ETL Process in Data Warehouse

Manoj Kumar

Outline

- ETL
- Extraction
- Transformation
- Loading

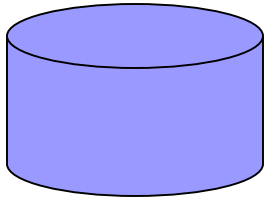
ETL Overview

- Extraction Transformation Loading – ETL
- To get data out of the source and load it into the data warehouse – simply a process of copying data from one database to other
- Data is extracted from an OLTP database, transformed to match the data warehouse schema and loaded into the data warehouse database
- Many data warehouses also incorporate data from non-OLTP systems such as text files, legacy systems, and spreadsheets; such data also requires extraction, transformation, and loading
- When defining ETL for a data warehouse, it is important to think of ETL as a process, not a physical implementation

ETL Overview

- ETL is often a complex combination of process and technology that consumes a significant portion of the data warehouse development efforts and requires the skills of business analysts, database designers, and application developers
- It is not a one time event as new data is added to the Data Warehouse periodically – monthly, daily, hourly
- Because ETL is an integral, ongoing, and recurring part of a data warehouse
  - Automated
  - Well documented
  - Easily changeable

ETL Staging Database

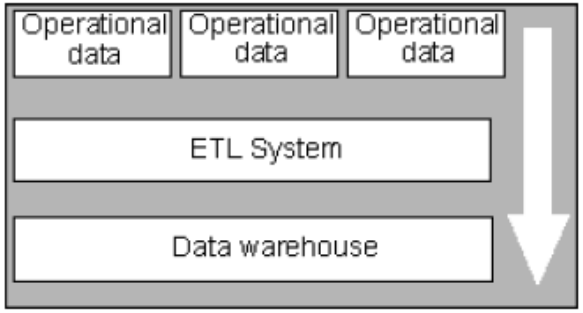


- ETL operations should be performed on a relational database server separate from the source databases and the data warehouse database
- Creates a logical and physical separation between the source systems and the data warehouse
- Minimizes the impact of the intense periodic ETL activity on source and data warehouse databases

Extraction

Part of ETL

Extraction



- The integration of all of the disparate systems across the enterprise is the real challenge to getting the data warehouse to a state where it is usable
- Data is extracted from heterogeneous data sources
- Each data source has its distinct set of characteristics that need to be managed and integrated into the ETL system in order to effectively extract data.

Extraction

- ETL process needs to effectively integrate systems that have different:
  - DBMS
  - Operating Systems
  - Hardware
  - Communication protocols
- Need to have a logical data map before the physical data can be transformed
- The logical data map describes the relationship between the extreme starting points and the extreme ending points of your ETL system usually presented in a table or spreadsheet

Target			Source			Transformation
Table Name	Column Name	Data Type	Table Name	Column Name	Data Type	

- The content of the logical data mapping document has been proven to be the critical element required to efficiently plan ETL processes
- The table type gives us our queue for the ordinal position of our data load processes—first dimensions, then facts.
- The primary purpose of this document is to provide the ETL developer with a clear-cut blueprint of exactly what is expected from the ETL process. This table must depict, without question, the course of action involved in the transformation process
- 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

Extraction in Phases

- The analysis of the source system is usually broken into two major phases:
  - The data discovery phase
  - The anomaly detection phase

Extraction - Data Discovery Phase

- Data Discovery Phase
  key criterion for the success of the data warehouse is the cleanliness and cohesiveness of the data within it
- Once you understand what the target needs to look like, you need to identify and examine the data sources

Data Discovery Phase

- It is up to the ETL team to drill down further into the data requirements to determine each and every source system, table, and attribute required to load the data warehouse
  - Collecting and Documenting Source Systems
  - Keeping track of source systems
  - Determining the System of Record - Point of originating of data
  - Definition of the system-of-record is important because in most enterprises data is stored redundantly across many different systems.
  - Enterprises do this to make nonintegrated systems share data. It is very common that the same piece of data is copied, moved, manipulated, transformed, altered, cleansed, or made corrupt throughout the enterprise, resulting in varying versions of the *same* data

Data Content Analysis - Extraction

- Understanding the content of the data is crucial for determining the best approach for retrieval

  - **NULL values.** An unhandled NULL value can destroy any ETL process. NULL values pose the biggest risk when they are in foreign key columns. Joining two or more tables based on a column that contains NULL values will cause data loss! Remember, in a relational database NULL is not equal to NULL. That is why those joins fail. Check for NULL values in every foreign key in the source database. When NULL values are present, you must *outer* join the tables

  - **Dates in nondate fields.** Dates are very peculiar elements because they are the only logical elements that can come in various formats, literally containing different values and having the exact same meaning. Fortunately, most database systems support most of the various formats for display purposes but store them in a single standard format

Data Content Analysis - Extraction

- 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.
- Later the ability to capture data changes in the source system instantly becomes priority
- The ETL team is responsible for capturing data-content changes during the incremental load.

Determining Changed Data

- Audit Columns - Used by DB and updated by triggers
- Audit columns are appended to the end of each table to store the date and time a record was added or modified
- 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 – example using outer joins

Determining Changed Data

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.
- Not the most efficient technique, but most reliable for capturing changed data

Determining Changed Data

Initial and Incremental Loads

- 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 drops the previous load table, 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.

Transformation

Part of ETL

Transformation

- Main step where the ETL adds value
- Actually changes data and provides guidance whether data can be used for its intended purposes
- Performed in staging area

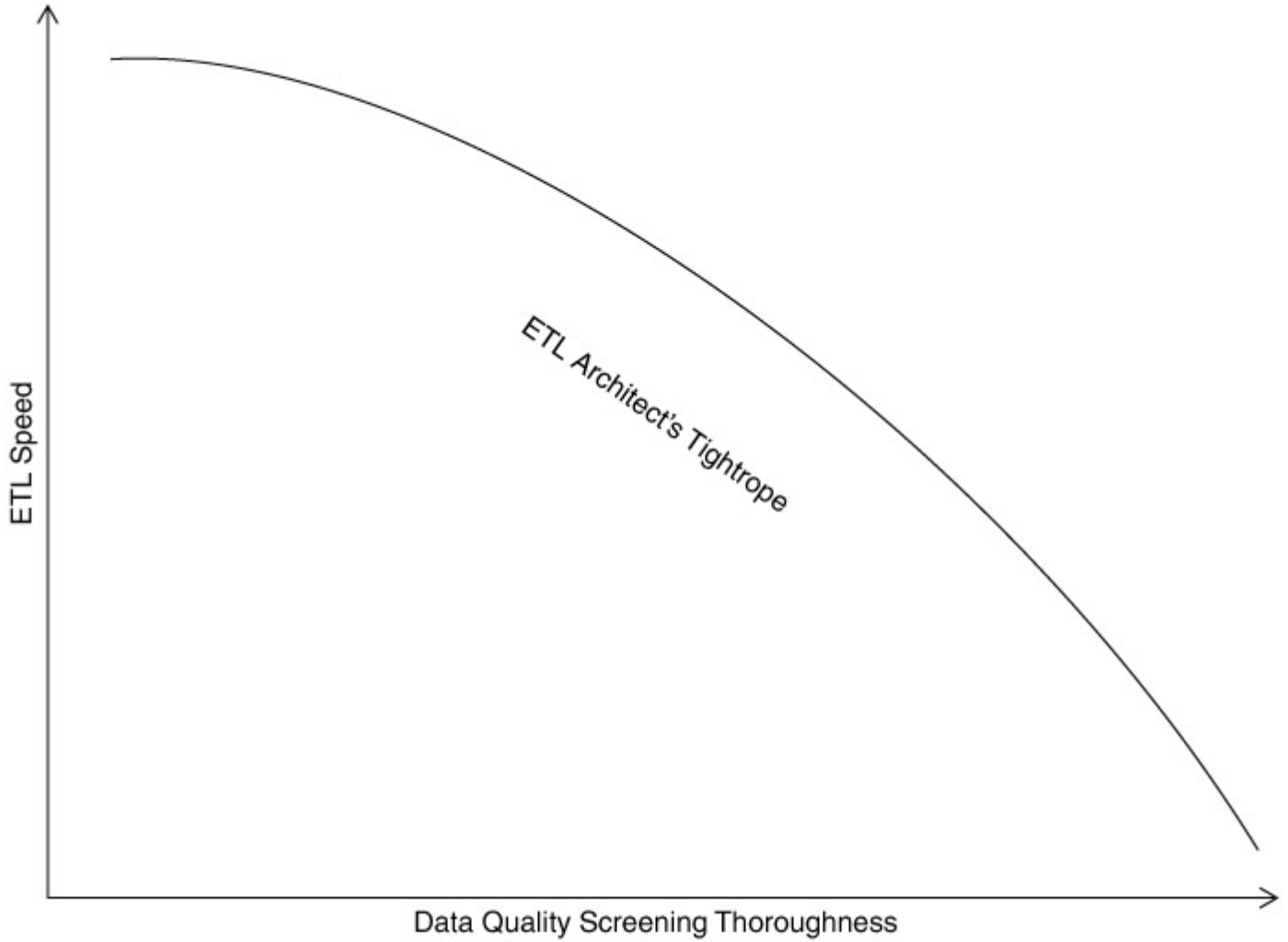
Transformation

Data Quality paradigm

- Correct
- Unambiguous
- Consistent
- Complete
- Data quality checks are run at 2 places - after extraction and after cleaning and confirming additional check are run at this point

Transformation - Cleaning Data

- Anomaly Detection
  - Data sampling – count(*) of the rows for a department column
- Column Property Enforcement
  - Null Values in reqd columns
  - Numeric values that fall outside of expected high and lows
  - Cols whose lengths are exceptionally short/long
  - Cols with certain values outside of discrete valid value sets
  - Adherence to a reqd pattern/ member of a set of pattern

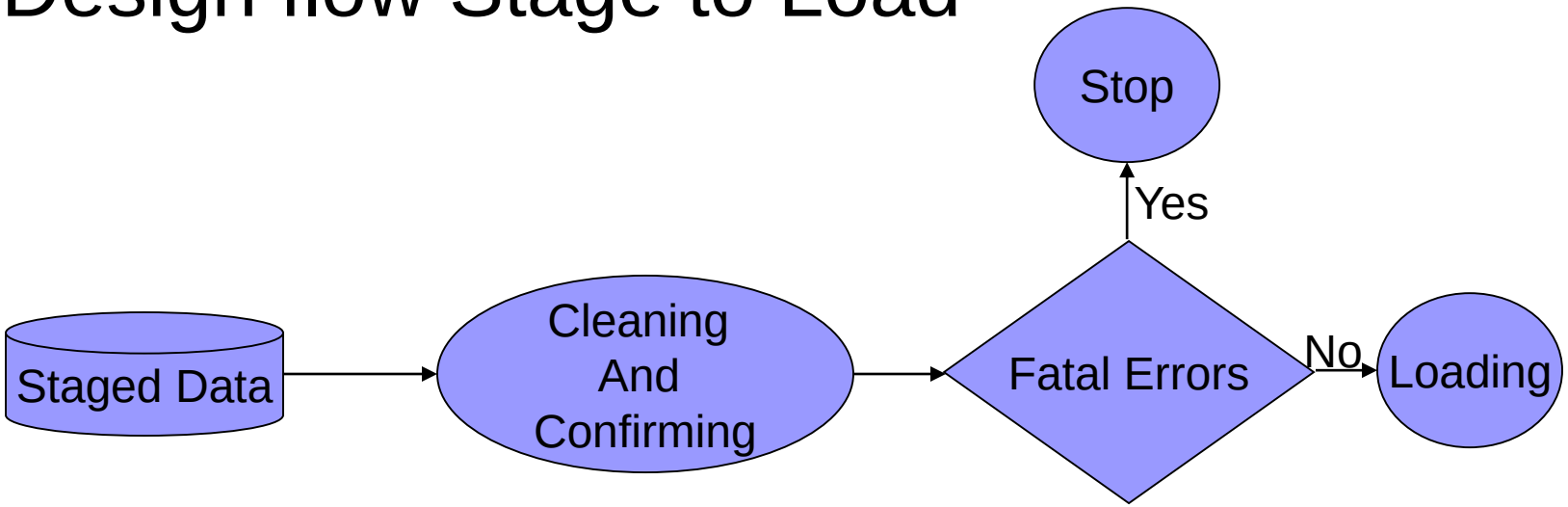

ETL Speed
ETL Architect's Tightrope
Data Quality Screening Thoroughness

Transformation - Confirming

- Structure Enforcement
  - Tables have proper primary and foreign keys
  - Obey referential integrity
- Data and Rule value enforcement
  - Simple business rules
  - Logical data checks

Map Staging to Loading

- Design flow Stage to Load



Loading

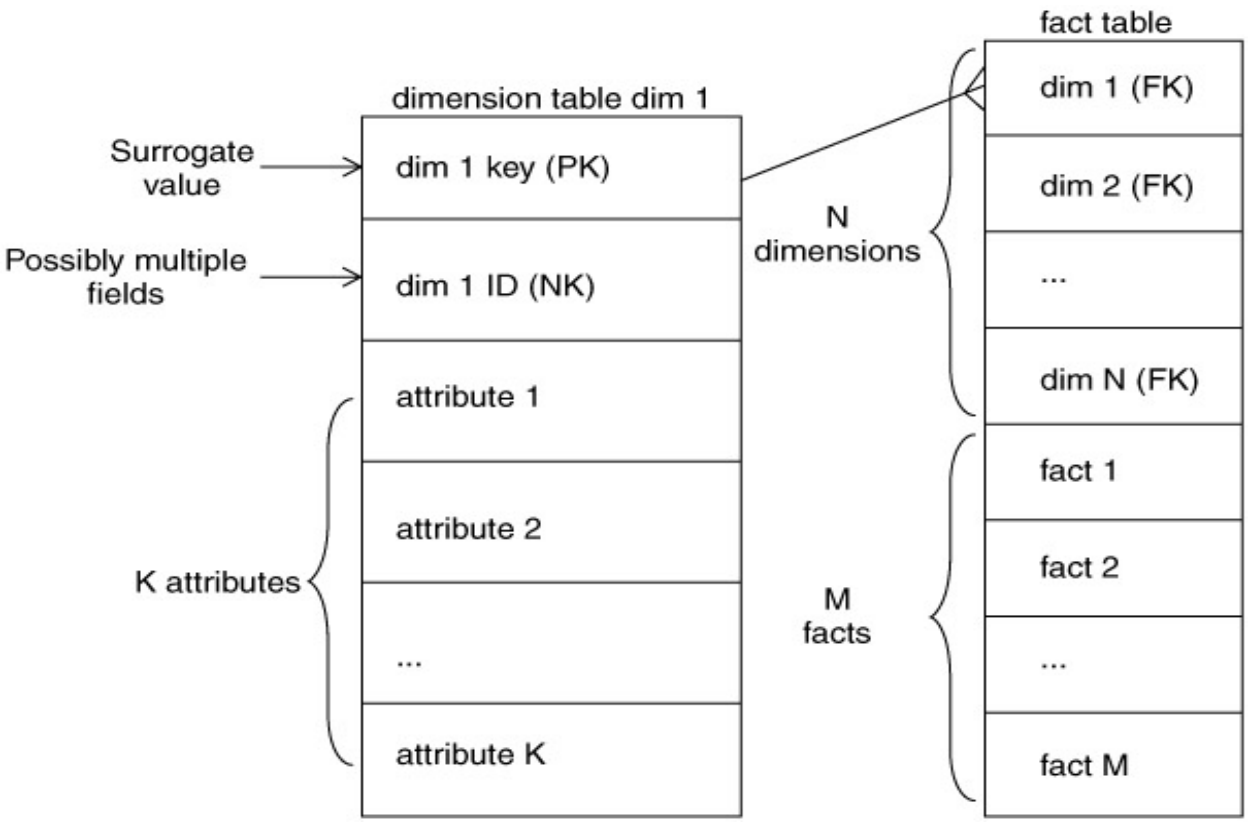
Part of ETL

Loading Dimensions &Facts

Loading Dimensions

- Physically built to have the minimal sets of components
- The primary key is a single field containing meaningless unique integer – Surrogate Keys
- The DW owns these keys and never allows any other entity to assign them
- De-normalized flat tables – all attributes in a dimension must take on a single value in the presence of a dimension primary key.
- Should possess one or more other fields that compose the natural key of the dimension

Loading Dimensions



Loading Dimensions

- The data loading module consists of all the steps required to administer slowly changing dimensions (SCD) and write the dimension to disk as a physical table in the proper dimensional format with correct primary keys, correct natural keys, and final descriptive attributes.
- Creating and assigning the surrogate keys occur in this module.
- The table is definitely staged, since it is the object to be loaded into the presentation system of the data warehouse.

Loading Dimensions

- When DW receives notification that an existing row in dimension has changed it gives out 3 types of responses
  - Type 1
  - Type 2
  - Type 3

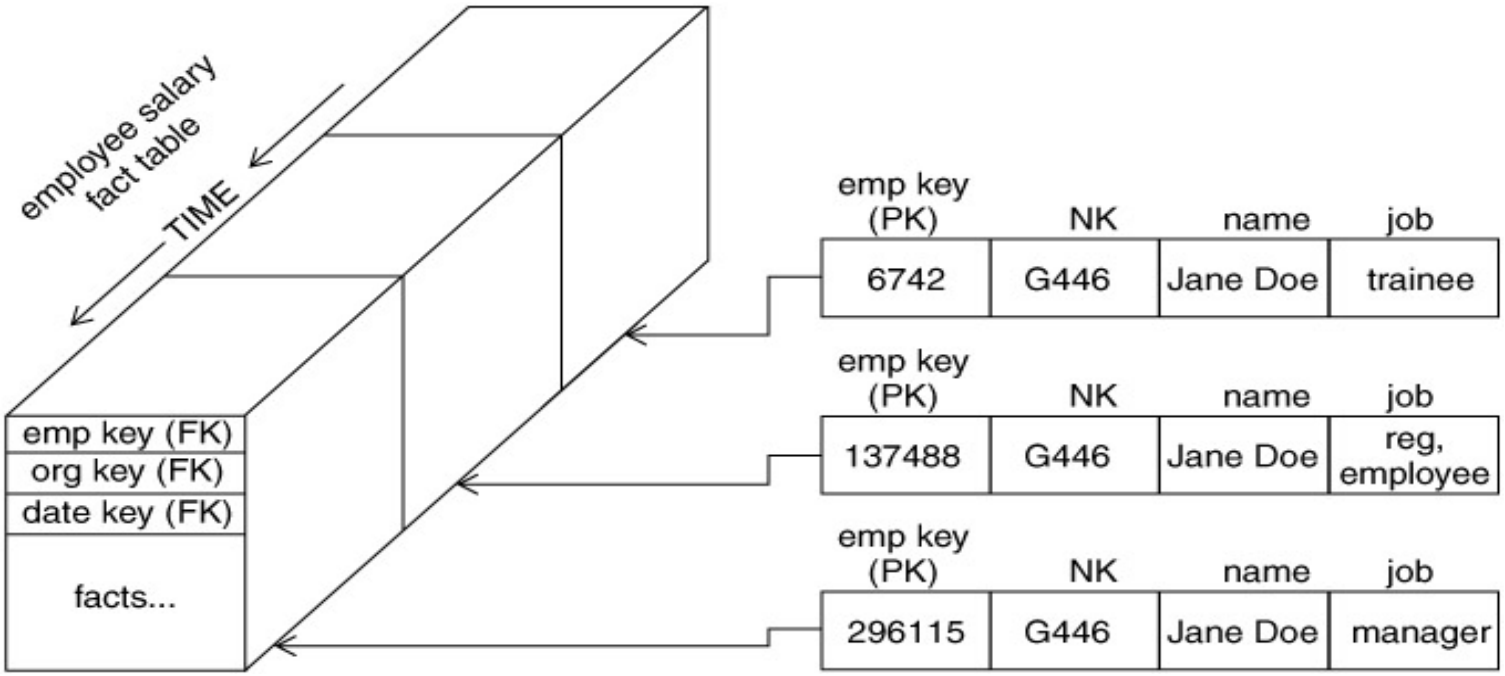
Type 1 Dimension

Primary Key	Natural Key	Prod Name	Category	Package Type
23708	AB29	120zCola	Soft Drinks	Glass

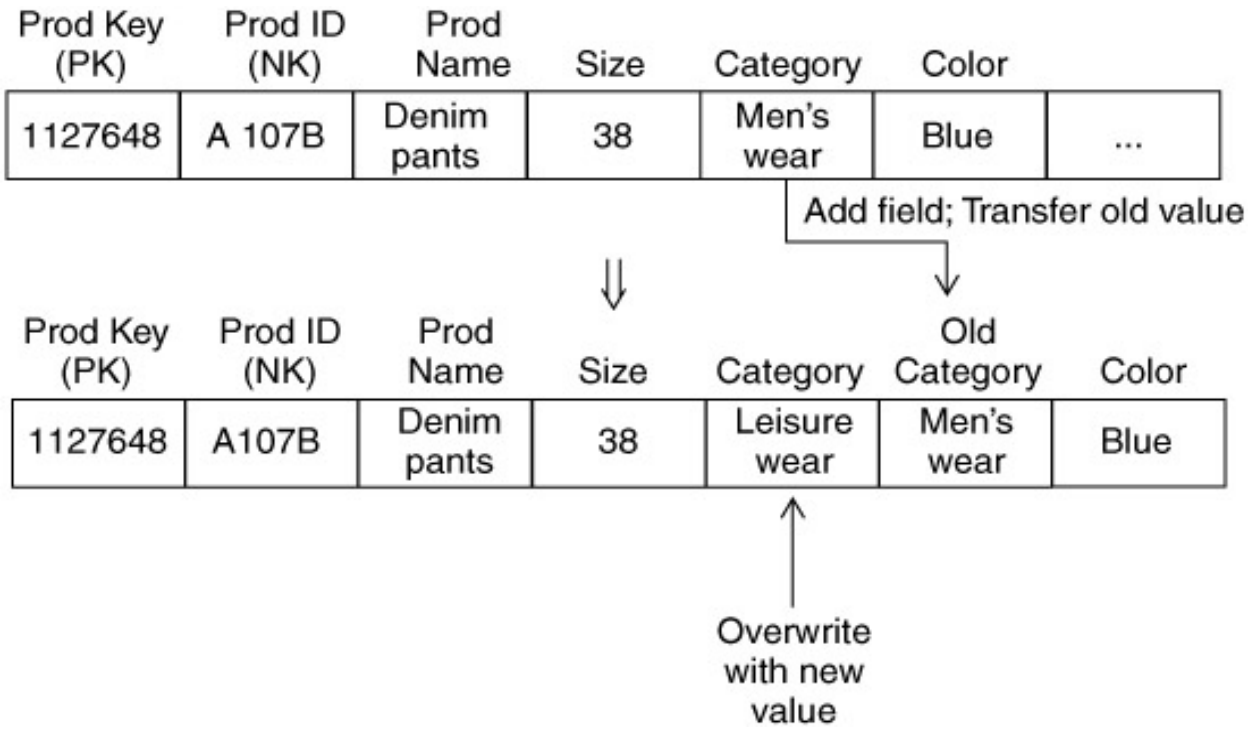
becomes

23708	AB29	120zCola	Soft Drinks	Plastic

Type 2 Dimension



Type 3 Dimensions



Loading facts

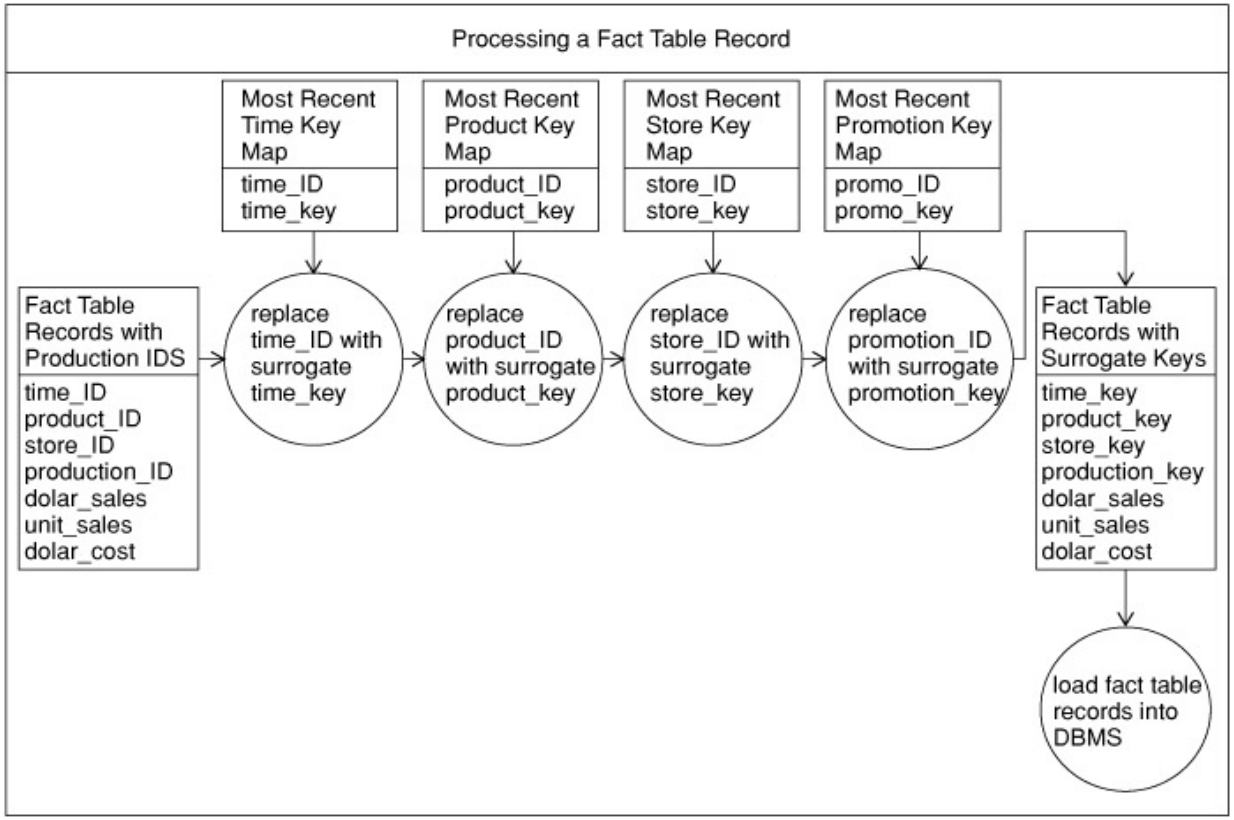
- Facts

  Fact tables hold the measurements of an enterprise. The relationship between fact tables and measurements is extremely simple. If a measurement exists, it can be modeled as a fact table row. If a fact table row exists, it is a measurement

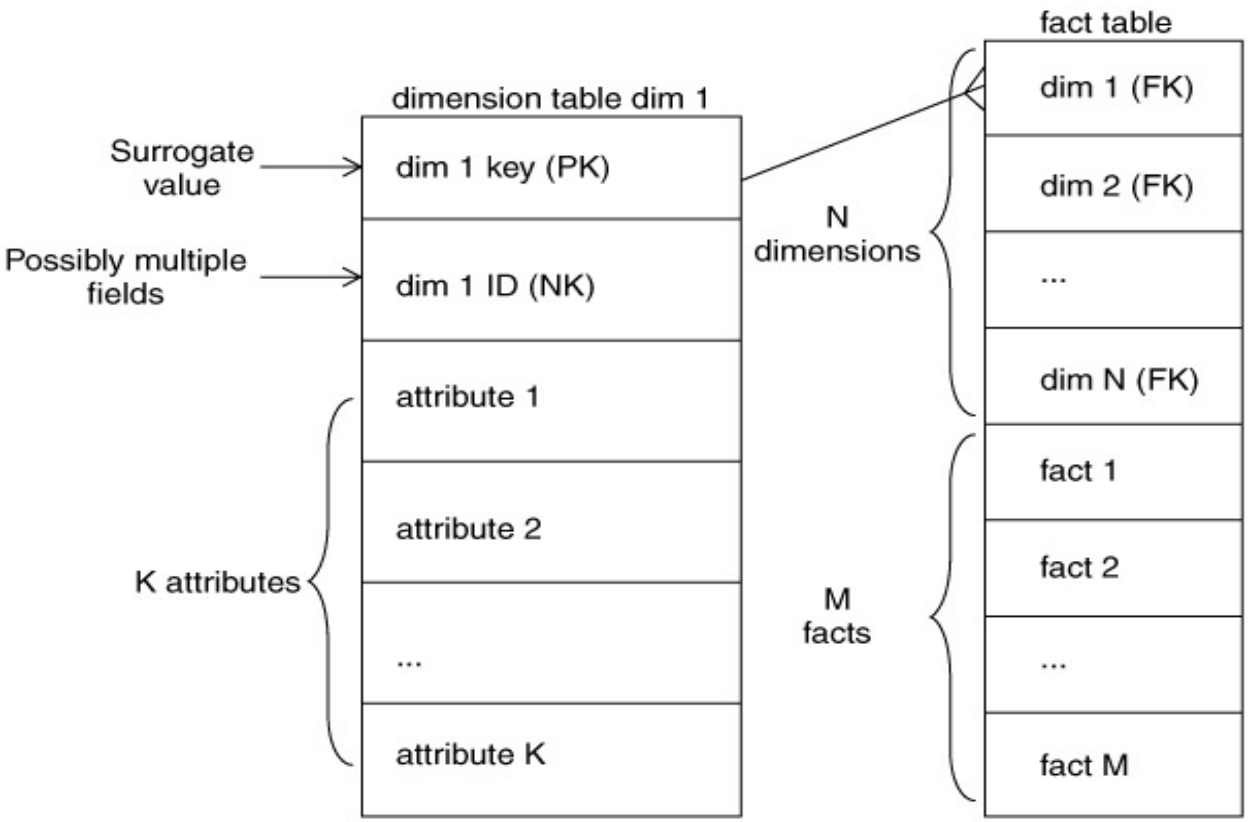
Key Building Process - Facts

- When building a fact table, the final ETL step is converting the natural keys in the new input records into the correct, contemporary surrogate keys
- ETL maintains a special surrogate key lookup table for each dimension. This table is updated whenever a new dimension entity is created and whenever a Type 2 change occurs on an existing dimension entity
- All of the required lookup tables should be pinned in memory so that they can be randomly accessed as each incoming fact record presents its natural keys. This is one of the reasons for making the lookup tables separate from the original data warehouse dimension tables.

Key Building Process



Key Building Process



Loading Fact Tables

- Managing Indexes
  - Performance Killers at load time
  - Drop all indexes in pre-load time
  - Segregate Updates from inserts
  - Load updates
  - Rebuild indexes

Managing Partitions on Facts

- Partitions allow a table (and its indexes) to be physically divided into *minitables* for administrative purposes and to improve query performance
- The most common partitioning strategy on fact tables is to partition the table by the date key. Because the date dimension is preloaded and static, you know exactly what the surrogate keys are
- Need to partition the fact table on the key that joins to the date dimension for the optimizer to recognize the constraint.
- The ETL team must be advised of any table partitions that need to be maintained.