

Data Warehousing and Business Intelligence On IBM i

Alan Jordan

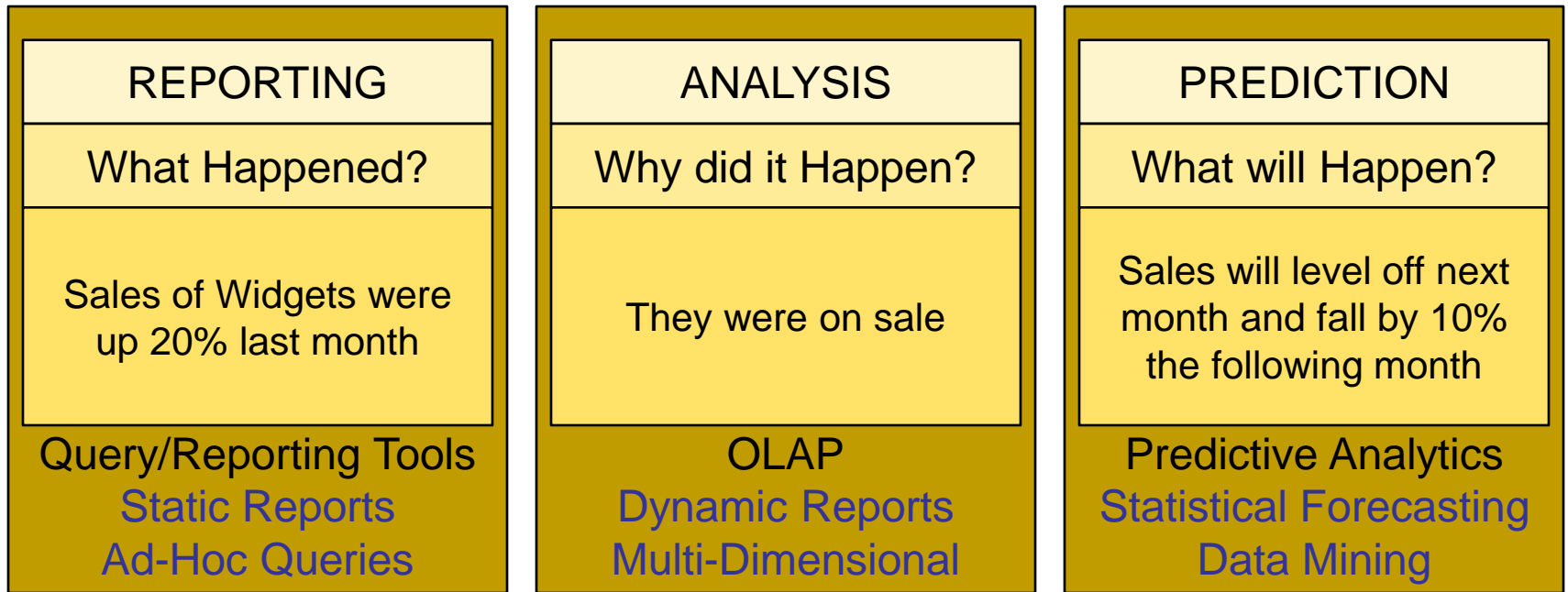
Coglin Mill, Rochester, Minnesota USA

ajordan@coglinmill.com

What is Business Intelligence?

- Business intelligence (BI) is a broad category of applications and technologies for gathering, storing, analyzing, and providing access to data to help enterprise users make better business decisions.
- BI applications include data warehouses, data marts, decision support systems, query and reporting, online analytical processing (OLAP), statistical analysis, forecasting, and data mining.
- Business intelligence applications can be:
 - Mission-critical and integral to an enterprise's operations or occasional to meet a special requirement
 - Enterprise-wide or local to one division, department, or project

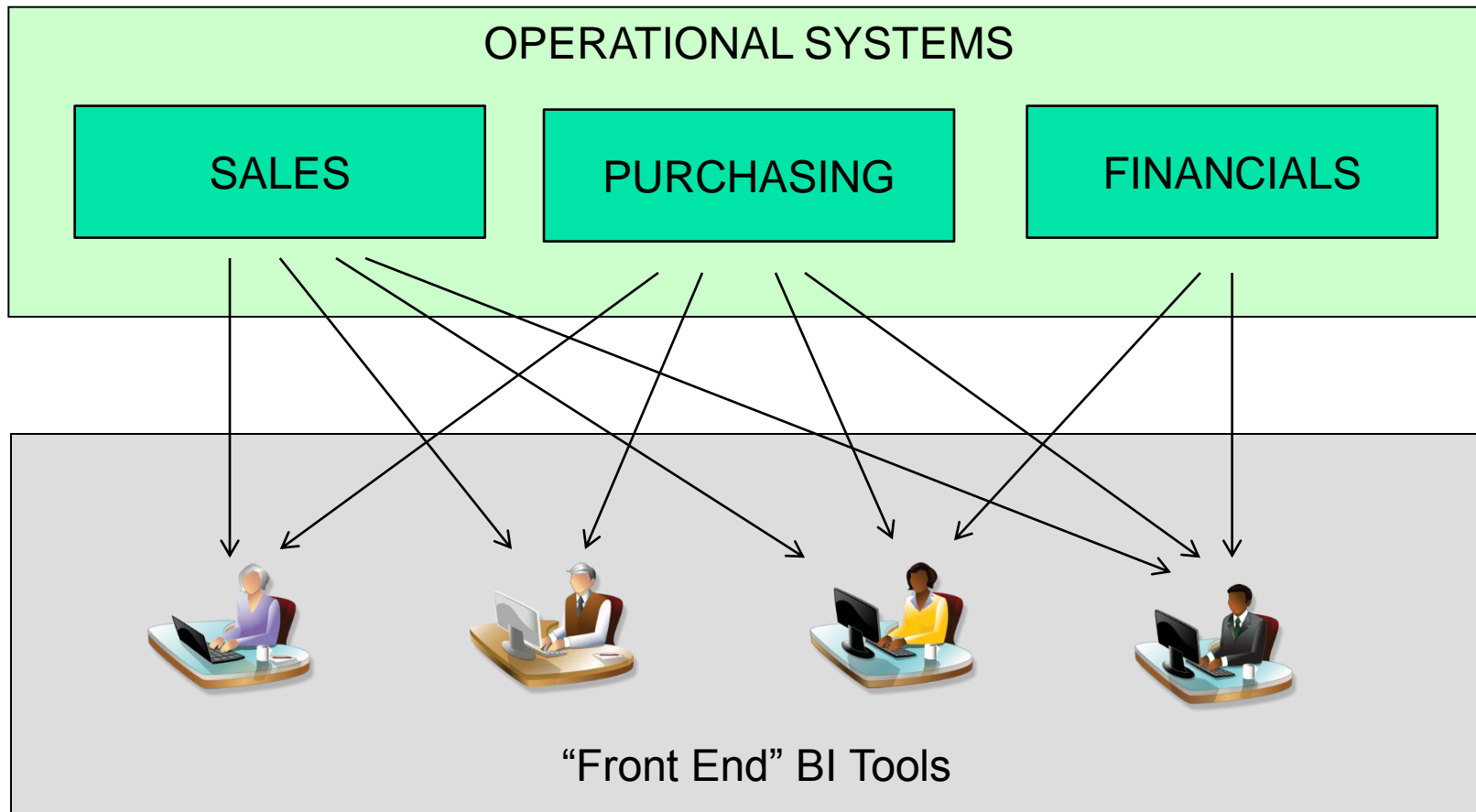
What is Business Intelligence?



Increasing Business Value

Increasing Complexity

Simple Implementation

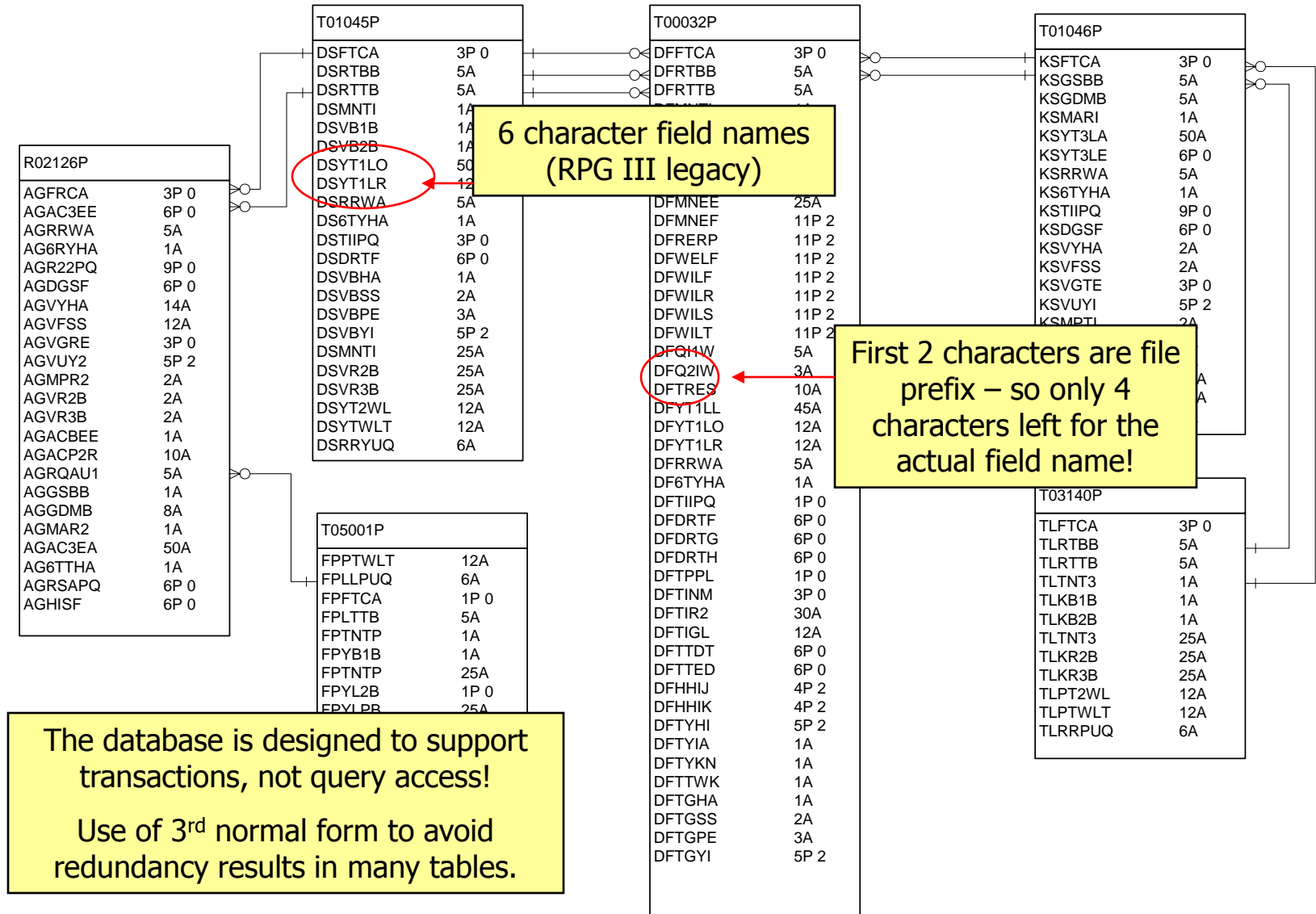


- I do not understand where/how the data is stored
- The database was not designed for YOU to access
 - It was designed for the software application (ERP, MRP)
 - Column names are not meaningful
- There is probably no 'user manual' for the tables
 - The user manual/documentation is for the application
- There is probably no consistency of design
 - The application grew over many years
 - Different developers
 - Merger of different applications
 - Redundant fields reused for different purpose
- Significant number of tables/columns
- Many are not useful for BI

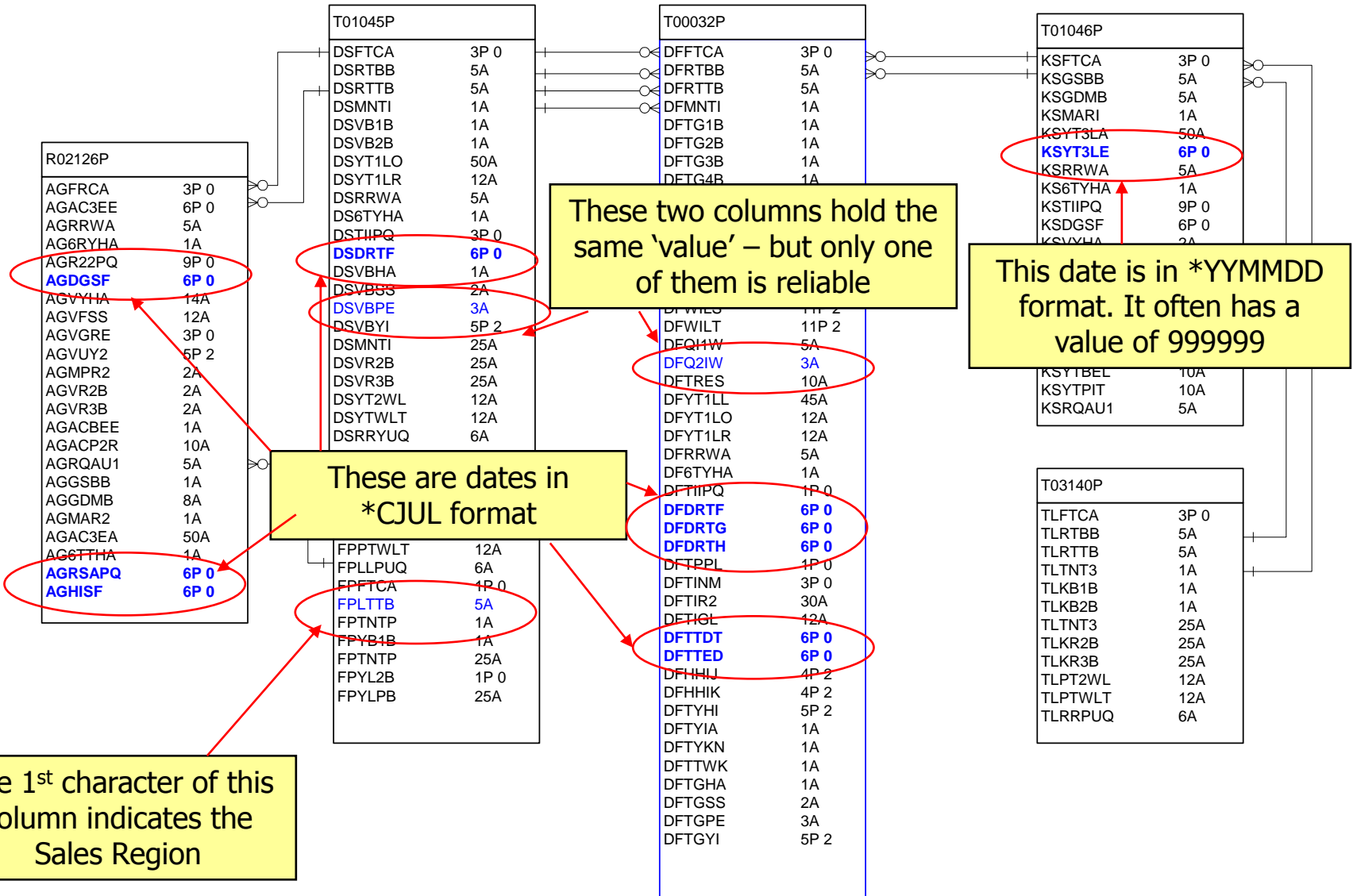
- Operational Data is in many tables – hard to join
- Because of principle of 3rd Normal Form
 - Avoiding redundancy of data results in many tables
- May need to include intermediate tables to get to required table
 - No data needed from intermediate tables
- Join may require data manipulation
 - Derived values, different data types etc.
- May not be exact join
 - e.g. Join by 'closest to' value

- Cryptic or difficult data values
- Often the legacy of costly disk
 - Why use 10 bytes when 1 or 2 will do
 - Maybe even use bits instead of bytes
- Embedded values and conditional rules
 - 2nd character of column X means ..
 - If column Y = 'S', value Z must be multiplied by -1
 - If record type is '1', there must be a matching record in table B
 - If type is '2, there may be a record
- Dates
 - When dates are just numbers, how do you report by quarter or add a month to a date?
 - How do you handle a date of zero, or all 9s?
 - Converting numeric dates to date data type in a query is very costly

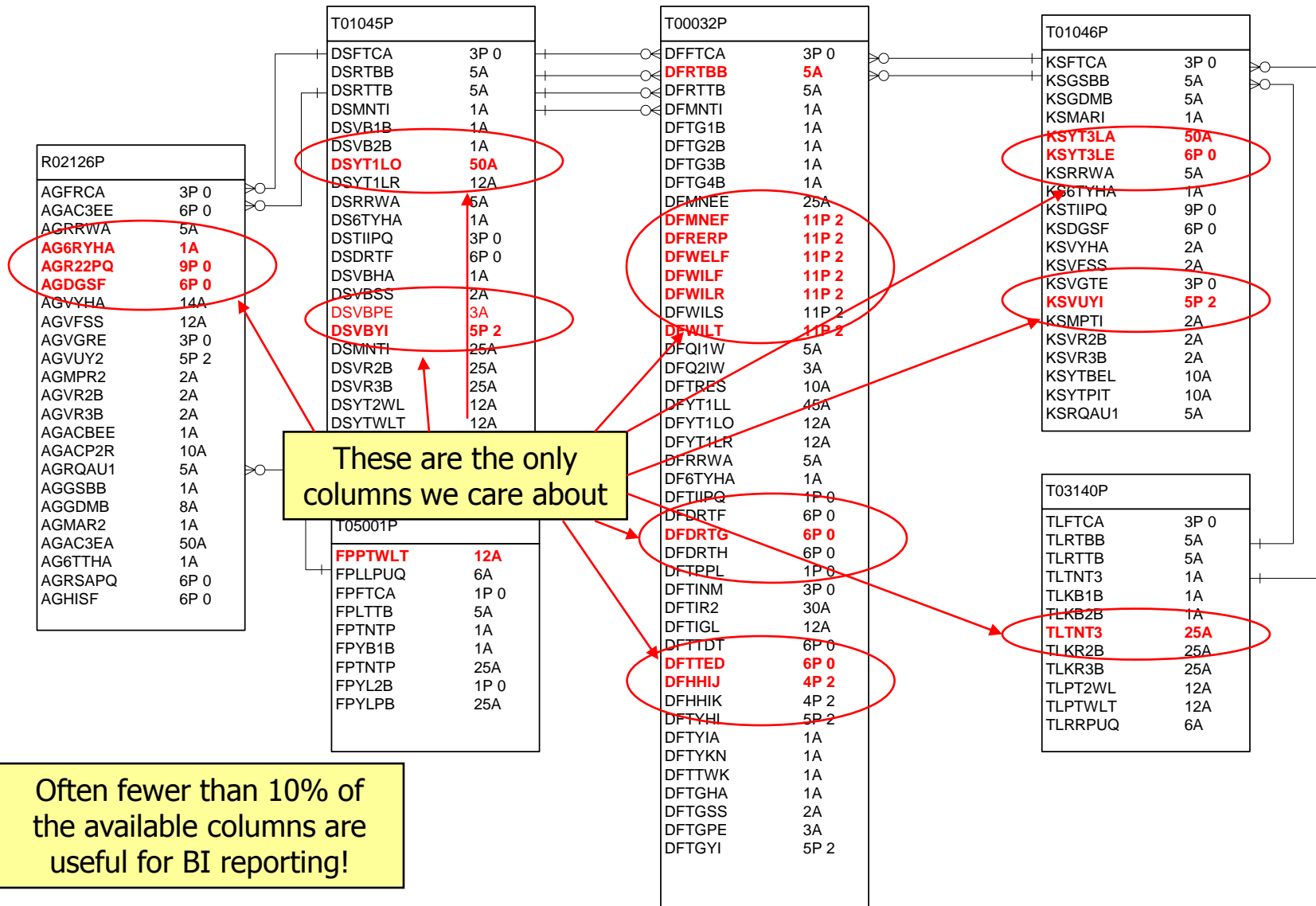
Complex Data Example



Complex Data Example



Complex Data Example



- Poor/unknown data quality

- Missing/incorrect Values
 - Inadequate edit checks in application
 - Software errors
 - Changing business rules
 - Incorrect data conversions

- Resulting in
 - Failed joins
 - Incorrect reports
 - Bad decisions

- **Data Quality is one of your most important issues!**
 - Your company will be making strategic decisions based on the data used in reports. You hope it is accurate!
 - Poor data quality is the most common reason for failure of business intelligence initiatives

A 2002 study by TDWI found that
poor data quality costs US businesses

\$600 BILLION per year!

Conservative estimates suggest that
poor data quality costs the typical organization
Up to 10% of potential revenue and
Up to 20% of potential earnings

■ Data Quality Example

- 2005 Valparaiso, Indiana
- Somehow a property assessment value for the home shown below was incorrectly changed to \$400M in the property tax database
- The expected property tax revenue was included in the county budget - but the \$8M property tax bill on the house was (of course) not paid
- The county had a huge revenue shortfall, resulting in lots of cuts
- The school district was forced to return \$2.7M
- All extracurricular activities and sports were cancelled that year

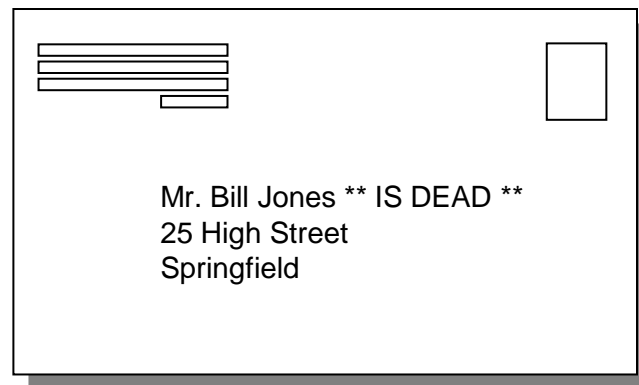
**Just because of
ONE
bad data value!**

see more examples at www.thinkrodin.com



■ Data Quality Example

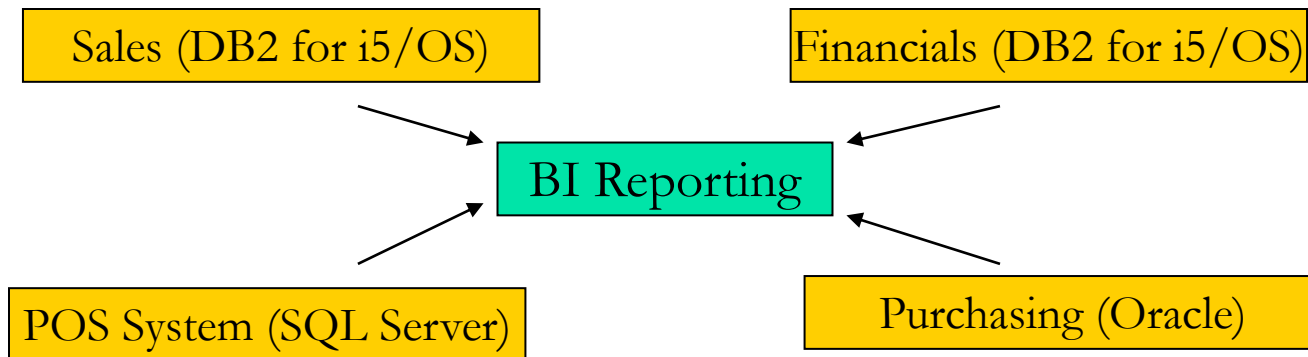
- A travel agency used telemarketing to sell new vacations to its past customers
- On occasion, it happened that a customer had passed away. Their system would not let them delete the customer, since there were transaction records tied to it. Someone came up with the idea of appending the customer name with "*** IS DEAD **", so operators would not call in the future and upset the family of the deceased.
- This worked fine until the company implemented a data warehouse and switched to direct mail. Imagine the grief caused to Mrs. Jones when she received this letter



*This really DID
happen!
(in the UK)*

Different applications/databases

- Totally different structures – but related information



- Very difficult, if not impossible to join tables across databases
- Different security, availability etc.
- Adds significant complexity
- Poor performance

Incompatible Data Example

- Multiple instances of same table, with duplicate key values

Customer File - US	
CUSTNO	CUSTNAME
1001	John Smith
1002	Mary Jones
1003	Chris Anderson
1004	David Perry

Customer File - Canada	
CUSTNO	CUSTNAME
1001	Harry Potter
1002	Jeremy Carr
1003	Penny Hayes
1004	Debbie Thornton

- or different versions of same entity
 - Incompatible data types
 - Duplicates

Customer File - US	
CUSTNO	CUSTNAME
1001	John Smith
1002	Mary Jones
1003	Chris Anderson
1004	David Perry

Customer File - Canada	
CUSTID	CUSTNAM
AA234	Julie Johnson
AA235	Fred Hunter
AB670	John Smith
BD309	Alan Jordan

Changing Dimensions

2008	100	Acme Flooring	Small Retailer	Jenny Brown
2010	100	Acme Flooring	Major Retailer	Jenny Brown
2011	100	Acme Flooring	Major Retailer	Rob McAdam

2008 Report

2008 Sales by Sales Rep/Customer Group

Acme Flooring	250,000
Regal Rugs	150,000
Total Small Retailer	400,000
Carpet Warehouse	2,500,000
Hardwood Hank	2,100,000
Total Large Retailer	4,600,000
Total Jenny Brown	5,000,000

Same Report, re-run in 2010

2008 Sales by Sales Rep/Customer Group

Regal Rugs	150,000
Total Small Retailer	150,000
Acme Flooring	250,000
Carpet Warehouse	2,500,000
Hardwood Hank	2,100,000
Total Large Retailer	4,850,000
Total Jenny Brown	5,000,000

Changing Dimensions

2008	100	Acme Flooring	Small Retailer	Jenny Brown
2010	100	Acme Flooring	Major Retailer	Jenny Brown
2011	100	Acme Flooring	Major Retailer	Rob McAdam

2008 Report

2008 Sales by Sales Rep/Customer Group

Acme Flooring	250,000
Regal Rugs	150,000
Total Small Retailer	400,000
Carpet Warehouse	2,500,000
Hardwood Hank	2,100,000
Total Large Retailer	4,600,000
Total Jenny Brown	5,000,000

Same Report, re-run in 2011

2008 Sales by Sales Rep/Customer Group

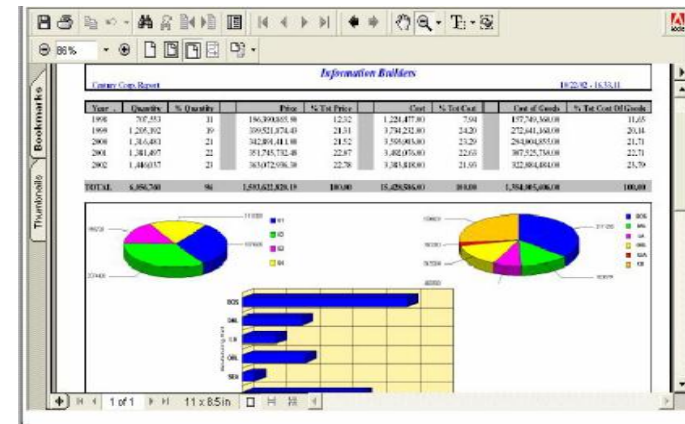
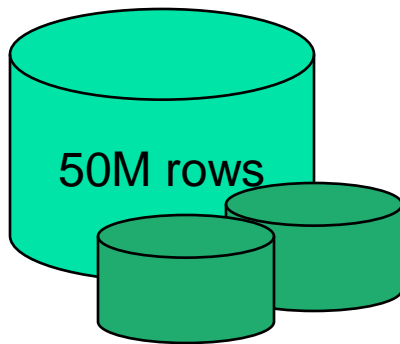
Regal Rugs	150,000
Total Small Retailer	150,000
Carpet Warehouse	2,500,000
Hardwood Hank	2,100,000
Total Large Retailer	4,600,000
Total Jenny Brown	4,750,000

Similar issues when a report compares This Year vs Last Year

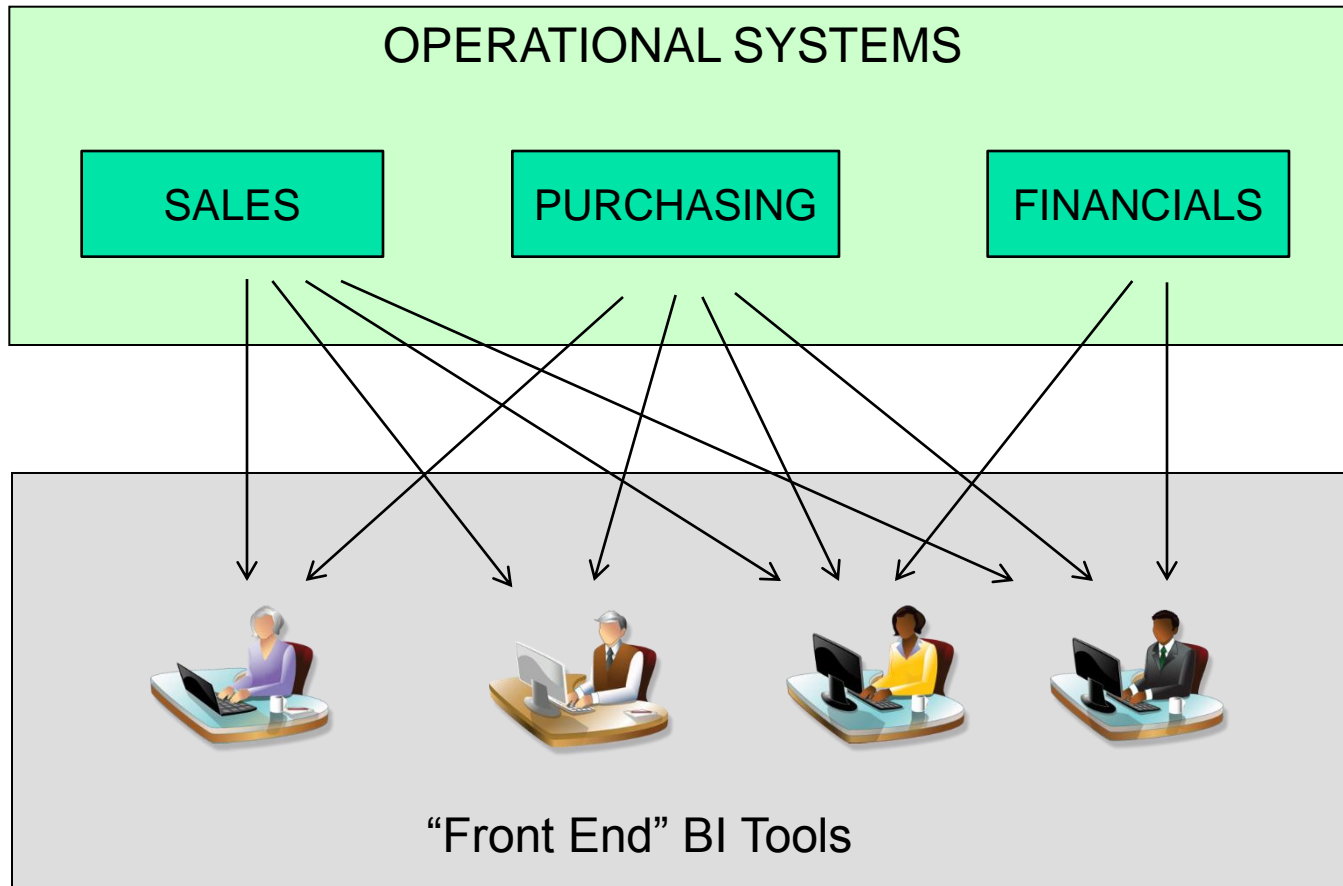
Common Scenarios

■ Poor performance

- large transaction table
- many related tables
- most reports are at a summary level
- reports and queries are long running and consume significant system resources



Simple Implementation



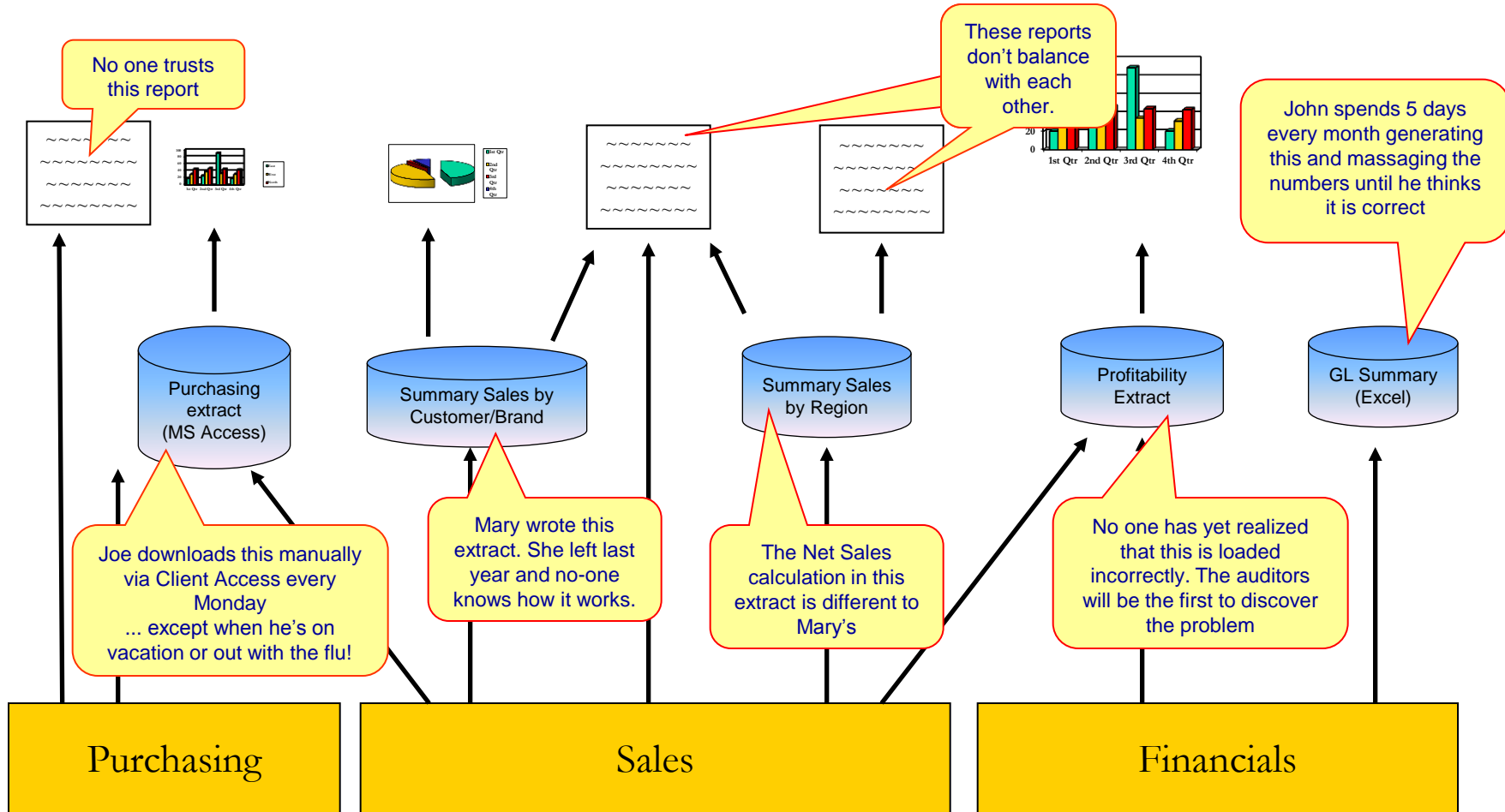
Issues and complexity pushed to the front end tools

The “Solution”?

- These issues are often *solved* in an ad-hoc way
 - Create *extract files* and write RPG programs to load them
 - As each reporting problem occurs, a new extract is written
 - No consistent approach
 - No documentation produced
 - Frustrated users create their own “solutions”
 - Download data to Excel and manipulate it
 - Decide on their own rules

The Usual Result

A chaotic reporting environment!



Develop a Sustainable Business Intelligence Architecture

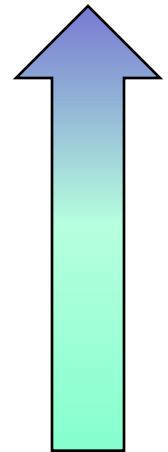
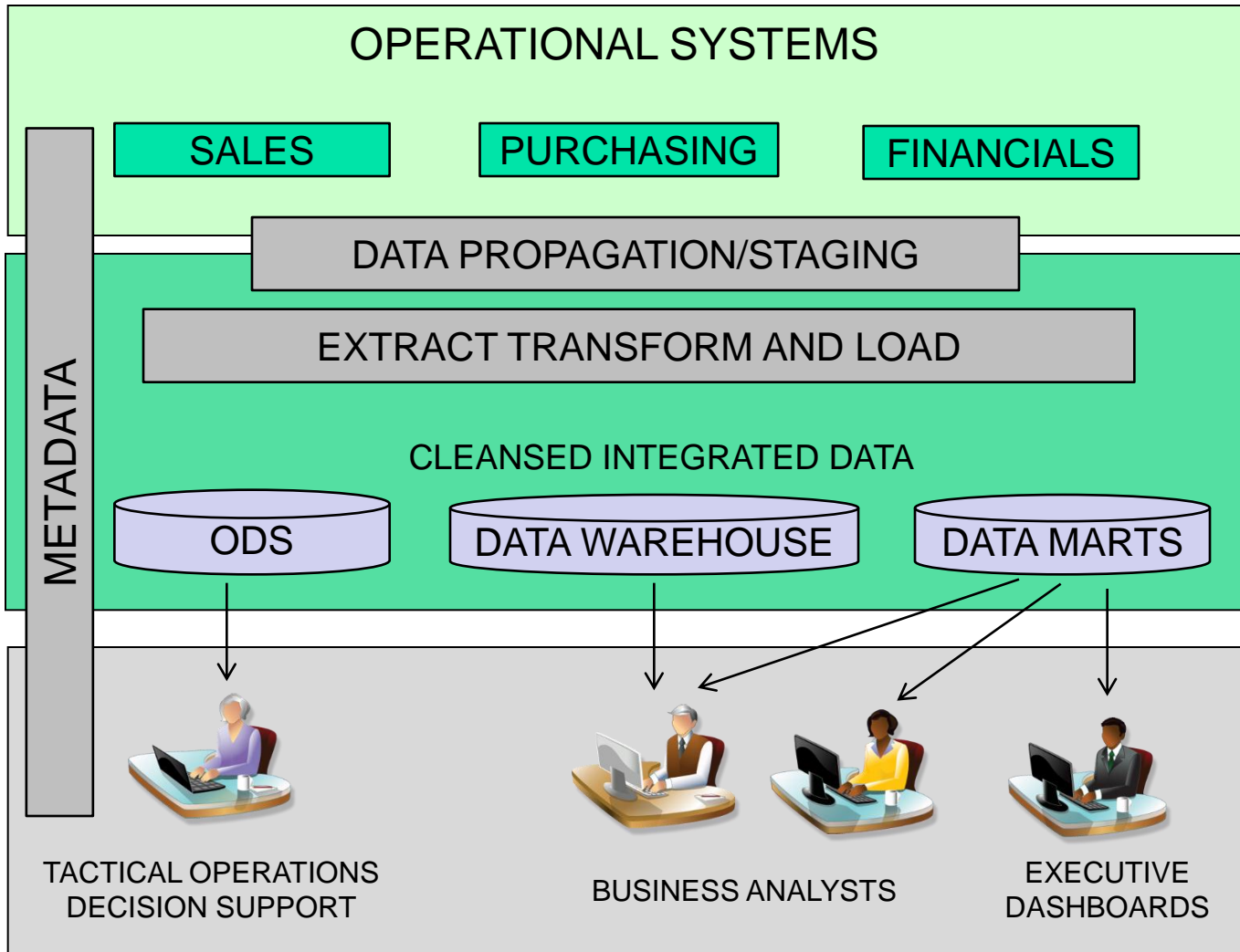
Fortune 500 Companies have (almost without exception) have spent millions of dollars building Enterprise Data Warehouses to resolve these issues.

For the vast majority of IBM i customers, that sort of expenditure is completely outside the realm of possibility.

But with the right tools, and a sensible approach it doesn't need to cost anywhere near that much. In fact it will pay for itself very quickly.

An IDC study of 52 implementations found the average 3-year ROI of a data warehouse was 401%!

Enterprise Data Warehouse Architecture



Issues and complexity pushed to the "back end"

■ Data Warehouse

- A centralized repository of mostly historical information, built from operational data sources
- Usually contains several different subject areas
- Always in open database tables
- Usually detailed level information, but can include summary data too
- A single version of the truth

■ Data Mart

- Built from the Data Warehouse to support a specific business reporting requirement
- Often summarized, but may be detailed
- If in database tables, often a star schema structure
- May be in a proprietary format - ie MOLAP (cube), such as Essbase
- Updated (or re-built) on a regular basis from the data warehouse

■ Operational Data Store

- A reporting database containing the 'current' view of the operations of the business
- Contains little or no historical data
- Contains incomplete or in-progress entities (ie sales orders not yet fulfilled)
- Completely re-built on a regular (usually daily) basis
- Often creates outputs that feed back into operational systems

■ Metadata

- “Data that describes data”
- Technical metadata – e.g. table and column names, length, data type, decimals
- Business metadata - e.g. validation rules, transformation rules, source/target relationships
- Administrative metadata - e.g. users, authorities, size, usage, performance and data quality statistics, change history

Data Mart Example



PRODUCTS	
PRODUCT_NUMBER	5P 0
PRODUCT_DESCRIPTION	42A
BRAND_CODE	5A
BRAND_DESCRIPTION	20A
ORIGIN_CODE	5A
ORIGIN_DESCRIPTION	20A
FAMILY_CODE	5A
FAMILY_DESCRIPTION	20A
COST	9P 2
BASE_PRICE	9P 2
PRODUCT_WEIGHT	9P 4
PRODUCT_VOLUME	9P 4
LOAD_DATE	DATE
LAST_CHANGE_TIME	TSTP
STATUS_FLAG	1A

INVOICE_LINES	
INVOICE_NUMBER	7P 0
INVOICE_LINE_NUMBER	3P 0
PRODUCT_NUMBER	5P 0
CUSTOMER_NUMBER	10A
SELLING_COMPANY	5A
SUPPLY_WAREHOUSE	5A
QUANTITY_ORDERED	
QUANTITY_SHIPPED	
TOTAL_DISCOUNT	
NET_PRICE	
BASE_PRICE	
UNIT_COST	
EXTENDED_COST	11P 2
EXTENDED_PRICE	11P 2
MARGIN	11P 2
SALES_REP	5A
COMMISSION_VALUE	7P 2
INVOICE_DATE	DATE
SHIP_DATE	DATE
DELIVERY_DATE	DATE
INVOICE_TIME	TIME
MONTH_NUMBER	2P 0
WEEK_NUMBER	2P 0
LOAD_DATE	(DATE)

CUSTOMERS	
CUSTOMER_NUMBER	10A
CUSTOMER_NAME	35A
ADDRESS_LINE_1	35A
ADDRESS_LINE_2	35A
PHONE	35A
PHONE_2	35A
PHONE_3	35A
PHONE_4	35A
PHONE_5	35A
PHONE_6	35A
PHONE_7	35A
PHONE_8	35A
PHONE_9	35A
PHONE_10	35A
PHONE_11	35A
PHONE_12	35A
PHONE_13	35A
PHONE_14	35A
PHONE_15	35A
PHONE_16	35A
PHONE_17	35A
PHONE_18	35A
PHONE_19	35A
PHONE_20	35A
PHONE_21	35A
PHONE_22	35A
PHONE_23	35A
PHONE_24	35A
PHONE_25	35A
PHONE_26	35A
PHONE_27	35A
PHONE_28	35A
PHONE_29	35A
PHONE_30	35A
PHONE_31	35A
PHONE_32	35A
PHONE_33	35A
PHONE_34	35A
PHONE_35	35A
PHONE_36	35A
PHONE_37	35A
PHONE_38	35A
PHONE_39	35A
PHONE_40	35A
PHONE_41	35A
PHONE_42	35A
PHONE_43	35A
PHONE_44	35A
PHONE_45	35A
PHONE_46	35A
PHONE_47	35A
PHONE_48	35A
PHONE_49	35A
PHONE_50	35A
PHONE_51	35A
PHONE_52	35A
PHONE_53	35A
PHONE_54	35A
PHONE_55	35A
PHONE_56	35A
PHONE_57	35A
PHONE_58	35A
PHONE_59	35A
PHONE_60	35A
PHONE_61	35A
PHONE_62	35A
PHONE_63	35A
PHONE_64	35A
PHONE_65	35A
PHONE_66	35A
PHONE_67	35A
PHONE_68	35A
PHONE_69	35A
PHONE_70	35A
PHONE_71	35A
PHONE_72	35A
PHONE_73	35A
PHONE_74	35A
PHONE_75	35A
PHONE_76	35A
PHONE_77	35A
PHONE_78	35A
PHONE_79	35A
PHONE_80	35A
PHONE_81	35A
PHONE_82	35A
PHONE_83	35A
PHONE_84	35A
PHONE_85	35A
PHONE_86	35A
PHONE_87	35A
PHONE_88	35A
PHONE_89	35A
PHONE_90	35A
PHONE_91	35A
PHONE_92	35A
PHONE_93	35A
PHONE_94	35A
PHONE_95	35A
PHONE_96	35A
PHONE_97	35A
PHONE_98	35A
PHONE_99	35A
PHONE_100	35A
SALES_REP_DEFAULT	5A
CUSTOMER_CATEGORY	5A
CUSTOMER_CLASS	5A
REGION_CODE	5A
LOAD_DATE	DATE
LAST_CHANGE_TIME	TMSTP
STATUS_FLAG	1A

Meaningful table and column names

Complex calculations already done

Dates are true date columns

De-normalized design reduced to only 3 tables

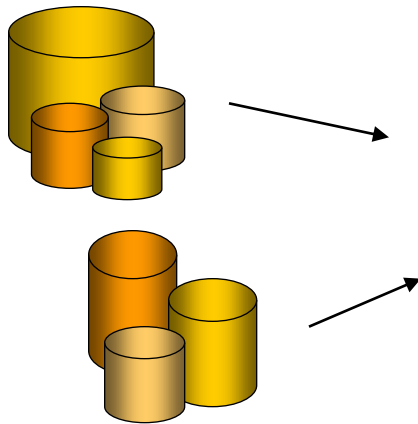
Only includes the columns we care about

- By definition, ETL implies some type of reporting database
 - data warehouse
 - data marts
 - operational data store

and the design and maintenance of this database

- Implemented correctly, this is significantly different to the ad-hoc creation of extract and summary files found in many organizations
 - designed for query access
 - conformed dimensions
 - integrated and controlled
 - includes data quality management
 - supported by metadata

- **Extract** data from sources
 - Database tables
 - Legacy file systems
 - Text or delimited files
 - May need to stage the data



Extract / Stage Options

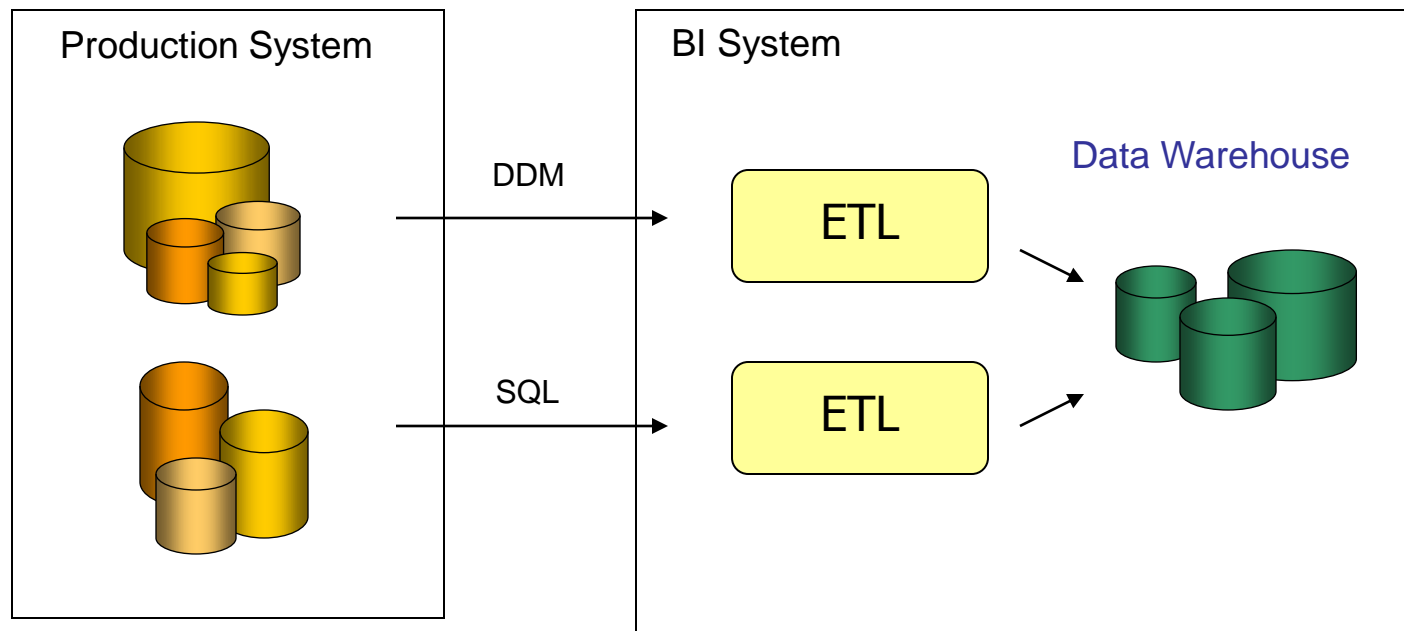
- Identify and understand the required source data
 - Local or remote
 - DB2 for i or other database
 - Non-database (legacy structures, text files etc)

- Filtering/Selection
 - All rows or by transaction date
 - Identifying changed rows
 - Exclude certain rows

- Decide on access mechanism
 - Direct access
 - Change data capture
 - Staging of remote data

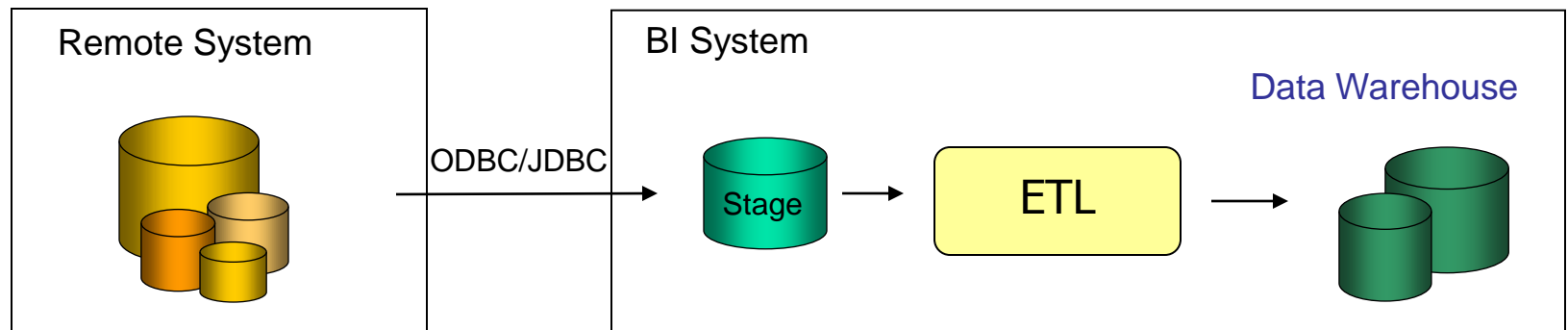
Extract / Stage Options

- Direct Connection to remote DB2 for i data
 - DDM – performs well, especially if another LPAR on same system
 - Remote SQL connection
 - No need to stage data



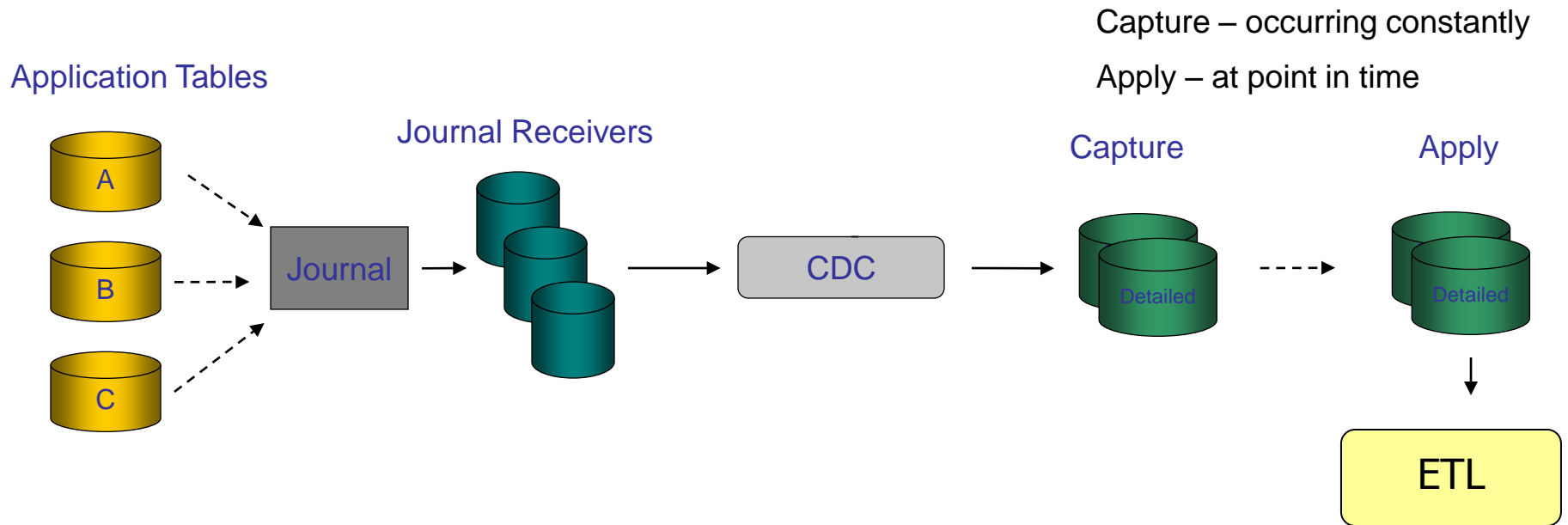
Extract / Stage Options

- Staging is often best approach for non IBM i data
 - Non IBM Family databases do not support full DRDA architecture
 - Sometimes need to access same table in several ETL processes
 - Can simplify error recovery in event of a failure

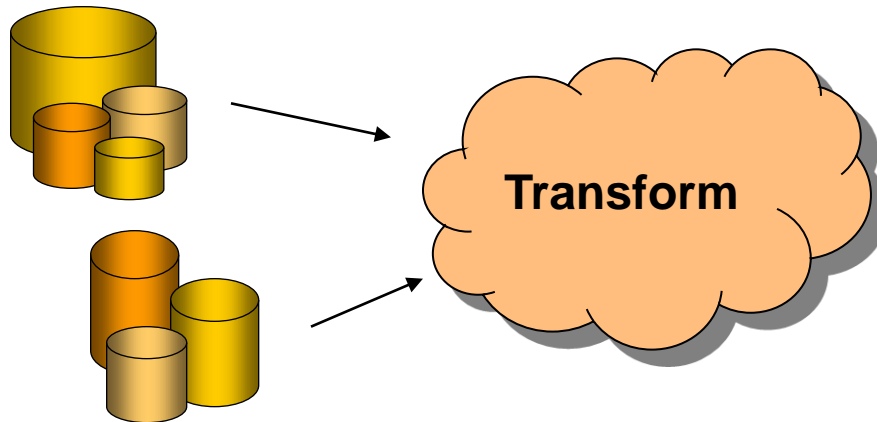


Extract / Stage Options

- Change Data Capture & Apply
 - Non-intrusive on source systems
 - Can be based on remote journals
 - Possibly piggy-back on HA strategy



- **Transform** the extracted data
 - arithmetic calculations
 - string operations
 - lookup/replace
 - date/time conversions and calculations



Transformations

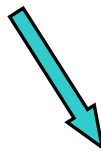
- The obvious (easy) transformations
 - calculations
 - string operations
 - lookup/replace
 - data conversion
 - numeric to character or vice versa
 - numeric dates & times to real date/time fields
- Handling exceptions
 - if, then, else
- Surrogate Keys
 - artificial key, replacing the natural keys
 - provide support for slowly changing dimensions

Surrogate Keys

- Merging Similar tables

Customer File - US	
CUSTNO	CUSTNAME
1001	John Smith
1002	Mary Jones
1003	Chris Anderson
1004	David Perry

Customer File - Canada	
CUSTNO	CUSTNAME
1001	Harry Potter
1002	Jeremy Carr
1003	Penny Hayes
1004	Debbie Thornton



Surrogate key is a sequential number with no correlation to replaced value(s)

Customer File - Data Warehouse			
CUSTNUMBER	CUSTNAME	REGION	OLDNUM
1	John Smith	US	1001
2	Mary Jones	US	1002
3	Chris Anderson	US	1003
4	David Perry	US	1004
5	Harry Potter	CANADA	1001
6	Jeremy Carr	CANADA	1002
7	Penny Hayes	CANADA	1003
8	Debbie Thornton	CANADA	1004

PK

Secondary Index



Surrogate Keys

- Slowly Changing dimensions
 - Each change to a 'key' dimension results in a new customer record
 - Identify changed dimension during ETL
 - Generate new customer record, with 'open' effective-to date
 - Update current record to set effective-to date to yesterday
 - New transactions are written with new surrogate key value

CUSTOMERS table

CUSTNUMBER	CUSTNAME	OLDNUM	GROUP	TERRITORY	EFFECTIVE TO DATE
10105	Acme Flooring	1001	Small Retail	Jenny Brown	3/15/2005
24505	Acme Flooring	1001	Small Retail	Rob McAdam	7/23/2006
33099	Acme Flooring	1001	Major retail	Rob McAdam	12/31/9999

Sales Transactions

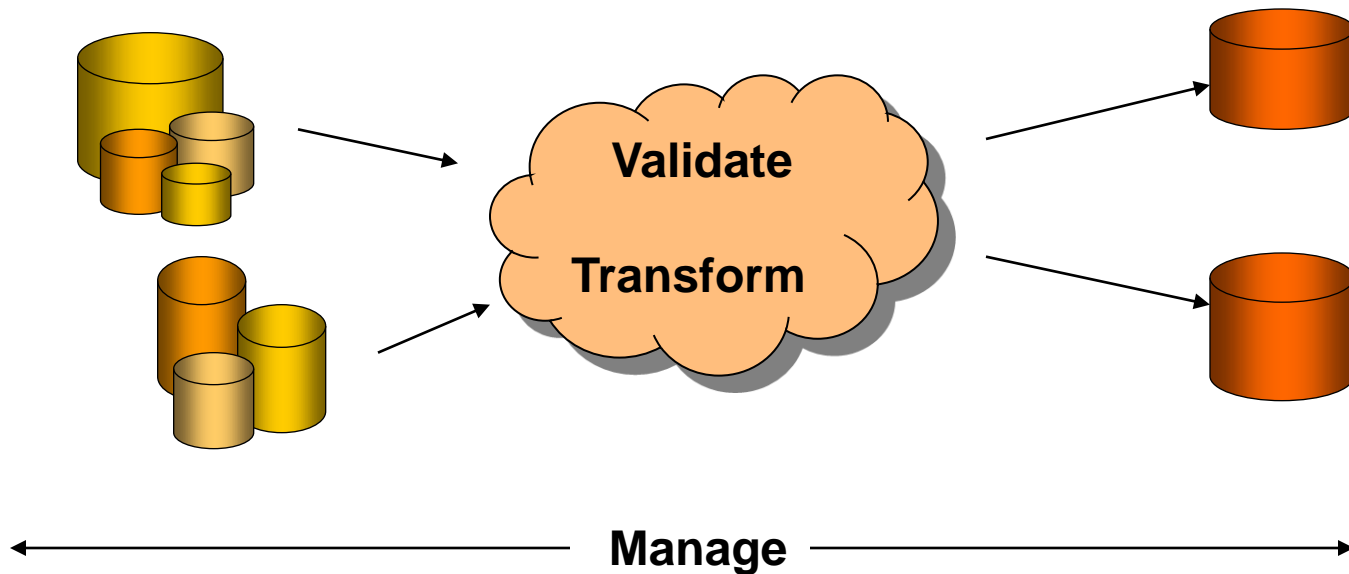
INVOICE	CUSTNUMBER	TOTAL	INVOICE DATE
456895	10105	3300.56	2/10/2005
654905	24505	2049.05	3/13/2006
1010323	33099	200.95	7/30/2007

- If we re-run a report for 2008, the customer is correctly grouped as at the time of the transaction
- Territory sales reports will report against the correct sales rep

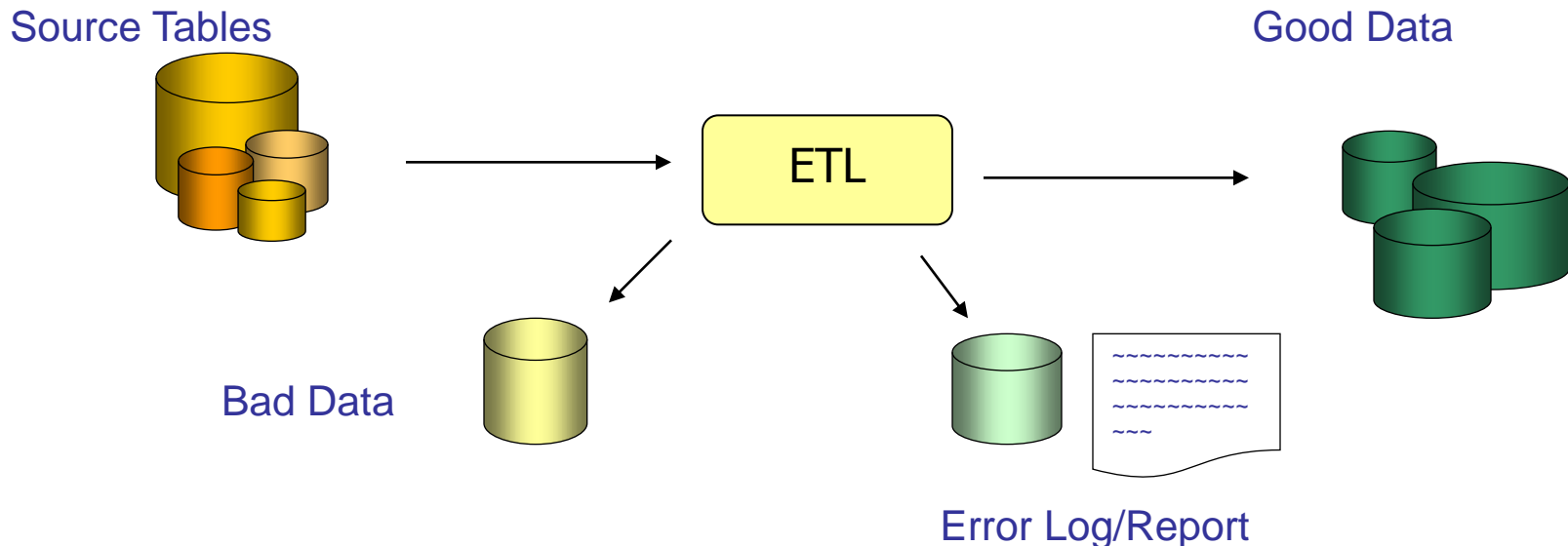
- **Load** the transformed data
 - one or more database tables
 - detail or summary level
 - insert or update



- But... There are two VITAL additional requirements
 - Validate – define business rules
 - Manage – data errors
 - the overall environment



- Data Quality rules imply errors – which implies error reporting
 - stage the 'bad' data
 - do not load it, but just as importantly do not ignore it
 - provide a means to correct it and re-process
 - keep audit trail of errors



■ What is metadata?

- descriptive information about the data sources, ETL processes and target database (data warehouse, data marts etc)
 - Technical metadata – column names, data types, keys etc
 - Business metadata – describes the use and meaning of pieces of data, documents data quality rules etc
 - Administrative metadata – change history, authorities, load statistics, usage statistics etc

■ Why is it important?

- it is the 'roadmap' to the information stored in the data warehouse or data mart tables.
- unlike the majority of end-user applications where access is via program interfaces, some of your users will access the tables directly. They need to know what the data means.

ETL - Build or Buy ?

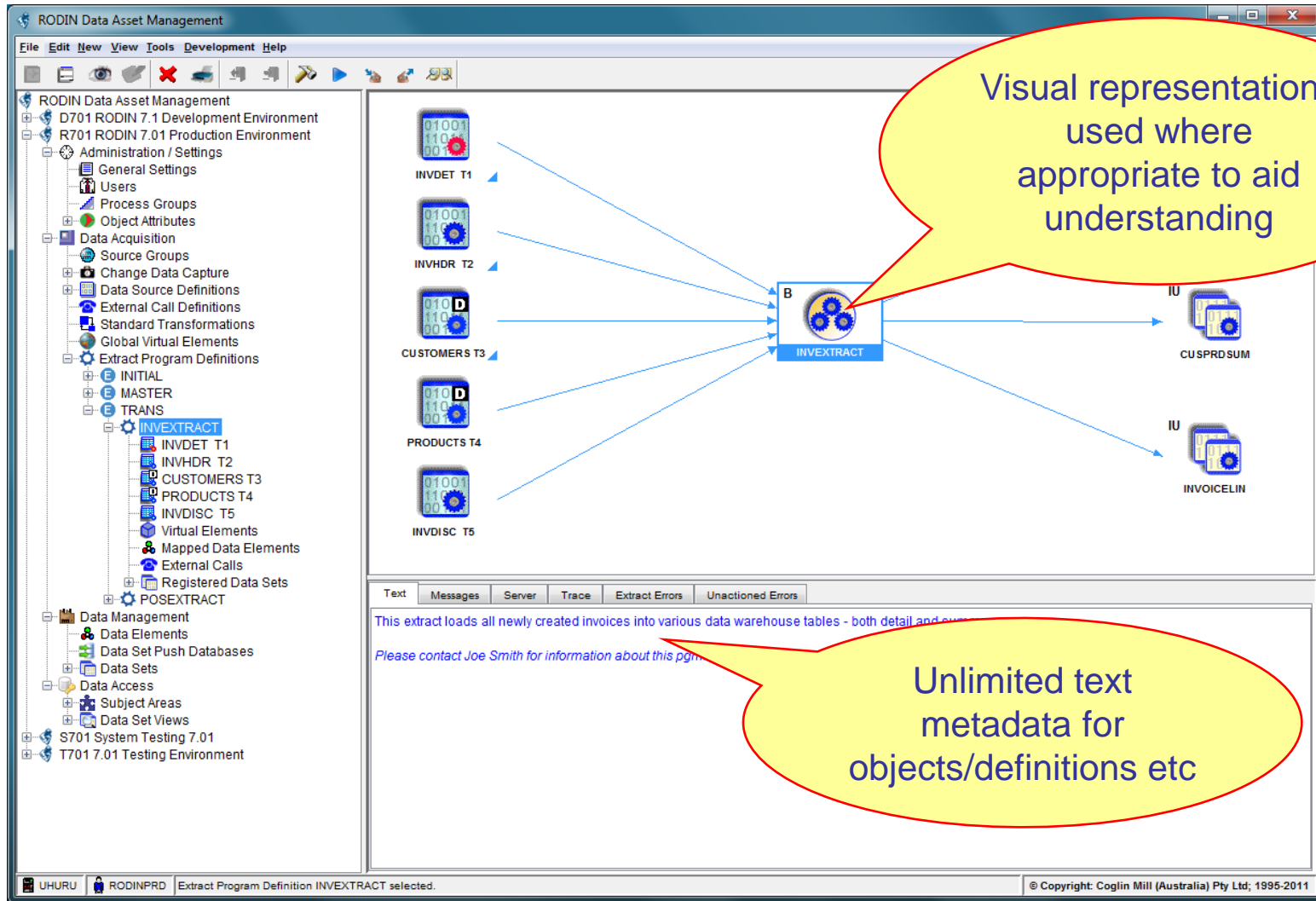
- Should I build my own ETL processes, or buy a tool?
 - do I have the resources?
 - do my people resources have all the necessary skills?
 - do I understand the size of the effort?
 - do I have the time to write a significant amount of code?
 - the answers to these questions is usually NO

- The right ETL tool will
 - dramatically reduce the total cost
 - significantly reduce development time
 - allow you to concentrate on the business requirements
 - provide the environmental support for long term success

ETL Tool Requirements

- The tool should:
 - Be native to your platform (IBM i)
 - critical for good performance
 - Provide remote data access
 - Support time and effort saving concepts such as change data capture from journal images
 - Be highly productive
 - 5 to 10 times increase in productivity compared to hand coding is not uncommon
 - Provide comprehensive error handling and reporting
 - Provide comprehensive auditing
 - Have comprehensive metadata support
 - automatic capture
 - allow export to other tools
 - Include impact analysis tools
 - Provide a visual development environment

RODIN Development Client



The screenshot shows the RODIN Data Asset Management application window. The left pane displays a tree view of the data asset hierarchy, with 'INVEXTRACT' selected under 'Extract Program Definitions'. The main workspace shows a diagram where 'INVEXTRACT' (a central blue box with a gear icon) is the target of arrows from several source objects: 'INVDET T1', 'INVHDR T2', 'CUSTOMERS T3', 'PRODUCTS T4', and 'INVDISC T5'. Arrows also point from 'INVEXTRACT' to two output objects: 'CUSPRD SUM' and 'INVOICE LIN'. A yellow callout bubble points to the 'INVEXTRACT' box with the text: "Visual representations used where appropriate to aid understanding".

Below the diagram, a text pane displays the following message:

Text Messages Server Trace Extract Errors Unactioned Errors

This extract loads all newly created invoices into various data warehouse tables - both detail and summary tables.

Please contact Joe Smith for information about this program.

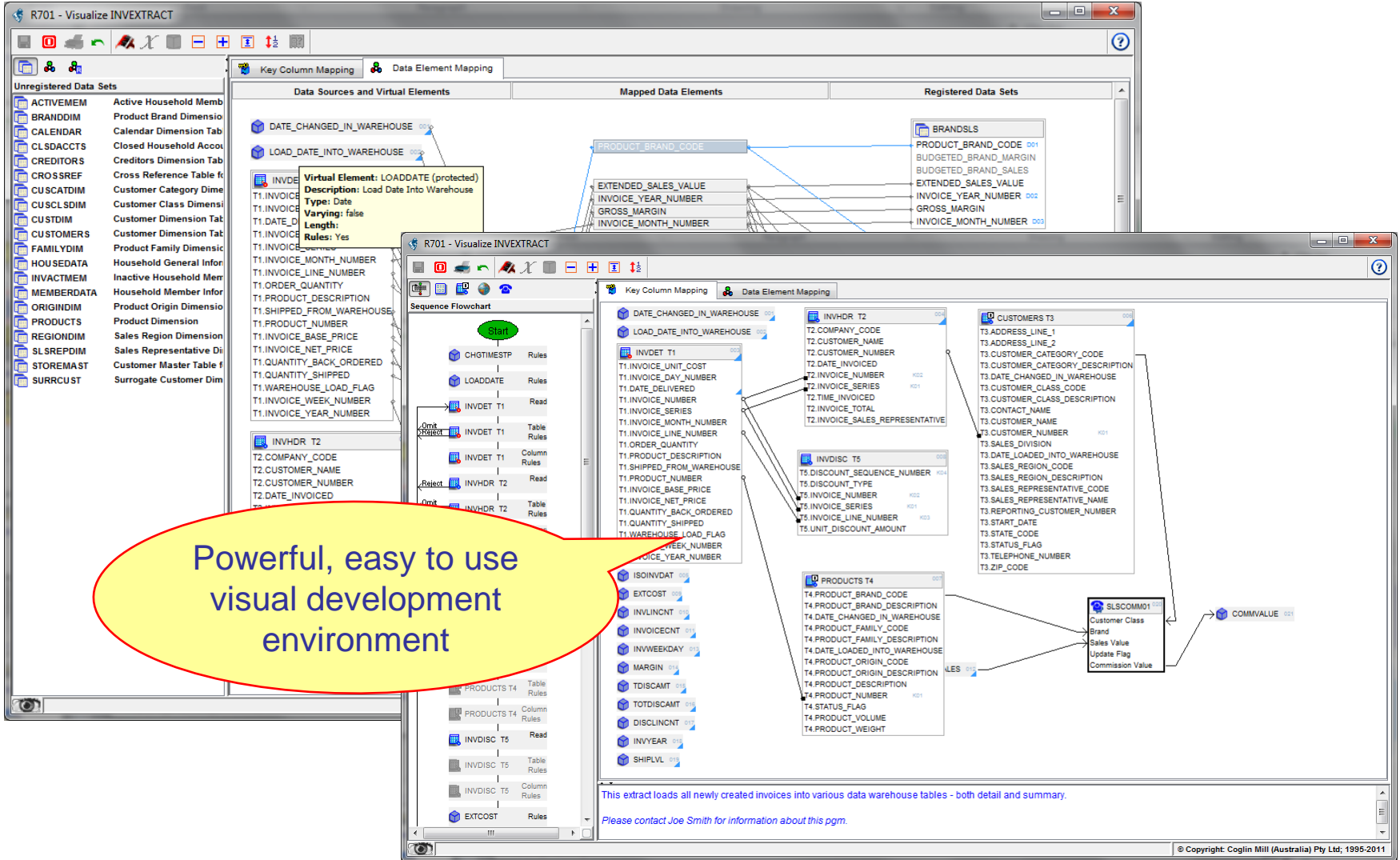
A second yellow callout bubble points to this text with the text: "Unlimited text metadata for objects/definitions etc".

The status bar at the bottom of the window shows: UHURU RODINPRD Extract Program Definition INVEXTRACT selected. © Copyright: Coglin Mill (Australia) Pty Ltd; 1995-2011

Visual representations used where appropriate to aid understanding

Unlimited text metadata for objects/definitions etc

RODIN Development Client



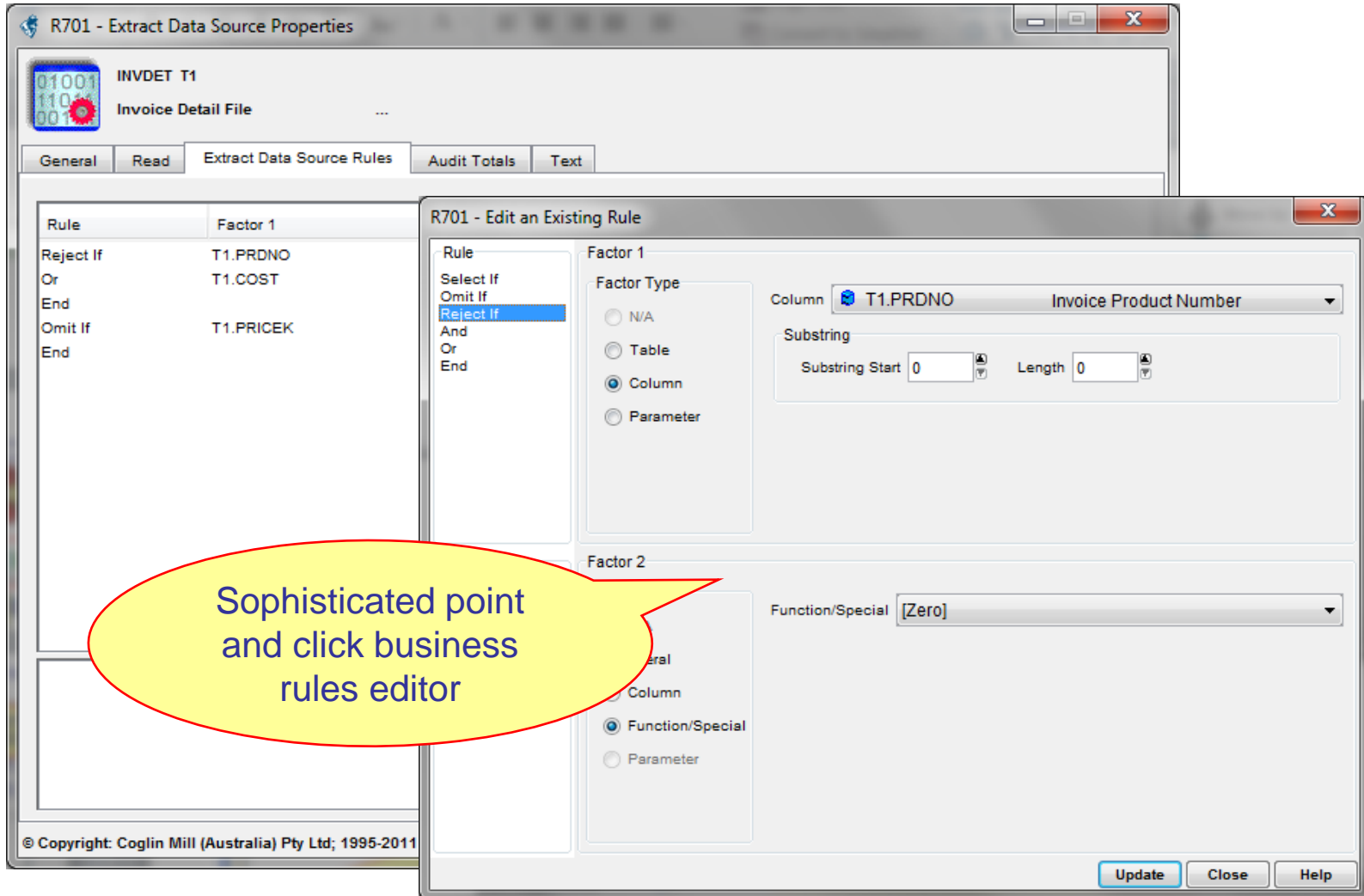
The screenshot displays the RODIN Development Client interface, which is used for visual development of data extracts. The interface is divided into several panes:

- Unregistered Data Sets:** A list of data sets on the left side, including ACTIVEMEM, BRANDDIM, CALENDAR, CLSDACTS, CREDITORS, CROSSREF, CUSCATDIM, CUSCLSDIM, CUSTDIM, CUSTOMERS, FAMILYDIM, HOUSEDATA, INACTIVEMEM, MEMBERDATA, ORIGINDIM, PRODUCTS, REGIONDIM, SLSREPDIM, STOREMAST, and SURRCUST.
- Data Sources and Virtual Elements:** A central pane showing data sources and virtual elements. A virtual element named 'LOADDATE (protected)' is highlighted with a description: 'Description: Load Date Into Warehouse', 'Type: Date', 'Varying: false', 'Length: 8', and 'Rules: Yes'.
- Mapped Data Elements:** A pane showing mapped data elements such as DATE_CHANGED_IN_WAREHOUSE, LOAD_DATE_INTO_WAREHOUSE, PRODUCT_BRAND_CODE, EXTENDED_SALES_VALUE, INVOICE_YEAR_NUMBER, GROSS_MARGIN, and INVOICE_MONTH_NUMBER.
- Registered Data Sets:** A pane showing registered data sets like BRANDSLS, PRODUCT_BRAND_CODE, BUDGETED_BRAND_MARGIN, BUDGETED_BRAND_SALES, EXTENDED_SALES_VALUE, INVOICE_YEAR_NUMBER, GROSS_MARGIN, and INVOICE_MONTH_NUMBER.
- Sequence Flowchart:** A flowchart on the left showing the sequence of data processing steps, including 'Start', 'CHGMTSTP', 'LOADDATE', 'INVDET T1', 'INVHDR T2', 'ISOINVDAT', 'EXTCOST', 'INVLINCNT', 'INVOICECNT', 'INVWEEKDAY', 'MARGIN', 'TDISCAMT', 'TOTDISCAMT', 'DISLINCNT', 'INVYEAR', and 'SHIPVLV'.
- Key Column Mapping and Data Element Mapping:** Two panes on the right showing the mapping of columns and data elements between different data sets and tables.

A yellow callout bubble with a red border contains the text: "Powerful, easy to use visual development environment".

At the bottom of the interface, there is a note: "This extract loads all newly created invoices into various data warehouse tables - both detail and summary. Please contact Joe Smith for information about this pgm."

The copyright notice at the bottom right reads: "© Copyright: Coglin Mill (Australia) Pty Ltd, 1995-2011".



R701 - Extract Data Source Properties

INVDET T1
Invoice Detail File

General Read Extract Data Source Rules Audit Totals Text

Rule	Factor 1
Reject If	T1.PRDN0
Or	T1.COST
End	
Omit If	T1.PRICEK
End	

R701 - Edit an Existing Rule

Rule
Select If
Omit If
Reject If
And
Or
End

Factor 1

Factor Type

N/A
 Table
 Column
 Parameter

Column T1.PRDN0 Invoice Product Number

Substring

Substring Start 0 Length 0

Factor 2

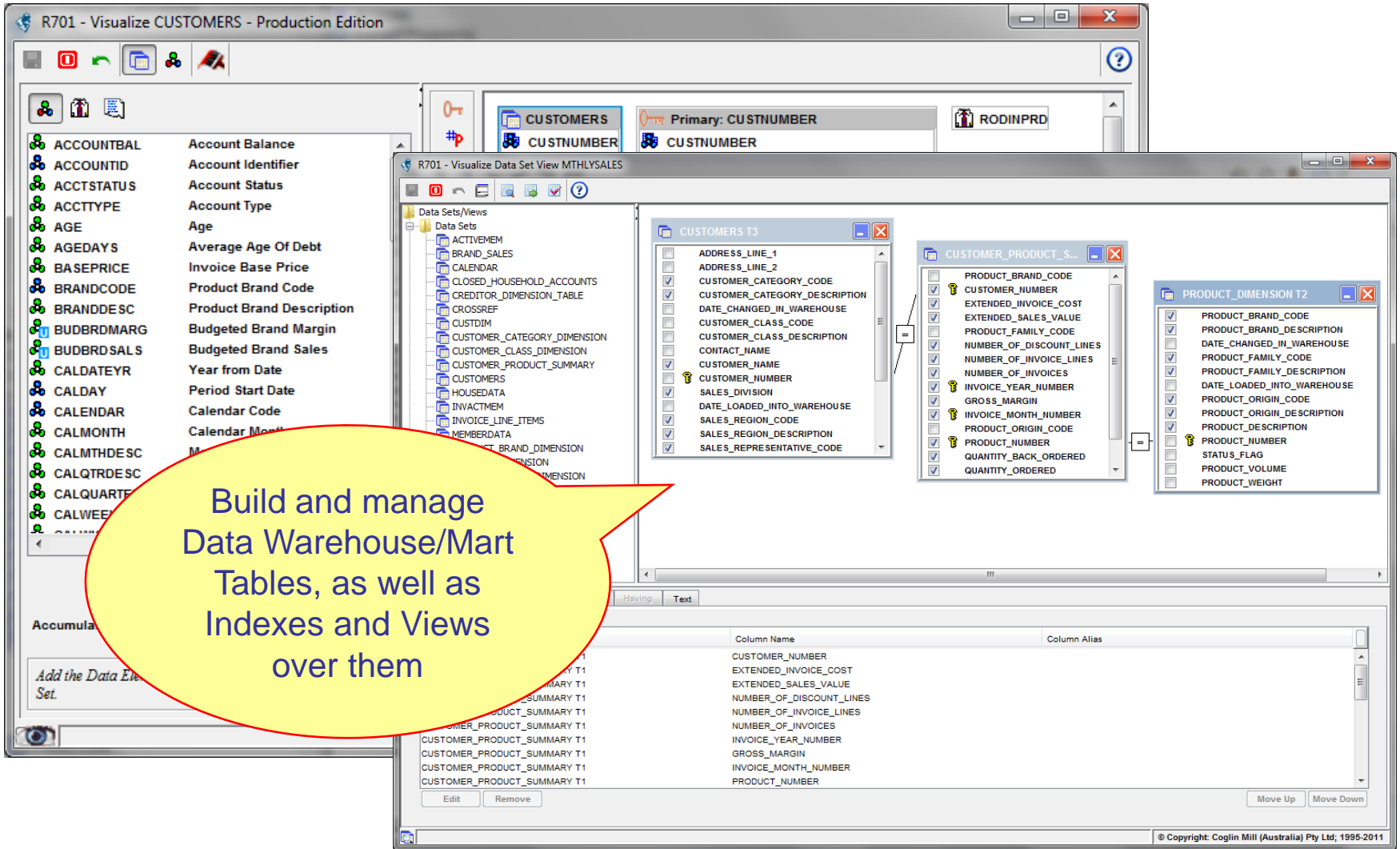
Function/Special [Zero]

Update Close Help

© Copyright: Coglin Mill (Australia) Pty Ltd; 1995-2011

Sophisticated point and click business rules editor

RODIN Development Client



The screenshot displays the RODIN Development Client interface with several windows open:

- R701 - Visualize CUSTOMERS - Production Edition:** Shows a list of data sets and views on the left, including ACCOUNTBAL, ACCOUNTID, ACCTSTATUS, ACCTTYPE, AGE, AGEDAYS, BASEPRICE, BRANDCODE, BRANDESC, BUDBRDMARG, BUDBRDSALS, CALDATEYR, CALDAY, CALENDAR, CALMONTH, CALMTHDESC, CALQTRDESC, CALWEEK, and CALYEAR.
- R701 - Visualize Data Set View MTHLYSALES:** Shows a tree view of data sets and views, including ACTIVEMEM, BRAND_SALES, CALENDAR, CLOSED_HOUSEHOLD_ACCOUNTS, CREDITOR_DIMENSION_TABLE, CROSSREF, CUSTDIM, CUSTOMER_CATEGORY_DIMENSION, CUSTOMER_CLASS_DIMENSION, CUSTOMER_PRODUCT_SUMMARY, CUSTOMERS, HOUSEDATA, INACTIVEMEM, INVOICE_LINE_ITEMS, MEMBERDATA, PRODUCT_BRAND_DIMENSION, and PRODUCT_FAMILY_DIMENSION.
- CUSTOMERS T3:** A table view showing columns such as ADDRESS_LINE_1, ADDRESS_LINE_2, CUSTOMER_CATEGORY_CODE, CUSTOMER_CATEGORY_DESCRIPTION, DATE_CHANGED_IN_WAREHOUSE, CUSTOMER_CLASS_CODE, CUSTOMER_CLASS_DESCRIPTION, CONTACT_NAME, CUSTOMER_NUMBER, SALES_DIVISION, DATE_LOADED_INTO_WAREHOUSE, SALES_REGION_CODE, SALES_REGION_DESCRIPTION, and SALES_REPRESENTATIVE_CODE.
- CUSTOMER_PRODUCT_S...:** A table view showing columns such as PRODUCT_BRAND_CODE, CUSTOMER_NUMBER, EXTENDED_INVOICE_COST, EXTENDED_SALES_VALUE, PRODUCT_FAMILY_CODE, NUMBER_OF_DISCOUNT_LINES, NUMBER_OF_INVOICE_LINES, NUMBER_OF_INVOICES, INVOICE_YEAR_NUMBER, GROSS_MARGIN, INVOICE_MONTH_NUMBER, PRODUCT_ORIGIN_CODE, PRODUCT_NUMBER, QUANTITY_BACK_ORDERED, and QUANTITY_ORDERED.
- PRODUCT_DIMENSION T2:** A table view showing columns such as PRODUCT_BRAND_CODE, PRODUCT_BRAND_DESCRIPTION, DATE_CHANGED_IN_WAREHOUSE, PRODUCT_FAMILY_CODE, PRODUCT_FAMILY_DESCRIPTION, DATE_LOADED_INTO_WAREHOUSE, PRODUCT_ORIGIN_CODE, PRODUCT_ORIGINDESCRIPTION, PRODUCT_DESCRIPTION, PRODUCT_NUMBER, STATUS_FLAG, PRODUCT_VOLUME, and PRODUCT_WEIGHT.

A yellow callout bubble with a red border contains the text: "Build and manage Data Warehouse/Mart Tables, as well as Indexes and Views over them".

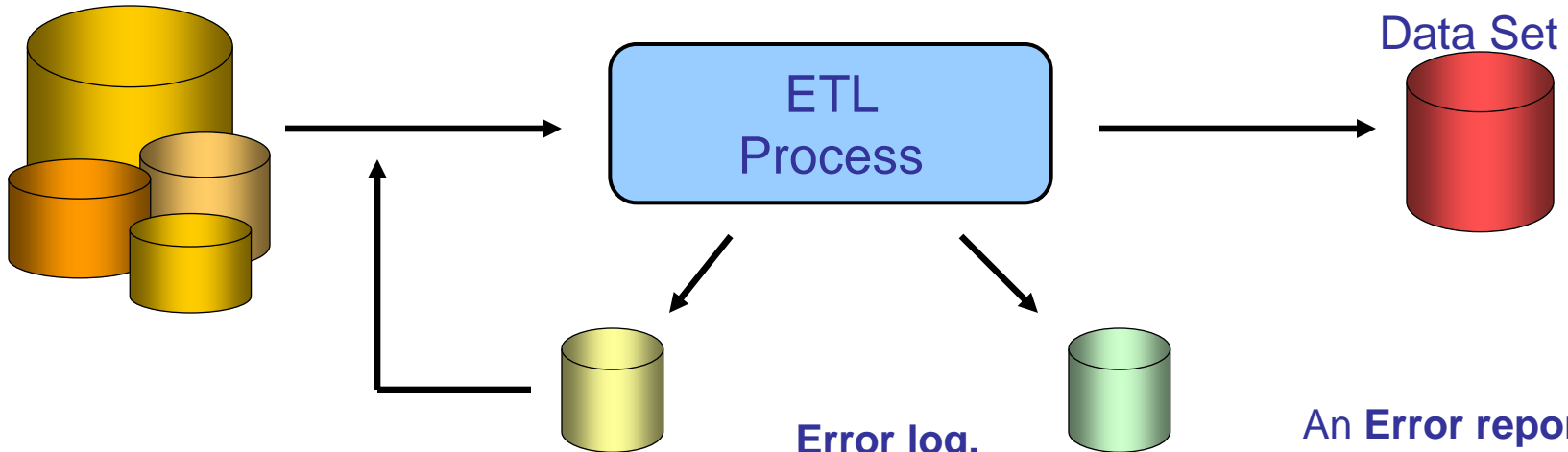
At the bottom of the interface, there is a table with the following columns: Column Name and Column Alias. The table contains the following data:

Column Name	Column Alias
CUSTOMER_NUMBER	
EXTENDED_INVOICE_COST	
EXTENDED_SALES_VALUE	
NUMBER_OF_DISCOUNT_LINES	
NUMBER_OF_INVOICE_LINES	
NUMBER_OF_INVOICES	
INVOICE_YEAR_NUMBER	
GROSS_MARGIN	
INVOICE_MONTH_NUMBER	
PRODUCT_NUMBER	

© Copyright: Coglin Mill (Australia) Pty Ltd, 1995-2011

RODIN Error Management

Source Tables



Error suspense table.

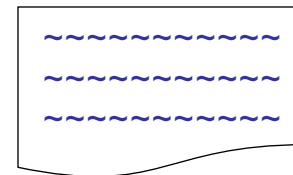
Rejected data is written to this table. After correction, this data can be re-processed by the extract in **error recovery mode**.

Error log.

Reason for error is written out to this table.

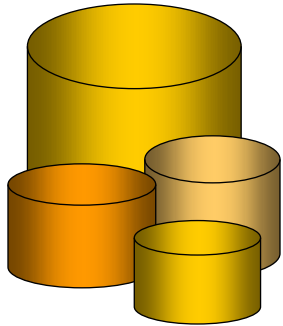
An Error report

is produced at end of processing if any errors occurred.

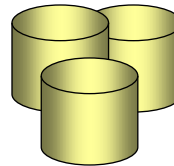
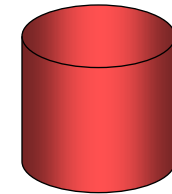


RODIN Auditing

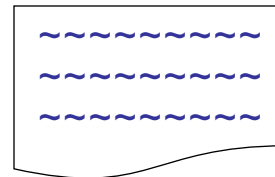
Source Tables



Data Set



Audit Tables



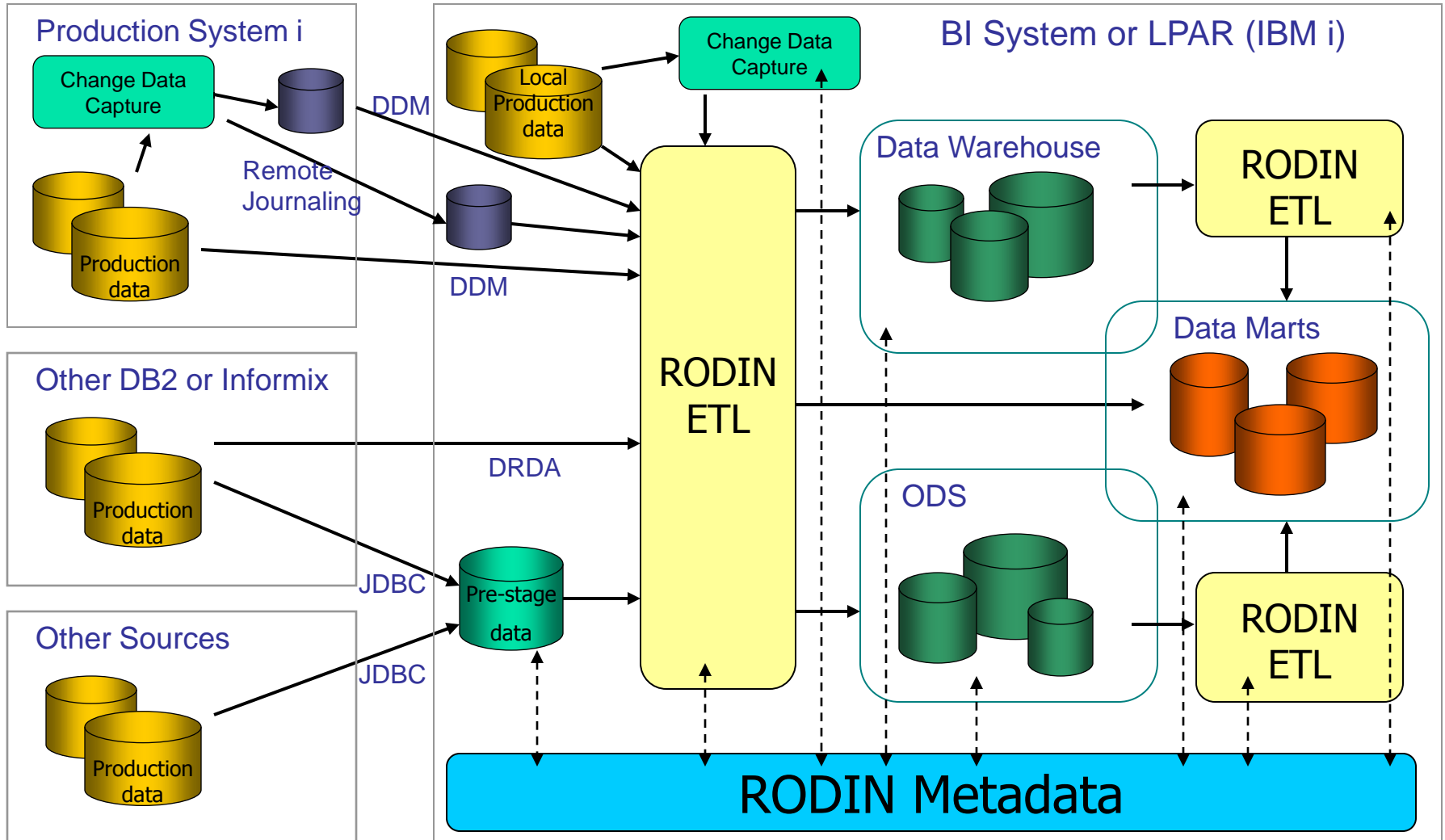
Audit report produced at end of extract.

Audit information includes

- Number of source rows read, omitted and rejected.
- Hash totals for up to 3 numeric source columns.
- Number of data sets updated and inserts/updates to each.
- Elapsed time for all jobs.
- etc.....

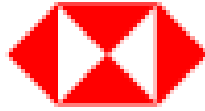
Audit information retained in meta-data indefinitely

RODIN ETL Architecture



A Few RODIN Success Stories

HSBC



ESTES

BANK OF GUAM



FISERV



HONDA

SEIKO
OPTICAL PRODUCTS OF AMERICA INC.

RBS
The Royal Bank of Scotland Group

tri counties bank

Welcome to
Tree of Life
The World's Leading Marketer of Natural and Specialty Foods

CAT
Logistics



Alro Steel
Metals • Industrial Supplies • Plastics

PIRAEUS
BANK

FOCUS ON THE
Family

Virgin
money



LIFE OF THE SOUTH.

Discovery
CHANNEL
STORE

Office DEPOT
Taking Care of Business

Crosman
CORPORATION

Source.
St Bank
Your partners from the first®

TIRE RACK
.com

SCHOLASTIC

RODIN

Data Engineering

EXTRACT ⇨ TRANSFORM ⇨ LOAD ⇨ MANAGE

For more information:

call **1-866-RODIN-DW**

visit www.thinkrodin.com