Data Preparation with Oracle Cloud

Cristiana COSTAN
Bucharest University of Economic Studies
Faculty of Cybernetics, Statistics and Economic Informatics
Bucharest, Romania
costancristiana21@stud.ase.ro

This article examines the Extract-Transform-Load (ETL) process as an essential method for preparing data from warehouses for analysis. It also covers the difficulties of combining data from numerous external sources, converting it for consistency, and storing it in the cloud for effective querying. Oracle Cloud Infrastructure (OCI) provides a range of services for ETL workflows, including Oracle Data Integrator (ODI), which offers a variety of code-based data processing options, and Data Transforms, also known as ODI Web, a modern no-code solution designed to simplify and automate data transformations and machine learning tasks. A practical example using a League of Legends dataset illustrates how Data Transforms and Autonomous Data Warehouse can be used for effective data preparation and analysis. The goal of this paper is to assist data scientists in simplifying and improving data workflows through the use of modern cloud ETL solutions.
Keywords: *Data Warehouse, ETL, Oracle Cloud, Data Integration, Machine Learning*

1 Introduction

Although the term itself had not yet been formally established, individuals have employed ETL-like processes since the early days of database systems. To extract data and make it suitable for analysis, it was necessary to process it into a format more appropriate for analytical tasks. These transformations can vary, ranging from duplicate removal and missing value replacement to the application of functions for generating new statistics. This process is frequently employed in data warehouses, which usually handle substantial amounts of data that are organized along several dimensions, the most significant of which is the temporal component.

Such data is usually sourced from a wide range of external systems such as APIs and must be cleaned and transformed before it can be integrated into databases for analytical use. Therefore, ETL or its more recent variant ELT are employed for extracting, transforming, and uploading data into various storage systems, or for extracting and loading data into the data warehouse and performing transformations directly within it.

OCI provides cloud-based services that facilitate both ETL and ELT workflows. Oracle Data Integrator, also known as ODI, offers extensive options for data processing. Data Transforms, another service from the OCI stack built on the ODI infrastructure, is a no-code solution capable of executing these processes as well as additional features such as machine learning model generation and graphical outputs. If needed, the code required for the transformations designed in the Data Transforms flow can be automatically generated at runtime. This represents a modern cloud-native approach that enables data warehouse processing without the need to write code manually.

2 Data Warehousing

"A data warehouse is a subject-oriented, integrated, nonvolatile, and time-variant collection of data in support of management's decisions." [1]. Depending on the specific needs of the business maintaining it, data warehouses can be utilized for storing and processing granular data for a wide range of analytical and strategic purposes.

The most important characteristic of the data warehouse is its integration. Multi-source data undergoes transformation processes in

order to become a unified corporate data representation. As the data is being loaded into the warehouse, these processes typically involve data conversion and summarization in order to guarantee consistency among datasets.

The variety of data sources and the particular needs of organizations make each implementation and application of data warehouses different. In general, data is loaded into the warehouse in bulk. Another important characteristic of data warehouses is time variance: each record is associated with a temporal element, such as a timestamp or transaction date, facilitating trend monitoring and historical analysis.

Key characteristics of a data warehouse include the structuring of data to facilitate ease of access and high-speed querying, the inclusion of large volumes of historical data, and the loading of data from multiple sources involving various transformations [2]. Additionally, data warehouse's architecture can be customized for different groups within the organization by adding data marts, which are systems designed for a particular line of business. These can be utilized to analyze historical data or make predictions for certain branches separately.

Data warehouses also support on-line analytical processing (OLAP), an essential element in decision support [3]. To enable complex analytics and visualization, data within the warehouse is typically organized in a multidimensional format, with time often being one of the primary dimensions. These dimensions commonly exhibit hierarchical structures, allowing users to explore data at different levels of granularity. Rollup (increasing aggregation level), drill-down (decreasing aggregation level or increasing detail), slice-and-dice (selection and projection), and pivot (altering the multidimensional perception of data) are examples of common OLAP procedures along one or more dimension hierarchies.

3 Overview of ETL and ELT

Originating in the early days of database systems, the Extract-Transform-Load (ETL) process was initially thought of as a programming task without a specific name or understanding of its importance [4]. Although ETL activities existed before the 2000s, it was only around that time that ETL began to be clearly defined and mentioned in well-known books. Any data processing software that modifies or filters records, performs calculations, and loads data into a different storage system than the original one can be considered a form of ETL.

ETL involves three sequential stages [5]. Data is first extracted from various sources, commonly in formats such as comma-separated values (CSV), Excel (XLS), text files (.txt), JSON, or retrieved through application programming interfaces (APIs). In the transformation stage, the extracted data undergoes a series of operations including data cleaning, normalization, duplicate removal, filtering, sorting, grouping, and the application of various functions if needed. After being transformed, the data is then imported into a specific system, such as a database, data warehouse, or data mart, in order to optimize further analytical processing.

Extract-Load-Transform (ELT) represents an alternative data-processing approach for analytical purposes, where data is initially loaded in its raw form and transformed directly within the target system [6]. Both ETL and ELT contain the same fundamental steps; however, the key difference lies within the sequence and location of the transformation step. In ETL, raw data is processed before being loaded into the target database, whereas in ELT, data is loaded directly into the data warehouse and transformed afterwards. Consequently, ETL involves moving data only once it has been fully transformed, while ELT permits multiple transformations on the data after it has been loaded.

4 Data Preparation Techniques for Machine Learning

Analysts often encounter various requirements depending on the characteristics of the data, since each dataset used to create machine learning models is naturally distinctive [7]. Regardless of the data source, the data preparation process includes general operations that are often required in addition to these specific requirements. These procedures typically include data cleaning, feature selection, data transformation, feature engineering, and dimensionality reduction.

Data cleaning, also known as data cleansing, is the process of correcting or removing inaccuracies, inconsistencies, or incomplete values from a dataset. This can consist of removing outliers, identifying and eliminating duplicates, or replacing missing values using statistical methods such as the mean, predicted values generated by models, or by utilizing values that significantly exceed a specific range.

Feature selection requires identifying the most relevant elements that influence the prediction of the target variable. This phase is essential for simplifying the model, reducing overfitting, and enhancing interpretability.

Data transformation handles variable type conversion and data scaling. Variables may be numeric (such as integers or floating-point numbers) or categorical (such as nominal, ordinal, or Boolean types). These transformations include the following conversions: numeric variables into ordinal ones ("Discretization Transform"), categorical values into integers ("Ordinal Transform"), or categorical variables into binary vectors ("One Hot Transform"). This step may also include normalization, which rescales the data to a range between 0 and 1, and standardization, which modifies values to follow a normal distribution.

Feature engineering is the process in which new variables are created from already existing data for enhancing the performance of machine learning models. Combining characteristics, computing statistics from variables, or creating dummy variables are some of the methods utilized to generate significant attributes. During this step, complex variables can be simplified by division into smaller parts or already existing variables can be extended.

Dimensionality reduction serves as an alternative to feature selection, lowering the quantity of input features while maintaining the data's fundamental structure and variance. Commonly employed methods for achieving this include Principal Component Analysis (PCA) and Singular Value Decomposition (SVD).

Another critical concern during data preparation is data leakage. This occurs whenever the preparation techniques are applied to the entire dataset prior to splitting the dataset into training and testing subsets. As a result, data from the test set can unintentionally influence the training process. For example, when calculating the mean for substituting missing values, the train set can include a mean calculated based on both sets if the splitting does not occur before this operation. Thereby, performance results may be overestimated since the model caught a part of the test data when training. Data preparation procedures should be applied independently to the training and testing data after the initial split in order to prevent data leakage.

5 Transforming Data with Oracle Cloud

Oracle Cloud Infrastructure (OCI) is a suite of cloud services for running secure, highly available applications with flexible storage [8]. For cloud-native development, OCI includes services such as API management, machine learning, serverless functions, autonomous and MySQL databases, as well as deployment, monitoring, and observability tools [9].

OCI's autonomous databases provide integrated data warehousing and analytics capabilities that support machine learning workflows [10]. Both data warehouses and data lakes are accessible through OCI's data warehouse services; the latter facilitates data

processing and analytics by incorporating built-in artificial intelligence, machine learning, graph, spatial, and other advanced features [11]. Data scientists can efficiently design and implement machine learning models within Oracle Autonomous Data Warehouse through the use of Oracle Machine Learning, which provides Python and SQL notebooks along with an AutoML user interface.

While Oracle Machine Learning is a powerful tool for in-database modeling, this article focuses on the data transformation and preparation stages within Oracle Cloud. The cleaned and transformed data will be accessed directly from the Autonomous Data Warehouse through Python for external machine learning analysis.

OCI Data Integration is primarily used by administrators, data engineers, ETL developers, and operators [12]. Since this article focuses on data transformation for machine learning, data engineers and ETL developers, who are responsible for developing, building, and testing data integration solutions, are of primary importance. The first step in the process involves data extraction from a data entity, such as a database table, view, or file. After that, various operators can be utilized to represent input sources, output targets, or transformations within the data flow. The pipeline is defined as a task, which can be executed to extract, transform, and load data into the database [13]. This flow of data between source and target datasets, including any operations performed, is defined by Data Flows, which are a component of Data Integration [14].

In Data Integration, the first step in designing a data flow involves selecting appropriate data operators, such as the source operator, which provides the input data, and the target operator, which defines the destination for the transformed data [15]. In order to further modify data as it passes through the pipeline, shaping operators are additionally utilized. These operators facilitate a variety of data processing tasks, such as filtering, joining, the use of expressions and aggregations, returning distinct values, and sorting. They also enable relational operations such as UNION, MINUS, and INTERSECT, along with additional transformations such as split, pivot, lookup, flatten, and the use of table function operators. The flatten operator is utilized to unnest hierarchical data structures in formats like JSON, multi-line JSON, Avro, and Parquet, with the ability to unnest data in the form of an Array. In order to build expressions, functions can be utilized with operators in these data flows; these functions are aggregate functions (COUNT, MAX, MIN, SUM, AVG, LISTAGG), analytic functions (FIRST_VALUE, LAG, LAST_VALUE, LEAD, RANK() OVER, ROW_NUMBER() OVER), arithmetic functions (ABS, CEIL, FLOOR, MOD, POWER, ROUND, TRUNC, TO_NUMBER), array functions, conditional functions, date and time functions, hash functions, hierarchical functions (SCHEMA_OF_JSON, FROM_JSON, TO_JSON, TO_MAP, TO_STRUCT, TO_ARRAY), higher-order functions, operator (comparison) functions, string functions and unique ID functions. Table function operators also provide many operations employed in data processing such as deduplication, N/A drop, N/A fill, N/A replace, and others. Attributes can also be processed in data flows through transformations, the most relevant ones being changing data types and filling up nulls.

Moreover, OCI supports the building, training, and management of machine learning models via the Data Science service, a fully managed, serverless platform [16]. Equipped with Python-centric tools, libraries, and packages, data scientists can easily acquire, profile, prepare and visualize data, perform feature engineering, train models, and evaluate, explain and interpret their results. Functions, Data Flow, Autonomous Data Warehouse (ADW), and Object Storage are among the other OCI stack components that this service integrates with. Data Science is accessible through the Console, REST API,

Software Development Kits (SDKs), or Command Line Interface (CLI). Model creation and saving can be performed using Accelerated Data Science (ADS) SDK, OCI Python SDK, or the Console, with programmatic creation and saving being the recommended method [17].

Data preparation and machine learning workflows can also be developed using Oracle Data Transforms, a graphical, no-code interface that enables the design of data loads, data flows, and workflows within Oracle Cloud [18]. Mappings created with Data Transforms for loading data into Autonomous Databases can be executed immediately or scheduled for later deployment. Upon execution, Data Transforms automatically generates the necessary underlying code. In the Data Flow editor, the available actions include Data Transform, Data Preparation, Machine Learning, Text, and Oracle Spatial [19]. Oracle Data Transforms supports both ETL and ELT processes [20].

When selecting an appropriate data integration tool, it is worth noting that Data Transforms represents the latest evolution of Oracle Data Integrator (ODI), being built upon its powerful engine [20, 21]. While Oracle Data Transforms shares many foundational concepts with ODI, it utilizes different terminology [22], and is sometimes informally referred to as ODI Web in the Oracle environment. Designed as a cloud-native data integration solution with a modern user interface, it prioritizes ease of use and rapid accessibility.

6 ETL Implementation in Oracle Cloud

For the practical part, I utilized a dataset containing over 50,000 ranked matches from the Europe West (EUW) server of the online multiplayer game League of Legends. The dataset is available on Kaggle [23] and includes three accompanying JSON files used for mapping champion and summoner spell IDs to their corresponding names. The data was collected via the official Riot Games API [24], and it contains historical ranked match information.

My approach involves uploading the files to the Oracle Cloud ADW, executing the necessary data processing, and loading the processed data into a database table located in a different schema within the created warehouse for further analysis.

Using Oracle Data Transforms, I uploaded the required files as objects to Oracle Cloud through the “Load Data” option, which imported them into an ADW instance previously configured for this project. The ADW was organized into two schemas: ETL_SOURCE, designated for the source files, and ETL_OUTPUT, intended for the creation of the target table. The JSON files were loaded from my computer into separate tables named CHAMPION_INFO, CHAMPION_INFO_2, and SUMMONER_SPELL_INFO, with their contents stored as CLOB data types. Because the JSON files contained hierarchical data, I used SQL Developer to flatten them by parsing and keeping the information that I considered relevant for this project. The data from the tables corresponding to the JSON files with champion information was combined into a single table to have the required champion data in one place. After uploading the CSV file, I checked the option to convert invalid numeric values to null and then the file was loaded into a table named GAMES. All of these tables were placed within the ETL_SOURCE schema. A corresponding target table, GAME_INFO, was created in the ETL_OUTPUT schema, aligned with the mapping defined in Figure 1. I decided to keep the match IDs, game duration, the winning team, as well as the champions, their primary roles, and the summoner spells used by each player.

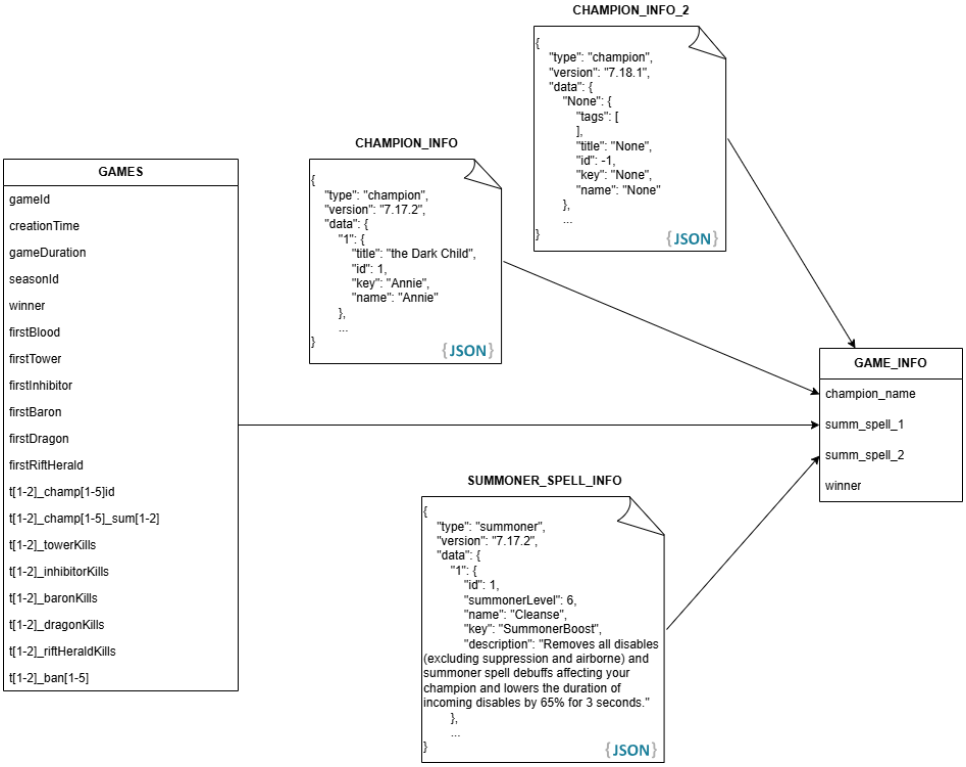

Fig. 1. Data Mapping

Following the data upload, I established two connections via the "Connections" page: ORACLE_SOURCE, linked to the source schema, and ORACLE_OUTPUT, linked to the target schema. I then navigated to the "Data Entities" tab to import the metadata corresponding to the uploaded objects from both schemas. Once the import was done, I created a new project under the "Projects" section and accessed "Data Flows" within it for designing a pipeline to integrate data obtained from the source objects into the target GAME_INFO table, which would then be utilized for analysis.

To eliminate matches where the winner value was equal to 0, which indicates that neither team won and the game ended up as a remake, a "Filter" operation was first used. Next, the "Expression" operator from the "Data Transform" panel was used to perform an unpivot operation, transforming the dataset by expanding values from multiple columns into multiple records, each row ending up with match data for only one player. Specifically, each row in the original dataset represented ten champions per match. Through the use of expressions, I extracted the game ID, winning team, champion name, and the two summoner spells associated with each champion.

Data for Team 1 and Team 2 were combined into a single dataset using the "Set" operator with the UNION operation. Before merging the two obtained sets, I transformed the winner column using the DECODE function to create separate binary indicators for each team: for Team 1, a value of 1 indicated a win (originally 1) and 0 indicated a loss (originally 2); for Team 2, a value of 1 indicated a win (originally 2) and 0 indicated a loss (originally 1). This approach made it possible to show each team's win and loss records separately. This transformation was performed prior to unifying the data to simplify the process and avoid complications related to column prefixes such as "T1_" for the first team and "T2_" for the second one. Following this, left outer joins were executed with the CHAMPION_INFO and SUMMONER_SPELL_INFO tables created from the flattened JSON files to retain all relevant data. Since the SUMMON-

ER_SPELL_INFO table contained two columns representing summoner spells, two join operations were necessary. Finally, the transformed data was loaded into the target table GAME_INFO, with the column mapping illustrated in Figure 2.

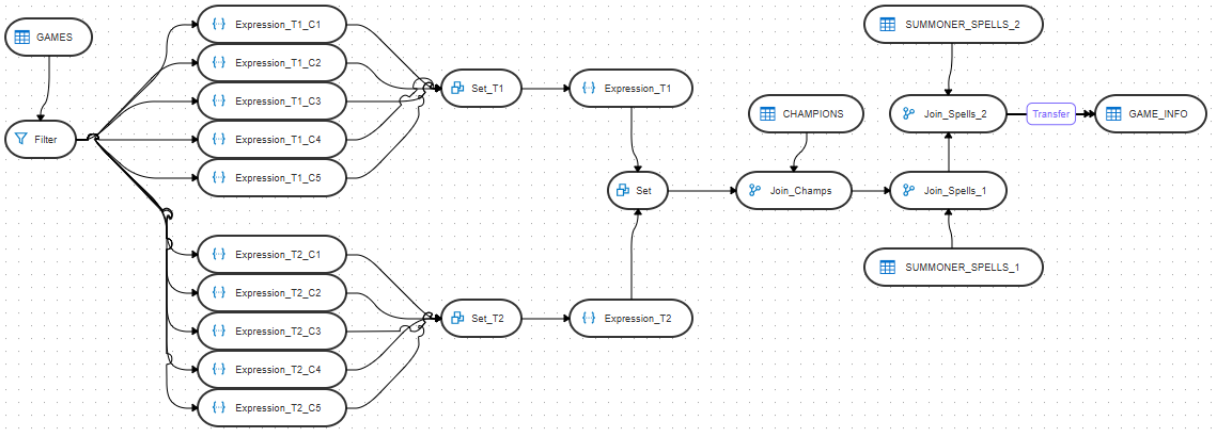

Fig. 2. ETL Workflow

This process resulted in a dataset comprising over 500,000 rows. The final structure of the GAME_INFO table is displayed in Figure 3.

Fig. 3. ETL Output

7 Integrating Data with Machine Learning

To highlight the importance of data preparation for machine learning, I created a simple model utilizing the data from the obtained table in order to estimate the probability of winning in League of Legends. Due to having limited access rights on Oracle Cloud, I couldn't fully utilize its tools designed for machine learning. Therefore, I developed a project on my local machine, in which I connected to the Oracle ADW and performed the necessary operations programmatically in Python.

For example, instead of creating an expression within the ODI Web to insert and utilize a random parameter for splitting the data, I directly performed an 80-20 split of the dataset in Python, dividing it into training and testing sets. Similarly, rather than using the *data_cleanse* feature from the "Data Preparation" tab, I manually handled missing values by replacing empty strings with "" and filling empty numeric fields with mean values, programmatically replicating the cleaning process displayed in Figure 4.

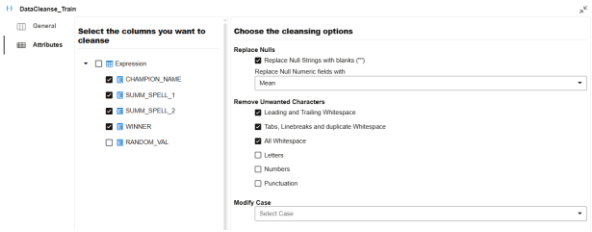

Fig. 4. ODI Web Data Cleanse

Moving on to the modeling stage, I implemented a simple logistic regression to analyze the binary dependent variable WINNER in relation to independent variables representing the champions and summoner spells used in each match, focusing exclusively on pre-game data. As a result, the model produced a low Area Under the Curve (AUC) of 0.52 on the Receiver Operating Characteristic (ROC) curve (Figure 5). According to standard interpretation, an AUC close to 0.5 suggests performance equivalent to random guessing, indicating

that the model had no discriminating ability [25].

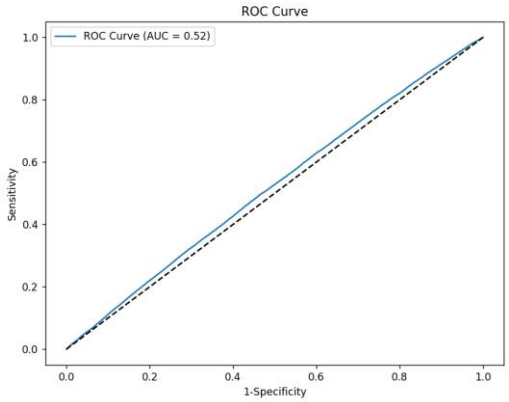

Fig. 5. ROC Curve

The shape of the ROC curve further supports this interpretation, as it leans toward 1-specificity axis, showing a higher rate of false positives [26]. The model incorrectly predicted negative outcomes as positive in several cases. This outcome implies that the selected features are not enough for the model to learn effectively. Consequently, the predictions resembled random classification rather than reflecting significant patterns.

This limitation is not unique; a similar pattern was observed in League of Legends Match Outcome Prediction, where models trained only on pre-game features performed close to random, while models incorporating in-game data achieved significantly better results, as measured by ROC curves [27].

8 Conclusions and Future Work

OCI provides a wide range of options for transforming large datasets from warehouses, and loading them into designated storage systems. For ETL and ELT processes, Oracle provides both ODI and Oracle Data Transforms, also known as ODI Web. The latter is built on the foundation of ODI, using a similar syntax and offering many of the same processing features in a no-code format which can also generate the code required for the defined operations. In Data Transforms, pipelines can be built in the Data Flow interface, making development easier. In addition to the visual workflow, the tool can automatically generate the underlying code once the pipeline is complete. The integration with ADW allows users to manage storage and processing entirely in the cloud.

The ETL process used for this paper demonstrates a modern approach to preparing data from a warehouse and loading it into a target table in a database schema. However, for the analysis part, the results from the machine learning model built from the processed dataset display that the current dataset, based only on pre-game information, lacks the complexity needed to build a strong predictive model.

For future work, the utilized table information could be enhanced by keeping columns which contain events occurring during the matches. Elements such as match duration, control over key objectives like dragons and towers, and other gameplay events can strongly influence the final result of the match. Therefore, adding these variables would likely improve the model's predictive accuracy.

Overall, this paper demonstrates how ODI Web from the OCI suite can be effectively used to develop and maintain ETL operations. It represents a modern solution for data processing and analysis, as it also aids in building machine learning models.

References

[1] W. H. Inmon, *Building the data warehouse*, 4th ed. Wiley Pub., 2005.

[2] Oracle, "Introduction to Data Warehousing Concepts," Data Warehousing Guide, docs.oracle.com. https://docs.oracle.com/en/database/oracle/oracle-database/23/dwhsg/introduction-data-warehouse-concepts.html#GUID-AC9D6CAF-8698-478A-946C-6932F1FCDE99 (accessed June 2025).

[3] S. Chaudhuri and U. Dayal, "An overview of data warehousing and OLAP technology," *ACM SIGMOD Record*, vol. 26, no. 1, pp. 65–74, Mar. 1997.

[4] A. Simitsis and P. Vassiliadis, "Extraction, Transformation, and Loading," in

Encyclopedia of Database Systems, New York, NY: Springer New York, 2018, pp. 1432–1440.

[5] S. K. Bansal and S. Kagemann, "Integrating Big Data: A Semantic Extract-Transform-Load Framework," *Computer (Long Beach Calif)*, vol. 48, no. 3, pp. 42–50, Mar. 2015.

[6] Amazon Web Services, "What's the Difference Between ETL and ELT?," aws.amazon.com. https://aws.amazon.com/compare/the-difference-between-etl-and-elt/ (accessed June 2025).

[7] J. Brownlee, "Data Preparation for Machine Learning: Data Cleaning, Feature Selection, and Data Transforms in Python," Jun. 2020.

[8] Oracle, Oracle Cloud Infrastructure Documentation, "Welcome to Oracle Cloud Infrastructure," docs.oracle.com. https://docs.oracle.com/en-us/iaas/Content/GSG/Concepts/baremetalintro.htm (accessed June 2025).

[9] Oracle, "Why Customers are Choosing OCI," oracle.com. https://www.oracle.com/cloud/why-oci/ (accessed June 2025).

[10] Oracle, "Oracle Autonomous AI Lakehouse", oracle.com. https://www.oracle.com/autonomous-database/autonomous-data-warehouse/ (accessed June 2025).

[11] Oracle, "Data Integration," Oracle Cloud Infrastructure Documentation, docs.oracle.com. https://docs.oracle.com/en-us/iaas/Content/data-integration/home.htm (accessed June 2025).

[12] Oracle, "Overview of Data Integration," Oracle Cloud Infrastructure Documentation, docs.oracle.com. https://docs.oracle.com/en-us/iaas/Content/data-integration/using/overview.htm (accessed June 2025).

[13] Oracle, "What are Data Flows," Oracle Cloud Infrastructure Documentation, docs.oracle.com. https://docs.oracle.com/en-us/iaas/Content/data-integration/using/data-flows.htm (accessed June 2025).

[14] Oracle, "Using Data Flow Operators," Oracle Cloud Infrastructure Documentation, docs.oracle.com. https://docs.oracle.com/en-us/iaas/Content/data-integration/using/using-operators.htm#using-operators (accessed June 2025).

[15] Oracle, "Data Science," Oracle Cloud Infrastructure Documentation, docs.oracle.com. https://docs.oracle.com/en-us/iaas/Content/data-science/using/home.htm (accessed June 2025).

[16] Oracle, "Creating and Saving a Model to the Model Catalog," Oracle Cloud Infrastructure Documentation, docs.oracle.com. https://docs.oracle.com/en-us/iaas/Content/data-science/using/models_saving_catalog.htm#saving_models_catalog (accessed June 2025).

[17] Oracle, "The Data Transforms Page," Oracle Cloud Infrastructure Documentation, docs.oracle.com. https://docs.oracle.com/en-us/iaas/autonomous-database-serverless/doc/data-transforms.html#GUID-CD707617-5A77-4399-B4D9-0DC497141CDE (accessed June 2025).

[18] J. Mahto, "Introducing Data Transforms: Built in Data Integration for Autonomous Database," The Autonomous AI Database Insider, blogs.oracle.com, May 2023. https://blogs.oracle.com/datawarehousing/post/introducing-data-transforms-built-in-data-integration-for-autonomous-database (accessed June 2025).

[19] J. Francoisse and E. Lopes, "Comparing Oracle ETL/ELT Tools,"

blogs.oracle.com, May 2024. https://blogs.oracle.com/ateam/post/comparing-oracle-etlelt-tools (accessed June 2025).
[20] J. Mahto, "Selecting the best Data Integration tool – Data Transforms or Oracle Data Integrator," The Autonomous AI Database Insider, blogs.oracle.com, Mar. 2024. https://blogs.oracle.com/datawarehousing/post/selecting-the-best-data-integration-tool-data-transforms-or-oracle-data-integrator (accessed June 2025).
[21] Oracle, "Introduction to Oracle Data Transforms," Using Oracle Data Transforms, docs.oracle.com. https://docs.oracle.com/en/database/data-integration/data-transforms/using/introduction-oracle-data-transforms.html (accessed June 2025).
[22] Kaggle, "(LoL) League of Legends Ranked Games," kaggle.com. https://www.kaggle.com/datasets/datasnaek/league-of-legends (accessed June 2025).
[23] Riot Games, Riot Developer APIs, developer.riotgames.com. https://developer.riotgames.com/apis (accessed June 2025).
[24] Z. H. Hoo, J. Candlish, and D. Teare, "What is an ROC curve?," *Emergency Medicine Journal*, vol. 34, no. 6, pp. 357–359, Jun. 2017.
[25] GeeksforGeeks, "AUC-ROC Curve in Machine Learning," geeksforgeeks.org. https://www.geeksforgeeks.org/auc-roc-curve/ (accessed June 2025).
[26] L. Lin, "League of Legends Match Outcome Prediction," Stanford, CA, USA, 2016.

Cristiana COSTAN graduated from the Faculty of Cybernetics, Statistics and Economic Informatics of the Academy of Economic Studies in 2024, obtaining a Bachelor's Degree in Economic Informatics. She is currently pursuing a master's degree in Databases - Business Support and will earn her degree in 2026. Her main areas of interest include data analysis, database management, and web development.