UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Luis Gustavo Bornia

A Partition Methodology For ER Conceptual Schemas With Workload Considerations

Florianópolis
2026

Luis Gustavo Bornia

A Partition Methodology For ER Conceptual Schemas With Workload Considerations

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação para a obtenção do título de mestre em Ciência da Computação.
Supervisor: Prof. Ronaldo dos Santos Mello, Dr.

Florianópolis
2026

Ficha catalográfica gerada por meio de sistema automatizado gerenciado pela BU/UFSC.
Dados inseridos pelo próprio autor.

Bornia, Luis Gustavo
A Partition Methodology For ER Conceptual Schemas With Workload Considerations / Luis Gustavo Bornia ; orientador, Ronaldo dos Santos Mello, 2026.
60 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Ciência da Computação, Florianópolis, 2026.

Inclui referências.

1. Ciência da Computação. 2. Banco de Dados. 3. Persistencia Poliglota. 4. Workload. I. Mello, Ronaldo dos Santos. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. III. Título.

Luis Gustavo Bornia

A Partition Methodology For ER Conceptual Schemas With Workload Considerations

O presente trabalho em nível de mestrado foi avaliado e aprovado, em 25 de Novembro de 2025, pela banca examinadora composta pelos seguintes membros:

Prof. Carina Friedrich Dornelles, Dra.
Universidade Federal de Santa Catarina

Prof. Renato Fileto, Dr.
Universidade Federal de Santa Catarina

Prof. Vanessa do Lago Machado, Dra.
Instituto Federal Sul-Rio-Grandense

Certificamos que esta é a versão original e final do trabalho de conclusão que foi julgado adequado para obtenção do título de mestre em Ciência da Computação.

Documento assinado digitalmente
Carina Friedrich Dorneles
Data: 18/05/2026 17:20:37-0300
CPF: ***.834.250-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Carina Friedrich Dornelles, Dra.
Coordenadora do Programa

Documento assinado digitalmente
RONALDO DOS SANTOS MELLO
Data: 06/05/2026 09:26:53-0300
CPF: ***.693.660-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Ronaldo dos Santos Mello, Dr.
Orientador

Florianópolis, 2026.

Agradeço a todos que sempre estiveram ao meu lado durante todo o mestrado. Nos melhores e piores momentos, vocês estiveram ao meu lado, sempre acreditando em meu trabalho. A todos que participam e participaram do Projeto Ceos, em especial a todos os professores, que permitiram meu crescimento e desenvolvimento.Não esquecerei o que todos vocês fizeram por mimDeixo, também, meus agradecimentos aos amigos que me acompanharam neste caminho.À minha família, que sempre me incentivou a perseguir os melhores resultados que eu pudesse.Espero que tenham orgulho. Um agradecimento especial aos diversos professores, colegas, chefes e amigos que durante minha jornada acreditaram em meu potencial. Tenho muito orgulho de poder compartilhar minhas conquistas com vocês.Ao Meu Orientador, Prof. Ronaldo Dos Santos Mello, que desafiou-me e apoiou-me durante todo este processo.

ACKNOWLEDGEMENTS

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001. O presente trabalho também agradece ao Projeto CEOS, desenvolvido pela Universidade Federal de Santa Catarina (UFSC) em parceria com o Ministério Público de Santa Catarina (MP-SC), pelo auxílio, colaboração e disponibilidade de estrutura e dados para desenvolvimento da proposta apresentada neste trabalho.

ABSTRACT

Polyglot Persistence conveys the idea that a database should use the appropriate technology that can best deal with businesses and technical requirements. Since it is a topic that has gathered interest more recently, there are still very few proposals that tackle the difficulties of partitioning an ER schema, specially with considerations such as workload. Even with no clear definition, all proposals end up considering workload at some level. That makes the direct comparison between proposals a very delicate balance of interpretations and considerations. To implement our proposal, we first evaluate current workload definitions to find out how to best consider it in our partitioning proposal. We propose a formal definition for workload in order to unify the different considerations. We implement a definition in order to evaluate workload at the conceptual level and use it to implement a partitioning proposal. We evaluate its efficiency and whether the partitions make sense when we take planned operations into consideration. Our evaluation shows promising results, with reasonable partitioning and very few operations that are inter-partition.

Keywords: Polyglot Persistence. Workload. Estimation. Partitioning.

RESUMO EXPANDIDO

Persistência Poliglota (PP) é um conceito que propõe o uso da tecnologia de banco de dados (BD) mais apropriada para lidar com os diferentes requisitos funcionais, não funcionais e de regras de negócio necessários para o bom funcionamento da aplicação. À medida que domínios de dados se tornam mais complexos, um único modelo de banco de dados regularmente não é suficiente para atender a todos os requisitos necessários ao bom funcionamento do sistema em questão.No entanto, dividir um esquema Entidade-Relacionamento (ER) é uma tarefa complexa e frequentemente feita na base da tentativa e erro. Como um dos principais objetivos de tal particionamento é a melhora do desempenho do BD gerado e, consequentemente,da aplicação, um dos principais desafios nesse processo é a falta de uma métrica que guie o fatiamento.
Este trabalho propõe uma métrica para cálculo de *workload*, avalia a aderência da métrica proposta ao que pode ser observado em BDs complexos e, por fim, utiliza a métrica proposta para auxiliar o particionamento otimizado de um esquema ER de um domínio complexo.

Objetivos

O objetivo principal deste trabalho é propor uma metodologia de particionamento para esquemas conceituais ER com considerações de *workload*. Os objetivos específicos para obtenção do objetivo principal proposto são:

- Propor uma definição formal para *workload*;
- Propor uma métrica de cálculo para o *workload*;
- Realizar uma revisão de literatura e, por fim;
- Sugerir uma estratégia de particionamento de esquemas ER que visa implementar uma PP.

Metodologia

A metodologia do trabalho é dividida em duas frentes principais:A primeira é a Definição e Estimativa de *Workload*.Nesta parte da metodologia propomos a seguinte definição formal: ”o consumo total de recursos computacionais resultante da soma de todas as operações CRUD (Criar, Ler, Atualizar, Deletar) durante um período de tempo.” Com o formalismo proposto, passamos,então, à proposta de cálculo para a estimativa de *workload* de um certo esquema ER e, por fim, avaliamos a relação entre nossa proposta e o real comportamento de uma implementação do *benchmark* TPC-H.
A segunda parte da metodologia foca na estratégia de particionamento. Nesta parte, propomos uma estratégia de particionamento do esquema ER baseada na modelagem do esquema ER e das operações conhecidas que serão executadas no BD como um hipergrafo, onde cada entidade representa um nó e cada operação conhecida representa uma hiperaresta do hipergrafo. Para a representação matricial, são utilizados os relacionamentos das entidades (A), as entidades acessadas pelas operações (O) e a similaridade entre as operações calculada utilizando o Índice de Jaccard (S). Foi utilizado o algoritmo *Affinity Propagation* para implementação.

Resultados e Validação

A validação da metodologia foi feita em dois cenários distintos. O primeiro cenário avaliou o quão bem a estimativa proposta na primeira etapa representa o observado em uma implementação do *TPC-H Benchmark*. A métrica de workload foi utilizada para tentar prever o tempo de execução de consultas disponíveis no *benchmark* em questão, utilizando-se um modelo GAM (*General Additive Model*).O resultado mostrou significância estatística na previsão de execução, indicando que a proposta tem aderência e validade, embora a acurácia ainda possa ser aumentada com a adição de mais variáveis. A segunda avaliação focou na aplicação do particionamento proposto no mundo real. Para tal, o domínio escolhido foi o domínio do Projeto CEOS, uma parceria entre o Ministério Público de Santa Catarina (MP-SC) e a Universidade Federal de Santa Catarina (UFSC) para a análise e busca de indícios de fraudes em processos licitatórios estaduais. O algoritmo de particionamento recomendou partições lógicas coerentes (como a consideração de Notas Fiscais Eletrônicas (NFe) como uma partição separada do restante das entidades) e que permitem aos desenvolvedores uma melhor identificação de entidades que são consultadas em conjunto.

Conclusão

A dissertação conclui que a proposta de métrica de workload apresenta resultados positivos e pode ser utilizada para auxiliar o design de BD, especialmente em domínios complexos,oferecendo uma estimativa viável e reprodutível do desempenho observado antes que haja uma implementação física. A aplicação do *Affinity Propagation* como algoritmo para a estratégia de particionamento simplifica o processo por não necessitar que haja uma indicação prévia de quantas partições deverão ser retornadas ao final do processo e auxilia os modeladores a agruparem entidades de forma lógica, o que facilita a identificação de arquiteturas e tecnologias ideias para cada partição sugerida.

LIST OF FIGURES

CONTENTS

1 INTRODUCTION

Polyglot Persistence (PP) is not a novel concept. First mentioned in 2008, it adapts the principles of polyglot programming, which was introduced two years prior, to the database domain (LEBERKNIGHT, 2008). In its core, it is a simple idea: The usage of the right option for the right data. Only very recently PP researches have greatly increased in number and, even with more researchers looking into it, it still proves to be a challenge to recommend partitions and technologies without trial and error on a case-by-case basis: Modelers want to partition their DBs in a way they minimize inter-partitions queries to minimize workload.

The concept of Workload is closely related to Database Management Systems' (DBMS) operation. Without its consideration, there is close to no possible technique to improve any database (DB) performance in any way. Still, its definition is somewhat uncertain in the literature. Some authors utilize workload definitions that are very similar to what can be found in dictionaries, some authors rely on their readers' understanding of what workload is and how it is being utilized in their papers without explicitly defining it and some authors even define workload as a specified set of queries that are executed into their DB. The biggest group of researchers does not define workload, instead they simply present it as a variable *w*, with the sum of its concurrent executions as *W* (HUANG et al., 2024; AHMAD et al., 2011).

As *workload* is defined differently by different researchers, its definition varies from one paper to another, making it more difficult to properly compare results, proposals and methods. However, said definition seems to always be around a common understanding: Workload refers to what is executed at a DB.

Still, many questions still are open to debate: What exactly it refers to? Is it considered during a period of time? Does any operation in a DB count towards the workload? How are recurrent queries taken into consideration?

With such observation and with these open questions in mind, we propose the following definition to unify workload understanding throughout researches:

> *The total computational resource consumption of the sum of all CRUD operations during a period of time.*

To evaluate our workload definition, we test its accuracy by trying to predict execution time in an implementation of the TPC-H benchmark. We chose this form of evaluation because workload can be calculated and directly measured by expected execution time to determine its predictive capacity. After our evaluation, we utilize our workload proposal to implement a partitioning suggestion with workload consideration aimed at assisting modelers who want to implement a distributed DB, with special interest into PP. We finalize with the implementation of our proposal into a highly complex domain: The CEOS Project, where its domain encompasses the Brazilian

Procurement process,a domain with rules that are taken directly from federal, state and municipal laws. We show that, despite not penalized, our implementation provides a recommendation with very few inter-partition operations.

1.1 OBJECTIVES

Our main objective with this Dissertation is:
To propose a partition methodology for ER schema with workload considerations.
To attain our main objective, we planned the following specific objectives:

- To propose a formal definition for workload;
- To propose a novel workload estimation calculation;
- To perform a review on the state of the art of workload estimation proposals;
- To perform a review on the state of the art of polyglot persistence partitioning;
- To propose a partitioning strategy to aid DB partitioning, especially for Polyglot Persistence DBs.

1.2 CONTRIBUTIONS

This dissertation's main contributions are as follows:

- A novel approach for workload estimation;
- A methodology for ER schema partitioning with workload considerations

1.3 STRUCTURE

The remaining chapters of this dissertation are organized as follows: Chapter 2 presents our theoretical background that supports our proposal. Chapter 3 presents the current state of the art of workload estimation and DB partitioning, especially with workload considerations. It also discusses workload definition found in the literature; Chapter 4 presents our workload proposal, its definitions and calculations, as well as our partitioning strategy with workload considerations; Chapter 5 presents a validation implementation using TPC-H benchmark, with calculations for workload and a step-by-step proposed partitioning example; Chapter 6 implements both our proposals in a DB from a highly complex domain; Chapter 7 discusses our results, accuracy, precision, pros and cons of our proposals and provides an overview of our results; Finally, Chapter 8 concludes this dissertation and provide suggestion for future works as well as for other contributions.

2 FUNDAMENTALS

2.1 WORKLOAD

Workload is, conceptually, very clear and direct to understand. Its definition (as used in Computing) seems to be derived from its concept defined in physics. Most professionals and researchers understand what it means and what it represents, why it's important to understand it and what are its consequences. Still, its metrics are partial and vary according to its category: Database workloads are said to be I/O intensive; Real-time workloads demand very high bandwidth and CPU processing power; Batch-workloads can be CPU and memory intensive, and so on.

Still, when a paper proposes to utilize workload, it generally means that it has access to historical data and implements a machine learning algorithm to understand how operations and/or group of operations affect each others' performanceEven when that is not the case, there is rarely a metric that is utilized to predict/represent workload. Its representation appears more frequently as the processing power utilized during normal operation and, sometimes, peak consume during stressful operations.This can be observed in works such as (SHEIKH et al., 2011; CHEN; DAVOUDIAN; LIU, 2022) that define workload as the sum of concurrent operations at a particular moment. Contrary to (SEN; RAMACHANDRA, 2018), who does not present an explicitly clear definition throughout their paper and, instead, trust that the reader knows what workload is. Still, there are works like (MUELLER et al., 2013), that present a clear definition and explicitly references a dictionary definition. Although it does not explicitly state their understanding throughout their paper, it just mentions that the dictionary's definition is such. However, since the authors explicitly mention that their research does not focus on the workload, this form of definition presentation fully meets their needs. This is because the authors do not require a more precise workload definition.

Although conceptually very sound, a lack of proper definition also presents another challenge: Conflicting definitions between papers. Such example can be found throughout many comparisons, but we use as an example the definition of (KUNJIR et al., 2017), when compared with (SHEIKH et al., 2011).The first presents two workloads: ”Sales Workload” and the second,that is not explicitly defined. In contrast, the latter presents workload as ”the sum of zero or more instances of each query type”(SHEIKH et al., 2011). Its considerations are similar, however the former considers a discrete group of queries as an instance of workload while the latter considers that there is only one workload: The sum of all instances. It's not uncommon to find papers that consider workload a set of operations executed at a DB, while other papers consider workload a predetermined set of operations or even papers that consider workload as a user, operation pair. Even when papers share the same kind of definition, it is possible they

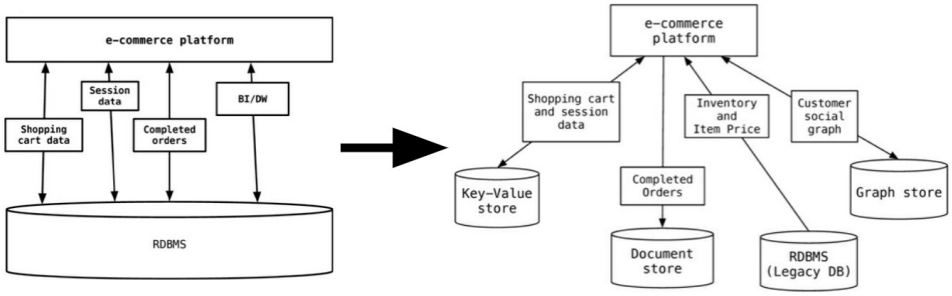


Figure 1 – Polyglot Persistence In An E-commerce Domain
source: (SADALAGE; FOWLER, 2013)

still have different considerations about workload. Hence,there is uncertainty around workload's definition.

2.2 POLYGLOT PERSISTENCE

Polyglot Persistence was likely first mentioned by Scott Leberknight (2008) in his text, which he mentions being inspired in the idea of polyglot programming (LEBERKNIGHT, 2008). Later, in their 2013 book, Sadalage and Fowler (SADALAGE; FOWLER, 2013) help to popularize its definition and present the most well known example of an application of polyglot persistence: the e-commerce example.

An brief explanation would be: Imagine you want to build an e-commerce DB. For this e-commerce, you must persist various types of data, with various different set of requirements: Shopping cart and session data need to be readily available, but hardly need strong consistency.It is not a major problem if some data is overwritten or if some products are not readily registered.It is bothersome, but it does not affect confidence in the businesses' quality; Unlike inventory, which requires strong consistency and should not ever let a customer buy a product that is not in stock; Recommendation engines, unlikely both aforementioned, needs a structure that allows graph traverses efficiently. It is possible to do so in RDBMSs but it would be more efficient to be done in a Graph DB.

If we take into consideration that an e-commerce has at least three opposing requirements that cannot exist under the same technology, which technology should said DB be developed into? Polyglot Persistence proposes that each "table" (partition, cluster, shard...) is allocated into a different technology that fits best its requirements instead of bending a DB or the data to meet all requirements (usually with lower performance than a specialized DB). Figure 1 presents the idea proposed by Polyglot Persistence applied into an e-commerce domain. In recent years, Polyglot Persistence has taken up interest of many researchers worldwide. Its proposal fits many domains better than a single DB, specially when a complex domain is in place, since more complex domains tend to have contradictory requirements for many entities in its modeling.

Still, the majority of current proposals are made in a case-to-case basis. Only one selection recommendation methodology was found in literature and, even with a structured recommendation, it requires a specialized professional to categorize every entity's 10 characteristics related to functional and non-functional requirements expected from said entity. It is a considerably burdensome task when a model has many entities (even though we only selected a small group of entities that are being used today, our implementation's current domain has 60+ entities).

Still, Polyglot Persistence is not an one-size-fits-all solution. It has to be made taking into consideration many diverse characteristics from data, operations, technologies, use cases, size, computing power and so on. At best, it can improve performance, correctly attend all functional requirements, partition data effectively and provide a reasonable,scalable solution to any domain necessary.At worst, it can hinder performance, increase complexity, create new incongruence and inflate storage requirements, as well as create a space where no information is trusted anymore.

2.3 GRAPH PARTITIONING

Graphs are widely utilized in simulation of grids, social networks, road planning and many other areas (ÇATALYÜREK et al., 2023). In its paper, Çatalyürek et al. present a comprehensive overview ofadvancements in (hyper)graph partitioning. Although (hyper)graph partitioning is a NP-Hard problem (GAREY; JOHNSON; STOCKMEYER, 1974), the development of more efficient algorithms help to maintain such partitioning practical. (Hyper)Graphs are usually found in diverse areas, one of which we highlight: *Distributed Databases and Query Optimization* (ÇATALYÜREK et al., 2023).

In the aforementioned area, proposals such as (QUAMAR; KUMAR; DESHPANDE, 2013; YANG et al., 2018; ATREY et al., 2020) utilize (hyper)graph partitioning to aid partitioning throughout machine clusters in order to minimize cost and optimize performance. With this focus in mind, we highlight SWORD, a scalable workload-aware data partitioning and placement approach for OLTP workloads (QUAMAR; KUMAR; DESHPANDE, 2013). To achieve their proposal, the authors utilize a simple two-step hypergraph compression technique. Such proposal allows them to group partitions in a way that each compressed hyperedge represents a *shard* of their distributed DB. The authors then partition the compressed hypergraph considering it a *k-way balanced min-cut partition problem*, with its balance constraint being the number of distributed transactions. The authors achieve better fault tolerance, throughput, lower response times and diminished data movement.

Similarly, proposals such as (LI; YU; KOUDAS, 2021), that utilize said (hyper)graphs to propose a more efficient solution to the *set similarity search problem*, shows (hyper)graphs as a very common data representation in data-related proposals,

included but not limited to incomplete information filling, data extraction from graphs and temporal graph patterns (FAN, 2022).

2.3.1 K-Way Partition Problem

The k-way partition problem is defined as the problem to find a balanced *k-way* partition of a (hyper)graph such as said partitioning minimizes a given objective function over the *k* partitions (ÇATALYÜREK et al., 2023). The authors mention specifically *k-way partition problem* as a common problem found in (hyper)graph partitioning. Such partition is most commonly solved using an approach based on the *Recursive Bisection Paradigm* (KARYPIS; KUMAR, 1999). Albeit most common, many paradigms have been proposed in the last ten years, with sequential and parallel proposals, such as *Multilevel Paradigm*, *Label Propagation*, *Fiduccia-Mattheyses Algorithm* and *Deep Multilevel Partitioning* (ÇATALYÜREK et al., 2023).

2.4 AFFINITY PROPAGATION

Affinity Propagation (FREY; DUECK, 2007) is a clustering algorithm proposed in 2007. Differently from *k-centers*, it works by passing messages between data pairs in order to find how well the one data point is an exemplar of another. An exemplar data point can be considered as the center of a cluster since data points are grouped around their exemplar. Instead of searching randomly for a good match, Affinity Propagation takes a structured approach where it evaluates recursively all options. Such strategy provides consistent results at the cost of performance, hence its performance is noticeably slower than *k-centers* for bigger datasets. In its core, Affinity Propagation works by passing messages between data pairs and it relies on three main metrics: i) *availability*; ii) *responsibility* and; iii) *similarity*.

Similarity ($s(i, k)$) represents how well point *i* is represented by point *k*. Strictly speaking, similarity is the main function which will be minimized. If, as an example, our final objective is to minimize the squared error ($(x_i - x_k)^2$), then similarity would be negative square error ($-||x_i - x_k||^2$). Similarity is also utilized in *responsibility's* calculation.

Responsibility ($r(i, k)$) refers to the message passed from point *i* to point *k* and it refers to the accumulated evidence point *k* has to be considered an exemplar of *i*. Its update follows the rule:

$$r(i, k) \leftarrow s(i, k) - \max_{k' \in \{k' \neq k\}} \{a(i, k') + s(i, k')\} \quad (2.1)$$

Availability ($a(i, k)$) refers to the "answer" sent from point *i* to point *k* and represents the accumulated evidence of point *i* being a proper exemplar of point *k*. Both

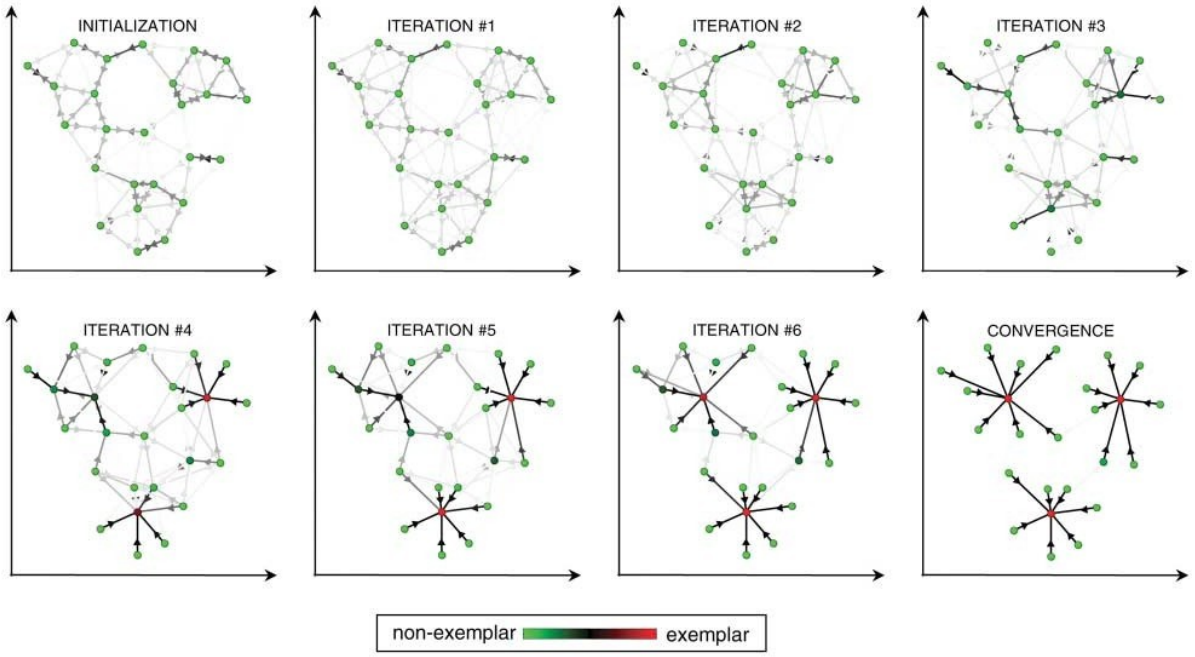


Figure 2 – Affinity Propagation Steps
source: (FREY; DUECK, 2007)

availability and *responsibility* values can be interpreted as the log-probability ratios of their proper representative aspects. Similarly to *responsibility*, it is updated following the rule:

$$a(i, k) \leftarrow min\{0, r(k, k) + \sum_{i' / \{i, k\}} \max\{0, r(i', k)\}\} \quad (2.2)$$

However to calculate the ”self-availability” ($a(i, i)$), the following formula is utilized:

$$a(k, k) \leftarrow \sum_{i' / \{i, k\}} \max\{0, r(i, k)\} \quad (2.3)$$

Through as many interactions as necessary, the algorithm will increasingly cluster data points around an exemplar that they show the most affinity towards. Figure 2 shows an example of how affinity propagation works.It utilizes an example where its main objective is to minimize the Euclidean distance.

3 STATE OF THE ART

In order to compare how this proposed workload estimation and partition strategy are inserted into current researches, we performed a Systematic Review of Literature (KITCHENHAM et al., 2009). Our main objective with such review was to answer the following questions:

Q1: How is Workload defined in literature?

Q2: How is Workload calculated in literature?

Q3: Which strategies exist for partitioning DBs?

Q4: Which of aforementioned strategies can be applied before implementation?

Q5: Which strategies consider Workload? How?

To answer the following questions we performed a search in different databases, namely: IEEE Xplore, ACM Computer Surveys, Web of Science,Elsevier and Google Scholar. We broaden our research to encompass as many articles as possible since our previous attempts with more strict definitions returned very few papers. We found the most results with the following search string: ”Database” AND (”Workload” OR ”Polyglot Persistence”) AND ”Modeling”. We updated our filter in January 2026 to include papers published until December/2025. To further improve our selection, we defined inclusion and exclusion criteria in order to only select the papers with relevant content related to our proposal. The inclusion criteria were:

- Paper written in English;
- Paper available for Download;
- Paper related to Computer Science;
- Paper must be focused on or use workload/polyglot persistence.

Exclusion criteria were the following:

- Paper Not available;
- Paper published before than 2010;
- Paper written in language other than English;
- Paper focused on different area/only briefly mentions workload/polyglot persistence.

Table 1 – Summary of Papers by Criteria

Filter	Total #
Pre-2010	555
Title Excluded	1783
Unavailable	30
Abstract Excluded	34
Full-Article Excluded	53
Included	72
Total	2527

After de-duplication, we listed 2527 papers to evaluate. Our first filter was related to publication date. We opted to exclude paper published before 2010 to leave a considerable margin to find older publications proposals that might have interesting contributions towards our proposal. 555 papers were published before 2010, hence, they were eliminated. From the remaining papers, 30 of them were not available for download. Upon reading papers' titles, we excluded 1783 papers. We highlight that in this group we found many papers that were not focused on workload nor polyglot persistence,as we found many Healthcare papers,who utilize *workload* as a metric of labor demand, specially for healthcare professionals. The remaining papers had their abstract read and evaluated, which led us to exclude further 34 papers. Lastly, as our last filter, we read the entirety of the papers and excluded 53 of them. Finally, we included a total of 72 papers into our review. Table 1 presents the summary of our filtering.

To better explain both workload and polyglot persistence proposals, we present them separately. Chapter 3.1 presents a review of workload proposals and chapter 3.2 presents a review of polyglot persistence proposals and researches.

3.1 WORKLOAD

Workload-related proposals can be observed by many different perspectives. They can be categorized by the *purpose* of the proposal (prediction, pattern mining, workload-driven partitioning etc.); *technique* (Statistical/Algorithm vs Machine Learning), *type* (Static vs Adaptive) and so on.

Lima and Mello (LIMA; MELLO, 2015) utilize a workload-driven design approach to recommend a more efficient database design. The proposal focuses on NoSQL Document Databases.The proposal also is referenced by other papers who focused on Document databases recommendations such as (RENIERS et al.,2020; HEWASINGHAGE et al., 2023).

Reniers *et al.* (RENIERS et al., 2020) proposes a workload-driven document database schema recommender. Although the authors mention it being workload-aware. Its workload consideration is limited to query plans and structure, which does not allow

for partition/grouping/balancing based on performance.

Similarly, Hewasinghage *etal.* (HEWASINGHAGE et al., 2023) proposes a prototype that recommends database design for document stores with multi criteria optimization automatically. The authors utilize workload as a set of design-independent queries with frequency.

On another topic related to workload is the prediction of completion time of queries (or sets of queries).In this topic, papers mainly try to predict completion time of queries, or batch of queries. Papers who research into this topic utilize a variety of strategies and variables.

Ahmad *et al.* (AHMAD et al., 2011) propose a prediction estimation based on interaction of query mixes. The proposal considers workload as a set of queries. The authors do not formally define workload. The authors conclude by assessing its performance and identifying that its accuracy as very high.

Cruz *et al.* (CRUZ et al., 2013) propose MeT, a prototype framework to control scalability in cloud-based NoSQL databases. The authors consider workload as a predetermined set of queries.The authors consider that it can exist multiple workloads concurrently in a database, as they consider it to be representative of a multi-tenant database.

With such variety of topics, it is no surprise for a categorization for workload utilization in papers to exist. However, no previous study or related survey seems to categorize the way workload is formally represented and calculated for DBs before implementation. In order to fill this gap, we propose a categorization with this focus as follows.

Category I: Papers classified in this category define a fraction of workload (w, also sometimes called ”workload” by the papers) as queries per period or (query,user) pairs. The sum of all parts (W) is, then, the workload of their DBs in a period of time. We classified 10 papers in this category.To be classified in this category, a paper must consider workload as a variable that represents the sum of all operations executed or expected to be executed, preferably during a period. Researches that utilize this approach generally fall into one of two subcategories: i) Workload is the sum of all executions during a period or; ii) Workload is the sum of all executions.

Category II: Our second category encompasses papers that utilize workload yet without explicitly defining what workload is. Papers in this category usually rely on commonly perceived characteristics of workload to implicitly mention what they consider. We classified 16 papers into this category.

Category III: This category encompasses papers that do not fit into earlier categories.This is supposed to be, as of now, a catch-all category. We classified 3 papers into this category. As a general rule-of-thumb, when a paper is into this category, its consideration of workload strays from dictionary-like definition.

Table 2 presents the papers found in literature, which category they fit them

Table 2 – Papers Found in Literature

Papers	Category	Phase
(LIMA; MELLO, 2015)	I	Modeling
(CHEN; DAVOUDIAN; LIU, 2022; PAUL et al., 2021; AMJAD et al., 2020; FARIAS et al., 2016; CRUZ et al., 2013; DUGGAN et al., 2013; TELANG; CHAKRAVARTHY; LI, 2013; AHMAD et al., 2011; HALFPAP; SCHLOSSER, 2019; DEEP et al., 2020; LASO et al., 2025)	I	Production
(HEWASINGHAGE et al., 2023)	II	Modeling
(SANTANA; MELLO, 2019; LUEBCKE; KOEPPEN; SAAKE, 2013; HUANG et al., 2024; WU et al., 2013; PADIYA; KANWAR; BHISE, 2016; CHOI et al., 2022; ZHOU et al., 2025)	II	Implementation
(SHAHEEN; RAZA; MALIK, 2018; RAZA et al., 2019; SEN; RAMACHANDRA, 2018; CUBUKCU et al., 2021; PERALTA et al., 2020; DANAOUI et al., 2024; DUGGAN et al., 2011; DOMASCHKA et al., 2023; GE; CHAI; CHAI, 2021; NGUYEN et al., 2025)	II	Production
(MUELLER et al., 2013; KUNJIR et al., 2017; CRUZ et al., 2013)	III	Production

into and at which phase (modeling, implementation or after-implementation) workload/workload estimations can be obtained.

As can be seen in Table 2, there are few papers who propose implementations at modeling phase. Only Lima and Mello (LIMA; MELLO, 2015) and Paul *et al.* (PAUL et al., 2021) provide workload estimation proposals who can be measured at modeling phase (before a physical DB is implemented). It is also important to notice that no paper focus on workload definition. All of them apply a particular definition throughout their paper. We believe a clear definition will help to compare results between different papers that might have used different metrics/definitions otherwise.

3.2 POLYGLOT PERSISTENCE

Even though it was first mentioned in 2008, the interest in such a topic has not shown much growth until very recently, when we can see that there is a sharp increase in appearance of papers related to PP proposals. Through our review, we only found 17 papers that research within the area of PP. Table 3 presents related articles and how they compare to our proposal

As can be visualized in table 3, papers who research over polyglot persistence proposals do not converge to the same topics and, even when researching the same topic, workload considerations are still very limited. Only one paper utilizes some

Table 3 – Polyglot Persistence Papers

Paper & Year	Uses Workload	Implements
(SRIVASTAVA; SHEKOKAR, 2016)	No	[X]
(SELLAMI; BHIRI; DEFUDE, 2016)	No	[ ]
(SANTANA; MELLO, 2019)	No	[X]
(KHINE; WANG, 2019)	No	[ ]
(POKORNY, 2019)	No	[ ]
(KOLONKO; MUELLENBACH, 2020)	No	[ ]
(KOšMERL; RABUZIN; ŠESTAK, 2020)	No	[ ]
(SINGHAL et al., 2021)	No	[X]
(FARAJ, 2022)	No	[ ]
(CALATRAVA; BECERRA; CUCCHIETTI, 2022)	No	[X]
(ROY-HUBARA; SHOVAL; STURM, 2022)	Yes	[X]
(CANDEL; RUIZ; GARCIA-MOLINA, 2022)	No	[ ]
(LAJAM; MOHAMMED, 2022)	No	[ ]
(KAZANAVICIUS; MAZEIKA; KALIBATIENE, 2022)	No	[X]
(PEREIRA; GUERRA; ROSA, 2023)	No	[ ]
(SACHDEVA; VASAVA, 2024)	No	[X]
(MELLO et al., 2024)	No	[ ]
Our Proposal	Yes	[X]

type of workload consideration that resembles a workload definition (ROY-HUBARA; SHOVAL; STURM, 2022). The other papers either do not even mention it or fall into what we defined as Category III in Section 3.1.

Roy-Hubara et al. propose a structured method for partition and selection of database technologies for a PP application (ROY-HUBARA; SHOVAL; STURM, 2022). The method considers Functional Requirements, Non-Functional Requirements as well as data-related characteristics. Although it considers workload, it does it in a indirect form. Instead, it compares various characteristics from each entity (i.e.volume of data) and the requirement to each entity in a model (i.e. required consistency) with DBs strong aspects to suggest which technology is the best fit for a group of data. Such process has to be made by an specialized professional, with domain knowledge and, for bigger domains (with many entities) it can be a challenging task.

Other uncertainties in such partitioning strategy relate to operations that might interact with entities with different requirements. If, as an example, Operation *A* queries over Tables *x, y, z*, but table *x* requires strong consistency and table *y* does not, should they be materialized into different technologies? If should not, is it better to select strong consistency or eventual consistency? Does that selection differs if *z* requires strong or eventual consistency? Are those consistencies dependent on operations that will be performed into them? If so, is there a way to not overwork domain experts to perform the best selection?

Our proposal tries to solve such problem by grouping together entities that share operations in order to minimize inter-technology operations that could increase development complexity and execution time. Our proposal also tries to address the workload consideration by utilizing an approach that focuses on trying to partition it as if it was a hypergraph. Our proposal is detailed in the next chapter.

Our categorization provides a comprehensive overview of workload definitions in the literature. Although we identified a small number of papers addressing polyglot persistence,only one proposes a methodology comparable to ours.Regarding Q1 and Q2 (workload calculation), we observed that workload is rarely explicitly calculated; instead, it is often merely acknowledged as ’existing’ or ’important.’

As for Q3, we found that partitioning, especially for Polyglot Persistent DB, was almost always performed on a case-by-case basis. These implementations generally lacked descriptions or indications of generalizations applicable to other domains. The only exception found was the proposal by (ROY-HUBARA; SHOVAL; STURM, 2022), which provides a methodology to identify the number of partitions, group tables, and select the best-fitting technologies.

Q4 addresses two aspects of our review: workload estimation and DB modeling. The first part (estimation before or during implementation) is answered by Table 2. The second part (applicability of the methodology) is answered by Table 3, with the caveat that adaptation may be required to handle different data types (e.g., converting quantitative metrics to qualitative inputs). Finally, Q5 is also answered by Table 3. Similar to Q3, this is addressed only by (ROY-HUBARA; SHOVAL; STURM, 2022), who utilize a qualitative approach to workload (classifying it as ’high,’ ’medium,’ or ’low’) rather than using specific quantitative units.

4 PROPOSAL

We propose a formal definition and formalism for calculations related to workload implementations or considerations, related to dictionary's definition, with the aim to unify its understanding in literature and allow for a simpler comparison between methods and models. And, through the aforementioned workload calculation, we propose a partitioning strategy for conceptual schemas that takes workload in consideration.

Since actual, real world workload, is influenced by hardware-, data- and software-related characteristics, our initial proposal does not consider all characteristics as it is simply not feasible since hardware characteristics are very unique to each own hardware, having even considerable variance even within processors of the same model. We decided to leave out of consideration any hardware- and software-related characteristic as of now and focus on three specific data-related characteristics and one time constraint. The characteristics we chose are: i) Frequency; ii) Queried Volume; iii) Complexity. We present their proposed formalism and description in section 4.1. We present calculations in section 4.2.1.

4.1 WORKLOAD DEFINITION

Making a better definition of workload is a considerable contribution to enable an effective evaluation of results and proposals, specially when comparisons between papers are necessary or wanted. As introduced before, our proposal is to define workload as follows.

Definition 1. Workload is defined as: *The Total Computational Resource Consumption of the Sum of All Operations During a Period of Time*

We clarify that *operation* means any CRUD (Create, Read, Update, Delete) operation executed or planned to be executed during a period of time. Our definition tries to adjust to necessities and generalize all considerations in papers with dictionary's definition, as it is crucial to understand how much resource can – and should – be allocated to a specific operation or group of operations for them to be able to execute in a timely manner and concurrently without significant delay to final users.

4.2 WORKLOAD ESTIMATION

Our proposal also includes a metric for workload estimation that considers *query frequency, complexity* and *accessed data volume* in order to quantify expected workload of an application in any stage of the DB design. The metric aims to, at first moment, indicate which operations are expected to take more time to execute and how an opera-

tion is expected to impact DB's performance in the future. We formalize our workload estimation calculations in the following subsection.

4.2.1 Proposed Estimation Formalism

We first define *Complexity, Frequency,Queried Volume* and *Period*. We formalize calculations after all definitions and, in Section 5, we present an example of calculation.

Complexity identifies how much a DBMS has to work to deliver the result as the user requested, with higher numbers representing more complex operations. We propose an extension of the complexity ranking that was proposed by Roy-Hubara, Shoval and Sturm in (ROY-HUBARA; SHOVAL; STURM, 2022). Our proposal is as follows:

1 Simple SELECT Function, with a filter (WHERE) or LIMIT functions;

2 Simple SELECT Function, without any filter or LIMIT;

3 Aggregate Functions (GROUP BY, MAX, MIN, AVERAGE...);

4 Joins, except FULL OUTER/CROSS join, with filter or LIMIT;

5 Joins, except FULL OUTER/CROSS join, without filter or LIMIT;

6 FULL OUTER/CROSS joins, with or without filter;

7 Graph Traverses and Recursive Queries.

Each operation, including subqueries, receives a complexity level based on what is the highest complexity structure inside it. A single operation must have one and only one complexity level. All levels are available to any operation, hence our definition is as follows.

Definition 2. $Complexity \in [1, 7]$

We propose this extension based on our experiments that showed our proposed complexity to be statistically more suitable than the original proposed by the work of (ROY-HUBARA; SHOVAL; STURM, 2022). Our reason for complexity as a variable was that more complex operations (i.e. Operations that require a more complex interaction, such as Joins, Aggregations etc.) require, naturally, more time to execute, even though it might not be immediately clear by reading the operation: If an aggregation is required, then it is necessary to group data, to maintain control of Counting, Sum, Average or any other aggregation operation. In any form, there is a requirement of additional steps that may not be clearly understood by just direct evaluation of said operation. Since (ROY-HUBARA; SHOVAL; STURM, 2022) proposed and utilized their metrics of complexity, we reproduced it with a simple addition that unrestrain the

most common operations over whether there is a limiting factor of the result size or not (in SQL, it was decided to use the LIMIT clause).

Period, or *Period of Time*, indicates a window of time in which all considerations will be based on. Our proposal, for simplicity, is to use a default of one day (1d) but a different period can be defined by the modeler to best fit their requirements. It is imperative that all values are calculated with the same period. Formally, we define *period* as follows.

Definition 3. *Period* = Δt

Our reasoning derives from our observations during our review, specifically with papers that fall into our Category I presented in 3.1. As we mention, papers within Category I fall into two interpretations, being the second one,*"workload is the sum of all executions"*, that is tackled. Without a clear understanding of the time where such sum of executions occur, one might judge such sum encompasses every execution during the lifespan of said DB. Such uncertainty leaves margin for different interpretations such as whether DB lifespan is considered from creation until current time, or lifespan must encompass future usage and, if so, how to consider future utilization? No matter the answer, to compare different interpretations of workload, a relative metric has to be utilized. Hence, we propose such problem to be solved by the addition of a clear indicative that time is important while also leaving enough room for adaptability for DBs that might require different periods.

Queried Volume represents the total amount of data an operation needs to access in disk or memory to perform the requested operation. Its proposed calculation takes in consideration the number of entities accessed (k),the number of occurrences in an entity $t \in k$, represented by O_t, as well as the number of attributes in that entity A_t. We propose the sum of all occurrences times all attributes as it is the most direct calculation possible. If a DB design intends to use a nested attribute, all its lowest levels should be considered a single attribute, i.e., a nested structure as *{"id":123,"Item":{"Price":14.56,"Description":"T-Shirt"}}*, is considered to hold three attributes: *id*, *item.price* and *item.description*. Our proposed formalization is then the following.

Definition 4. *AccessedVolume* $\sum_{k=t_1}^{t_n} O_k \times A_k$

Our reasoning for this parameter is straightforward:When an operations queries 10 thousand occurrences,we expect a lower time if said operation executed through, say, 10 million occurrences.However, even if said operation is expected to return only one attribute, it queries the whole occurrence and select only one attribute to return. Since the data is effectively queried, we evaluate based on existing attributes only. A possible critic over our choice is that, when observed in real DBs, said differentiation (return one attribute *vs* return all attributes) has different total execution time. To that

critic we answer that, in fact, that happens but we considered said difference to be a constraint on Read/Write on disk, which we do not take into consideration.

Frequency is the *Number of CRUD operations per period*. It is defined as how many times an operation *O* is executed at the DB during the previously defined period.

Definition 5. $Frequency_O = \sum Executions_O(Period)$

We reason that *Frequency* captures the quantity of an operation occurs during the period of time defined previously.It is relevant to do it because, even if an operation is considerably light, if said operation is frequently repeated, it will have a bigger impact on final DB performance than a heavier operation that occurs less frequently.

Taking 2 and 4, we are able to estimate the impact of an operation *O* at design phase. We propose its calculation to be made as follows.

Definition 6. $OperationImpact_O = Complexity_O \times AccessedVolume_O$

Based on equations 5 and 6 and on our proposed definition, we propose the following formula to calculate the workload, specially before implementation of any DB is as follows.

Definition 7. $Workload = \sum_{i=O_1}^{O_n}(Frequency_i \times OperationImpact_i)$

with $O_{[1,..n]}$ being the list of operations.

4.3 PARTITIONING STRATEGY

To obtain an effective partitioning, it is imperative to identify operations that have similar access patterns (operations that access the same entities or entities closely related). Even though such identification could be of great contribution to a partitioning strategy, its utilization is unrelated to other characteristics, such as the cost of such joins. An immediate advantage of such identification is the ability to use such knowledge to minimize cross-technology joins, which are costly operations. We model such problem as a hypergraph clustering problem, where its operations (hyperedges) are to be grouped in an optimal number of clusters.

With our proposed workload estimations, we are now able to estimate partitions based on operations' patterns and workload. Such partitions are suggestions based on estimations and can be further grouped, if further considerations indicate the need to do so. For example, if multiple partitions require strong consistency, it is natural a modeler would group them in a RDBMS.

As defined, a partition is a division of a set of objects into a family of subsets that are mutually exclusive (there are no elements present in more than one subset) and jointly exhaustive (the sum of all subsets contain all the members of the original set) (BRITANNICA, 2025). Mathematically, a partition *P* of a set *X* is a set of non-empty

subsets of X such that each and every element $x \in X$ is in exactly one subset. To clarify, P is a partition of X if and only if each and all element x is in exactly one subset.

With a list of operations and workload related to that operation, we propose an approach based on *graph representation* and *entity clustering based on entity accesses*. Our approach leverages workload and operations' patterns to execute an *Affinity Propagation* clustering to identify how many clusters should be created and which entities should be grouped (FREY; DUECK, 2007).

Consider q an operation planned to be executed in a conceptual data model. The operation q requires accesses to entities $e_1, e_2, ..., e_n$ and has a predicted workload of w. It is natural to assume all entities $e_1, ..., e_n$ can be represented as a graph (G), with its nodes being entities $e_1, ..., e_n$ and its edges being relationships between entities. Each edge receives part of the operation's estimated workload based on their proportional contribution to the whole estimation. Hence, the workload of any operation can be partitioned into

$$w = \sum w(edges) \tag{4.1}$$

Similarly, if we understand the ER model as an acyclic graph, q's entity accesses can also be represented as a graph, with nodes being entities said query access and edges being junctions (joins) between its nodes, with all series of operations $q_1, q_2, ..., q_n$ being represented as a series of graphs.When both (graph-represented ER model and operations' graphs) are combined, our resultant is a combination of graphs that reproduce relationships inside the same domain. Thus, its interpretation as a hypergraph, with ER's entities being nodes and $q_1, q_2, ..., q_n$ being hyperedges, comes naturally.Consequently, our problem is reduced to: "How to best make a k-way partition of this hypergraph based on a given ER schema and planned workload?".

With a series of planned operations $q_1, q_2, ..., q_n$ and its interpretation as a hypergraph, the last question that arises is: How do we group operations in order to, ideally, fully encompass its requirements into a single hypergraph? How many shards (clusters) is the recommended amount to fully group all operations?

To fully assess the similarities of the operations, we represent G as an adjacency matrix, with its values being the workload between the two edges. If there is no junction between entities, then the value is equal to zero. We also represented operations as a matrix, with shape [x,y] with x being the number of expected operations and y the total number of entities in the future database. To finalize, we also represented the similarity between operations as a matrix, with its values being the Jaccard Index (JACCARD, 1901) between q_u and q_v (s). The Jaccard Index metric identifies similarities between two sets (in our case, two operations' accessed entities) by dividing the intersection between both sets by the union of both sets. Mathematically, its representation is:

$$s(q_u, q_v) = \frac{|\,(E_u \cap E_v)\,|}{|\,(E_u \cup E_v)\,|}$$

with E_u and E_v being the entities accessed by operations u and v, respectively. The result is matrix $S_{u \times v}$, where each element represents the Jaccard Index of operations u and v, respectively.A value of 0 indicates that the evaluated operation has no accessed entities in common and, conversely, a value of 1 indicates that all entities are accessed by both operations. In short, we have the matrices:

$$A = \begin{bmatrix} w(e_1e_1) & w(e_1e_2) & \cdots & w(e_1e_j) \\ w(e_2e_1) & w(e_2e_2) & \cdots & w(e_2e_j) \\ \cdots & \cdots & \cdots & \cdots \\ w(e_ie_1) & w(e_ie_2) & \cdots & w(e_ie_j) \end{bmatrix} \quad O = \begin{bmatrix} q_1e_1 & q_1e_2 & \cdots & q_1e_l \\ q_2e_1 & q_2e_2 & \cdots & q_2e_l \\ \cdots & \cdots & \cdots & \cdots \\ q_ke_1 & q_ke_2 & \cdots & q_ke_l \end{bmatrix}$$

$$S = \begin{bmatrix} s(q_1q_1) & s(q_1q_2) & \cdots & s(q_1q_v) \\ s(q_2q_1) & s(q_2q_2) & \cdots & s(q_2q_v) \\ \cdots & \cdots & \cdots & \cdots \\ s(q_uq_1) & s(q_uq_2) & \cdots & s(q_uq_v) \end{bmatrix}$$

where matrix $A_{i \times j}$ (Adjacent Entities) is the matrix representation of the provided ER schema with workload consideration, matrix $O_{k \times l}$ (Operations Accesses) is the matrix representation of entities accessed by operation k, with each column $l$representing one entity of the provided ER schema (in other words: $k = i$ and $l = j$), finally matrix $S_{u \times v}$ (Similarity) is the Jaccard Index of every combination of operations u and u, respectively. It is important to note that $u = v$ in any situation, as well as $i = j$. Matrix O provides us a way to compute matrices of different sizes (since i is not necessarily equal to m) in an unique matrix with both considerations. With matrices A, O, S, we then perform $D = S \cdot O \cdot A$ matrix multiplications to have a resulting matrix of shape $[x, y]$ with x being the number of operations and y the number of entities into the ER model. Finally, the transposed matrix D^T serves as an input into the affinity propagation algorithm. The purpose of the transposition is to group e (entities) since, if not transposed, the grouping would occur over the planned operations. The proposed algorithm's purpose is to group entities into as many clusters as it considers necessary. Its results will, then, aid modelers into identifying what needs to be grouped and what does not.

Our proposal solves both questions presented at the beginning of this section by utilizing the affinity propagation algorithm. Our selection for this algorithm is based on its advantages over similar clustering algorithms, such as its ability to estimate a number of clusters to fully group all data, its deterministic characteristics and its better result in certain areas, such as Computer Vision (FREY; DUECK, 2007). Although the algorithm is not recommended for large datasets, we find it highly unlikely that an ER model will be large enough to pose a problem for such algorithm to be considered inefficient. We present an example of our proposal in the next section.

4.4 APPLICATION EXAMPLE

To demonstrate the application of our proposed workload metric and partitioning strategy, we present a working example of a simplified E-commerce. In this example we will guide the reader through every step of the complete process, from the beginning up until the proposed partitioning.

4.4.1 Scenario

Consider a (simplified) conceptual schema with three entities:

- Customer (C)
- Order (O)
- Product (P)

With Order having relationship with Customer and Product. And the 5 (five) planned operations that are:

- Q1: Gets information about the customer (Accesses:C);
- Q2: Gets information about all customers (Accesses:C);
- Q3: Gets orders from a customer (Accesses:O,C);
- Q4: Gets details from orders and products inside such orders (Accesses:O,P);
- Q5: Searches the (product) catalog (Accesses:P).

4.4.2 Workload Estimation

For executing this experiment on a concrete DB, we map the aforementioned conceptual schema to a relational schema composed of three tables, one for each entity. With values and calculations proposed in 4.2.1, as well as information provided in the subsection above, we infer that our planned operations' complexity are:

- Q1: Complexity 1 (Simple SELECT with filter);
- Q2: Complexity 2 (Simple SELECT without filter);
- Q3: Complexity 4 (JOIN with filter);
- Q4: Complexity 4 (JOIN with filter);
- Q5: Complexity 1 (Simple SELECT with filter).

Table 4 – Planned Operations and Their Total Workload

Operation	Workload	Complexity	Total Workload
Q1	8000	1	8000
Q2	8000	2	16000
Q3	188000	4	752000
Q4	276000	4	1104000
Q5	96000	1	96000

Our example e-commerce is a small company, with 1000 registered customers, 20000 orders and 6000 different products registered in its system. The Customer (C) table stores information about the customers in 8 attributes. The Orders (O) table stores information about the orders in 9 attributes. Finally, the Product (P) table stores information about the products sold by the company in 16 attributes.

The accessed volume of the planned operations are presented below:

- Q1: 1000 × 8 attributes (C);
- Q2: 1000 × 8 attributes (C);
- Q3: 20000 × 9 attributes (O) + 1000 × 8 attributes (C);
- Q4: 20000 result × 9 attributes (O) + 6000 × 16 attributes (P);
- Q5: 6000 × 16 attributes (P).

Taking into account both accessed volume and complexity, the estimated workload for each operation is presented in table 4.

4.4.3 Partitioning Strategy

Our first requirement to perform our partition strategy is to build the Operations Access (O), Adjacency (A) and Similarity (S) matrices. Matrix O is as follows:

$$O = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

Similarly, the Similarity Matrix (S) indicates similarity between operations as follows:

$$S = \begin{bmatrix} 1 & 1 & 0.5 & 0 & 0 \\ 1 & 1 & 0.5 & 0 & 0 \\ 0.5 & 0.5 & 1 & 0.33 & 0 \\ 0 & 0 & 0.33 & 1 & 0.5 \\ 0 & 0 & 0 & 0.5 & 1 \end{bmatrix}$$

Table 5 – Tables and Labels Recommended by Affinity Propagation

Table	Label
Customer	1
Orders	0
Products	1

Finally, Adjacency (A) is presented as follows:

$$A = \begin{bmatrix} 8000 & 188000 & 104000 \\ 188000 & 180000 & 114000 \\ 104000 & 114000 & 96000 \end{bmatrix}$$

With matrices S, O and A we calculate the dot product of them to identify the resultant matrix D, as mentioned in 4.3 ($D = S \cdot O \cdot A$).Matrix D is as follows:

$$D = \begin{bmatrix} 134000 & 134000 & 316360 & 411320 & 250000 \\ 650000 & 650000 & 892420 & 711840 & 351000 \\ 317000 & 317000 & 422980 & 473940 & 345000 \end{bmatrix}$$

Using matrix D as input for the Affinity Propagation algorithm, it labels tables as follows:

4.4.4 Results Interpretation

Based on the results presented in Table 5, the algorithm suggestion is, in short, that Customer and Products tables are to be grouped together and Orders table is to be considered another group. The labels 1 and 0 that appear in this case are simply considered for grouping reasons. Tables with the same label are recommended to be put together. The labels do not occur in any particular order and have no other meaning outside grouping (they do not serve the purpose of categorizing data in any way, such as quality, importance or anything else).

Based on this recommendation, a modeler should consider Functional and Non-Functional requirements of Tables Customer and Products as a whole,as well as consider Functional and Non-Functional requirements of Orders separately. If the three tables have similar requirements (e.g., Strong Consistency), both groups can be further grouped into one technology, such as, in this example, Relational. However, the opposite is not true: a modeler should not divide tables Products and Customers into two different technologies.

4.4.5 Discussion

In this example we provided a step-by-step overview of our methodology, as well as demonstrated that our methodology:

1. Provides a numerical estimation of Workload before its effective implementation;
2. Identifies relationships between entities and provides a semi-automated method of partitioning;
3. Provides a data-driven partitioning suggestion for ER Schemas prior to implementation in a database.

Our method can improve PP utilization by bringing an easier and reproducible partitioning strategy to aid both existing projects that want to migrate as well as new databases that are developed and implemented. In the next chapter we will validate the proposal utilizing a well-known, freely-available benchmark to validate our proposal in a more complex environment.

5 VALIDATION USING TPC-H BENCHMARK

This chapter presents a validation of our proposal in order to validate its functionality in complex domains, as well as to further exemplify its application. We use TPC-H's[1] benchmark database and queries to verify whether our proposed workload calculation is able to predict the execution time necessary to fully perform an operation. Our objective with this verification is to validate whether our metric is sound and if it can correctly capture differences between operations in an arbitrary domain. After this prediction, we evaluate the partitioning suggestions based on our metric.We chose to use the official PostgreSQL container image available in DockerHub[2] to create a more reproducible test environment. The container remained at default configuration as it was not our purpose to optimize it in any way. We then executed the script provided by TPC-H's benchmark to create database tables, indexes and relationships. After all the database schema were created, we use the also provided data generator to generate simulated data for all tables. We opted to utilize 10 times the default size to replicate a larger database and create more distinguishable differences between fast and slow operations. After all data was generated, it was then inserted into the newly created database.

Next, we evaluated the execution time of all 11 queries provided by TPC-H to be executed in PostgreSQL. Since the queries were originally developed to be executed by a JVM (Java Virtual Machine), some adaptations were made to each query to ensure all of them were able to execute, namely: values that were supposed to be parsed at the moment of execution were previously defined and kept the same for all further executions. Figure 3 presents the logic schema of TPC's benchmark database. The TPC is a non-profit corporation focused on developing data-centric benchmark standards and disseminating objective, verifiable data to the industry[3]. TPC describes its TPC-H benchmark as a decision support benchmark with ad-hoc queries that were chosen specifically for their industry-wide relevance.

We evaluated table sizes,number of attributes, number of occurrences for all tables, as well as complexity and tables accessed by each query to calculate its workload according to our proposal. Table 6 presents Tables names,number of attributes and occurrences.Table 7 presents Queries' names, number of accessed tables per query, and names of queried tables. Table 8 presents TPC's queries with total accessed volume, complexity and workload of each of them.

All operations were then evaluated utilizing individual executions using *pgbench*, a simple program for running benchmark tests on PostgreSQL, already built into Post-

1 https://www.tpc.org/tpch/
2 https://hub.docker.com/_/postgres
3 https://www.tpc.org/default5.asp

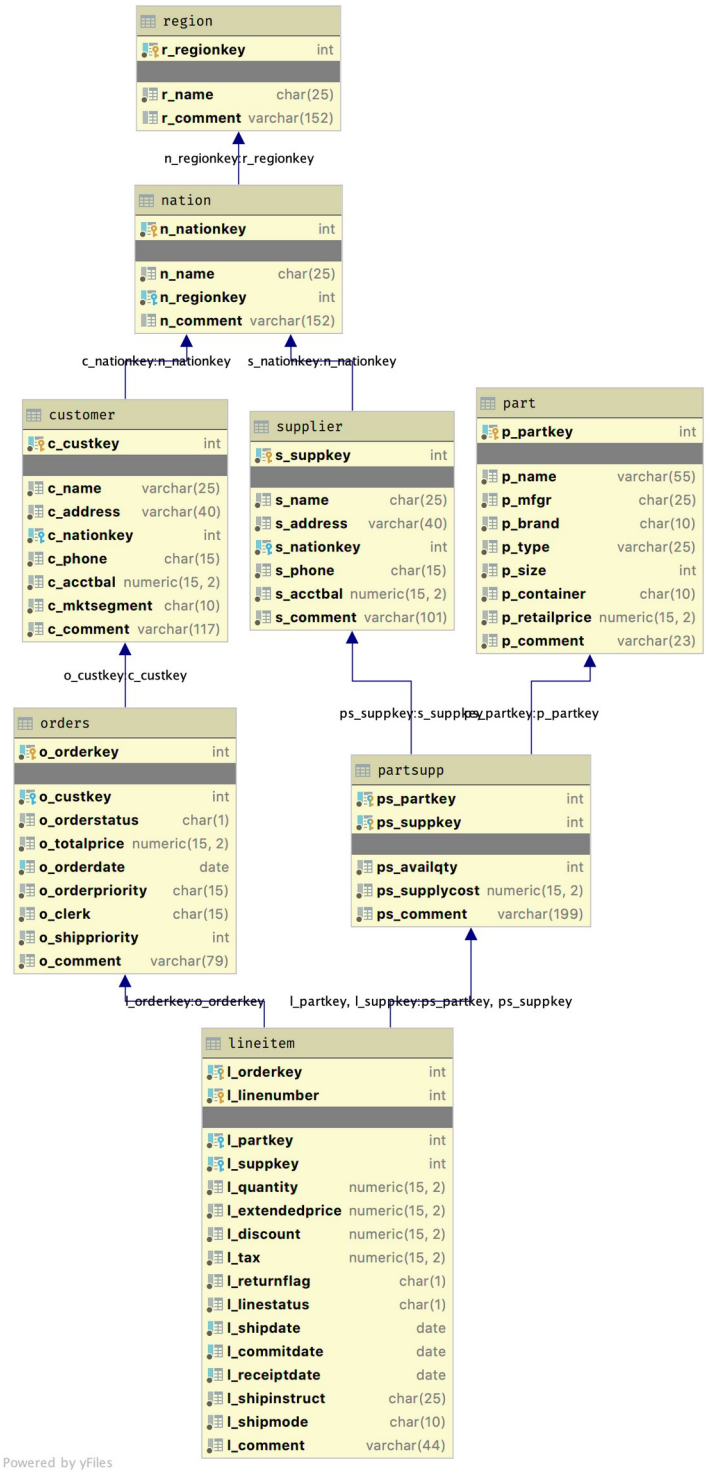


Figure 3 – TPC-H's Logic Schema

Table 6 – TPC Schemas' Tables, Attributes and Occurrences

Table	Attributes	Occurrences
lineitem	16	59986052
customer	8	1500000
orders	9	15000000
supplier	7	100000
nation	4	25
region	3	5
part	9	2000000
partsupp	5	8000000

Table 7 – TPC's Query Templates

Query	# Tables	Tables
TPC1	1	line_item
TPC3	3	customer, line_item, orders
TPC4	2	line_item, orders
TPC5	6	customer, line_item, nation, orders, region, supplier
TPC6	1	line_item
TPC10	4	customer, line_item, nation, orders
TPC12	2	line_item, orders
TPC14	2	line_item, part
TPC15	1	line_item
TPC20	5	line_item, nation, part, part_supp
TPC22	2	customer, orders

greSQL's database[4]. The results were then copied to a data sheet along with workload information. Executions were done isolated, so frequency was disregarded in this evaluation. Apart from TPC20, each query was repeated 200 times to generate a bigger subset and allow an evaluation of its execution time distribution. TPC20 was the only exception since its executions were abnormally long. Each one took 20+ minutes to execute, which made us limit its repetitions to 20 times.

All executions were performed with the same configuration to minimize hardware-related variations. The hardware where the container was executed is detailed in table 9.

5.1 DATA NORMALIZATION

Before an analysis of our evaluation, we performed a descriptive analysis of our observations to find whether its behavior was normal or if there were the need to perform any sort of data normalization. We identified outliers and discarded them to ensure consistency within the result. Figure 4a shows Query Execution Time and helps us identify outlier queries.

[4] https://www.postgresql.org/docs/current/pgbench.html

Table 8 – TPC's Query Acesses

Query	Volume	Complexity	Workload
TPC1	59986052	3	179958156
TPC3	76486052	4	305944208
TPC4	15000000	3	45000000
TPC5	76586082	4	306344328
TPC6	59986052	3	179958156
TPC10	76486077	4	305944308
TPC12	74986052	4	299944208
TPC14	61986052	4	247944208
TPC15	59986052	3	179958156
TPC20	100025	4	400100
TPC22	1500000	3	4500000

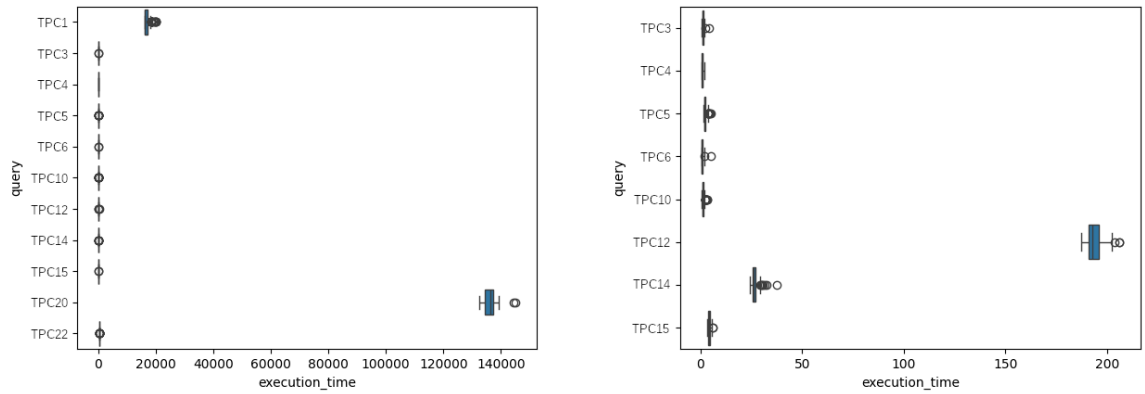
Table 9 – Host Machine Configuration

Name	Value
CPU	i7-13620H
GPU	Nvidia GeForce 4050
RAM	64Gb
SSD1	512Gb NVME 4.0
SSD2	4Tb NVME 4.0

As shown in Figure 4a, we identify that queries TPC1 and TPC20 were outliers for our variables (execution times were very different from other queries) and far from what was expected (as by our workload calculation, presented in Table 8TPC1 should have had similar time to TPC6 and TPC15, which was not observed.A further analysis identified that TPC1 failed to utilize any available index during the operation, which might be the reason for the unexpected results. Since our proposed metric does not deal with existence (or usage) of indexes, we disregarded TPC1 results, as it makes it necessary to revise variables in order to take indexes in consideration. Similarly, TPC20 has the lowest workload, which means it should have been the fastest query. Further analysis identified that TPC20 also failed to utilize any available index during its execution. We also opted to disregard it in order to not influence our evaluation, as, once again, our metric does not deal with such problem.

After the removal of both queries from our data, we reevaluated it and found one instance of TPC12 that took twice the average time of all TPC12 executions to execute. We have not been able to identify what caused the unexpected behavior. We opted to disregard that observation in order to avoid outlier influences within our results.

As our proposed workload metric easily stays in the hundreds of millions (10^8), the comparison against our execution time in milliseconds (10^{-3}) provides a scaling challenge, which results in very small differences in workload values not matching proportionally to what is observed in our end results. Hence, to mitigate such challenge



(a) All Queries (b) Outliers Removed

Figure 4 – Execution Time Grouped By Queries

and to more easily identify and understand such behavior, we have divided the workload by 1 million.

5.2 PARTITIONING SUGGESTION

Based on Table 8 and Figure 3, matrices represent entities adjancency (A), operations' tables accesses (O) and similarity between operations (S), respectively.After multiplying the matrices and transposing their result, the resultant matrix

By calculating the dot product of the matrices presented above, we then obtain the matrix D: The transposed matrix of D (D^T) is then used as input for the Affinity Propagation algorithm. Its results are then interpreted as the recommendation of partitions and recommendation of which entities should be grouped. The aforementioned matrix affinity propagation presents the same recommendation as of K-Means algorithm (MCQUEEN, 1967) (with the same number of clusters), unlike the DBSCAN algorithm (ESTER et al., 1996), which considers all observations as noise.We present in Table 10 a brief comparison of the results given by each algorithm. We reiterate that, although the numbers are inverted, what is important is not which number it is but which tables are grouped with which tables.

Table 10 – Comparison between Affinity Propagation, K-Means and DBSCAN

Entities	Affinity Propagation	K-Means	DBSCAN
lineitem	0	2	-1
customer	1	1	-1
orders	2	0	-1
supplier	1	1	-1
nation	1	1	-1
region	1	1	-1
part	1	1	-1
partsupp	1	1	-1

$$A = \begin{bmatrix} 1.92e09 & 9.72e08 & 1.09e09 & 9.60e08 & 9.60e08 & 9.60e08 & 9.78e08 & 1.00e09 \\ 9.72e08 & 2.40e07 & 1.47e08 & 1.27e07 & 1.20e07 & 1.20e07 & 3.00e07 & 5.20e07 \\ 1.09e09 & 1.47e08 & 2.70e08 & 1.36e08 & 1.35e08 & 1.35e08 & 1.53e08 & 1.75e08 \\ 9.60e08 & 1.27e07 & 1.36e08 & 1.40e06 & 7.00e05 & 7.00e05 & 1.87e07 & 4.07e07 \\ 9.60e08 & 1.20e07 & 1.35e08 & 7.00e05 & 2.00e02 & 1.15e02 & 1.80e07 & 4.00e07 \\ 9.60e08 & 1.20e07 & 1.35e08 & 7.00e05 & 1.15e02 & 3.00e01 & 1.80e07 & 4.00e07 \\ 9.78e08 & 3.00e07 & 1.53e08 & 1.87e07 & 1.80e07 & 1.80e07 & 3.60e07 & 5.80e07 \\ 1.00e09 & 5.20e07 & 1.75e08 & 4.07e07 & 4.00e07 & 4.00e07 & 5.80e07 & 8.00e07 \end{bmatrix}$$

$$O = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$S = \begin{bmatrix} 1 & 0.33 & 0.50 & 0.17 & 1 & 0.25 & 0.50 & 0.50 & 1 & 0.25 & 0 \\ 0.33 & 1 & 0.67 & 0.50 & 0.33 & 0.75 & 0.67 & 0.25 & 0.33 & 0.17 & 0.67 \\ 0.50 & 0.67 & 1 & 0.33 & 0.50 & 0.50 & 1 & 0.33 & 0.50 & 0.20 & 0.33 \\ 0.17 & 0.50 & 0.33 & 1 & 0.17 & 0.67 & 0.33 & 0.14 & 0.17 & 0.25 & 0.33 \\ 1 & 0.33 & 0.50 & 0.17 & 1 & 0.25 & 0.50 & 0.50 & 1 & 0.25 & 0 \\ 0.25 & 0.75 & 0.50 & 0.67 & 0.25 & 1 & 0.50 & 0.20 & 0.25 & 0.33 & 0.50 \\ 0.50 & 0.67 & 1 & 0.33 & 0.50 & 0.50 & 1 & 0.33 & 0.50 & 0.20 & 0.33 \\ 0.50 & 0.25 & 0.33 & 0.14 & 0.50 & 0.20 & 0.33 & 1 & 0.50 & 0.50 & 0 \\ 1 & 0.33 & 0.50 & 0.17 & 1 & 0.25 & 0.50 & 0.50 & 1 & 0.25 & 0 \\ 0.25 & 0.17 & 0.20 & 0.25 & 0.25 & 0.33 & 0.20 & 0.50 & 0.25 & 1 & 0 \\ 0 & 0.67 & 0.33 & 0.33 & 0 & 0.50 & 0.33 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} 2.99e08 & 7.79e08 & 1.79e08 & 1.53e09 & 3.59e08 & 5.79e08 & 1.79e08 & 1.85e08 \\ 5.49e08 & 1.28e09 & 3.66e08 & 2.42e09 & 6.40e08 & 9.75e08 & 3.66e08 & 3.78e08 \\ 4.38e08 & 1.09e09 & 2.75e08 & 2.12e09 & 5.20e08 & 8.21e08 & 2.75e08 & 2.84e08 \\ 6.37e08 & 1.35e09 & 4.59e08 & 2.46e09 & 7.26e08 & 1.05e09 & 4.59e08 & 4.69e08 \\ 2.99e08 & 7.79e08 & 1.78e08 & 1.53e09 & 3.59e08 & 5.79e08 & 1.78e08 & 1.85e08 \\ 6.13e08 & 1.37e09 & 4.23e08 & 2.56e09 & 7.09e08 & 1.06e09 & 4.23e08 & 4.34e08 \\ 4.38e08 & 1.09e09 & 2.75e08 & 2.12e09 & 5.20e08 & 8.21e08 & 2.75e08 & 2.84e08 \\ 2.88e08 & 7.30e08 & 1.77e08 & 1.42e09 & 3.43e08 & 5.46e08 & 1.77e08 & 1.84e08 \\ 2.99e08 & 7.79e08 & 1.78e08 & 1.53e09 & 3.59e08 & 5.79e08 & 1.78e08 & 1.85e08 \\ 4.48e08 & 9.72e08 & 3.17e08 & 1.79e09 & 5.14e08 & 7.54e08 & 3.17e08 & 3.25e08 \\ 3.60e08 & 8.07e08 & 2.49e08 & 1.50e09 & 4.16e08 & 6.21e08 & 2.49e08 & 2.55e08 \end{bmatrix}$$

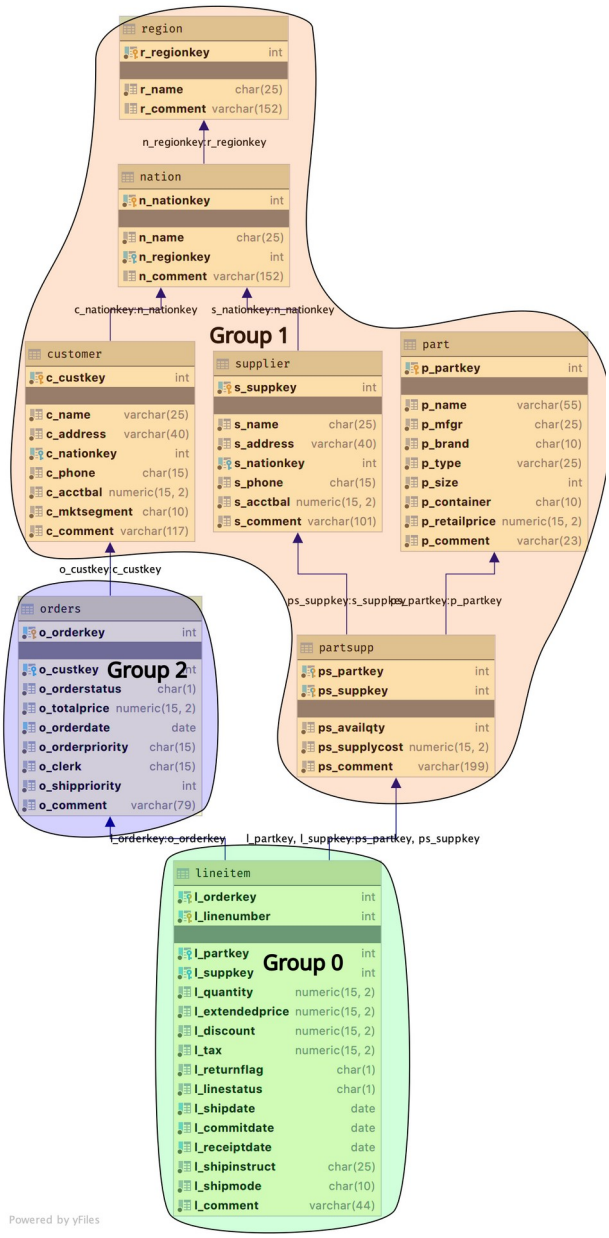


Figure 5 – Entities by Cluster Reccomendation

This suggests that tables customer, supplier, nation, region, part and partsupp should be grouped together, with a second cluster being formed by only table lineitem while the third group is formed by only the orders table. We highlight that lineitem and orders are the biggest tables by a significant margin (orders is at least ten times bigger than the third biggest) and there is no penalty for inter cluster operations (operations that interact with more than one cluster). Figure 5 shows entities recommended by clusters.

6 REAL-WORLD DOMAIN APPLICATION

To reinforce the utilization of the proposed approach into a real-case scenario, we implemented our solution into the domain of the *Projeto Ceos*, a partnership between the Federal University of Santa Catarina (UFSC) and the public sector, more specifically, the Santa Catarina public ministry (MP-SC) to develop an AI solution to aid the identification of corruption in procurement processes[1]. In particular, the Ceos DB is an exceptionally difficult modelling case not just because of the sheer amount of information that is processed, organized and analyzed,but also because its domain is very complex, with many domain requirements that come from laws in federal, state and municipal level, as well as many characteristics of big data, such as variability, variety, viscosity, and so on.

The current project Database utilizes a fully relational structure, with its design resembling the third normal form, as shown in Figure 6 presents the DB structure as it's obtained from different sources (orange and green).

A prototype panel is under development to aid prosecutors to visualize which procurements are suspected to have irregularities and why. Although under development, there are a total of 8 operations already in place to provide data for the aforementioned panel. These operations are presented in Table 11. Since there are tables that are not needed to be accessed for such operations, we chose to disregard such tables and focus instead on those who are queried by the panel operations.

Table 11 – Operations and Accessed Tables

Operation	Accessed Tables
Q1	ItensNFe; CabecalhoNFe
Q2	ProcessoLicitatorio; Contrato; ModalidadeLicitacao; TipoLicitacao; SituacaoProcessoLicitatorio; UnidadeGestora; Ente
Q3	Documentos
Q4	ItemLicitacao; Cotacao; ParticipanteLicitacao
Q5	ParticipanteLicitacao; Cotacao; CNPJ
Q6	Cotacao; ItemLicitacao
Q7	CNPJ
Q8	ViewProcessoLicitatorio

As can be observed in Figure 7, there are only a handful of tables that are being accessed at the moment. Similarly, Figure 8 repeats Figure 7 with the addition of a visualization of entities accessed per operation. Once again, as mentioned in Section 4.3, if we consider all entities (or tables, when such analysis occurs in an already implemented database) of an ER Model as vertices from a hypergraph, then all operations that occur over it (over its entities/tables) can be interpreted as hyperedges of said hypergraph.

[1] https://ceos.ufsc.br/

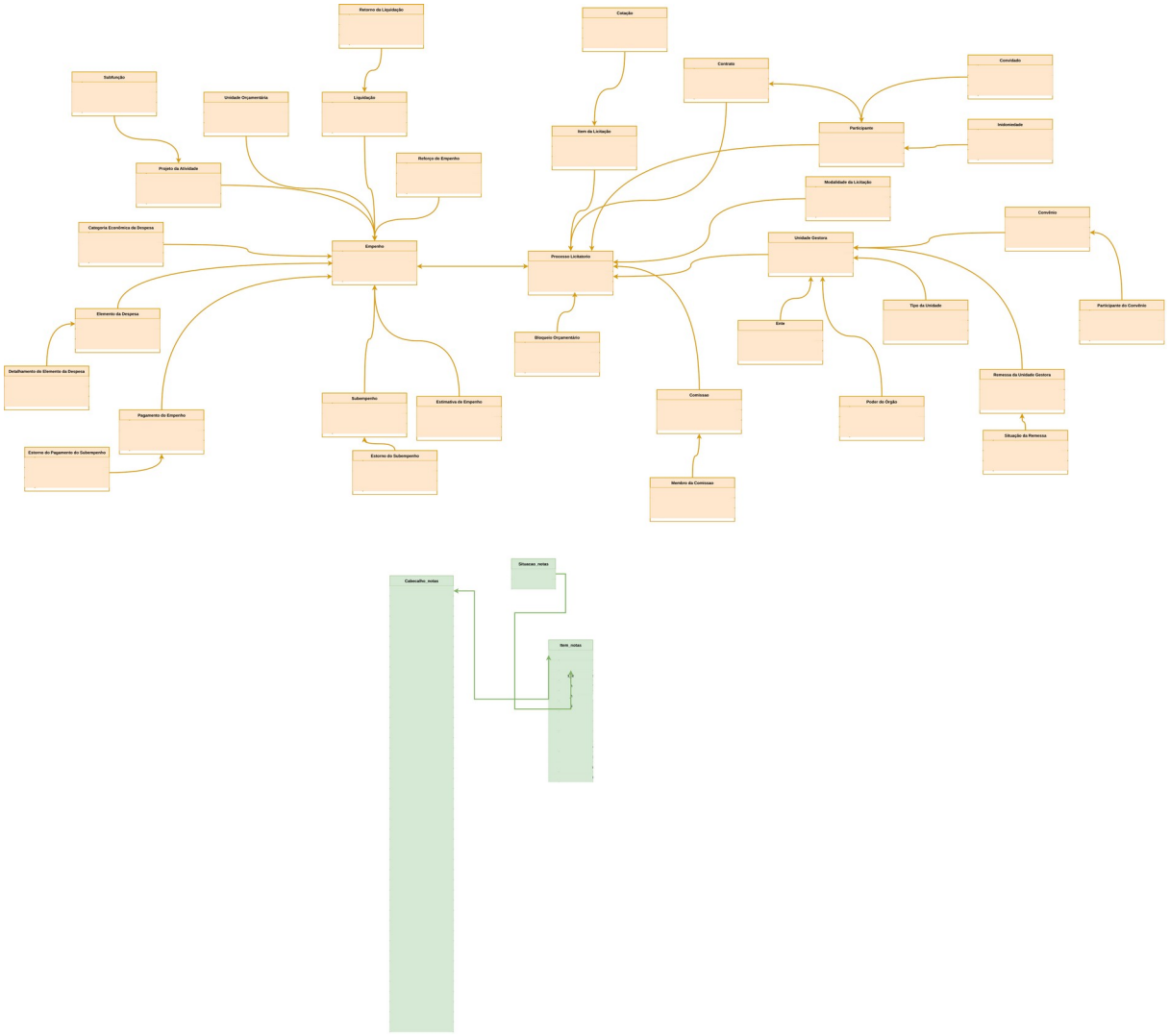


Figure 6 – Ceos Database Relational Schema

Thus, our problem can be reduced to the most efficient way to partition a hypergraph utilizing workload as an objective function.

Similarly to our example presented at Chapter 5, planned operations are presented in Table 12.Ceos Database current structure can be represented as the adjacency matrix with similarity matrix:

Table 12 – Current Operations and Their Workloads

Entity	Workload
item_tabela	1848457376
empresas_por_licitacao	3388314892
consulta_empresas	1536022089
lista_documentos	41968
itens_por_licitacao	1963164800
itens_semelhantes	24874403160
todas_licitacoes	52289820
licitacao	250610528

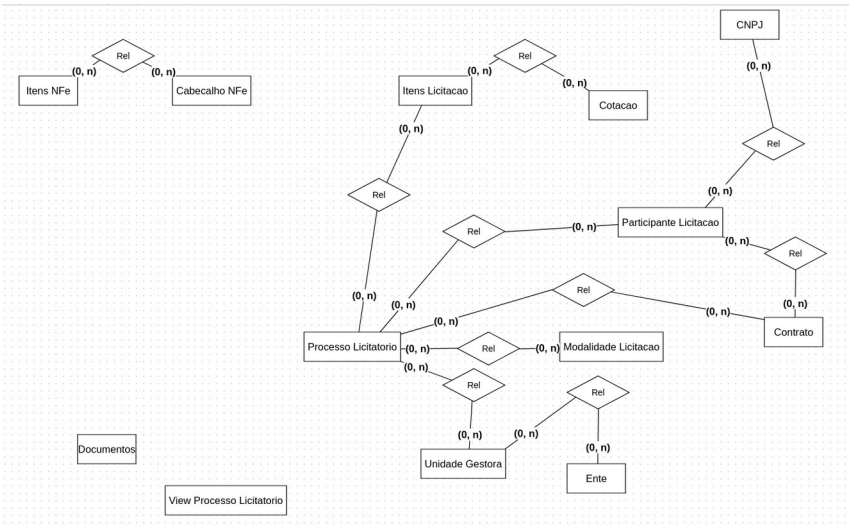


Figure 7 – Simplified Ceos Database (only entities with access)

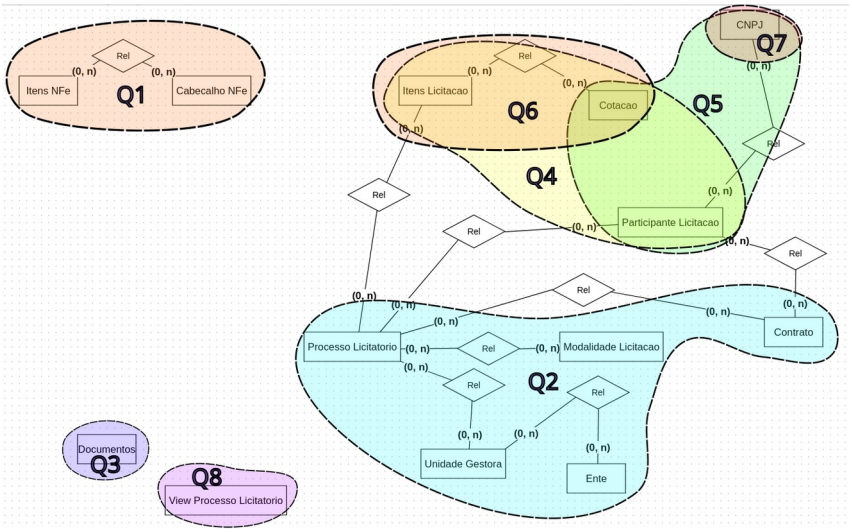


Figure 8 – Ceos Database With Operations' Accesses

$$
Adj = \left[\begin{array}{cccccccccccccc|c}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \textit{Cotacao} \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \textit{ItemLicitacao} \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & \textit{ParticipanteLicitacao} \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \textit{CNPJ} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \textit{Documentos} \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \textit{ItensNFe} \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \textit{CabecalhoNFe} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \textit{ViewProcessoLicitatorio} \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & \textit{ProcessoLicitatorio} \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \textit{Contrato} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \textit{ModalidadeLicitacao} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \textit{TipoLicitacao} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \textit{SituacaoProcessoLicitatorio} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & \textit{UnidadeGestora} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \textit{Ente}
\end{array}\right]
$$

$$S = \begin{bmatrix} 1 & 0.25 & 0 & 0 & 0.67 & 0 & 0 & 0 \\ 0.25 & 1 & 0.33 & 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0.33 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0.67 & 0.5 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 7.7e8 & 3.5e8 & 0 & 0 & 7.7e8 & 0 & 0 & 0 \\ 8.9e8 & 8.1e8 & 1.5e8 & 0 & 1.0e9 & 0 & 0 & 1.9e8 \\ 1.4e8 & 7.2e8 & 7.2e8 & 0 & 2.7e8 & 0 & 0 & 1.2e8 \\ 5.0e8 & 8.1e8 & 1.8e8 & 0 & 8.1e8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3.1e7 \\ 0 & 0 & 0 & 0 & 0 & 6.2e9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 6.2e9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3.7e8 & 2.3e8 & 2.0e7 & 3.1e7 & 4.0e8 & 0 & 0 & 1.9e8 \\ 5.5e7 & 9.0e7 & 2.0e+ & 0 & 9.0e7 & 0 & 0 & 6.3e7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3.1e7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3.1e7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3.1e7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3.1e7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5.6e4 \end{bmatrix}$$

After matrices multiplication and transposition, the resultant matrix: serves as an input to the affinity propagation algorithm that returns Table 13 as its suggestion on how to partition the data. We highlight once more that Affinity Propagation and K-Means returned the same grouping when K-Means utilizes the same number of clusters as Affinity Propagation. DBSCAN once more considered all entities as noise, as presented in Table 13.

With the suggestion presented in table 13, it is now up to the modeler to decide whether they will follow the recommendation, ignore it entirely or anything else in between. It can be seen in Figure 9.

Table 13 – Partition Suggestions Affinity Propagation vs K-Means vs DBSCAN

Entity	Affinity	K-Means	DBSCAN
Cotacao	0	2	-1
ItemLicitacao	0	2	-1
ParticipanteLicitacao	0	2	-1
CNPJ	0	2	-1
Documentos	2	0	-1
ItensNFe	1	1	-1
CabecalhoNFe	1	1	-1
ViewProcessoLicitatorio	2	0	-1
ProcessoLicitatorio	2	0	-1
Contrato	2	0	-1
ModalidadeLicitacao	2	0	-1
TipoLicitacao	2	0	-1
SituacaoProcessoLicitatorio	2	0	-1
UnidadeGestora	2	0	-1
Ente	2	0	-1

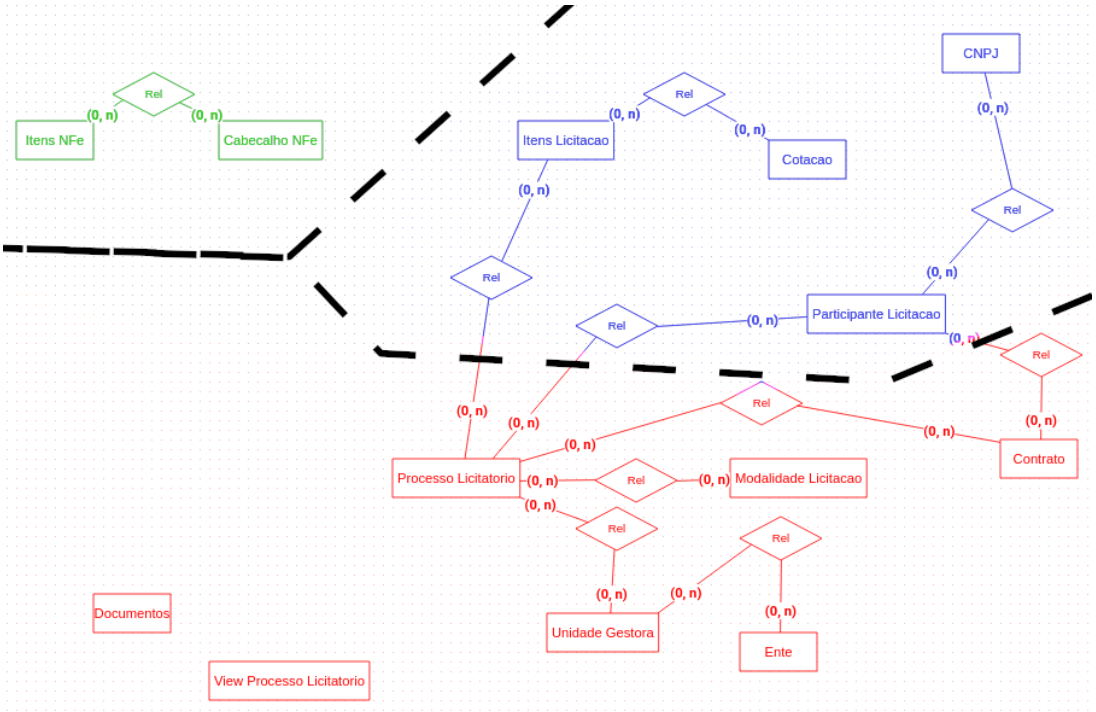


Figure 9 – Ceos DB Partitioned

7 DISCUSSION

In this Chapter, we present a discussion about our results, its accuracy and argue that our results show our objective was achieved albeit there is room for improvement in many different fronts. We divided the discussion into two topics: workload discussion and partitioning strategy.

7.1 WORKLOAD DISCUSSION

Our first evaluation compared Ordinary Least Square Linear Model (OLS), General Linear Model (GLM) and General Additive Model (GAM). The utilized metrics were chosen for their simplicity and, as there are no proposals to compare against, they do not rely on complex interactions, being simply represented as a simple linear regression

$$Y = \alpha_1 X_1 + \alpha_2 X_2 + \varepsilon$$

being Y the dependent variable (what we want to estimate), X the independent variable (what we use to estimate), α the parameters of aforementioned variables and ε the measured error. GLM as well as GAM simply generalize a linear model to make it more robust. We utilized this metrics to find out whether our workload proposal results in accurate predictions to be able to evaluate wether our proposed metric was sound enough to be utilized for partitioning. We chose to compare GAM and GLM against OLS because both GLM and GAM have more flexible behavior with non-normal residuals. We evaluated both (GAM and GLM) models utilizing a normal model family with Identity link function.

The OLS resulted in a very low R^2 ($R^2 = 0.087$), which indicates very low accuracy.We notice a very high result on our Jarque-Bera test(JARQUE; BERA, 1980), which indicates our data does not have a normal distribution. Hence, our selection for GLM and GAM models. It is also important to mention that, although with low accuracy, it suggests our proposal is statistically significant with a very high probability (*p-value* < 0.0001). A probable cause is that we missed at least one other variable that influences the result. We opted to select very few variables in order to be able to summarize our calculations to a single metric in order to make it easier to distinguish and compare operations.

Our GLM evaluation showed very similar results as the OLS presented before. We observe the same coefficients for both intercept and workload in both models.The GAM however, with its adaptability to non-linear relationships, resulted in a *PseudoR*^{2} of 0.32 in its best combination, with 3 total *formulae*. Although better, it can be still considered low accuracyOnce again, the p-value continues to be statistically significant, which suggests the validity of the approach and further reinforces the theory that we

Generalized Linear Model Regression Results

Dep. Variable:	execution_time	No. Observations:	1599
Model:	GLMGam	Df Residuals:	1595
Model Family:	Gaussian	Df Model:	3.00
Link Function:	Identity	Scale:	2846.7
Method:	PIRLS	Log-Likelihood:	-8626.0
Date:	Thu, 10 Jul 2025	Deviance:	4.5404e+06
Time:	10:56:26	Pearson chi2:	4.54e+06
No. Iterations:	3	Pseudo R-squ. (CS):	0.3205
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	29.6146	4.869	6.083	0.000	20.072	39.157
workload	0.0908	0.017	5.471	0.000	0.058	0.123
workload_s0	-24.8730	1.212	-20.516	0.000	-27.249	-22.497
workload_s1	11.2792	0.786	14.359	0.000	9.740	12.819

Figure 10 – GAM Model

Generalized Linear Model Regression Results

Dep. Variable:	execution_time	No. Observations:	1599
Model:	GLMGam	Df Residuals:	1596.00
Model Family:	Gaussian	Df Model:	2.00
Link Function:	Identity	Scale:	3515.0
Method:	PIRLS	Log-Likelihood:	-8795.1
Date:	Thu, 10 Jul 2025	Deviance:	5.6099e+06
Time:	12:19:19	Pearson chi2:	5.61e+06
No. Iterations:	3	Pseudo R-squ. (CS):	0.1155
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-62.1191	5.999	-10.355	0.000	-73.877	-50.361
query_cost_1m	0.0601	0.108	0.557	0.578	-0.151	0.271
complexity	41.1548	4.324	9.517	0.000	32.679	49.630
complexity_s0	-62.1191	5.999	-10.355	0.000	-73.877	-50.361
complexity_s1	-1.809e-14	1.75e-15	-10.350	0.000	-2.15e-14	-1.47e-14
complexity_s2	-1.809e-14	1.75e-15	-10.350	0.000	-2.15e-14	-1.47e-14

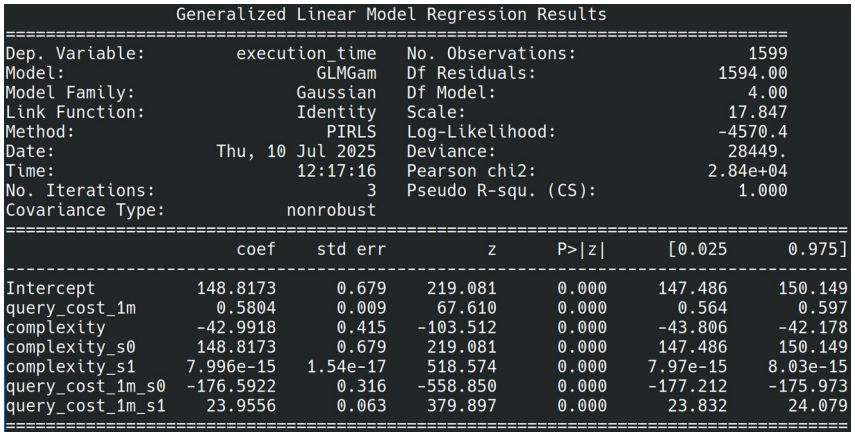
Figure 11 – GAM With Complexity Addition

chose too few variables.

With the selection for the GAM being the most logical, since it performed the best, we performed two additional GAM evaluations: One with only Complexity Addition (Figure 11) and another with only query cost (Accessed Volume in our proposal) addition(Figure 13). While the GAM with only Complexity resulted in worse accuracy, as shown by the *Pseudo* R^2 in Figure 11, the GAM with query cost additions resulted in *Pseudo* $R^2 = 1$, which generally indicated a high probability of over-fitting in such model. Figure 13 shows the summary of this model.

Overall our workload proposal has signs of being a viable option to estimate execution time and to evaluate performance of a DB. It still requires improvement before it can be recommended for general use but, being a first version and with more contributions to tackle the many fronts that need development, it can be safely assumed to be an interesting proposal to aid DB modeling in the future.

Additional evaluations, such as sensitivity analysis of the dependent variables, different models, different interactions (e.g. x^i, $\ln(x)$ etc.) can be very beneficial to determine how robust the model is and future directions to improve the proposal to increase accuracy while maintaining versatility to be useful in different situations and with different hardware configurations. Proposals could also evaluate different hardware

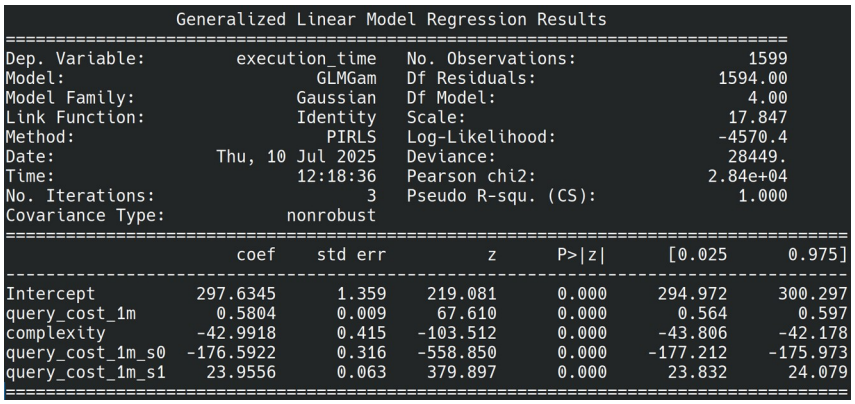


Generalized Linear Model Regression Results

Dep. Variable:	execution_time	No. Observations:	1599
Model:	GLMGam	Df Residuals:	1594.00
Model Family:	Gaussian	Df Model:	4.00
Link Function:	Identity	Scale:	17.847
Method:	PIRLS	Log-Likelihood:	-4570.4
Date:	Thu, 10 Jul 2025	Deviance:	28449.
Time:	12:17:16	Pearson chi2:	2.84e+04
No. Iterations:	3	Pseudo R-squ. (CS):	1.000
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	148.8173	0.679	219.081	0.000	147.486	150.149
query_cost_1m	0.5804	0.009	67.610	0.000	0.564	0.597
complexity	-42.9918	0.415	-103.512	0.000	-43.806	-42.178
complexity_s0	148.8173	0.679	219.081	0.000	147.486	150.149
complexity_s1	7.996e-15	1.54e-17	518.574	0.000	7.97e-15	8.03e-15
query_cost_1m_s0	-176.5922	0.316	-558.850	0.000	-177.212	-175.973
query_cost_1m_s1	23.9556	0.063	379.897	0.000	23.832	24.079

Figure 12 – GAM With Both Additions



Generalized Linear Model Regression Results

Dep. Variable:	execution_time	No. Observations:	1599
Model:	GLMGam	Df Residuals:	1594.00
Model Family:	Gaussian	Df Model:	4.00
Link Function:	Identity	Scale:	17.847
Method:	PIRLS	Log-Likelihood:	-4570.4
Date:	Thu, 10 Jul 2025	Deviance:	28449.
Time:	12:18:36	Pearson chi2:	2.84e+04
No. Iterations:	3	Pseudo R-squ. (CS):	1.000
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	297.6345	1.359	219.081	0.000	294.972	300.297
query_cost_1m	0.5804	0.009	67.610	0.000	0.564	0.597
complexity	-42.9918	0.415	-103.512	0.000	-43.806	-42.178
query_cost_1m_s0	-176.5922	0.316	-558.850	0.000	-177.212	-175.973
query_cost_1m_s1	23.9556	0.063	379.897	0.000	23.832	24.079

Figure 13 – GAM With AccessedVolume Addition

configurations to evaluate whether (and how much) hardware influences execution time. Such evaluations would also help to determine the robustness of the proposal and future research paths related to robustness and flexibility.

Finally, to evaluate the workload proposal, we utilized a parallel to execution time. Future analysis might focus on evaluation using a different interpretation with the aim to improve robustness. A possible evaluation could focus, as an example, on ranking operations based on perceived workload by providing a predicted relative query execution time (relative to a specific, known operation). Such a method brings advantages by providing a overall evaluation of operations while just having to execute a small portion of them in a DB in order to have enough data for proper evaluation.

7.2 PARTITIONING DISCUSSION

As presented in Table 10 and Table 13, the utilized algorithm provides the same result as another widely utilized clustering algorithm with the considerable advantage that it does not need a number of clusters as input; the number of clusters is defined by the data fed into the model.

In our example utilizing TCP-H benchmark, the algorithm understood that the

best division for that data was two clusters, with one of them having 3 of the total 8 tables and another cluster having the remaining 5. Its consideration takes into account proximity, quantity of queries and similarities between queries. As mentioned before, our proposal did not implemented any sort of penalty for queries that execute between clusters.

With the implementation into CEOS Project's DB, presented at table 13, it can be said that, in short, there are at least two clusters: CabecalhoNFe and ItensNFe, which comes from a different source of the remaining data. Similarly, when we focus on the utilized operations, the one who queries ItensNFe also queries CabecalhoNFe but no other entity. No other operation queries neither entity. Operations which could theoretically be grouped independently were those who queries Documentos and View-ProcessoLicitatorio since they query only that single table and they are queried alone. Although ViewProcessoLicitatorio says it is not adjacent to any other table, it is actually a Materialized View based on ProcessoLicitatorio. We opted to leave it as is to simplify the process.

Future works who decide to improve partitioning suggestion will find that focusing on hypergraph partitioning with different optimization objectives. Future proposals can also evaluate the addition of penalties for executions who execute through more than one partition at a time.

7.3 RESULTS DISCUSSION

Our workload calculation yielded positive results.Although with lower accuracy than expected, the utilized models are statistically significant, which indicates that our proposal has adherence. Generally speaking, poor performance was expected in all models since we knew our data had non-linear behavior, which heavily impacted accuracy. GAM's accuracy was better than other tested models, likely because of its flexibility. When compared to GLM and OLS, GAM is almost 4 times more accurate. Our observation on models' deviances and *Pseudo*R^2 reiterate clearly that Complexity and Accessed Volume weight differently at the execution time for an operation. We also verified that there are likely more variables that need to be considered to fully evaluate execution time of an operation. All considered, our initial proposal requires a weighted variable calculation in order to perform accordingly to what is expected of it.

It is also very likely we have co-linearity problems between our variables since both evaluate different characteristics but indirectly interact with one another (i.e., A more complex query has a higher probability to require more tables, which in turn results in higher Accessed Volumes, which translates into higher query costs).Our attempted strategy to diminish such effects was the utilization of our Complexity metric. Still, it is clear complexity is related to Accessed Volume. Although co-linearity can be considered a non-crucial improvement, it is still an important factor to be taken into

account. A new definition/metric for Complexity might be a better consideration than current proposal.

TPC-H had a considerable amount of queries available. Still, since its focus is on business-oriented ad-hoc queries,only operations with complexity 3 and 4 were found and evaluated, leaving 5 of our 7 proposed Complexities without representation. It is imperative that broader tests are performed before such proposal can be fully ready to its utilization as a general aid for any database modeling. TPC-H's also presented operations that had unpredictable behavior (namely, operations TPC1 and TPC20). Those operations failed to utilize available indexes without any clear sign of the reason. That reinforces our perceived necessity to include a metric for index utilization as an explanatory variable. It also adds the need to understand the execution planning as an requirement to evaluation of inconsistencies.

Our proposal was tested and evaluated with a RDBMS, being it considered insufficient for a broader proposal. Further tests with different DBs, specially with different technologies (i.e. Document Stores) is also required to further evolve our workload proposal into a more reliable, technology agnostic evaluation. Such tests are suggested to be performed to increase confidence with such results. Still, it has to be taken into consideration that NoSQL performance relies on a less direct junction strategy. A series of tests with same data in different structures is recommended to better evaluate accuracy between many strategies.

Although our partition strategy provided satisfying results, there are many similarity metrics that could potentially work as well as or better than Jaccard's. Similarly, with workload improvements or addition of penalties for operations that occur in more than one cluster.

8 CONCLUSION

This dissertation proposes a formal workload definition, as well as a novel workload calculation and its utilization to partition entities of an conceptual/logical schema. We performed a series of statistical tests to evaluate our proposal accuracy when it estimates execution time. Our evaluations showed promising results with the workload estimation.

Our workload proposal also is statistically significant, which indicates our proposal seems to be sound and viable. It presents a novel strategy to estimate DB requirements and strategies prior to implementation.Our proposal also can aid in the DB life-cycle by being a first sign of entities that might become a problem for performance.

Although with considerably good results, we believe it is still necessary to broaden our evaluations to create a more robust metric before it can be widely utilized. Still, its potential utilization to better guide DB development, modeling and to make easier to compare different proposals throughout a single metric, something that was very time-consuming before our proposal.For the presented reasons, we argue our workload proposal impacts positively DB design and implementation.

Overall, our partitioning proposal had success in its tests.Affinity Propagation can be considered an excellent option to be utilized into our proposal because it solves both problems we identified with clustering. As mentioned in Section 7.2, there are other variables that could also aid partitioning with potentially better results. Overall, the partitioning provides a valid alternative with arguably easy to understand partitions and metric.

8.1 SUGGESTIONS FOR FUTURE CONTRIBUTIONS

Future contributions are focused on three main groups: Workload Estimation; Partitioning Strategy; and (current) Proposal Extension.

Suggestions for contributions to workload estimation includes but are not limited to: Proposal and evaluation of additional variables; Evaluation of additional models; Evaluation of workload estimations in different technologies, specially NoSQL. The reasoning behind proposals within this group are to expand accuracy, improve robustness,include more relevant variables, improve reliability throughout different configurations, such as different hardware, and overall improvements to the estimation model itself. We believe proposals focused on workload estimation to improve the accuracy of our proposed model and to improve robustness by taking into account a better representation of the relationship between proposed variables (Complexity, Frequency, ...) and, possibly, new variables.

Suggestions for contributions to partitioning strategy includes but are not limited to: Evaluation of additional variables; Evaluation of added constraints into the

partitioning logic; Evaluation of different similarity metrics; Evaluation of added penalties for operations that execute in different clusters; and application of methods that allow replication of entities through more than one partition. Our reasoning for the suggestions inside this group are related to efficient partitioning when other/multiple optimization are required. Our evaluation considered the overall workload, but it did not lead into balanced partitions, did not introduced a penalty for operations who occur over more than one partition and utilized the default parameters. All three factors can influence the number and format of partitions. Hence, we point them as future paths. We expect future developments in this path to propose/implement different considerations into partitioning, as it is unlikely only one optimization could cover the vast quantity of possible optimizations required by a real-world DB. As complexity grows, our partition suggestion can be an interesting initial proposal but we believe it will likely be changed depending on the domain and (non- and functional) requirements. To offer a better starting point to modelers, our model will be the basis on which other researchers' proposals will optimize for different aspects, considerations and domains.

Future work extending this proposal should focus on recommending technologies for each partition, based on workload, functional, and non-functional requirements. Proposals who enable current utilization are also recommended, specially to better understand how users are going to use these proposals. We reason that our proposal lacks the final step of a "useful" partition. We suggest a form of partition, but there are more decisions,such as the aforementioned selection of technology,that can be given based on which entities are within the partition, as well as the operations over them and desired functional and non-functional requirements. Such decisions can be aided by future proposals that extend ours and, as such, are suggested in this group.
While we believe our proposal has the potential to improve and aid modelers, its interpretation is currently not easy. We expect to see future contributions within this path to focus on user experience improvements, extensions to recommend the best technology based on requirements and real implementations into systems to evaluate metrics such as closeness to modelers objectives.

BIBLIOGRAPHY

AHMAD, M. et al. Predicting completion times of batch query workloads using interaction-aware models and simulation. Proceedings of the 14th International Conference on Extending Database Technology, p. 449–460,2011. Disponível em: https://doi.org/10.1145/1951365.1951419.

AMJAD, M. et al. Estimate the total completion time of the workload. International Journal of Advanced Computer Science and Applications, The Science and Information Organization, v. 11, n. 6, 2020. Disponível em: http://dx.doi.org/10.14569/IJACSA.2020.0110606.

ATREY, A. et al. Unifydr: A generic framework for unifying data and replica placement. IEEE Access, v. 8, p. 216894–216910, 2020.

BRITANNICA. 2025. Disponível em: https://www.britannica.com/science/partition-of-a-set.

CALATRAVA, C. G.; BECERRA, Y.; CUCCHIETTI, F. M. Introducing Polyglot-Based Data-Flow Awareness to Time-Series Data Stores. IEEE ACCESS, v. 10, p. 69398–69411, 2022. ISSN 2169-3536.

CANDEL, C. J. F.; RUIZ, D. S.; GARCIA-MOLINA, J. J. A unified metamodel for NoSQL and relational databases. INFORMATION SYSTEMS, v. 104, 2022. ISSN 0306-4379.

ÇATALYÜREK, Ü. et al. More recent advances in (hyper) graph partitioning. ACM Computing Surveys, ACM New York, NY, v. 55, n. 12, p. 1–38, 2023.

CHEN, L.; DAVOUDIAN, A.; LIU, M. A workload-driven method for designing aggregate-oriented NoSQL databases. Data & Knowledge Engineering, v. 142, p. 102089, 2022. ISSN 0169-023X. Disponível em: https://www.sciencedirect.com/science/article/pii/S0169023X22000805.

CHOI, K. et al. Workload-optimized sensor data store for industrial IoT gateways. FUTURE GENERATION COMPUTER SYSTEMS-THE INTERNATIONAL JOURNAL OF ESCIENCE, v. 135, p. 394–408, 2022. ISSN 0167-739X.

CRUZ, F. et al. MeT: workload aware elasticity for NoSQL. Proceedings of the 8th ACM European Conference on Computer Systems, p. 183–196,2013. Disponível em: https://doi.org/10.1145/2465351.2465370.

CUBUKCU, U. et al. Citus: Distributed PostgreSQL for Data-Intensive Applications. Proceedings of the 2021 International Conference on Management of Data, p. 2490–2502, 2021. Disponível emhttps://doi.org/10.1145/3448016.3457551.

DANAOUI, M. E. et al. Leveraging Workload Prediction for Query Optimization in Multi-Tenant Parallel DBMSs. Proceedings of the 2024 8th International Conference on Cloud and Big Data Computing, p. 41–48, 2024. Disponível em: https://doi.org/10.1145/3694860.3694866.

DEEP, S. et al. Comprehensive and efficient workload compression. Proc. VLDB Endow., v. 14, n. 3, 2020. ISSN 2150-8097. Disponível em: https://doi.org/10.14778/3430915.3430931.

DOMASCHKA, J. et al. Using eBPF for Database Workload Tracing: An Explorative Study. Companion of the 2023 ACM/SPEC International Conference on Performance Engineering, p. 311–317,2023. Disponível em: https://doi.org/10.1145/3578245.3584313.

DUGGAN, J. et al. Performance prediction for concurrent database workloads. In: Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data. New York, NY, USA: Association for Computing Machinery, 2011. (SIGMOD '11), p. 337–348. ISBN 978-1-4503-0661-4. Disponível em: https://doi.org/10.1145/1989323.1989359.

DUGGAN, J. et al. Packing light: Portable workload performance prediction for the cloud. 2013 IEEE 29th International Conference on Data Engineering Workshops (ICDEW), p. 258–265, 2013.

ESTER, M. et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In: kdd. [S.l.: s.n.], 1996. v. 96, n. 34, p. 226–231.

FAN, W. Big graphs: Challenges and opportunities. VLDB Endowment, p. 3782, 2022.

FARAJ, H. A. Moving RDBMS to NoSQL Paradigms. 2022 International Conference on Computational Science and Computational Intelligence (CSCI), p. 684–689, 2022. ISSN 2769-5654.

FARIAS, V. A. E. et al. Machine Learning Approach for Cloud NoSQL Databases Performance Modeling. 2016 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), p. 617–620, 2016.

FREY, B. J.; DUECK, D. Clustering by Passing Messages Between Data Points. Science, v. 315, n. 5814, p. 972–976, 2007. ISSN 0036-8075, 1095-9203. Disponível em: https://www.science.org/doi/10.1126/science.1136800.

GAREY, M. R.; JOHNSON, D. S.; STOCKMEYER, L. Some simplified np-complete problems. In: Proceedings of the sixth annual ACM symposium on Theory of computing. [S.l.: s.n.], 1974. p. 47–63.

GE, J.-K.; CHAI, Y.-F.; CHAI, Y.-P. WATuning: A Workload-Aware Tuning System with Attention-Based Deep Reinforcement Learning. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY, v. 36, n. 4, p. 741–761, 2021. ISSN 1000-9000.

HALFPAP, S.; SCHLOSSER, R. Workload-Driven Fragment Allocation for Partially Replicated Databases Using Linear Programming. 35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019, 2019. ISSN 1084-4627. Disponível emhttps://doi.org/10.1109/ICDE.2019.00188.

HEWASINGHAGE, M. et al. Automated database design for document stores with multicriteria optimization. KNOWLEDGE AND INFORMATION SYSTEMS, v. 65, n. 7, p. 3045–3078, 2023. ISSN 0219-1377.

HUANG, H. et al. Sibyl: Forecasting Time-Evolving Query Workloads. Proc. ACM Manag. Data, v. 2, n. 1, 2024. Disponível em: https://doi.org/10.1145/3639308.

JACCARD, P. Étude comparative de la distribution florale dans une portion des Alpes et du Jura. Bulletin de la Société Vaudoise des Sciences Naturelles, v. 37, n. 142, p. 547, 1901. ISSN 0037-9603.

JARQUE, C. M.; BERA, A. K. Efficient tests for normality, homoscedasticity and serial independence of regression residuals. Economics letters, Elsevier, v. 6, n. 3, p. 255–259, 1980.

KARYPIS, G.; KUMAR, V. Multilevel k-way hypergraph partitioning. In: Proceedings of the 36th Annual ACM/IEEE Design Automation Conference. New York, NY, USA: Association for Computing Machinery, 1999. (DAC '99), p. 343–348. ISBN 1581131097. Disponível em: https://doi.org/10.1145/309847.309954.

KAZANAVICIUS, J.; MAZEIKA, D.; KALIBATIENE, D. An Approach to Migrate a Monolith Database into Multi-Model Polyglot Persistence Based on Microservice Architecture: A Case Study for Mainframe Database. APPLIED SCIENCES-BASEL, v. 12, n. 12, 2022.

KHINE, P. P.; WANG, Z. A Review of Polyglot Persistence in the Big Data World. INFORMATION, v. 10, n. 4, 2019. ISSN 2078-2489.

KITCHENHAM, B. et al. Systematic literature reviews in software engineering–a systematic literature review. Information and software technology, Elsevier, v. 51, n. 1, p. 7–15, 2009.

KOLONKO, M.; MUELLENBACH, S. Polyglot Persistence in Conceptual Modeling for Information Analysis. 2020 10TH INTERNATIONAL CONFERENCE ON ADVANCED COMPUTER INFORMATION TECHNOLOGIES (ACIT), p. 590–594, 2020.

KOŠMERL, I.; RABUZIN, K.; ŠESTAK, M. Multi-Model Databases - Introducing Polyglot Persistence in the Big Data World. 2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO), p. 1724–1729, 2020. ISSN 2623-8764.

KUNJIR, M. et al. ROBUS: Fair Cache Allocation for Data-parallel Workloads. Proceedings of the 2017 ACM International Conference on Management of Data, p. 219–234, 2017. Disponível em: https://doi.org/10.1145/3035918.3064018.

LAJAM, O.; MOHAMMED, S. Revisiting Polyglot Persistence: From Principles to Practice. INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS, v. 13, n. 5, p. 872–882, 2022. ISSN 2158-107X.

LASO, S. et al. Energy consumption and workload prediction for edge nodes in the Computing Continuum. Sustainable Computing: Informatics and Systems, v. 46, p. 101088, 2025. ISSN 2210-5379.Disponível em: https://www.sciencedirect.com/science/article/pii/S2210537925000083.

LEBERKNIGHT, S. Scott Leberknight's Weblog. 2008. Disponível em: http://www.sleberknight.com/blog/sleberkn/entry/polyglot_persistence.

LI, Y.; YU, X.; KOUDAS, N. Les3: learning-based exact set similarity search. Proceedings of the VLDB Endowment, Association for Computing Machinery (ACM), v. 14, n. 11, p. 2073–2086,2021. ISSN 2150-8097.Disponível em: http://dx.doi.org/10.14778/3476249.3476263.

LIMA, C. de; MELLO, R. dos S. A workload-driven logical design approach for NoSQL document databases. In:ANDERST-KOTSIS, G.; INDRAWAN-SANTIAGO, M. (Ed.). Proceedings of the 17th International Conference on Information Integration and Web-based Applications & Services, iiWAS 2015, Brussels, Belgium, December 11-13, 2015. New York, NY, USA: Association for Computing Machinery, 2015. (iiWAS '15). ISBN 978-1-4503-3491-4.Disponível em: https://doi.org/10.1145/2837185.2837218.

LUEBCKE, A.; KOEPPEN, V.; SAAKE, G. Heuristics-Based Workload Analysis for Relational DBMSs. INFORMATION SYSTEMS: METHODS, MODELS, AND APPLICATIONS, UNISCON 2012, v. 137, p. 25–36, 2013. ISSN 1865-1348.

MCQUEEN, J. B. Some methods of classification and analysis of multivariate observations. In: Proc. of 5th Berkeley Symposium on Math. Stat. and Prob. [S.l.: s.n.], 1967. p. 281–297.

MELLO, R. D. S. et al. Uma Proposta de Modelagem de Dados no Domínio de Fraudes em Licitações Públicas. In: Anais Estendidos do XXXIX Simpósio Brasileiro de Banco de Dados (SBBD Estendido 2024). Brasil: Sociedade Brasileira de Computação - SBC, 2024. p. 220–226.Disponível em: https://sol.sbc.org.br/index.php/sbbd_estendido/article/view/30797.

MUELLER, S. et al. Workload-Aware Aggregate Maintenance in Columnar In-Memory Databases. 2013 IEEE INTERNATIONAL CONFERENCE ON BIG DATA, 2013. ISSN 2639-1589.

NGUYEN, C. D. T. et al. Are Database System Researchers Making Correct Assumptions about Transaction Workloads? Proc. ACM Manag. Data, Association for Computing Machinery, v. 3, n. 3, 2025. Disponível em: https://doi.org/10.1145/3725268.

PADIYA, T.; KANWAR, J. J.; BHISE, M. Workload Aware Hybrid Partitioning. Proceedings of the 9th Annual ACM India Conference, p. 51–58, 2016. Disponível em: https://doi.org/10.1145/2998476.2998479.

PAUL, D. et al. Database Workload Characterization with Query Plan Encoders. PROCEEDINGS OF THE VLDB ENDOWMENT, v. 15, n. 4, p. 923–935, 2021. ISSN 2150-8097. Disponível emhttps://doi.org/10.14778/3503585.3503600.

PERALTA, V. et al. Detecting coherent explorations in SQL workloads. INFORMATION SYSTEMS, v. 92, 2020. ISSN 0306-4379.

PEREIRA, F.; GUERRA, E.; ROSA, R. R. Patterns for Polyglot Persistence Layer. Proceedings of the 29th Conference on Pattern Languages of Programs, 2023.

POKORNY, J. Integration of Relational and NoSQL Databases. VIETNAM JOURNAL OF COMPUTER SCIENCE, v. 6, n. 4, p. 389–405,2019. ISSN 2196-8888.

QUAMAR, A.; KUMAR, K. A.; DESHPANDE, A. SWORD: scalable workload-aware data placement for transactional workloads. Proceedings of the 16th International Conference on Extending Database Technology, p. 430–441,2013. Disponível em: https://doi.org/10.1145/2452376.2452427.

RAZA, B. et al. Autonomic workload performance tuning in large-scale data repositories. KNOWLEDGE AND INFORMATION SYSTEMS, v. 61, n. 1, p. 27–63, 2019. ISSN 0219-1377.

RENIERS, V. et al. A Workload-Driven Document Database Schema Recommender (DBSR) - Conceptual Modeling. Conceptual Modeling - 39th International Conference, ER 2020, Vienna, Austria, November 3-6, 2020, Proceedings, v. 12400, 2020. Disponível em:https://doi.org/10.1007/978-3-030-62522-1 $_3$5.

ROY-HUBARA, N.; SHOVAL, P.; STURM, A. Selecting databases for Polyglot Persistence applications. Data & Knowledge Engineering, v. 137, p. 101950, 2022. ISSN 0169-023X.

SACHDEVA, S.; VASAVA, J. Plugging and Playing with Variety of Data using Multi-Model Database and Polyglot Persistence. 2024 Third International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT), p. 1–8, 2024.

SADALAGE, P. J.; FOWLER, M. NoSQL distilled: a brief guide to the emerging world of polyglot persistence. [S.l.]: Pearson Education, 2013.

SANTANA, L. H. Z.; MELLO, R. d. S. A Middleware for Polyglot Persistence of RDF Data into NoSQL Databases. 2019 IEEE 20th International Conference on Information Reuse and Integration for Data Science (IRI), p. 237–244, 2019.

SELLAMI, R.; BHIRI, S.; DEFUDE, B. Supporting Multi Data Stores Applications in Cloud Environments. IEEE TRANSACTIONS ON SERVICES COMPUTING, v. 9, n. 1, p. 59–71, 2016. ISSN 1939-1374.

SEN, R.; RAMACHANDRA, K. Characterizing Resource Sensitivity of Database Workloads. IEEE International Symposium on High Performance Computer Architecture, HPCA 2018, Vienna, Austria, February 24-28, 2018, 2018. Disponível em: https://doi.org/10.1109/HPCA.2018.00062.

SHAHEEN, N.; RAZA, B.; MALIK, A. K. A CBR Model for Workload Characterization in Autonomic Database Management System. In: 2018 14th International Conference on Emerging Technologies (ICET). [s.n.], 2018. p. 1–6. Disponível em: https://ieeexplore.ieee.org/document/8603615/.

SHEIKH, M. B. et al. A bayesian approach to online performance modeling for database appliances using gaussian modelsProceedings of the 8th ACM International Conference on Autonomic Computing, p. 121–130,2011. Disponível em: https://doi.org/10.1145/1998582.1998603.

SINGHAL, H. et al. PolyGlot Persistence for Microservices-Based Applications. INTERNATIONAL JOURNAL OF INFORMATION TECHNOLOGIES AND SYSTEMS APPROACH, v. 14, n. 1, p. 17–32, 2021. ISSN 1935-570X.

SRIVASTAVA, K.; SHEKOKAR, N. A Polyglot Persistence approach for E-Commerce business model. 2016 International Conference on Information Science (ICIS), p. 7–11, 2016.

TELANG, A.; CHAKRAVARTHY, S.; LI, C. Personalized ranking in web databases: establishing and utilizing an appropriate workload. Distributed Parallel Databases, v. 31, n. 1, p. 47–70,2013. ISSN 0926-8782.Disponível em: https://doi.org/10.1007/s10619-012-7106-2.

WU, W. et al. Towards predicting query execution time for concurrent and dynamic database workloads.Proc. VLDB Endow., v. 6, n. 10, 2013. ISSN 2150-8097. Disponível em: https://doi.org/10.14778/2536206.2536219.

YANG, W. et al. Hepart: A balanced hypergraph partitioning algorithm for big data applications. Future Generation Computer Systems, v. 83, p. 250–268, 2018. ISSN 0167-739X. Disponível em: https://www.sciencedirect.com/science/article/pii/S0167739X17316473.

ZHOU, Y. et al. PBench: Workload Synthesizer with Real Statistics for Cloud Analytics Benchmarking. Proc. VLDB Endow., VLDB Endowment, v. 18, n. 11, p. 3883–3895, 2025. ISSN 2150-8097. Disponível em:https://doi.org/10.14778/3749646.3749661.