# Towards Designing Conceptual Data Models for Big Data Warehouses: The Genomics Case

João Galvão[1[0000-0003-4263-8726]], André Leite[1 [0000-0003-4263-8726]],
Ana León Palacio[2[0000-0003-3516-8893]], Carlos Costa[1[0000-0003-0011-6030]],
Maribel Yasmina Santos[1[0000-0002-3249-6229]], Óscar Pastor López[2[0000-0002-1320-8471]]
[1]ALGORITMI Research Centre, University of Minho, Guimarães, Portugal
{joao.galvao,andre.leite,carlos.costa,maribel}@dsi.uminho.pt
[2] Research Center on Software Production Methods (PROS), Universitat Politècnica de València, Valencia, Spain
{aleon,opastor}@pros.upv.es

**Abstract.** Data Warehousing applied in Big Data contexts has been an emergent topic of research, as traditional Data Warehousing technologies are unable to deal with Big Data characteristics and challenges. The methods used in this field are already well systematized and adopted by practitioners, while research in Big Data Warehousing is only starting to provide some guidance on how to model such complex systems. This work contributes to the process of designing conceptual data models for Big Data Warehouses proposing a method based on rules and design patterns, which aims to gather the information of a certain application domain mapped in a relational conceptual model. A complex domain that can benefit from this work is Genomics, characterized by an increasing heterogeneity, both in terms of content and data structure. Moreover, the challenges for collecting and analyzing genome data under a unified perspective have become a bottleneck for the scientific community, reason why standardized analytical repositories such as a Big Genome Warehouse can be of high value to the community. In the demonstration case presented here, a genomics relational model is merged with the proposed Big Data Warehouse Conceptual Metamodel to obtain the Big Genome Warehouse Conceptual Model, showing that the design rules and patterns can be applied having a relational conceptual model as starting point.

**Keywords:** Big Data Warehousing, Big Data Modelling, Conceptual Modeling.

## 1 Introduction

Analytical contexts have been highly influenced by Big Data where new challenges arise both in terms of data modeling approaches and technological concerns that must be considered. Traditional Data Warehousing (DWing) systems lost the capacity to handle data with different characteristics, such as high data volumes, produced at high speed and considering different data varieties. To overcome those challenges, as organizations still need structured data repositories supporting decision making tasks, data warehouses are now implemented using Big Data technologies [1, 2].

Due to its novelty, research in Big Data Warehousing (BDWing) has been quite scarce with some works based on unstructured approaches and use case technology-driven solutions [2–5]. The work of [2] overcomes these practices by proposing a structured approach that includes guidelines for the design and implementation of Big Data Warehouses (BDWs). In this paper, the conceptual modeling of BDWs is formalized defining a Big Data Warehouse Conceptual Metamodel ($BDW_{CMT}$) with the constructs made available in the data modeling approach of [2], and proposes a method that includes patterns and rules to guide practitioners from the $BDW_{CMT}$ to the design of a specific Big Data Warehouse Conceptual Model ($BDW_{CM}$).

As demonstration case, and due to the complexity of this application domain, the proposed modeling method is applied to Genomics with the aim to implement the Big Genome Warehouse System. This is intended to be implemented using Big Data tools and technologies in the Hadoop Ecosystem, using Hive as the main storage technology, as this is considered the *de facto standard* for DWing in Big Data. This physical implementation in Hive must consider background knowledge inherited from the Big Genome Warehouse Conceptual Model ($BGW_{CM}$) and from the Human Genome Conceptual Model ($HG_{CM}$). Both models comply with specific constructs that follow UML Class Diagrams Metamodels, assuming that a system is represented by a model that conforms to a metamodel [6].

The $HG_{CM}$ is the result of a research work on a complex domain where the use of conceptual models has been proved to be a feasible solution for the integration of data coming from heterogeneous and disperse set of genomic sources. One of the most challenging problems in the genomic domain is the identification of DNA variants that could be a potential cause of disease. The huge amounts of available data, characterized by their heterogeneity, either in terms of content and structure, as well as the problems for collecting and analyzing them under a unified perspective has become a bottleneck for the scientific community. In [7], the authors face this problem by presenting a conceptual model that provides the required unified perspective to collect, structure and analyze the key concepts of the domain under a well-grounded ontological basis. Using this background knowledge of the $HG_{CM}$ with the key concepts in this application domain, namely the main identified identities and their relationships, the $BGW_{CM}$ here modeled must conform to the $BDW_{CMT}$. The aim of this paper is to show how to move from the $BDW_{CMT}$ to the $BGW_{CM}$ using the knowledge explicitly available in the $HG_{CM}$. A simplification of the $HG_{CM}$ is used in this paper intended to solve a specific task and ease the validation process of this approach. The details about the physical implementation of the $BGW_{CM}$ in Hive is out of the scope of this paper.

The method for data modeling of BDWs proposed in this paper follows an iterative and goal-driven approach that performs a map between the conceptual model of the domain ($HG_{CM}$) and the $BDW_{CMT}$. This method considers the main data modeling constructs and proposes the data modeling rules and the data modeling patterns for implementing BDWs. This work is evaluated with the identification of the $BGW_{CM}$.

This paper is structured as follows. Section 2 presents the related work. Section 3 formalizes the $BDW_{CMT,}$ its main constructs and their characteristics. Section 4 addresses the proposed data modeling rules and patterns, which are instantiated to the Genomics case. Section 5 outlines the presented work and future work.

## 2 Related Work

Data models are essential in information systems design and development as they ensure that data needs are properly considered [4]. In a traditional organizational environment, relational data models are quite popular and are strictly considering the business requirements. However, in an organizational context making use of Big Data, the ability to process data increases with the use of flexible schemas, and thus the data modeling methods change significantly [8, 9], as the database schemas can change during application runtime according to the analytical needs of the organization [4, 9]. Taking this into consideration, BDWs are significantly different from traditional data warehouses, since schemas must be based on new logical models that allow more flexibility and scalability, hence the emergence of new design and modeling proposals for BDWs [2]. In Big Data, there are multiple challenges when addressing multidimensional data, namely the capability to ensure schema-less or dynamic schema changes, huge number of dimensions and cardinality, recommendations for automatic partitioning and materialization, or real-time processing [3, 10].

The works of [4] and [11] propose an almost automatic design methodology using the key-value model to represent a multidimensional scheme at the logical level, instead of applying the traditional star/snowflakes schemes. Moreover, a multidimensional model is provided by a graph-oriented representation as the basis for the conceptual design of this methodology, aiming at the construction of attribute trees representing facts related to the integrated data source and automatically remodeling these trees based on the restrictions resulting from the requirements analysis phase. Still in the NoSQL realm, the work of [5] proposes three types of translations of a conceptual model to a columnar model, showing the implementation of columnar data warehouses in NoSQL. Additionally, there are works focused on OLAP-oriented technologies for Big Data, being Hive a popular example. The work of [12] proposes a set of rules/guidelines for transforming a traditional dimensional model [13] into a Hive tabular data model for BDWs, adjusting the table's grain to the domain requirements.

A context-independent design and implementation approach for BDWs has been addressed in [2, 14] where several design patterns for modeling performant BDWs were evaluated, targeting advancing decision making with huge amounts of data, collected at high velocity and with different degrees of heterogeneity. In these, a data modeling method is proposed, supporting mixed and complex analytical workloads (e.g., streaming analysis, ad hoc querying, data visualization, data mining).

Research in this area is still relatively ambiguous and yet at an early stage, lacking common approaches [15], reason why this paper proposes a more straightforward rationale for modelling BDWs, supported by the constructs of [2].

## 3 The Big Data Warehouse Conceptual Metamodel

In [2], the set of constructs for modelling a BDW were proposed without any formalization in a conceptual metamodel that clearly states how those constructs are organized and how they complement each other in this data system. Extending the work of [2],

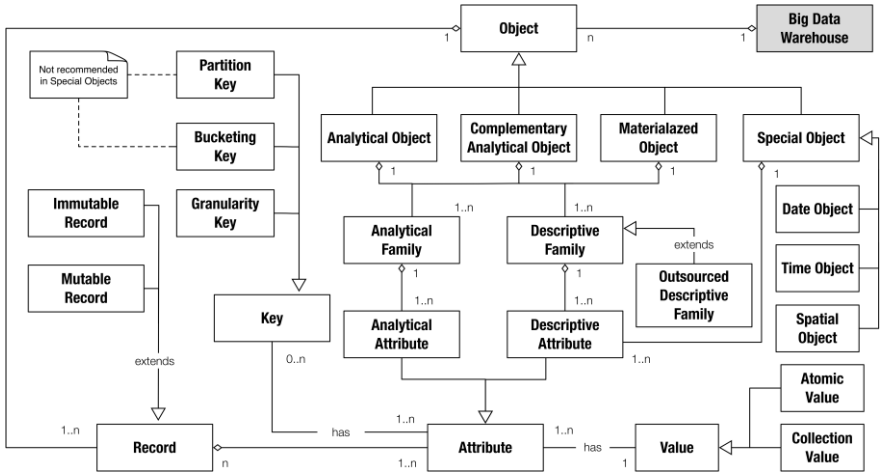


**Fig. 1.** The BDW's Conceptual Metamodel

those constructs are in this paper used to propose the $BDW_{CMT}$, a metamodel that formalizes the elements used to model a BDW (**Fig. 1**).

Each object in a BDW can be classified as an Analytical Object (AO), a Complementary Analytical Object (CAO), a Materialized Object (MO) or a Special Object (SO). The AOs are subjects of interest in an analytical context, for instance Sales or Products. Usually, objects start as AOs, but as the data modelling approach proceeds, they can turn into CAOs if they are shared by several AOs and if they comply with the characteristics and guidelines presented afterwards in this section. In these cases, the AOs outsource the descriptive families to the CAOs. In a parallelism with Kimball's Dimensional Modeling [13], this type of object is similar to the conformed dimensions of a dimensional data model but capable of combining the concept of aggregated facts for dimensions as well, due to the analytical value of CAOs. The MOs store the results of time-consuming queries aiming to improve the BDW performance by providing pre-aggregated results to the several data consumers, instead of processing heavy aggregations multiple times. Finally, the SOs are of three types: Spatial, Date and Time. They are used to standardize concepts like Date, Time and Space, ensuring that the attributes related to those concepts have the same meaning and format across the BDW. The several objects, excluding the SOs as these are seen as a particular case, include analytical and/or descriptive families, which in their turn include analytical and descriptive attributes, respectively. The attributes include atomic or collection values. The atomic values can be used as (or as part of) the partition, bucketing and/or granularity keys of an object. The records of the objects can be mutable or immutable, allowing or avoiding update operations. These several constructs included in the conceptual metamodel are described in **Table 1**. Extending the descriptive families, outsourced descriptive families allow relationships between AOs and CAOs. These are useful for having a flexible data modelling approach that enhances the performance of the BDW [14].

**Table 2** summarizes a set of guidelines and good practices proposed in [2] for the use of outsourced descriptive families and nested attributes taking into consideration

the domain requirements, helping practitioners to identify contexts where the same can be useful.

**Table 1.** Description of the BDW's Conceptual Metamodel Constructs

| Construct | Class | Description |
|---|---|---|
| **Object** | Represents | Constructs used in the conceptual modeling of a BDW |
| | Characteristics | Highly performant structures to provide analytical value to decision support scenarios |
| | Includes | Analytical, complementary analytical, materialized, and special (date, time and spatial) objects |
| **Analytical Object** | Represents | Isolated object of a subject of interest for analytical purposes |
| | Characteristics | Highly denormalized and autonomous structures able to answer queries without the constant need of joins with other data structures |
| | Examples | Sales, purchases, inventory management, customer complaints, among others |
| | Includes | Analytical families, descriptive families, records, granularity key, partition key, bucketing (or clustering) key |
| **Analytical Family** (including Analytical Attributes) | Represents | A set of attributes with numeric values that can be analyzed using different descriptive attributes (e.g., grouped or filtered by) |
| | Characteristics | Logical representation of a set of indicators or measures (analytical attributes) relevant for analytical purposes. Can include factual (numeric evidence of something) or predictive (an estimative or a prediction of what could happen) attributes |
| | Examples | Sold quantity, discount value and sold value |
| | Includes | Analytical families include analytical attributes |
| **Descriptive Family** (including Descriptive Attributes) | Represents | A set of descriptive values that are used to interpret analytical attributes by different perspectives, using aggregation or filtering operations, for example |
| | Characteristics | A descriptive family is a logical representation of a set of attributes usually used to add meaning to a numeric indicator |
| | Examples | Customer name, product description and discount type |
| | Includes | Descriptive families include descriptive attributes |
| **Record** | Represents | The set of values for the attributes of an occurrence of an analytical object |
| | Characteristics | Can be mutable (allow updates) or immutable records (forbid updates) |
| | Examples | The values that characterize the purchase of a product, by a customer, on a store, with a factual and/or predicted quantity |
| | Includes | Atomic values (integer, float, double, string, or varchar) or collections (complex structures like arrays, maps, or JSON) |
| **Granularity Key** | Represents | The level of detail of records to be stored in an analytical object |
| | Characteristics | Is defined using one or more descriptive attributes that uniquely identify a record. It may not need to be physically implemented in a data system as a primary key |
| | Examples | Sales order, product identifier, among others |
| | Includes | One or more descriptive attributes that uniquely identify a record |
| **Partition Key** | Represents | The physical partitioning scheme applied to the data, fragmenting the analytical objects into more manageable parts that can be accessed individually |
| | Characteristics | Is defined using one or more descriptive attributes (although analytical attributes can also be used) that form the partition key |
| | Examples | Time and/or geospatial attributes are the most useful ones, as data is typically loaded and filtered in hourly/daily/monthly batches for specific regions or countries |
| | Includes | One or more descriptive attributes that form the partition key |

| Construct | Class | Description |
|---|---|---|
| **Bucketing Key** | Represents | The physical clustering applied to the data, grouping records of an analytical object |
| | Characteristics | Is defined using one or more descriptive attributes that form the bucketing key |
| | Examples | Attributes such as products or customers distributing the data by similar volumes |
| | Includes | One or more descriptive attributes that form the bucketing key |
| **Complementary Analytical Object** | Represents | Object that complements other analytical objects, providing an autonomous structure with analytical value that is used to complement the different analytical perspectives provided by the analytical objects |
| | Characteristics | Object whose granularity key (whole or part of it) is used by other analytical object, meaning that a join between two or more objects is possible |
| | Examples | Customer account, product, supplier, among others |
| | Includes | Analytical families, descriptive families, records, granularity key, partition key, bucketing (or clustering) key |
| **Materialized Object** | Represents | Object that includes an aggregation of the records of an analytical or complementary analytical object, based on frequent access patterns to the data |
| | Characteristics | Enhances the performance of frequent queries by performing a pre-aggregation of the data and the pre-computing time-consuming joins between large objects |
| | Examples | Views on any analytical, complementary analytical and special objects |
| | Includes | Can be created based on any analytical or complementary analytical objects |
| **Special Object** (Time, Date and Spatial Object) | Represents | Objects that include several temporal and/or spatial attributes that complement the analytical objects (or complementary analytical objects). |
| | Characteristics | Use standard time, date and spatial representations in autonomous objects, avoiding the increase of the size of the analytical or complementary analytical objects |
| | Examples | Time: hour, minute, second; Date: day, month, year; Spatial: city, country. |
| | Includes | Descriptive families, records, and Granularity keys |

**Table 2.** Guidelines for Outsourced Descriptive Families and Nested Attributes

| Construct | Guidelines |
|---|---|
| **Outsourced Descriptive Family** | The descriptive family is frequently included in other analytical objects |
| | The descriptive family has low cardinality, i.e., its distinct records will form a low volume CAO that easily fits into memory, enabling the capability to perform map/broadcast joins in SQL-on-Hadoop engines |
| | The data ingestion frequency of the resulting CAO is equivalent to the other AOs it is related to |
| | The CAO resulting from the outsourced descriptive family can provide analytical value by itself |
| | The records of the CAOs formed by the outsourced descriptive families are recommended to be immutable |
| **Nesting Attributes (in a Collection)** | Avoid nested attributes in a collection if there is the need to perform heavy aggregations on that data |
| | Avoid nested attributes in a collection when using filtering operations based on nested values |
| | Nested attributes included in a collection are not meant to grow rapidly |
| | Estimate the collection initial size and its potential growth before adopting nested attributes |

# 4 The Big Genome Warehouse Conceptual Model

The method presented in this paper is based on rules and design patterns that aim to gather the information of a certain application domain mapped in a relational conceptual model, merging it with a $BDW_{CMT}$ in order to obtain a $BDW_{CM}$. In this work, the method is presented and demonstrated in the Genomics application domain.

### 4.1 Data Modeling Rules

The data modeling rules are based on a goal-driven approach that identifies the analytical value of the entities present in the conceptual model of the domain and characterizes the data volume and querying frequency of those entities, as this information is later used for applying the data modeling patterns. The three data modeling rules are:

⊙ **R1. Entities with High Analytical Value**. Identification of the main entities of the domain, pointing those queried for decision support, providing the main business or analytical indicators. In this process, it is relevant to consider that the approach is goal-driven, so an entity may have a relevant analytical value in a particular domain or business process, being identified as a key entity by this rule, but in other contexts it may only be used for providing contextual information, such as *who* or *what*. In the data modelling perspective, these entities usually receive multiple relationships with the M cardinality (M:1, many-to-one), integrating several concepts of the domain.

↗ **R2. Entities with High Cardinality**. Characterization of entities with high data volume, helping in the process of identifying the entities that are candidates for outsourced descriptive families, since high cardinality entities are good candidates to denormalization processes, avoiding joins with huge amounts of data. This high cardinality classification of the entities cannot be exclusively based on row counting processes, requiring additional knowledge from the domain. The data engineer with the help of the domain expert should estimate data growing rates based on a deep knowledge of the application domain.

↻ **R3. Entities with Frequent Access Patterns**. Characterization of the entities with frequent access patterns which, combined with R2, point entities that are candidates for MOs, increasing the overall performance of the BDW system.

Taking into consideration the $HG_{CM}$ available at [7] and the constructs and guidelines presented above, the data modeling rules were applied classifying a subset of the entities of this domain attending to R1, R2 and R3 (**Fig. 2**).

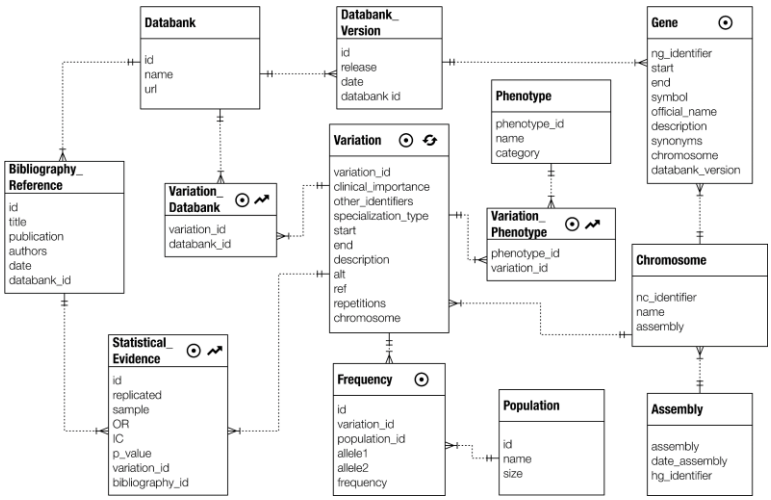


**Fig. 2.** The Human Genome Conceptual Model. Adapted from [7].

### 4.2 Data Modeling Patterns

The data modeling patterns take into consideration that a BDW is built with the goal of supporting decision-making tasks and that those tasks highly depend on the identified goals or main analytical activities.

This data modeling method has the flexibility that is needed in a Big Data context, allowing the evolution of the models when: i) new business processes/data sources are identified; ii) new data is available for the existing processes/data sources; or iii) new data requirements change the classification of the entities in terms of design rules. As described in section 3, one of the main constructs in the $BDW_{CMT}$ is the concept of AOs. These are highly denormalized and autonomous structures able to answer queries without the constant need of joining different structures. Often based on flat structures, for better performance [14], these completely or mostly flat structures significantly increase the storage size of the BDW, a problem that has even more impact when multiple AOs share the same descriptive families. To face this balance between data volume and processing performance, the proposed data modeling patterns allow for the identification of data models that are: i) highly flexible, as the data engineers have instruments that guide the modeling process, without limiting the human decisions; ii) highly performant, identifying objects that answer the main domain questions considering both data volume and performance concerns; and iii) highly relevant, providing different analytical views on the data under analysis. The design patterns take into consideration the need to identify the different objects in the BDW, their type (AOs, CAOs, MOs or SOs), and the descriptive and analytical families included in those objects.

Considering a traditional relational context in which data is highly normalized and each entity details a specific set of attributes with some level of detail, a BDW uses the same data but denormalizes the data structures as much as possible, without compromising the BDW sustainability in terms of storage space or its usability in terms of performance. In this process, data at different levels of detail can be stored, making available objects that may answer more detailed queries, while others can support aggregated and very performant answers to more general questions (**Fig. 3**).

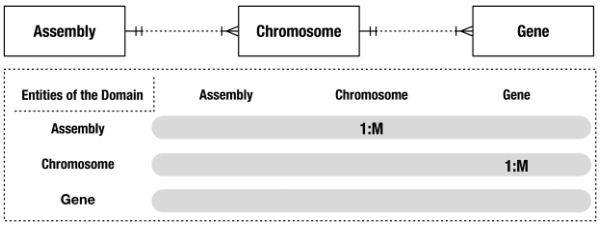


**Fig. 3.** Levels of Detail for the several Entities

In **Fig. 3**, the three entities available in the domain are possible AOs for the BDW and all of them could be included in this data repository using a similar data model (highly normalized). However, the BDW must be aligned with the analytical queries and must consider storage and performance concerns. As an example for a BDW, two possible data models are depicted in **Fig. 4**.

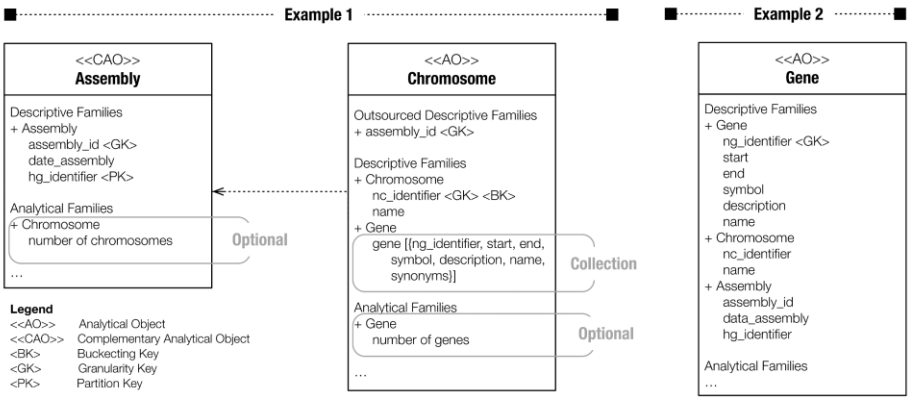


**Fig. 4.** Examples of possible Data Models

In **Fig. 4**, Example 1 includes **Chromosome** as an AO that outsourced a descriptive family, included in a CAO named **Assembly** and nested the descriptive family of **Gene** into a collection, in accordance with the **Chromosome** granularity. Each object has its own granularity and the denormalization must respect that granularity. Example 2 includes a fully denormalized AO called **Gene** with **Chromosome** and **Assembly** as denormalized descriptive families of this object.

Based on the assumption that all the entities present in a data model, such as a relational-based one, are candidate or possible objects in a BDW, the design patterns are:

<<AO>> **P1**. **Analytical Objects**. Entities classified as of type R1 are identified as possible AOs due to their high analytical value.

<<CAO>> **P2. Complementary Analytical Objects**. Entities can be outsourced to CAOs if they comply with the best practices summarized in **Table 2**, if they are not classified as of type R2 and if they maintain relationships of cardinality 1:M (one-to-many) with more than one entity of the domain.

<<DF>> **P3. Descriptive Families**. Entities not identified as <<AO>> or <<CAO>> by the design patterns P1, P2 are candidates to be denormalized as descriptive families of AOs or nested to a collection of AOs, in accordance to their granularity. In **Fig. 4**, Example 1, **Chromosome** includes **Gene** as a collection, as a chromosome includes multiples genes, whereas in Example 2 **Gene** denormalizes **Chromosome** and **Assembly**, as a gene maintains a unitary relationship with a chromosome and an assembly.

<<SO>> **P4. Special Objects**. Entities including temporal and/or spatial attributes point the need for SOs that include the calendar, temporal or spatial descriptive attributes relevant in the application domain.

<<MO>> **P5. Materialized Objects**. Entities of type R3 can be, in addition to the previous patterns, labeled as possible MOs with aggregates usually used in analytical tasks.

Considering the defined patterns and the domain knowledge expressed in the $HG_{CM}$ presented in **Fig. 2**, which already includes the classification of the entities considering the data modeling rules, it is now possible to integrate the specific characteristics of a

BDW with the domain knowledge of the human genome in order to propose the conceptual model of the Big Genome Warehouse ($BGW_{CM}$).

**Fig. 5** presents a synopsis of the approach that guides practitioners to apply the method and obtain the $BGW_{CM}$. The entities identified in the domain knowledge are mapped against themselves in a matrix, to identify their common relationships. It also maps the design rules and patterns with the entities of the domain. The application of the data modeling patterns gives a first overview of the main objects of the BDW, which are refined in successive iterations as the data modeling method proceeds.

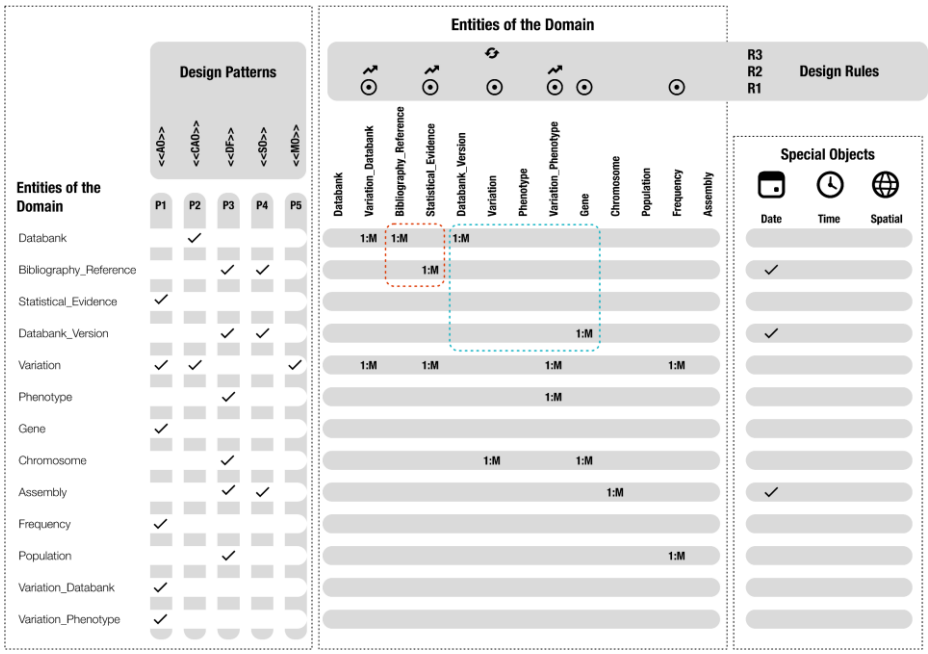


**Fig. 5.** Applying the Design Rules and Patterns to the $HG_{CM}$

In the first step, the application of pattern P1 allows the identification of 6 analytical objects (<<AO>> Variation_Databank, <<AO>> Statistical_Evidence, <<AO>> Variation, <<AO>> Variation_Phenotype, <<AO>> Frequency, <<AO>> Gene). The second step identifies the CAOs, starting by choosing the entities that have more than one 1:M relationships to other entities. In this case, the possible CAOs are Databank, Variation and Chromosome, but only Databank and Variation are classified as CAOs (<<CAO>> Databank, <<CAO>> Variation) since Chromosome alone cannot provide analytical value, complying with the best-practices presented in **Table 2**. Note that the classification as <<CAO>> can change if the size of the object makes joins inefficient. Although Databank was not classified by R1, domain experts may show interest in knowing the databanks that are not used in the study of a variation. This is possible with a query that checks records of <<CAO>> Databank not included in <<AO>> Variation_Databank, for instance. In this step, one object previously classified as an AO is now reclassified as a CAO (<<CAO>> Variation) due to its use by other objects.

With the identification of AOs and CAOs, the entities without these classifications are candidates to be denormalized to the AOs or CAOs previously identified (P3), in accordance to their granularity. The fourth step identifies the entities that have relationships with the SOs, namely Date, Time or Spatial. In this, Bibliography_Reference, Assembly and Databank_Version are identified as having relationships with the Date object (P4). In P5, and due to its frequent access pattern, one object is identified as possible candidate to an additional materialized object that answers frequent queries of the application domain, <<MO>> Variation Aggregates.

Following the design patterns, the $BGW_{CM}$ is obtained (**Fig. 6**). With P1 and P2, 5 AOs and 2 CAOs (<<AO>> Variation_Databank, <<AO>> Statistical_Evidence, <<AO>> Variation_Phenotype, <<AO>> Frequency, <<AO>> Gene, <<CAO>> Databank, <<CAO>> Variation) were identified. Now, the relationships of the entities are analyzed.

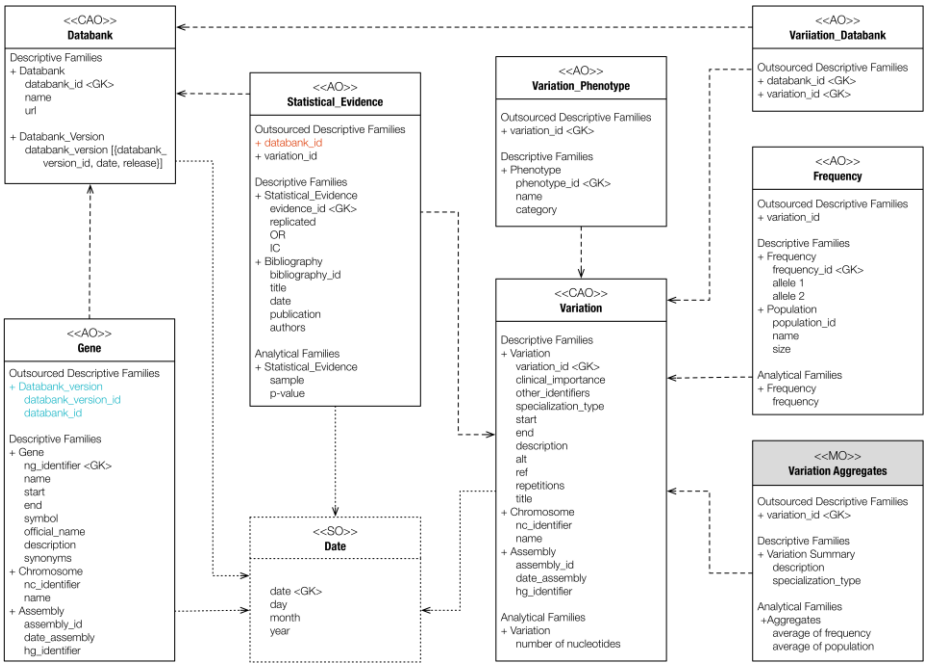


**Fig. 6.** The Big Genome Warehouse Conceptual Model

Starting by Databank, this entity has relationships with Variation_Databank, Bibliography_Reference and Databank_Version, linking <<CAO>> Databank and <<AO>> Variation_Databank. Taking into consideration the domain knowledge, Databank_Version is an entity with a set of properties of Databank which, also, complies with the nested attributes best practices presented in **Table 2**. For this, Databank_Version is nested and included in the model as a collection of the <<CAO>> Databank. These decisions are left to the data engineer, as the approach is meant to be flexible enough to accommodate the data analysis requirements in the data model that best suits those analytical needs, while complying with the defined data modeling rules and patterns.

Variation_Databank does not have any additional relationship, besides the ones inherited from the application domain, outsourcing the descriptive families of <<CAO>> Databank and <<CAO>> Variation. Although this AO only contains outsourced descriptive families without additional descriptive or analytical families, it could be used to perform event tracking analyses, like the variations available in some databanks, or as a coverage table that with <<CAO>> Databank allow the identification of databanks that do not include a specific variation (as already pointed). These are usually known as factless fact tables in the Kimball's Dimensional Modeling [13] approach. The same result, with a different approach, can be achieved with a collection of Databank inside <<CAO>> Variation, if the relationship between Variation and Databank is not of high cardinality.

Bibliography_Reference is a candidate to denormalization. As it maintains a relationship with Statistical_Evidence, the information of the entity Bibliography_Reference is denormalized to the <<AO>> Statistical_Evidence as a descriptive family of this object. Additionally: i) as Databank and Bibliography_Reference are related in the application domain, and as Bibliography_Reference is denormalized to the <<AO>> Statistical_Evidence, a relationship between <<AO>> Statistical_Evidence and <<CAO>> Databank is established. This scenario is depicted in red in **Fig. 5** and in **Fig. 6**; ii) as the entity Databank_Version has a relationship with Gene, but it was nested to the <<CAO>> Databank, this object will inherit the relationships of the entity Databank_Version, reason why <<CAO>> Databank is related with <<AO>> Gene, having databank_id and databank_version_id as outsourced descriptive families. This scenario is depicted in blue in **Fig. 5** and in **Fig. 6**; iii) as the entity Variation and all its related entities are already present in the model as objects (independently of their type), there is only the need to establish the relationships between the objects.

Phenotype is denormalized to the <<AO>> Variation_Phenotype, Chromosome is denormalized both to the <<CAO>> Variation and <<AO>> Gene, while Population is denormalized to the <<AO>> Frequency. Lastly, Assembly is denormalized to Chromosome, that was previously included in the objects <<CAO>> Variation and <<AO>> Gene.

Regarding MOs, only Variation is used to propose a MO as an example, <<MO>> Variation Aggregates, which maintains a relationship with <<CAO>> Variation to access details of Variations, in case those are needed.

For the analytical attributes, those can be available in the domain conceptual model or can be created/derived by practitioners in accordance to the queries that need to be answered. Besides the explicit relationships included in the obtained data model (**Fig. 6**), those linking AOs, CAOs, MOs and SOs, implicit relationships exist between these objects, allowing join operations between objects that share common attributes. For instance, to know how the variants are distributed in a specific gene, a join can be done between <<AO>> Gene and <<CAO>> Variation, since both share the attributes included in the Chromosome descriptive family.

Following the proposed method, the $BGW_{CM}$ is now identified but not finished. This is seen as a continuous process that refresh the BDW structure as new data requirements or new data sources are available. Based on the method best practices, the data engineer can tune this model based on data access patterns or analytical goals.

Regarding the utility and value of the obtained model, also validating the proposed method, a preliminary analysis about cardiomyopathies was carried out. The data have

been extracted and integrated in the Big Genome Warehouse implemented in Hive, using data from different sources and include information about nine well-known related phenotypes which genotypic characteristics are under study. The aim of this study is to identify patterns and similarities among the phenotypes that could help to understand the mechanisms of disease, a task that constitutes a bottleneck in the genomic analysis process. The results of the preliminary analysis, depicted in **Fig. 7**, show the relationships among the type of variants that occur in the affected chromosomes and the selected phenotypes.

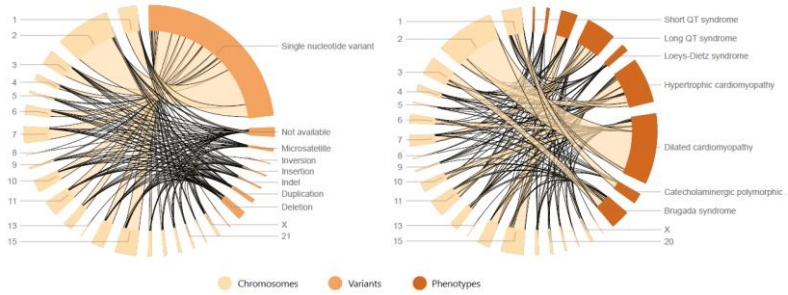


**Fig. 7.** Variants in the affected chromosomes and phenotypes

Based on these results, it can be concluded that the most interesting genetic components are chromosomes 2, 7 and 15 as these present a huge number of polymorphisms, mainly associated with Dilated Cardiomyopathy. Following this premise, an extended analysis can be done, focusing on more specific genes and proteins, and helping the experts to extract meaningful insights about these areas of interest.

## 5 Conclusions

This work extends the knowledge in conceptual modeling applied to Big Data contexts, proposing a conceptual metamodel that formalizes the constructs for designing and implementing BDWs. This paper proposes a method that includes data modelling rules and patterns that guide practitioners from the conceptual metamodel to a specific conceptual model. Due to the challenges of the Genomics domain, the method was demonstrated in this context, being able to provide the Big Genome Warehouse Conceptual Model. As future work, the method must be extended to consider the evolution of the data models when new business processes, data sources or analytical requirements are available. Additionally, best practices must be addressed for the definition of partition or bucketing keys, as these impact the system performance.

**Acknowledgements**. This work has been supported by FCT – *Fundação para a Ciência e Tecnologia* within the Project Scope: UID/CEC/00319/2019, the Doctoral scholarship PD/BDE/135100/2017 and European Structural and Investment Funds in the FEDER component, through the Operational Competitiveness and Internationalization Programme (COMPETE 2020) [Project nº 039479; Funding Reference: POCI-01-0247-FEDER-039479]. We also thank both the Spanish State Research Agency and the

Generalitat Valenciana under the projects DataME TIN2016-80811-P, ACIF/2018/171, and PROMETEO/2018/176. Icons made by Freepik, from www.flaticon.com

## References

1. Krishnan, K.: Data warehousing in the age of big data. Morgan Kaufmann is an imprint of Elsevier, Amsterdam (2013).
2. Santos, M.Y., Costa, C.: Big Data: Concepts, Warehousing and Analytics. River Publishers (2020).
3. Cuzzocrea, A., Moussa, R.: Multidimensional database modeling: Literature survey and research agenda in the big data era. In: 2017 International Symposium on Networks, Computers and Communications (ISNCC). pp. 1–6 (2017).
4. Di Tria, F., Lefons, E., Tangorra, F.: Design process for Big Data Warehouses. In: Data Science and Advanced Analytics (DSAA), 2014 International Conference on. pp. 512–518. IEEE (2014).
5. Dehdouh, K., Bentayeb, F., Boussaid, O., Kabachi, N.: Using the column oriented NoSQL model for implementing big data warehouses. Presented at the In Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA) (2015).
6. Bézivin, J.: On the unification power of models. Softw Syst Model. 4, 171–188 (2005).
7. Reyes Román, J.F., Pastor, Ó., Casamayor, J.C., Valverde, F.: Applying Conceptual Modeling to Better Understand the Human Genome. In Conceptual Modeling. pp. 404–412. Springer International Publishing, Cham (2016).
8. Embley, D.W., Liddle, S.W.: Big data - Conceptual modeling to the rescue. Lecture Notes in Computer Science. 8217 LNCS, 1–8 (2013).
9. Giebler, C., Gröger, C., Hoos, E., Schwarz, H., Mitschang, B.: Modeling Data Lakes with Data Vault: Practical Experiences, Assessment, and Lessons Learned. In Conceptual Modeling. pp. 63–77. Springer International Publishing, Cham (2019)
10. Gil, D., Song, I.-Y.: Modeling and Management of Big Data: Challenges and opportunities. Future Generation Computer Systems. 63, 96–99 (2016).
11. Di Tria, F., Lefons, E., Tangorra, F.: GrHyMM: A graph-oriented hybrid multidimensional model. Lecture Notes in Computer Science. 6999 LNCS, 86–97 (2011).
12. Santos, M.Y., Costa, C.: Data Warehousing in Big Data: From Multidimensional to Tabular Data Models. In: Proceedings of the Ninth International C* Conference on Computer Science & Software Engineering. pp. 51–60. ACM, New York, NY, USA (2016).
13. Kimball, R., Ross, M.: The data warehouse toolkit: The definitive guide to dimensional modeling. John Wiley & Sons (2013).
14. Costa, C., Santos, M.Y.: Evaluating Several Design Patterns and Trends in Big Data Warehousing Systems. In: Lecture Notes in Computer Science. pp. 459–473. Springer (2018).
15. Santos, M.Y., Costa, C., Galvão, J., Andrade, C., Pastor, O., Marcén, A.C.: Enhancing Big Data Warehousing for Efficient, Integrated and Advanced Analytics - Visionary Paper. In Information Systems Engineering in Responsible Information Systems - CAiSE Forum 2019, Proceedings. pp. 215–226. Springer (2019).