# Cardano Analytics Platform Design

## Technical Report

*Draft version 1.0 – February 12th, 2025*

Project Catalyst Fund13
*Project id: 1300034*

**Contributors**

Marcio Moreno (mmoreno@mobr.ai)
Rafael Brandão (rafael@mobr.ai)

MOBR SYSTEMS

CC BY SA

# 1. Introduction

The Cardano Analytics Platform (CAP) is designed to simplify access to Cardano blockchain data through natural language queries powered by large language models (LLMs). This technical report discusses existing ontologies and Cardano glossaries, as well as strategies for supporting natural language queries on top of Cardano network data. These strategies include data extraction, ontology alignment, and visualization required to implement the platform. By leveraging existing APIs – such as the ones provided by cardano-node and cardano-db-sync – CAP aims to democratize Cardano blockchain analytics by providing easy-to-use tools for real-time insights into Cardano's ecosystem.

Large Language Models (LLMs) offer powerful natural language understanding capabilities, including flexible natural language processing, contextual understanding, and the ability to generate human-like responses. However, they face challenges with factual consistency and domain-specific knowledge, often producing hallucinations or inconsistent outputs when dealing with specialized domain knowledge.

In contrast, Knowledge Graphs (KGs) excel at representing structured domain knowledge and supporting precise querying. Their strengths include explicit relationship modeling, formal reasoning capabilities, and guaranteed factual consistency within their domain. However, KGs require specialized query languages and lack the flexibility to handle natural language inputs and outputs.

A Knowledge Graph-enhanced LLM combines these approaches to mitigate their respective limitations. By integrating LLM responses with a formal ontology and knowledge graph, it is possible to ensure factual accuracy while maintaining natural language interaction capabilities. In the context of blockchain analytics, this integration enables CAP to leverage the LLM's ability to interpret user queries while ensuring that responses are based on accurate Cardano blockchain data and relationships defined in our ontology and KG.

The ontological foundation of the CAP system consists of two main components: the TBox (terminological box), which contains concept definitions and relationships from the Cardano network, and the ABox (assertional box), which contains actual blockchain data instances extracted from Cardano. This structure enables both formal reasoning about blockchain concepts and practical analytics over real-world data.

# 2. Related Work

## 2.1 Ontologies

A number of ontologies have been conceived and applied in the context of Web3. However, to the best of our knowledge, no ontology has been specifically proposed to model the Cardano network. In this section, we briefly highlight four of the ontologies identified during our literature review.

### a. EthOn

The EthOn Ontology [1] is a formal representation of the Ethereum ecosystem that aims to provide a standardized vocabulary and ontology for Ethereum-related concepts and entities. Modeled using the Web Ontology Language (OWL), it is closely aligned with the Ethereum yellow paper[1]. Its use cases include enabling the semantic annotation of content produced by Ethereum-based tools and decentralized applications (DApps).

EthOn is organized into several modules, each corresponding to a specific aspect of the Ethereum ecosystem. These modules include: Core (fundamental classes and properties), Blockchain (blocks, transactions, and mining, etc.), Contracts (smart contracts, their state variables, and functions), Tokens (including ERC20, ERC721, and ERC1155 tokens), and DeFi (decentralized exchanges, liquidity pools, lending protocols, and others).

### b. BLONDIE

BLONDIE (BLockchain ONtology with Dynamic Extensibility) [2] is an ontology expressed in OWL language, aiming to support interoperability among different blockchain systems. In its current version, it covers Bitcoin, Ethereum and IBM Hyperledger. It provides a common vocabulary for describing chain-specific concepts and relationships. This can be useful for developing applications that need to interact with multiple blockchains or for comparing and contrasting different blockchain systems.

A potential limitation of BLONDIE is that it may not capture all the nuances and details of other blockchain systems, as it does not aim to provide a general and abstract model of blockchain technologies. Nonetheless, the ontology represents an important contribution toward the development of a standardized vocabulary for blockchain technology and has the potential to facilitate the development of interoperable blockchain applications.

### c. DLT Ontology

The DLT (Distributed Ledger Technologies) Ontology[2] [3] defines a set of concepts and relationships that capture the essential characteristics of DLT systems, including their

[1] https://ethereum.github.io/yellowpaper/paper.pdf
[2] https://dlt-ontology.github.io/

architecture, consensus mechanisms, transaction models, and data structures. The ontology is designed to be modular and extensible, allowing developers and researchers to tailor it to their specific needs. It includes concepts to model security aspects such as technical threats and vulnerabilities of distributed ledger systems, application domains, as well as relevant standards and regulations.

The ontology was developed based on collection of information and competency questions. It covers technical setup and components of DLT systems, security aspects, as well as applications and use cases. It covers 115 classes and 15 properties, and consists of a total of 571 triples.

### d. Smart Contracts

McAdams [4] presents a foundational ontology for reasoning about smart contract behavior through modal logic. The ontology conceptualizes smart contracts as stateful computations with fixed transition functions, where states can range from simple finite state machine tokens to complex symbolic data. Key concepts in this ontology include:

- Agent: Represents any participant in a contract (person, organization, or software) that can take actions affecting the contract's state.
- Event: Describes transitions between states, initiated by Agents invoking the contract's transition function.
- Object: Represents assets or items manipulated by the contract that may be transferred between Agents under contract-defined constraints.
- Time: Captures the external notion of calendar time, distinct from the blockchain's internal ordering.
- Modality: Describes relationships between possible states, divided into:
    - Vertical Modalities: Quantifying over alternative futures from a given state
    - Horizontal Modalities: Quantifying over sequences of states in possible futures

This ontology provides a formal framework for expressing and verifying important contract properties like rights (what actions agents may take) and obligations (ensuring contracts don't deadlock). While focused on behavioral aspects rather than implementation details, it complements other blockchain ontologies by providing tools for reasoning about contract correctness and security.

## 2.2 Cardano Glossaries

Although not standardized in a formal sense, glossaries represent curated collections of human-written definitions that reflect the evolving understanding of a domain. When creating an ontology for the Cardano blockchain, these glossaries are valuable because they capture different perspectives and practical knowledge about key concepts. They offer insights into the

terminology used by developers, community members, and stakeholders, which helps ontology designers align their models with real-world usage. There are key glossaries available in the Cardano ecosystem.

The Cardano Glossary[3] offers clear definitions of key terms and serves as a style guide to ensure consistency in how the community writes and communicates about the Cardano ecosystem. It is a resource for both understanding concepts and aligning on terminology.

The Essential Cardano Glossary[4] provides definitions of terms related to the Cardano blockchain, including entries on developments such as SanchoNet and Intersect. The Cardano Web3.js Glossary offers definitions and explanations of terms specific to the CardanoWeb3.js library, serving as a quick reference for developers working with this tool.

The Cardano for the Masses Glossary[5] provides definitions of various terms within the Cardano ecosystem, supporting users in understanding the platform's terminology.

There are other interesting glossaries related to the Cardano ecosystem, including the Lovelace Academy's Cardano Developer Vocabulary[6], with a concise glossary designed to explain important and commonly misunderstood terms in Cardano development, grouped by major concepts to assist developers in navigating the ecosystem.

## 2.3 Knowledge Graph Enhanced LLMs

Recent developments in combining Knowledge Graphs with Large Language Models have shown promising results in domain-specific applications [5]. For instance, RAG (Retrieval-Augmented Generation) enhances LLM responses by retrieving relevant information from a knowledge graph before generating answers. In blockchain analytics, this ensures that responses are grounded in actual blockchain data and relationships.

Additionally, KG-Attention Mechanisms modify an LLM's attention patterns to consider knowledge graph structure when processing queries. This helps maintain consistency with the underlying blockchain ontology. Advanced semantic parsing techniques assist by converting natural language queries into formal knowledge graph queries while preserving semantic intent, bridging the gap between user questions and structured blockchain data.

CAP will use these approaches to try to ensure that natural language queries are accurately mapped to its Cardano ontology and executed against the blockchain data. The system will maintain semantic coherence by aligning LLM outputs with the formal relationships defined in our knowledge graph.

[3] https://cardano.org/docs/glossary
[4] https://www.essentialcardano.io/glossary
[5] https://cardanobook.com/glossary
[6] https://learn.lovelace.academy/getting-started/cardano-developer-vocabulary

# 3. The Cardano Analytics Platform Design

The CAP is designed with a comprehensive methodology and workflow that integrates ontological reasoning, knowledge graph querying, and natural language processing. A summary of the designed workflow is illustrated in Figure 1.

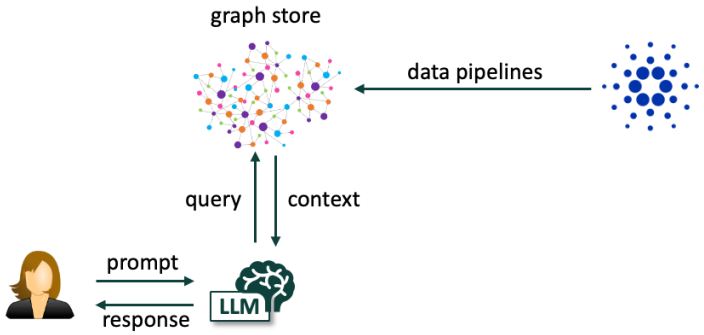



Figure 1. CAP's Workflow

Our methodology begins with use case development, where we analyze common blockchain analytics needs to identify representative queries and required data patterns. This analysis informs both the ontology design and the LLM training approach. CAP's ontology (see Section 5) is developed iteratively to ensure it covers all necessary concepts for the identified use cases while maintaining alignment with established blockchain ontologies.

For knowledge graph population, CAP relies on data pipelines to extract and transform Cardano data into a structured knowledge graph while maintaining consistency with the ontology. The data will be extracted mainly from a cardano-node instance and a PostgreSQL database managed by cardano-db-sync. The populated KG is the basis for supporting CAP's LLM model, which will be fine-tuning on a curated dataset of Cardano network queries and responses, emphasizing accuracy in technical terminology and data interpretation. The design includes both API endpoints and a user interface that supports natural language queries and visualization generation.

The system workflow begins with query reception, as users input natural language queries (prompts) through the interface. This is followed by semantic processing, where the fine-tuned LLM analyzes the query to identify relevant blockchain concepts and relationships, generating formal queries aligned with CAP's ontology. Another LLM component processes query results, generating natural language explanations and visualization recommendations. Finally, the dashboard component controller generates appropriate visualizations based on data characteristics and query intent.

# 4. Use Cases

CAP's use cases address a range of queries relevant to the Cardano community, including enthusiasts, CNT (Cardano Native Token) hodlers and traders, market analysts, SPOs (Stake Pool Operators), developers, and researchers. This section presents these use cases organized into four categories. Each use case includes the queries, data types, purpose, and strategies for data extraction and visualization. It is important to highlight that the use cases role is to support designing and evaluating the system, and not to limit its capabilities with only these queries.

The Transaction Analysis (see Section 4.1) primarily serves traders and market analysts who focus on transaction volume, fees, and transfer patterns across different time frames (e.g., hours, weeks). Account Analysis (see Section 4.2) provides insights for investors and researchers examining balance distribution, rewards, and growth over periods ranging from days to months.

Ecosystem Analysis (see Section 4.3) serves developers, market analysts, enthusiasts, and project managers investigating trends and usage patterns over periods. Governance Analysis (see Section 4.4) supports stake pool operators and community members in understanding participation rates and voting patterns over epochs and years.

The use cases address distinct user profiles with varying needs. Blockchain analysts possess high technical expertise and seek detailed network metrics through complex multi-metric analysis, preferring in-depth technical information. Investors have medium technical expertise and focus on market trends and value flows, seeking asset distribution and movement data with clear trend indicators.

Enthusiasts, or general community members, range from low to medium technical expertise, primarily focusing on network participation and governance. They are interested in stake pool performance and voting results, preferring simple, interpretable charts. Developers have high technical expertise and focus on smart contract metrics and network parameters, analyzing detailed technical data.

## 4.1 Transaction Analysis

### 4.1.1 ADA Transaction Analysis

This use case identifies the largest ADA transactions in the last 24 hours. This query requires transaction metadata, block timestamps, sender and receiver addresses, and transaction amounts. The purpose is to identify significant network activity for market analysis. Data is filtered by value thresholds, and the visualization can be presented as a ranked list or time-series chart.

**Natural Language Query:** What were the 10 largest ADA transactions in the last 24 hours?

**Purpose**: Monitor significant on-chain activity.

**Strategy**: Extract recent transactions using time filters and sort by amount. Visualize results as a ranked table.

**Data Types:** Transaction CNT amounts, sender/receiver addresses, block timestamps.

### 4.1.2 Transaction Fee Analysis

This use case calculates the average transaction fee over the last week. Transaction fee data and timestamps are sourced from cardano-db-sync. This analysis helps assess network efficiency and cost trends. Average fee trends are visualized as a line graph to highlight fluctuations.

**Natural Language Query:** What were the average transaction fees last week?

**Purpose**: Understand cost trends for users.

**Strategy**: Compute average fees from recent transaction records.

**Data Types:** Transaction fee amounts, block timestamps.

## 4.2 Account Analysis

### 4.2.1 Account Distribution Analysis

This use case explores the distribution of ADA by analyzing how many accounts hold over 200,000 ADA. Account balances and address data are used to verify individual account states. The purpose is to assess ADA distribution and concentration. Visualization of aggregated account data can be presented as a histogram to highlight distribution patterns.

**Natural Language Query:** How many accounts hold over 200,000 ADA?

**Purpose**: Evaluate ADA distribution among holders.

**Strategy**: Aggregate account data and filter based on balance thresholds.

**Data Types:** Account balances, address.

### 4.2.2 Staking Rewards Analysis

This use case analyzes staking rewards based on account balances. Reward and balance data from cardano-db-sync help evaluate return patterns for larger stakeholders. The purpose is to understand reward distribution patterns among significant ADA holders. Analysis results can be visualized as statistical summaries or as a scatter plot comparing stake amounts to rewards.

**Natural Language Query:** What are the average staking rewards for accounts holding more than 1 million ADA?

**Purpose**: Analyze reward patterns.

**Strategy**: Filter accounts by balance threshold and compute average rewards.

**Data Types:** Account balances, reward amounts.

## 4.3 Ecosystem Analysis

### 4.3.1 Smart Contract Deployment Trends

This use case examines the trend of monthly smart contract deployments over the last year. Required data includes contract deployment timestamps and metadata. This analysis supports understanding smart contract adoption trends. Monthly counts can be visualized as a bar chart to illustrate growth or decline over time.

**Natural Language Query:** Plot a bar chart showing monthly smart contract deployments in the last year.

**Purpose**: Analyze growth trends in smart contract activity.

**Strategy**: Group contracts by month and count deployments. Visualize it as a bar chart.

**Data Types:** Transactions referring to smart contract deployments, block timestamps.

### 4.3.2 NFT Activity Tracking

This use case tracks NFT activity, specifically the number of NFTs minted in the last month. Metadata and minting timestamps are used to monitor ecosystem growth. NFT minting data visualization can be presented as a count or as a bar chart illustrating monthly activity levels.

**Natural Language Query:** List the number of NFTs minted in the last month.

**Purpose**: Monitor NFT market dynamics.

**Strategy**: Count transactions with NFTs minting.

**Data Types:** Transactions referring to NFT minting, block timestamps.

### 4.3.3 Token Transfer Trends

This use case identifies the most frequently transferred tokens over the last week. Required data includes token IDs, transfer counts, and transaction details. The purpose is to highlight popular tokens for market insights. Ranked token data can be visualized as a bar chart to reveal transfer patterns.

**Natural Language Query:** Which tokens were most frequently transferred in the last week?

**Purpose**: Track token popularity.

**Strategy**: Group transactions by token and count transfers.

**Data Types:** Transactions with CNT amounts, block timestamps.

### 4.3.4 Growth Metrics

This use case tracks account growth by analyzing the number of new accounts created in the last month. Account creation timestamps are extracted from cardano-db-sync. This information supports user adoption assessments. Account creation trends are visualized as a time-series graph to depict growth rates.

**Natural Language Query:** How many new accounts were created daily over the last month?

**Purpose**: Assess network growth.

**Strategy**: Count new account records per day and visualize as a line graph.

**Data Types:** Transactions referring to account creation, block timestamps.

## 4.4 Governance Analysis

### 4.4.1 Delegation Patterns

This use case analyzes stake pool patterns, specifically determining the percentage value of ADA delegated to the top 10 stake pools. It requires stake pool registration data and epoch-specific delegation information. The purpose is to monitor decentralization metrics within the network. Data can be visualized as a numerical count or a trend line over epochs.

**Natural Language Query:** What percentage of delegated ADA is to the top 10 stake pools?

**Purpose**: Assess decentralization.

**Strategy**: Rank pools by delegation amount and calculate percentages.

**Data Types:** Stake pool address balance.

### 4.4.2 Governance Proposal Engagement

This use case investigates governance participation, focusing on the number of accounts that voted on a governance proposal. Voting transaction data and account address details support this analysis. The purpose is to evaluate community engagement. Visualization can be presented as a pie chart comparing active and inactive voting accounts.

**Natural Language Query:** How many accounts voted on the latest governance proposal?

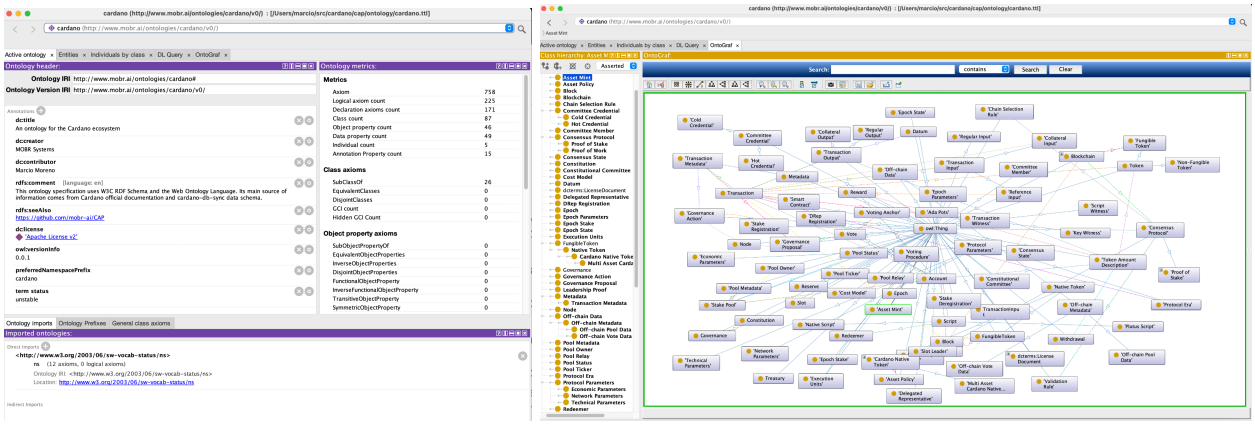
**Purpose**: Measure community participation.

**Strategy**: Count unique voting accounts per proposal.

**Data Types:** Account addresses, votes, block timestamps.

# 5. A Cardano Ontology

CAP's ontology provides a comprehensive conceptual framework for modeling the Cardano blockchain ecosystem. The ontology extends general blockchain concepts with Cardano-specific elements across several key domains. Figure 2 illustrates the ontology being curated by using the Protégé[7] tool.



(a) Ontology metrics (b) Main Concepts Visualization

Figure 2. Using Protégé to curate CAP's Ontology

The ontology's foundation includes standard blockchain elements such as Blocks, Transactions, and Accounts, enhanced with Cardano's unique implementation details. For instance, the Transaction model incorporates specialized input and output types, including collateral and reference inputs that support Cardano's extended UTXO model.

The ontology models Cardano Native Tokens (CNTs) as an extension of the general Token concept, with properties for policy IDs and minting rules. This enables the representation of both fungible and non-fungible tokens, along with their associated metadata and policies.

The Ouroboros proof-of-stake protocol is modeled through classes representing Slots, Epochs, and Leadership Proofs. The ontology captures both the technical mechanics of block production and the economic incentives associated with stake pools and rewards.

A substantial portion of the ontology models Cardano's governance framework, including Delegated Representatives (DReps) and various types of governance actions, such as parameter changes and treasury withdrawals. The ontology also represents voting procedures, proposal lifecycles, and the relationships between different governance actors.

By providing comprehensive coverage of the Cardano ecosystem, the ontology enables structured queries and advanced analysis, ensuring semantic clarity and formal reasoning capabilities.

The complete ontology documentation is available on CAP's GitHub repository at http://github.com/mobr-ai/cap and online at https://www.mobr.ai/cardano.

[7] https://protege.stanford.edu

# 6. Revisiting the Use Cases with SPARQL

This section demonstrates how CAP's ontology supports the use case queries. The ontology entities are used in SPARQL queries that can be executed against the knowledge graph. Each use case is revisited, showing the conversion from natural language to structured queries, which the platform will automatically generate using an LLM model. This process illustrates how the semantic model captures domain concepts and relationships.

The SPARQL queries leverage the ontology's rich semantic model to:

- Map user terminology to formal blockchain concepts;
- Handle temporal constraints (e.g., "last 24 hours", or "last week");
- Aggregate data at appropriate granularity levels;
- Apply complex filtering and sorting logic;
- Compute derived metrics and statistics.

These queries are part of CAP's test suite, available in the project's Github[8] repository.

## 6.1 Transaction Analysis

**Use Case 4.1.1**

**Natural Language Query:**

What were the 10 largest ADA transactions in the last 24 hours?

**SPARQL Query:**

```
SELECT ?tx ?amount ?timestamp
WHERE {
    ?block a blockchain:Block ;
        blockchain:hasTransaction ?tx ;
        blockchain:hasTimestamp ?timestamp .
    ?tx cardano:hasOutput ?output .
    ?output blockchain:hasTokenAmount ?tokenAmount .
    ?tokenAmount blockchain:hasCurrency cardano:ADA ;
                 blockchain:hasAmountValue ?amount .
    BIND(NOW() - "P1D"^^xsd:dayTimeDuration as ?oneDayAgo)
    FILTER(?timestamp >= ?oneDayAgo)
}
ORDER BY DESC(?amount)
LIMIT 10
```

[8] http://github.com/mobr-ai/cap

**Use Case 4.1.2**

**Natural Language Query:**

What were the average transaction fees last week?

**SPARQL Query:**

```
SELECT (AVG(?fee) as ?avgFee)
WHERE {
    ?block a blockchain:Block ;
        blockchain:hasTransaction ?tx ;
        blockchain:hasTimestamp ?ts .
    ?tx cardano:hasFee ?fee .
    BIND(NOW() - "P7D"^^xsd:dayTimeDuration as ?oneWeekAgo)
    FILTER (?ts >= ?oneWeekAgo)
}
```

## 6.2 Account Analysis

**Use Case 4.2.1**

**Natural Language Query:**

How many accounts hold over 200,000 ADA?

**SPARQL Query:**

```
SELECT (COUNT(DISTINCT ?account) as ?numAccounts)
WHERE {
    ?account a blockchain:Account ;
             blockchain:hasTokenAmount ?tokenAmount .
    ?tokenAmount blockchain:hasCurrency cardano:ADA ;
                 blockchain:hasAmountValue ?amount .
    FILTER(?amount > 200000000000) # Lovelace conversion
}
```

**Use Case 4.2.2**

**Natural Language Query:**

What are the average staking rewards for accounts holding more than 1 million ADA?

**SPARQL Query:**

```
SELECT AVG(?rewardValue)
WHERE {
  ?account blockchain:hasTokenAmount ?tokenAmount .
  ?tokenAmount blockchain:hasCurrency cardano:ADA .
  ?tokenAmount blockchain:hasAmountValue ?adaAmount .
  ?account cardano:hasReward ?reward .
  ?reward cardano:hasRewardAmount ?rewardTokenAmount .
  ?rewardTokenAmount blockchain:hasAmountValue ?rewardValue .
  FILTER (?adaAmount > 1000000000000) .
}
```

## 6.3 Ecosystem Analysis

### Use Case 4.3.1

**Natural Language Query:**

Plot a bar chart showing monthly smart contract deployments in the last year

**SPARQL Query:**

```
SELECT (MONTH(?ts) as ?month) (COUNT(DISTINCT ?contract) as ?deployments)
WHERE {
    ?block a blockchain:Block ;
        blockchain:hasTransaction ?tx ;
        blockchain:hasTimestamp ?ts .
    ?contract a blockchain:SmartContract ;
        cardano:embeddedIn ?tx .
    BIND(NOW() - "P365D"^^xsd:dayTimeDuration as ?oneYearAgo)
    FILTER (?ts >= ?oneYearAgo)
}
GROUP BY MONTH(?ts)
ORDER BY ?month
```

### Use Case 4.3.2

**Natural Language Query:**

List the number of NFTs minted in the last month?

**SPARQL Query:**

```
SELECT COUNT(DISTINCT ?nft)
WHERE {
    ?block a blockchain:Block ;
        blockchain:hasTransaction ?tx ;
        blockchain:hasTimestamp ?ts .
    ?tx cardano:hasMintedAsset ?nft .
    ?nft a blockchain:NFT .
    BIND(NOW() - "P30D"^^xsd:dayTimeDuration as ?oneMonthAgo)
    FILTER (?ts >= oneMonthAgo)
}
```

### Use Case 4.3.3

**Natural Language Query:**

Which tokens were most frequently transferred in the last week?

**SPARQL Query:**

```
SELECT ?token (COUNT(DISTINCT ?output) as ?transfers
WHERE {
    ?block a blockchain:Block ;
        blockchain:hasTransaction ?tx ;
        blockchain:hasTimestamp ?ts .
    ?tx cardano:hasOutput ?output .
    ?output blockchain:hasTokenAmount ?tokenAmount .
    ?tokenAmount blockchain:hasCurrency ?token .
    BIND(NOW() - "P7D"^^xsd:dayTimeDuration as ?oneWeekAgo)
    FILTER (?ts >= ?oneWeekAgo)
}
GROUP BY ?token
ORDER BY DESC (?transfers)
LIMIT 10
```

**Use Case 4.3.4**

**Natural Language Query:**

How many new accounts were created daily over the last month?

**SPARQL Query:**

```
SELECT (xsd:date(?timestamp) as ?date) (COUNT(DISTINCT ?account) as
?new_accounts)
WHERE {
  ?account blockchain:firstAppearedInTransaction ?tx .
  ?block blockchain:hasTransaction ?tx .
  ?block blockchain:hasTimestamp ?timestamp .
  BIND(NOW() - "P30D"^^xsd:dayTimeDuration as ?oneMonthAgo)
  FILTER (?timestamp >= ?oneMonthAgo)
}
GROUP BY xsd:date(?timestamp)
ORDER BY ?date
```

## 6.4 Governance Analysis

**Use Case 4.4.1**

**Natural Language Query:**

What percentage of delegated ADA is to the top 10 stake pools?

**SPARQL Query:**

```
SELECT (SUM(?stakeAmount) * 100 / ?totalStaked as ?percentage)
WHERE {
  {
    SELECT ?pool (SUM(?amount) as ?stakeAmount)
    WHERE {
      ?account cardano:delegatesTo ?pool ;
               cardano:hasStakeAmount ?amount .
      ?pool a cardano:StakePool .
    }
    GROUP BY ?pool
    ORDER BY DESC(?stakeAmount)
    LIMIT 10
  }
  {
    SELECT (SUM(?amount) as ?totalStaked)
    WHERE {
      ?account cardano:hasStakeAmount ?amount .
    }
  }
}
```

**Use Case 4.4.2**

**Natural Language Query:**

How many accounts voted on the latest governance proposal?

**SPARQL Query:**

```
SELECT COUNT(DISTINCT ?account)
WHERE {
  {
      SELECT ?proposal
      WHERE {
          ?block blockchain:hasTimestamp ?timestamp ;
             blockchain:hasTransaction ?tx .
          ?tx cardano:hasTransactionMetadata ?metadata .
          ?metadata cardano:hasGovernanceProposal ?proposal .
      }
      ORDER BY DESC(?timestamp)
      LIMIT 1
  }
  ?account cardano:castsVote ?vote .
  ?proposal cardano:hasVote ?vote .
}
```

# 7. Final Remarks

The Cardano Analytics Platform represents a significant advancement in facilitating access to blockchain data analysis. By combining a comprehensive domain ontology with natural language processing capabilities, it bridges the gap between technical blockchain data and meaningful insights accessible to a wide range of stakeholders.

The platform's architecture introduces several key innovations for the Cardano community. The use of semantic technologies enables rich context-aware queries that understand the relationships across different aspects of the Cardano ecosystem. The LLM-powered query system enhances accessibility, allowing users without expertise in SPARQL or blockchain internals to perform complex analytics. In addition, the modular design supports continuous improvements to both the ontological model and analytical capabilities as the Cardano ecosystem evolves.

By focusing on user personas and specific use cases, the platform ensures practical utility while maintaining the flexibility to adapt to emerging needs in blockchain analytics. Upon completing the milestones outlined in the project proposal, the team envisions productizing the platform and integrating cross-chain analytics to enable comparative analysis across multiple blockchain platforms. Another potential future enhancement is the expansion of the knowledge graph to incorporate off-chain data sources, further enriching the analytical scope.

# 8. Acknowledgments

This work was supported by a grant from the Fund13 of Project Catalyst. We would like to express our gratitude to the Cardano community and the Project Catalyst Team.

# References


[1] Pfeffer, J., Beregszazi, A., & Li, S. (2016). EthOn-an Ethereum ontology.

[2] Ugarte-Rojas, Hector. Chullo-Llave, Boris. (2020). BLONDiE: Blockchain Ontology with Dynamic Extensibility.

[3] König, Lukas. Neumaier, Sebastian. (2023). Building a Knowledge Graph of Distributed Ledger Technologies.

[4] McAdams, Darryl. (2017). An Ontology for Smart Contracts.

[5] Collection of papers and resources about unifying large language models and knowledge graphs. Available at: https://github.com/RManLuo/Awesome-LLM-KG?tab=readme-ov-file#kg-enhanced-llms