

InfoSphere™



Three courses of DataStage, with a side order of Teradata

Stewart Hanna
Product Manager

Agenda

- **Platform Overview**
- **Appetizer - Productivity**
- **First Course – Extensibility**
- **Second Course– Scalability**
- **Desert – Teradata**



Agenda

- **Platform Overview**
- Appetizer - Productivity
- First Course – Extensibility
- Second Course– Scalability
- Desert – Teradata



Accelerate to the Next Level

Unlocking the Business Value of Information for Competitive Advantage

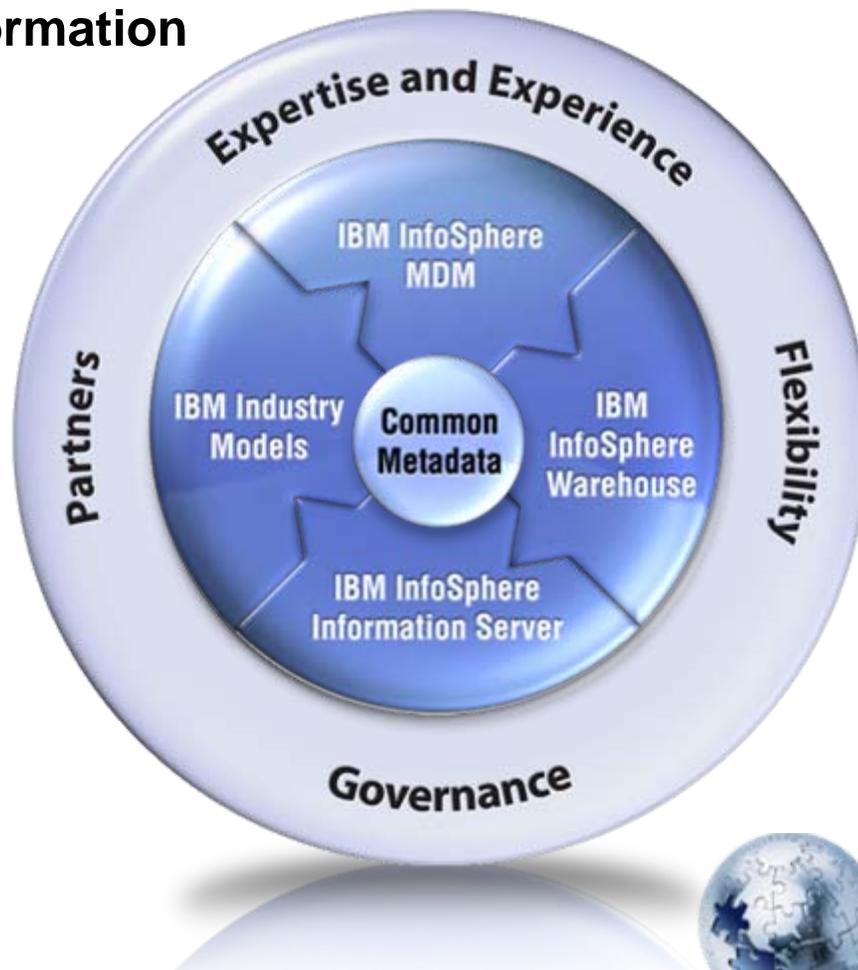


The IBM InfoSphere Vision

An Industry Unique Information Platform

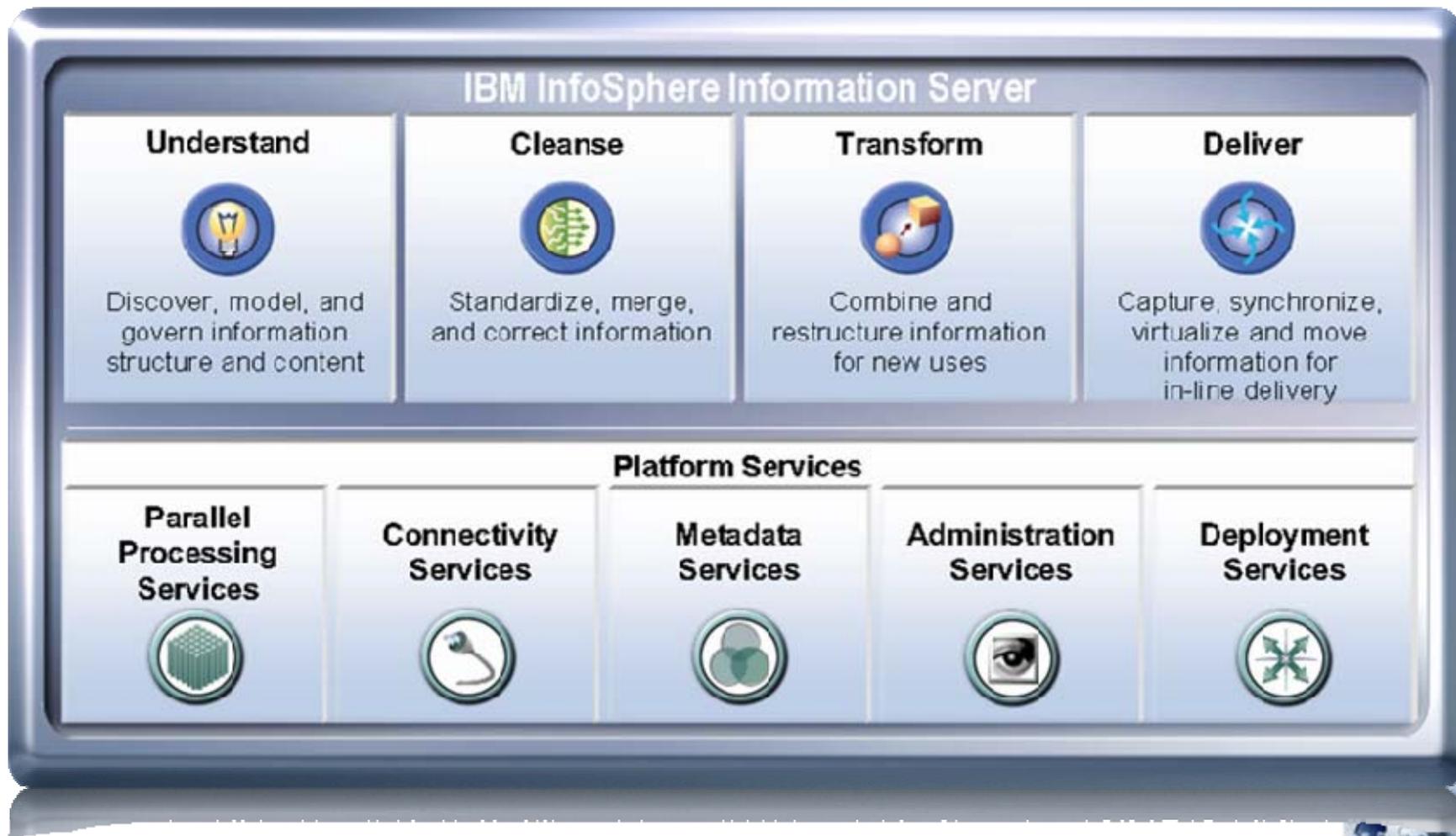
- Simplify the delivery of Trusted Information
- Accelerate client value
- Promote collaboration
- Mitigate risk
- Modular but Integrated
- Scalable – Project to Enterprise

InfoSphere™



IBM InfoSphere Information Server

Delivering information you can trust



InfoSphere DataStage

- Provides codeless visual design of data flows with hundreds of built-in transformation functions
 - Optimized reuse of integration objects
 - Supports batch & real-time operations
 - Produces reusable components that can be shared across projects
- Complete ETL functionality with metadata-driven productivity
- Supports team-based development and collaboration
- Provides integration from across the broadest range of sources



Developers



Architects

Transform

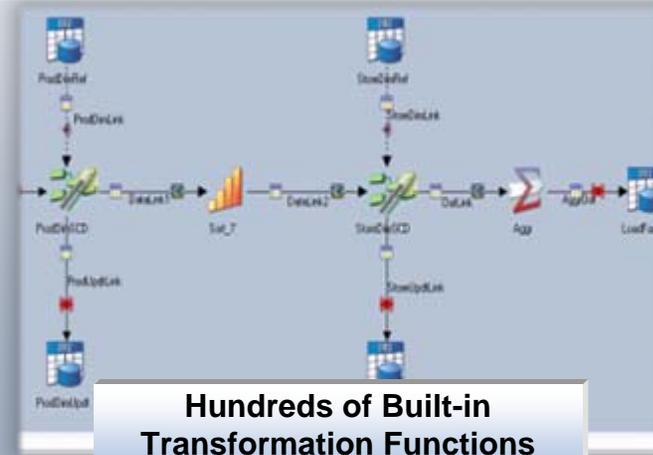


Deliver



InfoSphere DataStage®

Transform and aggregate any volume of information in batch or real time through visually designed logic



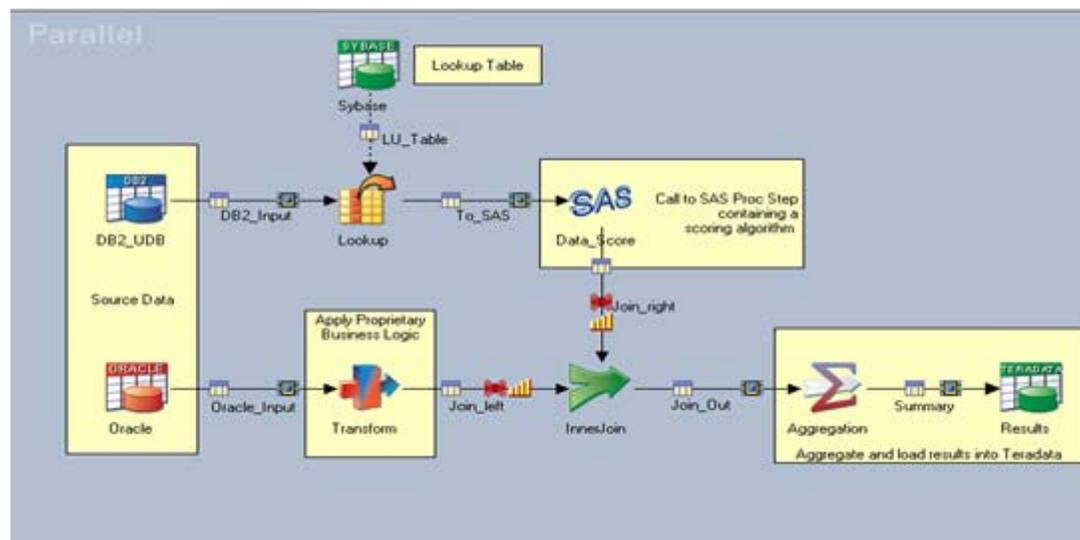
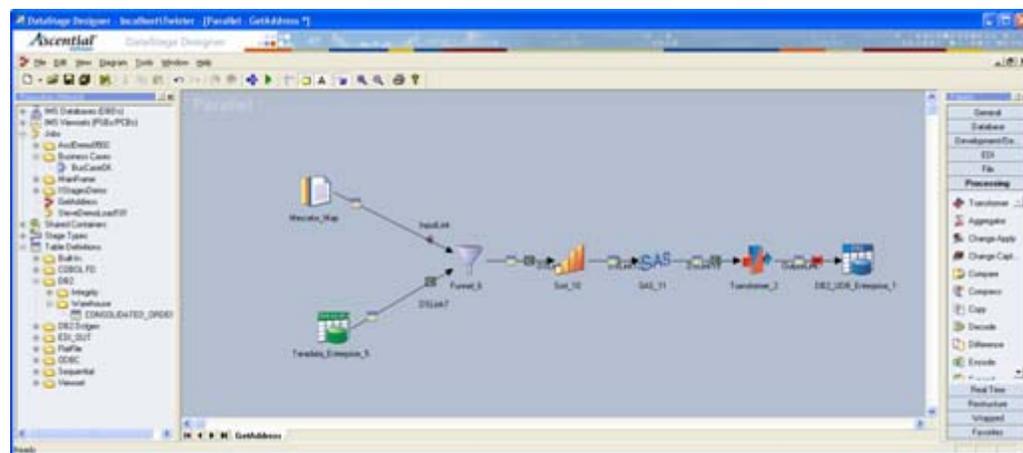
Agenda

- Platform Overview
- **Appetizer - Productivity**
- First Course – Extensibility
- Second Course– Scalability
- Desert – Teradata



DataStage Designer

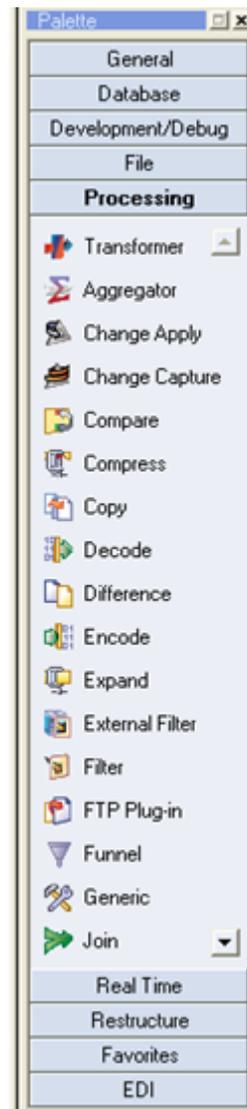
- Complete development environment
- Graphical, drag and drop top down design metaphor
- Develop sequentially, deploy in parallel
- Component-based architecture
- Reuse capabilities



Transformation Components

- **Over 60 pre-built components available including**
 - Files
 - Database
 - Lookup
 - Sort, Aggregation, Transformer
 - Pivot, CDC
 - Join, Merge
 - Filter, Funnel, Switch, Modify
 - Remove Duplicates
 - Restructure stages

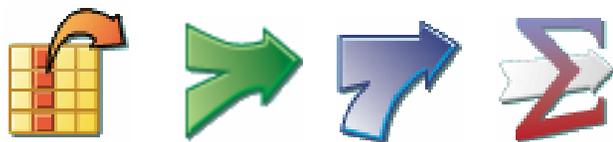
Sample of
Stages
Available



Some Popular Stages



- Usual ETL Sources & Targets:
 - *RDBMS, Sequential File, Data Set*



- Combining Data:
 - *Lookup, Joins, Merge*
 - *Aggregator*



- Transform Data:
 - *Transformer, Remove Duplicates*



- Ancillary:
 - *Row Generator, Peek, Sort*



Deployment

- Easy and integrated job movement from one environment to the other
- A deployment package can be created with any first class objects from the repository. New objects can be added to an existing package.
- The description of this package is stored in the metadata repository.
- All associated objects can be added in like files outside of DS and QS like scripts.
- Audit and Security Control

The screenshot shows the IBM Information Server interface. On the left is a 'Repository' tree with folders like 'size', 'slsk', 'DataStage', 'DataMask', and 'ProjectOne'. On the right is the 'Package' view, which contains a table of selected items.

Target View	Source Path	Description
Jobs		
BTW		
BW_PORCDate	SIZZE:iod2007_b\Jobs\BTW	
BW_PO_CDC	SIZZE:iod2007_b\Jobs\BTW	
BW_PORC	SIZZE:iod2007_b\Jobs\BTW	
CDW		
LDW_Products	SIZZE:iod2007_b\Jobs\CDW	
Order_Conditions	SIZZE:iod2007_b\Jobs\CDW	
jb00		
final		
jb00CreateTes	SIZZE:iod2007_b\Jobs\jb00\final	
jb02MergeRot	SIZZE:iod2007_b\Jobs\jb00\final	
DS_demo		
BankDemoXML	SIZZE:iod2007_b\DS_demo	

```

import          : Import from a file.
build package   : Build a Deployment Package.
deploy package  : Deploy a Deployment Package.
create package  : Create a deployment package.
delete package  : Delete a deployment package.
create folder   : Create a Folder.
delete          : Delete a DataStage asset.

To list Command_Options, enter: istool Command -help

Possible values for: Options
Long Name:          Short Name:      Description:
-help              -h           : print command usage
-verbose           -v           : display progress
-silent            -s           : silence command output

(istool>
    
```

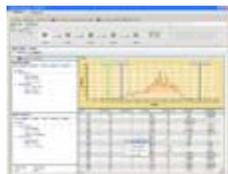
Role-Based Tools with Integrated Metadata



Business Users



Subject Matter Experts



Architects



Data Analysts



Developers



DBAs



Unified Metadata Management



Design Operational

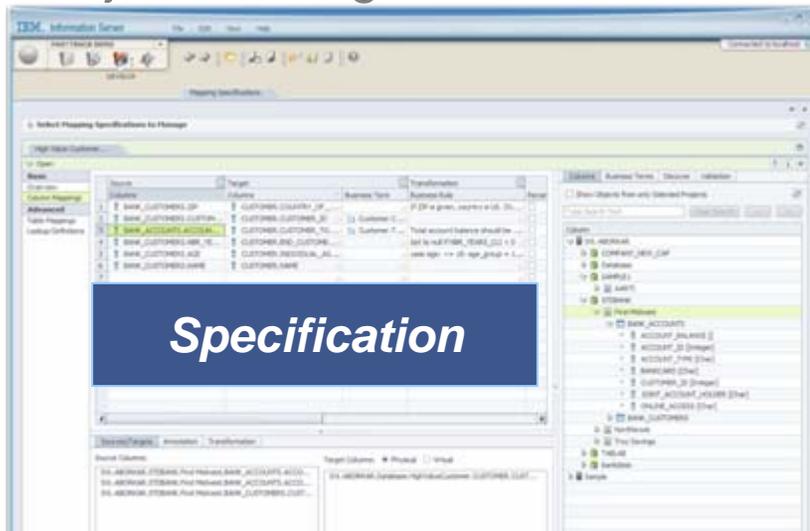
- Simplify Integration
- Increase trust and confidence in information
- Facilitate change management & reuse
- Increase compliance to standards



InfoSphere FastTrack

To reduce Costs of Integration Projects through Automation

- Business analysts and IT collaborate in context to create project specification
- Leverages source analysis, target models, and metadata to facilitate mapping process
- Auto-generation of data transformation jobs & reports
- Generate historical documentation for tracking
- Supports data governance



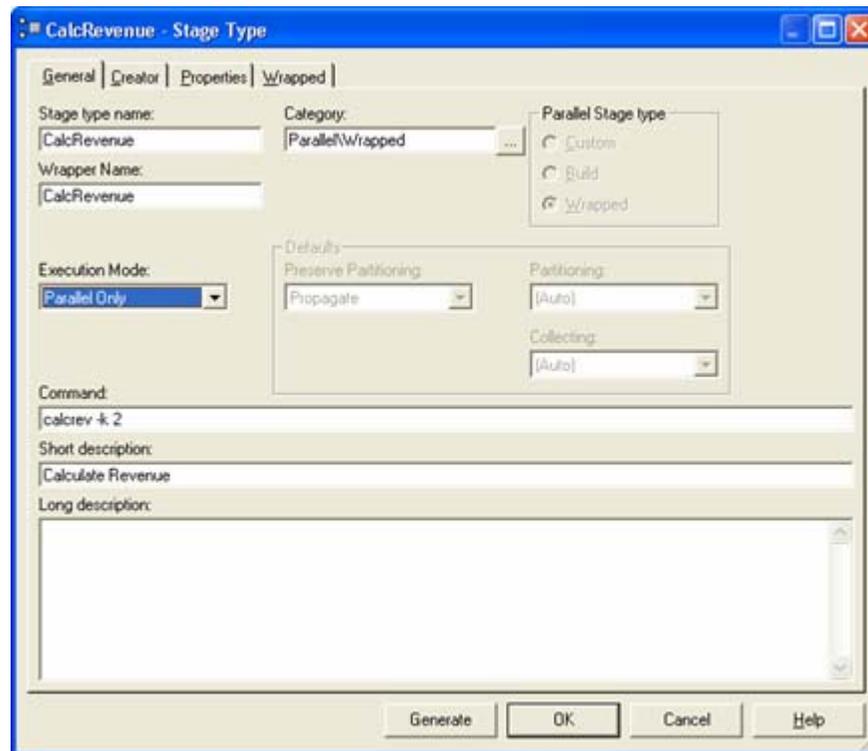
Agenda

- Platform Overview
- Appetizer - Productivity
- **First Course – Extensibility**
- Second Course– Scalability
- Desert – Teradata



Extending DataStage - Defining Your Own Stage Types

- **Define your own stage type to be integrated into data flow**
- **Stage Types**
 - **Wrapped**
 - Specify a OS command or script
 - Existing Routines, Logic, Apps
 - **BuildOp**
 - Wizard / Macro driven Development
 - **Custom**
 - API Development
- **Available to all jobs in the project**
- **All meta data is captured**



Building “Wrapped” Stages

- **In a nutshell:**
- **You can “wrap” a legacy executable:**
 - *binary,*
 - *Unix command,*
 - *shell script*

... and turn it into a bona fide DataStage stage capable, among other things, of parallel execution,

... as long as the legacy executable is

- *amenable to data-partition parallelism*
 - *no dependencies between rows*
- *pipe-safe*
 - *can read rows sequentially*
 - *no random access to data, e.g., use of fseek()*



Building BuildOp Stages

- **In a nutshell:**
 - *The user performs the fun, glamorous tasks: encapsulate business logic in a custom operator*
 - *The DataStage wizard called “buildop” automatically performs the unglamorous, tedious, error-prone tasks: invoke needed header files, build the necessary “plumbing” for a correct and efficient parallel execution.*



Major difference between BuildOp and Custom Stages

- Layout *interfaces* describe what columns the stage:
 - needs for its inputs
 - creates for its outputs
- Two kinds of interfaces: *dynamic* and *static*
- **Dynamic**: adjusts to its inputs automatically
- Custom Stages can be **Dynamic or Static**
- *IBM DataStage supplied Stages are dynamic*
- **Static**: expects input to contain columns with specific names and types
- BuildOp are **only** Static



Agenda

- Platform Overview
- Appetizer - Productivity
- First Course – Extensibility
- **Second Course– Scalability**
- Desert – Teradata



Scalability is important everywhere

It take me 4 1/2 hours to wash, dry and fold 3 loads of laundry (1/2 hour for each operation)

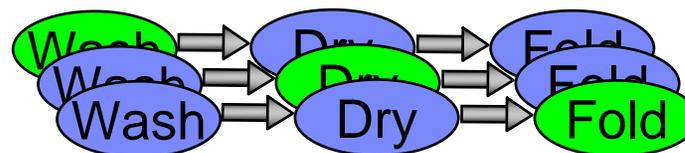
Sequential Approach (4 1/2 Hours)

- Wash a load, dry the load, fold it
- Wash-dry-fold
- Wash-dry-fold



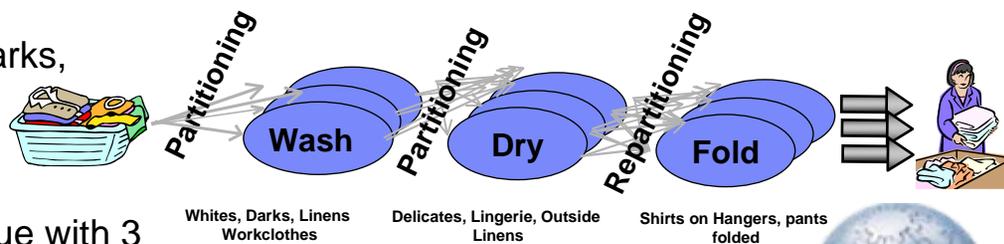
Pipeline Approach (2 1/2 Hours)

- Wash a load, when it is done, put it in the dryer, etc
- Load washing while another load is drying, etc

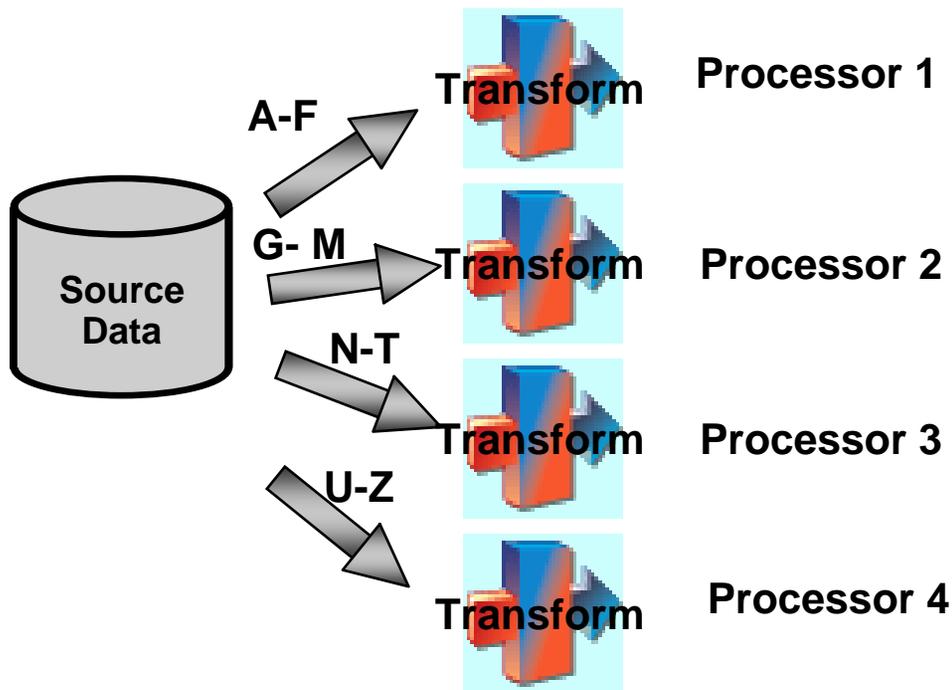


Partitioned Parallelism Approach (1 1/2 Hours)

- Divide the laundry into different loads (whites, darks, linens)
- Work on each piece independently
- 3 Times faster with 3 Washing machines, continue with 3 dryers and so on



Data Partitioning

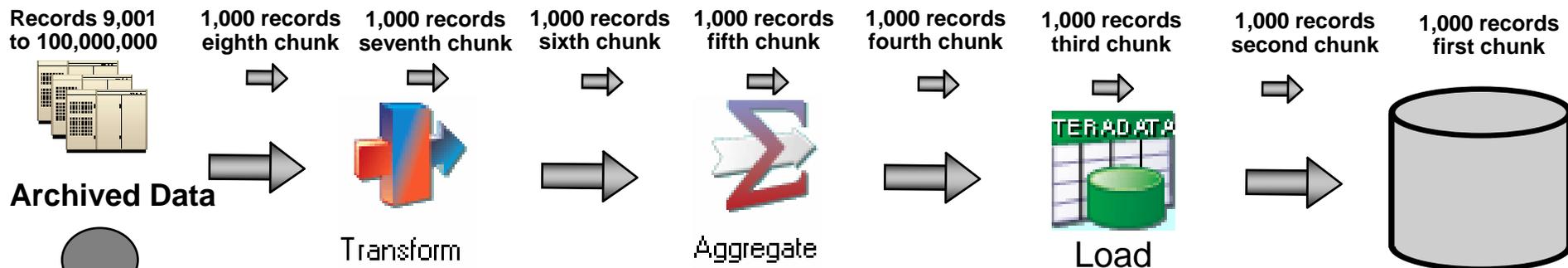


The screenshot shows the 'Join_19 - Join' configuration window. The 'General' tab is active, showing 'Execution mode' set to 'Default (Parallel)', 'Concurrency mode' set to '(Auto)', and 'Preserve partitioning' set to 'Default (Propagate)'. The 'Partitioning / Collecting' tab is also visible, showing 'Partition type' set to 'Hash'. A list of partitioning options is shown: (Auto), DB2, Entize, Hash, Modulus, Random, Range, Round robin, and Same. A callout box points to this list with the text 'Types of Partitioning options available'. The 'Sorting' section has 'Perform sort' checked, and 'Stable' and 'Unique' are unchecked. The 'Selected' section is empty.

- Break up big data into partitions
- Run one partition on each processor
- 4X times faster on 4 processors; 100X faster on 100 processors
- Partitioning is specified per stage meaning partitioning can change between stages



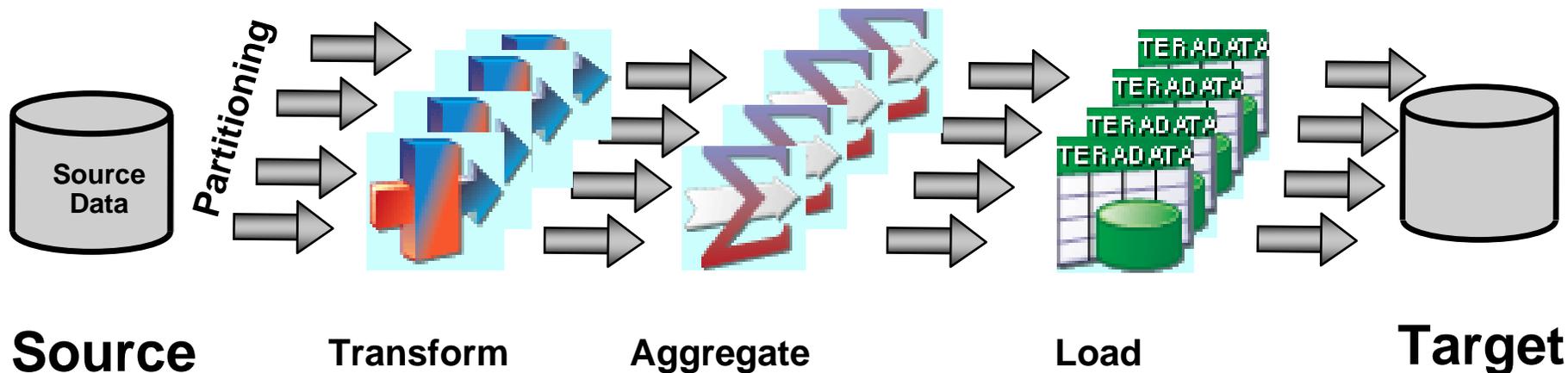
Data Pipelining



- Eliminate the write to disk and the read from disk between processes
- Start a downstream process while an upstream process is still running.
- This eliminates intermediate staging to disk, which is critical for big data.
- This also keeps the processors busy.



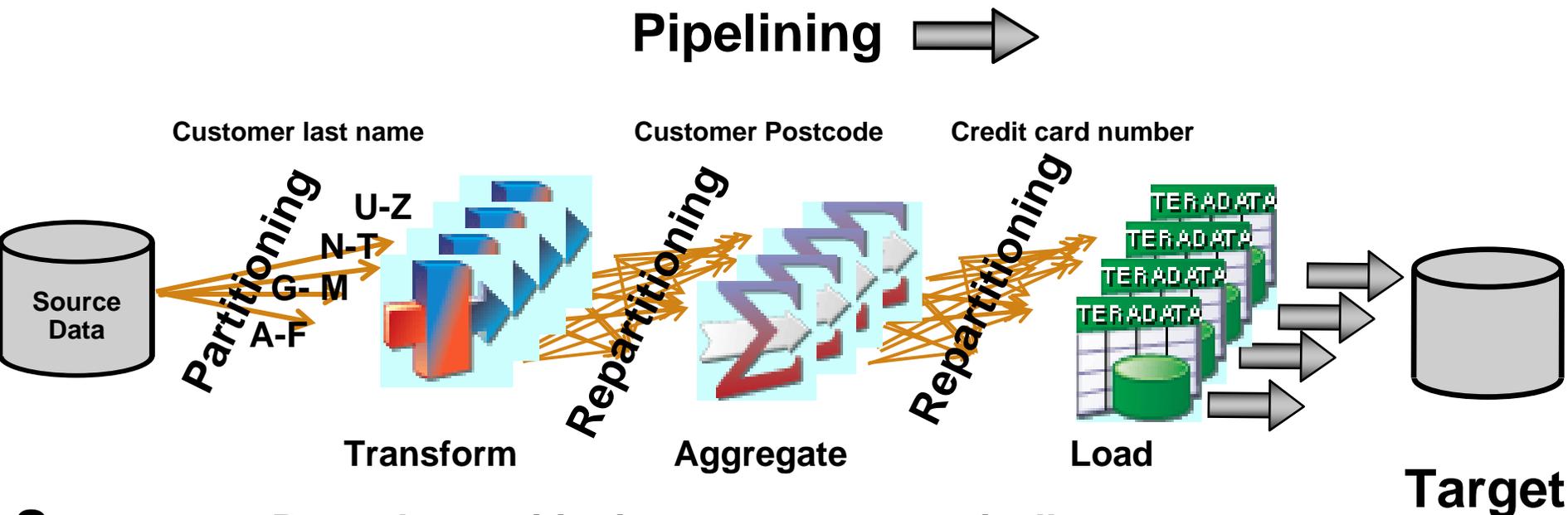
Parallel Dataflow



- **Parallel Processing achieved in a data flow**
- **Still limiting**
 - Partitioning remains constant throughout flow
 - Not realistic for any *real* jobs
 - For example, what if transformations are based on customer id and enrichment is a house holding task (i.e., based on post code)



Parallel Data Flow with Auto Repartitioning



Source

- Record repartitioning occurs automatically
 - No need to repartition data as you
 - add processors
 - change hardware architecture
 - Broad range of partitioning methods
 - Entire, hash, modulus, random, round robin, same, DB2 range



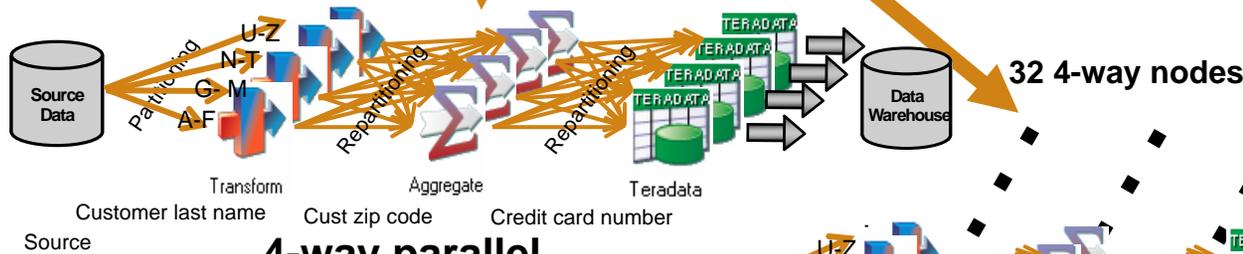
Application Assembly: One Dataflow Graph Created With the DataStage GUI



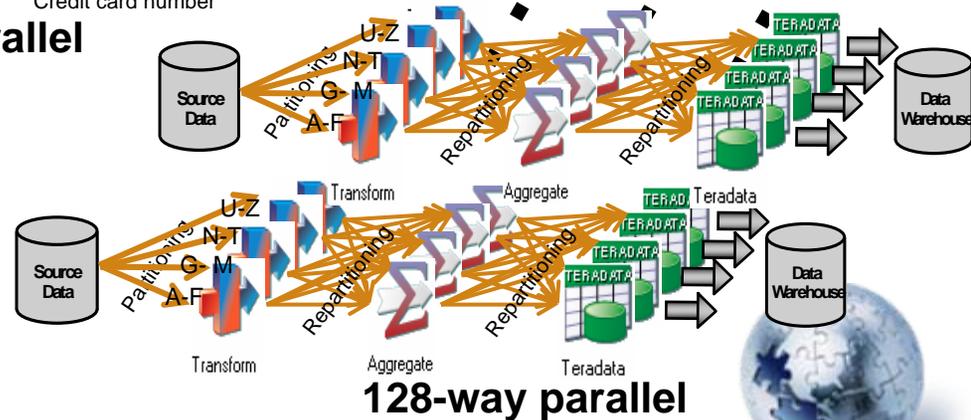
Application Execution: Sequential or Parallel



Sequential



4-way parallel



128-way parallel

Robust mechanisms for handling big data



Data Set stage: allows you to read data from or write data to a data set. Parallel Extender data sets hide the complexities of handling and storing large collections of records in parallel across the disks of a parallel computer.



File Set stage: allows you to read data from or write data to a file set. It only executes in parallel mode.



Sequential File stage: reads data from or writes data to one or more flat files. It usually executes in parallel, but can be configured to execute sequentially.



External Source stage: allows you to read data that is output from one or more source programs.



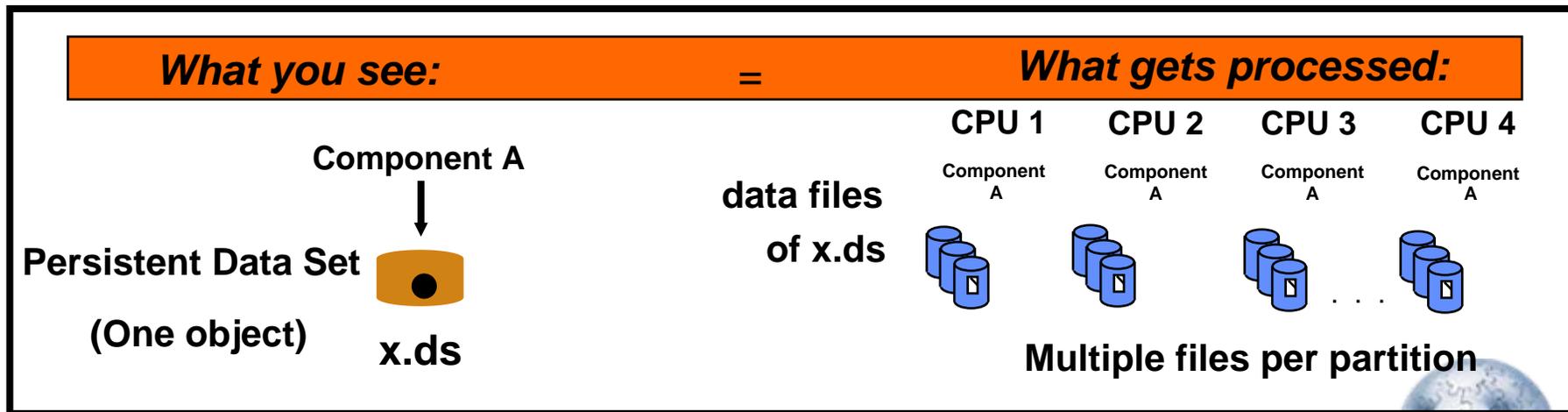
External Target stage: allows you to write data to one or more target programs.



Parallel Data Sets



- Hides Complexity of Handling Big Data
- Replicates RDBMS Support Outside of Database
- Consist of Partitioned Data and Schema
- Maintains Parallel Processing even when Data is staged for a checkpoint, point of reference or persisted between jobs.



DataStage SAS Stages



SAS stage:

- Executes part or all of a SAS application in parallel by processing parallel streams of data with parallel instances of SAS DATA and PROC steps.

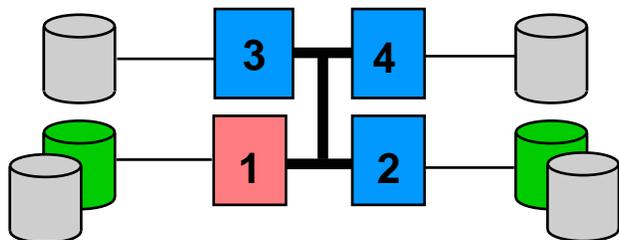
Parallel SAS Data Set stage:



- Allows you to read data from or write data to a parallel SAS data set in conjunction with a SAS stage.
- A parallel SAS Data Set is a set of one or more sequential SAS data sets, with a header file specifying the names and locations of all of the component files.



The Configuration File



Two key aspects:

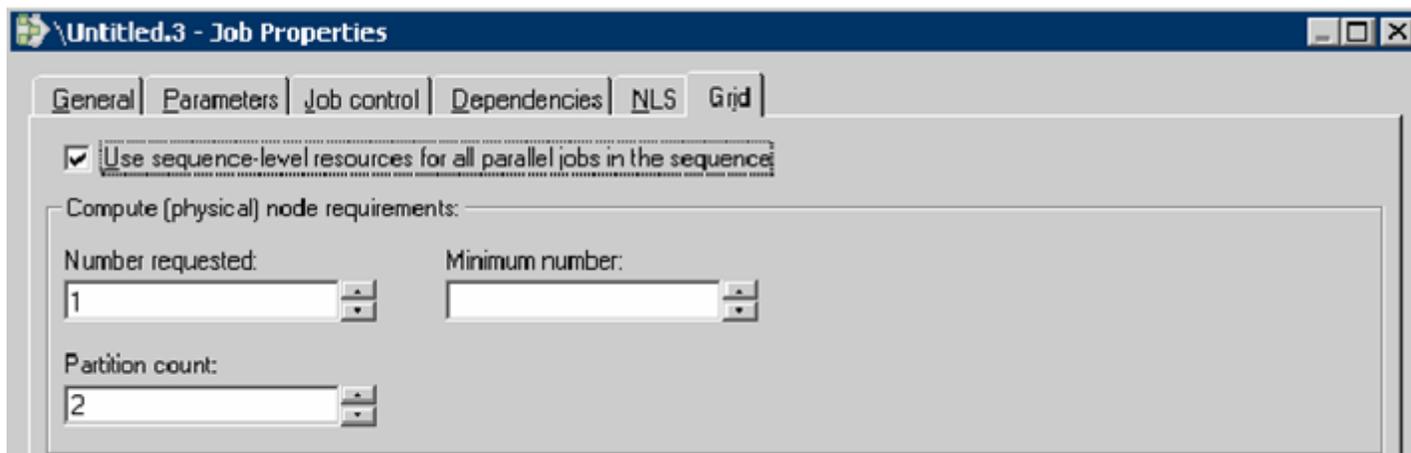
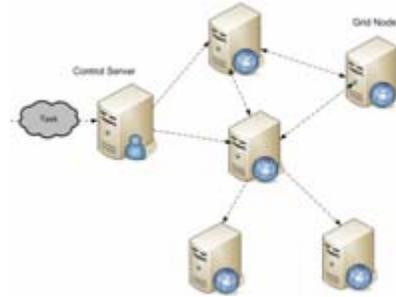
1. # nodes declared
2. defining subset of resources
"pool" for execution under
"constraints," i.e., using a
subset of resources

```
{
node "n1" {
  fastname "s1"
  pool "" "n1" "s1" "app2" "sort"
  resource disk "/orch/n1/d1" {}
  resource disk "/orch/n1/d2" {"bigdata"}
  resource scratchdisk "/temp" {"sort"}
}
node "n2" {
  fastname "s2"
  pool "" "n2" "s2" "app1"
  resource disk "/orch/n2/d1" {}
  resource disk "/orch/n2/d2" {"bigdata"}
  resource scratchdisk "/temp" {}
}
node "n3" {
  fastname "s3"
  pool "" "n3" "s3" "app1"
  resource disk "/orch/n3/d1" {}
  resource scratchdisk "/temp" {}
}
node "n4" {
  fastname "s4"
  pool "" "n4" "s4" "app1"
  resource disk "/orch/n4/d1" {}
  resource scratchdisk "/temp" {}
}
}
```



Dynamic GRID Capabilities

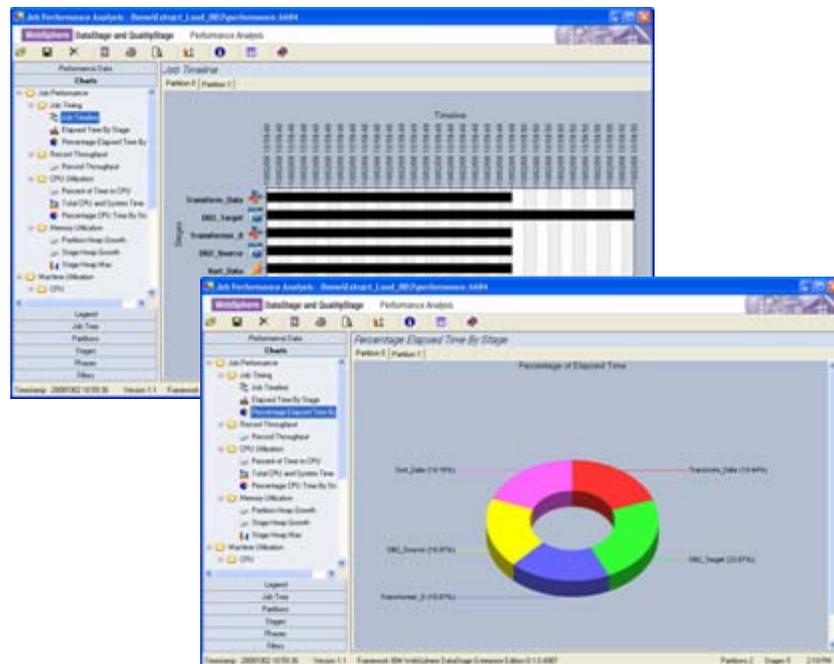
- **GRID Tab to help manage execution across a grid**
- **Automatically reconfigures parallelism to fit GRID resources**
 - Managed through an external grid resource manager
 - Available for Red Hat and Tivoli Work Scheduler Loadleveler
- **Locks resources at execution time to ensure SLAs**



Job Monitoring & Logging

- **Job Performance Analysis**
- **Detail job monitoring information available during and after job execution**
 - Start and elapsed times
 - Record counts per link
 - % CPU used by each process
 - Data skew across partitions
- **Also available from command line**
 - `dsjob -report <project> <job> [<type>]`
 type = **BASIC**, **DETAIL**, **XML**

Stage/Link name	Link type	Status	Num rows	Started at	Elapsed time	Rows/sec	%CP
PxSequentialFile0		Finished	1000000	11:37:32	00:00:34	29411	78
CTransformerStage3 X4		Finished	1000000	11:37:32	00:00:34	29411	92
CTransformerStage3.1		Finished	250000				21
CTransformerStage3.2		Finished	250000				19
CTransformerStage3.3		Finished	250000				20
CTransformerStage3.4		Finished	250000				32
PxSequentialFile1		Finished	1000000	11:37:32	00:00:34	29411	78



InfoSphere DataStage Balanced Optimization

Data Transformation

- Provides automatic optimization of data flows mapping transformation logic to SQL
- Leverages investments in DBMS hardware by executing data integration tasks with and within the DBMS
- Optimizes job run-time by allowing the developer to control where the job or various parts of the job will execute.

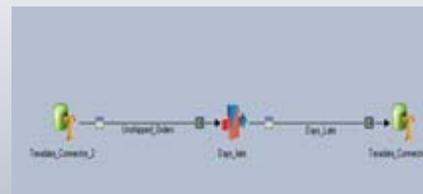


Developers



Architects

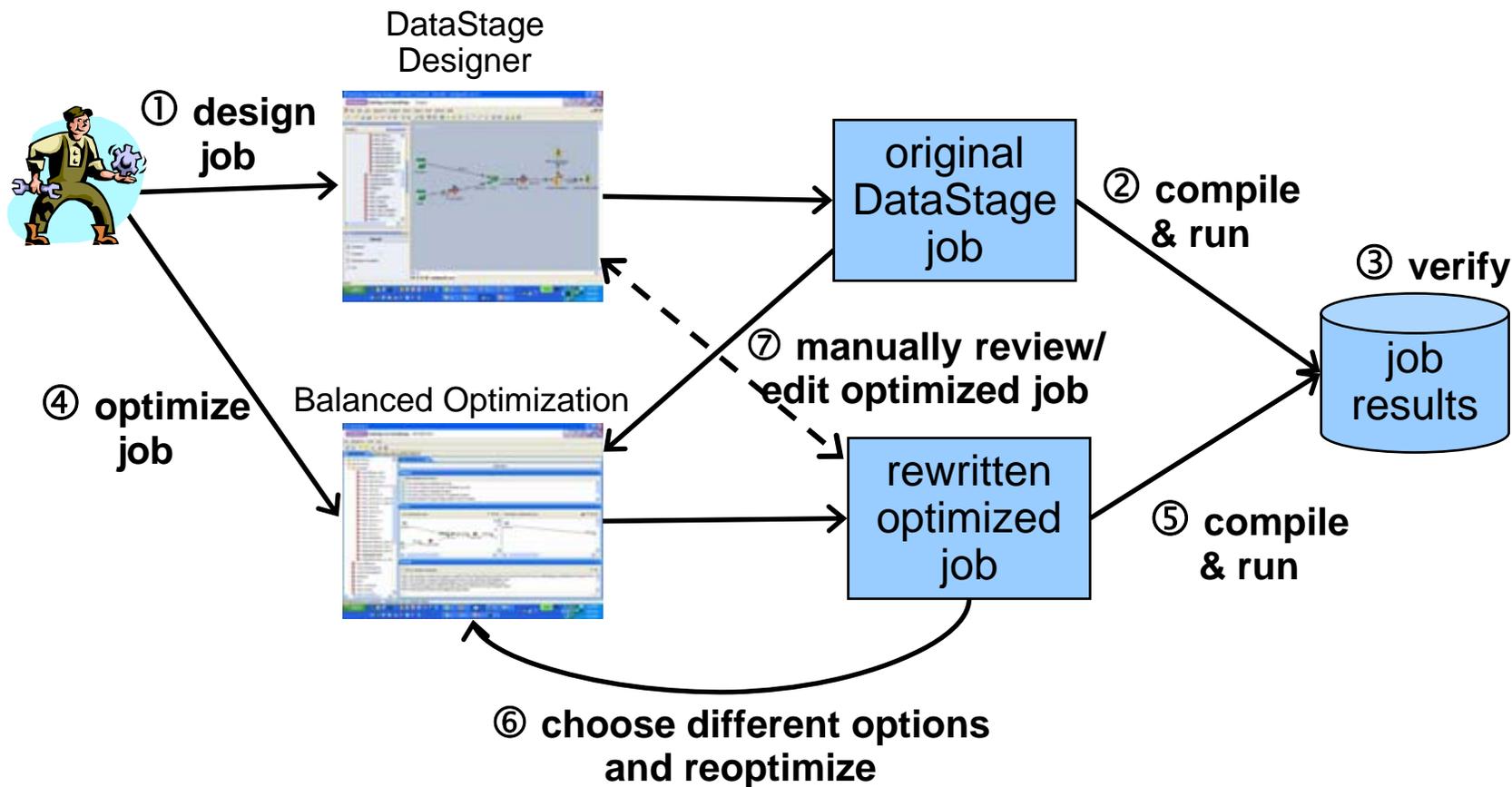
Transform and aggregate any volume of information in batch or real time through visually designed logic



Optimizing run time through intelligent use of DBMS hardware

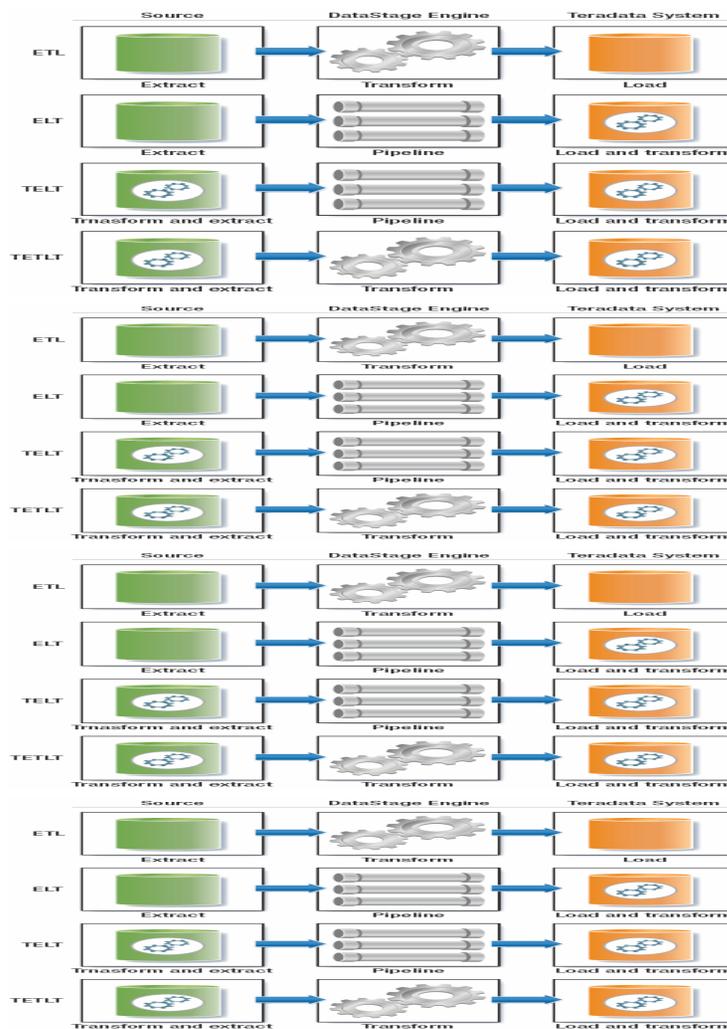


Using Balanced Optimization



Leveraging best-of-breed systems

- Optimization is not constrained to a single implementation style such as ETL or ELT
- InfoSphere DataStage Balanced Optimization fully harnesses available capacity and computing power in Teradata and DataStage
- Delivering unlimited scalability and performance through parallel execution everywhere, all the time



Agenda

- Platform Overview
- Appetizer - Productivity
- First Course – Extensibility
- Second Course– Scalability
- **Desert – Teradata**



InfoSphere Rich Connectivity

General Access
 Sequential File
 Complex Flat File
 File Set
 Data Set
 Named Pipe
 FTP (standard, secure)
 Compressed / Encoded Data

Real-Time
 WebSphere MQ
 SeeBeyond
 Java Messaging Services (JMS)
 Java (Client & Transformer)
 XML (Read / Write)
 XSL-T XSL-T Transformer
 Web Services (SOAP)

Enterprise Applications
 JD Edwards Oneworld (direct)
 Oracle Applications (Direct, Hierarchy)
 PeopleSoft (Direct, Trees)
 SAP BW (BAPI, IDOC)
 SAP R/3 (ABAP, BAPI, IDOC)
 Siebel (EIM, Business)

Connect
 Allbase/SQL
 Cache
 C-ISAM
 Datacom/DB
 DB2 UDB
 DB2/400
 DBMS

MS Analysis
 Nomad
 NonStopSQL
 Nucleus
 ODBC
 OLAP Services
 Oracle
 Progress

Rich Connectivity

Shared, easy to use connectivity infrastructure

Best-of-breed, metadata-driven connectivity to enterprise applications

High volume, parallel connectivity to databases and file systems

Event-driven, real-time, and batch connectivity

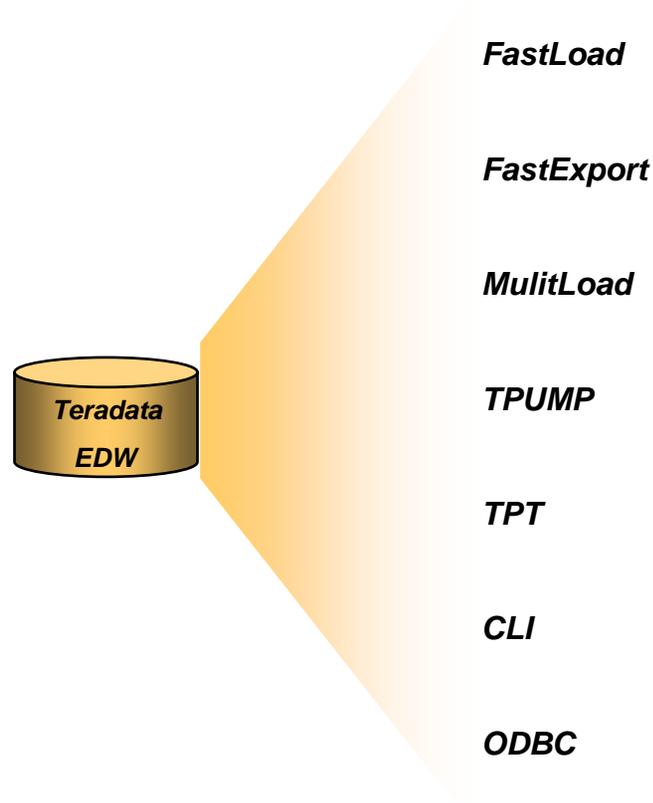
The screenshot shows the DataStage 'Properties' window for a stage named 'Customer_Records'. It includes sections for 'Connections' (Data source, Username, Password) and 'Stage' (End of Wave, Isolation level, Enable Unicode). The background features logos for various enterprise systems: SAP, PeopleSoft, Siebel, JDA, Ariba, Sybase, IBM, Oracle, and Microsoft.

Frictionless Connectivity



Teradata Connectivity

- 7+ highly optimized interfaces for Teradata leveraging Teradata Tools and Utilities: FastLoad, FastExport, TPUMP, MultiLoad, Teradata Parallel Transport, CLI and ODBC
- You use the best interfaces specifically designed for your integration requirements:
 - Parallel/bulk extracts/loads with various/optimized data partitioning options
 - Table maintenance
 - Real-time/transactional trickle feeds without table locking

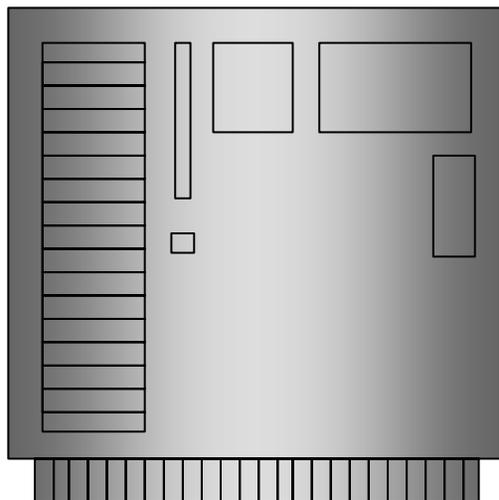


Teradata Connectivity

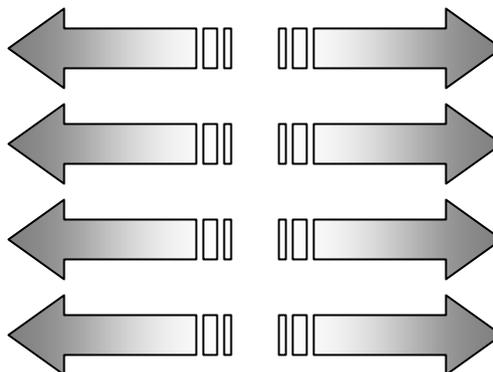
- **RequestedSessions** determines the total number of distributed connections to the Teradata source or target
 - When not specified, it equals the number of Teradata VPROCs (AMPs)
 - Can set between 1 and number of VPROCs
- **SessionsPerPlayer** determines the number of connections each player will have to Teradata. Indirectly, it also determines the number of players (degree of parallelism).
 - Default is 2 sessions / player
 - The number selected should be such that $\text{SessionsPerPlayer} * \text{number of nodes} * \text{number of players per node} = \text{RequestedSessions}$
 - Setting the value of *SessionsPerPlayer* too low on a large system can result in so many players that the job fails due to insufficient resources. In that case, the value for -SessionsPerPlayer should be increased.



Teradata SessionsPerPlayer Example



DataStage Server



Teradata Server

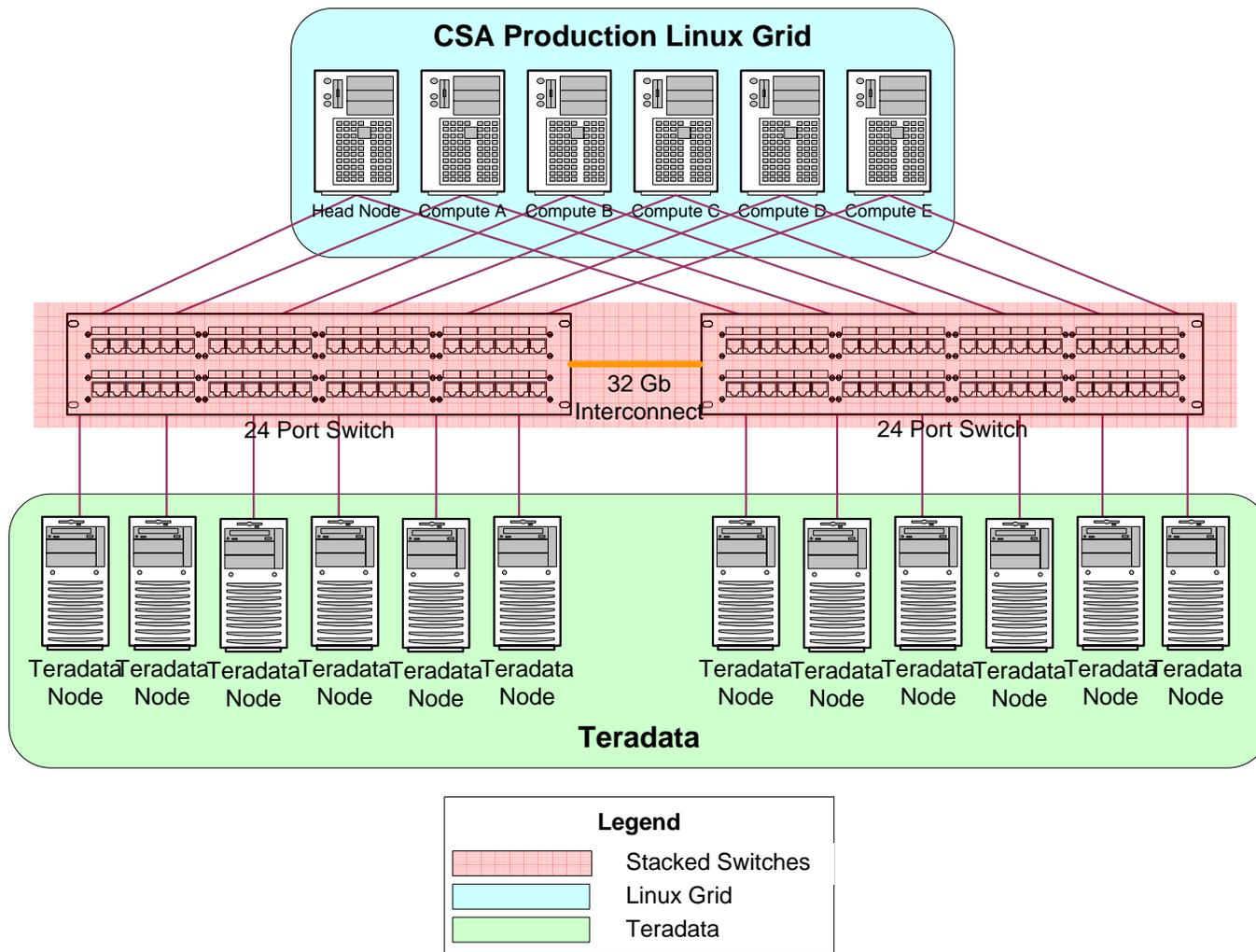
- MPP with 4 TPA nodes
- 4 AMP's per TPA node

Example Settings

Configuration File	Sessions Per Player	Total Sessions
16 nodes	1	16
8 nodes	2	16
8 nodes	1	8
4 nodes	4	16



Customer Example – Massive Throughput



The IBM InfoSphere Information Server Advantage

A Complete Information Infrastructure

- A ***comprehensive, unified foundation*** for enterprise information architectures, scalable to any volume and processing requirement
- ***Auditable data quality*** as a foundation for trusted information across the enterprise
- ***Metadata-driven integration***, providing breakthrough productivity and flexibility for integrating and enriching information
- ***Consistent, reusable information services*** — along with application services and process services, an enterprise essential
- Accelerated time to value with ***proven, industry-aligned solutions*** and expertise
- ***Broadest and deepest connectivity*** to information across diverse sources: structured, unstructured, mainframe, and applications



Thank
YOU

