PROPOSAL FOR PROBLEM STATEMENT 3-
EXTRACT DATA, ALTER FOR ANALYTICS PURPOSES

By Team- The Exterminators
Janani K ; 3rd year ; CSE ; 190501045 ; 2019cs0464@svce.ac.in ; 9884432694
Allen Manoj ; 3rd year ; CSE ; 190501015 ; 2019cs0499@svce.ac.in ; 9962842783
Poonam B N ; 3rd year ; INT ; 190801053 ; 2019it0522@svce.ac.in ; 9566111917
Akaash V ; 3rd year ; CSE ; 190501010; 2019cs0465@svce.ac.in ; 9344299809

Sri Venkateswara College of Engineering

PROBLEM STATEMENT (3) - BUILDING AN EFFICIENT ETL PIPELINE AND ALTERING FOR ANALYTICAL PURPOSES

Analytical data are rich sources of information obtained from a variety of sources. Accuracy, completeness, and conformity with standards cannot be guaranteed. To meet the requirements for design, data quality and validation for large volumes of data are crucial. By designing an efficient ETL pipeline, the above-mentioned problem will be optimized by increasing quality, reducing costs, generating better performance, and handling faults. Further, analytics on data must be performed to make better decisions and effectively present results.

SUMMARY OF SOLUTION

In this proposal, we have summarized several techniques we have planned to optimize the ETL pipeline and workflow: (1) Basic approach using parallelization and partitioning (2) Optimizing session performance by tuning methods to remove bottlenecks (3) Fault tolerance and using Cost based optimizers (4) Additional methods for optimization (5) Building framework (6) Analytics

APPROACH/ DESIGN PRINCIPLES

- **Basic Approach**:
  - Extract data in parallel using Sequence containers in control flow. A package can be designed in such a way that it can pull data from non-dependant tables or files in parallel to reduce execution time. Implementing partitioning at the source and target side to perform operations.
  - Since the ETL system that we aim to design is dynamic, design approaches such as Meta Driven ETL can be used.
  - SSIS can be used to achieve complex tasks for execution. For Synchronous transformation, components like Lookup, and Derived can be used. Asynchronous transformations for Sort, Aggregate, Merge, and Join components can be used.
  - Dropping existing clustered indexes in the pre-execution phase and re-create them in the post-phase to reduce performance-related issues due to huge DML operations
  - Avoiding implicit conversions to accommodate more rows in a single buffer.
- **Optimizing session performance**:
- Database level tuning was performed to remove bottlenecks and handle duplication and missing records on the DB side to ensure sources are synchronized and full utilization of the source system is achieved. Optimizer Statistics and an Explain Plan can be used.
- Informatica level tuning is used to remove unwanted fields, to avoid constraints in WHERE clauses. Further, LOOKUP and FILTER transformation can be used to increase performance.
- **Fault tolerance**: Implemented by using recovery points at various places to roll back in case of fault occurrence, sending ACKs, using replication, or using Resilient Distributed Datasets.
- **Reducing execution time**: Implementation of Push Down optimizer logic. The transformation logic is pushed to the DB side and the optimizer of DB prepares the optimal plan.
- **Cost-based optimizer**: Identifying optimizable blocks and execution trees (Only realistic ones are identified). For each tree, a statistics set is generated by an optimizer and minimal costs are chosen.
- **Building framework**: The framework ingests data frames and destination lists from JSON configuration files. Each item contains a command that generates a data frame and a list of destinations to write the data frame to. Commands and destinations are specified by type in configuration, along with attributes and another file like a SQL file. By using configurable commands and destinations, best practices such as securing information in our destinations, ensuring data is saved in the correct locations, and connecting to valid sources when we retrieve data for the environment in which the job is run can be achieved.

- **Dynamic transformation**: To make the entire process configurable, use SSIS for extracting, unzipping, and importing data.
- **Additional**: Using shared caches for transferring data to reduce memory and IO usage, and sorters to rearrange the order of record sets. Duplications can be avoided using SSIS. The entire system can be made in real time using the SSIS ADO.NET Destination tool.
- **Analytics**:
  - By using PySpark we can achieve fast processing of huge amounts of data.
  - Spark core not only provides many robust features apart from creating ETL pipelines, but also provides support for machine learning (MLib), data streaming (Spark Streaming), SQL (Spark Sql), and graph processing (GraphX).

All these methods are performed hand in hand to increase the overall efficiency of the system.

RECOMMENDED TECHNOLOGY ELEMENTS

1. Informatica PowerCenter will be used for establishing and maintaining enterprise-wide data warehouses.
2. To extract data from sources like ERP or CRM and pull data into Hadoop.
3. Transform and load data using a batch process in real-time.
4. Apache Pig and Spark in MapReduce will be used.
5. Web Application using React to develop a dashboard consisting of the data of the DB and a button to troubleshoot synchronization faults.
6. We can integrate with other systems such as a Redshift data warehouse using Amazon Kinesis.
7. Amazon DynamoDB is used for database storage.
8. SQL, GraphX, MLib, Spark Streaming for data analytics.

RISKS AND HOW TO MITIGATE THEM

1. If the quality of the data is unknown, this can lead to missing a deliverable deadline and can cause the project to go over budget and may not meet the agreed schedule. To mitigate this risk, we perform formal data profiling of source data before starting the work. All inherent errors like inaccuracy, duplication, etc. can be identified and resolved before or during an ETL process.
2. We tend to find defects in data at a later stage. To avoid this, profiling of all data sources and target data after each ETL, cleansing, and fixing of corrupt data all are significant to ensure a smooth ETL process.
3. With any change in datasets, the data engineers need to update their approach to accommodate them, which results in downtime and high operational overheads.
4. With large quantities of data being ingested from many disparate (often fast-moving) data sources, teams find it tough to maintain and refurbish critical ETL flows.
5. By reimagining the data integration solutions through self-service, we can empower the users to create new customer data connections in minutes, securely, and easily.

PHASE 2 PROCESS FOR FUNCTIONALITY

Our first step is to build a system that is configured to perform Extract, Load, Transform, and Integrity. Next, we will start optimizing our system using the various approaches we have planned. A web dashboard will be built to view the analytics of the data. We would then again work on the optimization of our entire system. A button in the dashboard will be integrated to fix the synchronization errors. Further, we plan to execute all our bonus points over the last few weeks: Building a neat framework, configuring transform steps to be more dynamic, fixing errors, and making the entire system real-time.