A case for ELT

Abhishek Tiwari

Citation: *A. Tiwari*, "A case for ELT", Abhishek Tiwari, 2017.
doi:10.59350/5m04s-dbr19

Published on: December 22, 2017

Cheap storage and on-demand compute in the cloud coupled with the emergence of new big data frameworks and tools are forcing us to rethink the whole ETL and data warehousing architecture. There is a strong argument for ELT i.e. extract, load, and transform model.

Classic ETL

In classic Extract, transform, and load (ETL) model, we store entities in their corresponding application databases i.e. as rows in the relational tables. Then we perform frequent batch ETL from application databases to a data warehouse. Often post-extraction data is staged in intermediate tables which is followed by transformation and load steps to migrate data into a target database or data warehouse.

As you can see data transformation before the load is an important and necessary step in this classic ETL model, and with ELT approach we are making data transformation more on-demand.

Late transformation

In ELT model, you can load your events and entities in raw format into a data lake backed by a cloud object storage service such as Amazon S3 or Google Cloud Storage. Big data frameworks such as Presto, Apache Spark, Apache Drill running on cloud virtual servers will enable you to perform the interactive or on-demand queries against your data lake. You can also use cloud and SaaS services such as Google BigQuery, Amazon Redshift Spectrum, Amazon Athena, Qubole to implement ELT approach.

When you execute a query against the data lake, data transformation are performed at run-time or in-memory. Transformation is happening at a very late stage. Transformed data or ad-hoc views can be transient or persistent.

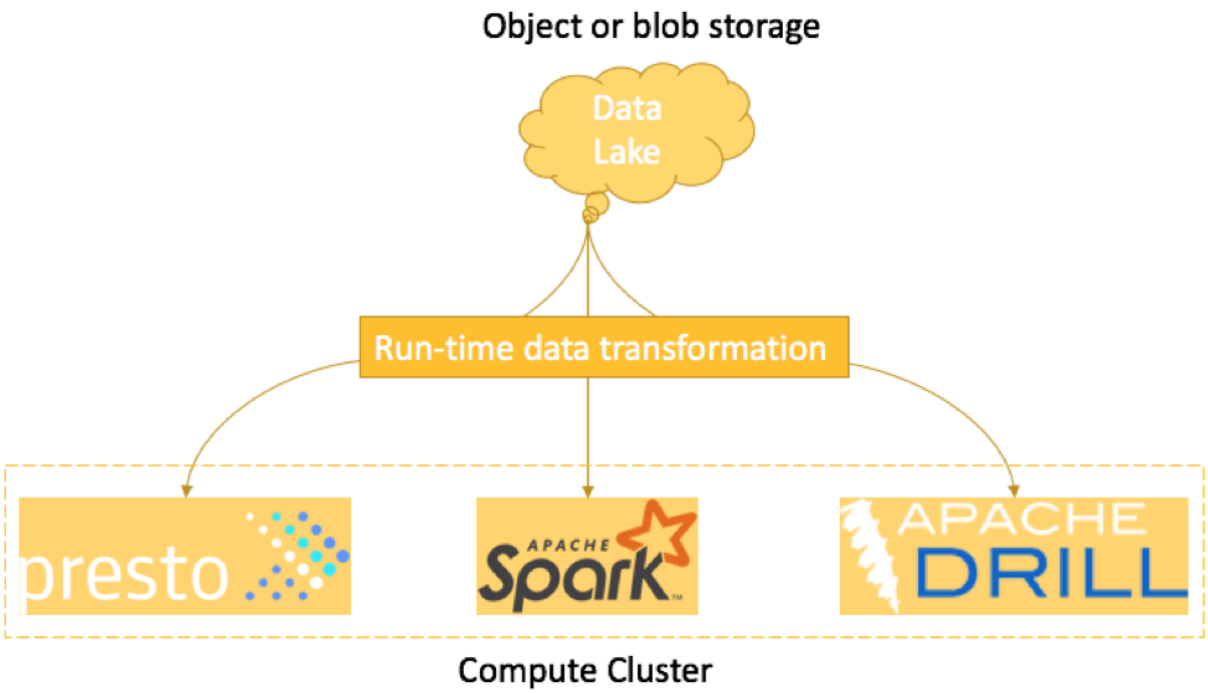

Figure 1: In ELT, data transformation is performed at run-time i.e. late data transformation.

ELT approach is more flexible, agile and requires low maintenance. It basically enables you to experiment with your data to extract more insights, especially when compared with classic ETL model where you get a fixed set of views. Using ELT, you can always create ad-hoc views by running the interactive queries and write results back to data lake. In addition, this approach is more tailored for both structured as well unstructured data sets. If the majority of your data is unstructured such as text, images, documents, etc. then ELT is a more preferred option.

Stateless and elastic

More importantly, ELT architecture is stateless and elastic because compute and storage layers are decoupled and they can scale independently. There is no capacity planning needed. By utilizing the elastic nature of the cloud, organizations can avoid under or over-provisioning of resources required for the data warehouse. There are SaaS platforms such as Qubole and Snowflake to make ELT approach more accessible.

Different audience

Last but not least, ELT attracts a very different type of audience - data scientists who want to perform more exploratory analysis and ad-hoc interactive queries. In our example retail organization, using data lake data scientists can perform analysis around product recommendations, customer propensity, pricing strategy, customer segmentation etc. This type of analysis is greatly eased by open source tools such RStudio, Jupyter, Zeppelin along with scripting languages R and Python.

Challenges

Obviously, ELT approach has its own challenges. For instance, in-depth knowledge of data schema and underlying relationships is required for anyone who is interested to perform ad-hoc queries against raw data. This can be time-consuming and bit challenging if you are dealing with hundreds of data sources and thousands of event types.