The software processes that facilitate the original loading and the periodic refreshment of the data warehouse contents are commonly known as Extraction-Transformation-Loading (ETL) processes. The intention of this survey is to present the research work in the field of ETL technology in a structured way. To this end, we organize the coverage of the field as follows: (a) first, we cover the conceptual and logical modeling of ETL processes, along with some design methods, (b) we visit each stage of the E-T-L triplet, and examine problems that fall within each of these stages, (c) we discuss problems that pertain to the entirety of an ETL process, and, (d) we review some research prototypes of academic origin. [Article copies are available for purchase from InfoSci-on-Demand.com]

Keywords: Data Warehouse Refreshment; ETL

INTRODUCTION

A data warehouse typically collects data from several operational or external systems (also known as the sources of the data warehouse) in order to provide its end-users with access to integrated and manageable information. In practice, this task of data collection (also known as data warehouse population) has to overcome several inherent problems, which can be shortly summarized as follows. First, since the different sources structure information in completely different schemata the need to transform the incoming source data to a common, “global” data warehouse schema that will eventually be used by end user applications for querying is imperative. Second, the data coming from the operational sources suffer from quality problems, ranging from simple misspellings in textual attributes to value inconsistencies, database constraint violations and conflicting or missing information; consequently, this kind of “noise” from the data must be removed so that end-users are provided data that are as clean, complete and truthful as possible. Third, since the information is constantly updated in the production systems that populate the warehouse, it is necessary to refresh the data warehouse contents regularly, in order to provide the users with up-to-date information. All these problems require that the respective software processes are constructed by the data warehouse development team (either manually, or via specialized tools) and executed in appropriate time intervals.
for the correct and complete population of the data warehouse.

The software processes that facilitate the population of the data warehouse are commonly known as Extraction-Transformation-Loading (ETL) processes. ETL processes are responsible for (i) the extraction of the appropriate data from the sources, (ii) their transportation to a special-purpose area of the data warehouse where they will be processed, (iii) the transformation of the source data and the computation of new values (and, possibly records) in order to obey the structure of the data warehouse relation to which they are targeted, (iv) the isolation and cleansing of problematic tuples, in order to guarantee that business rules and database constraints are respected and (v) the loading of the cleansed, transformed data to the appropriate relation in the warehouse, along with the refreshing of its accompanying indexes and materialized views.

A naïve, exemplary ETL scenario implemented in MS SQL Server Integration Services is depicted in Figure 1. The scenario is organized in two parts. The first part, named Extraction task (Figure 1a), is responsible for the identification of the new and the updated rows in the source table LINEITEM. The idea is that we have an older snapshot for line items, stored in table LINEITEM which is compared to the new snapshot coming from the sources of the warehouse (depicted as NEW_LINEITEM) in Figure 1b. Depending on whether a row is (a) newly inserted, or, (b) an existing tuple that has been updated, it is routed to the table LINEITEM, via the appropriate transformation (remember that insertions and updates cannot be uniformly handled by the DBMS). Once the table LINEITEM is populated, the second part of the scenario, named Transform & Load task (Figure 1a) is executed. This task is depicted in Figure 1c and its purpose is to populate with the update information several tables in the warehouse that act as materialized views. The scenario first computes the value for the attribute Profit for each tuple and then sends the transformed rows towards four “materialized” views that compute the following aggregated measures (keep in mind that ExtendedPrice refers to the money clients pay per line item, PartKey is the primary key for items and SuppKey is the primary key for suppliers):

- **View A**: aggregate profit and average discount grouped by PartKey and SuppKey
- **View B**: average profit and extended price grouped by PartKey and LineStatus
- **View C**: aggregate profit and extended price grouped by LineStatus and PartKey
- **View D**: aggregate profit and extended price grouped by LineStatus

As one can observe, an ETL process is the synthesis of individual tasks that perform extraction, transformation, cleaning or loading of data in an execution graph – also referred to as a workflow. Also, due to the nature of the design artifact and the user interface of ETL tools, an ETL process is accompanied by a plan that is to be executed. For these reasons, in the rest of this survey we will use the terms ETL process, ETL workflow and ETL scenario interchangeably.

The historical background for ETL processes goes all the way back to the birth of information processing software. Software for transforming and filtering information from one (structured, semi-structured, or even unstructured) file to another has been constructed since the early years of data banks, where the relational model and declarative database querying were not invented. Data and software were considered an inseparable duo for data management by that time and thus, this software was not treated as a stand-alone, special purpose module of the information system’s architecture. As Vassiliadis and Simitsis (2009) mention “since then, any kind of data processing software that reshapes or filters records, calculates new values, and populates another data store than the original one is a form of an ETL program.”

After the relational model had been born and the declarative nature of relational database querying had started to gain ground, it was quite natural that the research community would try to apply the declarative paradigm to data
Related Content

Mining Topological Patterns in Spatio-Temporal Databases
Wynne Hsu, Mong Li Lee and Junmei Wang (2008). *Temporal and Spatio-Temporal Data Mining* (pp. 130-165).
[www.igi-global.com/chapter/mining-topological-patterns-spatio-temporal/30265?camid=4v1a](www.igi-global.com/chapter/mining-topological-patterns-spatio-temporal/30265?camid=4v1a)

A Framework for Integrating Ontologies and Pattern-Bases
[www.igi-global.com/chapter/framework-integrating-ontologies-pattern-bases/7580?camid=4v1a](www.igi-global.com/chapter/framework-integrating-ontologies-pattern-bases/7580?camid=4v1a)
Ethical and Legal Data Mining: Paradigms of Organizational Development Practitioners and Organizational Psychologists
Ben Tran (2013). *Ethical Data Mining Applications for Socio-Economic Development* (pp. 201-229).
www.igi-global.com/chapter/ethical-legal-data-mining/76263?camid=4v1a

A Parameterized Framework for Clustering Streams
www.igi-global.com/article/parameterized-framework-clustering-streams/1822?camid=4v1a