ETL

Extracting

Designing the ETL plan

- The **logical data map** is provided by the data warehouse architect and is the specification for the ETL team to create the physical ETL jobs.
- This document is sometimes referred to as the **data lineage report**.
- The logical data map is the foundation of the **metadata** that is eventually presented to **quality-assurance tester**s and ultimately to **end users** to describe exactly what processing is done between the source system and the data warehouse.

Target					Source				Transformation
Table Name	Column Name	Data Type	Table Type	SCD Type	Database Name	Table Name	Column Name	Data Type	
EMPLOYEE_DIM	EMPLOYEE_KEY	NUMBER	Dimension	1				NUMBER	Surrogate key.
EMPLOYEE_DIM	EMPLOYEE_ID	NUMBER	Dimension	1	HR_SYS	EMPLOYEES	EMPLOYEE_ID	NUMBER	Natural Key for employee in HR system
EMPLOYEE_DIM	BIRTH_COUNTRY_NAME	VARCHAR2(75)	Dimension	1	HR_SYS	COUNTRIES	NAME	VARCHAR2(75)	select c.name from employees e, states s, countries c where e.state_id = s.state_id and s.country_id = c.country
EMPLOYEE_DIM	BIRTH_STATE	VARCHAR2(75)	Dimension	1	HR_SYS	STATES	DESCRIPTION	VARCHAR2(255)	select s.description from employees e, states s where e.state_id = s.state_id
EMPLOYEE_DIM	DISPLAY_NAME	VARCHAR2(75)	Dimension	1	HR_SYS	EMPLOYEES	FIRST_NAME	VARCHAR2(75)	select initcap(salutation) ||' '||initcap(first_name) ||' '|| initcap(last_name) from employee
EMPLOYEE_DIM	BIRTH_DATE	DATE	Dimension	1	HR_SYS	EMPLOYEES	DOB	DATE	trunc(DOB)
EMPLOYEE_DIM	SALUTATION	VARCHAR2(12)	Dimension	1	HR_SYS	EMPLOYEES	SALUTATION	VARCHAR2(12)	initcap(salutation)
EMPLOYEE_DIM	FIRST_NAME	VARCHAR2(30)	Dimension	1	HR_SYS	EMPLOYEES	FIRST_NAME	VARCHAR2(30)	initcap(first_name)
EMPLOYEE_DIM	LAST_NAME	VARCHAR2(30)	Dimension	1	HR_SYS	EMPLOYEES	LAST_NAME	VARCHAR2(30)	initcap(last_name)
EMPLOYEE_DIM	MARITAL_STATUS	VARCHAR2(12)	Dimension	2	HR_SYS	MARITAL_STATUS	DESCRIPTION	VARCHAR2(12)	select nvl(m.name,'Unknown') from employee e marital_status m where e.marital_status_id = m.marital_status_id
EMPLOYEE_DIM	DIVERSITY_CATEGORY	VARCHAR2(30)	Dimension	1	HR_SYS	EMPLOYEES	EEO_CLASS	VARCHAR2(30)	decode(eeo_class,null, 'Not Stated', decode(eeo_class,'N', 'Not Stated',eeo_class))
EMPLOYEE_DIM	GENDER	VARCHAR2(12)	Dimension	1	HR_SYS	EMPLOYEES	SEX	VARCHAR2(12)	nvl(sex, 'Unknown')
EMPLOYEE_DIM	EMPLOYEE_STATUS	VARCHAR2(24)	Dimension	1	HR_SYS	EMPLOYEES	STATUS	VARCHAR2(24)	select es.name from employee e employee_status es where e.employee_status_id = m.employee_status_id
EMPLOYEE_DIM	POSITION_CODE	VARCHAR2(12)	Dimension	2	HR_SYS	POSITIONS	POSITION_CODE	VARCHAR2(12)	select p.code from employees e, positions p where p.position_id = e.position_id
EMPLOYEE_DIM	POSITION_CATEGORY	VARCHAR2(30)	Dimension	2	HR_SYS	POSITIONS	POSITION_CATEGORY	VARCHAR2(30)	select p.category from employees e, positions p where p.position_id = e.position_id
EMPLOYEE_DIM	HIRE_DATE	DATE	Dimension	1	HR_SYS	EMPLOYEES	DATE_HIRED	DATE	trunc(date_hired)
EMPLOYEE_DIM	DEPARTMENT_CODE	VARCHAR2(12)	Dimension	2	HR_SYS	DEPARTMENTS	CODE	VARCHAR2(12)	select d.code from employee p, employee_department pd, departments d where p.employee_id= pd.employee_id and pd.department_id = d.department_id
EMPLOYEE_DIM	DEPARTMENT_NAME	VARCHAR2(75)	Dimension	2	HR_SYS	DEPARTMENTS	DESCRIPTION	VARCHAR2(75)	select d.description from employee p, employee_department pd, departments d where p.employee_id= pd.employee_id and pd.department_id = d.department_id
EMPLOYEE_DIM	PART_TIME_FLAG	VARCHAR2(1)	Dimension	1	HR_SYS	EMPLOYEES	PERCENTAGE	VARCHAR2(1)	select decode(sign(percentage-100),-1,'Y','N') from employee
EMPLOYEE_CONTRACT_FACT	EFFECTIVE_DATE_KEY	NUMBER	Fact	N/A	DW_PROD, HR_SYS	DATE_DIM, EMPLOYEE_CONTRACT	DATE_KEY	NUMBER	where employee_contract.eff_date = dw_prod.date_dim.cal_date
EMPLOYEE_CONTRACT_FACT	END_DATE_KEY	NUMBER	Fact	N/A	DW_PROD, HR_SYS	DATE_DIM, EMPLOYEE_CONTRACT	DATE_KEY	NUMBER	where employee_contract.end_date = dw_prod.date_dim.cal_date
EMPLOYEE_CONTRACT_FACT	CURRENCY_KEY	NUMBER	Fact	N/A	DW_PROD, HR_SYS	CURRENCY_DIM, EMPLOYEE_CONTRACT	CURRENCY_KEY	NUMBER	where employee_contract.currency_code = dw_prod.currency_dim.currency_code
EMPLOYEE_CONTRACT_FACT	RATE_TYPE_KEY	NUMBER	Fact	N/A	DW_PROD, HR_SYS	RATE_TYPE_DIM, EMPLOYEE_CONTRACT	RATE_TYPE_KEY	NUMBER	where employee_contract.rate_type_id = dw_prod.rate_type_dim.rate_type_id
EMPLOYEE_CONTRACT_FACT	PROJECT_KEY	NUMBER	Fact	N/A	DW_PROD, HR_SYS	PROJECT_DIM, EMPLOYEE_CONTRACT	PROJECT_KEY	NUMBER	where employee_contrac.project_code = dw_prod.project_dim.project_code
EMPLOYEE_CONTRACT_FACT	EMPLOYEE_ROLE_KEY	NUMBER	Fact	N/A	DW_PROD, HR_SYS	EMPLOYEE_ROLE_DIM, EMPLOYEE_CONTRACT	EMPLOYEE_ROLE_KEY	NUMBER	where employee_contract.employee_role = employee_role_dim.contractor_role_name
EMPLOYEE_CONTRACT_FACT	CONTRACT_TYPE_KEY	NUMBER	Fact	N/A	DW_PROD, HR_SYS	CONTRACT_TYPE_DIM, EMPLOYEE_CONTRACT	CONTRACT_TYPE_KEY, CONTRACT_TYPE_ID	NUMBER	where dw_prod.employee_contract.contract_type = contract_type_dim.contract_type_id
EMPLOYEE_CONTRACT_FACT	CONTRACT_NUMBER	NUMBER	Fact	N/A	DW_PROD, HR_SYS	EMPLOYEE_CONTRACT	CONTRACT_NUMBER	NUMBER	Degenerate Dimension
EMPLOYEE_CONTRACT_FACT	RATE_AMOUNT_LOCAL	NUMBER	Fact	N/A	DW_PROD, HR_SYS	EMPLOYEE_CONTRACT	AMOUNT	NUMBER	sum(amount)
EMPLOYEE_CONTRACT_FACT	RATE_AMOUNT_USD	NUMBER	Fact	N/A	DW_PROD, HR_SYS	EMPLOYEE_CONTRACT	AMOUNT	NUMBER	select ec.amount * avg(cc.conversion_rate) from employee_contract ec, currency_conversion cc where ec.currency = cc.from_currency and cc.effective_Date between ec.effective_date and ec.end_date group by ec.amount

Components of the Logical Data Map

- Target table name: The physical name of the table as it appears in the data warehouse
- Target column name. The name of the column in the data warehouse table
- Table type. Indicates if the table is a fact, dimension, or sub dimension (outrigger).
- SCD (slowly changing dimension) type. For dimensions, this component indicates a Type-1, -2, or -3 slowly changing dimension approach. This indicator can vary for each column in the dimension.
- Source database. The name of the instance of the database where the source data resides. This component is usually the connect string required to connect to the database. It can also be the name of a file as it appears in the file system. In this case, the path of the file would also be included.
- Source table name. The name of the table where the source data originates. There will be many cases where more than one table is required. In those cases, simply list all tables required to populate the relative table in the target data warehouse.
- Source column name. The column or columns necessary to populate the target. Simply list all of the columns required to load the target column. The associations of the source columns are documented in the transformation section.
- Transformation. The exact manipulation required of the source data so it corresponds to the expected format of the target. This component is usually notated in SQL or pseudo-code.

Logical Data Map

- The table type gives us our queue for the ordinal position of our data load processes—first dimensions, then facts.
- Columns in the logical data mapping document are sometimes combined. For example, the source database, table name, and column name could be combined into a single source column.
- The information within the concatenated column would be delimited with a period, for example, ORDERS.STATUS.STATUS CODE.
- Know exactly which columns have historic relevance and the strategy required for capturing the history before you begin the development of the load process.
- The value in this column may change over time.

Logical Data Map

- Notice that the data types between the source and target for STATE get converted from 255 characters to 75 characters.
- Even though the data-scale reduction might be supported by the data-analysis documentation, should any future values with more than 75 characters be created, you would potentially lose the data.
- Moreover, some ETL tools would actually abort or fail the entire process with this kind of data overflow error.
- Notice the transformation notation for the STATE does not explicitly define this data conversion—the conversion is implied.
- Implied conversions are common and notorious for sneaking up and destroying your processes. To avoid calamity, the ETL team must assume responsibility for explicitly handling these types of implied data conversions.

Transformation

- The transformation can contain anything from the absolute solution to nothing at all.
- Most often, the transformation can be expressed in SQL.
- The SQL may or may not be the complete statement. Quite often, it is the segment of the code that cannot otherwise be implied from the other elements in the mapping, such as the SQL WHERE clause.
- In other cases, the transformation might be a method that is not SQL specific and is explained in plain English, like instructions to preload from a flat file or to base the load transformation on criteria outside of the database or to reject known data anomalies into a reject file.
- If the transformation is blank, this means the mapping is a straight load, from source-to-target, with no transformation required.

Building the Logical Data Map

Analysis of the source system

- The complete logical data mapping cannot exist until the source systems have been identified and analyzed.
- The analysis of the source system is usually broken into two major phases:
  - Data discovery phase
    - Once you understand what the target needs to look like, you need to identify and examine the data sources
  - Data content analysis / anomaly detection phase
    - A data anomaly is a piece of data that does not fit into the domain of the rest of the data it is stored with.

Data Discovery Phase

- Collecting and Documenting Source Systems
- Keeping Track of the Source Systems
- Determining the System-of-Record
- Analyzing the Source System: Using Findings from Data Profiling

Collecting and Documenting Source Systems

- The source systems are usually established in various pieces of documentation, including interview notes, reports, etc.
- The ETL team has to work with the team’s system and business analysts to track down appropriate source systems.
- In large organizations, the ETL team must find the data source of each user group.
- Typical organizations have countless distinct systems. It is the ETL team’s responsibility to keep track of the systems discovered and investigate their usefulness as a data warehouse source.

System Analyst

- A systems analyst is an information technology (IT) professional who specializes in analyzing, designing and implementing information systems.
- Systems analysts assess the suitability of information systems in terms of their intended outcomes and liaise with end users, software vendors and programmers in order to achieve these outcomes.
- A systems analyst is a person who uses analysis and design techniques to solve business problems using information technology.
- Systems analysts may serve as change agents who identify the organizational improvements needed, design systems to implement those changes, and train and motivate others to use the systems.

Business Analyst

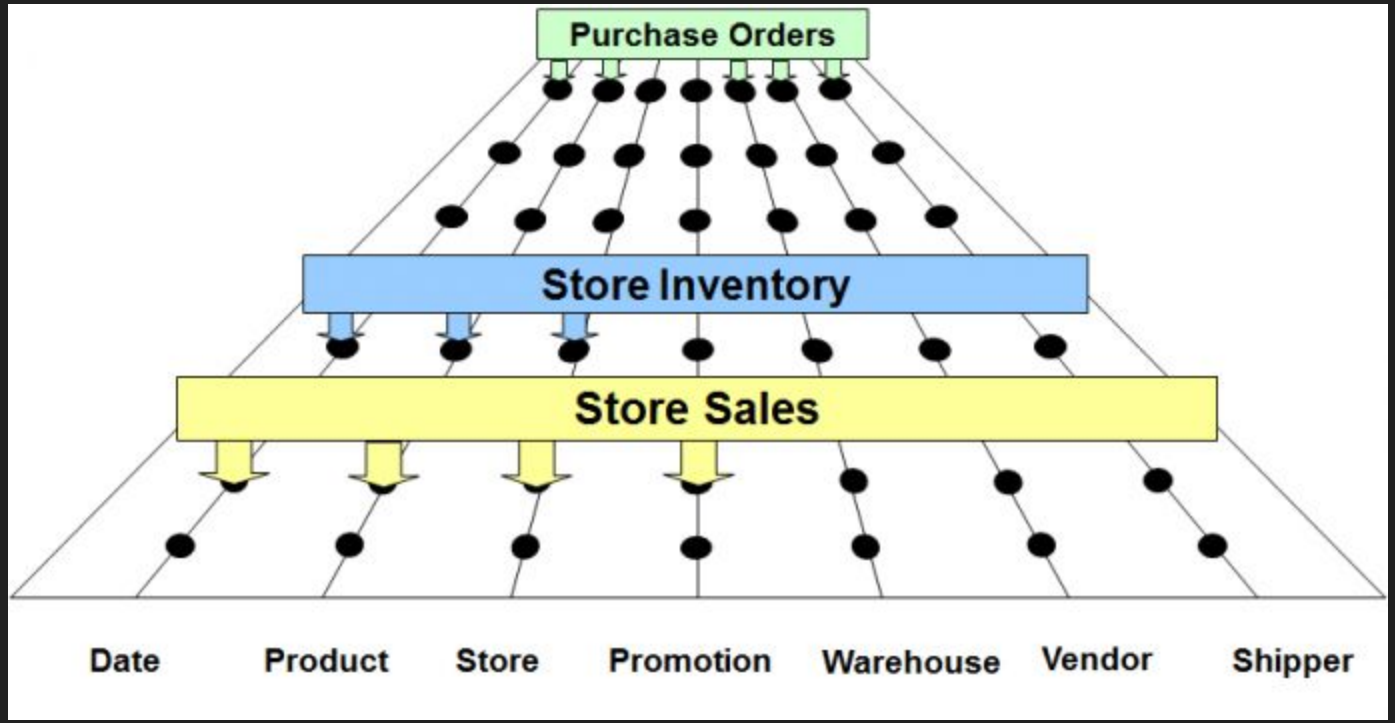
- Business analysts, also called management analysts, evaluate how an organization operates, and recommend approaches to the management team that might improve productivity and reduce costs.
- To do this, a business analyst gathers and examines data, uses that data to develop models, examines company financials, and researches customer habits.
- A business analyst uses this information to recommend ways to improve systems or processes to company leadership.

Keeping Track of the Source Systems

Subject Area	Interface Name	Business Name	Priority	Department/ Business Use	Business Owner	Technical Owner	DBMS	Platform	Daily User Count	DB Size (GB)	Transactions/ Day	Comments
Human Resources	PeopleSoft	HR Information	1	Human Resource Mgmt	Daniel Bumis	Margaret Kartin	Oracle	AIX on RS/6000	850	3	8,000	Everything revolves around HR
Finance	Oracle Financials	Oracle Financials	2	General Accounting	Mabel Karr	Edmund Bykel	Oracle	AIX on S80	400	60	20,000	Driven sponsor ; Department likes slicing & dicing data
Materials Management	WMS	Warehouse Management System	3	Manufacturing Panning & Control	Annabel Giutu	Christina Hayim	Oracle	AIX on S80	350	200	5,000	Initiative to cut Inventory of slow movers; Need analytical support.
Operations	BOMM	Bill of Materials Management	4	Production Planning	Lucy Kard	George Cimemblis	Oracle	AIX on RS/6000	60	8	TBD	Need analysis of "kits", Want to build kits in-house
Marketing	IMCM	Internal Marketing & Campaign Management	5	Marketing	Elizabeth Impzar	Brian Bridalurg	SQL Server	W2K on Compaq	50	350	Varies	Needs Data Mining Capabilities
Human Resources	Positions Website	Positions Website	5	Human Resource Mgmt	Florence Iyzar	Andrew Aoanthal	SQL Server	W2k on Compaq	50	1	10,000	Manager is new, eager for DW
Purchaing	PAS	Purchaing & Acquision System	6	Purchaing Department	Guy Croendil	Sybil Mai	SQL Server	NT on Compaq	75	4	1,500	
Customer Service	CSS	Customer Service System	7	Customer Care	Bernard Beais	Arthur Aticcio	Notes/ Domino	NT on Compaq	275	6	1,500	Plans to move application to Oracle next year.
Operations	QCS	Quality Control System	8	Quality Control	Thomas Fryar	Elias Caye	SQL Server	NT on Compaq	450	12	500	Needs failure ratio analytics by product, by vendor, by year
Sales	SFA	Sales Force Automation	9	Sales	Anthony Aran	Emma Inalonm	DB2	AS/400	1200	10	2,500	Politically challenging; Interim reporting system in place

- Subject area. Typically the name of the data mart that this system feeds
- Interface name. The name of the transaction application that the source system supports
- Business name. The name the system is commonly referred to by the business users.
- Priority. A ranking or ordinal position used to determine which data sources are to be loaded first. The priority is based on the data warehouse bus matrix.
- Department/Business use. The primary department using the database, for example, Accounting, Human Resources, etc. and the business use, for example, Inventory Control, Client tracking, etc.

BUSINESS PROCESSES	COMMON DIMENSIONS						
	Date	Product	Warehouse	Store	Promotion	Customer	Employee
Issue Purchase Orders	X	X	X				
Receive Warehouse Deliveries	X	X	X				X
Warehouse Inventory	X	X	X				
Receive Store Deliveries	X	X	X	X			X
Store Inventory	X	X		X			
Retail Sales	X	X		X	X	X	X
Retail Sales Forecast	X	X		X			
Retail Promotion Tracking	X	X		X	X		
Customer Returns	X	X		X	X	X	X
Returns to Vendor	X	X		X			X
Frequent Shopper Sign-Ups	X			X		X	X


Purchase Orders
Store Inventory
Store Sales
Date
Product
Store
Promotion
Warehouse
Vendor
Shipper

- Business owner. The person or group to contact for issues or questions related to the use of the application or database. This person or group is typically the data steward for the subject area.
- Technical Owner. Typically the DBA or IT project manager responsible for maintaining the database
- DBMS. The source database management system name. In most cases, it will be a relational database such as Oracle, DB2, or Sybase. It can also be nonrelational data stores like Lotus Notes or VSAM.
- Production server/OS. When known, this column includes the physical name of the server where the database lives. It also includes the operating system. You need this column when designing OS level scripts for your ETL. For example, you cannot use UNIX shell scripts when the server is operating on NT.

- # Daily users. Gives you an idea of how many employees in the organization the data is exposed to.
- DB size. The DBA should be able to provide this information. Knowing the raw size of the source data can help you determine the ETL priorities and efforts. Generally speaking, the larger databases tend to be higher on the priority lists because performance is usually lacking when large tables or several joined tables are queried in the transaction system.
- DB complexity. The number of tables and view objects in the system
- # Transactions per day. Estimate that gives you an indication of the capacity requirements for the incremental load process
- Comments. Usually used for general observations found while researching the database. It may include notes about future version releases of the database or reasons why it is or isn’t a system-of-record for certain entities

Data Content Analysis

- Common anomalies that you should be aware of include:

1. NULL values.

- Check for NULL values in every foreign key in the source database.
- When NULL values are present, you must outer join the tables.
    - An outer join returns all of the rows regardless of whether there is a matching value in the joined table.
- If the NULL data is not in a foreign key column but is in a column required by the business for the data warehouse, you must get a business rule on how the NULL data should be handled.
- We don’t like to store NULL values in the data warehouse unless it is indeed an unknown measure.
- Whenever possible, create default values to replace NULL values.

Data Content Analysis

2. Dates in non-date fields

- Dates can come in various formats, literally containing different values and having the exact same meaning.
- Database systems support various formats for display purposes but store them in a single standard format (for that specific database).
- But there are many situations where dates are stored in text fields, especially in legacy applications.
- Following slide contains a sample of the possible variations of the same date that can be found when the date is stored in a text field:

- 13-JAN-22
- January 13, 2022
- 01-13-2022
- 13/01/2022
- 01/13/2022 2:24 PM
- 01/13/202214:24:49
- 20220113
- 13012022

Collecting Business Rules in the ETL Process

- The business rules required for the data modeling team are quite different from those required by the ETL team.
- For example, the data modeling definition of the status dimension might be something like:
  - Status Code—The status is a four-digit code that uniquely identifies the status of the product.
  - The code has a short description, usually one word, and a long description, usually one sentence.
- Conversely, the ETL definition of status might be expressed like this:
  - Status Code—The status is a four-digit code.
  - However, there are legacy codes that were only three digits that are still being used in some cases.
  - All three-digit codes must be converted to their four-digit equivalent code.
  - The name of the code may have the word OBSOLETE embedded in the name.
  - OBSOLETE needs to be removed from the name and these obsolete codes must have an obsolete flag set to ‘Y’.
  - The description should always be in sentence case, regardless of the case used when entered into the source system.

Integrating Heterogeneous Data Sources

Analogy: Corporate Merger

- During corporate mergers, one or more companies are joined with other similar (or dissimilar) companies.
- When mergers occur, the business must decide which company is the surviving company and which companies get consumed
  - The surviving company in a merger is the company that takes over the rights and responsibilities of the firms that undergo the merger.
- Sometimes, negotiations are made when the parent company recognizes value in certain practices and techniques of its subsidiaries and incorporates those practices in its modified organization.
- The result of a successful corporate merger is a cohesive organization that has a single business interest.
- This requires a commitment to aligning terminology (dimension attributes) and aligning key performance indicators (facts in fact tables).

Loading Conformed Dimensions

- When you build a data warehouse, the most direct form of data integration is the implementation of conformed dimensions.
- In the data warehouse, conformed dimensions are the cohesive design that unifies disparate data systems scattered throughout the enterprise.
- In the following slides, we'll go through the steps for loading conformed dimensions in a disparate source system environment.

Data Profiling

- As per Jack Olson, data profiling is a necessary precursor to designing any kind of system to use that data.
- As he puts it: “Data profiling employs analytic methods for looking at data for the purpose of developing a thorough understanding of the content, structure, and quality of the data. A good data profiling system can process very large amounts of data, and with the skills of the analyst, uncover all sorts of issues that need to be addressed.”

System of Record

- A system of record (SOR) is an ISRS (information storage and retrieval system) that is the authoritative source for a particular data element in a system containing multiple sources of the same element.
- To ensure data integrity, there must be one and only one system of record for a given piece of information.
- **Systems of engagement** are the programs and applications that workers interact in their daily work lives.
- Systems of engagement are decentralized, their design is based on the needs of a given business unit, and are built for frontline team.

1. Identify the source systems

- During the data-profiling phase of the construction of the logical data mapping, the data warehouse team must work together to detect the various sources of your target dimensions and facts.
- The data warehouse architect should uncover most of the potential sources of each element in the data warehouse and attempt to appoint a system-of-record for each element.
- The system-of-record is considered the ultimate source for the data being loaded.

2. Understand the source systems

- Understand all of the attributes of all of the entities of all of the systems under consideration.
- This phase declares the reliability of the source system for the elements under scrutiny.
- During this phase of the project, the assignment of the system-of-record can actually be reassigned if data-quality issues persist or if reliability of the data is problematic for any reason.

3. Create record matching logic

- Your next objective is to design the matching algorithm to enable entities across the disparate systems to be joined.
- Sometimes, the matching algorithm is as simple as identifying the primary key of the various customer tables.
- But in many cases, disparate systems do not share primary keys.
- Perhaps, there is a social security number that can be used to uniquely identify your customer or maybe you need to combine the last name, e-mail address, and telephone number.
- The various business areas must be involved with and approve of your final matching logic.
- Record-matching logic must be consistent with the various legislated privacy rules, such as HIPAA in the health care arena.

4. Establish survivorship rules

- Once your system-of-record has been identified and the matching logic has been approved, you can establish the surviving record when data collisions occur in your ETL process.
- This means that if you have a customer table in your accounts receivable and sales systems, the business must decide which system has overriding power when attributes overlap.

5. Establish nonkey attribute business rules

- Dimensions are typically sourced from various tables and columns within a system.
- Moreover, many source systems usually contain different attributes that ultimately feed into a target dimension.
- For instance, a list of departments probably originated in your HR department; however, the accounting code for that department probably comes from your financial system.
- Even though the HR system may be the system-of-record, certain attributes may be deemed more reliable from other systems.
- Assigning business rules for nonkey attributes is especially important when attributes exist in several systems but not in the system-of-record.
- In those cases, documentation and publication of the data lineage metadata is crucial.

6. Load conformed dimension

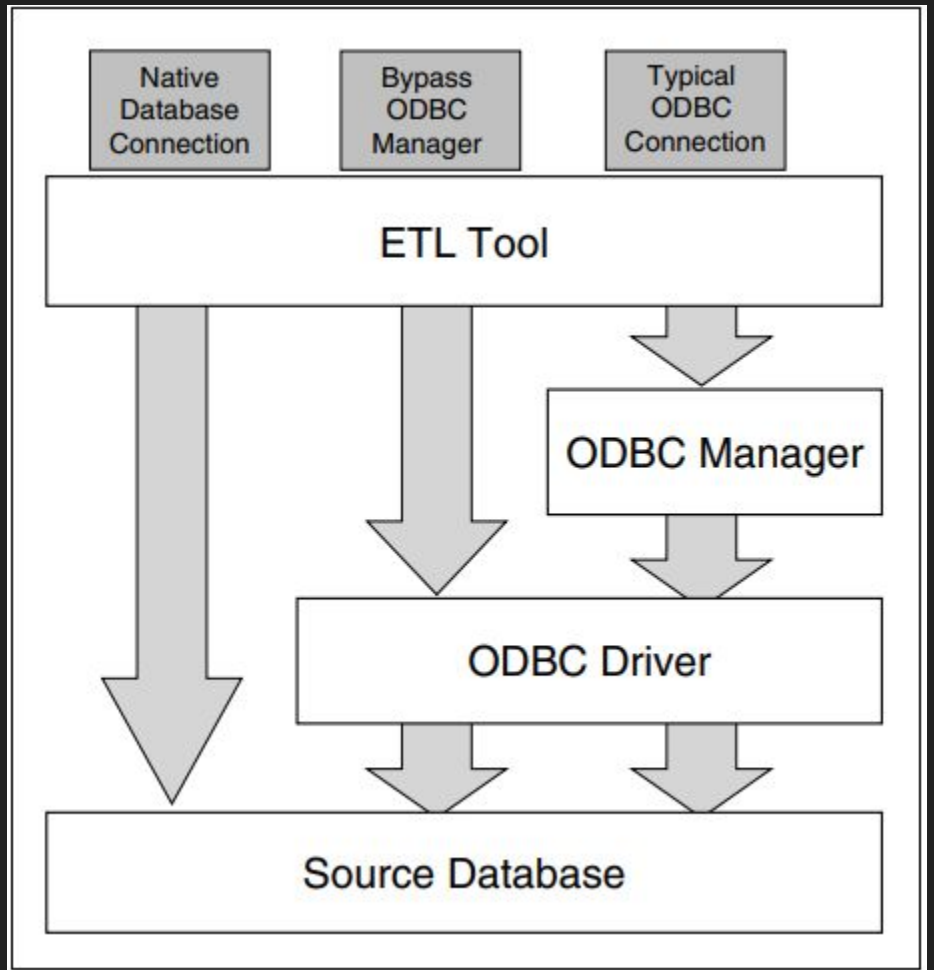
- The final task of the data-integration process is to physically load the conformed dimension.
- This step is where you consider the slowly changing dimension (SCD) type and update late-arriving data as necessary.

The Challenge of Extracting from Disparate Platforms

Connecting to Diverse Sources through ODBC

- Open Database Connectivity (ODBC) was created to enable users to access databases from their Windows applications.
- The original intention for ODBC was to make applications portable, meaning that if an application's underlying database changed—say from DB2 to Oracle—the application layer did not need to be recoded and compiled to accommodate the change.
  - Instead, you simply change the ODBC driver, which is transparent to the application.
- ODBC drivers are available for most DBMS on any platform.
- You can also use ODBC to access flat files.

The topology of ODBC in the ETL process



The topology of ODBC in the ETL process.

- For the ETL process to utilize data via ODBC, two layers are added between the ETL system and the underlying database.
  - ODBC manager.
    - The ODBC manager is a program that accepts SQL from the ETL application and routes it to the appropriate ODBC driver.
    - It also maintains the connection between the application and the ODBC driver.
  - ODBC driver.
    - The ODBC driver is the real workhorse in the ODBC environment.
    - The ODBC driver translates ODBC SQL to the native SQL of the underlying database.

- The drawback to ODBC’s flexibility is that it comes at a performance cost.
- Once you use ODBC you might lose much DBMS specific functionality.
- Particular non-ANSI standard SQL commands are not accepted by the ODBC manager because it needs to maintain an open solution.
- For the highest performance and native DBMS functionality you should look first to a native database driver.
- ODBC can provide a common format gateway to certain troublesome data sources that are otherwise not easily extracted.

Mainframe Sources

- In many large companies, much of the day-to-day business data is processed and stored on mainframe systems (and certain minicomputer systems, such as the IBM AS/400) and integrating data from these systems into the data warehouse involves some unique challenges.
- COBOL remains the dominant programming language used on mainframe computers, and the file layout for data is described in COBOL copybooks.
- A copybook defines the field names and associated data types for a mainframe data file.
- As with other flat files you encounter in your ETL process, only two data types exist in mainframe flat files: text and numeric.
- However, numeric values are stored in a variety of ways that you need to understand to accurately process.
- Likewise, dates are stored simply as strings of numbers (or text) and typically require transformation to be stored in date columns in the data warehouse.

A simple copybook that describes an employee record

		Len	Positions		
01 EMP-RECORD.					
05 FIRST-NAME	PIC X(10).	10	1	-	10
05 MIDDLE-INITIAL	PIC X.	1	11	-	11
05 LAST-NAME	PIC X(15).	15	12	-	26
05 SSN	PIC X(9).	9	27	-	35
05 EMP-DOB.		8	36	-	43
10 DOB-YYYY	PIC 9(4).	4	36	-	39
10 DOB-MM	PIC 9(2).	2	40	-	41
10 DOB-DD	PIC 9(2).	2	42	-	43
05 EMP-ID	PIC X(9).	9	44	-	52
05 HIRE-DATE.		8	53	-	60
10 HIRE-YYYY	PIC 9(4).	4	53	-	56
10 HIRE-MM	PIC 9(2).	2	57	-	58
10 HIRE-DD	PIC 9(2).	2	59	-	60
05 TERM-DATE.		8	61	-	68
10 TERM-YYYY	PIC 9(4).	4	61	-	64
10 TERM-MM	PIC 9(2).	2	65	-	66
10 TERM-DD	PIC 9(2).	2	67	-	68
05 TERM-REASON_CODE	PIC X(2).	2	69	-	70

- 70-byte, fixed length record that describes a simple employee record.
- The field names are preceded by level numbers.
- Nesting of level numbers is used to group related fields.
- Text and numeric data types are denoted using the PIC clauses. PIC X denotes text fields, while PIC 9 means the field is numeric.
- Field lengths are specified with numbers following the type. For example, the clause PIC 9(4) indicates a four-byte numeric field, whereas PIC X(15) indicates a 15-byte text field.

EBCDIC

- **EBCDIC**, in full **extended binary-coded decimal interchange code**, data-encoding system, developed by IBM and used mostly on its computers, that uses a unique eight-bit binary code for each number and alphabetic character as well as punctuation marks and accented letters and nonalphabetic characters.
- EBCDIC differs in several respects from Unicode and ASCII, the most widely used systems of encoding text, dividing the eight bits for each character into two four-bit zones, with one zone indicating the type of character, digit, punctuation mark, lowercase letter, capital letter, and so on, and the other zone indicating the value—that is, the specific character within this type.

EBCDIC to ASCII

- It’s automatic, assuming you use File Transfer Protocol (FTP) to transfer the data from the mainframe to your data warehouse platform.
- An FTP connection requires two nodes—a host and a client.
- When an FTP connection is made between systems, the FTP client identifies its operating system environment to the FTP host, and the host determines whether any translation is required when transferring data between the two systems.
- So when an FTP connection is made between a mainframe and a UNIX or Windows system, the FTP host translates mainframe data from EBCDIC to ASCII as it transfers the data.
- In addition, FTP adds the special line feed and carriage return characters used to designate the end of a line (or record) of data on UNIX and Windows.
- FTP also translates from ASCII to EBCDIC if the data movement is from UNIX or Windows to the mainframe.

Flat Files

Field Name	Length	Start	End	Type	Comments
Record Type	2	1	2	AlphaNumeric	Type of record can be either 'H' for header or 'D' for detail record.
SSN	9	3	11	Numeric	Employee's Social Security Number.
First Name	20	12	31	AlphaNumeric	First name of employee
Middle Initial	1	32	32	AlphaNumeric	Middle initial of employee
Last Name	20	33	52	AlphaNumeric	Last name of employee
Name Suffix	5	53	57	AlphaNumeric	Jr, Sr, III, etc.
Birth Date	8	58	65	Numeric	Employee's data of birth "YYYYMMDD".
Status Code	6	66	71	Numeric	Employee's Status ('A','R', 'T', etc).
Office Code	2	72	73	Numeric	The code of the employees branch office.
Department Code	2	74	75	Numeric	The code of the employees department within his/her office.
Position Code	2	76	77	Numeric	The code of the employees position in the organization.
Filler	1	78	78	AlphaNumeric	Filler space.
Add Date	8	79	86	Numeric	The date the record was add to the system.
Modified Date	8	87	94	Numeric	The date the record last modified.

- In most ETL tools, you have to manually input the file layout of the flat file once.
- After the layout is entered, the tool remembers the layout and expects that same layout each time it interacts with the actual flat file.
- If the file layout changes or the data shifts off of its assigned positions, the ETL process must be programmed to raise an error.
- Most ETL tools have a delimited file wizard that, once the developer indicates the actual delimiter characters, scans the flat file, or a sample of it, to detect the delimiters within the file and specify the file layout.
- Most often, the first row of delimited files contains its column names.
- The ETL tool should be intelligent enough to recognize the column names supplied in the first row to assign logical column names in the metadata layer and then ignore the row during all subsequent data processing.

Web Log Sources

- Virtually every company in the world has a Web site.
- Beneath each Web site are logs—Web logs—that record every object either posted to or served from the Web server.
- Web logs are important because they reveal the user traffic on the Web site.
- Understanding the behavior of users on your Web site is as valuable as following a customer around a store and recording his or her every move.
- Imagine how much more organized your store can be and how many opportunities you can have to sell more merchandise if you know every move your customers make while navigating your store.
- Web logs provide that information.
- The activity of parsing Web logs and storing the results in a data mart to analyze customer activity is known as ***clickstream data warehousing***.

ERP System Sources

- ERP systems are designed to be an integrated enterprise solution that enables every major entity of the enterprise, such as sales, accounting, human resources, inventory, and production control, to be on the same platform, database, and application framework.
- As noble as the effort to be an all-inclusive solution is, it's very rare to see an entire enterprise use only an ERP system to run a company.
- Because of the sheer number of tables, attributes, and complexity of a typical ERP implementation, it is a mistake to perform ETL on these systems like any other transaction source system.
- Performing system and data analysis from scratch is cost and time prohibitive and leaves much room for error.
- If your data warehouse sources are from an existing ERP system, it is best to acquire someone with vast experience of the underlying database structure of your specific ERP system as well as the business objectives of the application.
- To help you along, many of the major ETL vendors now offer ERP adapters to communicate with the popular ERP systems.
- They can help you navigate the metadata in these systems and make sense of the application.

Part 3: Extracting Changed Data

- During the **initial load**, capturing changes to data content in the source data is unimportant because you are most likely extracting the entire data source or a potion of it from a predetermined point in time.
- But once that load is complete, the ability to capture data changes in the source system instantly becomes priority number one.
- The ETL team is responsible for capturing data-content changes during the **incremental load**.
- Desires and hopes are dictated by the users, and the realities of the feasibility of those hopes are revealed by the source system DBA team.

Detecting Changes Using Audit Columns

- In most cases, the source system contains audit columns.
- Audit columns are appended to the end of each table to store the date and time a record was added or modified.
- Audit columns are usually populated via database triggers fired off automatically as records are inserted or updated.
- Sometimes, for performance reasons, the columns are populated by the front-end application instead of database triggers.
- When these fields are loaded by any means other than database triggers, you must pay special attention to their integrity.
- You must analyze and test each of the columns to ensure that it is a reliable source to indicate changed data.
- If you find any NULL values, you must to find an alternative approach for detecting change.

Detecting Changes Using Audit Columns(2)

- The most common environment situation that prevents the ETL process from using audit columns is when the fields are populated by the front-end application and the DBA team allows back-end scripts to modify data.
- If this is the situation in your environment, you face a high risk that you will eventually miss changed data during your incremental loads.
- A preventative measure to minimize your risk is to stipulate that all back-end scripts be validated by a quality-assurance team that insists and tests that the audit fields are populated by the script before it is approved.
- Once you are confident that audit columns are dependable, you need a strategy for utilizing them.
- There are various methods to implement the utilization of audit columns to capture changes to data.
- All of the methods have the same logical objective: to compare the last modified date and time of each record to the maximum date and time that existed during the previous load and take all those that are greater.

Detecting Changes Using Audit Columns(3)

- Essentially, the process selects the maximum date and time from the create date and last modified date columns.
- Some last modified columns are updated upon insertion and with each change to the record.
- Others are left NULL upon insertion and updated only with changes after the record has already been inserted.
- When the last modified date is not populated, you must default it with an arbitrary old date in order to not lose new records.
- In cases where the rows in the fact table are inserted but never updated, you can simply select records from the source system where the create date and time is greater than the maximum date and time of the previous load and ignore the last modified date column.
- Since fact tables and dimension tables can be sourced from many different tables and systems, and since fact tables consist only of foreign keys and measures, you do not store the audit dates in the fact table directly.
- You need to create an ETL last-change table that captures each source table and the maximum date time found in the source system audit columns at the time of each extract.

Database Log Scraping

- Log scraping effectively takes a snapshot of the database redo log at a scheduled point in time (usually midnight) and scours it for transactions that affect the tables you care about for your ETL load.
- Scraping the log for transactions is probably the messiest of all techniques. It's not rare for transaction logs to blow-out, meaning they get full and prevent new transactions from occurring.
- When this happens in a production-transaction environment, the knee-jerk reaction for the DBA responsible is to empty the contents of the log so the business operations can resume.
- But when a log is emptied, all transactions within them are lost.

Database Log Scraping

- If you've exhausted all other techniques and find log scraping is your last resort for finding new or changed records, persuade the DBA to create a special log to meet your specific needs.
- You presumably need transactions only for a few specific tables out of the hundreds in the source database.
- Those tables can populate your dedicated log via insert and update triggers.

Timed Extracts

- Select all of the rows where the date in the Create or Modified date fields equal SYSDATE-1, meaning you've got all of yesterday's records.
- Problems:
  - Time-based data selection loads duplicate rows when it is restarted from mid process failures.
  - This means that manual intervention and data cleanup is required if the process fails for any reason.
  - Meanwhile, if the nightly load process fails to run and misses a day, a risk exists that the missed data will never make it into the data warehouse.
- Unless your ETL process is extremely straightforward and the data volume is exceptionally small, avoid loading data based purely on time.

Process of Elimination

- Process of elimination preserves exactly one copy of each previous extraction in the staging area for future use.
- During the next run, the process takes the entire source table(s) into the staging area and makes a comparison against the retained data from the last process.
- Only differences (deltas) are sent to the data warehouse.
- Albeit not the most efficient technique, the process of elimination is the most reliable of all incremental load techniques for capturing changed data.
- Because the process makes a row-by-row comparison, looking for changes, it’s virtually impossible to miss any data.
- This technique also has the advantage that rows deleted from the source can be detected.
- These deleted rows are sometimes missed by other techniques.

Current and Previous Load tables

- Create two tables: previous load and current load.
- The initial process bulk loads into the current load table.
- Since change detection is irrelevant during the initial load, the data continues on to be transformed and loaded into the ultimate target fact table.
- When the process is complete, it renames the current load table to previous load, and creates an empty current load table.
- Since none of these tasks involve database logging, they are very fast.
- The next time the load process is run, the current load table is populated.
- Select the current load table MINUS the previous load table.
- Transform and load the result set into the data warehouse. Upon completion, drop the previous load table and rename the current load table to previous load.
- Finally, create an empty current load table.

Extraction Tips

- Constrain on indexed columns.
  - Work with the DBA to ensure all of the columns in your WHERE clause are indexed in the source system; otherwise you will probably provoke a relation scan of the entire production database.
- Retrieve the data you need.
  - The optimal query returns exactly what you need. You shouldn’t retrieve an entire table and filter out unwanted data later in the ETL tool. One situation that might break this rule is if the transaction system DBA refuses to index columns needed to constrain the rows returned in your query. Another exception is when you are forced to download the entire source database to search for the deltas.

Extraction tips(2)

- Use DISTINCT sparingly.
  - The DISTINCT clause is notoriously slow.
  - Finding the balance between performing a DISTINCT during the extract query versus aggregating or grouping the results in your ETL tool is challenging and usually varies depending on the percentage of duplicates in the source.
- Use SET operators sparingly.
  - UNION, MINUS, and INTERSECT are SET operators.
  - These, like DISTINCT, are notoriously slow.
  - It’s understood that sometimes these operators cannot be avoided.
  - A tip is to use UNION ALL instead of UNION. UNION performs the equivalent of a DISTINCT, slowing the process. The hitch is that UNION ALL returns duplicates, so handle with care.
- Use HINT as necessary.
  - Most databases support the HINT keyword.
  - You can use a HINT for all kinds of things, but most importantly to force your query to use a particular index.
  - This capability is especially important when you are using an IN or OR operator, which usually opts for scanning a full table scan rather than using indexes, even when usable indexes exist.

Extraction tips(3)

- Avoid NOT.
  - If at all possible, avoid non-equi constraints and joins. Whether you use the keyword NOT or the operators ‘< >’, your database will most likely opt to scan a full table rather than utilize indexes.
- Avoid functions in your where clause.
  - This is a difficult one to avoid, especially when constraining on dates and such. Experiment with different techniques before committing to use of a function in your WHERE clause. Try using comparison keywords instead of functions whenever possible. For example:
  - LIKE 'J%' instead of SUBSTR('LAST_NAME',1,1) = 'J'
  - EFF_DATE BETWEEN '01-JAN-2002' AND '31-JAN-2002' instead of TO_CHAR(EFF_DATE, 'YYY-MON') = '2002-JAN'

Detecting Deleted or Overwritten Fact Records at the Source

- Measurement (fact) records deleted or overwritten from source systems can pose a very difficult challenge for the data warehouse if no notification of the deletion or overwrite occurs.
- Since it is usually infeasible to repeatedly reextract old transaction records, looking for these omissions and alterations, the best we can offer are the following procedures:
    - Negotiate with the source system owners, if possible, explicit notification of all deleted or overwritten measurement records.
    - Periodically check historical totals of measurements from the source system to alert the ETL staff that something has changed. When a change is detected, drill down as far as possible to isolate the change.

- In cases of deleted or modified fact records, rather than just performing a deletion or update in the data warehouse, we prefer that a new record be inserted that implements the change in the fact by canceling or negating the originally posted value.
- In many applications, this will sum the reported fact to the correct quantity (if it is additive) as well as provide a kind of audit trail that the correction occurred. In these cases, it may also be convenient to carry an extra administrative time stamp that identifies when the database actions took place.