DBMS Syntax Errors
  2. Application Generated
  3. Data Generated
  4. Database Integrity
  5. DBMS Integrity

- Five Severity Levels

  1. Information
  2. Correctable
  3. Run Unit Fatal
  4. Database Fatal
  5. DBMS Fatal

- Shut-downs Due to Errors Should Be Few
- Messages Database a Critical Component
- Indicates a Quality DBMS.

5.0 Backup & Recovery

- Introduction
- Classes of Failure
- Recovery Vehicles
- Recovery Modes
- Lockout
- Types of Recovery
- Checkpoint Recovery
- Cost Effective Choices
- Database vs Teleprocessing
- Static vs Dynamic
- Summary

5.1 Backup & Recovery Introduction

Two Parts	
Backup	The Process of Creating a Database Copy Onto Another Media.
Recovery	Using Backup or Current Database With Update Log to Recover the Database, or Use of Automatic Transaction To Rollback Updates

5.2 Classes of System Failure

- Run Unit Exception
- Operating System or DBMS Failure
- Failure to Execute a Machine Instruction
- Part of a Database Is Unreadable
- Subtle Corruption Detected by Later Run Units
- Operational or Application Errors Detected by Users

5.3 Recovery Vehicle (Update Log)

- Sometimes Called Journal File
  - ♦ Contains Reapplicable Update Transactions
  - ♦ Used in Local/central Version Modes
  - ♦ Format: "Before-images" And/or "After-images"
- Or Alternatively,
  - ♦ A File of Specially Encoded Update Commands Stripped of All but That Necessary to Execute an Update
- Example:

  File <X>, Page <Y>, Byte <Z>, <Value>

Recovery Vehicles (Cont.)

- Before-image Page Log

  A Tape or Disk File of Rows, (Copies of Database Pages Before Update.

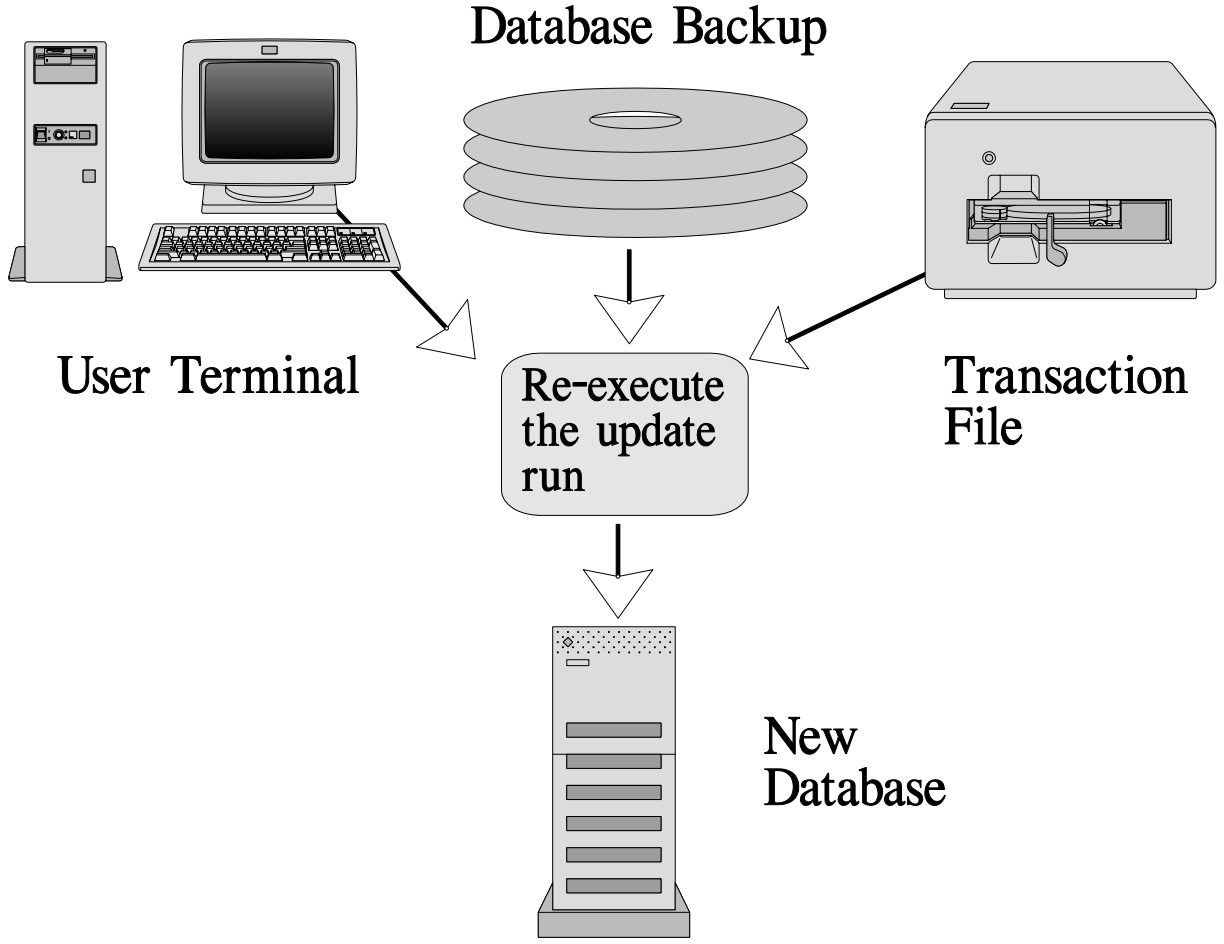
- After-image Page Log

  A Tape or Disk File of Rows, (Copies of Database Pages after Update.

5.4 Recovery Modes

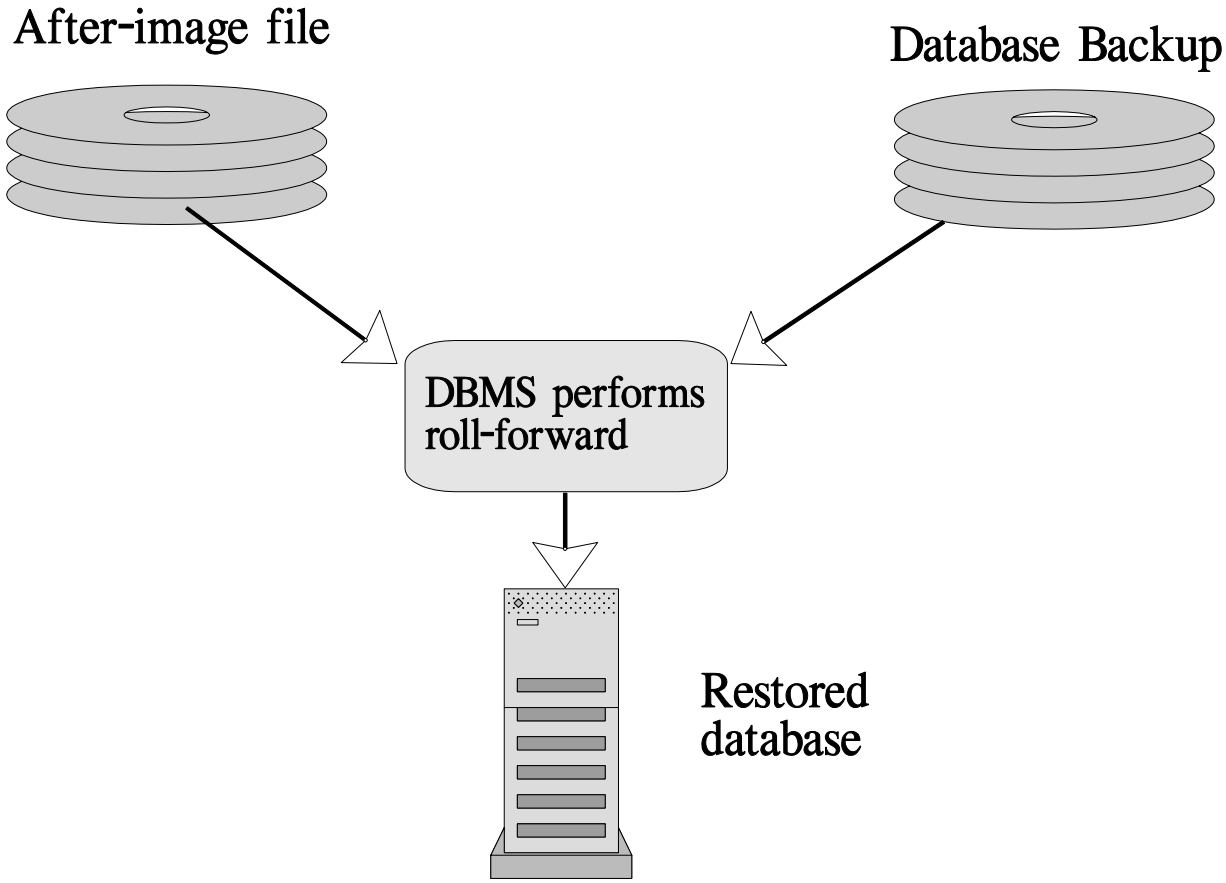
Single-user Recovery

Rerun the Job



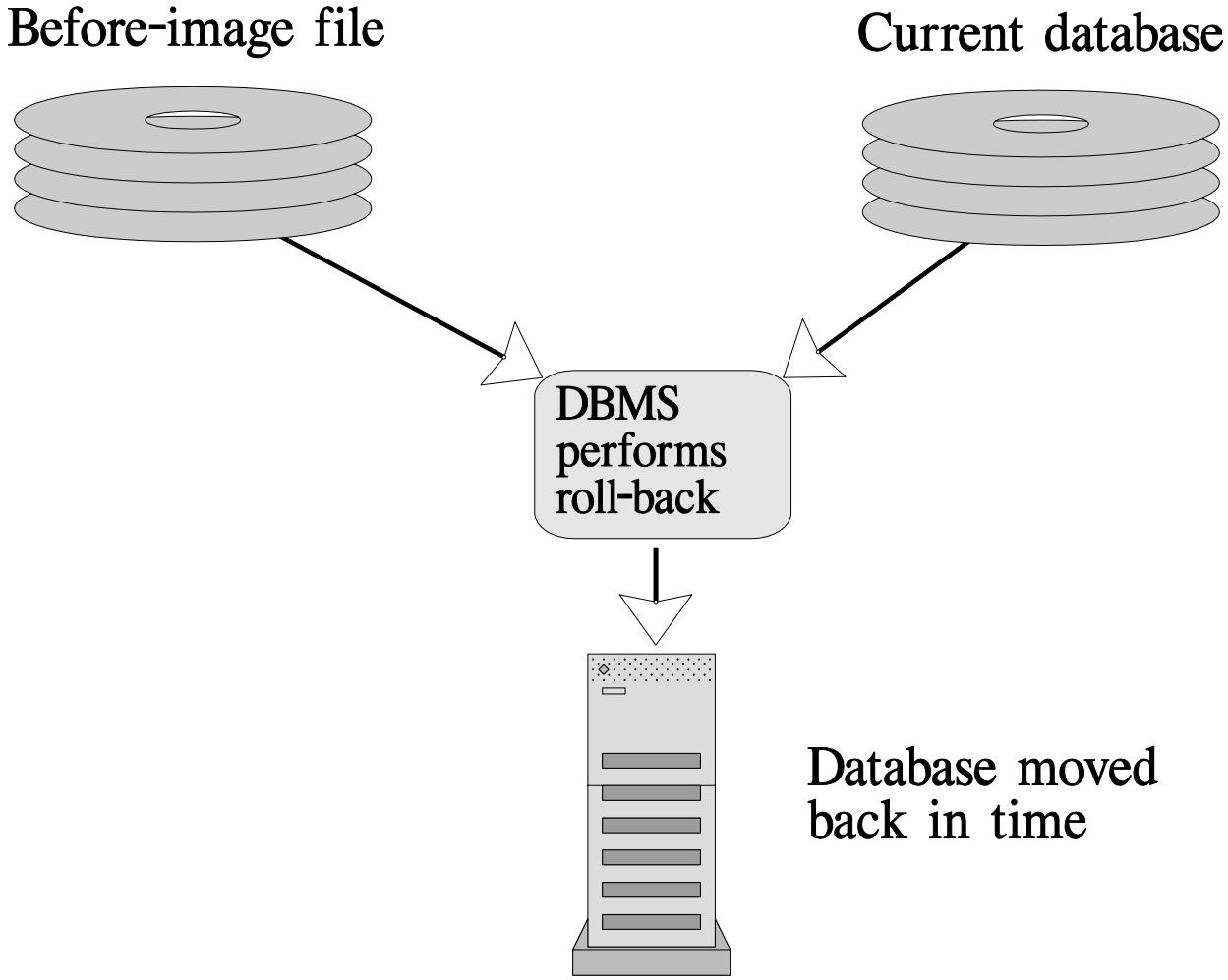
Single-user Recovery

Roll-forward (After Image Recovery)



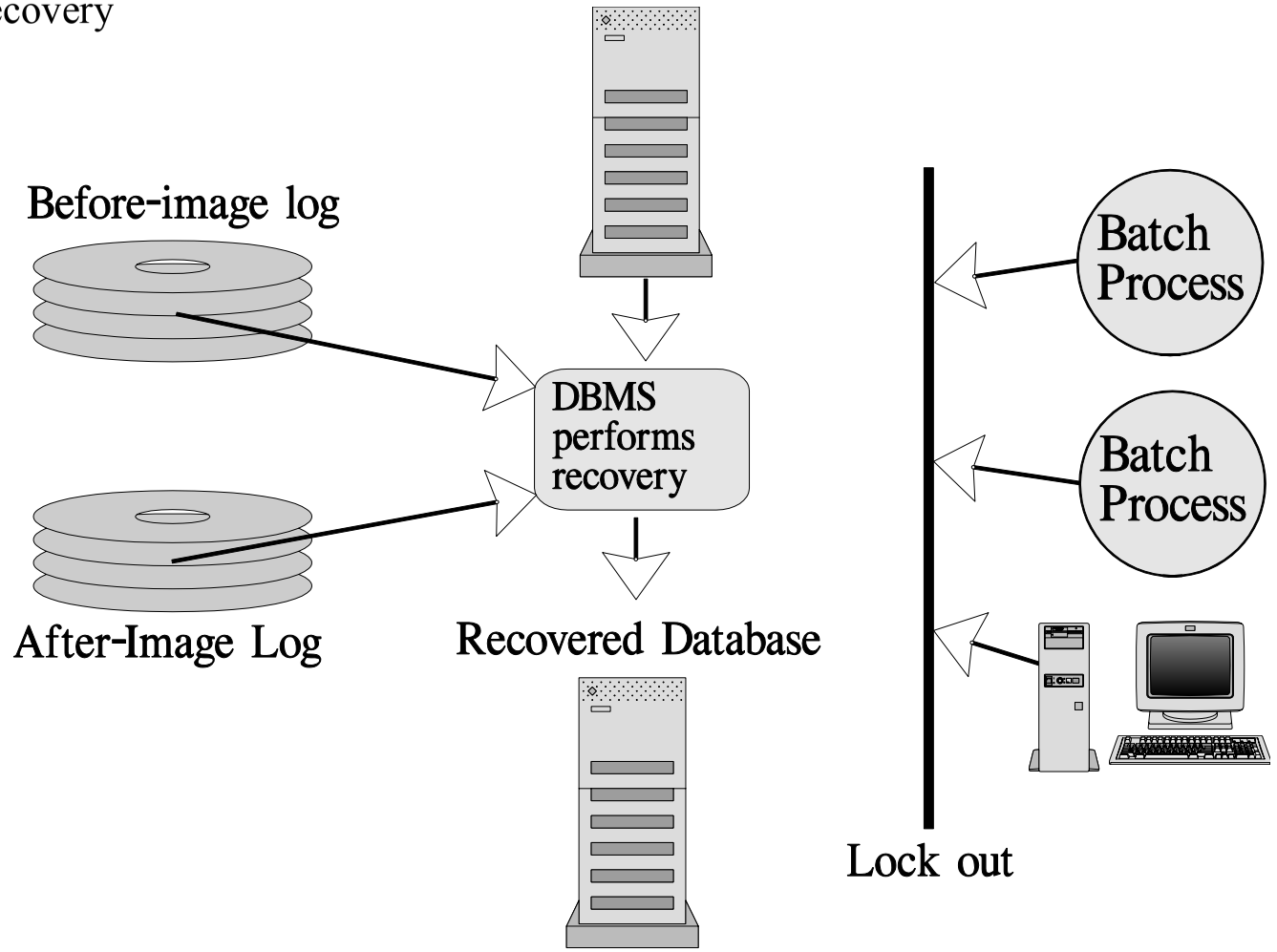
Single-user Recovery (Cont.)

Before Image Recovery



Recovery Methods (Cont.)

Multi-user Recovery



5.5 Lockout During Recovery

- A DBMS Process Which Precludes Conflicting Operations.
- General Process
- Recovery Instigates Lockout
- Static DBMS' Usually Lock the Entire Database / O/s File
- Dynamic DBMS Locks Only a Single Table (Usually an O/s File). Associated Indexes Are Also Locked
- Critical Issue: Coordination of the Recovery of a Multiple Database Environment

5.6 Types of Recovery

- Transaction Rollback
- Job Rerun
- Roll Forward
- Rollback
- Rollback & Roll Forward

Basic Definition of a Transaction

A DBMS Operation That Removes an Update Transaction from the Database

- Two Alternatives:
  - Explicit Start and Stop
    - <Start Frame>
    - Run-unit Updates
    - <End Frame>
  - Implicit Start and Stop
    - <Start DBMS> (Implicit Open)
    - Update Transactions
    - <Finish> (Commits Transactions, and And Opens next Frame)
- If Either <End Frame> or <Finish> Is Preceded by a <Rollback> Then the Transaction Is Canceled.

Transaction Rollback (Cont.)

Rollback Alternatives

- Automatic
  - Run-unit Is Interrupted During a Frame
  - DBMS Restarts and Finds Uncompleted Frame
  - DBMS "Backs-out" All Run-unit Transactions
  - Using the Transaction Roll-back Log

- User-interrupted
  - User Interrupts a Frame with <Cancel>/<Break>
  - DBMS Automatically Backs out Frame

- User Instigated
  - User Finishes Submitting Update Commands
  - Then Determines There Is a Mistake
  - Then the User Submits <Rollback>

Caution: Transaction Rollback Facilities Are Usually Only 1-transaction Long!

Business Transaction: Add Customer consisting of many run-unit transactions.

```
Start Transaction

        Do While Customer Exists
            Insert Customer. If Failure then rollback

                Do While Contract Exists
                    Insert Contract. If Failure then rollback

                        Do While Order Exists
                            Insert Order. If failure then rollback.

                                Do While Order-Item exits
                                    Insert Order-item. If failure then rollback

                                End-Order Item

                        End-Order

                End-Contract

        End-Customer

End-Transaction
```

Job Rerun

- Back-up Database Before Job (That Crashed) Starts
- Restore Database from a Back-up Copy
- Re-commence Processing the Original Job

- Advantages
  - ♦ Does Not Require Logging of Before or after Images
  - ♦ Input May Be Edited Before Rerunning

- Disadvantages
  - ♦ Database Is Unavailable for Some Time
  - ♦ Run Stream must Be Clearly Identifiable

Roll-forward

- Restore Database from a Backup Copy
- Align Log of after Images to a System Recovery Point Corresponding to the Restored State of the Database
- Apply after Images until a System Recovery Point Is Reached
- Restart Run-units from the Recovery Point

- Advantages
  - ♦ Does Not Require Logging of Before Images
  - ♦ Several Updates May Occur in Main Storage
  - ♦ Before an Image Is Logged

- Disadvantages
  - ♦ Database Is Unavailable for Some Time System
  - ♦ Quiet Points Require All Run-units To Be Halted

Roll-back

- Apply Before Images Back to One of
  - Start of Failed Command
  - Start of Success Unit
  - Start of Run-unit
  - System Recovery Point
- Realign Any Input Files to a Point Corresponding to the Roll-back
- Restart Processing
- **Advantages**
  - Can Be Most Readily Automated
  - Quick for Short Success Units
  - May Be Selective by Run-unit
  - Only Run-units Active at the Time of Failure Are Affected
- Disadvantages
  - Requires Before Images Logged Separately for Each Run-unit
  - Identifiable Success Unit Boundaries
  - Are Needed as Recovery Points

Roll-forward with Roll-back

- Restore Database from a Backup Copy
- Align Log of after Images to a System Recovery Point Corresponding to the Restored State of the Database
- Apply after Images until End of the Log
- To Achieve a Consistent State of the Database Apply Before Images Back to The Last System Recovery Point
- Restart Processing

- Advantages

  Can Be Used Where Knowledge of Status at the Point of Failure Has Been Lost

- Disadvantages
  - ♦ Requires Logging of Both Before and after Images
  - ♦ Database Is Unavailable for Some Time

5.7 Checkpoint and Re-start

- Checkpoint: A Special Transaction Placed on the Journal File That Indicates That Once a Database Is Restored to That Transaction, the Database Is Consistent.

- Opportunities for Checkpoints
  - Start of Job (Bind)
  - End of Job
  - DML Commit (Many Within Job)
  - DML Abort (Many Within Job from Rollbacks)
  - DML Ready Area (Codasyl)
  - Journal File Initialization
  - During DBMS Restart or Recovery
  - Job Rollbacks Go to Before <Start of Job>

5.8 Cost Effective Choices

- Level of Protection
  - ♦ Cost of Unavailability
    +
  - ♦ Cost of Recovery Processing

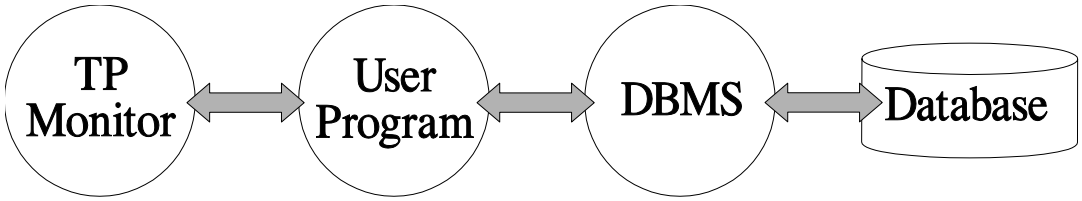
    Vs
  - ♦ Cost of "Insurance Premiums" in Logging and Processing Constraints

Processing	Backup
Batch Only	Tape Logging of Transactions
On-line retrieval, non critical	Tape logging for roll-forward
On-line Update, non-critical	Tape logging for roll-forward
Critical On-line	Disk logging for roll-back, dual files, possible other hardware redundancy.

CAUSE OF FAILURE	RECOVERY METHOD		
	IMMEDIATE	**RAPID**	**SLOW**
Run Unit Exception	Not impossible except for transient failures (full redundancy)	Run-unit rollback	Run-unit roll-back or roll-forward and restart
O/S or DBMS Failure	Not possible except for transient failures (full redundancy)	System roll-back; restart from checkpoints	System roll-back or roll-forward and restart or rerun
Failure to accept CPU instruction	Hardware Redundancy	SAME AS ABOVE	SAME AS ABOVE
Database Unreadable	Dual files	Dual Files	SAME AS ABOVE
Corruption detected by DBA	IMPOSSIBLE		Log analysis:DBA takes manual action
Corruption detected by user			Analysis of run log and results correction of applications . Rurun

5.9 Recovery–Database vs Teleprocessing (Tp)

Action	Responsible Party/Software
User to TP Monitor	User
TP Monitor Response to User	TP Monitor
Activation of Processing Module	TP Monitor
Processing Module to DBMS	TP Monitor
DBMS to Database	Database
DBMS Response to User Program	Database
User Program to TP Monitor	TP Monitor
TP Monitor to User	TP Monitor



Critical to Entire Processing Is a Completely Integrated Log That Can Be Used for Coordinated Recovery.

5.10 Recovery: Static Vs. Dynamic DBMS

ISSUE	STATIC	DYNAMIC
Inherent DB Structure	Many Tables	Few Tables
Number of Users Per DB	Many	Few
Probability of Damage	Higher	Lower
Time For Recovery	Longer	Shorter
Recover Completeness	Greater	Lesser

5.11 Backup and Recovery Summary

- Complex Critical Issue a Lost Database Will Kill Database

Recovery	
Static	Dynamic
Longer Slower More Expensive Easier for Complex Databases	Quicker Faster Cheaper More Difficult for Complex Databases

- Make Sure TP and DB Are Coordinated So Nothing Is Lost.

6.0 Reorganization

- Introduction
- Logical
- Physical
- Static vs Dynamic
- Summary

6.1 Reorganization Introduction

- DBA Procedures and DBMS Utilities to Restructure Database Storage Structure Components
- Types of Reorganization
- Logical: Addition, Deletion, or Modification of
  - Columns and Subclauses, or
  - Tables and Subclauses, or
  - Relationship Types and Subclauses
- Physical: Restructure of Physical Storage Structure Components.
  - Dictionary (See Logical Above)
  - Indexes
  - Relationships
  - Data

6.2 Logical Organization

- Addition, Deletion, or Modification of
  - Data Columns and Subclauses
  - Tables and Subclauses
  - Relationships and Subclauses

- General Process:
  - Special DBMS Software Module Is Accessed.
  - Changes Are Received,
  - Syntax Checks Are Made
  - If No Errors, Then
    - DBMS Makes Changes, Often Requiring
    - Some Level of Physical Database Reorganization.

Logical Reorganization (Cont.)

- Column Changes:
  - Add/delete Columns
  - Modify Column Characteristics
  - Change Data Column Length
  - Change Data Type
  - Change Editing and Validation Clauses

- Table Changes
  - Adds or Drops
  - Modification of Table Clauses
  - Physical Access Clauses
  - Editing and Validation Clauses
  - Stored Procedure References

Logical Reorganization (Cont.)

Relationship Changes	
DBMS Type	**Relationship Change Action**
Static (ANSI NDL, Network or Hierarchy)	Add, delete, or modify sets or subclauses
Dynanmic (ANSI SQL, ILF or Relational)	Column adds, deletes, or “value” modifications

6.3 Physical Re-Organization

- Restructuring Storage Structure Components of the Physical Database.
  - Dictionary
  - Indexes
  - Relationships
  - Data

Physical Reorganization (Cont.)

- Dictionary Reorganization
  - See Logical Reorganization
- Index Reorganization
  - Reallocate Levels to Improve Performance
  - Change Blocking Factors
  - Allocate Padding to Index Pages For Extra Values Without Reorganization
  - Resequence Index Pages to Achieve Optimum Logical Order

Physical Reorganization (Cont.)

- Relationship Reorganization
  - Not for Dynamic DBMS
  - Static DBMS
    - Embedded Pointer DBMS
      - Resort Rows into Order of Relationship
    - Segregated Pointer Static DBMS
      - Sort Rows into Relationship Order
      - Sort Segregated Pointers into Relationship Order
- Data Reorganization
  - Reclaim Deleted Row Space
  - Expand Space in Physical Rows For Precise Loading
  - Change Blocking Factors
  - Sorting Rows for Specific Processing Improvements

6.4 Static Vs. Dynamic DBMS Type Comparisons

CHANGE	DBMS TYPE	
	STATIC	DYNAMIC
Column	Reload at least an area	Reload at least a table
Row	Reload at least an area	Reload at least a table
Relationship	Add, delete, or modify relationship by writing a program to row relationship into rows	Change an column's value

6.5 Reorganization Summary

- Necessary Aspect of All Database Projects
- Vehicle for Database Evolution
- Permits Partial Optimization with Reloads

7.0 Concurrent Operations

- Introduction and Basic Definitions
- Locations of Conflict
- Transaction Management
- Deadly Embrace
- Database Lockout
- Static and Dynamic Comparison
- Various Computing Environments
- Summary

7.1 Current Operations Introduction

- Basic Definition

  More than One Program Attempting -- at the Some Point in Time – to Update the Same Units of Data.

- Serializability: the Ability to Have a Set of Bounded Updates Operate in a Multiple User Environment, Producing the Consistent Results Each Time They Are Run.

7.2 Locations of Concurrent Operations Conflict

- Updates to the Same Storage Structure Component
- Updates by Multiple Users under a Central Version
- Updates by Multiple Users under Different Central Versions to the Same Database

Single User Processing

- 1 User per DBMS Copy & per Database
  - ♦ No Possible Conflicts
  - ♦ Long Waits Definite Between Users {Good Only for Sequenced Batch Jobs}
- Multi-user, Single Thread, Multiple Users per DBMS Copy
  - ♦ DBMS Processes Only One Command to Completion
  - ♦ Regardless of User Long Waits Possible

Multi-user, Multi-thread, Multiple Users per DBMS Copy.

- DBMS Processes Each Command Independently
  - ♦ Long Commands Take a Long Time
  - ♦ Short Commands Take a Short Time
- Single Threading for Some Update Operations.

Other Areas of Concurrent Operation Slow-downs

- Some DBMS Only Allow One Operating Schema per Central Version
- Some DBMS Storage Structure Have Only One File per Table
- Some DBMS Storage Structure Have Only One File for All Data

7.3 Transaction Management

A database transaction is a unit of work within a DBMS environment that may affect multiple rows across multiple tables. It is expected that all transactions are either terminated and thus rolled back to their "before" state, or are completed leaving the database in a consistent state.

7.3.1 ACID Properties

A very well behaved transaction exhibits these properties...	
Atomic	all or nothing
Consistent	database must be consistent before and after a transaction
Isolated	no unwanted interference from other users
Durable	database changes are permanent after the transaction completes

7.3.2 Concurrent Operations Transaction Isolation Levels

First the "**Phenomena**"

Isolation Level Phenomena	
Phenomena Type	**Phantom Description**
P1 (Dirty Read)	SQL-transaction T1 modifies a row. SQL-transaction T2 then reads that row before T1 performs a COMMIT. If T1 then performs a ROLLBACK, T2 will have read a row that was never committed and that may thus be considered to have never existed.
P2 (Non-Repeatable Read)	SQL-transaction T1 reads a row. SQL-transaction T2 then modifies or deletes that row and performs a COMMIT. If T1 then attempts to reread the row, it may receive the modified value or discover that the row has been deleted.
P3 (Phantom)	SQL-transaction T1 reads the set of rows N that satisfy some <search condition>. SQLtransaction T2 then executes SQL-statements that generate one or more rows that satisfy the <search condition> used by SQL-transaction T1. If SQL-transaction T1 then repeats the initial read with the same <search condition>, it obtains a different collection of rows.

Second, the Transaction Isolation Levels

SQL-transaction isolation levels and the three phenomena			
Level	**P1**	**P2**	**P3**
READ UNCOMMITTED	Possible	Possible	Possible
READ COMMITTED	Not-Possible	Possible	Possible
REPEATABLE READ	Not-Possible	Not-Possible	Possible
SERIALIZABLE	Not-Possible	Not-Possible	Not-Possible

Third, the SQL Syntax for Establishing the Isolation Level for a Transaction

<set transaction statement> ::= SET [LOCAL] TRANSACTION <transaction characteristics>

<transaction characteristics> ::= [<transaction mode> [{ <comma> <transaction mode> }...]]

<transaction mode> ::= <isolation level>
| <transaction access mode>
| <diagnostics size>

<transaction access mode> ::= READ ONLY
| READ WRITE

<isolation level> ::= ISOLATION LEVEL <level of isolation>

<level of isolation> ::= READ UNCOMMITTED
| READ COMMITTED
| REPEATABLE READ
| SERIALIZABLE

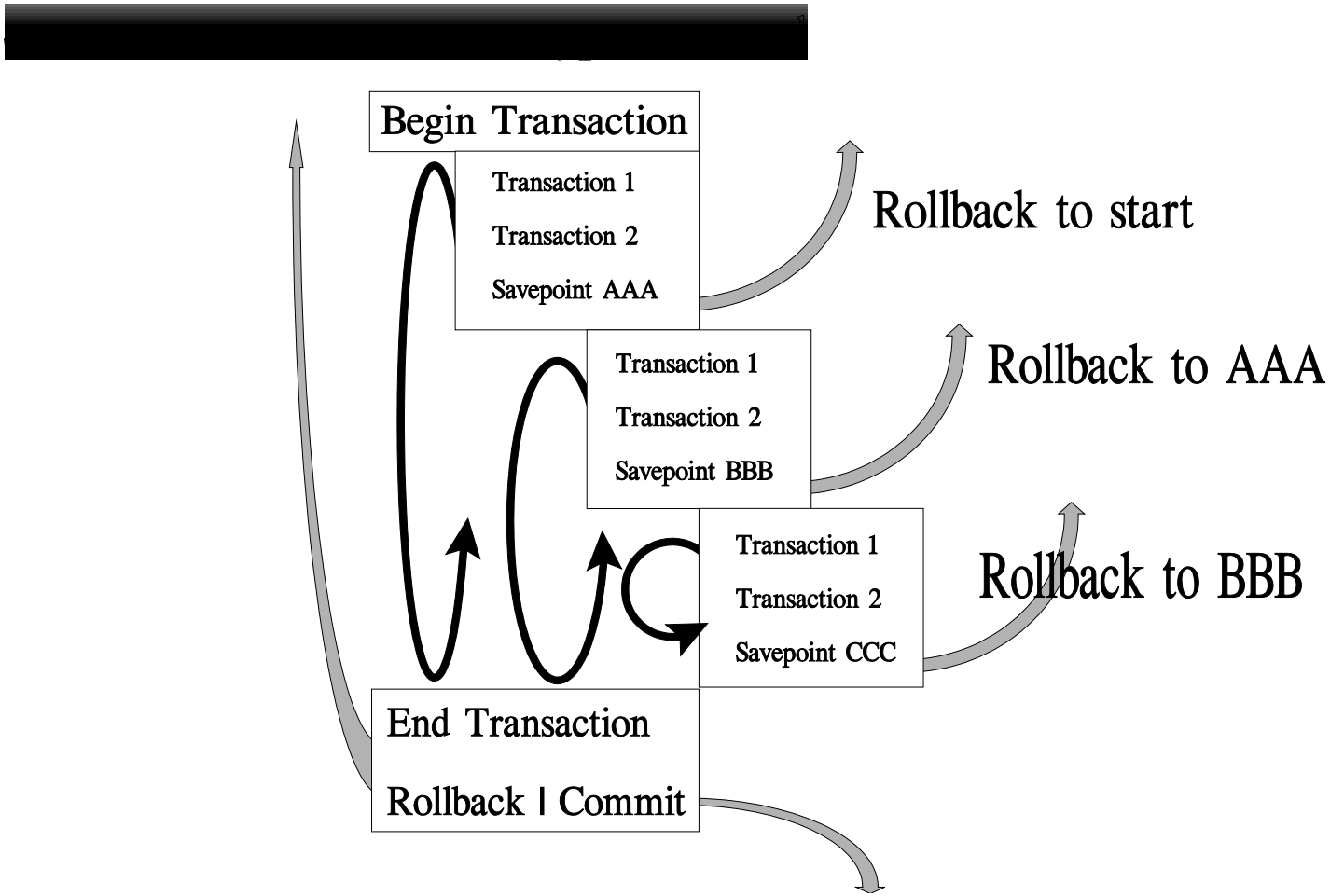
7.3.3 Set Transaction Statement "Rules"

- Execution of a <set transaction statement> is prohibited after the start of an SQL-transaction and before its termination.

- Execution of a <set transaction statement> before the start of an SQL-transaction sets the access mode, isolation level, and condition area limit for the single SQL-transaction that is started after the execution of that <set transaction statement>.

- If multiple <set transaction statement>s are executed before the start of an SQL-transaction, the last such statement is the one whose settings are effective for that SQL-transaction; their actions are not cumulative.

7.3.4 Transaction Rollback with SQL Savepoints



7.3.5 Concurrent Operations Deadly Embrace

- A State in Which Two or More Commands Mutually Await the Other's Locked Resource.
- Deadly Embrace Only Disables Poorly Designed DBMS.
- Example:
  - Run Unit 1 has read lock on table A AND needs Exclusive lock on Table B
  - Run Unit 2 has read lock on table B AND needs Exclusive update lock on Table A
- Deadlock Resolution: One Run-unit has to be terminated and then restarted
- Deadlock Prevention: Don’t start Run unit 2 until it has all the locks it needs

7.4 Concurrent Operations Database Lockouts

- Logical Database
  - ♦ Schema Modifications
  - ♦ Subschema/View Modifications
- Physical Database
  - ♦ Massive Data Loads
- Interrogation
  - ♦ User-instigated DBMS Locks
  - ♦ Run-unit System Control

Database Lockout (Cont.)

- System Control
  - Audit Trail Instigation, Suspension, or Modification
  - Messages
    - Severity 4 - Database Fatal
    - Severity 5 - DBMS Fatal
  - Backups and Recoveries
  - Logical and Physical Reorganization
  - Security & Privacy's Instigation and Maintenance

7.5 Concurrent Operations Static Vs. Dynamic

DBMS Type	General Scope and Effect
Static	Conflicts Affect More Users As Physical Database Encompasses Many Users
Dynamic	Conflicts Affect a Few as Physical Database Is Very Narrow in Scope.

7.6 Concurrent Operations Various Computing Environments

Levels of Sophistication

1. Single user, single database DBMS
2. Multiple user, single database, single thread DBMS
3. Multiple user, multiple database, single thread DBMS
4. Multiple user, single database, multiple thread DBMS
5. Multiple user, multiple database, multiple thread DBMS

DBMS Concurrent Operations Capabilities						
Increasing Level of Sophistication	**USER**		**DATABASE**		**THREADS**	
	Single	**Multiple**	**Single**	**Multiple**	**Single**	**Multiple**
1	YES		YES		N/A	N/A
2		YES	YES		YES	
3		YES		YES	YES	
4		YES	YES			YES
5		YES		YES		YES

Various Computing Environments (cont.)

Typical Platforms for Concurrent Operations Configurations						
Platforms	**USER**		**DATABASE**		**THREADS**	
	Single	**Multiple**	**Single**	**Multiple**	**Single**	**Multiple**
PC's	YES		YES			
PC Networks		YES	YES		YES	
Mini Networks		YES	YES		YES	
Mainframes		YES	YES		YES	
Mini computers and Mainframes		YES		YES	YES	
		YES		YES		YES

7.7 Concurrent Operation Summary

- Know the Areas of Conflict
- Verify the Lockouts or Slowdowns
- Know the sophistication of your DBMS
- Set up Tests to Stop/slow Processing via Nonconcurrent Operations
- Establish Procedures to Notify Users in Times of Emergency Lockouts

8.0 Multiple Database Processing

- Introduction
- Types of Multiple Databases
- Critical Issues
- Interrogation Facilities
- System Control Effects
- Static vs Dynamic
- Summary

8.1 Multiple Database Processing Introduction

Basic Definition:

Single User or Run-unit Requires Access To Multiple Databases (Schemas) Operating under One or More Central Versions.

8.2 Types of Multiple Databases

- Database Editions

  Time Based -- 1987, 1986, 1985

- Volume Based

  - Customers a - H
  - Customers I - P
  - Customers Q - Z

- Distributed Databases

  - Different Corporate Organizations
  - Same/different Computers

Types of Multiple Databases (Cont.)

- Different Applications
  - ♦ Personnel
  - ♦ Sales and Marketing
  - ♦ Manufacturing
  - ♦ MIS Projections and Modeling
- Multiple Database Levels
  - ♦ Executive Projections and MIS
  - ♦ Middle Management Control and Administration
  - ♦ Operational Transactions
- Database "Damage Control"
  - ♦ Multiple Databases of Same Schema
  - ♦ Divisions of Data for Different "Orders"
  - ♦ If One Division Is Down, Others Remain Up.

8.3 Multiple Database Processing Critical Issues

- Schema DDL Correspondence
  - ♦ Editions Processing Often Require Duplicate DDL down to Data Column Characteristics
- DBMS Versions
  - ♦ A 1984 Database Might Not Run under A 1987 DBMS Because of Storage Structure Redesigns
- Required Update Coordination
  - ♦ Updates to Multiple Databases Requires Careful Post-update Audits To Insure No Partial Updates.
  - ♦ Requires Central Version Log File for Coordinated Backout and Recovery

8.4 Multiple Database Processing Interrogation Facilities

- HLI or POL
  - DBMS and its Data Is Already a Foreign Intruder, Why Not Another?
  - Most DBMS Allow Multiple Active Databases
- Report Writer
  - Most Common in Static DBMS That Allow Complex Databases. Therefore, Typically Only One Database

Multiple Database Processing Interrogation Facilities (Cont)

- Query-update Languages
  - Language Is Typically Single Sentence Oriented
    - Single Print (Format) List
    - Single Sort List
    - Single Select List
  - Consequently, Use Is Usually Restricted To Single Database

8.5 Multiple Database Processing's System Control Facilities

- Backup and Recovery
- Most Backup and Recovery Is Restricted to Single Central Version Environment
- If Multiple Database Within >1 Central Version Then B & R Is Complex, but Not Impossible
- If There Are Multiple Central Versions Then Coordinated B & R Is Very Difficult
- Must Schedule Updates Around Transaction Rollback. Helpful Only with Multiple Databases under One Central Version
  - <Start Tran> ...< End Tran> Can Produce Multiple Updates
  - No Help for Multiple Central Versions

8.6 Multiple Database Processing Static vs Dynamic

- Real Issue Is: Central Version & Multiple Databases
- If Central Version and One Database Then No Problem
- If Central Version and Multiple Databases, Then Is the Journal File at the Level of the Database or the Central Version?
  - ♦ If at the Database level Then There's a Problem !
  - ♦ If at the Central Version level Then No Problems !

8.7 Multiple Database Summary

- When Needed: Always

- Why Needed: Method of Integrating

  - Database Editions (Time or Volume Based)
  - Distributed Databases (Different Sites/computers)
  - Different Applications (Personnel/engineering)
  - Multiple Database Levels (Operations/mis)
  - Most Capability Through HLI/POL

- Critical Issues

  - Schema Mapping
  - Evolved DBMS Storage Structure thru Time
  - Update Coordination
  - Backup and Recovery
  - Transaction Rollback

9.0 Security and Privacy

- Introduction
- Rationale
- Storage Protection
- Security Implementation
- Violation Notification
- Typical Implementations
- Summary

9.1 Security and Privacy Introduction

- DBMS Vendor-supplied Software and
- User-created Procedures to
- Ensure Data Is Not Viewed or Changed

9.2 Rationale

- Update Control
  - ♦ Modifying Existing Data-items
  - ♦ Adding New Data Items to Existing Rows
  - ♦ Inserting New Rows in a File
  - ♦ Appending New Rows to the End of the File
  - ♦ Amending Existing Relationships Between Rows

- Retrieval Control
  - ♦ Outputting in Detail
  - ♦ Use for Statistical Purposes or Summarization

- Selection Control For Updating, For Retrieval

- Administration Control

- Backup/recovery

- Assignment of Passwords and Authorities

9.3 Areas of Needed Protection

- Database's Dictionary or Dd/ds
  - Reading or Changing Schemas
  - Reading or Changing Subschemas
  - Changing Edit and Validation Tables/procedures
  - Changing Rules for Row Memberships or Sorting
- Indexes
  - Unique Values
  - Counts of Multiple Occurrences
  - Reading or Changing Address Pointers to Data
- Relationships (Static Segregated)
  - A Count of Relationships = Count of Rows
  - Quantity of Rows Within Relationships
  - Reading or Changing Subsequent Addresses

9.4 Alternatives for Security Implementation

- DBMS Instigated and Controlled
  - ♦ Most Common
  - ♦ Able to Be Bypassed by O/S
- Encryption
  - ♦ Very Safe, but Dangerous If Key Is Lost
- O/s Instigated and Controlled
  - ♦ Most Secure, as O/S Is Always in Control

9.5 Violation Notification

- Should You Indicate That Security Has Been Violated?
- Should You Give a Second Chance
- Should You Have a Call Back Scheme
- How Can You Capture Identifying Information
- How about Attaching Violators to "Dummy" Database?

9.6 Typical Implementations

- IDMS Security
  - Control on Use of DML Verb
  - Via Privacy Locks and Subschema
    - Find
    - Get
    - Store
    - Modify
    - Delete
    - Insert
    - Remove
- IMS Security
  - Authority to Retrieve a Segment
  - Authority to Update a Segment

- **System 2000 Security**

Component Number	Passwords			
	Able	**Baker**	**Charlie**	**Dog**
1	R.W.	RUWV	RW	R.WV
2	R.W.	RUWV	R.W.	R.WV
3	R.W.	RUWV	R.W.	R.WV
4	R.W.	RUWV	R...	R.WV

R = Output Authority
U = Update Authority
W = Selection for Purposes of Retrieval Authority
V = Selection for Purposes of Update Authority

Entry Key Security: User must Specify the Correct Value For a Designated Component Before Access to the Structure Is Allowed.

Relational Security (ANSI SQL)

- GRANT SELECT ON TABLE *employee* TO *ford*
- GRANT INSERT, DELETE ON TABLE *stats* TO *smith*
- GRANT SELECT ON TABLE *myrek* TO *public*
- GRANT SELECT, UPDATE (*salary*, *tax*) ON TABLE *stats* TO *nash*

9.7 Security Summary

- Easy to Formulate, But..... Difficult to Implement
- Must Be O/s Based to Provide Enforcement
- All Violators must First Be Logged, and
- All Violators must Be "Prosecuted", Else Security Is a Farce !

B I G Q U E S T I O N

If Security Is Changed, Do Currently Executing Run-units Come under Old Rules or New Rules?

10.0 Installation & Maintenance

- Introduction
- Key Activities
- Bug Verification and Bugbase
- New Releases, Etc
- Special Versions
- Version Archives
- One-last Backup
- Summary

10.1 Introduction

Installation and Maintenance Is:

The Creation of Execution-ready DBMS Copies

And

The Maintenance of Those Copies

10.2 Key Activities

- Installation Procedures
- Vendor Supplied Instructions
- Site Guidelines and Procedures
- Installed Software Backups
- Validation and Testing
  - Series of Layered Tests to Verify Correct Operation of Software
  - Duplicate Tests on Other DBMS, or Non-DBMS Mechanisms to Validate Tests

10.3 Bug Verification and Bugbase

- Listing and Explanation of Known Errors
- Method of Validating New Errors

10.4 New Releases, Etc.

- Installation of New Features
- Verification of Bug Correction
- Re-initialization of DBMS
- Expiration Mechanism

10.5 Special Versions

- Memory Trade-off
- Maximize Memory Size
- Minimize Memory Size
- I/O Trade-offs
- Physical Database Row Size, i.e., Blocking Factor
- Buffers
  - Size, Number
  - Memory Residency Size
  - Restrictions on Types of Usage

Special Versions (Cont)

- Language Restrictions
- Create Special Versions with
- Certain Commands or Functions Zapped
- Examples
  - No New Database
  - No Schema Modification
  - No Update Commands
  - No Audit Trail Suspension
  - No Security Modifications

10.6 Version Archives

- Backups Against "Wear-outs"
- Backups of Version Generation Procedures
- Database Conversions/upgrades
  - DBMS Storage Structure and Thus Access Strategy Will Change
  - Vendors Supply Conversion Routines to New Versions
  - "Old" Data must Be "Refurbished" Through Version Evolutions to Remain Operative

10.7 One Last Backup

- DDL
  - Schemas
  - Subschemas
- Source Data Unload and Load Program

10.8 Installation & Maintenance Summary

- All DBMS Have Undetected Bugs !
- You Will Find the next Bug!
- Special DBMS Version Can Achieve Specialized Performance and Greater Security And/or Privacy
- All Databases Will Eventually Become Unusable from DBMS Version Evolutions !
- All Databases Should Be Upgraded as New DBMS Versions Are Released!

11.0 Application Optimization

- Logical Database
- Physical Database
- Interrogation
- System Control
- Summary

11.1 Introduction

- The Process of Analyzing the Operational Characteristics Of:
  - Databases
  - Computer Hardware Support
  - Systems Software Support
  - Applications Software
- Creating Design Modifications to Achieve
  - Performance Enhancements
  - Greater User Satisfaction
  - Improved Corporate Command And Control.

11.2 Logical Database Application Optimization

- Database Area/scope
  - Too Big
  - Too Small
- Coherent Policy
  - Bigger than Thought
  - Too Many Conflicts
- User View Analysis
  - Not Enough
  - Not Well Done
- Column Specification
  - Too Many
  - Too Few
  - Not Well-understood
  - Too Cryptic Name
  - Tooooo Long Name

Logical Database Analysis (Cont)

- Relationships
  - Too Many
  - Too Few
  - Too Transitive
  - Not Well Understood
  - Rules for Insert, Modify, Etc. Not Followed

- Model Transformation
  - Too Much
  - Too Little
  - Wrong DBMS

- Sub Schemas / Views
  - Too Few
  - Too Many
  - Too Broad
  - Too Narrow
  - Renames Poor
  - Defaults, Nulls, Badly Utilized, Etc.

11.3 Physical Database Application Optimization

- Storage Structure
  - Dictionary -- See Logical Database
  - Index Design
    - Over Utilization of Multiple Secondary Keys To Derive a Unique Row Access
    - Use of Secondary Keys Rather than Sequential Access On Small Sets of Data
  - Relationships
    - Chains Too Long
    - Constant Changes and Wrong DBMS
  - Data
    - Fixed/variable Format
    - Fixed/variable Length Rows

Physical Database Analysis (Cont.)

- Access Strategy
  - Wrong DBMS

- Data Loading
  - Too Large Loads
  - Too Many, Too Small Loads
  - Process Needs Optimization

- Data Maintenance
  - Updates Destroy Database Performances
  - Wrong DBMS

- Database Maintenance
  - Daily Backup of Physical Database Takes 28 Hours

11.4 Interrogation Application Optimization

- Balanced Usage of Languages
- HLI: Change Requests Exceed Maintenance Capabilities
- NL: Very Stable Procedures "Chewing-up" the Machine
- Performance Details
  - Run Frequency
  - Average Data Volume
  - Data Selection Clauses
  - Navigation With/without Data Retrieved
  - Updating Key and Non-key
  - Relationship Modification

11.5 System Control Application Optimization

- Audit Trails
  - ♦ Correct
  - ♦ Optimum Design
  - ♦ Risk Efficient
- Backup and Recovery
  - ♦ Sufficient and Complete
  - ♦ Risk Efficient
- Message Processing
  - ♦ Sufficient and Complete
  - ♦ Easy to Understand
  - ♦ Utilized to Promote Training
- Reorganization
  - Logical -- Keeping Current
  - Physical -- Optimum Efficiency

System Control (Cont)

- Security and Privacy
  - Leaks Can Be Titanic
- Multiple Database Processing
  - Databases Too Finely Divided
  - Sufficient Interlocks
  - Multiple Levels Being Correctly Served
- Concurrent Operations
  - Correct Identification of Conflicts
- DBMS Installation and Maintenance
  - New Updated Versions
  - Bugs Fixed
  - Special Performance Versions

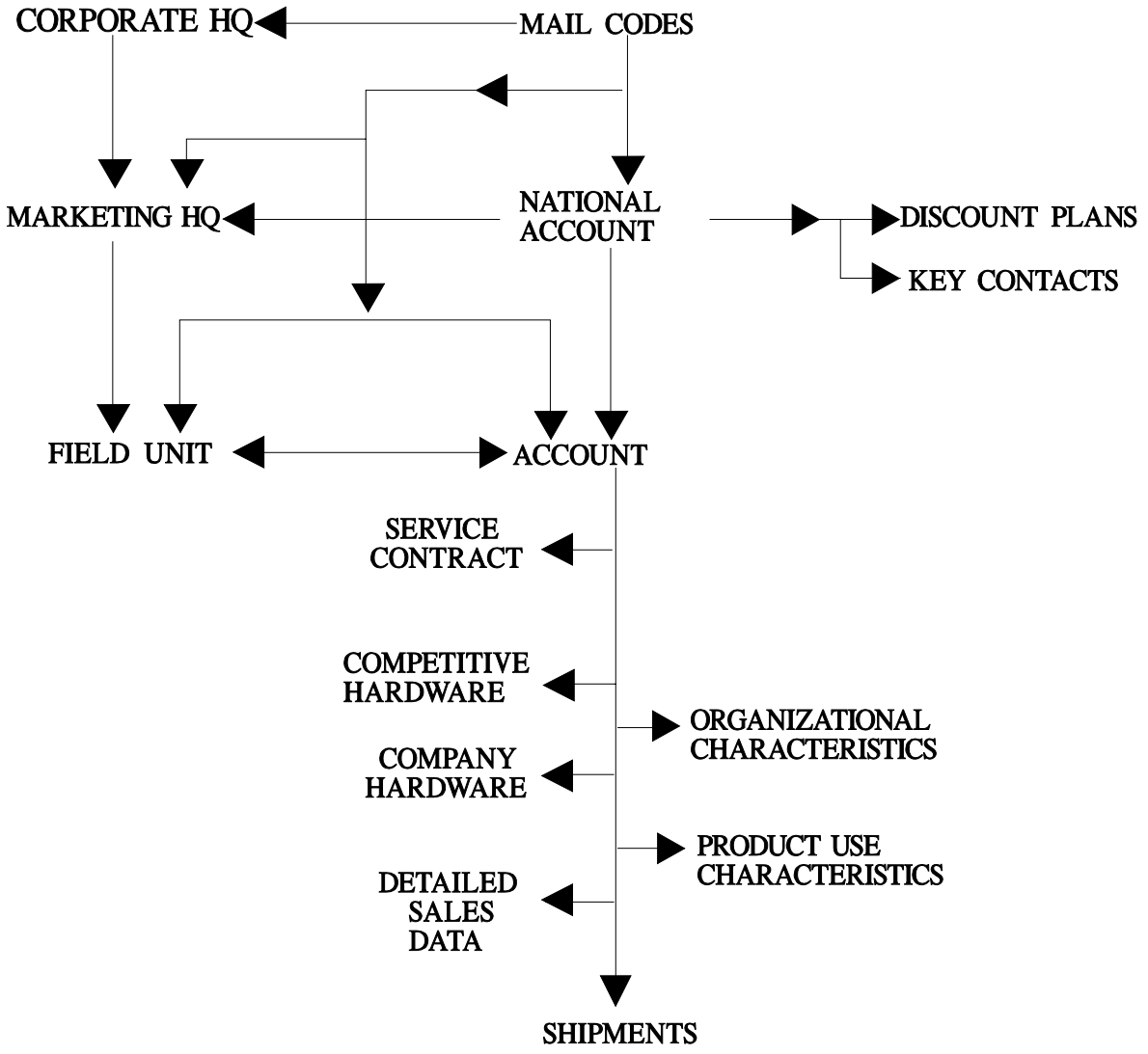
11.6 Summary

- An Integral Part of Total Database Projects
- Recursive and Cyclical
- Schedule Resources for Optimization on a Regular Basis
- Possible Outcomes
  - De-databasing
  - Re-DBMSing

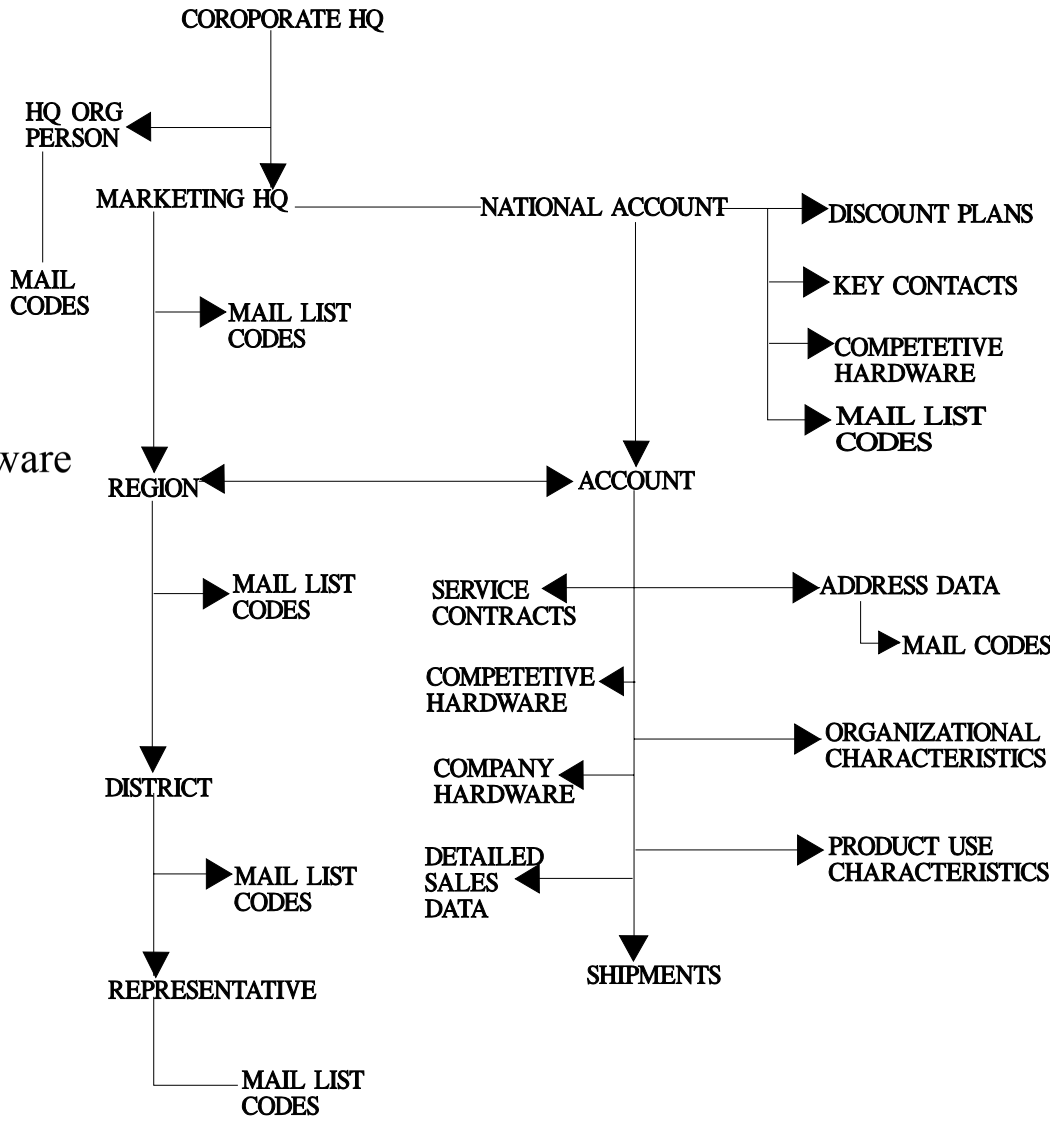
11.7 Application Optimization Example

Original Design

Problems: As a "data warehouse" there was little or no summary data. Everything was essentially in capture format. This was a Operational Data Store, not a data warehouse.



Round 1 Optimization



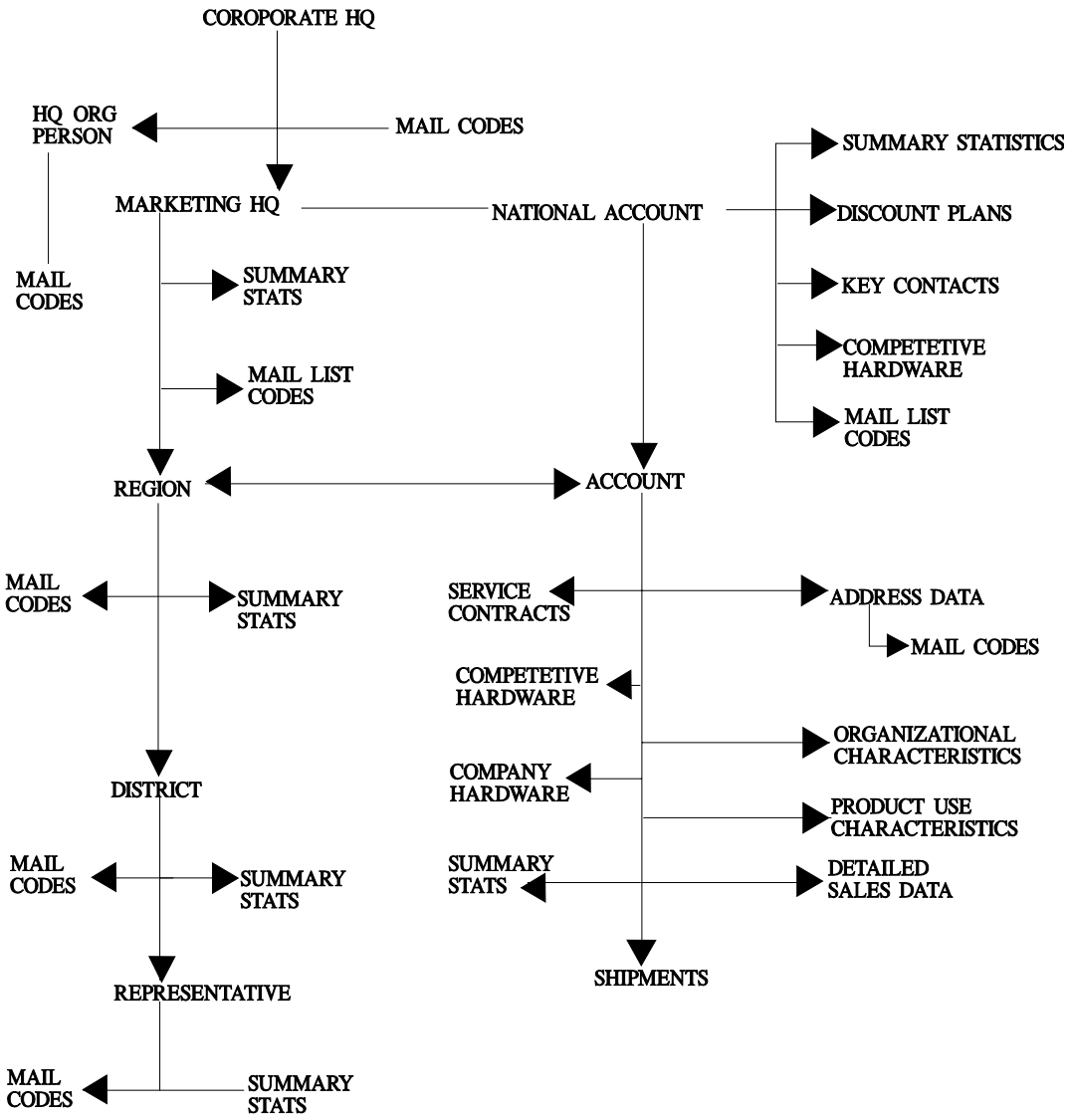
Changes

- Added Mail Codes
- Creation of Address History
- Creation of Competitive Hardware

Round 2 Optimization

Changes

- Added HQ Persons
- Added Summary Stats for Marketing, Region, National Account, and Account
- Added Account Detailed Sales data



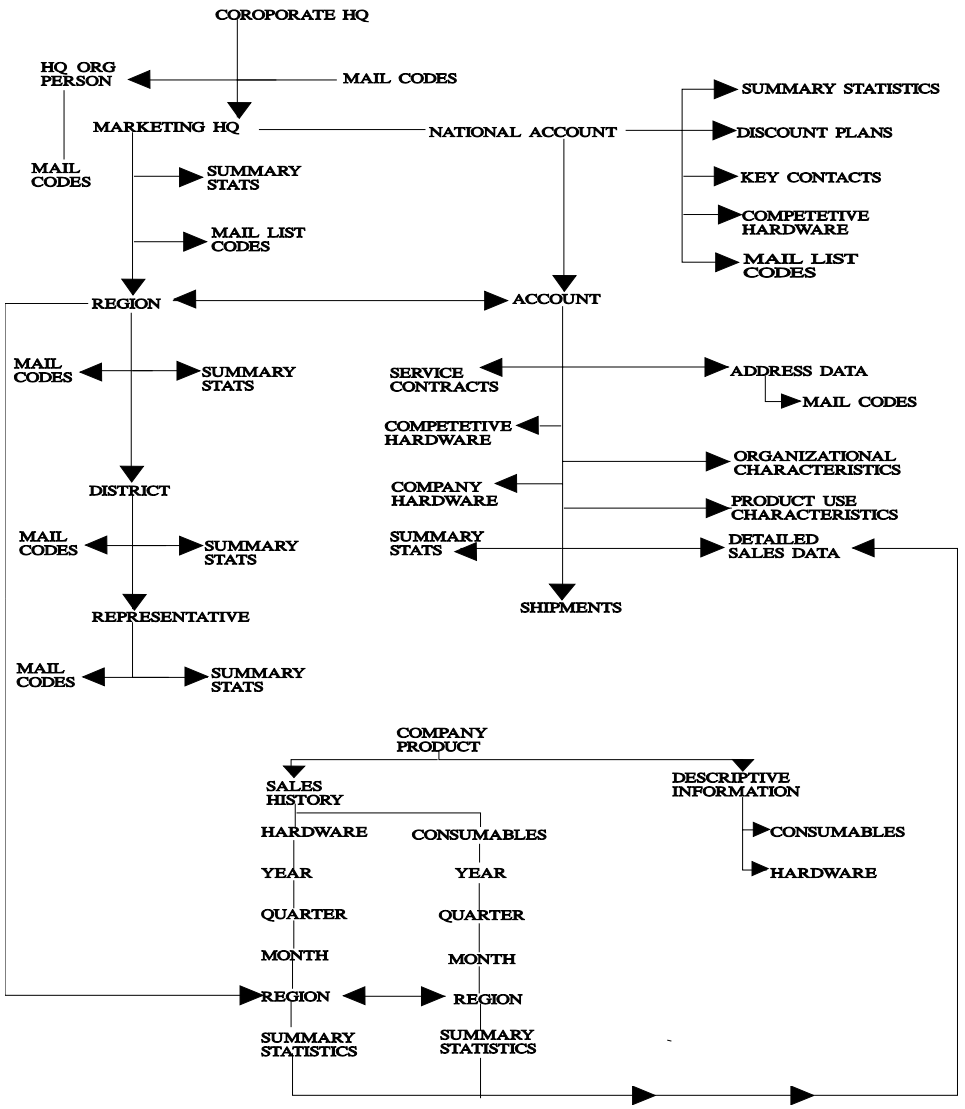
Round 3 Optimization

Changes

Mainly to create a "2nd" data warehouse tree for all company product data.

There are thus two trees:

- 1st for all sales data by corporate organization, and
- 2nd for all sales history by specific corporate product.



12.0 System Control Summary

- Audit Trails
- Message Processing
- Backup and Recovery
- Reorganization
- Security and Privacy
- Multiple Database Processing
- Concurrent Operations
- Installation and Maintenance
- Application Optimization

- **Audit Trails**

  - ♦ Carefully Assess Your Exact Needs
  - ♦ Critically Evaluate DBMS's Delivery Vs. Promises
  - ♦ Implement Your Own Audit Trail If Deficient

- **Messages**

  - ♦ Some Indicate That All Is "Roses", Most Do Not!
  - ♦ Implement a Messages Database for Centralized Meanings and Training Needs

- **Backup and Recovery**

  - ♦ Complex, Critical Issue
  - ♦ A Lost Database Will Kill Database
  - ♦ Make Sure TP and DB Are Completely Coordinated So No Transactions Are Lost

- **Reorganization**
  - ♦ Vital Component of Every Database Project
  - ♦ Vehicle for Database Evolution
  - ♦ Permits Optimization of Storage Structure Parts

- **Security**
  - ♦ Easy to Specify, But...
  - ♦ All Violators must First Be Trapped
  - ♦ All Violators must Be Dealt with Severely,
  - ♦ Else it Is a Farce

- **Multiple Database Processing**
  - ♦ Always Needed
  - ♦ Method to Integrate Physical Databases
  - ♦ Most Capability Is Through HLI

- **Concurrent Operations**
  - Know the Conflicts
  - Set up Procedures to Notify Users When
  - Unscheduled Shut Downs Occur

- **Applications Optimization**
  - Integral Part of All Database Projects
  - Set Aside Schedules and Money to Pay for it
  - No Evolution = No Progress

- **Installation and Maintenance**
  - All DBMS's Have Bugs
  - Special Versions Can Provide Improved Performance Greater Security and Protection
  - All "Old" Databases Will Cease to Run Unless Updated

In Short,

- Bad Audit Trails May Kill the Entire Enterprise
- Bad Security May Be Illegal
- "All" Databases Will Cease to Operate Without Updates
- Always Keep That One Last Backup of Every Database
- System Control Is Certainly...

The

Database Administrator's

Waterloo

Your Choice Is Whether

You Are Wellington

Or

You Are Napoleon