**International Journal of Sustainability and Innovation in Engineering (IJSIE)**
https://www.doi.org/10.56830/IJSIE2026
**Vol. 4 2026**

# AI-Driven Database Sharding Strategies for Nationwide Prescription Drug Monitoring Programs

https://www.doi.org/10.56830/IJSIE202620

**Karan Radadiya**
PhD, Information Technology, University of the Cumberlands, KY, USA
Email: karanradadiya787@gmail.com
*Received:23 July, 2026, Accepted:31h October, 2026, Published: 18th November, 2026*

**Abstract:**

Prescription Drug Monitoring Programs (PDMPs) are databases of prescription information that are used to assist in the monitoring of controlled substances. As more data accumulates and cross-shard queries increase, traditional database architecture can easily lead to an imbalanced workload, increased cross-shard queries, and delayed responses to jurisdictional needs. The objective of the study is to share a national PDMP data management system based on artificial intelligence (AI). It has a hierarchical partitioning structure, with geographic jurisdiction as the lowest (first) level in the hierarchy, and the patient identifier is hashed within each jurisdiction. Historical retrieval is supported by temporal indexing, and risk classifications are metadata rather than shards. Short-term (Days) workload predictions are made using query activity and resource metrics with a regression model named XGBoost. The forecasts determine where hotspots will be found and where they should be placed. Access was governed through jurisdiction-aware query routing, role- and attribute-based access controls, and audit logging. Rebalancing is performed with hysteresis and migration costs to minimise data movement during rebalancing. The methodology is based on a comparative evaluation using synthetically generated PDMP-like data in a PostgreSQL-Citus environment. Static and adaptive configurations are evaluated based on query latency, query throughput, shard-load distribution, forecasting error, hotspot detection and rebalancing overhead. The framework offers methods to handle PDMP data at scale. No quantitative improvement in performance or clinical outcome is claimed, as there are no operational measurements. More testing with workloads and approved operational data is required before deployment.

**Index Terms:** AI-Driven Sharding; Prescription Drug Monitoring Programs (PDMPs); Database Scalability; Healthcare Data Security

## 1. Introduction

Prescription Drug Monitoring Programs (PDMPs) are public health programs to gather and share data about dispensed controlled substances. Records are submitted to pharmacies and can be accessed by authorised prescribers, pharmacists, and public health agencies to check medication histories and for other assessment purposes. Records contain information such as patient identifiers, medication details, prescription dates, dosage, patient prescription history, pharmacy and prescriber information. The volume of data and interstate requests are growing, so PDMP infrastructure needs to enable the reliable storage, retrieval and protection of sensitive and confidential information.

Centralised databases may encounter problems as they grow. As data grows, storage and processing requirements increase, and uneven query activity can cause the workload to be distributed unevenly across certain nodes. Each State may have unique PDMP systems and varying data formats, reporting, access, and technical infrastructure. These differences make it

difficult to achieve interoperability and perform cross-jurisdictional queries. Poor partitioning can also require requests to be made to several shards, complicating processing. Database architecture needs to optimise data locality, workload distribution, query performance, and jurisdictional controls to achieve these goals.

Sharding allows records to be distributed across different database nodes, improving scalability and workload handling. Fixed partitioning, however, may not meet varying demands well. Geographic partitioning is good for jurisdictional locality, but may result in hotspots in high-activity regions. Hash-based partitioning can achieve more even load balancing of records, but can decrease locality for jurisdiction-specific queries. The proposed solution is a hierarchical structure: the first partition is geographic jurisdiction, the next is patient-ID hashing, and the final partition is temporal indexing for history retrieval. The metadata (risk categories) are not physical shards.

The proposed framework uses an AI approach based on XGBoost regression to predict near-future database workloads. It uses forecasts and system-monitoring information to help identify hotspots and inform shard placement. Jurisdiction-aware query routing: routing queries for access to relevant data while obeying access rules. The monitoring and threshold-based rebalancing solution handles workload imbalance across the system, and integrating hysteresis and migration-cost considerations helps prevent unnecessary data movement. The framework synthesises workload intelligence, database management and governance controls, and database optimisation, but it cannot replace clinical risk assessment.

The ultimate goal is to design and evaluate an AI-based database sharding solution for high scalability in PDMP scenarios. Specific Objectives are: To create a hierarchical sharding scheme with a compromise between locality of the jurisdictions and load distribution. To build an XGBoost model to predict workloads and identify potential hotspots. To understand the concept of jurisdiction-aware query routing, access control, monitoring and adaptive rebalancing. To define an experimental method for testing and comparing the setup to a statically sharded setup.

The study focuses on prescriber, patient, prescription, and controlled substance data. The proposed evaluation is based on synthetic PDMP-like data because of their sensitivity. The methodology will use a PostgreSQL-Citus environment and measure query latency, throughput, load distribution, forecasting accuracy, and rebalancing overhead. It evaluates database operational efficiency, rather than clinical outcomes such as overdose prevention or regulatory adherence.

The rest of the paper is organised as follows: Chapter 2 covers an overview of PDMP data systems, sharding, AI workload optimisation, and healthcare data governance. The framework is given in Chapter 3. The methodology used in the experiments is described in Chapter 4. The evaluation approach and performance interpretation are covered in Chapter 5. Implications and limitations are discussed in Chapter 6, recommendations and future research in Chapter 7, and the end of the study in Chapter 8.

## 2. Literature Review

### 2.1. Prescription Drug Monitoring Data Systems

Prescription Drug Monitoring Programs (PDMPs) need timely access to prescription information to perform their clinical and regulatory functions. Structured data in these systems includes patient information, prescriber information, pharmacist information, drug types, strengths, refill history, and prescription information. Adopting a solution that is continuously updated by pharmacies and integrated with clinical systems imposes significant data

management needs, including performance, security, and availability (Jha et al., 2025; Sardana et al., 2025).

Interoperability is another hurdle for PDMP systems, as prescription information may need to be shared across state lines. Information exchange between states can be complicated because of variations in data formats, access methods, and update procedures. For this reason, it is crucial that data is organised efficiently and queries are processed efficiently to facilitate timely access to prescription histories.

Figure 1 shows the relationship between a PDMP requester (prescriber/pharmacist) and a PDMP system: a query is sent to the PDMP system, and a response is returned, allowing the prescriber/pharmacist to access patient prescription history in real time to support informed clinical decision-making.

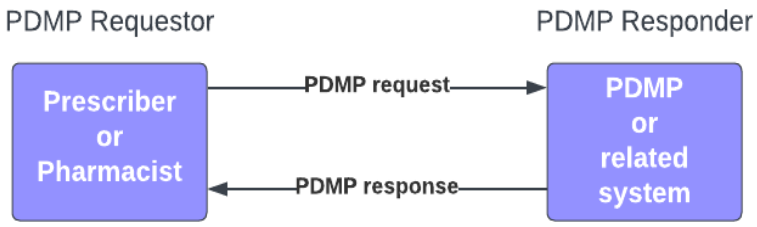



**Figure 1. The interaction between a PDMP requestor, such as a prescriber or pharmacist, and a PDMP system, where prescription queries are sent and responses are returned**

### 2.2. Standard Database Sharding Models

Sharding a database can improve scalability by distributing data across multiple database nodes. Typical types are range-based, hash-based, and geographically based sharding. Range-based sharding: Partitioning based on ranges, for example, geographic ranges or time ranges. While this makes data easier to organise, it can result in data being spread out across the nodes and causing nodes to become overloaded if data is coming from a particular area or time. In hash-based sharding, the key used in the hash function is a specific field in the record, such as a patient identifier. This can help balance workloads and simplify some queries that need to run across related data. However, it can reduce data locality and increase the coordination needed for queries that span related data. Geographic sharding is a way of partitioning data across administrative regions, for example, states or regions. It can enhance local data access and support jurisdiction-based organisation. There may be some variation in prescription activity by jurisdiction; however, this will still lead to some variation in resource use. The following methods involve trade-offs in data locality, workload distribution, and cross-shard query processing. Geographic + hash-based partitioning can therefore be a basis for handling distributed PDMP records and supporting their different access requirements (Durgam, 2025).

### 2.3. Database Workload Optimization with AI

In large-scale database systems, workload analysis and resource management are two areas that have been addressed using artificial intelligence (AI). Historical access patterns can be used to predict workload changes and periods of higher demand using techniques such as Long Short-Term Memory (LSTM) networks and AutoRegressive Integrated Moving Average (ARIMA) models. Workload forecasting can be used to ensure that resources are allocated in advance and congestion is alleviated during periods of heavy database use. Other methods are

based on the query log, identifying frequently accessed records and optimising data placement. Another approach to adaptive resource allocation is to apply RL to make management decisions based on system feedback. The methods presented here show the promise of predictive and adaptive approaches for enhancing the performance of distributed systems (Kumar, 2025a, 2025b; Lingamgunta et al., 2025). But they work well only under certain conditions of workload, prediction accuracy, and implementation costs for making management decisions. With this in mind, more work is needed to apply workload prediction to the sharding of PDMP databases and to take jurisdictional data organisation, sensitive prescription information, and controlled access into account. In this study, workload prediction is done via XGBoost regression and the prediction results are used for a rule-based shard-management mechanism.

### 2.4. Security and Compliance of Healthcare Data

Given the sensitive nature of patient and prescription information, several robust privacy and security measures are needed in a healthcare database. Encryption can help secure information in transit, and privacy-preserving methods, such as federated learning and privacy-preserving data-sharing technologies, can enable analysis across multiple environments without unnecessarily exposing patient-level information (Ubale, 2025). Access control is also crucial. Role-based access control (RBAC) uses user roles (e.g., prescribers, pharmacists, and regulators) to assign permissions. Attribute-based access control (ABAC) adds to that flexibility by allowing consideration of attributes like access purpose, jurisdiction, and user identity.

In addition to the aforementioned governance requirements, data sharing across interstate PDMPs adds additional requirements, since jurisdictions may have differing privacy requirements and access controls. Distributed systems must therefore implement proper authorisation, auditing, and secure data across different jurisdictions. Other distributed computing systems have also been investigated for their security and data-management aspects, using AI approaches (Chadha, 2025; Chadha et al., 2025). However, in PDMP applications, their use shall comply with relevant privacy, security, and jurisdiction requirements.

### 2.5. Research Gap

Existing research has focused on database scalability, workload optimisation, security in health care data, and the reliability of distributed systems. Unfortunately, these areas are not always part of an overall strategy to manage distributed PDMP databases. However, traditional sharding strategies can compromise data locality or workload distribution. With AI-based workload prediction, an appropriate mechanism is needed to translate a prediction into a database-management decision. Moreover, cross-jurisdictional access also raises other issues of authorisation, data governance and information security. The area of reliability and automation in distributed (and cloud) system research offers pertinent background for solving these challenges (Vanama, 2023, 2024). Other privacy-preserving methods also highlight the importance of protecting sensitive information throughout distributed analysis (Lingamgunta et al., 2025; Ubale, 2025). This study aims to fill the identified gap by proposing an AI-based sharding framework that integrates geographic partitioning, patient-ID hashing within jurisdictions, XGBoost-based workload prediction, jurisdiction-aware query routing, and monitored shard rebalancing. The framework aims to enhance workload management by ensuring appropriate data organisation and access control.

# 3. Proposed AI-Driven Sharding Framework

## 3.1. Framework Overview

The design proposes an artificial intelligence (AI)-based database sharding approach for a large-scale prescription drug monitoring program (PDMP). It is intended to handle vast amounts of prescription data, make queries more responsive, distribute the workload evenly across database nodes, and utilise resources efficiently. The framework incorporates workload prediction, data partitioning, query routing and adaptive shard management into a single solution to meet nationwide PDMP needs.

The architecture is comprised of three distinct layers: Data management layer, Workload intelligence layer, and Governance and access-control layer. The data management layer manages the classification, partitioning, and distribution of prescription data across the database nodes. The workload intelligence layer uses an XGBoost regression model to predict database needs and detect potential workload imbalances. The access control and governance layer provides access control, restrictions, and auditing of user database activity. Together, these layers manage information distribution, workload management, and access to prescription information (Vishwakarma, 2025a, 2025b).

The framework is based on a hierarchical partitioning structure that includes geographic partitioning, hash-based partitioning, temporal organisation, and risk-related metadata. Geographic jurisdiction determines the basic data boundary, and patient-identifier hashing ensures records are sharded across the distributed system within each jurisdiction. Temporal organisation is used to retrieve and manage historical prescription data, while risk-related attributes are stored as metadata and indexing features. This helps achieve data locality and minimises unwanted cross-shard operations. It also supports monitoring and adaptive rebalancing to adjust to shifting prescription usage patterns and database workload. A workload forecasting component will provide information on expected demand, and an independent shard-management component will indicate whether data needs to be redistributed. The management portion considers the workload predicted by the system, node fullness, available space, and the estimated data migration cost before deciding whether to change shard placement. This separation ensures that workload predictions do not directly influence the physical structure of the database, but rather how it is managed.

The proposed architecture will enable efficient retrieval of prescription history, including authorised queries across multiple jurisdictions. It offers a data-management framework for accessing prescription data and for supporting the analytical capabilities of PDMP applications. The framework does not automatically identify clinical risk, inappropriate prescribing and/or better patient health outcomes. The most significant aspect is the proposed data organisation, workload distribution, and query-execution optimisations in distributed PDMP databases.

Figure 2 illustrates the proposed AI-driven PDMP sharding framework, showing the interaction between data management, workload intelligence, and governance layers. It emphasises hierarchical partitioning, workload forecasting, intelligent query routing, monitoring, rebalancing, and security to support efficient database management.

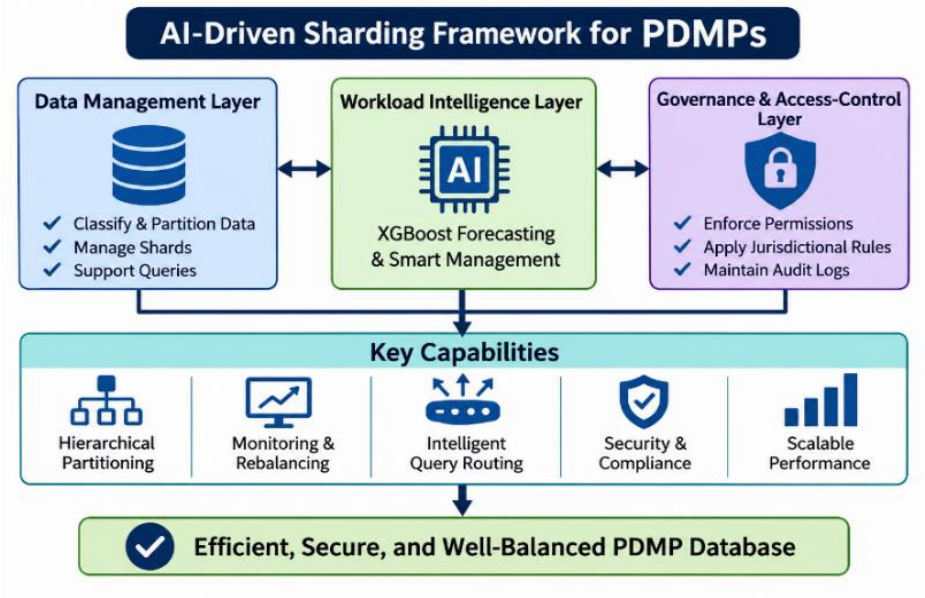



**Figure 2. AI-driven PDMP sharding framework**

### 3.2. Data Classification Layer

The data classification layer classifies PDMP data into recognisable categories before assigning records to the respective jurisdictional partitions. This classification enables the consistent storage, indexing, retrieval and access control in the distributed database environment.

These fall into the following main groups: Patient prescription history: patient information, prescription date, prescription details, dose information, refills, dispensing history. These records can be used to provide chronological information for authorised prescription-history requests. Prescriber records: Store relevant prescriber identifiers, prescriber information, and prescribing activity. They assist in obtaining and examining prescribing records. Pharmacy records: Contain information on the pharmacy that dispensed, information about the location of the pharmacy, and information on dispensing activity. These records are used to organise and identify prescription transactions connected to the dispensing facilities. Drug classification records: Record the identification and drug schedule classifications of medicines. These attributes are all used to classify and retrieve information on controlled substances. Prescription transactions and interstate transactions: Document relevant prescription transactions or authorised information exchanges between jurisdictions. They enable records to be identified and retrieved for cross-jurisdictional inquiries.

The classification layer also holds operational metadata such as jurisdiction, patient reference identifiers, prescription dates, record status, risk classifications, and other prescription-related attributes. Risk-related attributes can include controlled-substance schedules and pre-defined prescription pattern indicators for the relevant PDMP application. These attributes facilitate filtering, indexing, and analytical queries without knowing the specific location of prescription records. Records entering the system are checked for accuracy, and the appropriate authority to maintain the authoritative prescription record is assigned. Patient identifiers are mapped using an approved patient identifier mapping system to facilitate matching and retrieval of patient records without routing operations in the database that expose patient identifiers.

The classification process also maintains attributes from the source and jurisdiction that are required for authorised interstate enquiries. This classification system keeps the logical

organisation of prescription information separate from the distribution of the information. This means that records can be accessed based on attributes of the patient, prescriber, pharmacy, time period, and/or risk, without needing to store separate records for each attribute. It can also be used to manage large and diverse data sets in distributed database systems, where balancing query performance, consistency, and scalability is essential in the architecture (Dhanagari, 2025).

### 3.3. Machine Learning-Based Shard Selection

The machine-learning-based shard selection component helps distribute prescription records and workload among the database nodes. When determining whether a potential workload imbalance exists, it considers the number of regional prescriptions, the number of queries, and database resource utilisation. The monitoring component will continuously monitor database performance to track nodes that are becoming more demanding.

The framework proposed in this article is based on Extreme Gradient Boosting (XGBoost) regression for predicting the database workload in the near future. XGBoost was selected because the workload information consists mainly of structured numerical and categorical features, such as query frequency, prescription transaction volume, and node resource utilisation. A tree-based ensemble structure can be used to model nonlinear relationships between these variables, and it can be used to achieve manageable model training and prediction.

The model is fed workload observations over a specified monitoring period. Some of its input components are: (1) Number of queries per shard over time, (2) Read/Write transactions per prescription, (3) CPU utilisation of the database host node, measured at average and peak levels, (4) memory usage and storage consumption, (5) network use associated with database requests, (6) variation of the database workload per monitoring interval, (7) number of queries sent to each jurisdictional partition over time.

The prediction target is the workload that is expected for the next monitoring interval for each shard. The model generates several workload forecasts, which are compared with the available number of processing units on the node where the model will be deployed. These forecasts enable the discovery of shards likely to handle more load and nodes that can provide the capacity to handle more workloads. The supervised learning process for developing the models is based on past workload observations. The observations are split into three subsets: training, validation, and testing, to maintain the order of workload events over time. The validation subset is used to select model parameters, and the testing subset is used to evaluate the model's predictive performance. The Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Coefficient of Determination ($R^2$) are used to evaluate forecasting performance. Therefore, accuracy percentage (as often assumed) is not the basis for determining the effectiveness of workload forecasting; instead, effectiveness is determined by measuring prediction errors.

A forecasting function is integrated with another function (a monitoring function) that stores the database's actual performance. If the number of operations on the shard approaches the capacity of the node it runs on, the shard is deemed a hotspot and is considered for further action. To determine the nodes that need to be further investigated in terms of CPU utilisation, a threshold of 75% or higher can be used. However, crossing the threshold alone is not enough to start data migration. In addition, the management component assesses workload persistence, memory space, query demands, and the capacity of alternative nodes before deciding where to place the workload.

There are four steps in the shard-selection process. In the first step, the monitoring part collects the workload and resource utilisation information. Secondly, the XGBoost model predicts the amount of work to be done on each shard in the subsequent monitoring period. Thirdly, the management component predicts workload and compares it with node capacity to identify nodes that may become hotspots and suitable destination nodes. Lastly, the placement component assesses available destinations, and the placement location is determined to minimise projected workload imbalance while meeting jurisdictional placement requirements. The forecasting model does not have its own splitting, merging and migration capabilities. Shard management remains responsible for these operations and follows predefined rules, taking migration overhead into account before making changes. This separation reduces the risk of unwanted changes to the database structure due to prediction errors.

Also, workload fluctuations should be managed appropriately for adaptive shard management to work. If a shard is subjected to high demands for a long time, it can be split into smaller partitions, which would be spread among other nodes. On the other hand, if multiple shards are underutilised, there may be opportunities to merge them so they do not waste resources. Subdivision is therefore linked to a high workload, while consolidation is linked to a low workload and available capacity. They are adaptive techniques that support dynamic shard management and workload distribution in distributed database schemas (Samala, 2025a, 2025b).

### 3.4. Sharding Strategy Design

The proposed framework merges several data-management strategies to give PDMP systems the flexibility to operate. The design is based on partitioning, the primary physical distribution mechanism, with geographic and hash-based partitioning as primary mechanisms, and temporal organisation and risk classification as supplementary data management mechanisms. Each mechanism serves a distinct purpose in the architecture.

The primary partitioning boundary is geographic. Each prescription record has an authoritative prescription record that is nominated to the jurisdiction responsible for maintaining it. This enables data governance and keeps a prescription record in a jurisdiction's administrative homeland, within the scope of any interstate data-sharing agreements and legal requirements. It also enhances data locality by minimising the amount of data that is moved in queries that involve records that are stored in a single jurisdiction.

Each jurisdiction has a unique hash function that takes the patient identifier as input and maps it to a shard within a cluster. Thus, instead of partitioning across the national database, hash partitioning is performed within each jurisdiction. This maintains geographic data locality while spreading records across nodes within each jurisdiction more evenly. The geographic and hash-based structure is used to implement temporal organisation. Prescription timestamps support chronological indexing, filtering, and management of recent and historical records. If the underlying database system permits, records that are accessed often may be kept in a separate set from older historical records for efficient retrieval. Time is then used as a key to index and organise data, rather than as a separate national partitioning key. There are no shards that are based on risk classification -- searchable metadata and indexing attributes are used to maintain risk classification. Prescription records may also include elements that specify whether a drug is included in any drug schedule, as well as predefined risk elements. These attributes enable queries and authorised analytics to be performed on the records without moving them if the records' risk classification changes. Changing a risk-related attribute updates the corresponding metadata and indices, but doesn't automatically trigger a shard migration.

The hierarchical structure will also support patients with prescription histories across multiple jurisdictions. Prescription records are kept by individual jurisdictions, and all records are authoritative for prescriptions issued within that jurisdiction. If an authorised inquiry needs a more extensive prescription history, the query routing component determines which jurisdictions to query and fetches the authorised prescription records from each jurisdiction's partition. A patient is identified by the approved identity-resolution process; it isn't assumed that a patient has a single physical shard that is shared nationally.

This design overcomes the shortcomings of using only geographic partitioning or hash partitioning. Geographic partitioning sets the administrative boundary; hash partitioning distributes records across the boundary; temporal indexing enables chronological access; and risk-related metadata enables access to relevant prescription information. Together, these two form a system that balances data locality with workload distribution across shards, with minimal cross-shard communication cost (Rangu, 2025a, 2025b).

### 3.5. Query Routing and Access Control

Incoming queries are routed to the database shards where the required prescription information is stored, with AI assistance. The routing mechanism considers the query type, the query's geographic source, how often the query is requested, and the sensitivity of the requested data to determine an appropriate retrieval path. Routing decisions are mainly based on the jurisdictional characteristics of the requested records, the information identifying the requested records as provided by the query, and the permissions of the user who issued the query. Workload forecasts and node conditions can be used to provide supplemental information to facilitate the selection of an eligible database endpoint. Workload optimisation is not intended to override jurisdictional rules or data-access privileges, however.

The routing process is a controlled sequence of operations that includes the following: If the requested information spans multiple jurisdictions, only information from jurisdictions with the required authorisation in the requesting application is retrieved.

Query plans, selective data retrieval, and caching (when possible) support cross-shard query optimisation. To eliminate unwanted data transfers between nodes and to minimise the number of requests made to the database, popular data may be cached. Caching, however, must follow the same rules of authorisation, expiry, and data protection as the underlying records. Cached data is also subject to proper expiration and invalidation logic to ensure that it doesn't use outdated data following changes in the source database.

Different user groups, such as prescribers, pharmacists, and regulators, are differentiated by their permissions using role-based access control. Other jurisdictions and purpose-based restrictions derived from existing PDMP governance arrangements are also considered when making access decisions. In addition, audit logs are captured for authentication and authorisation, as well as for query execution and access to sensitive prescription information. This means that security and jurisdictional authorisation are not considered performance considerations when deciding how to execute a query, but are treated as necessary.

### 3.6. Monitoring and Rebalancing Layer

System performance and resource consumption are continuously monitored to ensure efficiency. The monitoring element tracks query response time, transaction count, CPU usage, memory usage, storage space usage, and network activity. These are meant to provide information on the state of the distributed database and to help detect workload imbalance between nodes. The rebalancing mechanism includes the workload-observed conditions, XGBoost predictions and fixed management conditions. An initial CPU utilisation threshold of 75% is used to determine nodes to be looked at more closely. However, if only one threshold

crossing occurs, data redistribution does not automatically begin. The system first checks whether the increased workload is sustained, whether the involved shard is likely to remain under load for a significant period, and whether there is an appropriate target node in the system with enough space to receive the data.

The framework uses hysteresis to avoid any repeated data transfers when the workload varies from one moment to the next. Subdivisions and/or migrations are not considered until a high-load condition is maintained over successive monitoring intervals. Similarly, consolidation is considered only when utilisation over a specified period is low enough, and the placement involved will guarantee processing capacity. The monitoring intervals and thresholds are adjustable, and their appropriateness should be determined through experimental testing. The rebalancing process consists of 4 stages: [1] Detection: The monitoring part detects a shard whose observed or forecast workload shows that the shard's resources are continuously being utilised and are under heavy stress or underutilised. [2] Evaluation: The management section analyses the expected load, the destination node's capacity, the data volume, the network transfer requirements, and the effect of the migration on query performance. [3] Decision: Decides the need for subdivision, migration or consolidation. High-load shards might be split into multiple shards, and persistently underutilised shards can be merged as long as sufficient capacity is available. [4]Execution and verification: The database management part implements the approved operation using the migration and consistency mechanisms of the underlying database system. It then verifies whether the records are available, whether the metadata is partitioned, and whether the queries are accessible, and completes the operation.

Another factor considered when deciding on migration is the cost of data migration. If the workload benefit is expected to outweigh the cost of transferring the workload to the destination node, and the destination node has the required capacity and governance, then the proposed rebalancing operation is performed. When it is not cost-effective, or the benefits are too small, placement remains unchanged, and continual monitoring is provided. This restriction helps minimise excess data transfers and the likelihood that rebalancing becomes a source of congestion in the database. The monitoring component monitors the performance of each rebalancing operation, including what data was migrated, migration time, impacted queries, and changes in node usage. These measurements can be used to evaluate the overhead of rebalancing and whether the improvement from the adaptive strategy is measurable compared with static partitioning. In addition, the framework includes monitoring and predictive analysis to identify potential performance issues earlier, before they become problematic for database operations. The system can automatically track query latency, node utilisation, and workload distribution, and take corrective measures by changing data placement and allocating resources accordingly. Real-time monitoring and automated management apply to distributed systems that need to optimise operation and reliability under fluctuating workloads (Gundla, 2025a, 2025b; Hariharan, 2024, 2025).

The framework is designed to help ensure that the database remains available when workloads change and/or nodes are interrupted. But migration or node-failure continuity relies on the replication, transaction management, and recovery mechanisms built into the database implementation. No architecture assumes that data movement is continuous or that eventual consistency is sufficient for immediate consistency. Rather, the implementation is required to remain consistent with the defined consistency requirements and ensure that affected records are available before performing a rebalancing operation.

The proposed solution is adaptive, handling distributed PDMP databases with workload forecasting, hierarchical partitioning, jurisdiction-aware query routing, and cost-aware rebalancing. It is assessed using the experimental methodology discussed in the next chapter,

which covers workload prediction, query performance, resource distribution, and the overhead of adaptive shard management.

Table 1 illustrates the four stages of adaptive rebalancing in the proposed system. It explains how workload imbalances are detected, evaluated, and addressed through appropriate decisions, followed by data migration and performance verification to maintain efficient database operations.

**Table 1. Four Stages of Adaptive Rebalancing**

| Stage | Description | Expected Action |
|---|---|---|
| **Detection** | Identifies overloaded or underused database shards. | Monitor shard workloads and detect imbalances. |
| **Evaluation** | Examines workload levels and resource usage. | Assess whether rebalancing is necessary. |
| **Decision** | Determines the best rebalancing action. | Decide whether to split or merge shards. |
| **Execution and Verification** | Carries out the changes and checks the results. | Move data, update shard distribution, and verify system performance. |

# 4. Methodology

## 4.1. Research Design

The study uses a quantitative Comparative Experimental approach to investigate the effectiveness of applying AI-powered database sharding to large-scale Prescription Drug Monitoring Program (PDMP) databases. The study compares traditional static sharding with an adaptive sharding approach that incorporates machine-learning-based workload forecasting, hierarchical data partitioning, and dynamic shard management.

It is assumed that there are two database configurations. The default system uses static geographic and hash-based partitioning, preconfigured geographic- and hash-based partitioning rules, and conventional query routing. It uses an Extreme Gradient Boosting (XGBoost) regression model to predict database workloads and help with shard placement and rebalancing. The database architecture, data structure, and definitions of queries and conditions for resource allocation are identical in both configurations. This means you will be comparing the effects of the sharding and workload-management strategies on the system, rather than the effects of different database platforms or experimental inputs.

Normal, increased, and uneven workloads are considered in the evaluation. It also seeks to understand and analyse local and cross-jurisdictional queries to ensure that distributed data retrieval does not reduce database responsiveness. Key areas of evaluation include query latency, transaction throughput, shard-load distribution, workload prediction accuracy, hotspot detection, and rebalancing overhead. The experimental design is used to assess database performance, not clinical effectiveness. Thus, this assessment centres on the efficiency of prescription data storage and retrieval, rather than clinical decision-making, overdose prevention, or prescription outcomes.

Table 2 highlights the differences between the baseline static sharding and the proposed AI-based sharding. It emphasises differences in data distribution, workload management, query routing, load balancing, monitoring, and rebalancing, demonstrating AI's role in enhancing adaptability, efficiency, and database performance under varying workloads.

**Table 2. Comparison of Database Configurations**

| Feature | Baseline Static Sharding | Proposed AI-Based Sharding |
|---|---|---|
| **Data Distribution** | Fixed distribution | Adaptive distribution |
| **Workload Management** | Manual management | AI-based management |
| **Query Routing** | Fixed routing rules | Intelligent routing |
| **Load Balancing** | Limited balancing | Automatic balancing |
| **Monitoring** | Basic monitoring | AI-based monitoring |
| **Rebalancing** | Manual adjustments | Automatic adjustments |
| **Performance** | May have delays | Aims for faster queries |

### 4.2. Experimental Environment

The experimental architecture is based on the distributed relational database platform PostgreSQL and the Citus extension. PostgreSQL offers relational data management, and Citus offers multi-node distributed table storage and query execution. The architecture has two components: a coordinator node and Worker nodes. The coordinator receives requests from the database, identifies the locations of the needed data, and passes them to the workers for execution in the distributed environment. Data partitions are stored on worker nodes, where database operations are performed. The default setting is based on default partitioning and data-placement rules.

The adaptive configuration uses the XGBoost workload-forecasting component and the shard-management mechanism as described in Chapter 3. Both configurations have the same amount of database resources and the same infrastructure conditions, so that the results can be compared. The adaptive architecture separates workload prediction from physical data management. The XGBoost component takes a set of workload observations from the past and makes a prediction for each shard for the next period. The shard-management component considers these forecasts alongside current resource usage, node capacity, jurisdictional limitations, and migration fees, and moves data from one shard to another only when it believes it is necessary.

The experimental environment is characterized by the following implementation components:

- PostgreSQL relational database management system.
- Citus distributed database extension.
- Coordinator and worker-node architecture.
- XGBoost workload-forecasting component.
- Database monitoring and workload-collection mechanism.
- Application-level query generator.
- Shard-placement and rebalancing management component.

Database indexing, connection management, caching, replication, and consistency settings are maintained consistently across the compared configurations. These settings are documented as part of the experimental configuration because they can independently influence database performance.

Figure 3 illustrates the experimental environment used to evaluate the proposed PDMP sharding framework. It shows the comparison between static and adaptive sharding configurations within a PostgreSQL-Citus distributed database. Synthetic PDMP data supports

testing, while performance evaluation measures latency, throughput, load balance, forecast accuracy, and system reliability.

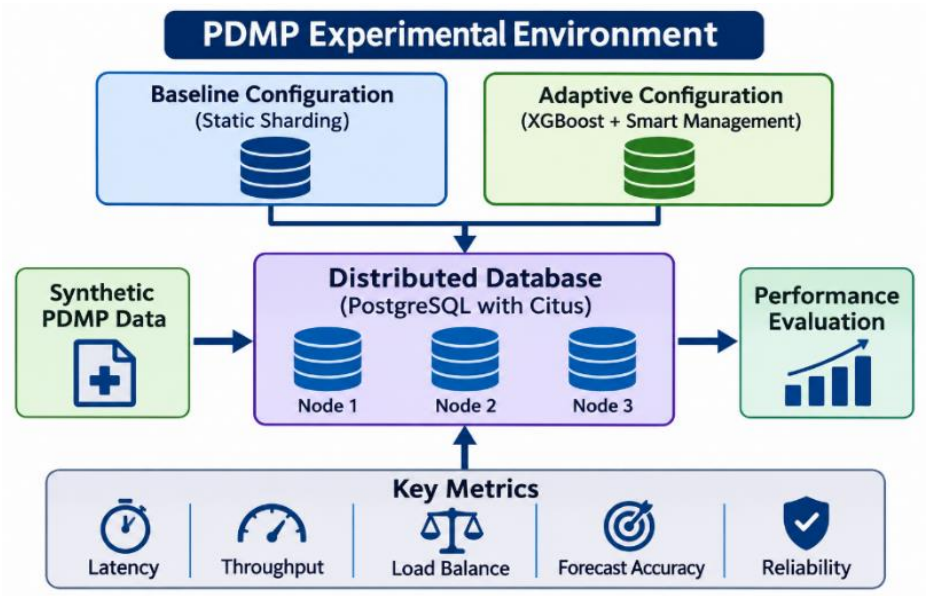



**Figure 3. Experimental environment used to evaluate the proposed PDMP sharding framework**

### 4.3. Data Collection and Dataset Preparation

The evaluation uses synthetic PDMP-like data to represent the structure and relationships of prescription-monitoring information. Synthetic records are appropriate for this evaluation because actual PDMP datasets contain sensitive patient and prescription information that cannot be assumed to be available for unrestricted experimental use.

The dataset represents the principal information categories identified in the proposed framework:

1. Patient prescription-history records.
2. Prescriber records.
3. Pharmacy records.
4. Drug classification records.
5. Prescription activity and interstate transaction records.

For each synthetic record, a set of attributes is required to facilitate storage, indexing, and retrieval in a database. These include: pseudonymous patient identifiers, jurisdiction identifiers, prescription timestamps, prescriber identifiers, pharmacy identifiers, medication classifications, and transaction attributes. The data is organised according to the hierarchical partitioning system set up in Chapter 3. Geographic jurisdiction is the main partitioning criterion, and patient identifiers are passed through a uniform hashing function to partition records between the shards within each jurisdiction. Prescription timestamps enable temporal indexing, and drug classification and pre-defined risk indicators are stored as metadata for searching.

These steps involve making synthetic records, adding jurisdictional identifiers, adding pseudonymous patient identifiers, creating links between prescription records and associated entities, and organising records to be stored in a distributed manner. It also measures differences in the number of records per jurisdiction. This enables workload balancing and allows the adaptive configuration to adjust to uneven data and query distributions. Records are

represented across jurisdictions rather than across patients through their individual "nationally shared physical record locations".

Data is generated as a dataset that differentiates between patient records, prescription records, prescriber records, pharmacy records, and transaction records. It also considers the distribution of records across jurisdictions and the relationships that must be established to support local and cross-jurisdictional queries. The observations are gathered using application-level logs and database monitoring. The following metrics are collected: shard workload distribution, Network activity, storage utilisation, memory utilisation, CPU utilisation, completed transactions, and Query execution time.

In addition, workload observations are collected over time to support XGBoost model development. Each observation is the workload and resource levels of a shard in a specified time interval during which it is monitored. These observations are used as input features and target values for supervised workload forecasting.

### 4.4. Experimental Workload and Scenarios

The experimental workload describes the types of database operations involved in managing information within the PDMP, such as prescription history retrieval, prescriber and pharmacy searches, access to historical data, and authorised cross-jurisdictional queries. The read and write operations are created separately so that they can have different resource requirements. The same query definitions and transaction distributions are retained in both configurations.

Five scenarios are studied: Scenario 1: Normal Workload. Both configurations are tested with normal transaction volumes. Scenario 2: Increased Workload. Compares and analyses performance under increased requirements, including latency, throughput, and resource usage. If possible, the workload is uneven across jurisdictions. Scenario 3: Uneven Jurisdictional Workload. Adds partitioned requests to specific partitions to test hot-spot detection and adaptive redistribution. Scenario 4: Cross-Jurisdictional Queries. Applies to consider the flexibility of distributed query latency and communication overhead for getting records from across jurisdictions. Scenario 5: Adaptive Rebalancing. Analyses redistribution frequency, data transfer volume, migration time, and changes in node utilisation. The conditions used in all scenarios are the same across all configurations. Multiple runs and recorded workload parameters allow consistent performance comparisons.

### 4.5. Machine Learning Model Development

The workload-forecasting part is based on XGBoost regression, which predicts demand for individual database shards over a short time horizon. For this reason, XGBoost is chosen because it is well-suited to structured workload features, such as numerical and categorical variables (Nangi & Obannagari, 2024).

The model takes as input the query volume, number of read and write transactions, CPU and memory utilisation, storage consumption, network activity, workload variation, and query frequency in the jurisdiction. Predicts the workload which should be experienced by each shard in the next monitoring interval. These forecasts can be used to identify overloaded and underutilised shards. Supervised learning is used to develop the models. To avoid information leakage, the historical observations are divided into three mutually exclusive sets: training, validation and testing.

Training data is used to assess the model's fit to the data, validation data to make adjustments to the model, and testing data to assess the accuracy of the model's predictions. Forecast accuracy is measured by the mean absolute error (MAE), root mean squared error

(RMSE) and coefficient of determination ($R^2$). There is no assumption that it will be accurate to a certain degree; it will be tested experimentally.

The final predictions are sent to the shard-management part, which compares the predicted demand with shard resource utilisation. XGBoost does not support automatic splitting, sharding or moving of shards. These operations still have predefined usage limits, hysteresis, destination capacity validation, and migration cost, preventing unnecessary database rebuilds due to mispredictions.

### 4.6. Key Performance Indicators

The evaluation is based on performance indicators: query responsiveness, transaction processing, resource distribution, workload prediction, and rebalancing overhead.

Table 3 summarises the key metrics that are used to measure the proposed AI-based sharding system. It breaks down each metric and how it's calculated: query speed, transaction capacity, workload balance, prediction accuracy and hotspot detection, rebalancing costs, and reliability.

**Table 3. Performance Metrics Used to Evaluate the Proposed System**

| Performance Metric | What It Measures | Evaluation Method |
|---|---|---|
| **Query Latency** | Time taken to answer database queries. | Measure query response time. |
| **Transaction Throughput** | Number of transactions completed per second. | Count successful transactions per second. |
| **Shard-Load Distribution** | How evenly workloads are distributed across servers. | Compare CPU, memory, and storage usage. |
| **Forecasting Accuracy** | How accurately AI predicts future workloads. | Compare predicted workloads with actual workloads. |
| **Hotspot Detection** | Ability to identify overloaded servers. | Measure correct hotspot identification. |
| **Rebalancing Overhead** | Time and resources required to move data. | Measure data transferred and migration time. |
| **System Reliability** | Ability of the system to operate without failures. | Record system errors, failures, and timeouts. |

#### *4.6.1. Query Latency*

Query Latency is the time it takes to send a request to a database and receive a full response. The latency of local queries and cross-jurisdiction queries is evaluated separately, and the difference between single-jurisdiction retrieval and distributed query execution is separated. The mean, median, 95th percentile (p95), and 99th percentile (p99) are used to summarise latency. These measures are used to give information on the average performance of the queries and the slow response times when the workload is increased.

#### *4.6.2. Transaction Throughput*

Transaction throughput measures the number of successfully completed database transactions per second. It is calculated as:

$$\text{Throughput} = N_c / T$$

Where:

- $N_c$ represents the number of successfully completed transactions.
- T represents the measurement duration in seconds.

The throughput measurements are compared across the baseline and adaptive configurations under equivalent workload conditions.

#### *4.6.3. Shard Load Distribution*

Shard-load distribution measures variations in resource utilization across database nodes. CPU utilization, memory consumption, and storage usage are monitored to identify overloaded and underutilized nodes.

The coefficient of variation is used to assess relative differences in node utilization:

$$CV = (\sigma / \mu) \times 100$$

Where:

- CV represents the coefficient of variation.
- σ represents the standard deviation of node utilization.
- μ represents the mean node utilization.

A lower coefficient of variation indicates more evenly distributed utilization, provided that average resource utilization and workload conditions are also considered.

#### *4.6.4. Workload Forecasting Accuracy*

To assess the accuracy of workload forecasting, the workloads predicted by the forecasting method are later compared with the observed workloads (Kim et al., 2022). Evaluation is carried out using MAE, RMSE, and $R^2$ to assess the size of the prediction errors and the model's ability to explain the variance in workload demand.

#### *4.6.5. Hotspot Detection Performance*

Hotspot detection assesses the shard-overload detection capabilities of the monitoring and forecasting components. Precision, recall, and F1-score are used to compare predicted hotspot conditions with observed workload conditions. These measures define the system's ability to detect overloaded nodes and avoid misclassifying them as hotspots.

#### *4.6.6. Rebalancing Overhead*

Rebalancing overheads are the extra resources used in the data redistribution step. The evaluation shows the amount of data transferred, migration duration, network usage, and differences in query latency during rebalancing.

#### *4.6.7. System Reliability*

The system's reliability is measured using failures to reply to queries, transactions, errors, timeouts, and node failures during assessment. If the database configuration permits, tests are conducted to investigate recovery and record availability in the event of node failure. Performance thresholds are evaluation criteria, not necessarily a PDMP requirement. All latency, throughput or availability goals that are claimed shall be backed by an appropriate operational requirement or be explicitly stated as a benchmark for the study.

### 4.7. Analytical Techniques

Descriptive statistics and comparative performance analysis are used to analyse the experimental results. Both mean and median values can be used to summarise query latency and throughput, and standard deviation and percentile measurements can be used to measure variation among queries and slower query responses. The two configurations are tested against a similar workload, emphasising query latency, transaction throughput, and node utilisation. Forecasting workload is assessed individually based on the MAE, RMSE and $R^2$. To measure prediction quality, hotspot detection is evaluated using precision, recall, and F1-score, which distinguish the effectiveness of the prediction from improvements in the database.

Time-series analysis involves examining how workloads, resources, and query performance change over time between monitoring intervals. The impact of rebalancing is measured by shard utilisation and rebalancing response time before and after rebalancing. If applicable, confidence intervals and statistical tests are used to test differences between configurations.

Last but not least, the additional computation cost of XGBoost inference, monitoring and migration is measured. This determines whether the likely performance gains justify the additional resource and operational expenses.

### 4.8. Data Governance and Experimental Limitations

The information used for the assessment is synthetic PDMP-like data, meaning there is no patient identification information or prescription histories that could be used to identify any patients. They are suitable for testing partitioning, workload distribution and query execution in a database without revealing any sensitive information.

Architectural constraints are still considered, including jurisdictional boundaries and Access permissions. Any queries and/or adaptive data placement should consider the authorisation conditions of the experimental environment. The results presented apply to the configurations and workloads simulated for the database.

The number of transactions, data types, scope, and many other operational needs of individual PDMPs can vary. So the findings may not be applicable at other locations throughout the country. Also, there is no confirmation that this evaluation is HIPAA- or state-compliant, or that it is superior in clinical results. Further evaluations are needed for these. The methodology can serve as a benchmark for the performance evaluation of conventional sharding and AI-enabled sharding, as discussed in the next chapter.

## 5. Evaluation Framework and Discussion

### 5.1. Evaluation Overview

This chapter discusses the performance of the suggested AI-based sharding mechanism for Prescription Drug Monitoring Programs (PDMPs). The proposed approach is evaluated by comparing it to the existing static sharding approach in a PostgreSQL-Citus distributed database system. Using synthetic PDMP-like data provides a controlled environment for studying PDMP performance while maintaining patient privacy. The evaluation metrics include query response time, transaction throughput, workload distribution, forecasting accuracy, hotspot identification, and rebalance overhead. These measures can be used to evaluate whether workload forecasting and adaptive shard management help make better use of database resources (Gadde, 2022).

### 5.2. Baseline Sharding Performance

By default, the partitioning is done with geographic partitioning and patient-identifier hashing. Geographic partitioning partitions data based on jurisdiction, and hashing partitions data based on shards within a jurisdiction. This helps bolster data locality and keep organisation data jurisdiction-specific. When the number of prescriptions and query workloads are not evenly distributed among jurisdictions, however, static partitioning can suffer from uneven workloads. A few pieces may receive more requests, while others don't. A few may receive far more requests than others, and some may receive none. Cross-jurisdictional requests may require multiple shards to communicate with each other, adding to the complexity of query processing. These restrictions are used to compare static and the proposed adaptive partitioning.

Figure 4 illustrates baseline sharding, where patient records are geographically partitioned by jurisdiction and distributed across shards using hashing. It highlights key limitations such as load balancing, slow response times, and complex queries across jurisdictions, which drive adaptive sharding methods.

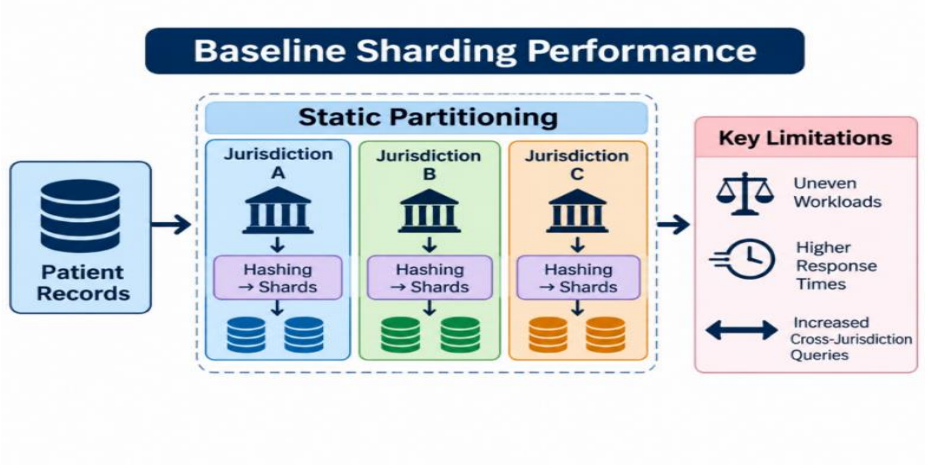



**Figure 4. Geographic sharding, data distribution, and baseline performance limitations**

### 5.3. Adaptive Sharding and Workload Optimisation

The proposed framework extends XGBoost regression to predict shard workload in the near future. The model takes into account statistical data on queries that have been executed in the past, resource consumption (CPU utilisation, memory usage, storage, read/writes), etc. These forecasts can be utilised to determine whether any shards might be overwhelmed and to decide where to place data. The adaptive approach differs from static partitioning because it considers workload trends to determine when to move shards and resources.

Monitoring information is used to supplement forecasts and indicate the real pressure on resources. This blend helps reduce the need to rely on a single workload observation for management decisions. The organising principle is geographic partitioning, and within each jurisdiction, patient identifiers are hashed. That said, temporal indexing is also incorporated to facilitate retrieval of historical data, and risk classification remains metadata, not a partition.

### 5.4. Query Routing and Cross-Jurisdictional Access

Query Routing determines how requests are routed to the correct database shards. The proposed framework considers data location, user permissions, and the request's jurisdiction. Local queries are sent to the appropriate jurisdictional shards, and authorised interstate queries may need to be coordinated across jurisdictions. Cross-shard queries incur extra processing and communication overhead. They may be impacted by network conditions, data distribution, query complexity, and whether relevant records are available. Caching can help to minimise the number of database accesses, especially for data that is accessed frequently. But there needs to be a proper mechanism for expiration and invalidation of cached records so as not to return outdated information. There should also be access restrictions for cached data and cross-jurisdictional requests.

### 5.5. Load Balancing and Rebalancing Considerations

Adaptive rebalancing aims to compensate for persistent workload imbalances between database shards. The framework is based on monitoring thresholds and workload forecasts that indicate when redistribution may be warranted. Hysteresis reduces the likelihood of re-deciding whether to migrate due to workload fluctuations. If a shard runs at a high workload for a long period, it can be subdivided and/or reassigned, depending on the database architecture. Where

appropriate, unused shards can be consolidated, both technically and operationally. When making rebalancing decisions, it is important to consider both the performance benefit that is expected as well as the volume of data to be moved, network usage, processing power, and temporary query disruption (Zhang et al., 2025). Therefore, migration should be considered only if the benefit outweighs the cost of the operation. This way, there is little unnecessary data transfer, and the redistribution is not likely to be 'unstable'.

### 5.6. System Reliability and Fault Tolerance

A distributed PDMP database should ensure reliable access during node failures, network outages, and other operational events. System resilience is provided by replication, monitoring, recovery and consistency settings. The proposed evaluation considers the impact of these mechanisms on database availability and recovery. Failures may include worker-node unavailability, loss of communication, temporary loss of access to specific shards, etc. Recovery performance should be evaluated in terms of consistency and resumption of normal query processing. Availability and consistency depend on the database configuration, replication strategy, and recovery plan, and should not be taken for granted. The following general fault-tolerance principles can be used to consider these conditions (Jhawar & Piuri, 2017).

### 5.7. Performance Evaluation and Interpretation

This framework is evaluated on various complementary performance indicators. Average and median query latency represent typical latency, while the 95th and 99th percentiles represent slower (but not necessarily the worst) requests. Transaction throughput measures how many transactions can be processed over a period of time. Shard-load distribution is the degree to which workloads are distributed across the database nodes. The forecasting error measures how well the XGBoost model performs, while hotspot precision, recall, and F1-score measure the XGBoost model's performance in identifying overloaded shards.

Data-migration volume, migration duration, and resources consumed are examined to assess rebalancing overhead. To make a fair comparison, both configurations must use the same hardware, dataset, database settings, and workload scenarios. Further experiments would confirm whether the observed difference is stable or transient.

### 5.8. Discussion and Limitations

The proposed framework includes workload forecasting, hierarchical partitioning, workload-aware query routing and rebalancing. These components address a range of PDMP aspects and continue to be based on the concept of geographic jurisdiction. But synthetic data doesn't have to be as complex as the live PDMP environment. The workloads are not the same, although the number of prescriptions, requests for cross-border prescriptions, user activity and user reporting may differ.

Forecasting models also must be regularly reviewed and retrained due to changes in workload characteristics. This framework should thus be considered as a database-management approach that needs to be empirically validated. The evaluation is not intended to support the product's effectiveness in a clinical setting, regulatory approval, or the prevention of prescription drug misuse. Further testing with representative workloads and accurate operational data should be performed before deployment.

# 6. Discussion

## 6.1. Impact on Prescription Drug Monitoring

To improve the scalability and responsiveness of Prescription Drug Monitoring Programs (PDMPs), one potential solution is to implement AI-based sharding. AI-based sharding is one possible solution to increase the scalability and responsiveness of Prescription Drug Monitoring Programs (PDMPs). Geographic partitioning, workload forecasting, and adaptive shard management are features of the framework that can help mitigate data imbalance and cross-jurisdiction query issues.

The recommended XGBoost model is used to estimate database workloads for shard placement and resource allocation. Jurisdiction-aware routing: routes queries to the right data source and enforces access control. However, cross-jurisdictional performance remains constrained by network, interoperability, and data-sharing policies.

Optimising databases so that their prescription records can be retrieved faster may aid in proper and timely clinical decisions, but will not prevent prescription overdoses or inappropriate prescribing. The results must be understood, along with the clinical implications and the actions that must be taken in response.

## 6.2. Advantages of AI-Based Sharding

AI workload forecasting allows for more flexible, dynamic database management than static partitioning. Geographic partitioning will be implemented to maintain data locality within each jurisdiction, and patient-ID hashing will be implemented to distribute data within each jurisdiction of the proposed design. Historical data can be retrieved using temporal indexing, and risk categories are metadata rather than physical shards.

Adaptive rebalancing addresses workload imbalances. If the benefits are greater than the cost of breaking and merging the shards, then the overloaded shards can be broken up into smaller shards, while the underutilised shards can be merged into one. Monitoring thresholds and hysteresis can prevent redistribution during temporary workload reconfiguration. These mechanisms should be evaluated on their query latency, throughput, resource consumption, forecasting accuracy and rebalancing overhead. Static sharding configurations need to be compared to quantify the benefit. Workload forecasting, geographic partitioning, and adaptive rebalancing are three approaches.

Figure 5 shows AI-based sharding techniques. It illustrates how to split overloaded shards, combine underutilised shards, improve database performance, optimise resource usage, reduce latency, and improve forecasting.

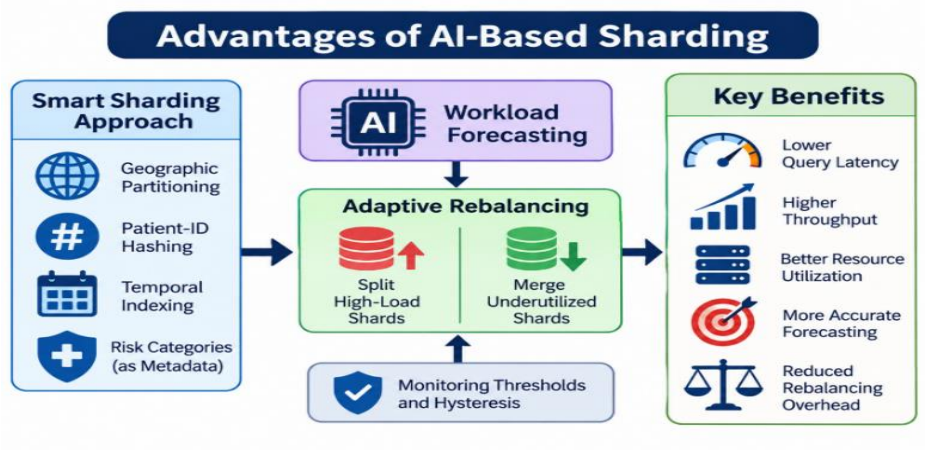



**Figure 5. Advantages of AI-based Sharding**

### 6.3. Privacy and Compliance

PDMP records include patient, prescriber, pharmacy, and prescription data. Therefore, appropriate security and governance controls are needed to manage these records (Oliva, 2019). Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC) provide access restrictions based on the role, jurisdiction and need for the request.

Data access, updates and administration activities should be documented in Audit logs. Information should be protected when captured, stored, and transmitted by using encryption, secure communication methods, identity verification, and key management. Access restrictions and data-sharing requirements should be maintained when performing shard migration and cross-jurisdictional queries. Federated learning can also support distributed analysis while keeping patient-level data decentralised. But these technical controls alone don't ensure compliance with healthcare privacy laws and jurisdictional agreements.

### 6.4. Limitations and Practical Implications

There are several restrictions to the proposed framework. First, it is evaluated using synthetic PDMP-like data, which may not fully reflect the distribution of data and workloads in operational settings. Second, forecasting accuracy depends on the quality of historical data and could decline if workload patterns change. Therefore, periodic model evaluation and retraining are important. Implementing cross-jurisdictional programs also requires interoperability, seamless data sharing, identity matching, and coordination among independently managed PDMP programs.

In addition, improvements in database performance should not be mistaken for improved clinical outcomes or regulatory compliance. To be used in practice, it must be tested with authorised operational data, undergo security assessment, have interoperability validated, and receive jurisdictional approval. The whole system is designed as a PDMP; it combines workload forecasting, hierarchical sharding, query routing and governance into a single database design. It needs to be tested empirically for effectiveness and suitability for nationwide implementation.

## 7. Recommendations and Future Scope

### 7.1. Practical Recommendations

The Prescription Drug Monitoring Programs (PDMPs) are recommended to use a hybrid sharding approach for large-scale PDMPs. Geographic partitioning should be the primary structure, with patient-ID hashing within each jurisdiction to better distribute data. Resources should be retrievable and stored with prescription records, and risk categories should not be separate physical shards. XGBoost workload forecasting should aid resource planning and help determine potential "hotspots". It is not a good idea to rebalance unless there is a long-term workload imbalance which warrants the effort of data migration. Shards can either be split up among multiple shards to handle high loads or merged into a single shard to handle low loads.

### 7.2. Governance Recommendations

Where information is sensitive, such as prescription information and data, effective governance is necessary. Access control should be role- and attribute-based, based on user roles, jurisdiction and purpose of access. All system activity, whether searching, updating, administration, or interstate data requests, should be logged to support accountability and auditing.

Sensitive records should be protected when stored, communicated, and transferred to other systems through encryption, tokenisation, secure communication, and appropriate identity and key management. Compliance should also be maintained for data sharing and

movement across jurisdictions, in accordance with laws, agreements, and access restrictions. These technical safeguards help to ensure security, but do not independently constitute compliance with regulation (Khatiwada et al., 2024).

### 7.3. Future Scope

Authorising data sharing while maintaining jurisdictional control is an opportunity for federated PDMP architectures that should be explored in future research. An alternative evaluation could use federated learning or privacy-preserving methods, such as secure multi-party computation, to enable analysis of cross-jurisdictional data without centralising patient-level information. Research should also focus on integrating workload forecasting and real-time monitoring, patient identity matching, and interoperability mechanisms. Prescribing patterns can be analysed with machine learning, but it is important to validate it separately as a method to detect abnormal activity and predict clinical risks. Last but not least, the proposed framework needs to be evaluated with larger, more varied datasets, typical workloads, and fault scenarios. Performance, migration cost, security, and interoperability should be evaluated through pilot deployments before wider deployment.

## 8. Conclusion

Based on this study's findings, an AI-supported sharding solution is proposed to address the scalability, workload imbalance, and data-access challenges faced by multi-jurisdictional Prescription Drug Monitoring Programs (PDMPs). It uses geographic partitioning and patient identifier hashing within jurisdictions, along with temporal indexing of historical records. Risk classifications are not partitions, but metadata. To identify hotspots and place shards, an XGBoost regression model predicts workloads. Requests are made locally and across states, and access is controlled by jurisdiction-aware routing and access controls; rebalancing is based on thresholds, and hysteresis and migration costs are considered to avoid unnecessary data movement.

The methodology proposes a way to compare by creating synthetic PDMP-like data in a PostgreSQL-Citus environment. These include latency, throughput, load distribution, forecasting accuracy, and rebalancing overhead. The framework is a means of assessing PDMP data management. Still, it has not been proven effective through operational measurement; further testing with representative workloads, data, and jurisdiction-specific governance requirements is needed before deployment. Database performance enhancements should not be considered evidence of clinical effectiveness, regulatory compliance, or overdose prevention.

**References:**

Chadha, K. S. (2025). Machine learning–augmented ETL pipelines for fraud-resistant insurance claims processing. *International Journal of Data Science and Machine Learning, 5*(1), 410–436. https://doi.org/10.55640/ijdsml-05-01-30

Chadha, K. S., Vennamaneni, P. R., & Sardana, J. (2025). Leveraging advanced analytics and AI-powered solutions for transforming healthcare and retail e-commerce: Enhancing data security, personalization, and compliance. *International Journal of Applied Mathematics, 38*(9s), 1310–1334. https://doi.org/10.12732/ijam.v38i9s.862

Dhanagari, M. R. (2025). Aerospike vs traditional databases: Solving the speed vs. consistency dilemma. *International Journal of Computational and Experimental Science and Engineering, 11*(3). https://doi.org/10.22399/ijcesen.3780

Durgam, S. (2025). Peer benchmarking systems for RIA performance evaluation in investment technology. *Computer Fraud & Security*. https://computerfraudsecurity.com/index.php/journal/article/view/785

Gadde, H. (2022). AI in dynamic data sharding for optimized performance in large databases. *International Journal of Machine Learning Research in Cybersecurity and Artificial Intelligence, 13*(1), 413–440. https://www.academia.edu/download/119017192/413_440_ijmlrcai_2022.pdf

Gundla, S. R. (2025a). AI-augmented testing: GitHub Copilot for JUnit/Mockito generation. *Computer Fraud & Security*. https://computerfraudsecurity.com/index.php/journal/article/view/784

Gundla, S. R. (2025b). PostgreSQL tuning for cloud-native Java: Connection pooling vs. reactive drivers. *International Journal of Computational and Experimental Science and Engineering*. https://ijcesen.com/index.php/ijcesen/article/view/3479

Hariharan, R. (2024). *API gateway threat prevention in large-scale applications*. SciPubHouse. https://scipubhouse.com/wp-content/uploads/2025/10/011-API_gateway_threat_prevention_in_large-scale_applications.pdf

Hariharan, R. (2025). AI-driven identity and access management in enterprise systems. *International Journal of Internet of Things*. https://www.academicpublishers.org/journals/index.php/ijiot/article/view/4300

Jha, C., Gannavarapu, P., & Durgam, S. (2025). *Machine learning-driven security system for next-generation ultra-low-latency financial network governance*. IEEE. https://ieeexplore.ieee.org/abstract/document/11374574

Jhawar, R., & Piuri, V. (2017). Fault tolerance and resilience in cloud computing environments. In J. R. Vacca (Ed.), *Computer and information security handbook* (pp. 155–173). Morgan Kaufmann. https://doi.org/10.1016/B978-0-443-13223-0.00009-6

Khatiwada, P., Yang, B., Lin, J. C., & Blobel, B. (2024). Patient-generated health data (PGHD): Understanding, requirements, challenges, and existing techniques for data security and privacy. *Journal of Personalized Medicine, 14*(3), 282. https://doi.org/10.3390/jpm14030282

Kim, I. K., Wang, W., Qi, Y., & Humphrey, M. (2022). Forecasting cloud application workloads with CloudInsight for predictive resource management. *IEEE Transactions on Cloud Computing, 10*(3), 1848–1863. https://drive.google.com/file/d/17MLIufoh5OVAlw3qu62qCF4jtMaiBCDW/view

Kumar, N. (2025a). Accessibility first design systems for multi-brand food apps. *American Journal of Technology, 4*(3). https://doi.org/10.58425/ajt.v4i3.457

Kumar, N. (2025b). Edge computing patterns for real-time order flow optimization. *International Journal of Applied Mathematics, 38*(12s). https://doi.org/10.12732/ijam.v38i12s.1494

Lingamgunta, R. K. K., Ubale, A., & Vanama, S. K. R. (2025). Edge AI for on-site health risk scoring: A RAG-enabled framework. *American Journal of Technology, 4*(3), 1–14. https://doi.org/10.58425/ajt.v4i3.451

Nangi, P. R., & Obannagari, C. K. R. N. (2024). High-performance distributed database partitioning using machine learning-driven workload forecasting and query optimization. *American International Journal of Computer Science and Technology, 6*(2), 11–21. https://aijcst.org/index.php/aijcst/article/download/132/117

Oliva, J. D. (2019). Prescription-drug policing: The right to health information privacy pre- and post-Carpenter. *Duke Law Journal, 69*, 775. https://scholarship.law.duke.edu/cgi/viewcontent.cgi?article=4007&context=dlj

Rangu, S. (2025a). Analyzing the impact of AI-powered call center automation on operational efficiency in healthcare. *Journal of Information Systems Engineering and Management*. https://www.jisem-journal.com/index.php/journal/article/view/8901

Rangu, S. (2025b). Enterprise digital transformation in financial services: Emerging trends and technologies. *Computer Fraud & Security*. https://computerfraudsecurity.com/index.php/journal/article/view/786

Samala, S. (2025a). Architecting multi-instance Jira deployments: Scalability challenges for global enterprises. *International Journal of Engineering Applications*. https://gprjournals.org/journals/index.php/ijea/article/view/353

Samala, S. (2025b). Reducing release failures by 35%: A case study on Jira–Jenkins–Azure DevOps integration. *Journal of Information Systems Engineering and Management*. https://www.jisem-journal.com/index.php/journal/article/view/8904

Sardana, J., Chadha, K. S., & Durgam, S. (2025). Advancing healthcare technology through secure data systems. *International Journal of Applied Mathematics, 38*(10s). https://doi.org/10.12732/ijam.v38i10s.1138

Ubale, A. (2025). Privacy-preserving federated learning frameworks for telematics data in auto insurance analytics. *International Journal of Artificial Intelligence, Data Science, and Machine Learning, 6*(4), 41–48. https://ijaidsml.org/index.php/ijaidsml/article/view/314

Vanama, S. K. R. (2023). Integrating site reliability engineering (SRE) principles into enterprise architecture for predictive resilience. *International Journal of Emerging Trends in Computer Science and Information Technology, 4*(3), 164–170. https://ijetcsit.org/index.php/ijetcsit/article/view/514

Vanama, S. K. R. (2024). AI-augmented CI/CD pipeline optimization for scalable cloud-native deployment. *International Journal of Artificial Intelligence, Data Science, and Machine Learning, 5*(4), 175–187. https://ijaidsml.org/index.php/ijaidsml/article/view/368

Vishwakarma, S. K. (2025a). Circular economy in aerospace: Recycling composites and rare metals. *International Journal of Management, Business, and Development*. https://aimjournals.com/index.php/ijmbd/article/view/102

Vishwakarma, S. K. (2025b). Sustainable aviation fuel (SAF) procurement challenges. *Journal of Information Systems Engineering and Management*. https://www.jisem-journal.com/index.php/journal/article/view/9420

Zhang, Y., Zhao, K., Yang, Y., & Zhou, Z. (2025). Real-time service migration in edge networks: A survey. *Journal of Sensor and Actuator Networks, 14*(4), 79. https://www.mdpi.com/2224-2708/14/4/79