Thuan Nguyen

Improving the ETL process for a case company

Metropolia University of Applied Sciences

Master of Engineering

Information Technology

Master's Thesis

01 June 2023

PREFACE

After all, I did it, and this chapter ends. I would like to take this opportunity to thanks all the people who supported me throughout my thesis journey.

First of all, I want to express my sincere gratitude to my thesis supervisor, Senior Lecturer Sami Sainio for his support and insightful guidance throughout this thesis. I would like to thank Ville Jääskeläinen for providing interesting topics to the students and pushing me to complete it, as well as my colleagues for their help and patience with my questions.

I could not have done this journey without the unconditional support from my wife Phuong. Finally, I would like to thank my little girl, Emily, who was with me all the time. Her presence gave me the energy I needed to finish this thesis.

Finland, 01 June 2023

Thuan Nguyen

Abstract

Author:	Thuan Nguyen
Title:	Improving the ETL process for a case company
Number of Pages:	51 pages
Date:	01 June 2023
Degree:	Master of Engineering
Degree Programme:	Information Technology
Supervisors:	Sami Sainio, Senior Lecturer

This thesis aimed to propose new extract, transform, and load (ETL) processes for Oracle Hyperion Financial Management (HFM) using Python language, and to improve the existing ETLs for Oracle Enterprise Resource Planning (ERP) cloud by adjusting Oracle procedures PL/SQL and parameters on Oracle ERP Web Services. The scope of this thesis was focused on two major sources of data which are HFM and ERP.

There has been a steady shift of the cloud migration, including database, data, applications, and information technology (IT) processes from local to cloud. Thus, the current ETLs designed to perform in the local environment (built by i.e. batch scripts) have shown limitations in communicating with the cloud applications. For example, they may run out of memory when uploading very large file to cloud environment S3. Additionally, the current ETLs do not transfer the complex metadata extracted from HFM to the data warehouse. This metadata often requires more cleansing, formatting, and flattening into a specific hierarchy before it can be transferred to data warehouses. Currently, in the case company, this metadata is managed using Microsoft Excel and manually inserted into the database.

The currently existing ETL processes for Oracle ERP cloud were developed using Oracle Procedures PL/SQL. However, these processes faced challenges in extracting a large amount of data from Oracle ERP cloud via web services. While they functioned properly with small dataset, they would produce errors with the larger ones.

The study was based on action research that was carried out with qualitative and quantitative methodologies. This involved reviewing system documentation, consulting with IT experts, analysing the performance of existing ETL processes, and gathering user feedback.

This work produced improved Python-based ETLs that were implemented into the production environment. They are compatible with cloud environments enhancing daily processes.

Keywords: ETL, Data Warehouse, Oracle HFM, Oracle EPM, Oracle ERP, EPM Maestro, EPM Automate, Data Modelling

List of Figures

Contents

List of Abbreviations

ETL	Extract, Transform and Load
ERP	Enterprise Resource Planning
BI	Business Intelligence
HFM	Hyperion Financial Management
ODI	Oracle Data Integrator
DW	Data Warehouse
EPM	Enterprise Performance Management
OLAP	Online Analytical Processing
AWS	Amazon Web Service
S3	Simple Storage Services
CRM	Customer Relationship Managements
DBMS	Database Management System
CMD	Command Prompts
IT	Information Technology
CSA	Current State Analysis

List of Tables

1 Introduction

Nowadays, ETL processes play a critical role in data warehousing by extracting data from multiple sources, transforming it into a consistent format, and loading it into a centralized repository (Codeless, 2023). The ETL aims to consolidate data from different sources, such as transactional databases and legacy systems, and provide a consistent view of the data for analysis purposes. If an ETL is not built properly, data analysts would face challenges in dealing with data stored in different locations with varying formats and structures, leading to poor data quality and inaccurate insights. A well-designed ETL process is essential for maintaining the quality of data. It can break down data silos, eliminate unnecessary data, and optimize the data processes for reporting and analysis (Astera, 2020). Currently, there are many ETL tools available in the market, such as ODI and Informatica, and they can be developed using programming languages.

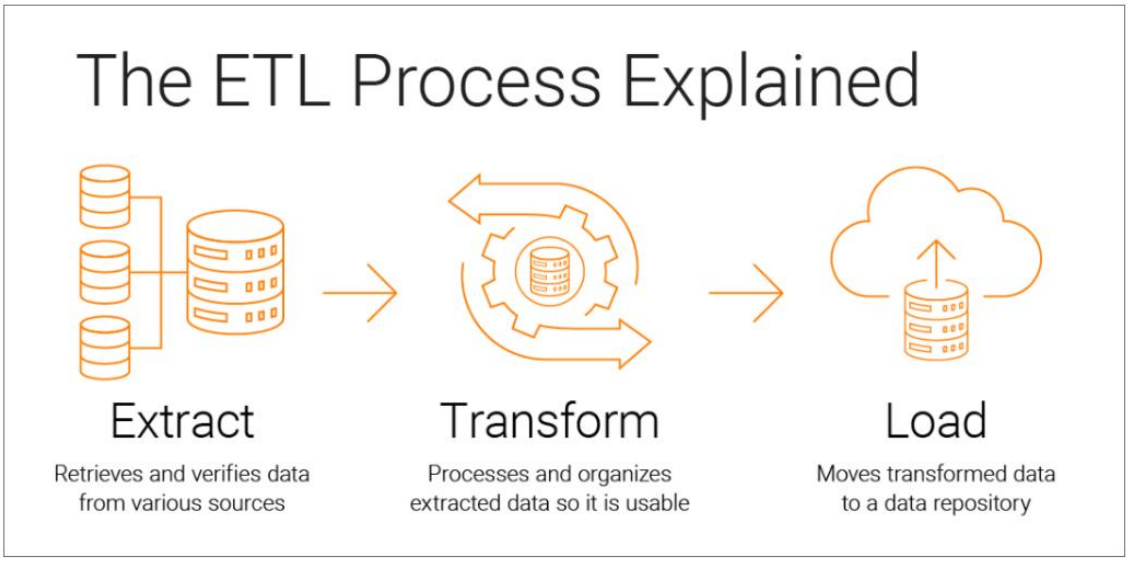

Figure 1. The ETL Process Explained (Informatica, 2022)

As seen from Figure 1 is an example of an ETL process that typically includes three main steps, which are extract, transform, and load. These steps form a complete process to integrate data from disparate sources, reduce data redundancy, and load it into a data repository.

This thesis focused on improving the existing ETL processes for Oracle HFM and Oracle ERP cloud in the case company. In addition, the thesis identified the challenges of existing ETLs, including data complexities and data quality issues, and proposed potential solutions for addressing them. By enhancing the current ETL processes as recommended in this document, the case company can improve its data processes and quality, which could ultimately lead to better decision for the business.

1.1 Business Context

During the pandemic, the case company has faced challenges in ensuring that their systems and data are available to employees anywhere. To address this issue, the case company has moved many applications to cloud environments. This has helped in cutting costs and has improved work efficiency.

The case company has encountered an increased need to perform detailed financial reporting, requiring complex data. To gain a competitive advantage, the case company needs the financial reporting to be more accurate and it needs to be delivered faster, with processes that require as little human intervention as possible. Automating the data processes for financial reporting process can help reduce errors and provide more reliable data for decision making.

1.2 Business Challenge, Objective and Outcome

The challenge for the case company was related to their existing ETL processes which are limited in their ability to communicate with cloud environments. Additionally, the current ETL processes encounter difficulties in extracting, manipulating big datasets, and transforming the complex data structures. Moreover, some of ETLs were built more than seven years ago and need to be updated to meet the current requirements of the business.

The objective of this thesis was to propose new ETL solutions for Oracle HFM using Python language and improve the existing ETLs for Oracle ERP cloud by

adjusting Oracle procedures PL/SQL and the parameters on Oracle ERP Web Services. The scope of this thesis is limited to study of the problems related to the ETL processes in two major sources of data, which are Oracle HFM and ERP.

The outcome of this thesis is a new ETL solution that was proposed and implemented to improve data quality, data manipulation, and cloud compatibility for Oracle HFM and ERP.

1.3 Thesis Outline

This thesis aimed to identify the needs of stakeholders, the weaknesses of the current ETL processes, and propose ways to improve the processes. The scope of this study is limited to two main data sources: Oracle HFM and Oracle ERP cloud.

This thesis is categorized into seven sections. Section 1 is the thesis introduction. Section 2 describes the study methods and materials used for analysis of the business challenges and the research approach to solve the problem. Section 3 explains terminology and explore the relevant literature to provide a conceptual framework as a guideline to improve the current ETL processes. Section 4 discusses the current stage analysis, describes the current ETL processes for Oracle HFM and Oracle ERP cloud and identifies the needs of stakeholders. This section summarizes the strengths and the weaknesses of the current ETL processes. Section 5 proposes solution to address the challenges identified in the current stage analysis. Section 6 discusses on the validation of the proposal and improvement made to initial proposal. This section also includes the implementation of the proposals. Section 7 discusses the next steps and the conclusion of this study.

2 Method and Material

This section describes the research approach and research design applied for exploring, investigating, and addressing business challenges.

2.1 Research Approach

From technical perspective, the objective of this thesis is to develop and improve the existing ETL, which is a technical problem and a practical issue that directly affects the organization. Therefore, an applied research approach was adopted to identify and solve immediate problems or provide innovate solutions to the issues. This is a typically case study that utilizes many data sources, such as documents, observations, and interviews to gain an in-depth understanding of the phenomenon (Kananen, 2017). Therefore, action research was considered as appropriate methodology that consist of a cycle of steps, including data gathering, data feedback, data analysis, action planning, implementation, and evaluation (Coughlan & Coghlan, 2002) (see Figure 2).

The thesis employs two main research methodologies: qualitative and quantitative. Qualitative research methods are commonly used to collect the documents, training videos from the previous implementation phase of the project, and the interview data from stakeholders, in order to gain better understand of the current system (Bhandari, 2020). This approach is considered flexible when interpreting data. On the other hand, quantitative research methods are used to collect numerical data, such as the number of rows of data loaded into the data warehouse and the loading time. Statistical tools are then used to provide an overall summary of the study objects, and investigate the relationship between study variables (Bhandari, 2020). In addition, a combination of these two research methods can be used to compensate for their weaknesses (Cooper & Schidler, 2012).

For implementation phase, agile methodologies in Jira tool were used for project management. This method involves a cycle of the process of planning, executing,

and evaluating tasks, which allows for continuous improvement and delivery of increments.

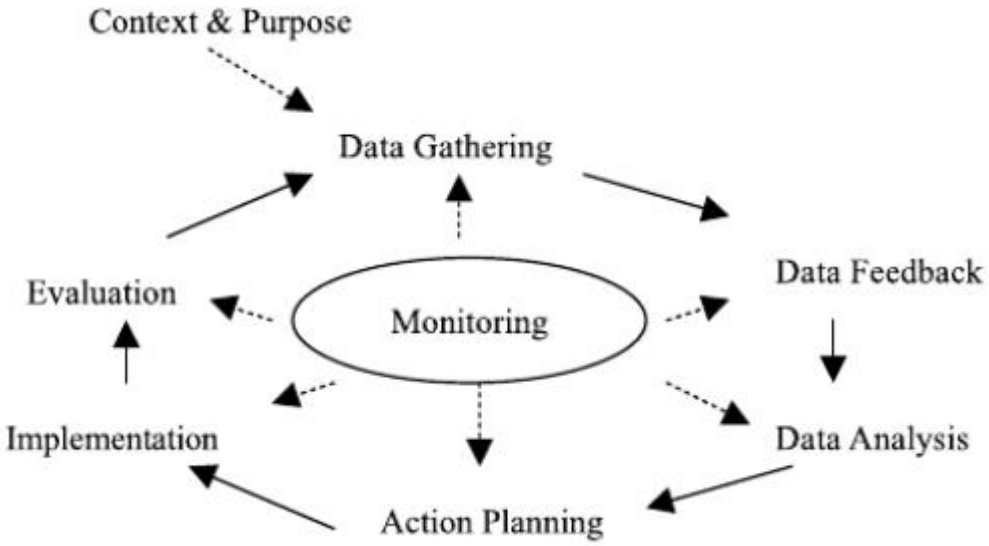

Figure 2. Action research cycle (Coughlan & Coghlan, 2002).

2.2 Research Design

As shown in Figure 3 below, the first step of the study clearly identified the object of the thesis. The case company needed a new ETL solution for HFM as well as improvements to existing ETLs for ERP.

The second step was to find the best practice from the relevant literature in the industry, in order to create a framework that could define an overall roadmap for developing the thesis. The next step involved collecting input and output data on the processes from company documents, interviews, and observations of the production environment (Data 1 from Figure 3). The data was then analysed to identify the weaknesses and the strengths of the current solution.

After creating the framework and analysing current solution, regular meetings and review sessions were organized to find the possible solutions (Data 2 from Figure

3). Several proposals were considered in practice, and while some of the proposals were promising, they could not be applied due to high resource cost. In the end, one solution was identified and planned for implementation by stakeholders.

As the final part, the feedback from stakeholders was analysed to improve the proposal (Data 3 from Figure 3), and the outcome of this step was a final proposal that would improve the current solution and to meet the business needs.

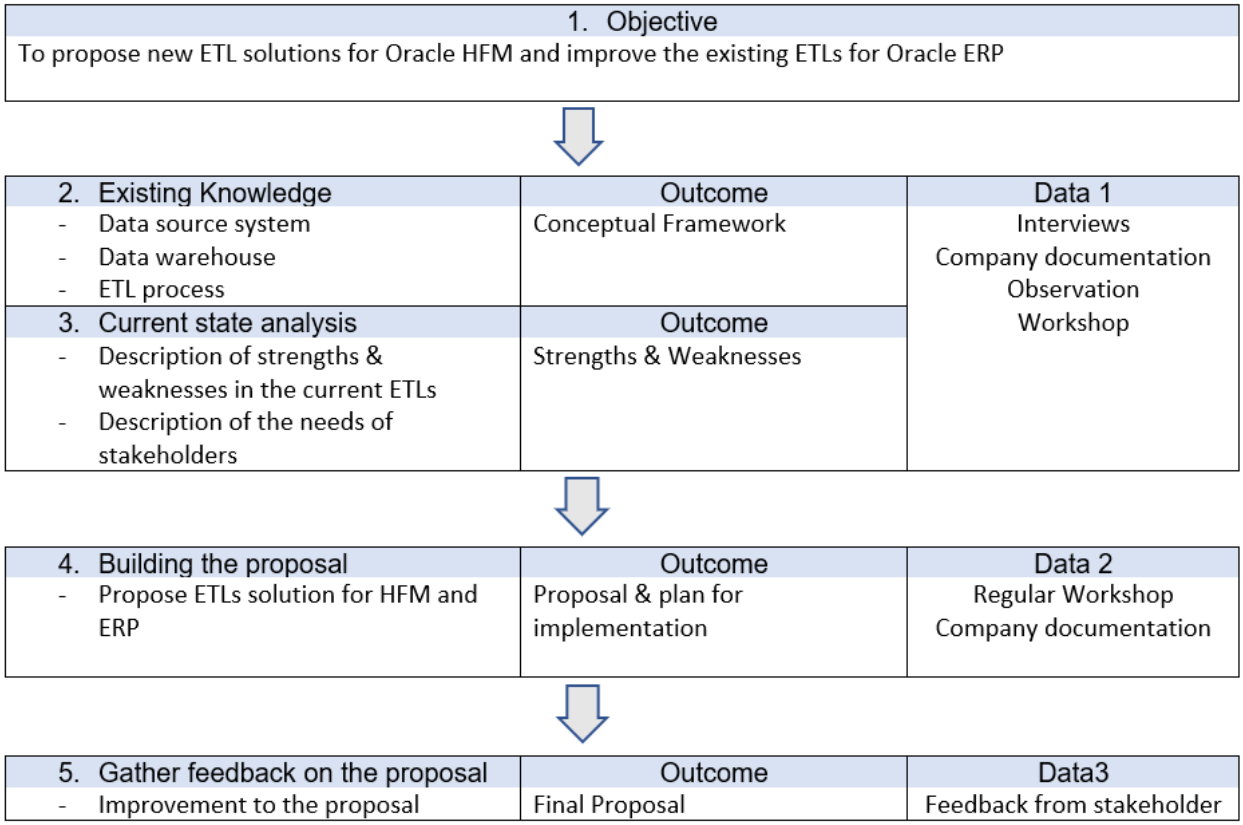

Figure 3. Research design of this thesis

Figure 3. Research design of this thesis

2.3 Data Collection and Analysis

The study was performed during the pandemic time, so the interview, workshops, and meetings with stakeholders were conducted via Microsoft Teams. Additionally, company documents were used to gain knowledge about the system components and functionalities and to use to propose a new solution.

Table 1. Details of interviews, workshops, and data collection in Data 1-3 in the thesis.

Data 1				
Participants	Topic	Date	Method	Outcome
Service manager, Developers, Financial controller	Discussion on the need to improve the current ETLs or develop new ones, and the challenges in the current system	05-2021	Meeting and interview via Microsoft Teams	Obtain access rights to the servers and the internal documentations to identify the weaknesses in current processes
Service manager, Product Owner, Developers, Financial controller	Discussion on the details of data models, metadata, and stakeholder's responsibilities, and plan for building the demo processes	05-2021	Workshop via Microsoft Teams	List of stakeholder's responsibilities and current state analysis
Service manager, Product Owner, Developers, Financial controller	Open discussion on creating a roadmap for the system over the next 3 years	06-2021	Workshop via Microsoft Teams	Collect new requirements and re-evaluate needs and findings
Data 2				
Participants	Topic	Date	Method	Outcome

Service manager, Product Owner, Developers, Financial controller	Status check and feedback	07.2021	Workshop via Microsoft Teams	Propose new solution to improve the processes and collect the feedback on the new solution
Data 3				
Participants	Topic	Date	Method	Outcome
Service manager, Product Owner, Developers, Financial controller	Validation, evaluation of the proposal	08.2021	Workshop via Microsoft Teams	Final proposal and implementation

As seen from Table 1, the data from the thesis can be divided into three rounds. The first round (Data 1) was conducted for listing known issues of data pipeline and used for current state analysis of ETL processes. During this round, the stakeholder's responsibilities were also identified, and technical developers were granted the access rights to the system to observe weaknesses and strengths of the system. In the next round of data collection Data 2, an initial proposal was released. The feedback from stakeholders in different departments was collected in regular workshops to analyse and find the best practical solution. In the final round (Data 3) was conducted for validating, evaluating of final proposal. Additionally, during this round, the potential for the future studies and improvements were discussed.

The current state analysis was the biggest part of analysis as there were a lot of existing knowledge, best practices, and internal documents needed to be reviewed to structure the topics for meetings and workshops. The terminology used in this thesis are explained in Section 3 below.

3 Literature Review

This section explains terminologies and discusses the best practices and the relevant literature to establish a conceptual framework for the study as a research direction to improve the current ETL processes.

3.1 Data Warehouse

A data warehouse is a central repository for information, where data is aggregated, optimized for the purpose of data analysis, and for querying large volumes of data (Altexsoft, 2021). According to (David, 2023), a data warehouse is defined as a core of BI systems and a process for collecting and transforming data from sources to target destinations in a consistent format to provide valuable business insights.

Today, the most well-known data warehouse architecture is organized into three tiers (see Figure 4) (Javatpoint, 2022). The top tier is the front-end layer and is also an important layer for presenting data to end-users through analysis and reporting tools. Data in top-tier is stored in a format that is optimized, well-structured, and validated for easier data analytics. The middle tier consists of an OLAP server that is used for fast querying. OLAP is defined as a software system designed for multidimensional data analysis that allows users to analyse data at high speeds from a data warehouse, data mart, or centralized data store (IBM). The bottom tier consists of a database server, which is typically a relational database system, where data is loaded and stored.

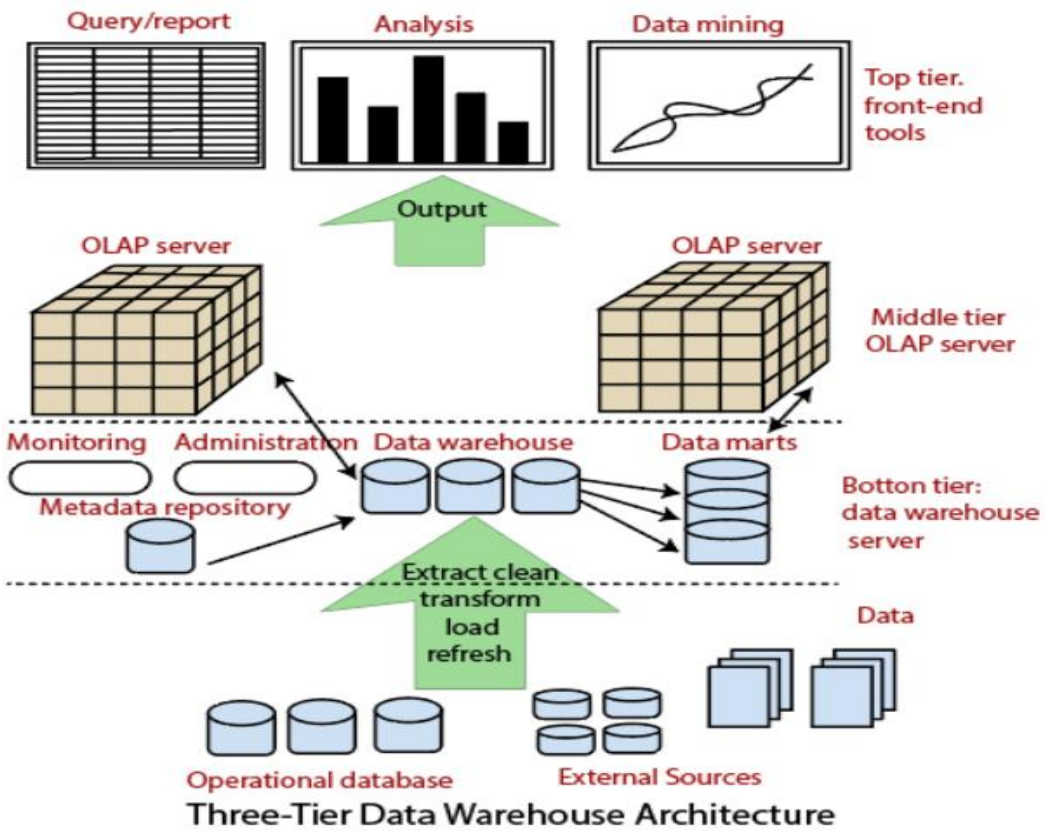

Figure 4. Three-Tier Data Warehouse Architecture (Javatpoint, 2022).

3.1.1 Oracle Data Warehouse

The Oracle Data Warehouse is an effective means of storing and processing business data and is a part of the Oracle Database system. The architecture of Oracle data warehouse includes single-tier, two-tier, and three-tier architectures. In addition to being a very popular database provider, the Oracle domain offers industry-standard database services for implementing data warehousing. It has great business features, such as being subject-oriented and integrating well with other systems, and providing a non-volatile platform. In terms of Data warehousing solutions, it provides data quality, metadata management features, and efficient data processing. This system is highly efficient for data-driven decision-making processes and is compatible with many ETL tools for data warehousing. (Priya, 2022)

3.1.2 Amazon S3 & Redshift

According to Amazon (2023), Amazon S3 is a web-based cloud storage service offered by AWS (Amazon Web Service). It is designed to store, protect, and retrieve any amount of data at any time, from anywhere and has great features like extremely high availability, high-speed, data available, security, and simple connection to other services. Amazon S3 consists of two key components which are buckets and objects. The data is stored as objects within buckets. Each object is identified by unique key that distinguishes it from other objects. The objects could be any type of files such as data files, photos, and videos. The buckets are containers for objects, and each bucket name must be unique. Each AWS account has a limit of 100 buckets, with no limit on the number of objects that can be stored in a bucket and has its own policies and configuration that support end-users to manage their data flexibly. The number of S3 buckets can be increased by more than 1000 buckets if a request is made to AWS support.

Amazon Redshift is defined as a powerful, petabyte-scale data warehouse cloud service from AWS. It is designed to analyse and store large amounts of data and handle database migrations, making it a popular platform for BI tools. Redshift is developed on top of parallel-processing technology that automatically splits workloads evenly and distributes it across multiple nodes in each cluster to reduce execution time. (Amazon, 2023). Therefore, it can perform queries on billions of rows at once or handle large datasets up to a petabyte or more, as noted by Kevin (2019).

3.2 ETL Process

According to IBM (2023), an ETL pipeline is seen as the set of processes used to get data from the sources and deliver it to a database called the data warehouse. The concepts of the ETL process were introduced in the 1970s as the process was introduced to integrate and load data from multiple applications for computation and analysis, finally becoming a key factor in data warehousing projects. Today, ETL is often used to extract raw data from different systems,

performing transformations into a format by cleansing and organizing data in a way that improves data quality for data analysis to address specific business needs, then loading data into a data warehouse. As shown in Figure 1 above ETL is a three-phase process.

Extract

In the data extraction phase, the required data is extracted from one of the data sources such as ERP, CRM, data from vendors, and others. It is then stored in a staging area, where data can be transformed, validated, or eliminated. Data processing is recommended to be done in the staging area to prevent potential performance issues from the source system. In addition, data can rollback easier in the staging area if any data corruption happens.

Transform

The second phase of ETL is transformation. This phase is considered the most complicated part since it requires expertise in data manipulation and domain knowledge. The raw data extracted from source the system need to be normalized and standardized by cleansing, filtering, splitting, joining, mapping, and transforming based on a set of rules from business to make the data fit with scenario analysis. This phase helps ensure that the data is accurate and consistent before moving it into the data warehouse.

Load

After the data is transformed, the load phase moves data into a target data warehouse. This data warehouse could be in the cloud or on-premises. Typically, there is two kinds of loading methods which are full load and incremental load. In full load, entire data from source is loaded to the data warehouse. With incremental load, only new and updated records are moved to data warehouse. Once the necessary data is loaded into data warehouse, it can be used for business analysis and business intelligence operations.

3.3 Data Source

Data sources are physical or digital repositories where the information is obtained. For instance, one physical source could be a paper. Data in physical format is often converted to a digital format so that the information system can read, store, and share it. In digital data sources, the information can be a management system, a database, and flat files such as XML files, spreadsheets, and others. (IGI global, 2022). Today, modern data sources can consist of structured data and unstructured data. Unstructured data can be the content of social media or emails such as texts, images, or videos. Structured data is the most used in enterprise systems such as EPM, ERP, and others. ERP and EPM are considered the most important data source of a company, their data can be used to enhance business reporting, increase productivity, improve management costs, eliminate repetitive tasks, and can speed up customer response and consolidate data.

3.3.1 Oracle ERP cloud

Oracle ERP cloud was introduced by Oracle Corporation in 2012. It was developed based on Oracle Fusion Applications, which is a cloud-based application, and an end-to-end software as a service. It is developed to help organizations to manage their business processes, including financial management, procurement, project management, supply chain management, risk management, and more. It offers both public and private cloud implementations and supports hybrid deployment. (Wikipedia, 2023). Moreover, the solution also consists of real-time insights and data access that can support organizations to identify trends and make better business decisions. One of the key benefits of Oracle ERP is its ability to integrate flexibly with many other applications.

3.3.2 Oracle HFM

According to Concentric Solution (2023), Oracle HFM is a comprehensive, web-based financial consolidation, reporting, and highly scalable software solution. It is designed to solve the tasks of consolidation, and financial reporting, and help companies manage their financial reporting better. HFM also supports defining the multiple consolidation methods, including currency translation and intercompany elimination. One of the main advantages of HFM is its ability to automate the financial consolidation processes that can reduce financial reporting time, and eliminate the risk of manual errors. In terms of implementation, HFM is a flexible solution, which can be configured and deployed easily. The solution is now available both cloud and on-premises.

3.4 Data Modelling

Data modelling is the process of creating graphical representation of data and the relationship between data in a way that is grouped, organized, stored, and retrieved. Data models are created based on business needs and it is also considered as a roadmap, an architect's blueprint or a document that accurately reflects the relationship between data structure in database and business processes. Data models play a critical aspect of supporting business processes and planning IT and database design (IBM, 2023).

Data is usually modelled on three different types, including conceptual, logical, and physical data models. Conceptual data models provide a high-level view of the business or analytics process and the relationship between business entities and business rules. It is not restricted to any specific database. Logical data models show the conceptual models and define data structures in detail, including tables, attributes, keys, data types, other characteristics, and the association between the tables. Like conceptual models, logical models are not tied to a specific technology platform. Physical data models show how the logical model is physically created in a database. They define the details of structures that the

database will use to store data, including tables, columns, fields, indexes, constraints, triggers, and other DBMS elements. (Craig Stedman).

There are different techniques of data modelling, including hierarchy, network, relational, entity-relationship, dimension, object-oriented, graph, and others. Each data modelling technique will have a different diagram. Figure 5 is a sample dimensional data model. Dimensional models are designed to optimize for aggregation, which can support querying and analysis. Dimension data modelling is used in data warehousing and BI applications. Data is organized into a set of dimensions and fact tables, where dimensions present entities or objects such as products, vendors, customers, time and locations, and facts present transactions and measures such as revenue, sales, or quantity. The common type of dimension model is star schemas, which connect a central fact to the surrounding dimensions. (Craig Stedman).

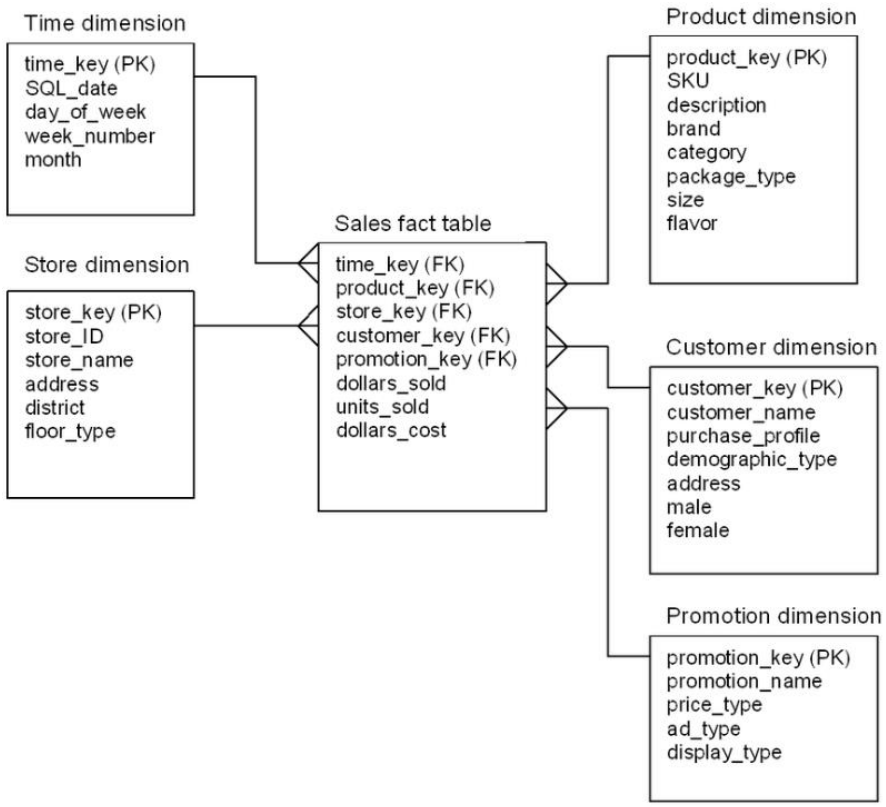

Figure 5. Sales Dimensional Model (Sergio Lujan-Mora, 2003).

3.5 Conceptual Framework

The main components of an ETL process are summarized in the conceptual framework presented in Figure 6, which is based on existing knowledge discussed above as well as the case company documentation.

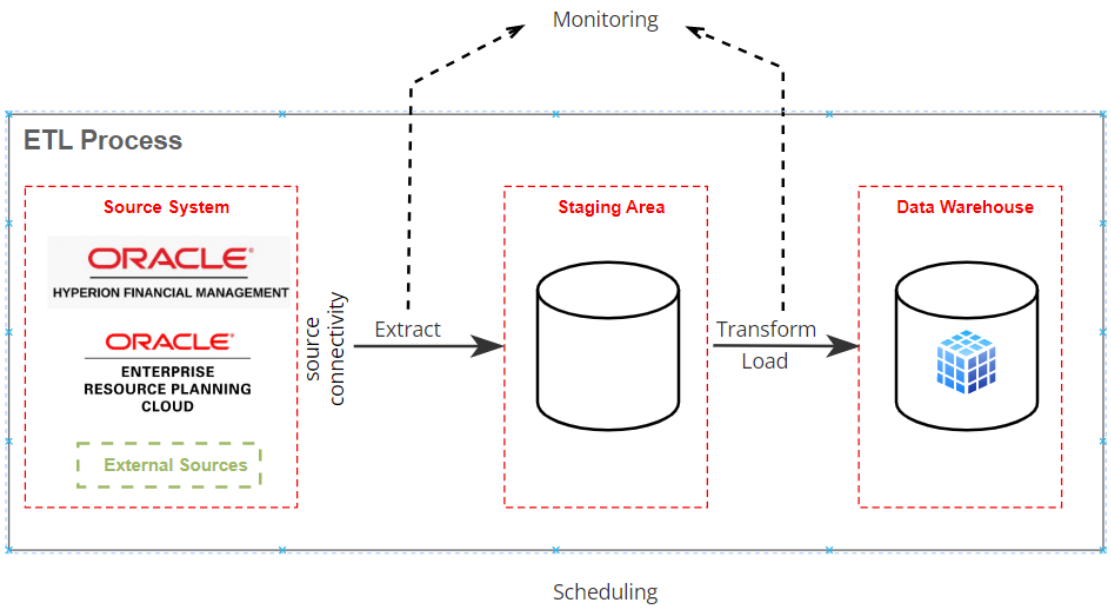

Figure 6. ETL process

The Figure 6 shows an ETL framework consisting of 6 stages, which are source connectivity, data extraction, data transformation, data loading, monitoring, and scheduling.

Table 2. ETL framework.

Stage	
Source connectivity	To create the connection to data sources
Data extraction	To retrieve data from various sources such as databases, files, and others, and data is then stored in a staging area
Data transformation	To cleanse, filter, and transform data into a common format

Data loading	To load transformed data in a data repository, such a data warehouse, or data lake
Monitoring	To monitor the ETL process to prevent data loss and maintain the integrity of the data
Scheduling	To ensure data is up to date on a regular basis based on specific intervals or time

The conceptual framework depicted in Figure 6 was assembled from concepts discussed in Section 3, and the case company documentation, to serve as a guide for analysing the case company challenge.

4 Current State Analysis

This section discusses the outcome from the current state analysis (CSA) of the ETL for Oracle HFM and ERP cloud implemented in the organization. The executed CSA identifies the weaknesses and the challenges of the processes. The study was performed based on the needs of stakeholders and the conceptual framework presented in Section 3.5.

4.1 Overview of the Current State Analysis

As mentioned in Section 2.3, regular workshops and meetings were organized with stakeholders to review the performance and efficiency of the current ETL processes for Oracle HFM and ERP cloud. These discussions focused on evaluating each step of the current ETL processes, including data extraction, data transformation, data loading, monitoring, and scheduling, to detect bottlenecks, problems, and inefficiencies. While discussing the barriers and the flaws in data pipeline, and reviewing the internal documents, it became clear that the ETL processes for Oracle HFM were inflexible and outdated, and the pipeline was not being evaluated each step of the ETL processes.

Regarding ETL processes for Oracle ERP cloud, a limitation with data output from web services was identified, which is restricted to only 10 MB during the data extraction step. To thoroughly evaluate the process, the CSA was conducted for all the stages of the ETL process.

Upon reviewing the existing literature and company documentation, an ETL framework was created as depicted in Figure 6. Key elements of the ETL processes, including data extraction, data staging, data transformation, data loading, others. were identified. The CSA is performed to evaluate the status of each stage of the ETL process, such as assessing the efficiency and reliability of the data extraction stage, assessing the accuracy and consistency of data transformation stage, assessing the efficiency and effectiveness of monitoring stage.

4.2 Overview of the Stakeholders Needs

There were different requirements raised by stakeholders in the regular workshops, which can be categorized into four main points.

The needs of the stakeholders include, firstly, the ability to view financial reports from different dimensions. These dimensions are defined as the metadata tables that are categorized and organized in the database. The reports need to have the drill-down capability so that users could see the figures at any level. It meant the reports need good data modelling, which includes hierarchy dimensions.

Secondly, as the cloudification is ongoing in the case company, the Oracle data warehouse will be moved to the Oracle cloud. The part of data will also be migrated to Amazon cloud storage. Additionally, Oracle HFM is expected to move to the cloud in the coming years. To meet the needs of stakeholders, the new ETL processes must be compatible with cloud applications and have the flexibility to adapt to changes in source connectivity and target data warehouse.

The next requirement is related to error monitoring and handling. The users require an end-to-end monitoring process that sends a prompt alert to data engineers when the issues occur at any stage of the ETL process. This will enable stakeholders to quickly troubleshoot any problems that may cause interruptions in the ETL process.

Finally, the last requirement concerns the issue in the Oracle ERP cloud where the data extraction step of the ETL process ends with an error message when a dataset is too large. The issue is not happening frequently, and is difficult to figure out, as the current ETL processes have not yet identified the extracted data size and the limitation of output data from the web services. The stakeholders require a comprehensive investigation of this case.

4.3 Analysis of the Current ETL process for Oracle HFM and Oracle ERP cloud

The conceptual framework presented in Figure 6 shows the main components and the stages that are the key factors in an ETL process. The analysis of each stage in the ETL process was performed in detail based on the conceptual work. As a result, the output of the analysis is the strengths and the weaknesses of the current ETL. And a new solution was proposed to improve the current processes.

4.3.1 ETL process for Oracle HFM

Figure 7 shows the main components of the current ETL process for Oracle HFM. The analysis of the process was conducted based on the conceptual framework in section 3.5.

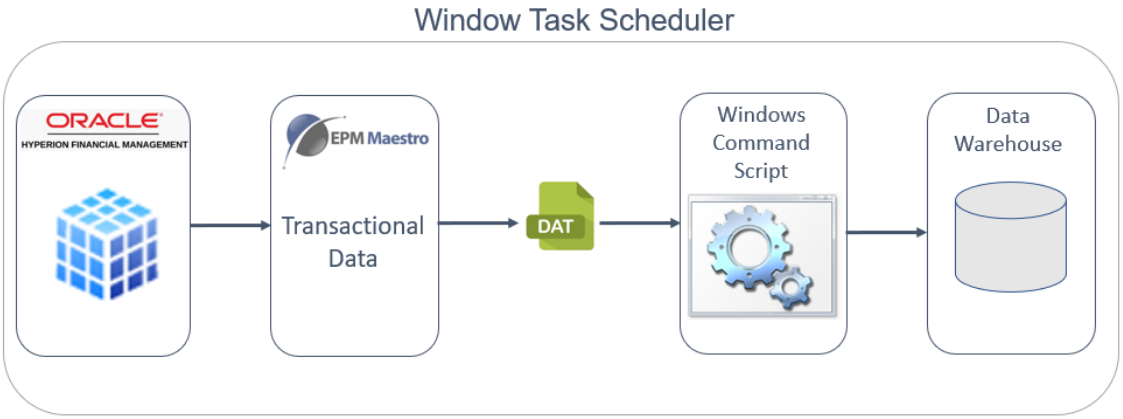

Figure 7. The current ETL process for Oracle HFM

Source connectivity

As shown in Figure 7, the case company is using EPM Maestro to extract data from Oracle HFM. EPM Maestro is a third-party software solution designed to enhance the user experience of the HFM system. EPM Maestro provides an administrator productivity and advanced automation capability that improves quality, reduces time for administrative tasks and manual work, and minimizes errors. (EPM Maestro, 2023).

According to internal documents, EPM Maestro was installed on top of HFM servers. Establishing a connection to HFM by providing the information as the screenshot below, including connection name, host, and port.

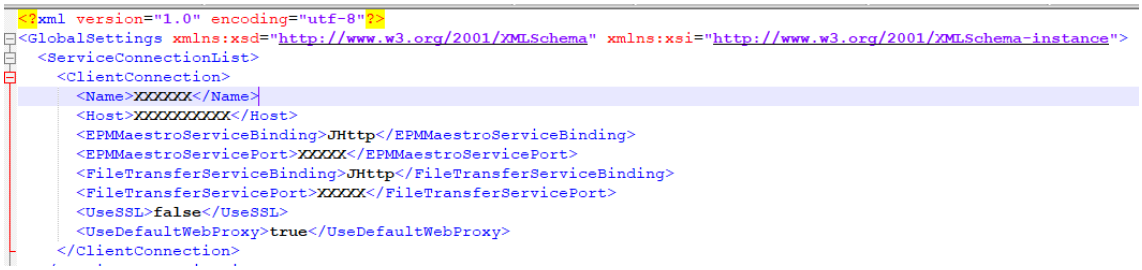

Figure 8. EPM Maestro connection

Data extraction

Only transactional data was extracted due to a lack of transformation methods, or libraries in window command scripts (CMD). To address this issue, data extraction using EPM Maestro was utilized as a user-friendly solution for data extraction from HFM. In addition, the configuration of extracted data and the creation of a data flow were straightforward in EPM Maestro. After completing the data extraction stage, a DAT-type flat file was extracted and stored on the local disk of the HFM server. Multiple DAT files were extracted separately using the EPM Maestro.

Data transformation

The process lacked the data transformation stage. The CMD scripts do not provide the capability for transforming data, only moving multiple DAT files to the data warehouse server independently. This is a script triggered to execute a task flow in EPM Maestro and move a data file to the warehouse server.

Data loading

Like data transformation, the process also lacked the data loading stage. As mentioned in data transformation, the CMD scripts only migrate the data file to the data warehouse server as shown in Figure 9. This solution was designed only for on-premises.

```
REM Thuan changed destination from \\xxxx\xxx\MS to \\xxx\xx\MS\WAF
Robocopy D:\hfmserver\Extract_WAF\QVServer "\\servername\hfm\MS\WAF" /MIR /LOG+:D:\Scripts\Logs\WAFTimesPerDay.log
```

Figure 9. Move the extracted files to the data warehouse server.

Monitoring

One advantage of the current process was that it monitored the end-to-end process using a log file. Each stage of the process was recorded in the log file, which was sent to data engineers for troubleshooting in case of any issues.

Scheduling

The process was triggered to run automatically at specific time by creating tasks in windows Task Scheduler.

4.3.2 ETL process for Oracle ERP cloud

Like Section 4.3.1, Figure 10 is the architecture of the current ETL for the Oracle ERP cloud. The analysis of the process included 6 stages that are aligned with the conceptual framework in section 3.5.

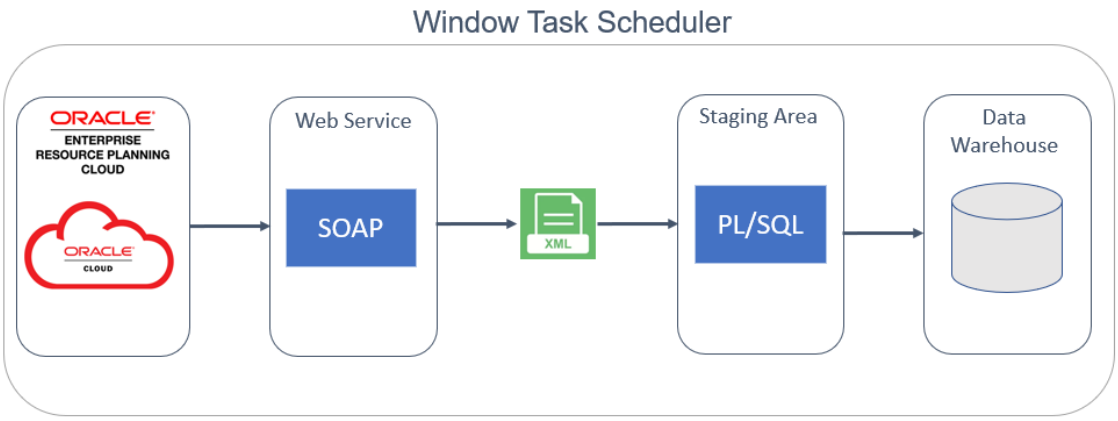

Figure 10. The current ETL process for Oracle ERP cloud

Oracle ERP cloud supports data extraction through both Web Service SOAP and API. The current ETL process uses SOAP requests to extract data via a BI Publisher Web Service. BI Publisher Web Service refers to a set of web services that allow client applications to access and extract data from the Oracle ERP

cloud. Oracle BI Publisher is a built-in reporting tool that enables users to create and publish reports, as well as extract data using SQL queries. (Oracle, 2023). The tool can be considered as a simple and quick way to configure a web service for the data extraction from the Oracle ERP cloud. However, a disadvantage of the BI Publisher web service is that the output file is limited to only 10 MB. The existing processes include metadata and transaction data, which are already defined in Oracle BI Publisher for use in the ETL processes.

The connectivity to the Oracle ERP cloud system is established when a soap request is sent to the system. The authenticity of the system, including host, username, and password is encrypted and passed as a part of the authentication header in the SOAP request. It is a secure way to authenticate the client with the system that ensures only authorized clients can access to the system. Figure 10 below is a sample of the request payload.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v2="http://xmlns.oracle.com/oxp/service/v2">
<soapenv:Header/>
<soapenv:Body>
    <pub:runReport>
        <pub:reportRequest>
        <pub:byPassCache>true</pub:byPassCache>
        <pub:parameterNameValues></pub:parameterNameValues>
        <pub:reportAbsolutePath>/XXXX/Reports/Account Master Data.xdo</pub:reportAbsolutePath>
        <pub:sizeOfDataChunkDownload>-1</pub:sizeOfDataChunkDownload>
        </pub:reportRequest>
        <pub:userID>xxxxx</pub:userID>
        <pub:password>**********</pub:password>
    </pub:runReport>
</soapenv:Body>
</soapenv:Envelope>
```

Figure 11. Request payload sample

As seen in Figure 10, the data extraction is developed using PL/SQL in the staging area. when the process is running, it sends the SOAP requests to the Oracle ERP cloud system (as Figure 11) and then receives an encrypted response in XML format (as Figure 12).

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soapenv:Body>
      <runReportResponse xmlns="http://xmlns.oracle.com/oxp/service/PublicReportService">
         <runReportReturn>
            <metaDataList xsi:nil="true"/>
       <reportBytes>
ctMTAtMTRUMDk6NTU6MjIuMDAxKzAwOjAwPC9DUkVBVElPTl9EQVRFPjxUQVhfQ09ERT5JUFBfTk9ORV9WQVQ8L1RBWF9DT0RFPjxFRkZFQ1RJVk
VfRU5EX0RBVEU+PC9FRkZFQ1RJVkVfRU5EX0RBVEU+PE1FTU9fTElORV9JRD4zMDAwMDAwMDE0NzI3NDM8L01FTU9fTElORV9JRD48REVTQ1JJUF
RJT04+RG9hbmggdGh1IGNobyB2YXk8L0RFU0NSSVBUSU9OPjxNRU1PX0xJTkVfTkFNRT5DSE8gVkFZPC9NRU1PX0xJTkVfTkFNRT4KPC9HXzE+Cj
wvREFUQV9EUz4=</reportBytes>
            <reportContentType>text/xml</reportContentType>
            <reportFileID xsi:nil="true"/>
            <reportLocale xsi:nil="true"/>
         </runReportReturn>
      </runReportResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

Figure 12. Encrypted response sample

After receiving the encrypted response from the web service, the transformation data decrypts the encrypted data and loads it into a temporary table in the staging schema. After that, the transform components involve cleaning, normalizing, and transforming data into a common format that is compatible with data warehouse tables.

The transformed data is then loaded into the data warehouse. This process involves writing the data from the staging tables to the fact tables.

There is a log table that contains the logged data from the process. Each stage of the data is recorded into the log table, allowing to access it in case of any issues to check what happened during the process.

Like the ETL process for Oracle HFM, the process is also triggered to run automatically at 2:00 AM every day by creating tasks in the Windows Task Scheduler.

4.3.3 Data Types

Data types are a critical factor in the ETL process since incorrect data types can lead to incorrect data analysis, results, and conclusions. The data type of the ETL depends on the type of data being extracted from source systems, transformed, and loaded to the data warehouse, which would determine how the data will be

processed, stored, and analysed. There are various data types in the ETL, but as mentioned before, the scope of the thesis was based on two data source systems, which are Oracle HFM and Oracle ERP cloud. Thus, the data types in the current ETL process can be categorized into numbers, characters, and date and time. Numeric data types include integers, floating-point numbers, and decimals, which are used to present measurement data such as revenue, sales, or quantity. Character data types refer to strings and characters and are utilized for the representation of text data, including accounts, customers, entities, currencies, and others. Date and time types include data, time, and timestamp data and are utilized for the representation of data and time information.

4.3.4 Data Structure

A specialized format for organizing, processing, retrieving, and storing data is known as a data structure (GeeksforGeeks, 2023). The data structure is important in the ETL since it can ensure that the data is properly organized in a way that data can be easily queried and analysed. The data structure of Oracle source systems in the data extraction stage is different. In Oracle HFM, the transactional data is extracted to the DAT file, where the data structure is organized in rows and columns. However, an extracted APP file metadata is disorganized, making it difficult to process and integrate into the existing ETL processes. Therefore, metadata cannot be effectively processed so that it can be used for data modelling. Meanwhile, the data extraction in Oracle ERP cloud is formatted using XML structure.

Due to the differences in the data structure from the source systems, distinct ETL processes are necessary to transform extracted data into the appropriate data structure in the data warehouse.

4.3.5 Data Modelling of the current ETL process

As described in Section 3.4, there are various techniques of data modelling. However, the case company only uses dimensional data modelling to analyse its

data. This data structure consists of relational tables stored in the data warehouse.

As mentioned in section 4.3.1, the ETL for Oracle HFM does not possess the capability of data transformation, as the extracted metadata file is disorganized and cannot be flattened into the hierarchy using a CMD. Only transactional data is loaded into the data warehouse, and it is stored in a single table. As a result, the data modelling for analysing HFM data only contains a single fact table and no dimensional tables, which means that the drill-down functionality is not supported in HFM visualization.

Regarding Oracle ERP cloud data modelling, it employs dimensional data modelling. Both fact tables and dimensional tables are created in the data warehouse. The current ETL processes are an up to date and simplify data extraction and data transformation. The star schema model, shown in Figure 13, is used to analyse Oracle ERP cloud data. The data is analysed based on dimensions such as business unit, ledgers, the charge of the account, and date time. The facts table provides the data for the balance sheet and income statement reports.

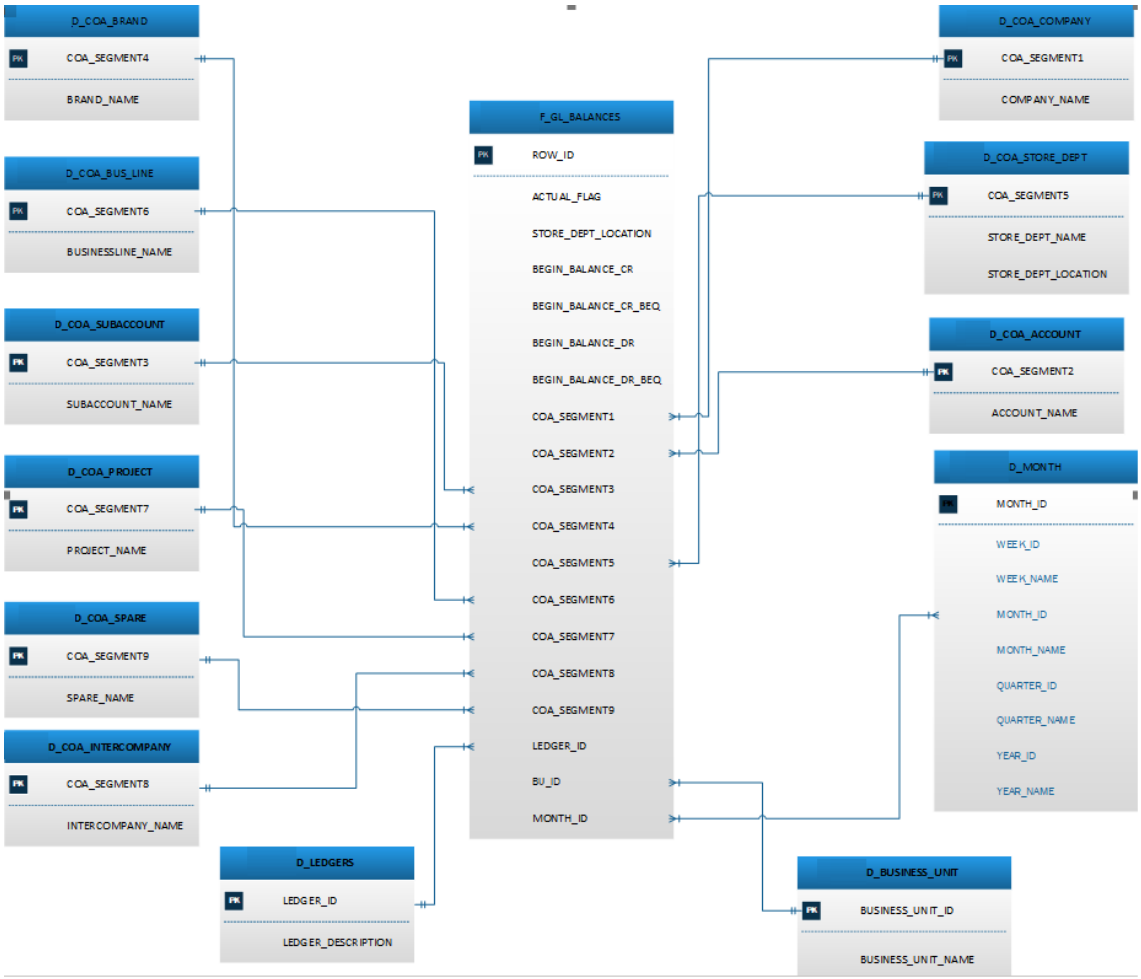

Figure 13. Star schema model for Oracle ERP cloud

4.4 Summary of Strengths and Weaknesses of the Current ETL Processes

Based on the information collected from the conceptual framework, the current state analysis, discussion with stakeholders, the internal materials, and experience, the strengths and the weaknesses of the current ETL process were identified.

The strengths of the current process are that the ETL scripts are simple and easy to modify. It does not require too much technical expertise to manage and monitor. The connection to source systems is one of the advantages of the current process, as it is easy to create and edit the connection in case the source

systems move to another location. The log is also clear, as each stage of the process is carefully recorded in the log files and the log tables. The data log can support the monitors to troubleshoot and investigate the issues effectively.

The weaknesses of the process have been identified for both Oracle HFM and Oracle ERP. Most of the issues with the current ETL process are related to the outdated ETL of the HFM system, which has not been updated in a long time. Additionally, the fact that the current systems are gradually being migrated to cloud environments has contributed to these issues. Furthermore, the current CMD scripts do not support connections to the new cloud environments such as Oracle HFM cloud or AWS S3. Moreover, the stakeholders have expressed a need for dimensional data modelling to enable the drill-down reports, but the ETL process lacks the necessary libraries to transform and flatten the metadata into the hierarchy structure. Additionally, the stakeholders also want to have a statistical report that shows the number of rows of data loaded into the data warehouse.

When performing the ETL on Oracle ERP cloud, there is only one issue: the limitation of data output from web services is only 10 MB. Therefore, if the dataset is too large or the data exceeds 10 MB, the process may encounter an overload issue.

The thesis proposes a solution to improve the current ETL processes based on the finding from the current state analysis. Further discussion on the topic will focus on the technical details of the new ETL process and its implementation.

5 Building Proposal

This section integrates the findings from the analysis of the current stage with the conceptual framework to develop a proposal. The proposal aims to tackle the problems, improve the current ETL processes, and mitigate any weaknesses, presented in section 4.4.

5.1 Overview of the Proposal Building Stage

As mentioned above, the proposal building aimed to improve the current ETL process and mitigate the weaknesses. The proposal was developed based on the findings of the analysis of the current stage, the conceptual framework, and the result of cooperating with the stakeholders during the workshops. The input Data 2 is the requirements and practical issues collected from the stakeholders during the meeting to analyse before making the proposal.

The typical ETL framework in Section 3 was developed based on the literature review and existing knowledge. In Section 4, the study conducted a CSA and identified four different weaknesses in the ETL for Oracle HFM, including the issues with connecting to the Oracle HFM cloud, lack of data transformation, metadata extraction, and connectivity to the data lake AWS S3. Additionally, a limitation in the Oracle ERP cloud source is identified where data output is restricted to a maximum of 10 MB. The proposal in this thesis was built based on these findings to address and overcome the weaknesses.

Using the conceptual framework as a guideline, the weaknesses in each stage of the current ETL process were analysed and developed a proposal to address each one. Due to technical differences between Oracle HFM and Oracle ERP Cloud source systems, separate proposals were created for each system. Moreover, observations of the system and the internal documents from the findings of Data 1 were used to propose the solution.

5.2 Initial Proposal

For ETL for Oracle HFM, Python was proposed for data transformation and managing the end-to-end process. Regarding Oracle EPM cloud, Python and EPM Automate and FDMEE were proposed to develop a new ETL solution. For ETL for Oracle ERP cloud, it was proposed to modify BI Publisher web service and PL/SQL procedures.

5.2.1 Proposal Element 1: The connectivity to Oracle HFM cloud

As discussed in the CSA, the source system Oracle HFM is planned to be migrated to a cloud environment. There are two options to choose from: Oracle HFM can be installed in a virtual machine in the cloud, or it can be migrated to the Oracle EPM cloud software as a service. The first option involves moving HFM to the virtual machine located in the cloud, which works in the same way as HFM on-premises. The only difference is the location of the system, which is installed in the virtual machine in the cloud. The connectivity of the option does not need to change, as the case company is still using EPM Maestro. In this configuration, EPM Maestro would be installed in Oracle HFM virtual machine in the cloud, just as it is on-premises.

The second option is to migrate Oracle HFM to Oracle EPM cloud software as a service (SaaS), which could be very risky for the business processes as it may cause changes to the way users work. In the scope of this work, the focus is on ETL technical aspects. Oracle EPM cloud is a SaaS offering in which Oracle provides the software and hosting environment for running the EPM applications. Customers are not able to install or customize anything in the environment and can only access the application over the internet through a web browser. This means that the application is fully managed and maintained by Oracle. Therefore, the connection and data extraction are shifted to new solutions that utilize standard functionalities within Oracle EPM cloud, especially Financial Data Quality Management Enterprise Edition (FDMEE) and EPM Automate, instead of using EPM Maestro.

The proposed solution for the EPM cloud option involves using FDMEE to extract data from the Oracle EPM cloud. EPM Automate is then employed to automate the data extraction in FDMEE and download data files into a local machine. EPM Automate enables administrators to execute FDMEE tasks from the command line, without having to navigate to the system through the web interface. This can save time and effort when performing repetitive tasks. To configure and define the data extraction rules in FDMEE, several steps are required. The process of

automating FDMEE tasks through the command line in EPM Automate is also presented in detail in Section 6.4 of the implementation.

5.2.2 Proposal Element 2: New data transformation for Oracle HFM source

It is proposed to use Python programming language. Python is currently the most popular programming language for data science (Rohit Sharma, 2022) and supports powerful libraries for data discovery and data processing, such as Pandas, Numpy, Re, String, NLTK, and others. These libraries enable fast processing of large amounts of data and make it easy to handle complex data structures.

As mentioned in Section 4.3.1, multiple DAT files are extracted separately and stored in the HFM server. Those files are the same structure and are transferred to a single table in the data warehouse independently. It is proposed to use Python libraries to filter, transform, delete silos data, and merge the files into a data frame before loading it to the data warehouse.

It is proposed to use Python to manage the end-to-end ETL processes since Python supports calling to run the subprocesses of EPM Maestro or EPM Automate to extract data from the source Oracle HFM. Additionally, Python is supported for the connection to S3 and Oracle data warehouse for data loading.

5.2.3 Proposal Element 3: New ETL process for metadata from Oracle HFM source

As discussed in 4.3.5, the current data model for HFM analysis only has a fact table, but the actual needs of stakeholders wish to have dimensional data modelling, which can support them building the drill-down reports. Thus, it is proposed to have new dimensions, including entity, and account in the data model.

Python is also suggested as the language for developing this end-to-end process. This is a new and very complicated ETL process as metadata files extracted from HFM, which is disorganized, and it requires significant technical effort to flatten from a messy APP file into a hierarchy structured that can be stored in a dimensional table in the data warehouse.

From metadata extraction aspects, EPM Maestro is proposed to extract metadata from Oracle HFM on-premise, and EPM Automate is suggested to connect to Oracle EPM cloud for metadata extraction on the cloud environments. Both tools can also be embedded into Python scripts to create an end-to-end ETL process. The details of technical source code are presented in Section 6.4 Implementation.

5.2.4 Proposal Element 4: The connectivity to a central repository AWS S3 and Oracle data warehouse for data loading

Python with powerful libraries for data processing in cloud environments is proposed for the data loading stage. Python is a common language used on top of AWS S3, and it supports the libraries such as boto3, and botocore that are used for uploading a data file to the S3 environment.

In terms of the Oracle Data warehouse, Python supports the libraries such as cx_Oracle, and sqlalchemy, which can write SQL queries to directly collect the data or filter data in the Oracle data warehouse from the Python environment and the most important part of the data loading stage is inserting the data into the tables.

5.2.5 Proposal for improving the ETL for Oracle ERP cloud

Generally, the current ETL for Oracle ERP cloud is working fine, with only one issue presented in Section 4.4: the data output file of the web service in Oracle ERP cloud is limited to 10 MB only and the process is often ended up with an extraction error when the data query in web service greater than 10 MB. As noted earlier in Section 4.3.2, the web service in the Oracle ERP cloud is built using

Oracle BI Publisher, which primarily relies on SQL queries to collect the data. To address the issue, it proposed to modify the web service by adding two parameters: “from row number”, and “to row number”, which can divide the output dataset into smaller batches. The number of rows in the dataset often exceeds one million when the issue occurs, so it is proposed to divide the dataset into five batches, each containing up to 300000 rows.

Following the addition of two parameters into the SQL queries in Oracle BI Publisher, it is also proposed to update the PL/SQL code in the staging area. Specifically, the code should iterate 5 times to call web services instead of a single call, as previously implemented. Each iterate should retrieve a maximum of 300000 rows of data from the web service.

5.3 Summary of the Proposal

This thesis proposal aims to improve the current ETL processes of the case company by developing a technical solution that aligns with the conceptual framework presented in Section 3.5. The proposal includes source code components, and the technical details are provided in Section 6.4.

There was a wide-ranging discussion with the end users during the initial proposal. Most of the discussion focused on whether the proposal could solve the current problems or be compatible with cloud environments, and other topics. As it was a technical proposal, the demo ETL processes were built in the Test environment to demonstrate its efficiency, and the results of the proposal were sent to stakeholders for validation.

6 Validation of the Proposal

In this section, the outcomes of the validation phase are presented, along with a discussion of the developments to the initial proposal that was made based on stakeholder feedback. At the end of the section is the implementation of the final proposal.

6.1 Overview of the Validation Stage

During workshops, the proposals for improving the current ETL processes for both systems Oracle HFM and Oracle ERP cloud were validated with the stakeholders. The proposed elements were assessed and accepted to build the demo processes based on current resources in Test environments. The resulting data from the demo processes, in excel format, was then sent to users for validation.

To validate the results, the first step was to test the resulting data from the demo processes. The users utilized the Power BI reporting tool to visualize the data based on their needs. It has been successfully tested, as the dimensional data model, including dimension tables and fact tables, was successfully created and the users were able to generate the drill-down reports.

The next step involved presenting the findings and the feedback resulting from the first step to the stakeholders.

After discussing and presenting the initial proposals, the final step was to collect the feedback from the presentation, use it to further develop the proposal based on the feedback (Data 3) and produce the final proposal.

6.2 Findings from Data 3 Collection

Based on the feedback collected during the workshops, the proposed solution effectively addresses the weaknesses of the current process, such as the

compatibility with cloud environments and issues with data processing in data transformation and ETL of Oracle ERP cloud. However, minor adjustments were needed for metadata transformation in HFM. The current process manually managed three pieces of metadata for each member: member code, member name, and member description and manually updated into the data warehouse. This was not yet mentioned in Data 1 and Data 2 as some users needed to see the results of the demo processes. Therefore, the initial proposal only flattened member code into the hierarchy. However, this adjustment was a simple request when the field member description already exists in data extraction.

As the proposal received much positive feedback, the stakeholders discussed the plan to build a fully end-to-end process in quality assurance environments (QA) so that users can have a clearer picture of the new solution and the validation is more accurate since users can validate the data on actual business datasets. The next steps are the final proposal in Section 6.3 and the implementation, including the configuration and the source code, which can be seen in Section 6.4.

6.3 Final Proposal

Based on finding Data 3 collection, it was determined that there were no significant alterations made to the initial proposals. The proposed main components of an ETL process were designed based on the conceptual framework and the stakeholders' needs to address the current weaknesses of the existing ETL processes. However, a minor adjustment was made to the metadata description for the Oracle HFM process. Since all the metadata information was available during data extraction, it was a simple step in the data transformation process in Python to add the metadata description to the dimension hierarchies.

The final proposed architecture for Oracle HFM and Oracle ERP cloud can be seen in Figure 14 and Figure 15.

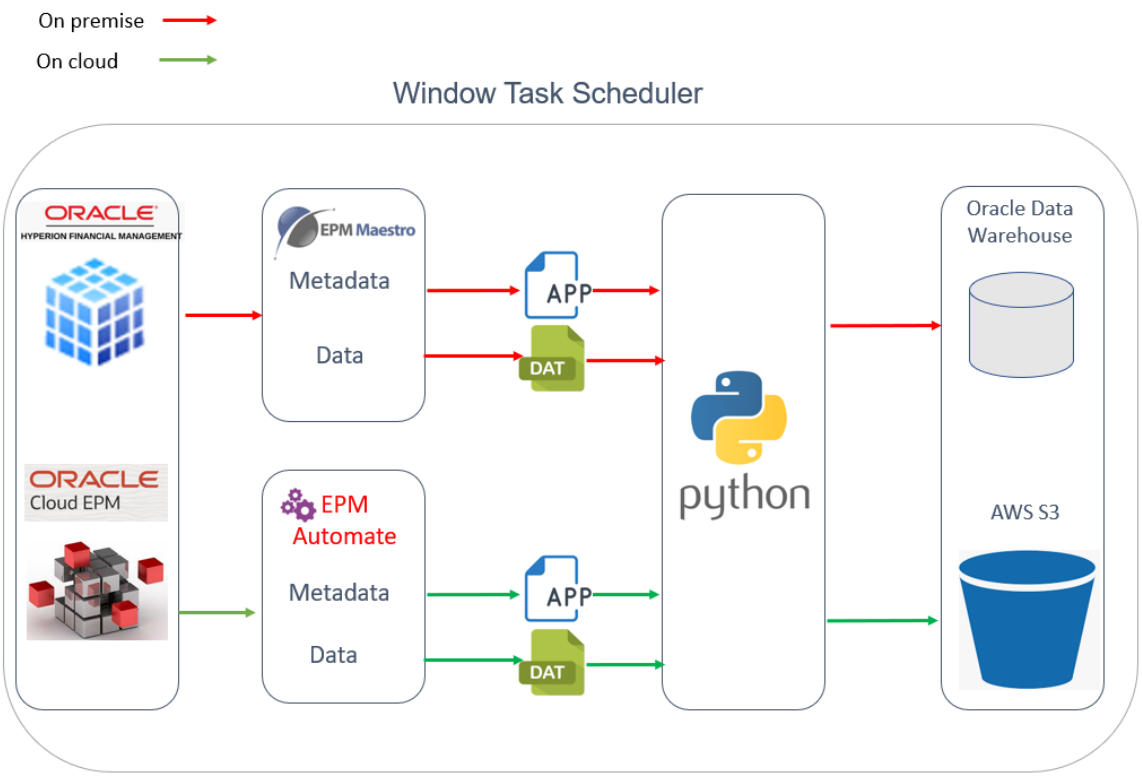

Figure 14. Final proposal for Oracle HFM

As seen in Figure 14, the red arrow represents the proposed on-premise process, and the green arrow represents the proposed on-cloud process. The main components and the details of the proposals were presented in the initial proposals.

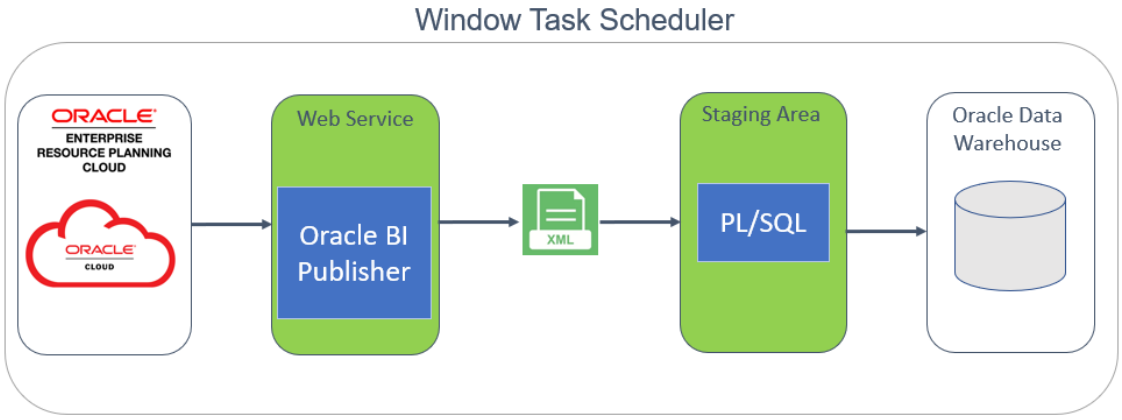

Figure 15. Final proposal for Oracle ERP Cloud

The green highlights indicate the final proposals for the Oracle ERP cloud, as mentioned in Section 5.2.5. The proposal involves adding two parameters (“from row number”, and “to row number”) to SQL queries in the web services BI Publisher and updating the PL/SQL code in the staging area.

6.4 Implementation

Several ETL processes were implemented in the system. However, this section presents three typical ETL processes: ETL for metadata for Oracle HFM, ETL for transactional data for Oracle ERP cloud, and ETL for Oracle ERP cloud. The implementation of new ETL processes included three stages: developing the solution in the Sandbox environment for technical testing, then moving it to QA environments for user testing, and deploying it to Production.

Python, EPM Maestro, and EPM Automate were required to be installed in the staging servers. To use Python for data processing and data load, several Python libraries were installed in the Python environment. Table 2 provides the details of the libraries and the third-party tools used in the proposals.

Stage	**Libraries and Third parties**
Data extraction	Subprocess, EPM Maestro, EPM Automate
Data transformation	Pandas, Numpy, Datetime, os
Data load	Boto3, psycopg2, cx_Oracle, sqlalchemy
Data monitoring	Smtplib, Email

Table 3. Python libraries and third-party tools.

6.4.1 ETL for metadata in Oracle HFM

This is the new ETL process that was proposed to improve the data modelling. It was developed by the combination of EPM Maestro and Python. EPM Maestro was used to extract metadata from HFM and Python was used for managing end-to-end process, data transformation, and data load.

For data extraction, a task flow to extract metadata from HFM was created in EPM Maestro as presented in Figure 16. This task flow is an example of the metadata Account extraction.

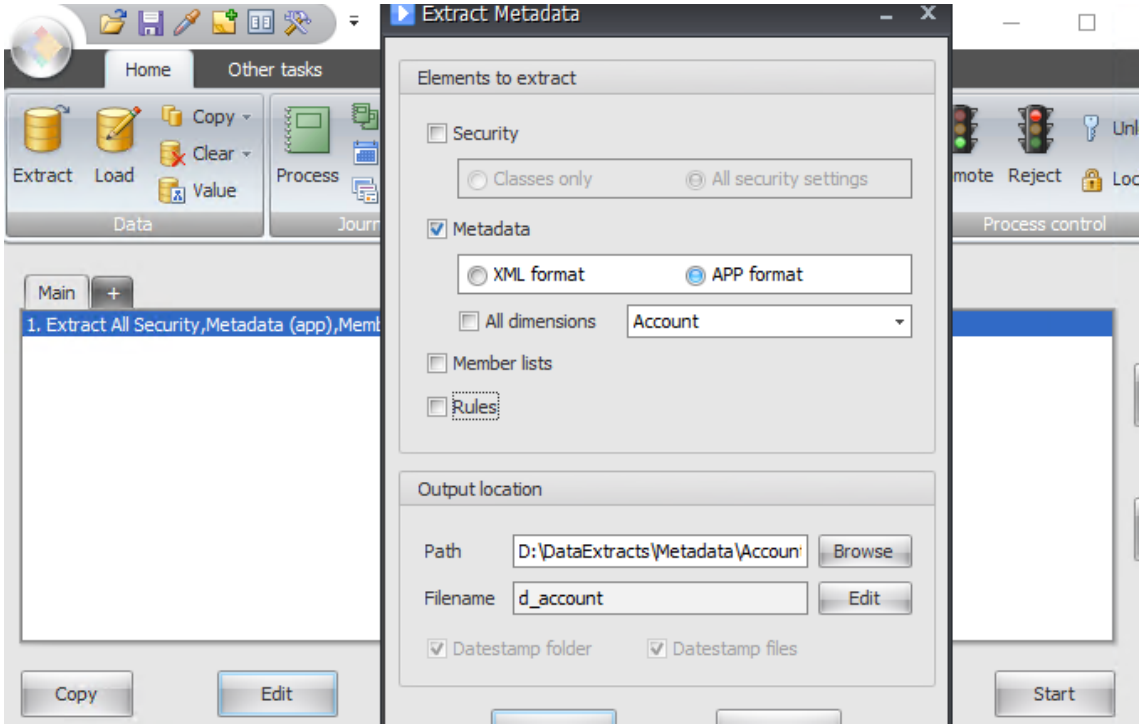

Figure 16. Task flow in EPM Maestro to extract metadata.

The extracted metadata is an app file and consists of two parts: member descriptions and hierarchies as shown in Figure 17. Therefore, the data transformation was divided into two parts: part 1 processing hierarchies and part 2 processing member descriptions. And finally, the data of parts 1 and 2 were combined to load into dimension tables in the data warehouse.

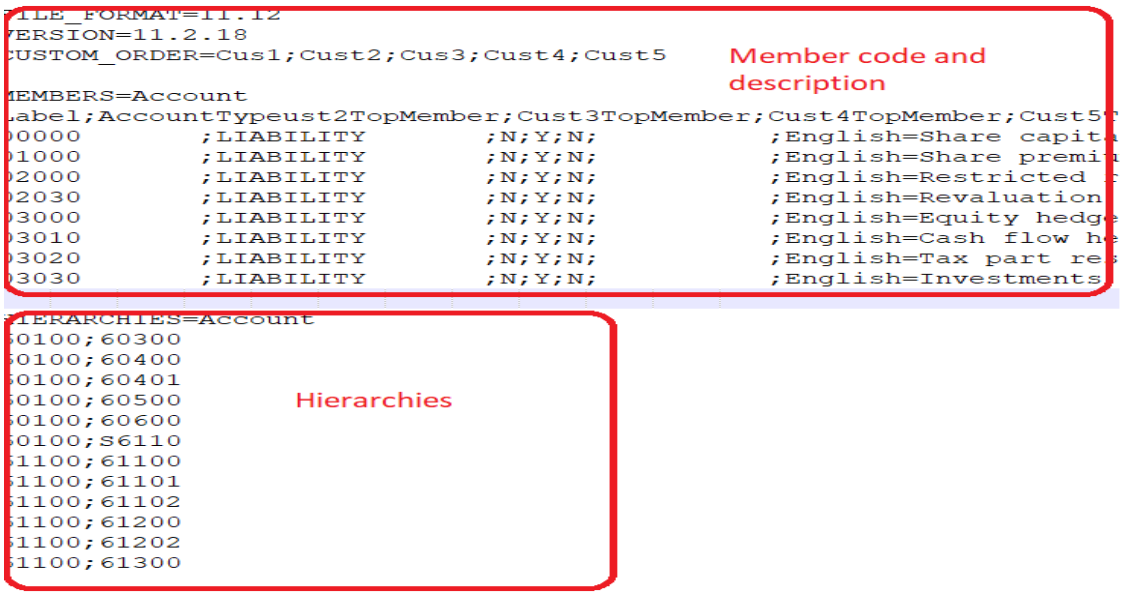

Figure 17. Extracted metadata from HFM.

The processing of Python code can be categorized into four main stages, which are data extraction, data transformation, data loading, and data monitoring. Firstly, the python code is activated to the task flow process from EPM Maestro, which has been configured to extract metadata as shown in Figure 16. The extracted metadata is then loaded into a list and converted to a data frame, as shown in Figure 18.

```
In [4]: import pandas as pd
        import numpy as np
        import subprocess

        # active task flow process in EPM Maestro
        subprocess.call('C:\Oracle\Maestro\Client\EPMMaestro.exe D:\Taskflows\extract_account_metadata.ssf TEST "" TEST username enc:pass

        #Load extracted filed to data frame
        path = r'D:\d_account.app'
        app = []
        with open (path, "r") as r:
            app = r.readlines()

        df = pd.DataFrame(app, columns=['content'])
        df.content = df.content.str.replace('\n', '')
        df
```

Out[4]:

	content
0	IFILE_FORMAT=11.12
1	IVERSION=11.2.18
2	ICUSTOM_ORDER=Cust1;Cust2;Cust3;Cust4;Cust5
3	
4	IMEMBERS=Account
...	...
4502	S253000;2530600

Figure 18. Data extraction in python code.

The second stage in the processing of Python code, after data extraction, is involved in data transformation. As mentioned earlier in Figure 17, data transformation for metadata was processed in two parts: part 1 for processing account hierarchy and part 2 for account description.

Part 1 is a complicated process that can be divided into three steps, including cleansing the original content from extracted metadata, converting the cleansed data into a hierarchy path, and transforming it into a dimensional table format. These steps are shown in Figure 19, which depicts the cleansing process; Figure 20, which shows the conversion of the cleansed data into a hierarchy path; and Figure 21, which illustrates the transformation of the hierarchy path into a dimensional table format in data warehouse."

```
In [15]: # cleansing the original content from extracted metadata

         line_from = df[df['content'] == '!MEMBERS=Account'].index[0]
         line_to = df[df['content'] == '!HIERARCHIES=Account'].index[0]

         df_content = df[line_from + 1:line_to]
         df_content = df_content.join(df_content['content'].str.split(";",expand=True))

         line = df[df['content'] == '!HIERARCHIES=Account'].index[0]
         df = df.drop(df.index[0:line+2])
```

```
In [17]: #split the original content into two columns: parent and child
         df1 = df.join(df['content'].str.split(";",expand=True))
         df2 = df1.rename({ 0:'parent', 1:'child'}, axis=1)

         df2.tail()
```

Out[17]:

	content	parent	child
4142	S253000;2530600	S253000	2530600
4143	S253000;C2530600	S253000	C2530600
4144	S253000;2550000	S253000	2550000

Figure 19. Cleansing the original content from extracted metadata.

As Seen in Figure 19, the process is involved in processing account hierarchy. Initially, unnecessary data is cleansed and deleted, leaving only the relevant hierarchy information. Next, the data is split into two columns: one for the parent account and another for the child account.

```
In [13]: #Converting cleansed data into a hierarchy path
def hierarchypath(df, ref):
    arr_rt=[]
    result = []
    row = df[df.child ==ref]
    while len(row)>0:
        arr_rt.append(row.child.iloc[0])
        result.append(("/".join(arr_rt), row.parent.iloc[0]))
        row = df[df.child == row.parent.iloc[0] ]

    rel = pd.DataFrame(result, columns=["Result", "parent"])
    last = rel['Result'].iloc[-1]
    return str(last)

df2['hierarchy'] = df2['child'].apply(lambda x: hierarchypath(df2, x))
```

```
In [14]: df2[df2['parent']== 'S253000']
```

Out[14]:

	content	parent	child	hierarchy
4051	S253000;2530000	S253000	2530000	2530000/S253000/S250001/S250002/S259300/S27000...
4052	S253000;2530500	S253000	2530500	2530500/S253000/S250001/S250002/S259300/S27000...
4053	S253000;2530600	S253000	2530600	2530600/S253000/S250001/S250002/S259300/S27000...

Figure 20. Converting cleansed data into a hierarchical path.

As seen in Figure 20, a function was defined to convert data into a hierarchy, which is presented in the column “hierarchy” in the output.

```
#Transformation of the hierarchy path into a dimensional table format

import numpy as np

df2['parent1'] = df2['parent'].shift(-1)
df2['flag'] = np.where((df2['child'] == df2['parent1']),'no','yes')

df3 = df2[['hierarchy','flag', 'child']]
df3 = df3[df3['flag']== 'yes']
df3 = df3.join(df3['hierarchy'].str.split("/",expand=True))
for index, row in df3.iterrows():
    try:
        lv1 = (row['hierarchy'].split('/')[-1])
    except:
        lv1 = ''
    level1.append(lv1)

    try:
        lv2 = (row['hierarchy'].split('/')[-2])
    except:
        lv2 = ''
    level2.append(lv2)

    .......

    try:
        lv13 = (row['hierarchy'].split('/')[-13])
    except:
        lv13 = ''
    level13.append(lv13)

df3["level1"] = level1
.......
df3["level13"] = level13
df3= df3.rename({ 'child':'ID', 'hierarchy': 'Path'}, axis=1)
dfF= df3[['ID','level1', 'level2', 'level3', 'level4', 'level5', 'level6', 'level7', 'level8', 'level9', 'level10','level11',

dfF[dfF['level7'] == 'S253000']
```

	ID	level1	level2	level3	level4	level5	level6	level7	level8	level9	level10	level11	level12	level13	
573	2530000	LIABILITIES	S280001	S270000	S259300	S250002	S250001	S253000	2530000						25300
574	2530500	LIABILITIES	S280001	S270000	S259300	S250002	S250001	S253000	2530500						25305
575	2530600	LIABILITIES	S280001	S270000	S259300	S250002	S250001	S253000	2530600						25306

Figure 21. transforming the hierarchy path into a dimensional table format

As Seen in Figure 21, the hierarchy of account dimension consists of 13 levels, which have been converted to 13 columns from level 1 to level 13 in the output.

Part 2 is involved in processing account description, which was an additional request from stakeholders after the validation phase. The details of the Python code are depicted in Figure 22.

```
# Adding account description to account hierarchy dataset

df_des = df_content[[0,28]]
df_des = df_des.rename({ 0:'level1', 28: 'level1_description'}, axis=1)
df_des.level1 = df_des.level1.str.strip()
df_des.level1_description = df_des.level1_description.str.strip()


df_des.level1_description = df_des.level1_description.str.replace('English=', '')
dfFinal_lv1 = dfFinal.merge(df_des, on='level1', how='left')

df_des = df_des.rename({ 'level1':'level2', 'level1_description': 'level2_description'}, axis=1)
dfFinal_lv2 = dfFinal_lv1.merge(df_des, on='level2', how='left')

df_des = df_des.rename({ 'level2':'level3', 'level2_description': 'level3_description'}, axis=1)
dfFinal_lv3 = dfFinal_lv2.merge(df_des, on='level3', how='left')

df_des = df_des.rename({ 'level3':'level4', 'level3_description': 'level4_description'}, axis=1)
dfFinal_lv4 = dfFinal_lv3.merge(df_des, on='level4', how='left')

df_des = df_des.rename({ 'level4':'level5', 'level4_description': 'level5_description'}, axis=1)
dfFinal_lv5 = dfFinal_lv4.merge(df_des, on='level5', how='left')

df_des = df_des.rename({ 'level5':'level6', 'level5_description': 'level6_description'}, axis=1)
dfFinal_lv6 = dfFinal_lv5.merge(df_des, on='level6', how='left')

.............................

dfFinal_lv12 = dfFinal_lv11.merge(df_des, on='level12', how='left')

df_des = df_des.rename({ 'level12':'level13', 'level12_description': 'level13_description'}, axis=1)
dfFinal_lv13 = dfFinal_lv12.merge(df_des, on='level13', how='left')

df_des = df_des.rename({ 'level13':'ID', 'level13_description': 'account_description'}, axis=1)
dfFinal_f = dfFinal_lv13.merge(df_des, on='ID', how='left')
```

```
dfFinal_f = dfFinal_f[['ID','account_description','level1', 'level1_description', 'level2', 'level2_description', 'level3','level
dfFinal_f[dfFinal_f['level7'] == 'S253000']
```

	ID	account_description	level1	level1_description	level2	level2_description	level3	level3_description	level4	level4_description
316	2530000	Other liabilities, cur,ext.	LIABILITIES	LIABILITIES	S280001	EQUITY AND LIABILITIES	S270000	TOTAL LIABILITIES	S259300	TOTAL CURRENT LIABILITIES
317	2530500	Other liabilities,cur,ass.	LIABILITIES	LIABILITIES	S280001	EQUITY AND LIABILITIES	S270000	TOTAL LIABILITIES	S259300	TOTAL CURRENT LIABILITIES
318	2530600	Other liabilities, non-IB, cur, IG	LIABILITIES	LIABILITIES	S280001	EQUITY AND LIABILITIES	S270000	TOTAL LIABILITIES	S259300	TOTAL CURRENT LIABILITIES

Figure 22. Processing and adding account description to account hierarchy dataset.

As seen in Figure 22, the Python code retrieves account descriptions based on the first command line, which fetches data for two columns: account number (0) and account description (28). The retrieved data is then merged with the hierarchy dataset, resulting in a hierarchical account metadata data frame that includes the account number and account description. This data frame is formatted in the dimensional table format used in the data warehouse.

The third stage of the process is to load data into the dimensional table in the Oracle data warehouse. The transformed data in the output in Figure 22 is finally loaded to the data warehouse as presented in Figure 23.

```
DIALECT = 'oracle'
SQL_DRIVER = 'cx_oracle'

USERNAME = 'username'
PASSWORD = 'pastword'
HOST = 'host'
PORT = 'port
SERVICE = 'sevicename'

ENGINE_PATH_WIN_AUTH = DIALECT + '+' + SQL_DRIVER + '://' + USERNAME + ':' + PASSWORD +'@' + HOST + ':' + str(PORT) + '/?service_

engine = create_engine(ENGINE_PATH_WIN_AUTH, max_identifier_length=128)

df_list = dfFinal_f.values.tolist()
connection = engine.raw_connection()
cursor = connection.cursor()

cursor.executemany("insert into d_account values (:1, :2, :3, :4, :5, :6, :7, :8, :9, :10, :11, :12, :13, :14, :15, :16, :17, :18
connection.commit()
```

Figure 23. Data loading.

Figure 23 illustrates a connection that is opened to the Oracle data warehouse based on the credential information such as host, port, service name, username, and password. Next, the data frame shown in Figure 22 is converted to a list. The data is then inserted into dimensional table d_account.

The final stage is data monitoring. After the data is loaded into the data warehouse, an email is sent to data engineers that include the number of rows of data loaded to the table, as shown in Figure 24. In addition to this email, data engineers can also check the logs from the staging server, as details of the process are also recorded in the log files.

```
#send an email to monitors
upload_date = now.strftime("%m-%d-%Y %H:%M:%S")
index = dfFinal_f.index
number_of_records = len(index)

msg = MIMEMultipart()
msg['From'] = 'server'
msg['To'] = 'monitors'
msg['CC'] = 'monitors'
msg['Subject'] = 'd_account to Oracle Data warehouse ' + str(upload_date)
body = """\
<html>
  <head></head>
  <body>
    <p>Hello, <br><br>
       Here is the summary of d_account data """ + str(upload_date) + """. See the detail below <br><br>

       # the number of records is loaded to d_account table: """ + str(number_of_records) + """ <br>

       <p>Best regards,<br/>
        Thuan<br/></p><br/>
    </p>
  </body>
</html>
"""
msg.attach(MIMEText(body, 'html'))
server = smtplib.SMTP('smtp.xxxx')
server.ehlo()
server.starttls()
server.ehlo()
server.send_message(msg)
server.quit()
```

Figure 24. Data monitoring.

6.4.2 ETL for transactional data in Oracle EPM cloud

The ETL process for Oracle EPM cloud was developed in a Sandbox environment only, since Oracle EPM cloud was being discussed as a promising cloud application that could replace Oracle HFM on-premise. Therefore, this implementation has not yet been applied to the reality. The Python code in this proposal can also be divided into four main stages: data extraction, data transformation, data loading, and data monitoring. However, the functionality of the stages of data transformation and data monitoring is similar to that of Section 6.4.1. Therefore, this section only focuses on presenting the stages of data extraction and data loading as Proposal Elements 1 and 4 in Section 5.2.

Firstly, the stage of data extraction is developed based on Python, FDMEE, and EPM Automate. FDMEE is used to configure the data extraction, which works similarly to EPM Maestro. The only difference is that FDMEE is a standard functionality in the Oracle EPM cloud. The configuration process in FDMEE includes many steps, so the thesis excludes that information. More information about FDMEE configuration can be found in Marco Rossodivita's blog (2017).

EPM Automate is used to connect the Oracle EPM cloud and execute FDMEE tasks. Once the tasks are completed, the exported file is downloaded to the staging machine.

In [6]:
```
import pandas as pd
import numpy as np
import subprocess

# connecting to Oracle EPM cloud to run FDMEE task and dowload the extracted file to staging machine
commands = '''
call epmautomate login username password https://xxx.ap1.oraclecloud.com
call epmautomate rundatarule EXTRACT_PL %fromperiod% %toperiod% REPLACE STORE_DATA
call epmautomate downloadfile EXTRACT_PL.csv D:\FDMEE\new\test\EXTRACT_PL.csv
'''
subprocess.call(commands, capture_output=True, shell=True)

#Loading the extracted filed to data frame
DF_PL = pd.read_table(r'D:\FDMEE\new\test\EXTRACT_PL.csv', delimiter=';',
                      names=['scenario', 'year', 'period', 'view', 'entity','value', 'account', 'ICP', 'cust1',
                             'cust2','cust3', 'cust4', 'cust5','amount'], encoding="utf-16")

DF_PL
```

Out[6]:

	scenario	year	period	view	entity	value	account	ICP	cust1	cust2	cust3	cust4	cust5	amount
0	ACTUAL	2023	Jan	Periodic	CORP	<Entity Curr Total>	S253000	[ICP Top]	TEST	Non	99	Non	Non	1000
1	ACTUAL	2023	Jan	Periodic	CORP	<Entity Curr Total>	S253000	[ICP Top]	TEST	Non	99	Non	Non	1000
2	ACTUAL	2023	Jan	Periodic	CORP	<Entity Curr Total>	S253000	[ICP Top]	TEST	Non	0	Non	Non	10000
3	ACTUAL	2023	Jan	Periodic	CORP	<Entity Curr Total>	S253000	[ICP Top]	TEST	Non	0	Non	Non	1000
4	ACTUAL	2023	Jan	Periodic	CORP	<Entity Curr Total>	S253000	[ICP Top]	TEST	Non	1	Non	Non	1000

Figure 25. Extracting transactional data from Oracle EPM cloud.

As seen in Figure 25, Python is used to manage the end-to-end process. First, EPM Automate is activated to connect to the EPM cloud, then a data rule is run from FMDEE, and the file is downloaded to the staging area. The extracted is then loaded into a data frame for preparing the data transformation stage.

As mentioned earlier, this section is not presenting the data transformation in detail since these steps are similar to those in Section 6.4.1, where the process also involves cleansing and standardizing the data. This includes several steps for deleting the unnecessary data as shown in Figure 26.

```
df_pl['value'] = df_pl['value'].astype(str).str.replace("<", "",regex=True)
df_pl['value'] = df_pl['value'].astype(str).str.replace(">", "", regex=True)
df_pl['ICP'] = df_pl['ICP'].astype(str).str.replace("[", "",regex=True)
df_pl['ICP'] = df_pl['ICP'].astype(str).str.replace("]", "", regex=True)
df_pl['cust1'] = df_pl['cust1'].astype(str).str.replace("[", "", regex=True)
df_pl['cust1'] = df_pl['cust1'].astype(str).str.replace("]", "", regex=True)
df_pl['cust2'] = df_pl['cust2'].astype(str).str.replace("[", "", regex=True)
df_pl['cust2'] = df_pl['cust2'].astype(str).str.replace("]", "", regex=True)
df_pl['cust3'] = df_pl['cust3'].astype(str).str.replace("[", "", regex=True)
df_pl['cust3'] = df_pl['cust3'].astype(str).str.replace("]", "", regex=True)
df_pl['cust5'] = df_pl['cust5'].astype(str).str.replace("[", "", regex=True)
df_pl['cust5'] = df_pl['cust5'].astype(str).str.replace("]", "", regex=True)


df_pl['id'] = (df_pl['scenario'].map(str) + df_pl['year'].map(str) + df_pl['period'] + df_pl['view'] + df_pl['entity'] + df_pl['v
               + df_pl['cust2'] + df_pl['cust3'] + df_pl['custom4'] + df_pl['cust5'])
now = datetime.now()
df_pl['datetime'] = now.strftime("%m-%d-%Y %H:%M:%S")
df_pl['uploaded_date'] = df_pl['datetime'].map(str)

df_pl = df_pl[['id', 'scenario', 'year', 'period', 'view', 'entity','value', 'account', 'ICP', 'cust1',
```

Figure 26. Data transformation for transactional data.

The final stage of this ETL is loading the data into the cloud environment AWS S3. Uploading the data to S3 requires a few configurations in S3 such as creating the bucket name and the source system. These configurations were done by the data platform team, and this thesis only concentrated on the Python scripts in the way to load the data to S3.

```
#Loading the file to AWS S3
aws_access_key_id = "access key id"
aws_secret_access_key = "secret key"
aws_bucket_name="bucket name"
source_system='source system'
table='table'

if len(df_pl) > 0:
    s3_conn = S3Conn(aws_access_key_id,
                             aws_secret_access_key,
                             aws_bucket_name)

    s3_conn.upload_to_s3(df_pl, source_system, table)
```

Figure 27. Loading transactional data to AWS S3.

Figure 27 shows the process of loading data to AWS S3. As seen in the script, several variables need to be defined, such as the access key ID, secret key, bucket name, source system, and table. Additionally, the S3Conn class is defined earlier in the script, which includes a function to upload data to S3.

6.4.3 ETL for Oracle ERP cloud

The development was built using the web service configured in Oracle BI Publisher and PL/SQL in the staging database. As shown in the green highlight in Figure 15, it proposed adding two parameters (“from row number”, and “to row number”) to SQL queries in the web services BI Publisher and then updating the PL/SQL code in the staging area. The parameters helped to limit the size of the output file to less than 10KB.

```
    FND_FLEX_VALUES_VL FFVV,
    GL_LEDGERS GL
  WHERE
    1 = 1
    AND FIFS.ID_FLEX_CODE = 'GL#'
    AND FIFS.APPLICATION_COLUMN_NAME = 'SEGMENT9'
    AND FIFS.FLEX_VALUE_SET_ID = FFVV.FLEX_VALUE_SET_ID
    AND FIFS.ID_FLEX_NUM = GL.CHART_OF_ACCOUNTS_ID
    AND GL.LEDGER_ID = :PR_LEDGER_ID
  ) SEGMENT9
WHERE
  1 = 1
  AND TRAN.SEGMENT1 = SEGMENT1.FLEX_VALUE (+)
  AND TRAN.SEGMENT2 = SEGMENT2.FLEX_VALUE (+)
  AND TRAN.SEGMENT3 = SEGMENT3.FLEX_VALUE (+)
  AND TRAN.SEGMENT4 = SEGMENT4.FLEX_VALUE (+)
  AND TRAN.SEGMENT5 = SEGMENT5.FLEX_VALUE (+)
  AND TRAN.SEGMENT6 = SEGMENT6.FLEX_VALUE (+)
  AND TRAN.SEGMENT7 = SEGMENT7.FLEX_VALUE (+)
  AND TRAN.SEGMENT8 = SEGMENT8.FLEX_VALUE (+)
  AND TRAN.SEGMENT9 = SEGMENT9.FLEX_VALUE (+)
  AND (UPPER(TRAN.STATUS) IN (:PR_STATUS) OR UPPER(TRAN.STATUS) IS NULL)
  AND ROWNUM >= :P_FROM_ROW_NUMBER AND ROWNUM <= :P_TO_ROW_NUMBER
GROUP BY
  UPPER(TRAN.STATUS),
  TRAN.MODULE,
  TRAN.MODULE_TYPE,
  TRAN.GACCOUNT,
  TRAN.ACCOUNTING_CLASS_CODE,
  TRAN.EVENT_TYPE_CODE,
  TRAN.TRANSACTION_NUMBER,
  TRAN.CREATED_BY,
  TRAN.CURRENCY_CODE,
```

Figure 28. Adding the parameters to SQL queries in Oracle BI Publisher.

As seen in the highlighted portion in Figure 28, two parameters named “P_FROM_ROW_NUMBER” and “P_TO_ROW_NUMBER” were added to the SQL query that was configured for data extraction in Oracle BI Publisher.

After adding parameters to the web service, an iteration to invoke the web service was added to PL/SQL procedure in the staging database. The amount of data was observed to be quite large, but never greater than 1.5 million rows. To handle this amount of data, the retrieve process was divided into five batches, each fetching a maximum of 300000 rows from the web service to make sure the output file is less than 10 KB.

```
DECLARE
  ERRBUF  VARCHAR2(240);
  RETCODE  NUMBER;
  PR_BATCH NUMBER;
  PR_ESS_PATH_REPORT  VARCHAR2(240) := 'BI Publisher/Reports/webservice name.xdo';
  PR_TABLE VARCHAR2(240) := 'stagingtable';
  P_FROM_ROW_NUMBER NUMBER;
  P_TO_ROW_NUMBER NUMBER;
BEGIN
  EXECUTE IMMEDIATE 'TRUNCATE TABLE stagingtable';
  PR_BATCH = 300000;
  P_FROM_ROW_NUMBER = 0;
  P_TO_ROW_NUMBER = PR_BATCH;
  FOR I IN 1 .. 5 LOOP
      BI.INVOKE_BI_PUBLISHER(ERRBUF, RETCODE, PR_ESS_PATH_REPORT, PR_TABLE, P_FROM_ROW_NUMBER, P_TO_ROW_NUMBER, #BI.pr_from_date,#BI.pr_to_date);
      if (ERRBUF = 'Error') then
        Raise_Application_Error (-20343, 'Please check table log BI_LOG for more details');
       end if;
      P_FROM_ROW_NUMBER = P_FROM_ROW_NUMBER + PR_BATCH;
      P_TO_ROW_NUMBER = P_TO_ROW_NUMBER + PR_BATCH;

    IF RETCODE = 0 THEN
        EXIT;
    END IF;

  END LOOP
END;
```

Figure 29. Invoke BI Publisher web service from PL/SQL.

As seen in Figure 29, a function that invokes the web service was defined beforehand. To divide the invoking process into five batches, an iteration of the web service invocation was added, and each time the parameter was increased by 300000 until the end of the loop.

7 Conclusions

This section provides a summary of the research conducted, the proposal, the thesis evaluation and the conclusion.

7.1 Executive Summary

This thesis focuses on improving the ETL processes for Oracle HFM and ERP cloud to enhance the data quality, processing efficiency, and cloud combability. This study identifies weaknesses in the existing systems and stakeholders'

needs, aiming to highlight the ineffective stages and components of the current processes and propose solutions to address these weaknesses.

The thesis was conducted using applied research, which employed both quantitative and qualitative methods to collect and analyse data. Data collection took place in three rounds as outlined in the research design. The analysis of the current stage was a significant component of the thesis, as it involved identifying the ineffective components and the stages of the existing ETL processes. As a result, the weaknesses were identified, including issues with connecting to cloud environments, a lack of data transformation and metadata extraction due to the absence of data processing libraries in CMD script, and limitation of data output from the web services in the Oracle ERP cloud.

To address the listed challenges, the main components of the ETL process were discussed in the literature review. Based on the existing knowledge and company documentation, the conceptual framework was created, which included the main stages and components of the ETL process such as the connectivity to other applications, data extraction, data transformation, data loading, data monitoring, and data scheduling. The thesis focused on the problems with the current ETL process for Oracle HFM and ERP cloud and proposed key components to improve it.

The proposal in the thesis was based on the conceptual framework, internal documents, and discussions with the stakeholders, and it comprises five elements.

The first of the proposal involves presenting a method for connecting to the data source Oracle HFM cloud using Python, FDMEE, and EPM Automate.

The second element describes a new data transformation approach using powerful data processing libraries in Python such as Pandas, Numpy, Re, String, and NLTK to delete and clean up siloed data.

The third element proposes the new ETL process for metadata to improve the data modelling for financial data analysis and to create financial drill-down reports.

The fourth element involves the discussion of the connectivity to the data warehouse such as Oracle data warehouse and AWS S3, for which Python libraries including cx_Oracle, and sqlalchemy, and boto3 are proposed.

Finally, to improve the ETL for the Oracle ERP cloud, the proposal suggests modifying the web services in Oracle BI Publisher and the PL/SQL procedure in the staging area.

The proposal was validated and tested by stakeholders, and more than 80 percent of the proposed changes were implemented in production, resulting in improved data quality and mitigation of the weaknesses in financial reporting.

7.2 Thesis Evaluation

The study was action research focused on the technical issues, and the challenges and objectives were defined at the beginning. The data was collected and analysed through meetings, internal documents, interviews, workshops, and observation using both qualitative and quantitative methods. From a validity standpoint, the outcomes of the thesis align with the objectives, and the proposed solutions were implemented to address the identified challenges.

From a reliability perspective, the thesis ensured that all the processes in the study were consistent. Stakeholders participated in the meetings and workshops and were involved in validating and testing the initial proposals. Additionally, stakeholders provided feedback to improve the proposals, and the final proposals and their implementation were developed based on stakeholder acceptance.

The logic of the thesis was also ensured, as the flow of the study aligned with the research design. First, the object of the thesis was defined at the beginning,

followed by a review of existing knowledge and literature. Next, current state analysis was discussed to identify the challenges, and the initial proposals were developed. Finally, the final proposals were developed and implemented.

Although more than 80 percentage of proposed changes were finally successfully implemented in the production environments. However, further improvements are needed for the thesis. For example, the ETL for Oracle EPM cloud is still pending as the business is considering the possibility of migrating the current system, Oracle HFM, to cloud. During the development of the ETL demo, a temporary Oracle EPM cloud was registered for only one month, which did not allow for comprehensive testing of the proposal.

7.3 Final Words

This thesis provides a valuable contribution to the ETL processes in Oracle HFM and Oracle ERP cloud. It demonstrates the importance of effective ETL processes and the potential benefits of adopting ETL processes in the case company, with the proposed changes resulting in improving data processes and reduced processing time. Despite some limitations, the success of the proposals in this study establishes a strong foundation for future research in this area. It recommended that the companies consider improving their ETL processes to enhance their data management capabilities. This thesis is expected to inspire further research in the field of data integration.

References

Codeless. What is ETL? – A Comprehensive Guide to Extract, Transform, and Load. Available from: https://www.codelessplatforms.com/blog/what-is-etl/. (Accessed 28 April 2023).

Astera. Break Down Data Silos and Unlock Trapped Data with ETL. Available from: https://www.astera.com/wp-content/uploads/2020/05/EBook-Break-Down-Data-Silos-with-ETL.pdf. (Accessed 28 April 2023).

Informatica. What is ETL (extract transform load)? Available from: https://www.informatica.com/resources/articles/what-is-etl.html. (Accessed 12 January 2023).

Sesia S., Toufik I., Baker M., LTE: The UMTS Long Term Evolution: From Theory to Practice, Second Edition. John Wiley & Sons. 2009.

Coughlan, P, & Coghlan, D (2002). Action research for operations management. Available from: https://www.researchgate.net/publication/235278074_Action_Research_for_Operations_Management. (Accessed 12 October 2022).

Bhandari, P (2020). What Is Qualitative Research? | Methods & Examples. Available from: https://www.scribbr.com/methodology/qualitative-research/. (Accessed 12 October 2022).

Cooper, D.R. & Schindler, P.S. 2012. Business research method. 12th. New York, McGraw-Hill Higher Education. Available from: http://www.mim.ac.mw/books/Donald%20R%20Cooper's%20Business%20Research%20Methods,%2012th%20Edition.pdf. (Accessed 12 October 2022).

Altexsoft (2021). Data Engineering and Its Main Concepts: Explaining the Data Pipeline, Data Warehouse, and Data Engineer Role. Available from: https://www.altexsoft.com/blog/datascience/what-is-data-engineering-explaining-data-pipeline-data-warehouse-and-data-engineer-role/. (Accessed 12 January 2023).

Javatpoint. (2022). Three-Tier Data Warehouse Architecture. Available from: https://www.javatpoint.com/three-tier-data-warehouse-architecture. (Accessed 12 January 2023).

Priya, P. Oracle Data Warehousing. Available from: https://www.educba.com/oracle-data-warehousing/. (Accessed 20 January 2023).

Amazon. What is Amazon S3? Available from: https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html. (Accessed 22 January 2023).

Amazon. What is Amazon Redshift? Available from: https://docs.aws.amazon.com/redshift/latest/mgmt/welcome.html. (Accessed 22 January 2023).

Kevin, G. What is Amazon Redshift? Available from: https://www.sumologic.com/blog/what-is-amazon-redshift/. (Accessed 22 January 2023).

IBM. ETL (Extract, Transform, Load). Available from: https://www.ibm.com/topics/etl. (Accessed 30 January 2023).

IBM. What is OLAP. Available from: https://www.ibm.com/topics/olap. (Accessed 30 April 2023).

IGI Global. What is Data Source. Available from: https://www.igi-global.com/dictionary/data-sources/87402. (Accessed 30 January 2023)

Wikipedia (2023). Oracle Cloud Enterprise Resource Planning. Available from: https://en.wikipedia.org/wiki/Oracle_Cloud_Enterprise_Resource_Planning. (Accessed 30 January 2023).

Concentric Solution (2023). Oracle Hyperion Financial Management (HFM). Available from: https://www.concentricsolutions.com/portfolio/technology-partners/oracle/oracle-hyperion-financial-management-hfm#:~:text=Oracle%20Hyperion%20Financial%20Management%20(HFM)%20is%20a%20comprehensive%2C%20web,%2C%20highly%2Dscalable%20software%20solution. (Accessed 30 January 2023).

IBM (2023). What is the data modelling. Available from: https://www.ibm.com/topics/data-modeling#:~:text=Data%20modeling%20is%20the%20process,between%20data%20points%20and%20structures. (Accessed 02 February 2023).

Craig, S. Data modelling. Available from: https://www.techtarget.com/searchdatamanagement/definition/data-modeling. (Accessed 30 January 2023).

EPM Maestro (2023). EPM Maestro suite. Available from: https://epmmaestro.com/#:~:text=A%20generic%20interface%20to%20any,experience%20of%20your%20HFM%20system. (Accessed 05 February 2023).

GeeksforGeeks (2023). Data Structures. Available from: https://www.geeksforgeeks.org/data-structures/. (Accessed 10 February 2023).

Oracle (2023). Fusion Middleware Developer's Guide for Oracle Business Intelligence Publisher. Available from: https://docs.oracle.com/cd/E28280_01/bi.1111/e22259/webservices.htm#BIPDV002. (Accessed 13 February 2023).

Marco, R. (2017). How to Extract Data from Oracle Planning Budget Cloud (part two). Available from: https://blogs.perficient.com/2017/01/23/how-to-extract-data-from-oracle-planning-budget-cloud-part-two/V002. (Accessed 17 February 2023).

Rohit, S. (2022). Top 6 Data Science Programming Languages 2023. Available from: https://www.upgrad.com/blog/data-science-programming-languages/#:~:text=Python%20is%20the%20most%20widely,language%20is%20inherently%20object%2Doriented. (Accessed 30 January 2023)