# Teradata RDBMS

Roland Pfeffer
NCR, Teradata Division

You've never seen your business like this before.

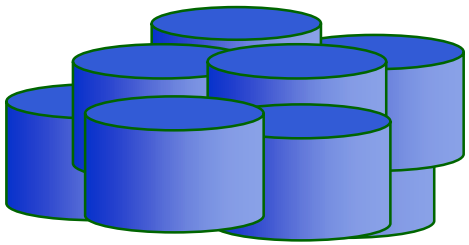
Teradata
a division of NCR

# Agenda

- **Teradata RDBMS**
  - shared nothing architecture
  - hash algorithmus, join, sort, aggregate
- **Teradata Tools & Utilities Overview**
  - Load Tools, Query Tools
  - Administrational Tools
- **Teradata Data Load**
  - fastexport, fastload, multiload, bteq
- **Teradata Documentation**

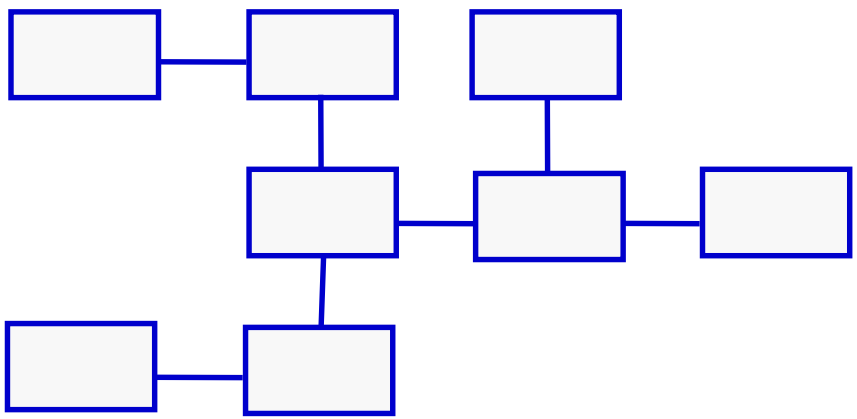
# Data Warehouses: It's Not Just About Size

## Amount of Detail Data



## Number of Concurrent Users

## Complexity of the Data Model



## Query Complexity

- Simple, direct index
- Moderate, multitable join
- Complex, 10-way table join; includes regression analysis

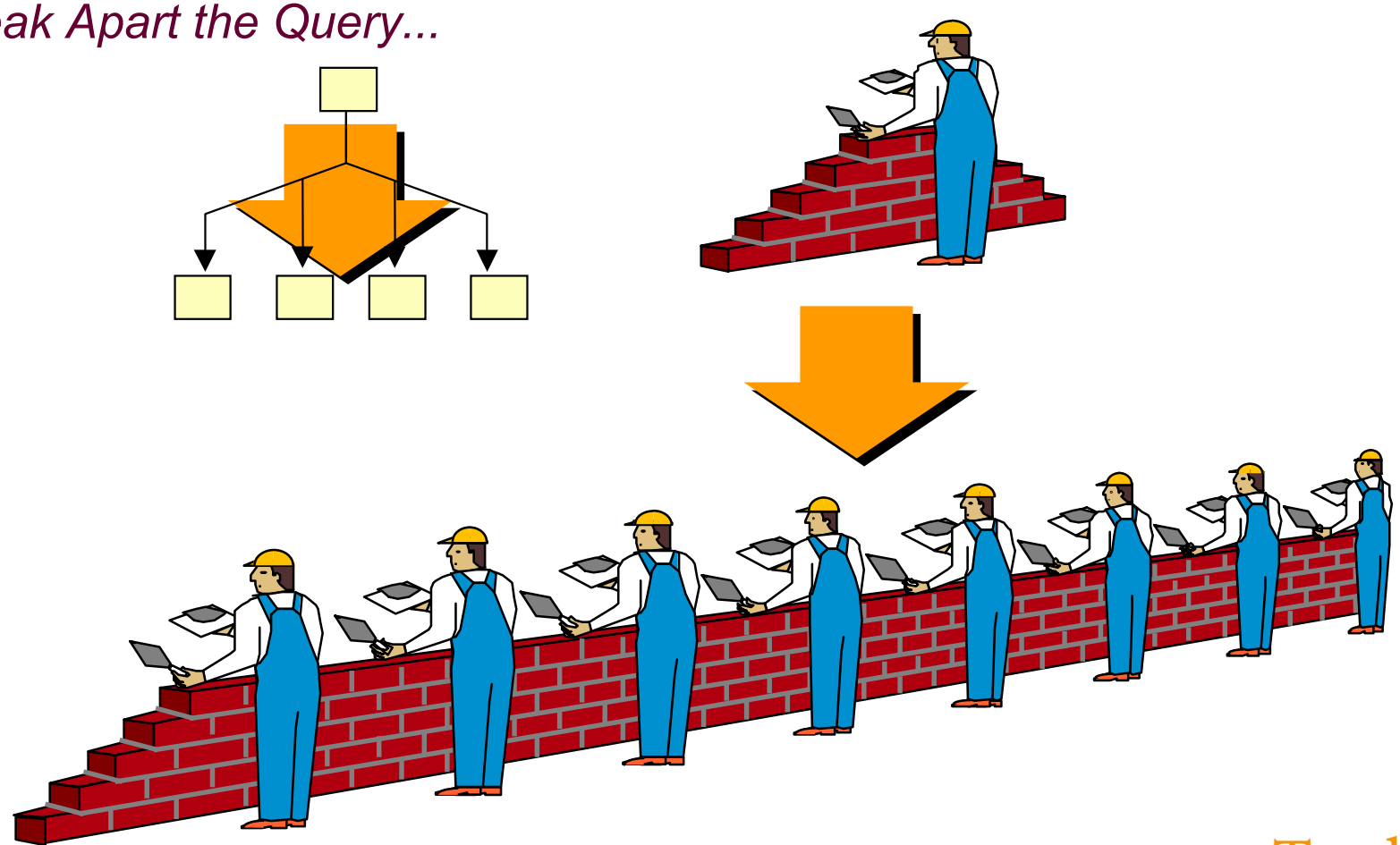
Teradata
a division of NCR

# Parallel Processing, the Foundation:

## Performance & Capacity & Scalability

*Break Apart the Query...*





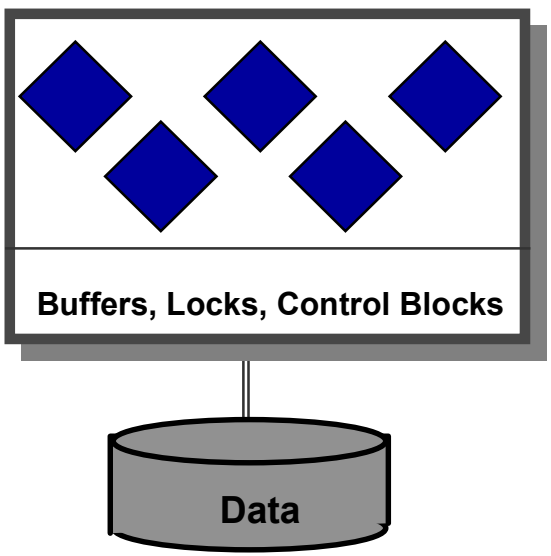
Teradata
a division of NCR

# Parallel Database Architectures

## *Shared Everything*

***well known RDBMS***

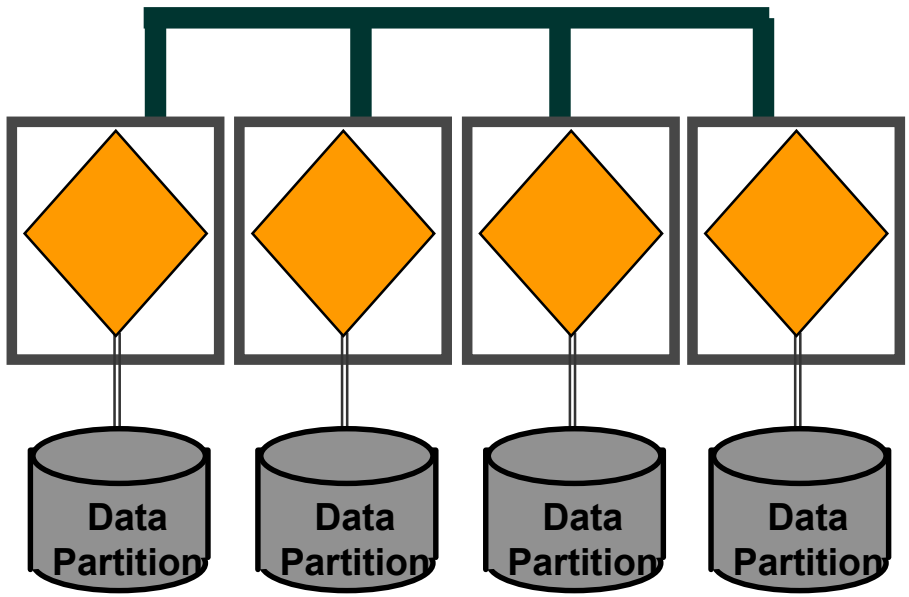
- **A single database buffer used by all UoP's**
- **A single logical data store accessed by all UoP's**
- **Scalability limited due to control bottlenecks and scalability of single SMP platform**




## *Shared Nothing*

***Teradata***

- **Each UoP is assigned a data portion**
- **Query Controller ships functions to UoP's that own the data**
- **Locks, buffers, etc. not shared**
- **Highly scalable data volumes**

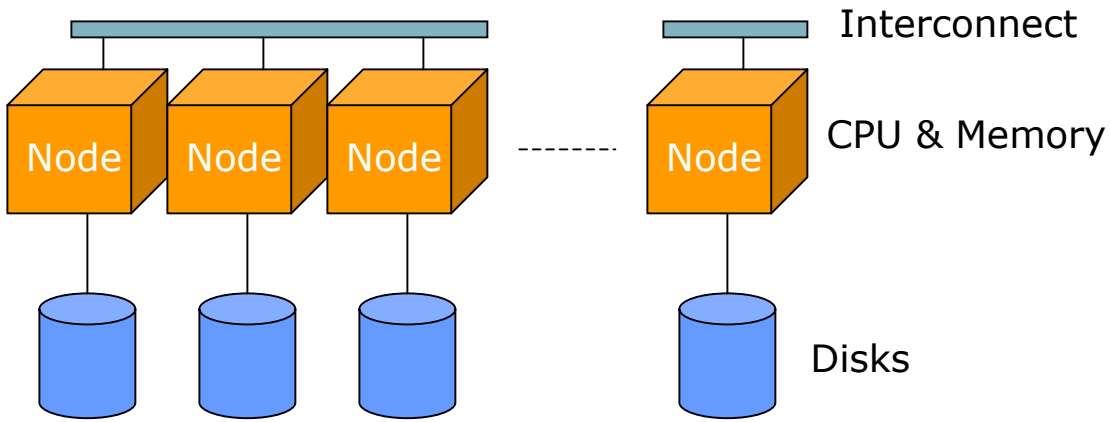



◆ **- Unit of Parallelism**

Teradata
a division of NCR

# Shared Nothing Architecture

**Massively Parallel Processing (MPP) - Loosely Coupled - "Shared Nothing"**



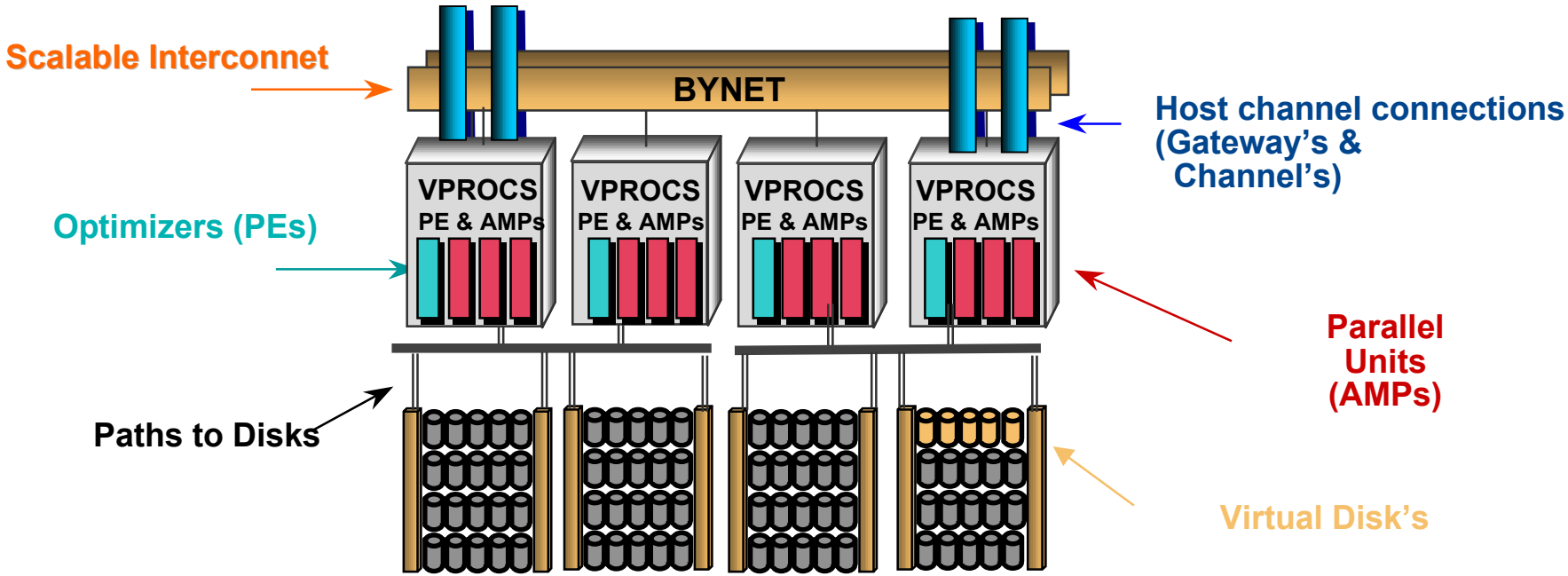

**Characteristics**

- **Each node owns a portion of the database**
- **Nodes are connected via interconnect**
- **Each node is a 2 way SMP**
- **Load balancing handled by Teradata**
- **Linear Scalability**
  - **To any size configuration**
  - **Allows flexible configurations**
  - **Incremental upgrades**
  - **Maximum utilization of SMP resources**

# NCR DWH Architecture

**Shared Nothing** enables ***hardware scalability***

Teradata's **Parallel Everything** enables ***software scalability*** by eliminating single points of control at all levels



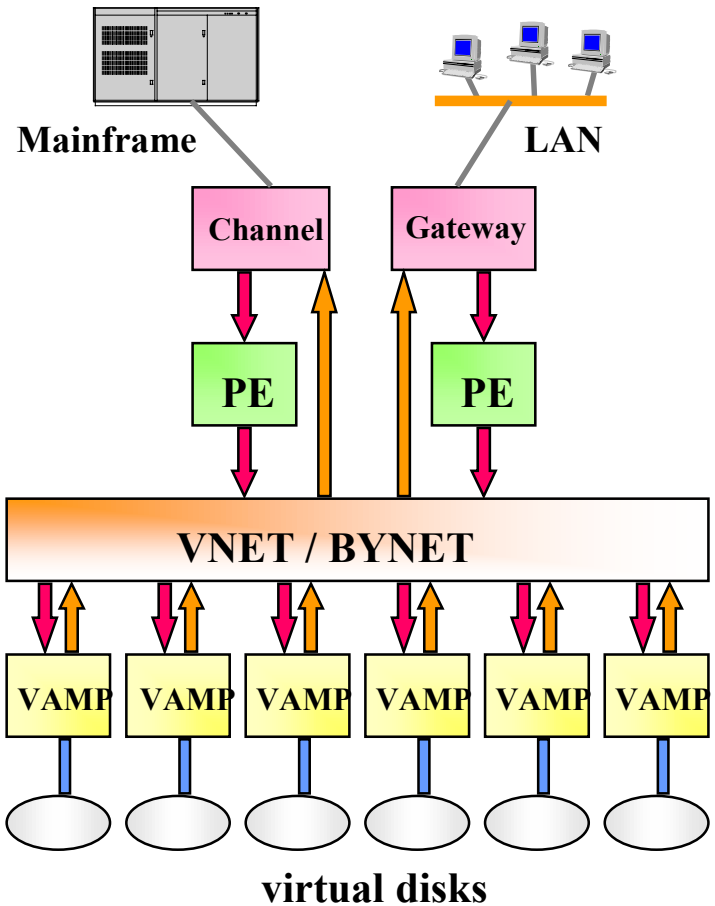

**Degree of Parallelism = Number of AMPS,**

**Number of parallel units independent from tables or querys**

***Teradata is "parallel aware" - Teradata <u>always</u> use <u>all</u> units of parallelism***

Teradata
a division of NCR

# Teradata Software Architecture




***Teradata RDBMS = Parallel Shared Nothing Architecture***

> PE: Parsing Engine
- Session control
- Parser, Optimizer & Dispatcher

> BYNET:

*Communication between VPROCs*
- Control- & Data messages
- Synchronization, Sort & Merge

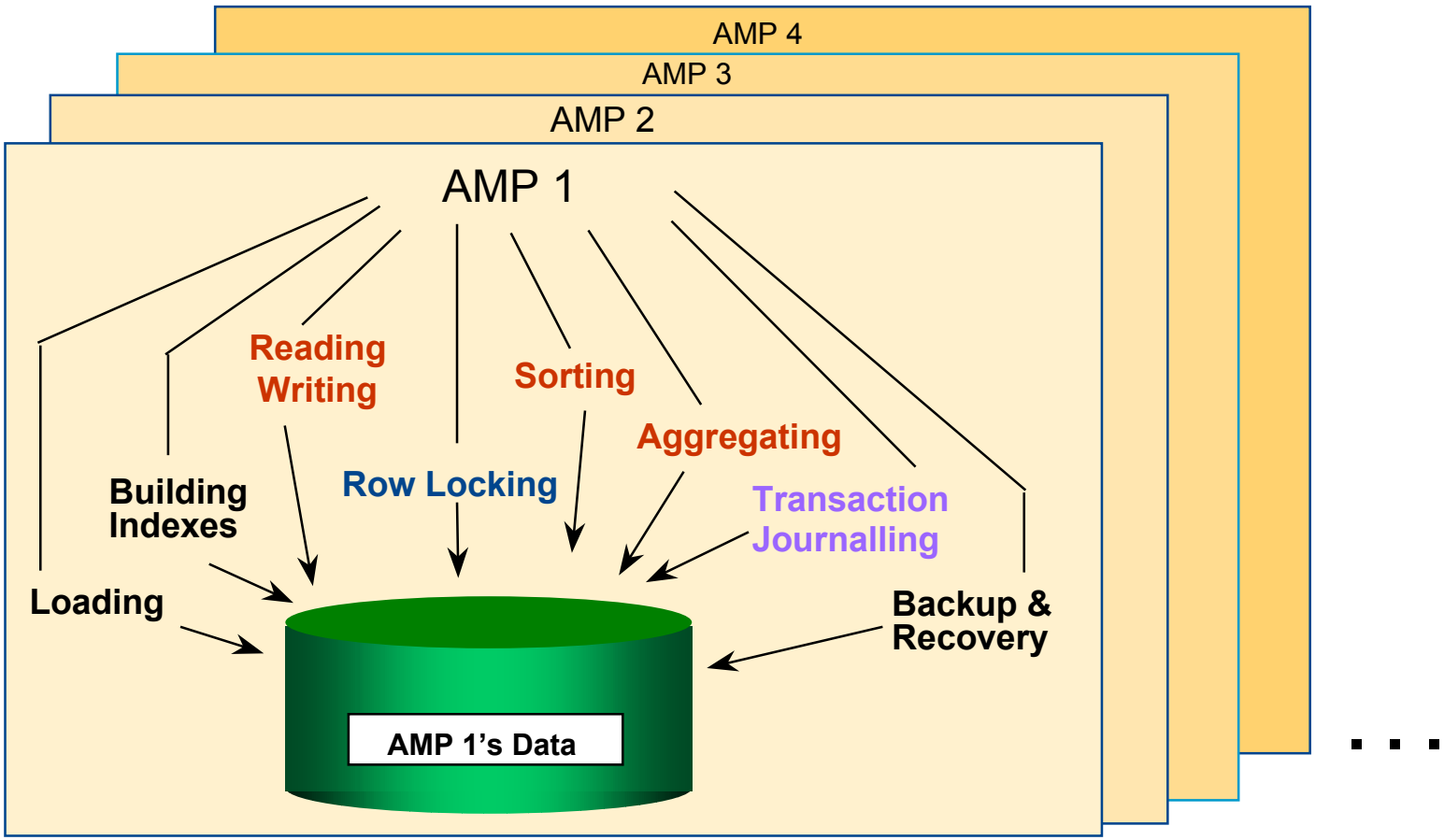
> VAMP: Virtual Access Module Program

*each AMP owns a portion of the database*
- Data access, Insert, Update, Delete
- Cache management
- Journaling, Backup & Recovery
- Concurrency control & Locking

Teradata
a division of NCR

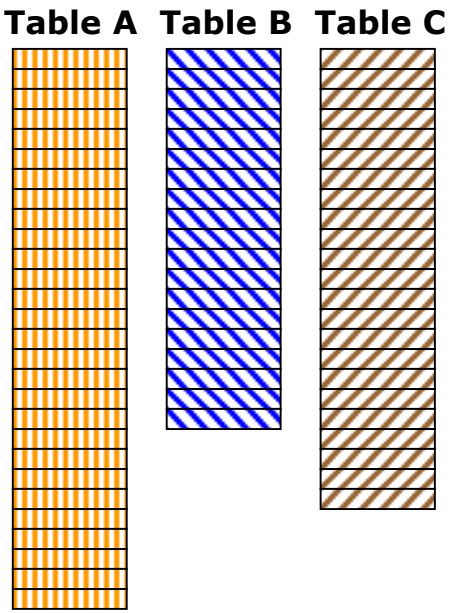
# The Parallel Foundation

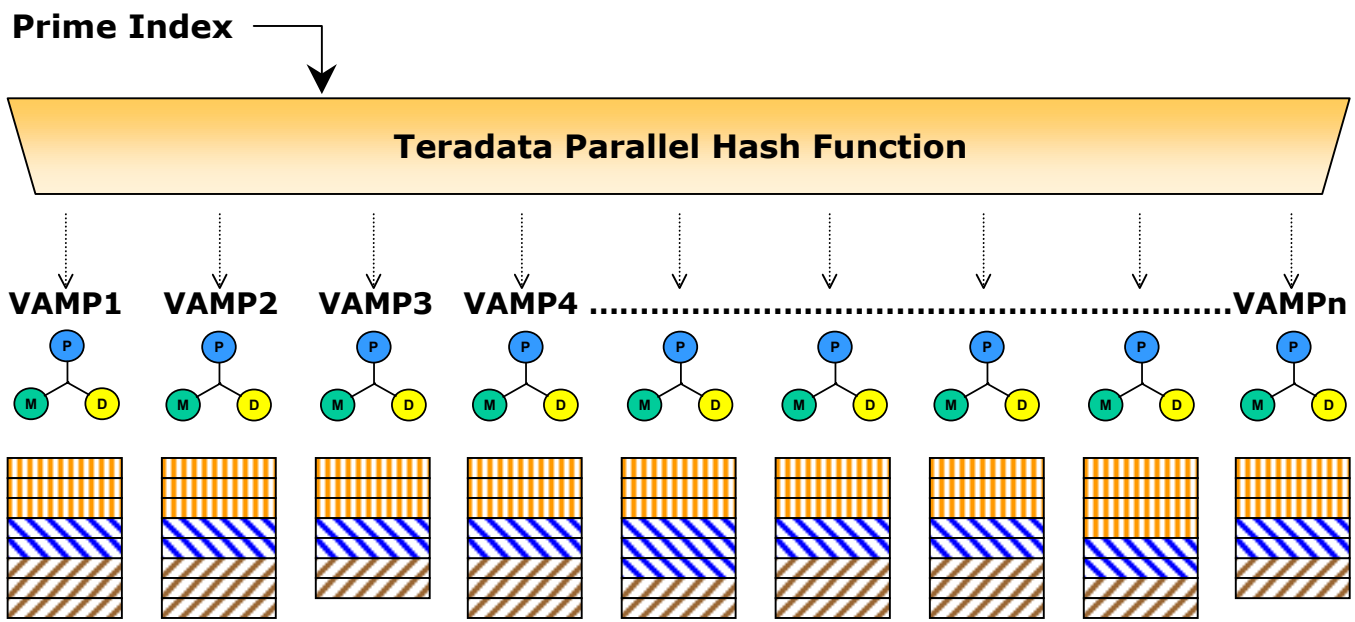
Each parallel unit (AMP) owns and manages it's own data



# Teradata Data Distribution

- Rows are distributed evenly by hash partitioning
  - Done in real-time as data are loaded, appended, or changed.
  - No reorgs, repartitioning, space management
- Shared nothing software:
  - Each VAMP owns an equal slice of the data.
  - Each VAMP works exclusively & independently on its rows
  - Nothing centralized: No single point of control for any operation (I/O, Buffers, Locking, Logging, Dictionary)

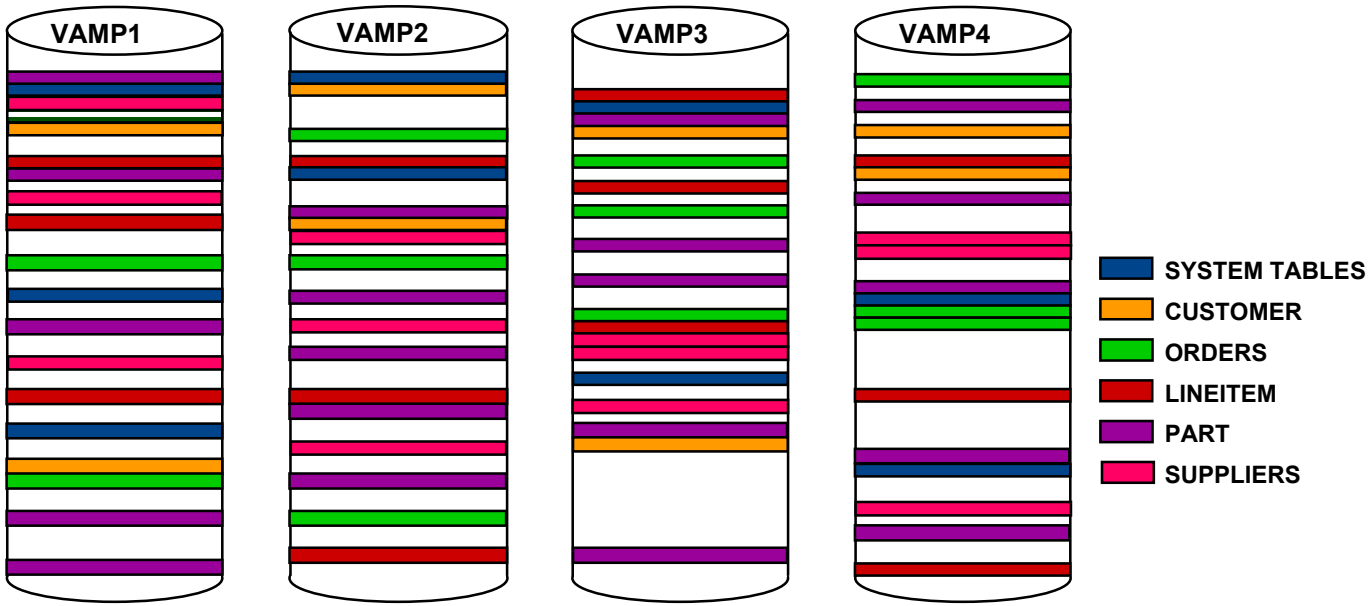



Teradata
a division of NCR

# Teradata Data Management Illustration

- **Random, automatic data distribution & placement**
- **Real-time, automatic data reorganization**



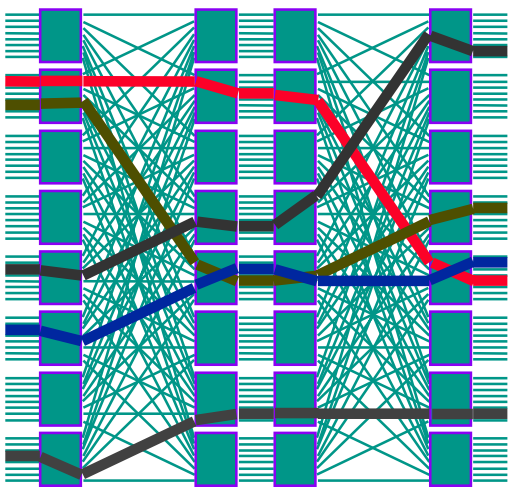

***With Teradata there is no sense of ORDER, therefore there is no sense of DISORDER, eliminating the need to REORDER!***

Teradata
a division of NCR

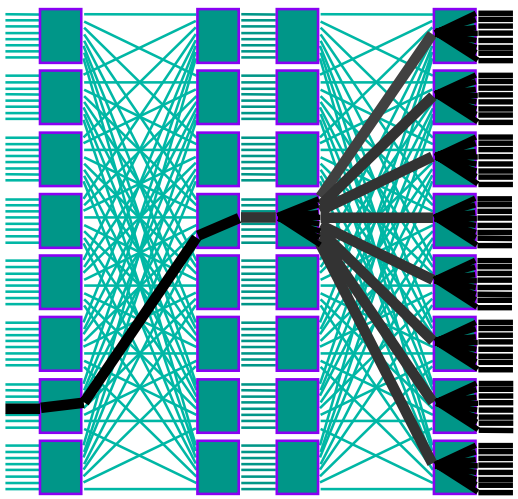
# BYNET™ Interconnect

BYNET works much like a telephone network, where many callers can establish connections, including conference calls and broadcast connections

**Point-to-Point Messaging**



**Broadcast Messaging**



**Specialised Services for Teradata:**

Message passing
Synchronisation
Merge

**Other services:**

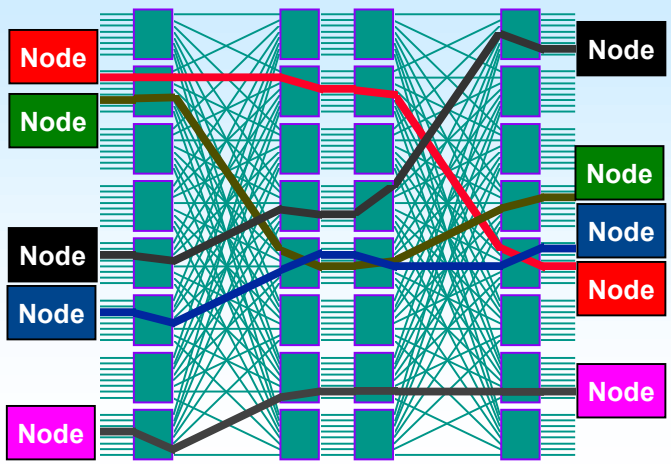
TCP/IP

Bandwidth growths with number of nodes connected

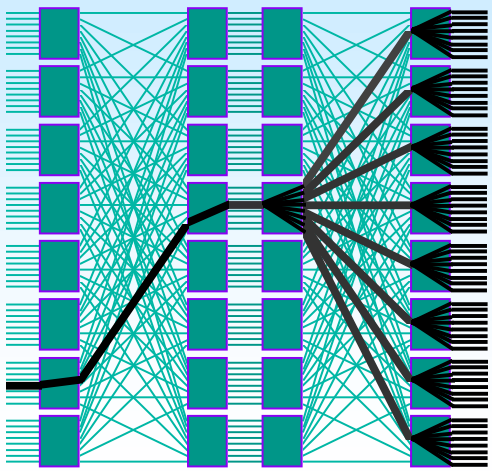
# BYNET™ V2.0

- Optimized for Teradata ADW Performance
  - > Linearly scalable bandwidth (up to 480MB/s per node)
  - > BYNET Low Latency Interface (BLLI) - "lite" communication protocol
  - > Teradata exploits unique BYNET features: broadcast message support, row merge support, multi-fabric message traffic shaping, hardware guaranteed message delivery (including broadcast)
- Proven High-Availability
  - > Each fabric is fault tolerant (multiple paths, redundant power & cooling)
  - > 2 to 4 independent fabrics (no single point of failure)

**Multiple Simultaneous Point-to-Point Messaging**



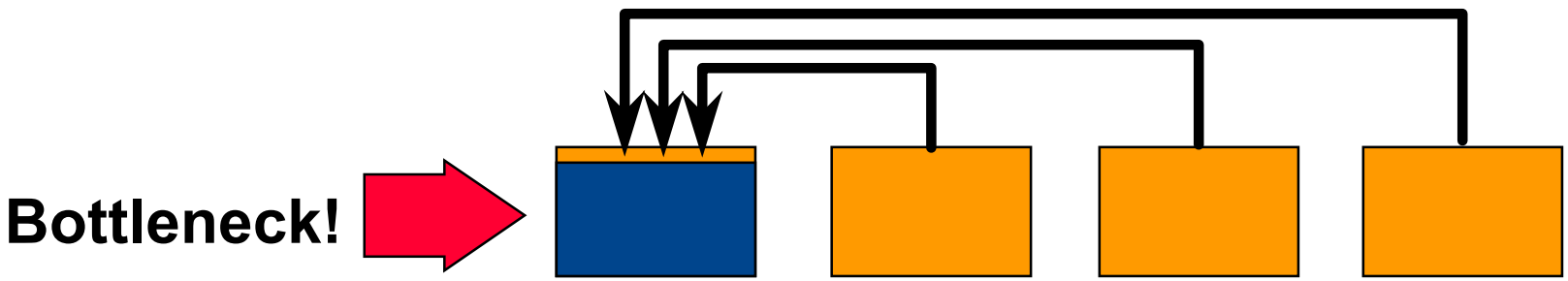

**Broadcast Messaging**



The **Teradata Optimizer** ***chooses*** between Point-to-Point and Broadcast Messaging to select the most effective communication.

# Why is a Join expensive in a parallel DB?

- In order to process records in a Join, they have to be on the same processing unit
- Most RDBMS have to send the records to a **single processing unit**, to perform the Join



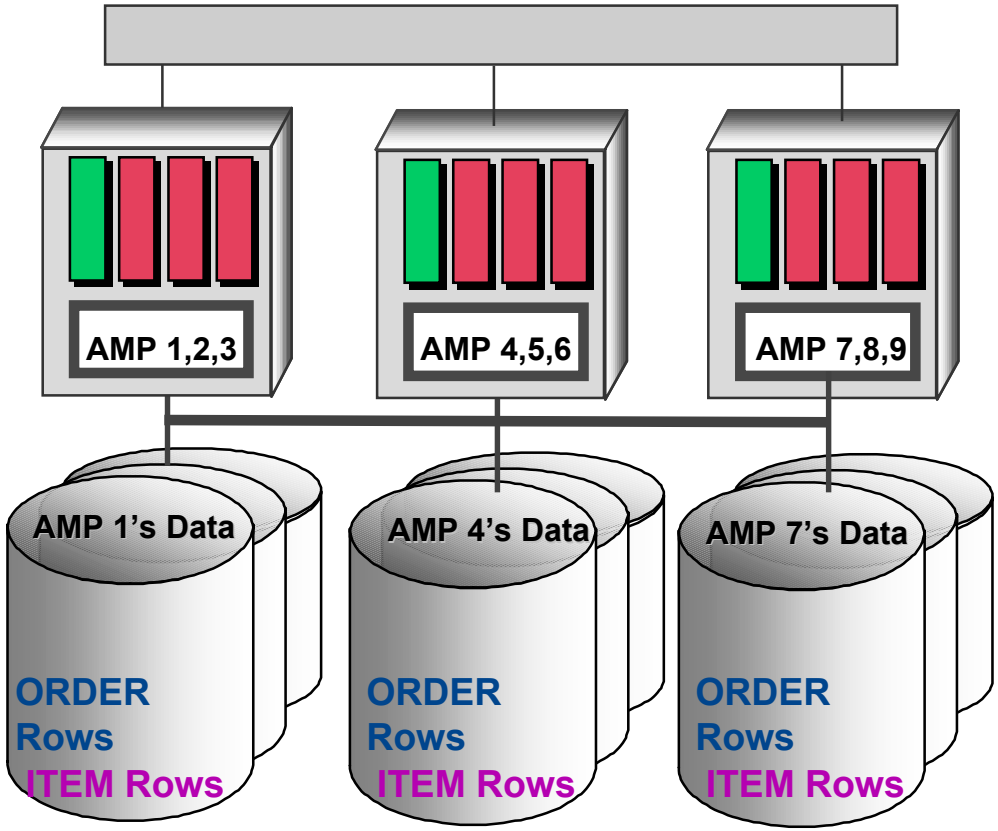
# Teradata Parallel Join

- Either the records are already on the same VAMP (because of proper choice of primary index), or the records to join will be redistributed using the Hash distribution.
- The Join is performed balanced on all nodes

**Balanced Workload**

# Background: MPP Join Geography

- Table Redistribution
  - Item rows are moved to the AMP where their related Orders are located
- Table Duplication
  - Smaller table is duplicated to each AMP
- AMP-local join
  - No movement: Both tables' related rows are already on the same AMP

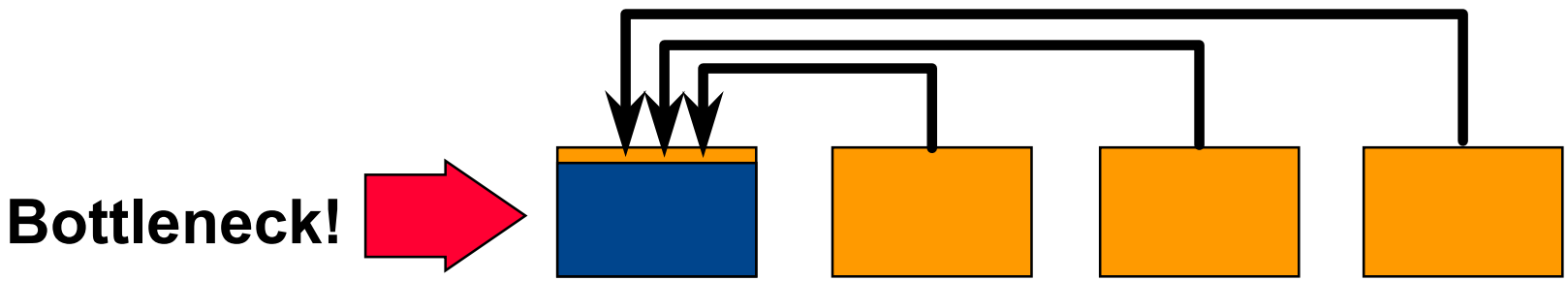



**Prerequisite is a scalable and very fast node interconnection - BYNET**

Teradata
a division of NCR

# Why is Aggregation expensive in a parallel DB ?

- In order to aggregate records, they have to be on the same processing unit
- This requires a redistribution of the records according to the "Group by" clause
- Aggregation is more expensive than a Join, because more records have to be processed, normally, and a mathematic operation (sum, avg, min, max) is performed additionally



# Teradata Parallel Aggregation

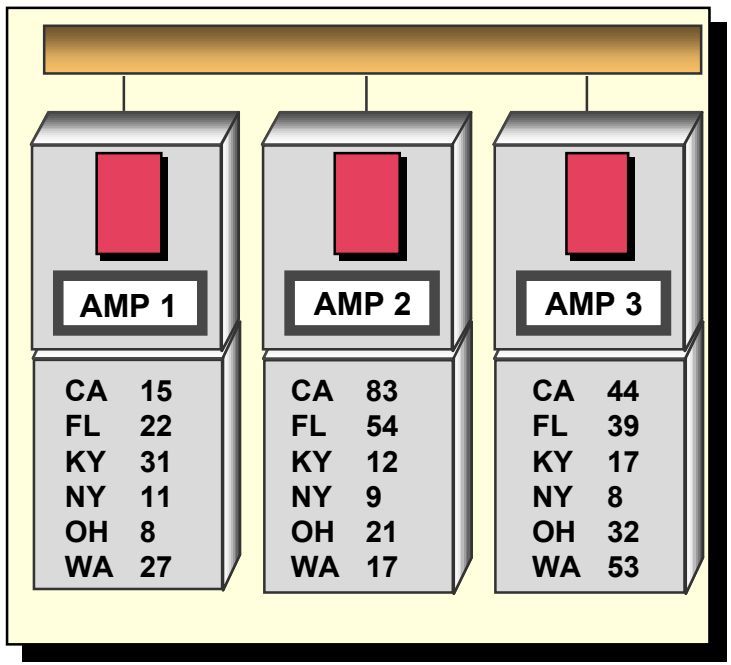
- Each VAMP aggregates his local records first
- Then  the intermediate results are redistributed using Hash for the final aggregation

**Balanced Workload**

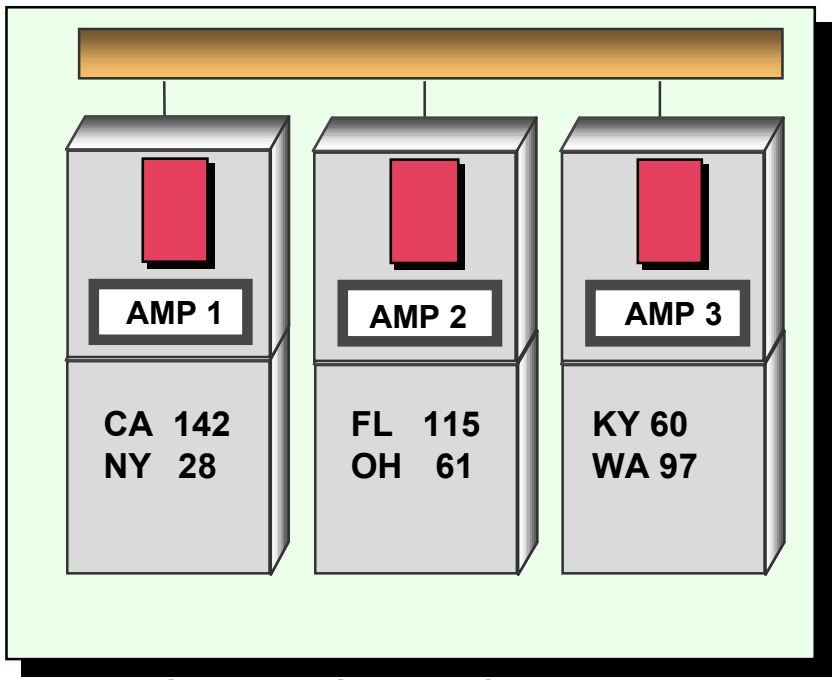
# Fully-Parallelized Aggregations

- Each AMP performs a local count, then executes a share of the global count

*A Count of the Number of Delinquent Customers by State*




**Step 1 - Local Sub-totals**




**Step 2 - Global-Grand-total**

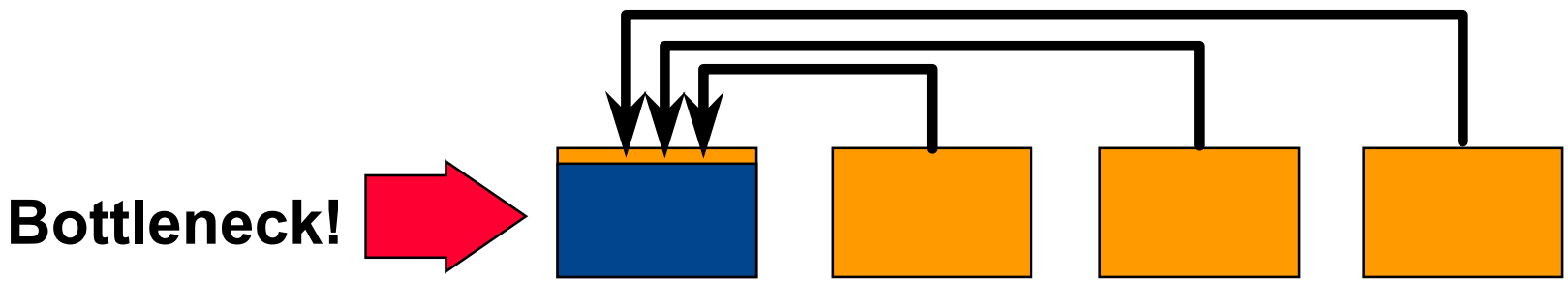
**ALL aggregations in parallel - No single node bottleneck**

Teradata
a division of NCR

# Why is Sorting expensive in a parallel DB ?

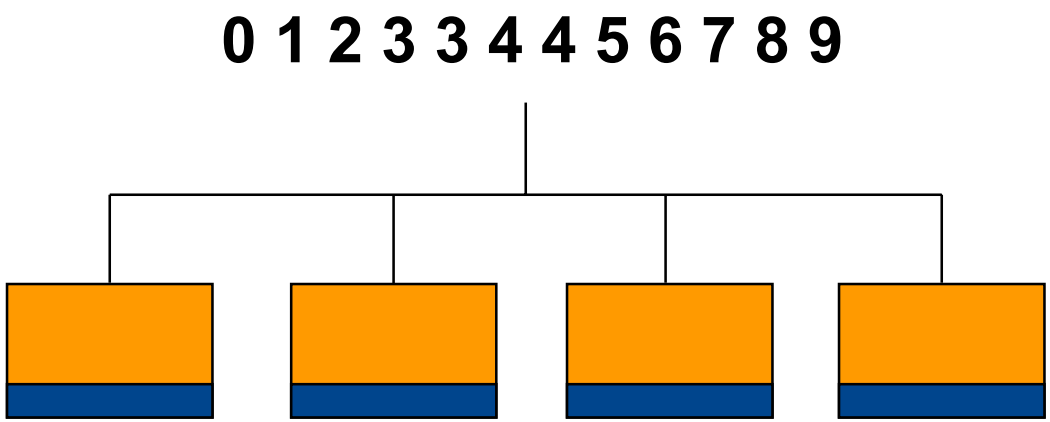
- In order to sort records, they have to be on the same processing unit
- All RDBMS, except Teradata, perform this task serially

**1 4 5 2 7 3 9 8 0 1 6 4 9 0 2 4 3 8 3 5 1 8 4 9 0 3 2**



# Teradata Parallel Sorting

- Each Teradata VAMP performs a local sort first
- The BYNet™ is performing a Sort-Merge of the intermediate results
- <u>No re-destribution necessary!</u>

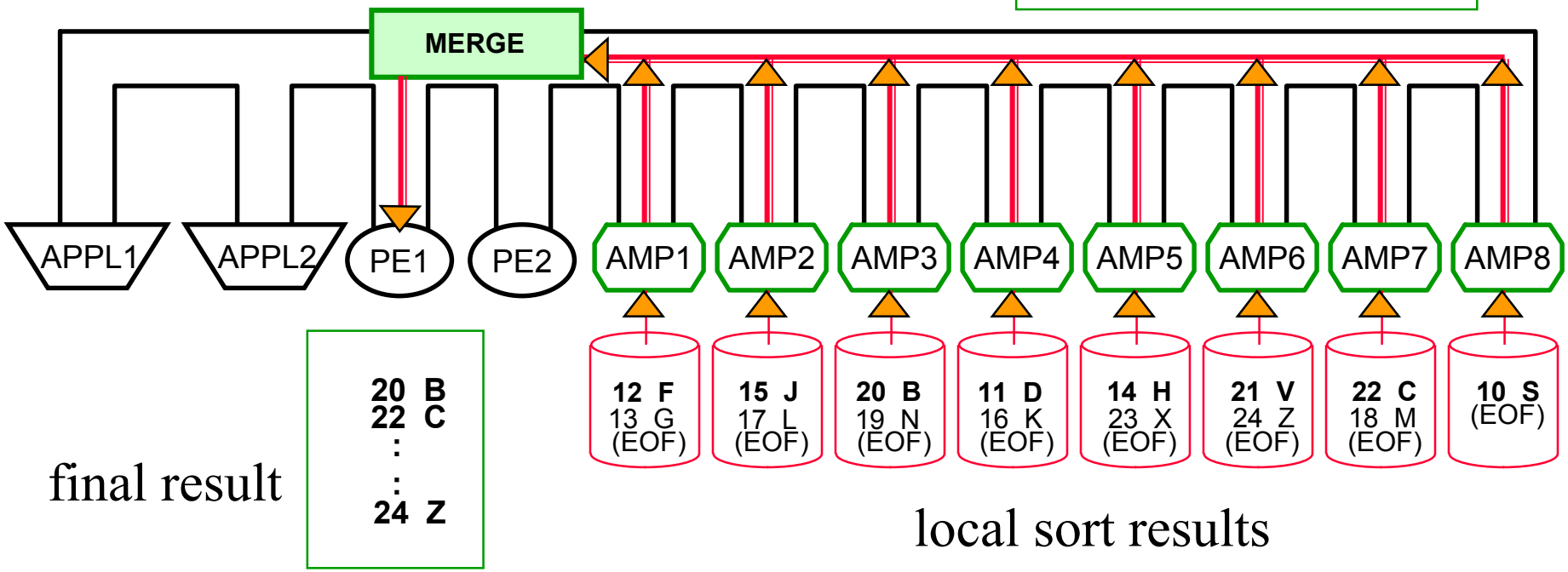



Teradata
a division of NCR

# Fully-Parallelized Sorts

Each AMP performs a local sort, then BYNET executes a final sort/merge on-the-fly as the rows are returned to the user

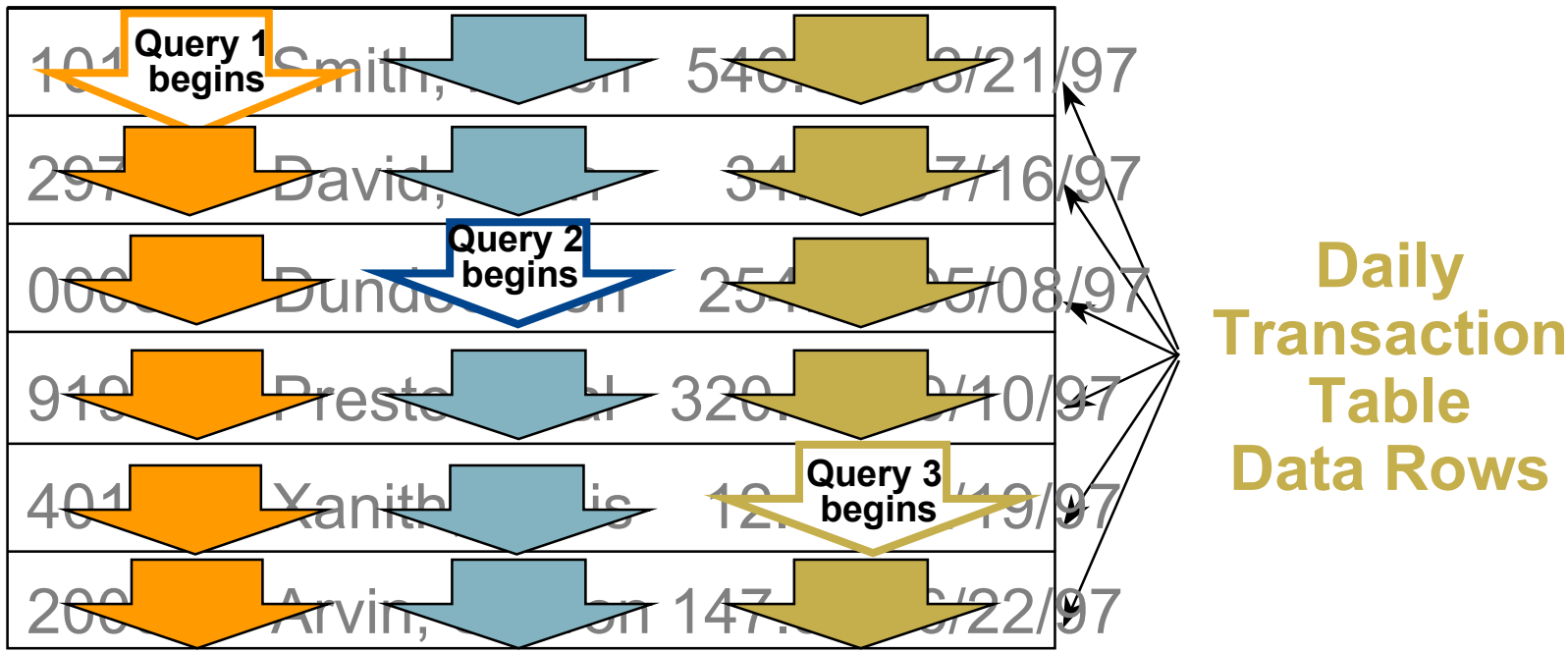
```
SELECT   NUMBER, LETTER
  FROM   SAMPLE
 WHERE   NUMBER > 9
ORDER BY LETTER
       ;
```




**ALL sorts in parallel - No single node bottleneck**

# Synchronized Table Scan & Joins

- Allows multiple simultaneous scans & joins against a single table to share data blocks
- A new query joins the scan / join at the current scan point




**Reduces significantly the query response time in a multiuser environment**

Teradata
a division of NCR

# Teradata Optimizer Intelligence

- Teradata has a parallel-aware, cost-based optimizer with full look ahead capability to **maximize throughput** and **minimize resource contention**:
  - All queries automatically acquire all units of parallelism -- it's built into the database, not the application
  - The optimizer can determine the lowest cost (time) to complete each and every intermediate step within the query plan in order to choose the fastest overall time for a query

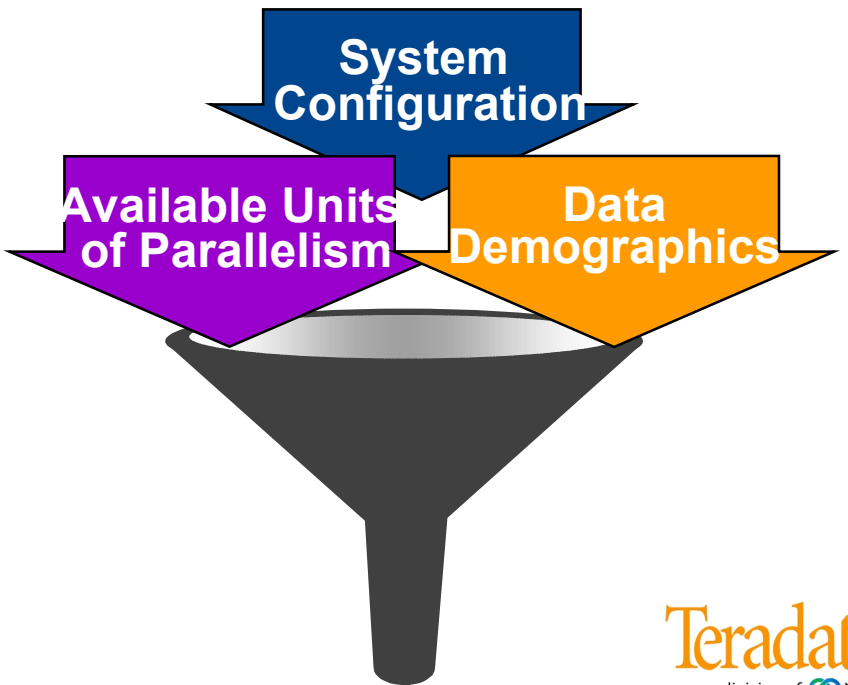
**Why is this important?**
**More and more SQL is generated from tools and quite often it's not *optimized*!**

# Teradata Optimizer: What does it Optimize?

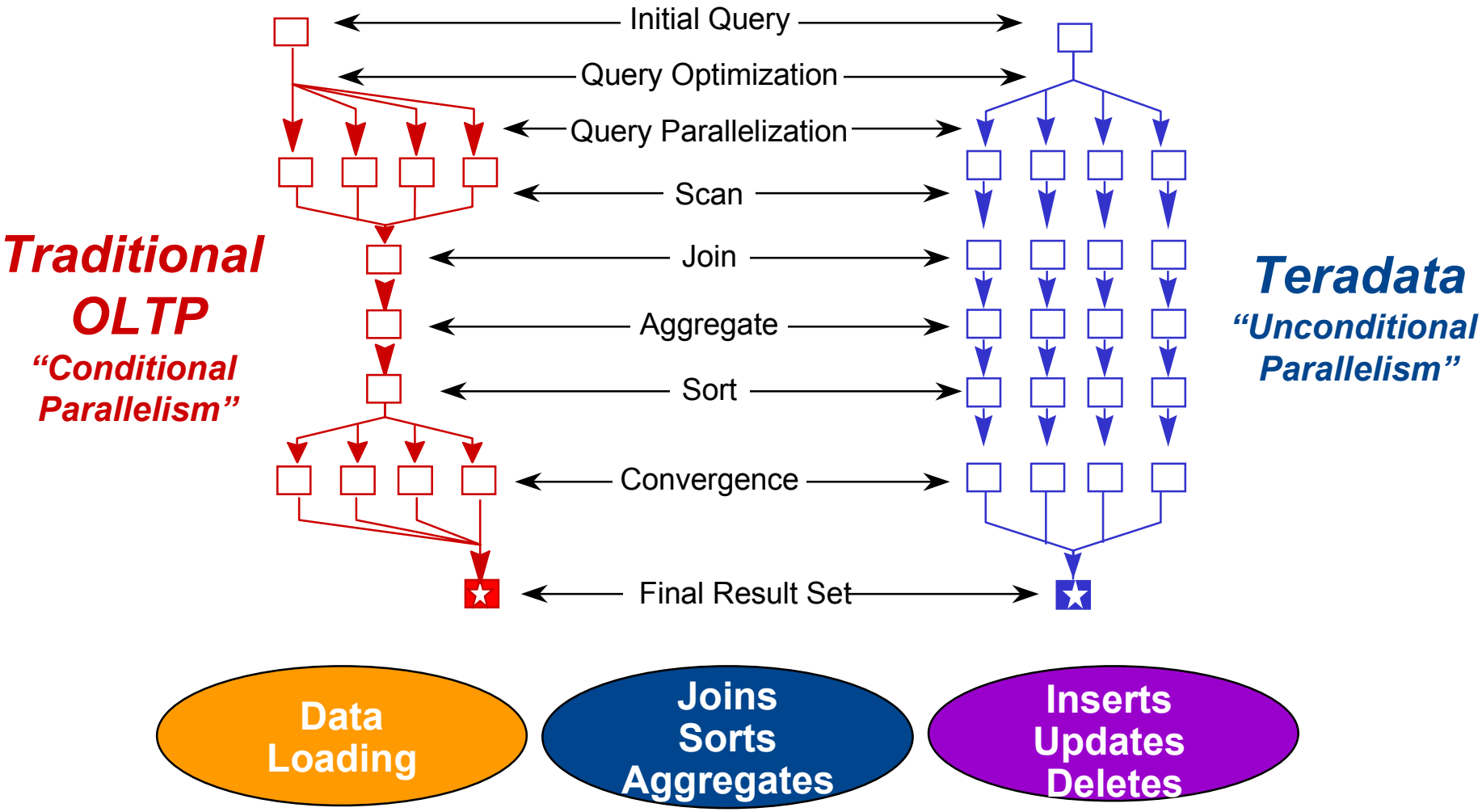
- Access Path: Method of accessing each table
  - Table Scan, Index Use, Bitmap Use
- Join Method: How pairs of table are joined
  - Merge Join, Product Join, Hash Join
- Join Geography: How rows are relocated prior to the join
  - Redistribute Rows, Duplicate Rows
- Join Order: Sequence of table joins
  - 5 table look ahead, pick the cheapest

# Teradata Optimizer
# What it Needs to Know?

- Environment information
  - Number of Nodes
  - Number and type of CPUs
  - Disk Array Information
  - Interconnect Information
  - Amount of Memory Available
  - Number of Virtual AMPs
- Statistics
  - Table Cardinality
  - Column Demographics
  - Collected by user
  - Random Sampling



# Unconditional Parallelism



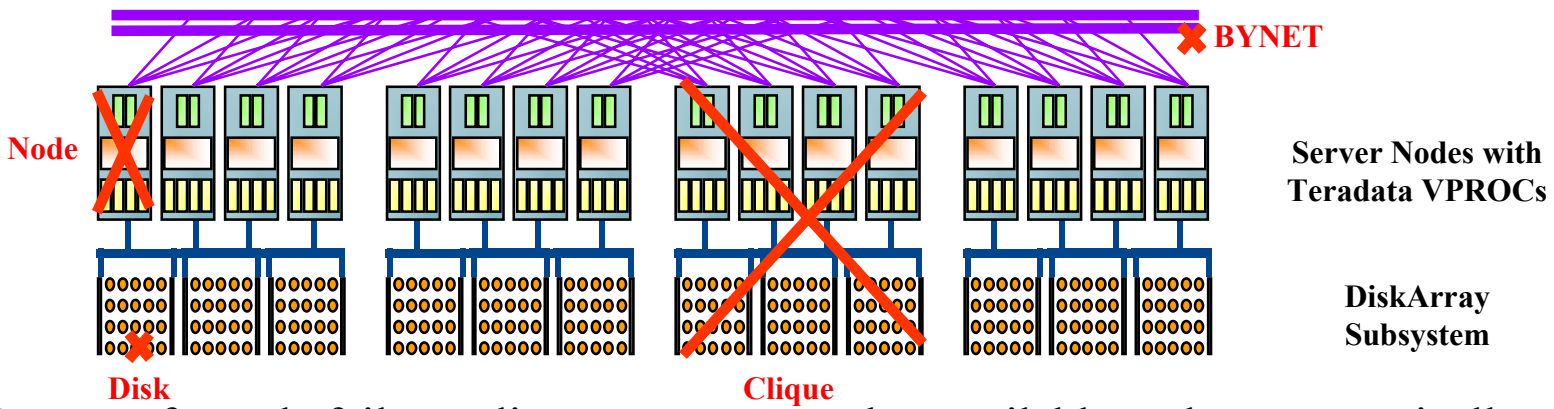

**All** functions of Teradata are done in **parallel**

Teradata
a division of NCR

# Teradata - Availability

## combination of hardware and software features to minimize system outage

| Type of Failure | Worldmark Hardware | Teradata RDBMS |
|---|---|---|
| Power Failure | UPS (redundant), Dual AC | |
| Server-Node Failure | | VPROC-Migration (VAMP, PE) |
| BYNET Network Failure | Redundant BYNET | |
| Single Disk Failure | RAID-1/-5 Disk-Subsystem | |
| Multiple Disk Failure | | Fallback-Option |
| Clique Failure | | Fallback-Option |
| Software / Data Failure | | Permanent Journal (Before Image - Rollback) |




In case of a node failure, clients are connected to available nodes, automatically

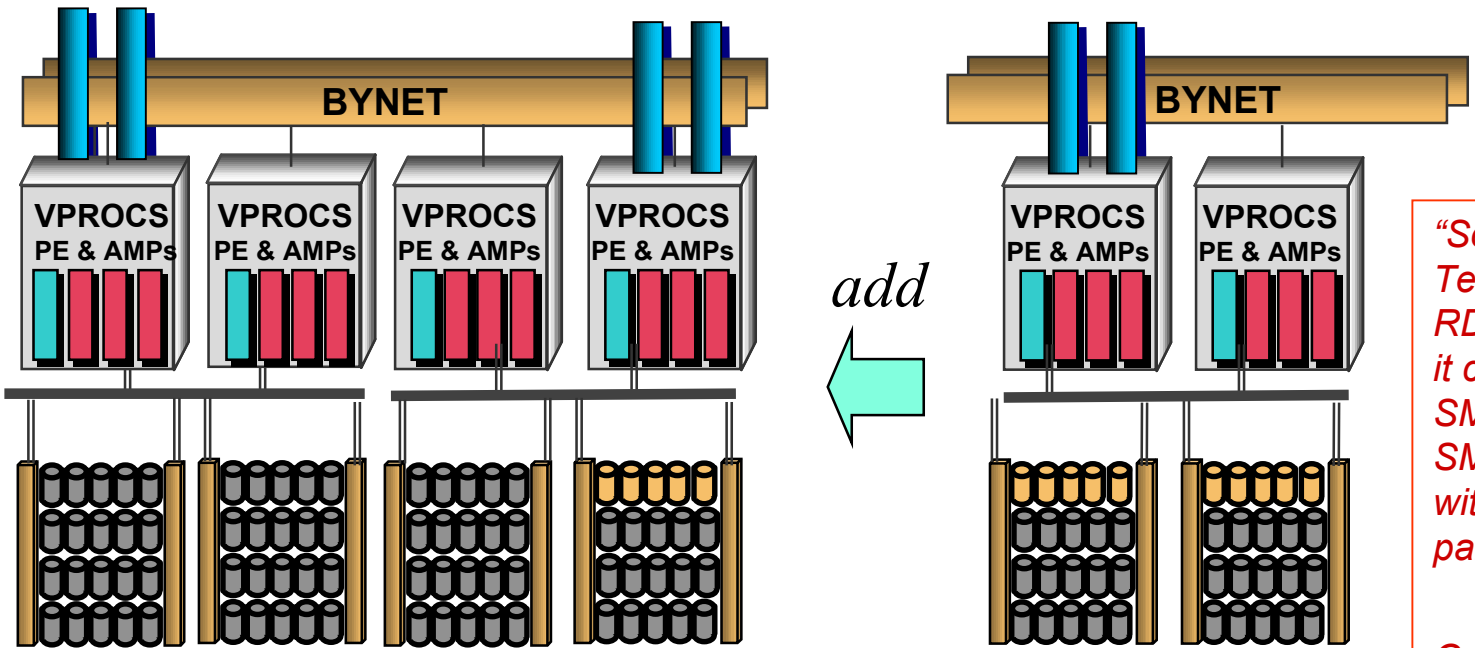
Teradata
a division of NCR

# High Availability - 3 factors

- "planned" System-Outage
  - Software Upgrades, Hardware Expansion, Preventive Maintenance
  - Disk Array & Firmware Upgrades
- "unplanned" System-Outage
  - Disk Array & SCSI problems, Node problems,
  - OS & RDBMS problems
- Database and Data Maintenance Outage
  - Data loading
  - Data purging
  - Partitioning and re-partitioning
  - Data and index reorganisation

# NCR Teradata Scalability

Add **more <u>data</u>, more <u>users</u>, and more <u>subjects</u>**
to your data warehouse with **<u>predictable performance</u>**




*"So far, only NCR's Teradata v.2 RDBMS has proven it can scale up from SMP to hybrid SMP/MPP hardware with internode query parallelism."*

*--*
*Gartner Group*

<u>e.g. double number of nodes:</u>
process twice the workload in same amount of time,
or process same workload in half amount of time

Teradata
a division of NCR

# Teradata MPP Growth

- Production Systems
  - 16 Nodes in 1995 (2 TB)
  - 32 Nodes in 1996 (8 TB)
  - 96 Nodes in 1997 (24 TB)
- Installed 128 Nodes in 1999
  - 512 550 MHz Xeon CPUx
  - 74 TB Spinning Disk
- Enabled 512 Nodes in 1999
- Installed 208 Nodes in 2000
  - 704 x 700 MHz CPUs
  - 704 GB Memory
  - 128 TB Spinning Disk

**Design Limits**

**4,000 Nodes**
**16,000 CPUs**
**16,000 GB Memory**
**4,000 TB Disk**
**(4 Petabytes!)**

# Space Management

- Space allocation is entirely dynamic
  - No tablespaces or journal spaces or any pre-allocation
  - Spool (temp) and tables share space pool, no fixed reserved allocations
- If no cylinder free, combine partial cylinders
  - Dynamic and automatic
  - Background compaction based on tunable threshold
- Quotas control disk space utilization
  - Increase quota (trivial online command) to allow user to use more space

# Complex Query Management Teradata
## Setup & Administration

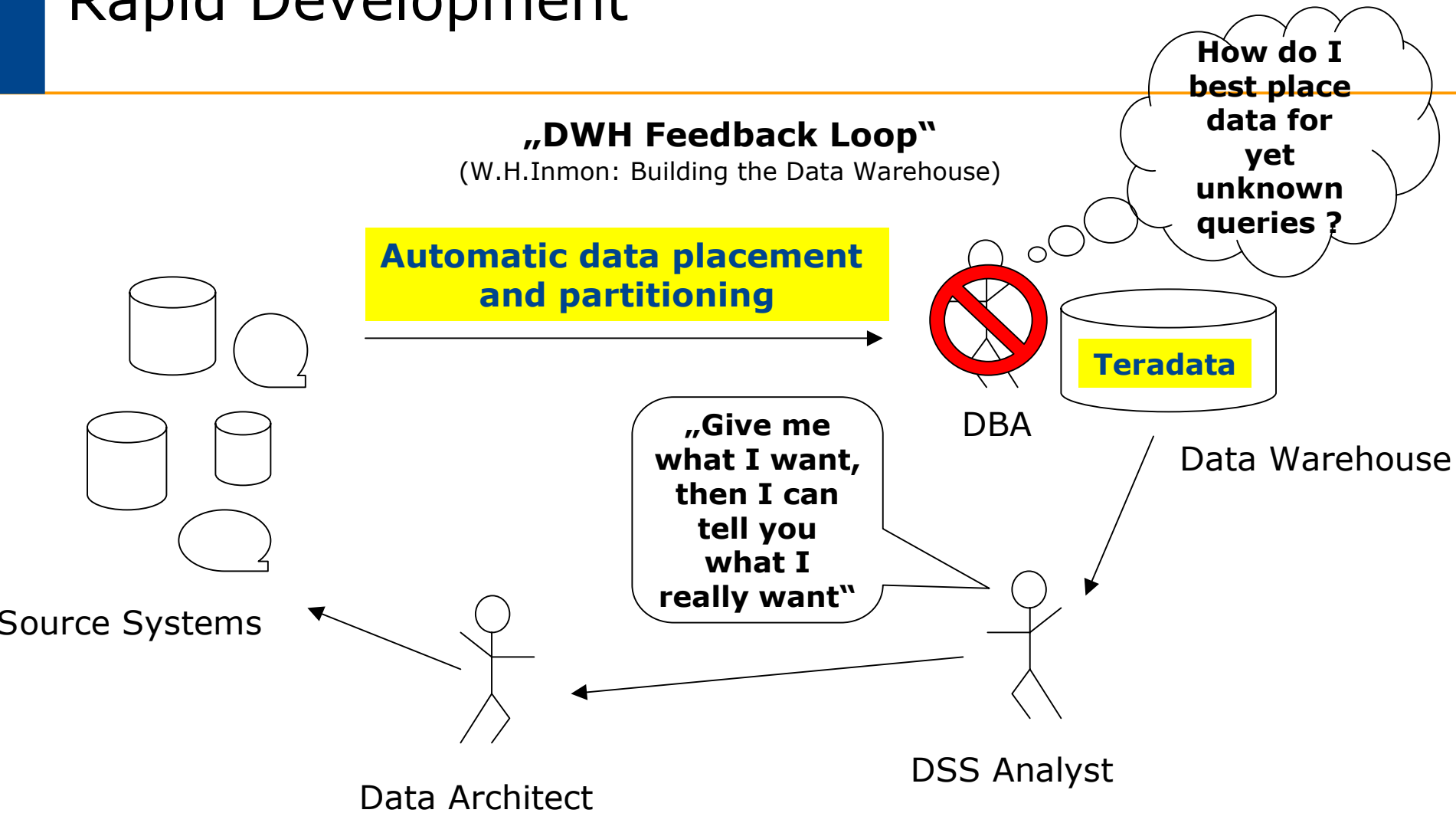
### Architected for unknown and ever-changing DWH queries without administrative intervention or downtimes

**Parallelism -** All work fully shared and coordinated across all parallel Units
**Index re-organisation -** Teradata use Hash-Index, i.e. no index re-organisation or rebuild
**Data re-organisation -**Teradata does not relay on value-sequencing data rows, i.e. no periodic table reorgs (unloads, reloads)
**Space Management -** Teradata share space pool for tables and spool/temp, dynamic and automatic background compaction, i.e. never have to pre-allocate table/index space, ..
**Partitioning -** Teradata use hash map to distribute data, i.e. never have to design, implement, support and re-design partition schemes.
**Schema Redesign -** Teradata supports "alter table", i.e. no unload/reload required
**Minimum reliance** on tuneable parameters, options or choices (no hints require


*"The database administrator (DBA) element of a data warehouse's total cost of ownership is often overlooked during the buying process." -- Gartner Group*

# Rapid Development

**„DWH Feedback Loop"**
(W.H.Inmon: Building the Data Warehouse)




**The shorter the feedback loop, the more successful the warehouse effort.**

Teradata
a division of NCR

# Hard- und Software
*Quotes from Analyst Research*

***Gartner: Data Warehouse Administration: TCO 'Rules of Thumb' (11-2001):***
Our research has shown that Teradata is the most-efficient DBMS in handling complex and diverse workloads compared to DB2 and Oracle. We find that **DB2 requires approximately 25 percent to 30 percent additional processors and that Oracle requires 40 percent additional processors to support similar workloads**

The increase in disk space over raw data space comes from denormalization strategies, indexing, summary tables and mirroring. Some DBMS products (e.g., DB2 UDB and Oracle) **require generous indexing and summary table schemes to obtain acceptable performance** of the user queries

***Gartner: In Search of Oracle Terabyte Data Warehouses (17-09-01):***
This financial institution began to implement a Oracle DW, but weak query performance and low query concurrency caused it to build a highly denormalized data model designed to address the user application (i.e., data mart). At first, this did not seem to be of major concern, but when a second application requirement was uncovered, a new data model and database with redundant data was required.
The financial institution has implemented 15 large data marts (several are more than 1TB), .... **These data marts manage 15TB of raw data, yet only approximately 3TB of this data is unique**.

Teradata
a division of NCR

# Development & Implementation

*Quotes from Analyst Research*

***Gartner: Data Warehouse Administration: TCO 'Rules of Thumb' (11-2001):***

„Because of the sophistication of Teradata's query optimizer, **less DBA effort is required for the data model design** with concern for performance requirements diminished

***Metagroup: Teradata as Tera Firma (12-2001):***

IBM typically uses a low-price strategy to win customers, **but has significantly higher implementation costs**. Additionally, our research shows IBM has often stumbled on delivery, opening the door for Teradata in some of the largest database sales.

***Gartner: In Search of Oracle Terabyte Data Warehouses (17-09-01):***

This financial institution began to implement a (*Oracle*) DW, but weak query performance and low query concurrency caused it to build a highly denormalized data model designed to address the user application (i.e., data mart). At first, this did not seem to be of major concern, but when a second application requirement was uncovered, **a new data model and database with redundant data was required**. These data marts manage 15TB of raw data, yet only approximately 3TB of this data is unique

# DWH Administration
*Quotes from Analyst Research*

***Gartner: Data Warehouse Administration: TCO 'Rules of Thumb' (11-2001):***
With the maturity and capabilities of Teradata to support complex DW implementations, we find that **significantly fewer resources are required to administer and manage the database.** Another area of DBA resource utilization is performing adjustments to table structures and indexes. **Teradata database administrators spend less time** making such data model adjustments based on data usage patterns by users, which is a function that is usually reserved for the most senior-level database administrators

***Gartner: In Search of Oracle Terabyte Data Warehouses (17-09-01):***
The financial institution has implemented 15 large (*Oracle*) data marts (several are more than 1TB), **which 14 full-time database administrators manage** — eight administrators support users and application developers with SQL issues, and the other six provide traditional DBA support (e.g., backup, recovery and disk space management). These data marts manage 15TB of raw data, yet only approximately 3TB of this data is unique.

***McKnight Associates: Choosing a DBMS for Data Warehousing (08-02):***
In Teradata, data placement, free space management, data partitioning, data reorg, index reorg, workspace management, query tuning and workload management are automatic. When you fail to look beyond hardware, software and maintenance when considering TCO, and **fail to look at administration, management and risk issues and the far less need for tuning, you miss out on significant components of TCO**

***Gartner: Averting a Data Warehouse Failure (14-06-02):***
The *(Oracle to Teradata)* **migration led to a reduction in resources**. For example, there were three database administrators managing the nascent Oracle implementation. There is now only one database administrator managing a significantly larger and more widely used system.

Teradata
a division of NCR

# Hidden Cost
*Quotes from Analyst Research*

***Metagroup: Over the Warehouse Walls (22-02-01):***
Exploding data warehouse volumes and increased complexity will not expose platform and database limitations until project failure, **when it is too late.**

**Scaling Oracle 8i or IBM DB2 will require more software skills and DBAs**
(up to 4x that of Teradata)

***Gartner: Averting a Data Warehouse Failure (14-06-02):***
The FSP's first try to implement a data warehouse began in the third quarter of 1999, using an Oracle relational DBMS on a Sun Microsystems 6500 (16-processor) server platform. It was deployed in the first quarter of 2001, but there were performance, scalability, flexibility and cost problems.
The lack of access to the detailed-level data, and instead only to pre-summary data, meant that unforeseen queries could not be easily supported.
Lack of attention to the technology and the selection of the **Oracle DBMS** led to scalability, performance and flexibility issues and **did not deliver business value**.

This led the FSP to a critical decision: reorganize the physical data warehouse into data marts to fit specific applications or look at other choices, such as changing technology.

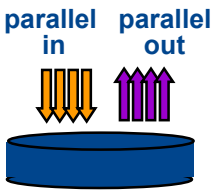
Because of the better performance *(Teradata compared to Oracle),* users have experienced a **tenfold improvement in productivity.**

Teradata
a division of NCR

# Agenda

- **Teradata RDBMS**
  - shared nothing architecture
  - joins, sorts, aggregates, indexes
- **Teradata Tools & Utilities Overview**
  - Load Tools, Query Tools
  - Administrational Tools
- **Teradata Data Load**
  - fastexport, fastload, multiload, bteq
- **Teradata Documentation**

# Teradata Warehouse Tools & Utilities

| | | | |
|---|---|---|---|
| Load & Unload |  | **FastLoad, MultiLoad, FastExport, Teradata TPump, Access Modules, and Teradata Warehouse Builder** | Fast, fail-safe, fully parallel extract and load utilities. The only products that offer automatic check-point restart and one-step load from mainframes. |
| Query & Database Management |  | **Teradata Manager, WinDDI, DBQM/TDQM, QueryMan, Teradata Performance Monitor, Teradata Visual Explain, Teradata System Emulation Tool, BTEQ, ITEQ, PP2** | Real-time, system analysis tools for DBA for ease of monitoring and system managing. |
| Metadata |  | **Meta Data Services** | Framework and method for storing and integrating metadata from various applications and software components in the Teradata environment |
| Storage Management |  | **NetVault, and ASF2 Reader** | High performance backups and restore. Continuous access to and protection of data resources. |
| Open Interface & Connectivity |  | **ODBC, JDBC, OLE DB Provider, XML, Mainframe Channel Connect, CLI, TS/API, CICS, and IMS** | Seamless mainframe integration for direct high-speed load, open client-server connectivity and interfaces for 3rd party access |

Teradata
a division of NCR

# Teradata Warehouse
# Open Connectivity & Interface

**ODBC**
**CLI**
**OLE DB**

**Desktop**
**Windows9x,**
**Windows NT**
**Windows 2000**

**JDBC**
**XML**

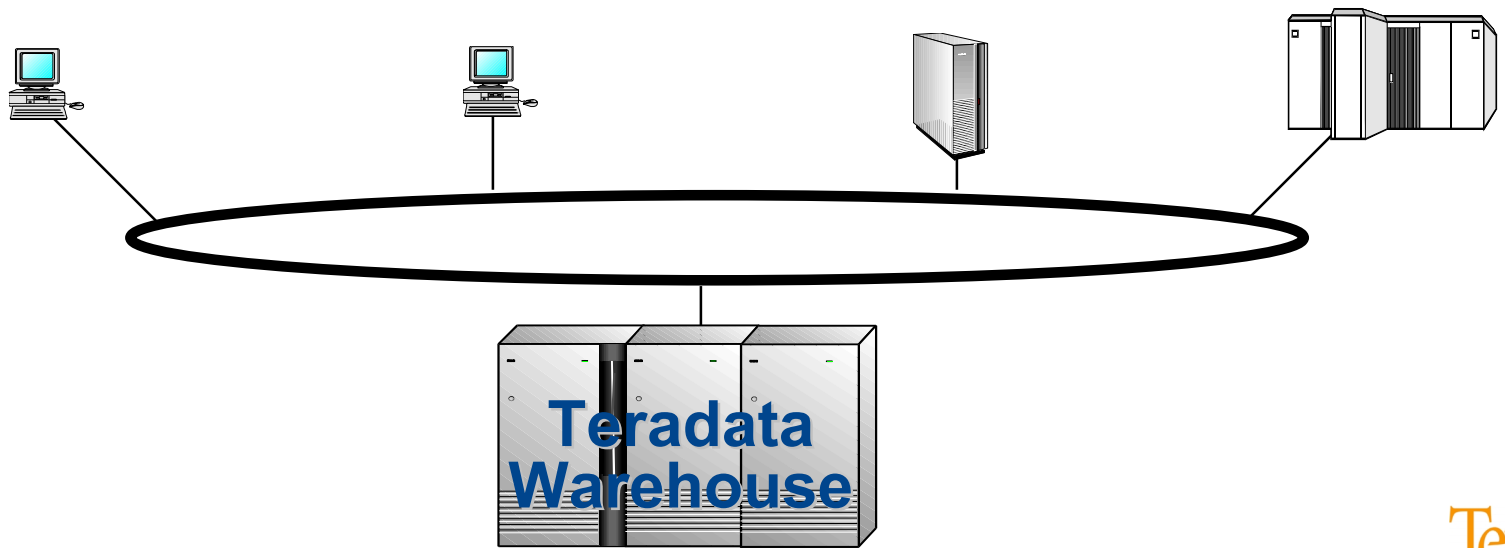
**Internet**
**Network Computers,**
**MS Internet Explorer**
**Netscape, Java**

**ODBC**
**CLI**

**UNIX**
**NCR, Sun,**
**HP, AIX**

**Channel Connect**
**HUTCNS**
**CICS**
**IMS/DC**
**TS/API**

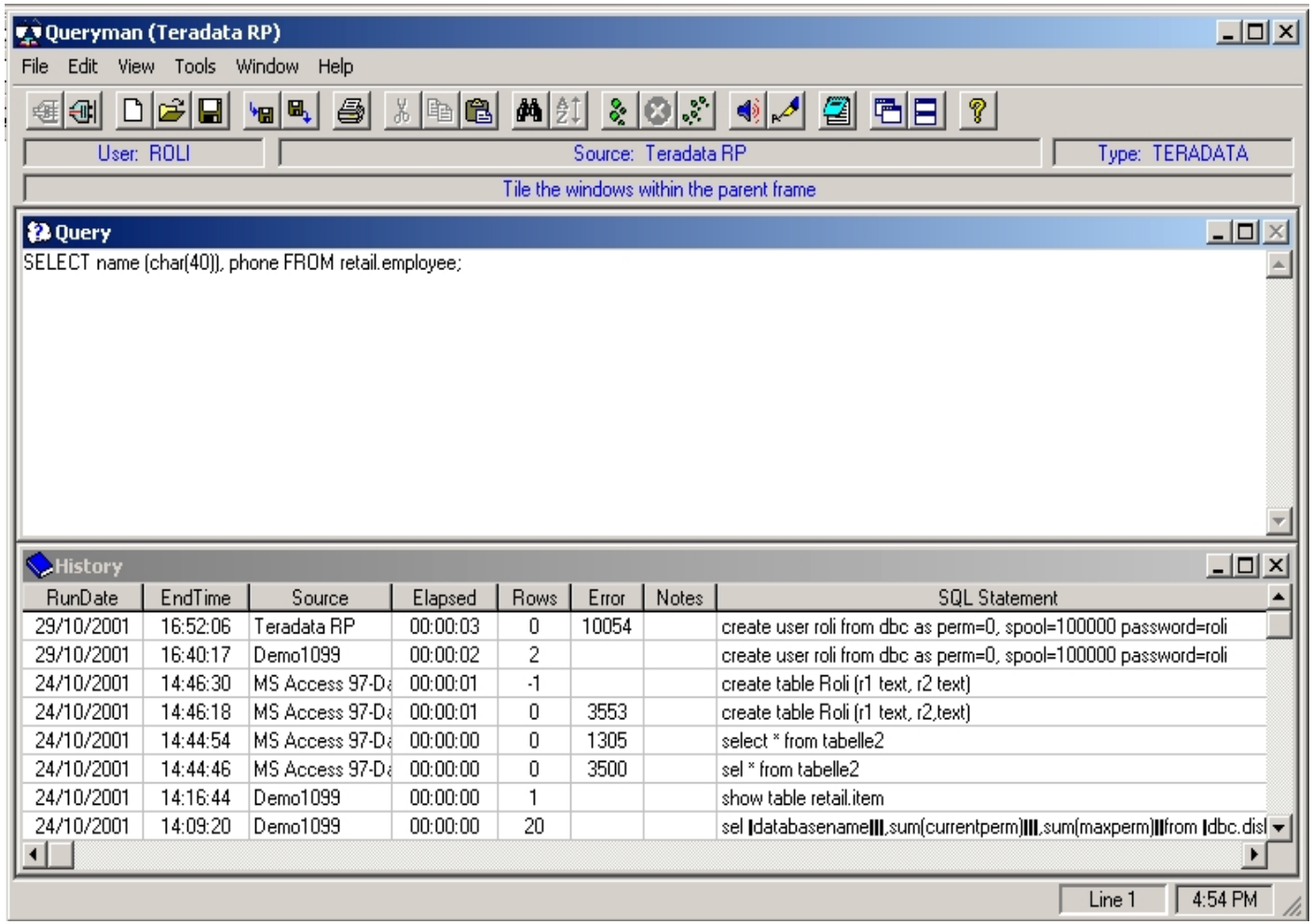
**Mainframes**
**IBM, Bull**
**and more...**




Teradata
a division of NCR

# Teradata Queryman (Application)

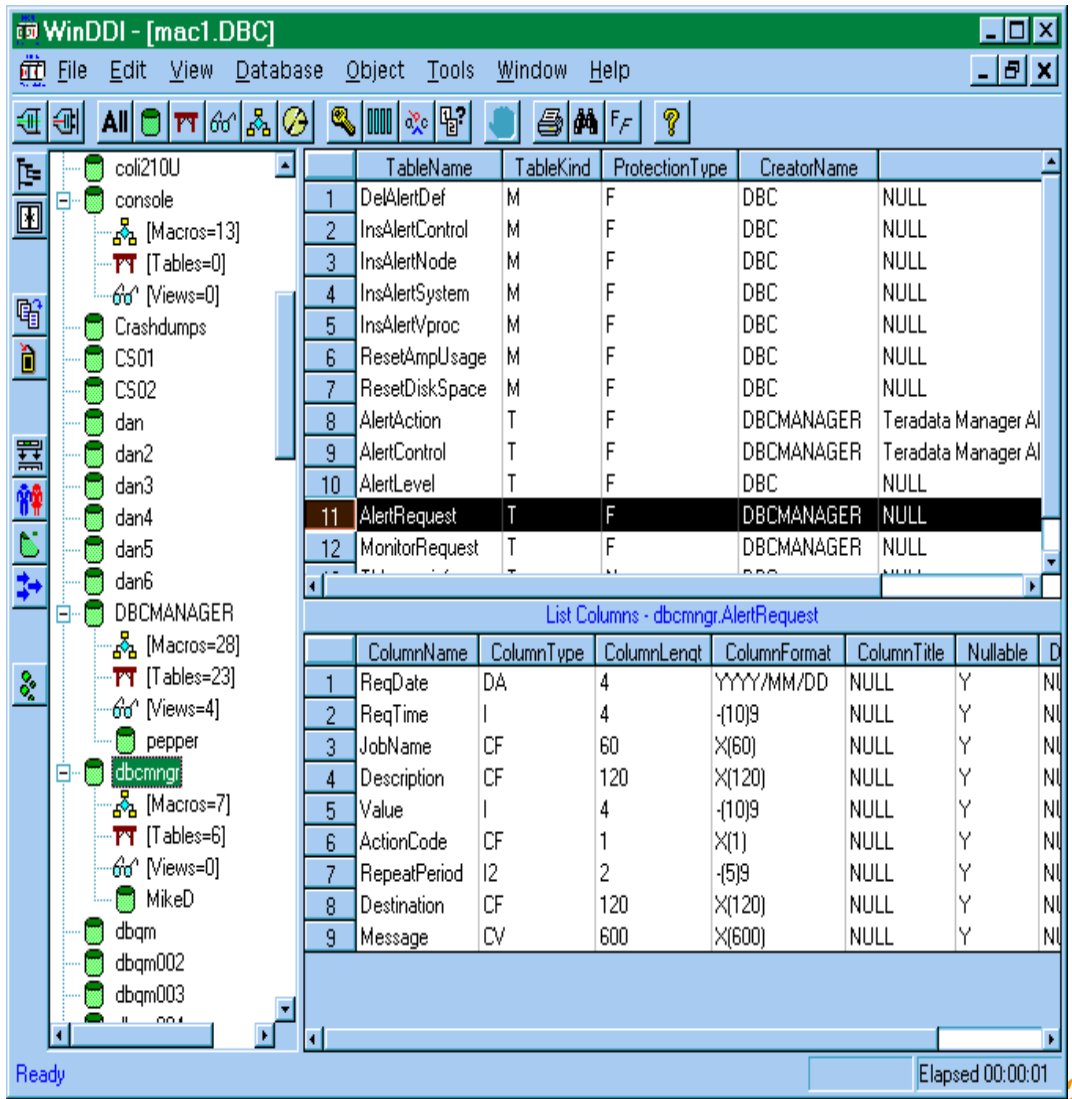
- Exports data from Teradata to PC
- Retains historical records of submitted SQL with timings and status




Teradata
a division of NCR

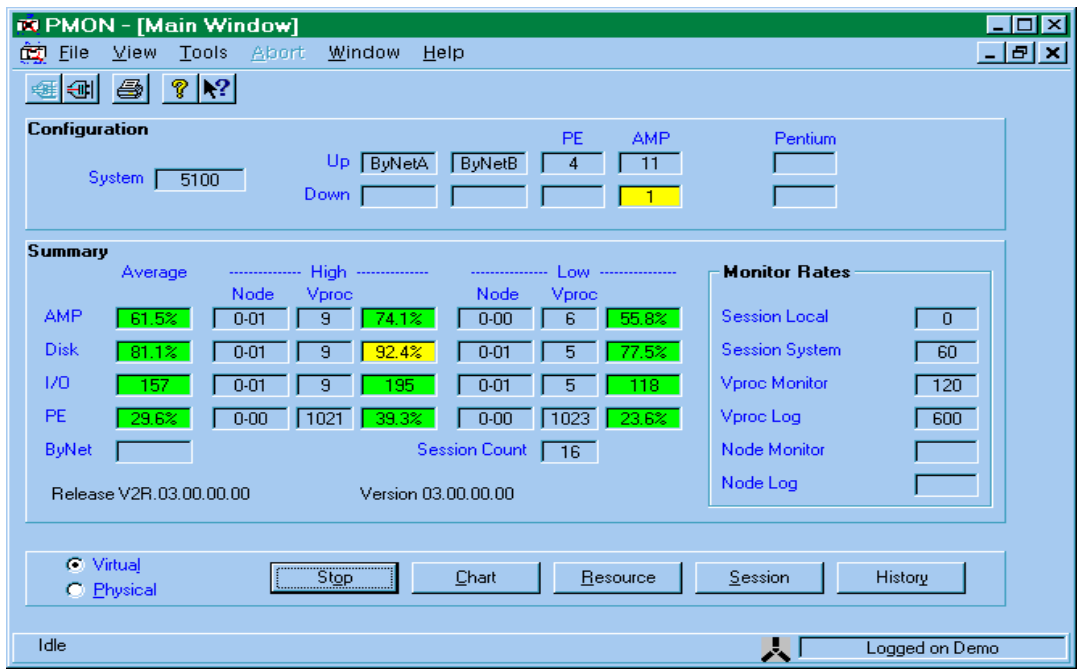
# Teradata Manager Database Administrator (WinDDi)

- Manage the database
- Perform typical database operations with a click of the mouse
- History of SQL operations



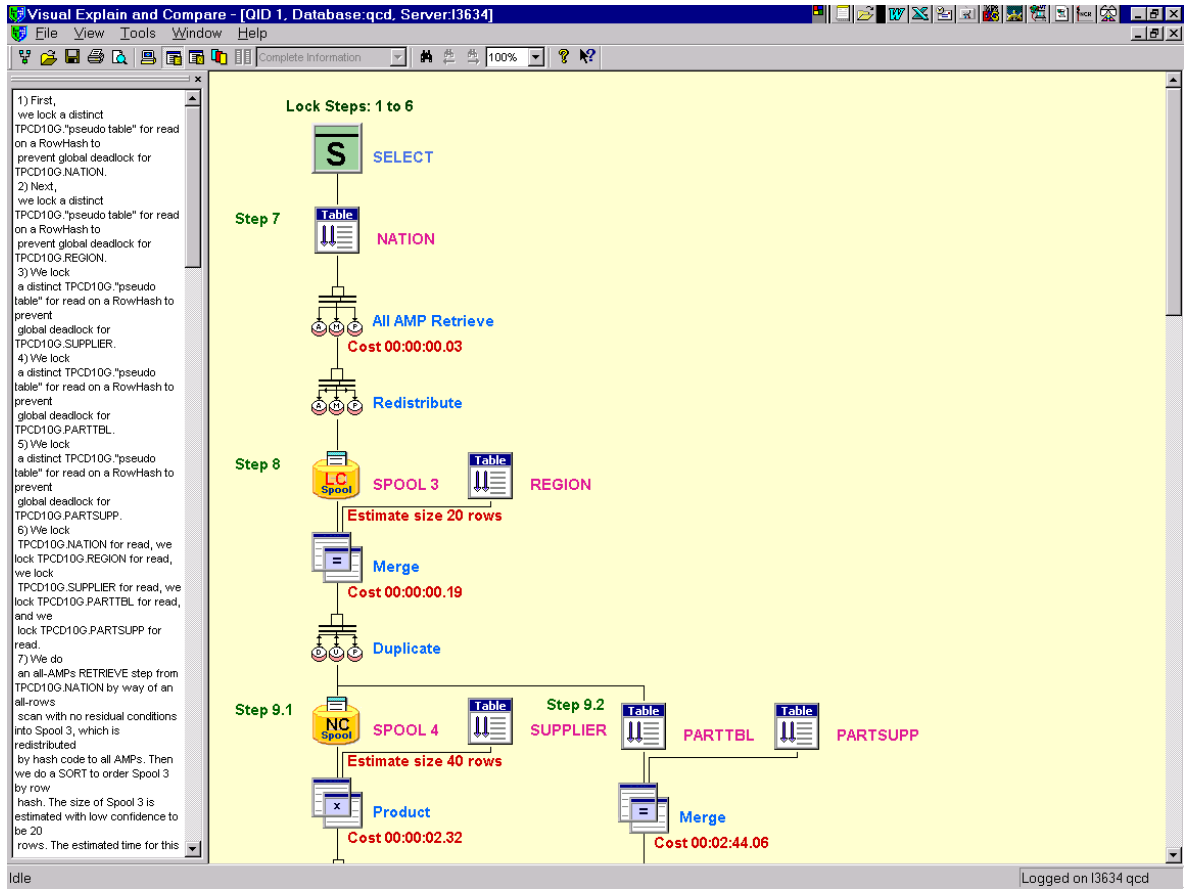
# Performance Monitor

- Performance Monitor (PMON) provides dynamic system status
  - Analysis of Running Queries
  - Configuration Summary
  - Performance Summary
  - Resource Usage
  - Session History

# Teradata Visual Explain

- Easy to use graphical interface describing the execution plan.
- Provides a text description of the optimal execution plan.




Teradata
a division of NCR

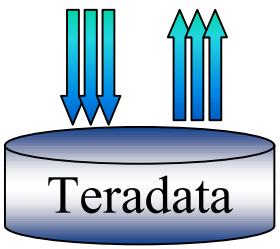
# Agenda

- **Teradata RDBMS**
  - shared nothing architecture
  - joins, sorts, aggregates, indexes
- **Teradata Tools & Utilities Overview**
  - Load Tools, Query Tools
  - Administrational Tools
- **Teradata Data Load**
  - fastexport, fastload, multiload, bteq
- **Teradata Documentation**

# Parallel Load & Unload Utilities

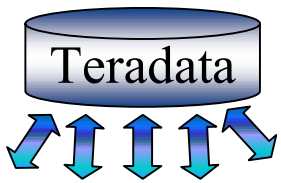
Batch Load & Unload Utilities:

**FastLoad**
**FastExport**
**MultiLoad**



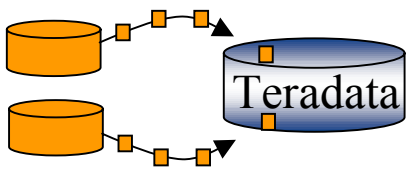

Universal Access

**OLE-DB Access Module**
**Named Pipes Access Module**
**JDBC Access Modules**



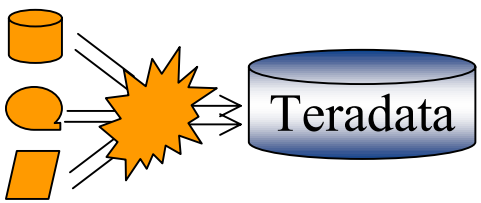

Continuous Load:

**Teradata TPump**




Parallel Load Environment

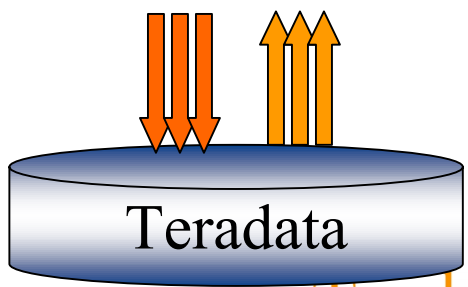
**Teradata Warehouse Builder with**
**Data Connector,**
**Load,**
**Export,**
**Update and**
**Continuous Load**




Evolution of the Teradata Load & Unload Product Suite

# Parallel Load & Unload Utilities

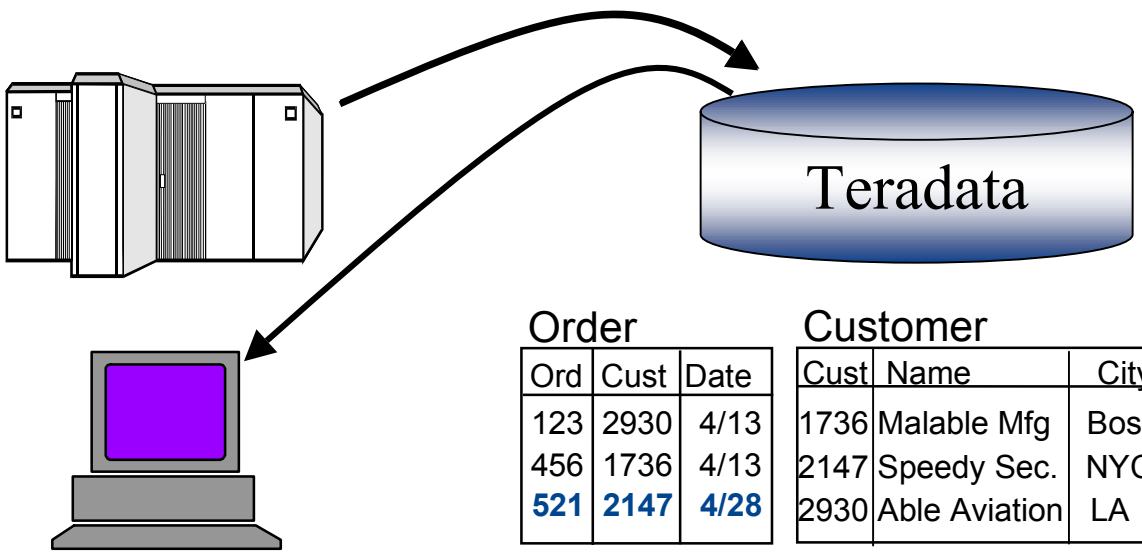
- Teradata Utilities are Fully Parallel
- Teradata Utilities Provide Automatic Checkpoint Restart Capabilities
- Supports seamless data movement from Mainframes, Solaris, UNIX and Windows/NT.
- Data Loads Directly from the Source into the Database
  - No file splitting!
  - No intermediary file transfers!
  - No manual data conversion!




teradata
a division of NCR

# Batch Load & Unload Utilities

- FastLoad
  - High volume initial loads into empty tables
- MultiLoad
  - High volume update, delete and insert of up to 5 tables within a single pass
- FastExport
  - High volume extract of tables and views




Order

| Ord | Cust | Date |
|---|---|---|
| 123 | 2930 | 4/13 |
| 456 | 1736 | 4/13 |
| **521** | **2147** | **4/28** |

Customer

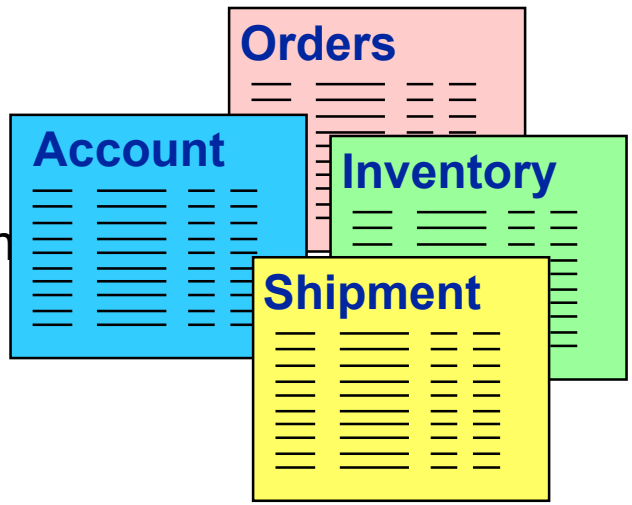
| Cust | Name | City |
|---|---|---|
| 1736 | Malable Mfg | Boston |
| 2147 | Speedy Sec. | NYC |
| 2930 | Able Aviation | LA |

Part

| Part | Qty | Descr |
|---|---|---|
| 345 | **925** | 8.5x11 Paper |
| 360 | 935 | 8.5x14 Paper |
| 421 | **120** | #2 Pencil |

# BTEQ

- Process Commands, Queries and Data
- Route Response to Terminal, Files or Printer
- Usage
  - Batch Job Scripts
  - Report Formatting
  - Ad Hoc Query Tool
  - Database Administration
  - SQL Development and Testing
- Consistent Look and Feel Across Platform



# SQL Example: BTEQ Script

```
DELETE FROM Million_Dollar_Customer  ALL ;
.IF ERRORCODE = 0 THEN  .GOTO TableOK
CREATE TABLE Million_Dollar_Customer
      (Account_Number          INTEGER
      ,Customer_Last_Name      VARCHAR(20)
      ,Customer_First_Name     VARCHAR(15)
      ,Balance_Current         DECIMAL(9,2)) ;

.LABEL TableOK
INSERT INTO Million_Dollar_Customer
SELECT     A.Account_Number
           ,Last_Name
           ,First_Name
           ,Balance_Current
    FROM  Accounts  A
          , Account_Customer B
          , Customer C
    WHERE  Balance_Current GT 1000000
    AND  A. Account_Number  = B.Account_Number
    AND  B. Customer_Number = C.Customer_Number ;
 .IF ACTIVITYCOUNT > 0 THEN .GOTO  Continue
 .QUIT
 .LABEL Continue
```

# SQL Example: BTEQ Script

```
.SET ERRORLEVEL 2168 SEVERITY 4,
            (2173, 3342, 5262) SEVERITY 8
.SET ERRORLEVEL UNKNOWN SEVERITY 16
SELECT
            . . . . . . . . . . . . . FROM. . . . . . . . . . . . . . . . . ;
.IF ERRORLEVEL >= 14  THEN .QUIT 17 ;
```

# SQL Example: Create Table

```
CREATE SET TABLE retail.item ,NO FALLBACK ,
    NO BEFORE JOURNAL,
    NO AFTER JOURNAL
    (
     L_ORDERKEY INTEGER NOT NULL,
     L_PARTKEY INTEGER NOT NULL,
     L_SUPPKEY INTEGER NOT NULL,
     L_LINENUMBER INTEGER NOT NULL,
     L_QUANTITY DECIMAL(15,2) NOT NULL,
     L_EXTENDEDPRICE DECIMAL(15,2) NOT NULL,
     L_DISCOUNT DECIMAL(15,2) NOT NULL,
     L_TAX DECIMAL(15,2) NOT NULL,
     L_RETURNFLAG CHAR(1) CHARACTER SET LATIN NOT CASESPECIFIC NOT NULL,
     L_LINESTATUS CHAR(1) CHARACTER SET LATIN NOT CASESPECIFIC NOT NULL,
     L_SHIPDATE DATE FORMAT 'YYYY-MM-DD' NOT NULL,
     L_COMMITDATE DATE FORMAT 'YYYY-MM-DD' NOT NULL,
     L_RECEIPTDATE DATE FORMAT 'YYYY-MM-DD' NOT NULL,
     L_SHIPINSTRUCT CHAR(25) CHARACTER SET LATIN NOT CASESPECIFIC NOT NULL,
     L_SHIPMODE CHAR(10) CHARACTER SET LATIN NOT CASESPECIFIC NOT NULL,
     L_COMMENT VARCHAR(44) CHARACTER SET LATIN NOT CASESPECIFIC NOT NULL)
PRIMARY INDEX ( L_ORDERKEY )
```

# SQL Example: Create Database

```
CREATE database test1 from dbc as perm=1000000;
```

# SQL Example: Create User

```
CREATE user test1 from dbc as perm=1000000, spool=1000,
password=pw1;
```

# SQL Example: Create Index

```
CREATE INDEX DeptIdx (DeptNo)
ON employee;
```

# SQL Example: Query Databasesize

```
SEL     databasename
        ,sum(currentperm)
        ,sum(maxperm)
FROM    dbc.diskspace
group by 1
order by 1
```

# SQL Example: Query tablesize

```
SEL     tablename
        ,sum(currentperm)
        ,sum(maxperm)
FROM    dbc.diskspace
WHERE databasename = 'retail'
group by 1
order by 1
```

# SQL Example: Accessrights

```
SEL      *
FROM     dbc.allrights
WHERE databasename= 'retail'
```

# SQL Example: Grant Rights

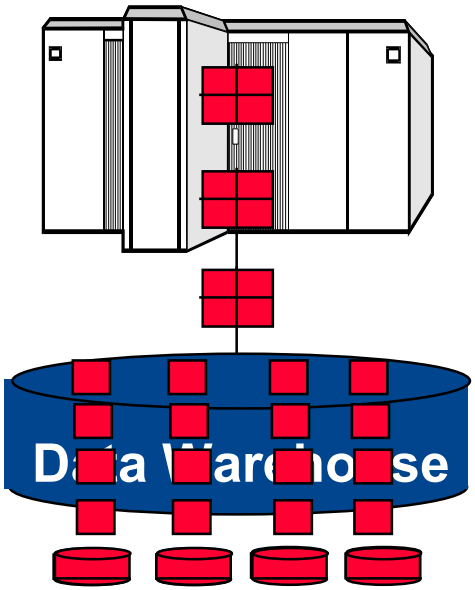
```
GRANT SELECT ON retail TO test1 with grant option
```

# SQL Example: Multistatement Request

```
Insert into retail.new
sel * from retail.test1
;insert into retail.new
sel * from retail.test2
;insert into retail.new
sel * from retail.test3
```

# FastLoad

- High performance initial table load
- Automated parallel data loading




Customer

| Cust | Name | City |
|---|---|---|
| 1736 | Malable Mfg | Boston |
| 2147 | Speedy Sec. | NYC |
| 2930 | Able Aviation | LA |
| 3852 | NCR | Dayton |

Part

| Part | Qty | Descr |
|---|---|---|
| 345 | 875 | 8.5x11 Paper |
| 360 | 935 | 8.5x14 Paper |
| 421 | 0 | #2 Pencil |
| 326 | 0 | Stapler |

Order

| Ord | Cust | Date |
|---|---|---|
| 123 | 2147 | 4/13 |
| 456 | 3852 | 4/13 |
| 789 | 4660 | 4/13 |

Shipped

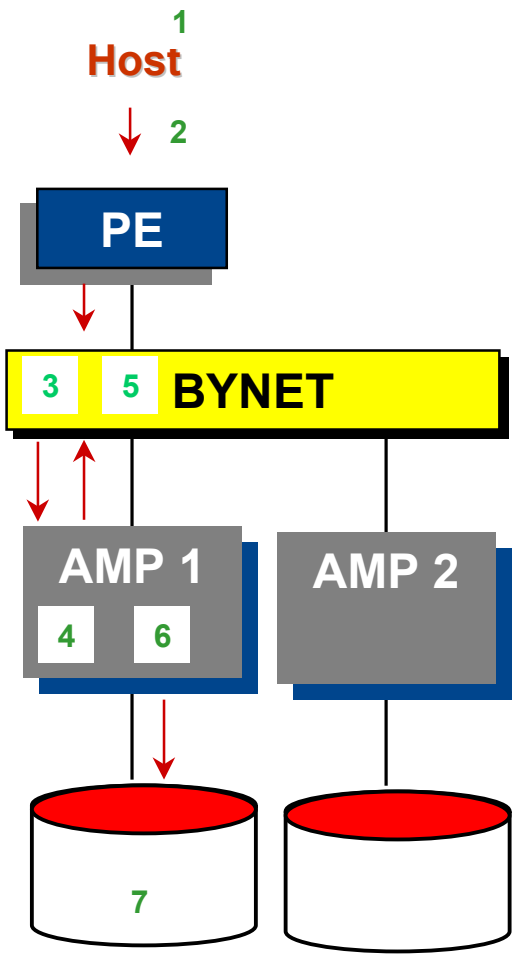
| Ord | Part | Qty | Date |
|---|---|---|---|
| 123 | 360 | 100 | 4/14 |
| 123 | 421 | 144 | 4/14 |
| 123 | 474 | 30 | 4/14 |
| 456 | 474 | 10 | 4/14 |

Location

| State | City |
|---|---|
| CA | LA |
| CA | SD |
| CN | Boston |
| NY | NYC |
| OH | Dayton |

Teradata
a division of NCR

# Two Phases of FastLoad




## Phase 1

- FastLoad uses one SQL session to define AMP steps.
- AMPs hash each record and redistribute them to the AMP responsible for the hash value.
- The PE sends a block to each AMP which stores blocks of unsorted data records.

## Phase 2

- Each AMP sorts the target table, puts the rows into blocks, and writes the blocks to disk.
- Fallback rows are then generated if required.

# Fastload Script

```
LOGON tdpid/username,password;
DROP TABLE Acct;
DROP TABLE AcctErr1;
DROP TABLE AcctErr2;

CREATE TABLE Acct, FALLBACK (
     AcctNum              INTEGER
    ,Number               INTEGER
    ,Street               CHAR(25)
    ,City                 CHAR(25)
    ,State                CHAR(2)
    ,Zip_Code             INTEGER)
UNIQUE PRIMARY INDEX (AcctNum);
LOGOFF;
```

**SETUP**

Create the table, if it dosen't already exist.

# Fastload Script

```
LOGON tdpid/username,password;
BEGIN LOADING Acct
    ERRORFILES AcctErr1, AcctErr2
    CHECKPOINT 100000;

DEFINE   in_AcctNum    (INTEGER)
        ,in_Zip        (INTEGER)
        ,in_Nbr        (INTEGER)
        ,in_Street     (CHAR(25))
        ,in_State      (CHAR(2))
        ,in_City       (CHAR(25))
  FILE=data_infile1;

INSERT INTO Acct VALUES (
 :in_AcctNum
,:in_Nbr
,:in_Street
,:in_City
,:in_Street
,:in_Zip);

END LOADING;
LOGOFF;
```

**FASTLOAD**

Start the utility. Error files must be defined.

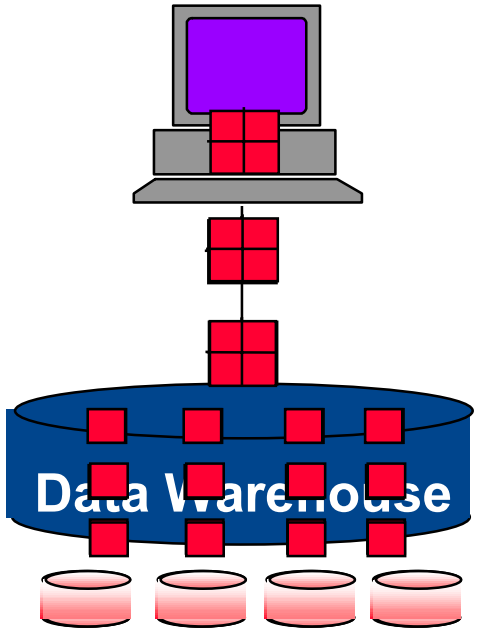
Checkpoint is optional.

INSERT must agree with table definition. **Phase 1 begins.** Unsorted blocks are written to disk.

**Phase 2 begins** with END LOADING. Sorting and writing blocks to disk.

# FastExport

- High speed utility for exporting data
- Presorts and merges data




Customer

| Cust | Name | City |
|---|---|---|
| 1736 | Malable Mfg | Boston |
| 2147 | Speedy Sec. | NYC |
| 2930 | Able Aviation | LA |
| 3852 | NCR | Dayton |

Part

| Part | Qty | Descr |
|---|---|---|
| 345 | 875 | 8.5x11 Paper |
| 360 | 935 | 8.5x14 Paper |
| 421 | 0 | #2 Pencil |
| 326 | 0 | Stapler |

Order

| Ord | Cust | Date |
|---|---|---|
| 123 | 2147 | 4/13 |
| 456 | 3852 | 4/13 |
| 789 | 4660 | 4/13 |

Shipped

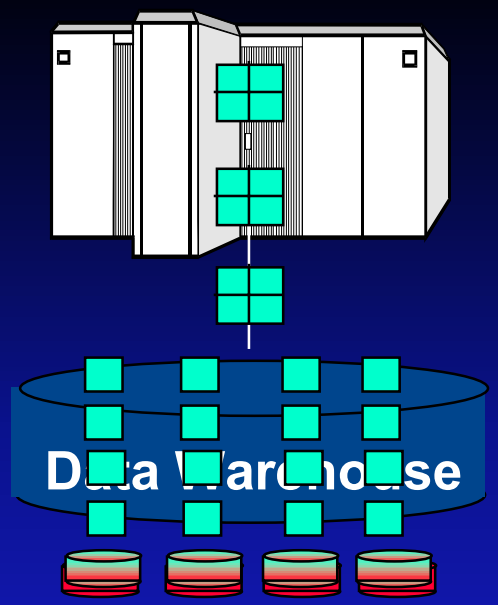
| Ord | Part | Qty | Date |
|---|---|---|---|
| 123 | 360 | 100 | 4/14 |
| 123 | 421 | 144 | 4/14 |
| 123 | 474 | 30 | 4/14 |
| 456 | 474 | 10 | 4/14 |

# FastExport Script

```
.LOGTABLE RestartLog_fxp;
.RUN       FILE logon ;
.BEGIN     EXPORT
           SESSIONS 4 ;
.EXPORT  OUTFILE  custacct_data;
SELECT     A.Account_Number
           ,  C.Last_Name
           ,  C.First_Name
           ,AC.Balance_Current
   FROM      Accounts A
           ,   Customer C
           ,   Accounts_Customer AC
  WHERE      A.Account_Number = AC.Account_Number
       AND   C.Customer_Number = AC.Customer_Number
       AND   City              = 'Los Angeles'
       AND   Zip_Code     = 90066
       ORDER BY 1 ;
.END EXPORT ;
.LOGOFF ;
```

# MultiLoad

- High performance bulk operations
- Processes large volumes of data




Customer

| Cust | Name | City |
|---|---|---|
| 1736 | Malable Mfg | Boston |
| 2147 | Speedy Sec. | NYC |
| 2930 | Able Aviation | LA |
| 3852 | NCR | Dayton |

Part

| Part | Qtv | Descr |
|---|---|---|
| 345 | 651 | 8.5x11 Paper |
| 360 | 780 | 8.5x14 Paper |
| 421 | 250 | #2 Pencil |
| 326 | 50 | Stapler |

Order

| Ord | Cust | Date |
|---|---|---|
| 123 | 2147 | 4/13 |
| 456 | 3852 | 4/13 |
| 789 | 4660 | 4/13 |
| 101 | 3951 | 4/16 |

Shipped

| Ord | Part | Qty | Date |
|---|---|---|---|
| 123 | 360 | 100 | 4/14 |
| 123 | 421 | 144 | 4/14 |
| 123 | 474 | 30 | 4/14 |
| 456 | 474 | 10 | 4/14 |
| 101 | 360 | 500 | 4/16 |

# Multiload: 5 Phases of IMPORT Task

| | |
|---|---|
| **Preliminary** | **Basic set up** |
| **DML Transaction** | **Send the DML steps to the AMPs** |
| **Acquisition** | **Send the input data to the AMPs** |
| **Application** | **Apply the input data to appropriate table(s)** |
| **Cleanup** | **Basic clean up** |

# Multiload Script

```
.LOGTABLE Logtable001_ml;
.LOGON  tdp3/user2,tyler;
.BEGIN MLOAD TABLES  Employee, Employee_History;
.LAYOUT Employee_Trans;
        .FILLER  in_Transcode   1  CHAR(3);
        .FIELD   in_EmpNo       *  SMALLINT;
        .FIELD   in_DeptNo      *  SMALLINT;
        .FIELD   in_Salary      *  DECIMAL (8,2);
```

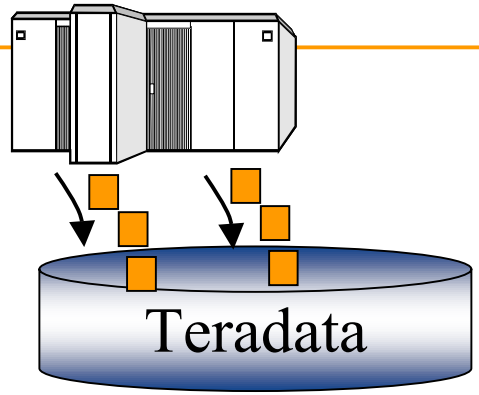
# Multiload Script (cont'd)

```
.DML  LABEL  Payroll
        DO INSERT FOR MISSING UPDATE ROWS ;
        UPDATE  Employee  SET  Salary = :in_Salary
                WHERE  EmpNo = :in_EmpNo;
        INSERT  INTO  Employee  (EmpNo,  Salary)
                VALUES  (:in_EmpNo,  :in_Salary);
.DML  LABEL  Terminate ;
        DELETE  FROM  Employee  WHERE  EmpNo = :in_EmpNo;
        INSERT INTO Employee_History  (EmpNo, DeptNo)
                VALUES  (:in_EmpNo,  :in_DeptNo);
```

# Multiload Script (cont'd)

```
.IMPORT  INFILE  infile1
         LAYOUT  Employee_Trans
         APPLY  Payroll  WHERE  in_Transcode = 'PAY'
         APPLY  Terminate  WHERE  in_Transcode = 'DEL';
.END  MLOAD;
.LOGOFF;
```

# Teradata Tpump: Continuous Load




- Teradata load utility that allows continuous updates
  - Continuous updates of data results in more accurate, timely data
- Updates without locking tables
  - Enables concurrent loads to a single table
  - Allows users to access the table during updates
- Easy to use graphical script builder
  - Build ETL applications faster
- Alternative to MultiLoad for low-batch maintenance of large databases

Order

| Ord | Cust | Date |
|---|---|---|
| 123 | 2930 | 4/13 |
| 456 | 1736 | 4/13 |
| 521 | 2147 | 4/28 |

Customer

| Cust | Name | City |
|---|---|---|
| 1736 | Malable Mfg | Boston |
| 2147 | Speedy Sec. | NYC |
| 2930 | Able Aviation | LA |

Part

| Part | Qty | Descr |
|---|---|---|
| 345 | 925 | 8.5x11 Paper |
| 360 | 970 | 8.5x14 Paper |
| 421 | 120 | #2 Pencil |

Teradata
a division of NCR

# Agenda

- **Teradata RDBMS**
  - shared nothing architecture
  - joins, sorts, aggregates, indexes
- **Teradata Tools & Utilities Overview**
  - Load Tools, Query Tools
  - Administrational Tools
- **Teradata Data Load**
  - fastexport, fastload, multiload, bteq
- **Teradata Documentation**

Acrobat Reader - [Teradata Library Contents]

File Edit Document View Window Help

Bookmarks Thumbnails



Welcome to the

# Teradata® Library

User Documentation for V2R4.1 and TUF 6.1

Is this your first time using this library?
Click <here> for important setup information.

100% 1 of 3 7 x 5 in

Online Documentation

www.info.ncr.com

Case Studies / Reports / White Papers

www.teradata.com

Teradata
a division of NCR

# Questions