DEGREE OF DOCTOR OF PHILOSOPHY IN
COMPUTER ENGINEERING AND SCIENCE

DOCTORATE SCHOOL IN
INFORMATION AND COMMUNICATION TECHNOLOGIES

XXV Cycle

UNIVERSITY OF MODENA AND REGGIO EMILIA

INFORMATION ENGINEERING DEPARTMENT

Ph.D. DISSERTATION

Information Integration for biological data sources

Candidate:
Abdul Rahman DANNAOUI
Advisor:
Prof. Domenico BENEVENTANO
Co-Advisor:
Prof. Nicola PECCHIONI
The Coordinator of the School:
Prof. Maurizio VINCINI
The Director of the School:
Prof.Giorgio Matteo VITETTA

DOTTORATO DI RICERCA IN
COMPUTER ENGINEERING AND SCIENCE

SCUOLA DI DOTTORATO IN
INFORMATION AND COMMUNICATION TECHNOLOGIES

XXV Ciclo

UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

TESI PER IL CONSEGUIMENTO DEL TITOLO DI DOTTORE DI RICERCA

Integrazione dell'informazione per sorgenti dati biologici

Tesi di:
Abdul Rahman DANNAOUI
Relatore:
Prof. Domenico BENEVENTANO
Co-Relatore:
Prof. Nicola PECCHIONI
Coordinatore del dottorato:
Prof. Maurizio VINCINI
Direttore della scuola di dottorato:
Prof. Giorgio Matteo VITETTA

Keywords:

Semantic Data Integration
Data Provenance
Biological Data integration
Breeder-friendly web interface
Cerealab DB

Abstract

Data integration is the process of combining data residing in different sources in order to offer the end user a unified view over the entire available information.

Data Provenance is the process of identifying where data came from, how it was derived, and how it was updated over time.

This PhD research activity on Information Integration for biological data sources was granted by the SITEIA project and was focused on improvements and extensions of the CEREALAB database. The CEREALAB database is a web-based tool realized to help cereal breeders in choosing molecular markers associated to economically important phenotypic traits; it contains phenotypic and genotypic data obtained from the integration of available open source databases with the data obtained by the CEREALAB project. Information integration in the CEREALAB database is obtained by using the MOMIS (Mediator Environment for Multiple Information Sources) system, developed by the DBGroup of the University of Modena and Reggio Emilia.

As a result of the wide use of the CEREALAB database, several extensions and improvements, that can be classified in two categories, were introduced. First, the CEREALAB database content was extended in order to offer to the breeders new significant data. To improve and simplify the access to the database, a new breeder friendly Graphic User Interface (GUI) was developed. To maximize and optimize the accessibility of the available information, new functionalities and additional tools were realized. Finally, a new data entry module was implemented.

Moreover, in order to meet the end-user needs, data provenance was introduced and partially implemented in the context of the CEREALAB database. Data Provenance is an open research problem; it is particularly required in data integration systems, where information coming from different sources, potentially uncertain or even inconsistent with each other, is integrated. In this context, having the possibility to trace the lineage of specific data can help identify possible unexpected or questionable results.

Sommario

Data integration è il processo di combinare i dati che risiedono in fonti diverse al fine di offrire all'utente finale una visione unificata dell'intera informazione disponibile.

Data Provenance è il processo dell'identificazione dell'origine del dato, come è stato derivato e come è stato modificato nel tempo.

Questa attività di ricerca sull'integrazione dell'informazione per fonti di dati biologici è stata finanziata dal progetto SITEIA e si è concentrata sul miglioramento e le estensioni del database CEREALAB. Il database CEREALAB è un strumento *web-based* realizzato per aiutare i *breeders* di cereali nella scelta di marcatori molecolari associati a caratteri fenotipici economicamente importanti, esso contiene i dati fenotipici e genotipici ottenuti dall'integrazione delle banche dati open source disponibili con i dati ottenuti dal progetto CEREALAB. L'integrazione dell'informazione nel database CEREALAB è stata ottenuta utilizzando il sistema MOMIS *(Mediator Environment for Multiple Information Sources)*, sviluppato dal DBGroup dell'Università degli Studi di Modena e Reggio Emilia.

Come risultato dell'estensivo uso del database CEREALAB, diverse estensioni e miglioramenti, che possono essere classificati in due categorie, sono state introdotte. In primo luogo, il contenuto del database CEREALAB è stato esteso in modo da offrire ai *breeders* nuovi dati significativi. Per migliorare e semplificare l'accesso al database, una nuova interfaccia grafica *Breeder-Friendly* è stata sviluppata. Per massimizzare e ottimizzare l'accessibilità delle informazioni disponibili, nuove funzionalità e strumenti aggiuntivi sono stati realizzati. Infine, un nuovo modulo di inserimento dati è stato implementato.

Inoltre, al fine di soddisfare le esigenze degli utenti finali, la *data provenance* è stata introdotta e parzialmente implementata nel contesto del database CEREALAB. La *Data Provenance* è un problema di ricerca aperto ed è particolarmente richiesto nei sistemi di integrazione dati, dove informazioni provenienti da fonti diverse, potenzialmente incerti o anche in contrasto tra di loro, sono integrati. In questo contesto, avendo la possibilità di risalire all'origine del dato può aiutare a identificare possibili risultati inattesi o discutibili.

Contents

List of Figures

Chapter 1

Data Integration and Data Provenance

1.1 Data Integration

Data integration is the problem of combining data residing at different sources, and providing the user with a unified view of these data [Lenzerini, 2002]. Data integration is important in different real world applications and different domains, for example in the case of large companies that have different data sources or in the case of two similar companies that need to merge their data. It is also important in scientific projects when different datasets are produced by different researchers who want to integrate these results together [Halevy et al., 2006b]. Finally, the data integration has always been important in the life sciences where there are amounts of data that are produced at increasing rates and where progress depends on the ability of researchers to analyse data from different sources. [Halevy et al., 2006b].

With the increase of the data produced and shared in collaborative environments, the problem of designing and implementing information integration techniques has become even more important. One of the main reasons of data integration is that often the data reside in different data sources, heterogeneous and distributed. The field of data integration has been heavily investigated over the past twenty years, especially by the databases research community, for his significant impact and the possibility to use it in various fields, and then various techniques have been developed for the integration of heterogeneous data sources. Data integration systems provide seamless access to a collection of data stored in various heterogeneous data sources. These data sources may vary from database systems to forms on the web and flat files.

In the last thirty years different approaches have been suggested and implemented:

- A collection of database systems autonomous and possibly heterogeneous integrated with different degree compound the *federated database system (FDBS)* [Heimbigner and McLeod, 1985, Sheth and Larson, 1990]. A global schema must be defined to integrate the data from the participating databases. The design of the integrated schema has to do with the problems of data integration to understand how to combine the same data or similar data represented differently in different participating databases. The schemata of the participating database refer to legacy systems, i.e. existed before the integrated schema has been designed. For this reason the design process becomes bottom-up, while the homogeneous databases are usually designed top-down [DBL, 2009].

- The *data warehouse* approach [Inmon, 2002]consists in storing copies of data residing in different databases and data sources in a single database with a star schema called warehouse. Before storing it in the warehouse,

the data is extracted, cleansed, transformed and then loaded in the warehouse, this process is called ETL (extract, transform and load). Since the warehouse is a physical structure, it needs to be updated periodically and because of the ETL process there is an element of latency in the warehouse data. This approach is mainly used to facilitate reporting and data analysis.

- Another approach is the *read-only data integration systems* which based on the mediator system. A mediator [Wiederhold, 1992] is a software component that supports a virtual database, which the user may query as if it were materialized. The mediator system schema is a logical schema, with no data stored, used as a middle layer to separate the application from the data sources. This schema forms a unified global schema of the source schemata to be integrated. User can interrogate the mediated schema in transparent way. The query is transformed into a set of sub-queries for the sources of interest by means of automatic rewriting operations. The different result for the different sub-queries are then unified using data reconciliation techniques. The global schema represents a conceptualization of the sources involved, i.e. a domain ontology

- *Data exchange* It is the problem of taking structured data from a source schema and creating an instance of destination schema that reflects the source as accurately as possible [Fagin et al., 2005, Kolaitis, 2005]. The architecture consists of a source schema, a destination schema and semantic mappings between the source and the destination schema to explain how to populate the target schema from the source. This approach is used in the data transferring between applications where the target schema is often independently created.

The rapid development of new applications with data from different sources is another data integration goal which hides many problems like the identifying of the best data source to use and the creation of an appropriate schema for the integrated data [Haas, 2007].

The definition of the global schema and the mappings between the data schemas of the different sources is often a manual process and this represents the main limitations of traditional data integration systems. To overcome this problem, semantic-integration researches have been proposed, with the following three main dimensions [Noy, 2004]:

- Mapping discovery: how similarities between two schemata are discovered.

- Declarative formal representations of mappings: to represent mappings between two schemata.

- Reasoning with mappings: what to do with the mappings, what type of reasoning they can be used for.

1.2 Data Provenance

Most of the data stored and generated by many application domains like data-warehousing, data integration, curated databases, scientific computing and workflow management is derived from existing data by using complex transformations. Having knowledge about how data is transformed and its origin is needful to understand its semantics and to estimate its quality. This type of knowledge is called data provenance or data lineage which describes how a data item (piece of data) is produced. In relational databases the data provenance information includes source and intermediate data which is represented by relations, tuples and attribute and transformations which is represented by SQL queries and/or functions.

In the data integration systems, especially in those based on mediator system, data provenance can be used to fix errors in the data sources by tracing back its origin, or to estimate its quality and its trustworthiness. An interesting challenge presented in [Moreau, 2009, Jagadish et al., 2007] regards human-computer interactions. Many operations performed by Integration Systems, ranging from schema-mapping design to querying, involve humans in the loop, who impact on decisions and processes. This scenario clearly delineates the necessity to include data lineage in human-computer interactions. By explaining how data are integrated, lineage plays an important role in this process and improves the quality of the human-computer interaction itself. Moreover, lineage has been used to debug schema mappings that may be imprecise or incorrect. The debugging tool developed in [Chiticariu and Tan, 2006] describes the relationship between source and target data with the schema mapping, allowing designers to see how ‘bad‘ data has been produced.

A fundamental task in data integration is *data fusion*, the process of fusing multiple records representing the same real-world object into a single, consistent, and clean representation. Data fusion involves the resolution of possible conflicts between data coming from different data sources [Bleiholder and Naumann, 2008]. A recent tutorial [Dong and Naumann, 2009] listed data lineage as one of the open problems and desiderata for data fusion systems. Merging data implies a partial loss of the original values of the local sources. For this reason database administrators and data owners are notoriously hesitant to merge data. Data lineage can help explaining merging decisions by tracking which original values were involved and how they have been fused together.

1.3 Outline of the Thesis

The rest of this thesis is organized as follows:

- *Chapter 2* is dedicated to the MOMIS System. In particular we present the definition of the MOMIS data integration system; the ODL_{I^3} language used by MOMIS and we define the process of the global schema generation followed by the MOMIS system. we also describe the MOMIS as a data fusion system; we present the MOMIS architecture and we briefly describe the query execution process.

- In *Chapter 3*, a description of the extension made for the CEREALAB database is introduced and the result is presented. In particular we present the problem definition and motivates the development of the second version of the CEREALAB database; define the problem domain to facilitate the comprehension of the terms used in the rest of the chapter; describe the data sources integrated to create the CEREALAB database; present an example of the creation of the global class Germplasm; describe the new ontology of the CEREALAB database; present the new CEREALAB web interface with its functionalities, report and data entry module and finally discuss the improvements made with respect to the first version.

- *Chapter 4* focuses on the study of the best data provenance model for the MOMIS data integration system. In particular we will present the problem of data provenance in the data integration systems; we will informally discuss the use of *lineage* and *why*-provenance for data fusion queries then we will formally define these concepts and finally we sketch the conclusions and future works.

- In *Chapter 5*, we present the implementation of a data provenance support PM_{MOMIS} and its application in the context of the CEREALAB database. We also present a problem encountered during the application of provenance in the context of biological data derived from the type of data and the specificity of the biological domain and finally we present a possible solution for it by using the *Trio* system.

- Finally, in *Chapter 6*, I make some concluding remarks and discuss the future work.

Chapter 2

MOMIS (Mediator envirOnment for Multiple Information Sources)

This chapter focuses on the MOMIS[1] system (Mediator EnvirOment for Multiple Information Sources), developed by the DBGroup of the University of Modena and Reggio Emilia, which represents the application context of the methods proposed in this thesis. MOMIS is an Intelligent Data Integration system that adopts a GAV approach (global-as-view approach describes the mediated schema as a set of view definitions over the source relations. the content of these views is expressed in terms of queries over sources) to integrates heterogeneous data sources. This chapter provides a general description of MOMIS. The DataRiver Spin-Off of the University of Modena and Reggio Emilia[2] has released the first open-source version of MOMIS on May 2010.

The rest of the chapter is organized as follows: section 2.1 presents the definition of the MOMIS data integration system; in particular, subsection 2.1.1 presents the ODL_{I^3} language used by MOMIS and subsection 2.1.2 defines the process of the global schema generation followed by the MOMIS system. section 2.2 describes the MOMIS as a data fusion system; finally in section 2.3 we present the MOMIS architecture and we briefly describe the query execution process.
Most of the content of section 2.1 and section 2.3 is a synthesis of the published material available for the MOMIS system (in particular from [Beneventano et al., 2003a, Bergamaschi et al., 1999]). In section 2.2 we focus on MOMIS as a data fusion and then we introduce the framework that will be used in chapter 4.2.1 to discuss the provenance of Data Fusion Queries [Beneventano et al., 2011b, Beneventano et al., 2011a, Beneventano et al., 2012b].

2.1 The MOMIS Data Integration System

Definition of the MOMIS Integration System

Definition 1. *Given a set N of data sources to be integrated, we can define a MOMIS Integration System IS = (GS, N, M) as constituted by:*

- *A Global Schema (GS), which is a schema expressed in the ODL_{I^3} language*
- *A set N of data sources; each source has a schema also expressed in ODL_{I^3}*
- *A set M of GAV mapping assertions between the GS and N, where each assertion associates to an element G in GS a query q_N over the schemas of the set N of local sources. More precisely, for each global class $G \in GS$ we define:*

[1] See `http://www.dbgroup.unimore.it` for references about the MOMIS project.
[2] `http://www.datariver.it`

1. *a (possibly empty) set of classes, denoted by L(G), belonging to the local sources in N*
2. *a conjunctive query q_G over the L(G) classes*

Intuitively, the GS is the intentional representation of the information provided by the Integration System, whereas the mapping assertions specify how such an intentional representation relates to the local sources managed by the Integration System. The semantics of an Integration System is defined in [Cali et al., 2002, Beneventano and Lenzerini, 2005].

MOMIS performs information extraction and integration from both structured and semi-structured data sources. An object-oriented language, with an underlying Description Logic, called ODL$_{I^3}$, described in Section 2.1.1 (see Appendix A for its syntax), is introduced for information extraction. Information integration is then performed in a semi-automatic way, by exploiting the knowledge in a Common Thesaurus (defined by the framework) and ODL$_{I^3}$ descriptions of source schemas with a combination of clustering techniques and Description Logics. This integration process gives rise to a virtual integrated view (the Global virtual Schema GS) of the underlying sources for which mapping rules and integrity constraints are specified to handle heterogeneity. Given a set of data sources related to a domain, it is possible to semi-automatically synthesize a GS that conceptualizes a domain and thus might be thought as a basic domain ontology for the integrated sources.

2.1.1 The ODL$_{I^3}$Language

The ODL$_{I^3}$ language used in the MOMIS system to represent data is an extension of the *Object Definition Language* (ODL), an object-oriented language developed by ODMG[34]. ODL$_{I^3}$ is transparently translated into a Description Logic [Beneventano et al., 2003b, Bergamaschi et al., 2001]. ODL$_{I^3}$ allows different kinds of data sources and the view resulting from the integration process to be represented in a common data model. In ODL$_{I^3}$ all the data sources are represented as a set of classes and attributes. Moreover, some constructors and rules are present in the language to handle the heterogeneity, in particular:

Intentional relationships. They are *terminological relationships* expressing intra- and inter-schema knowledge for the source schemas. Intentional relationships are defined between classes and attributes, and are specified by

[3] http://www.odmg.org/

[4] http://www.service-architecture.com/database/articles/odmg_3_0.html

considering class/attribute names, called terms. The following relationships can be specified in ODL$_{I^3}$:

- SYN (Synonym-of), defined between two terms t_i and t_j, with $t_i \neq t_j$, that are considered synonyms in every considered source (i.e., t_i and t_j can be indifferently used in every source to denote a certain concept).
- BT (Broader Terms), or hypernym, defined between two terms t_i and t_j such as t_i has a broader, more general meaning than t_j. BT relationship is not symmetric. The opposite of BT is NT (Narrower Terms), or hyponymy.
- RT (Related Terms), or positive association, defined between two terms t_i and t_j that are generally used together in the same context in the considered sources.

An intentional relationship is only a terminological relationship, with no implications on the extension or compatibility of the structure (domain) of the two involved classes (attributes).

Using ODL$_{I^3}$ for representing sources and ontologies is not a limitation: the interoperability of the ODL$_{I^3}$ descriptions is guaranteed by a software module able to translate the descriptions into the Web Ontology Language OWL [Orsini, 2004].

2.1.2 Global Schema Generation with MOMIS

The integrated global schema generated by the MOMIS system is composed of a set of global classes that represent the information contained in the underlying sources and the mappings that establish the connections among global class attributes and the source schemata.

The process of creating the GS and defining the mappings, shown in figure 2.1, can be summarized in the following steps:

Local source schemas extraction The first phase of the integration process is the choice of the data sources and their translation into the ODL$_{I^3}$ format. The translation process is performed by the MOMIS wrappers, which logically converts the source data structure into the ODL$_{I^3}$ model. The wrapper architecture and interfaces are crucial, because wrappers are the focal point for managing the diversity of data sources.

Lexical annotation of local sources The goal of the annotation phase is to semantically annotate terms denoting schema elements in data sources according to a common lexical reference, which means to assign a shared meaning

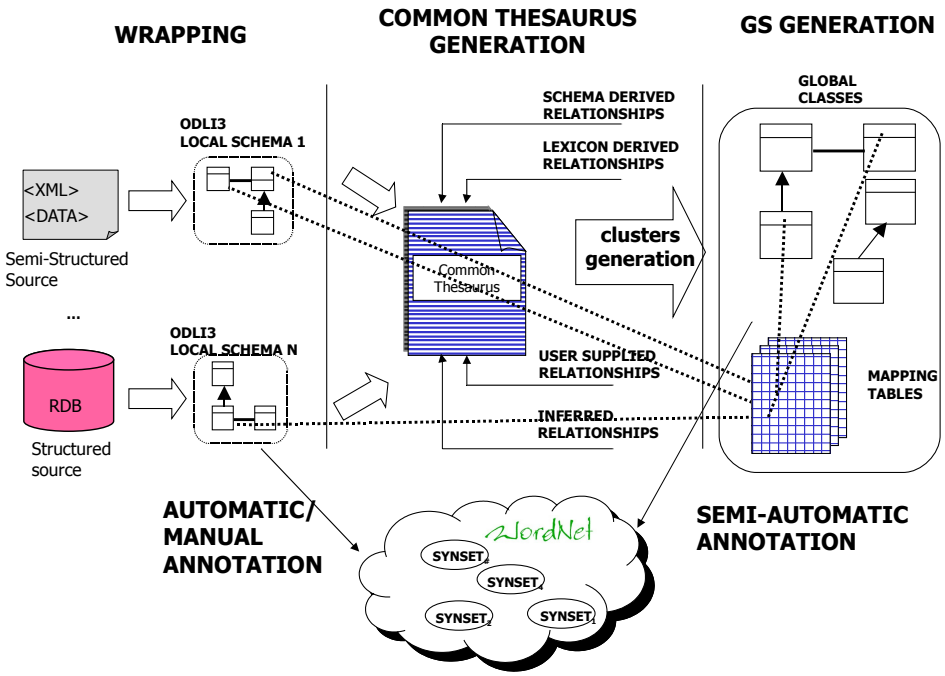


Figure 2.1: Global Schema generation process with the MOMIS system

to each of them. The WordNet lexical database is usually referred to as lexical reference [Fellbaum, 1998], but other lexical references might be used to cope with specific domains. The annotation step of the schema elements, preprocessed by the system applying stop-words and stemming techniques, can be performed manually by the integration designer, or automatically by the MOMIS system. The manual annotation is composed of two different steps: in the Word Form choice step, the WordNet morphological processor suggests a word form corresponding to the given term; in the Meaning choice step, the designer can choose to map an element to zero, one or more senses. In MOMIS the annotation step can also be performed automatically combining a set of Word Sense Disambiguation algorithms [Bergamaschi et al., 2007]: SD (Structural Disambiguation) [Bergamaschi et al., 2008], WND (WordNet Domains Disambiguation) [Bergamaschi et al., 2008], WordNet first sense heuristic, Gloss Similarity [Beneventano et al., 2008] and Iterative Gloss Similarity [Beneventano et al., 2008]. For further details about the annotation techniques in MOMIS see [Po, 2009].

Common thesaurus generation Starting from the annotated local schemata, MOMIS constructs a Common Thesaurus describing intra and inter-schema knowledge in the form of SYN(synonyms), BT/NT(broader terms/narrower terms), and RT(meronymy/holonymy) relationships. The Common Thesaurus is constructed through an incremental process in which the following relationships are added:

- schema-derived relationships: relationships holding at intra-schema level are automatically extracted by analysing each schema separately. Some heuristic can be defined for specific kind of sources. For example, MOMIS extracts intra-schema RT relationships from foreign keys in relational source schemas. When a foreign key is also a primary key, in both the original and referenced relation, MOMIS extracts BT and NT relationships, which are derived from inheritance relationships in object-oriented schemas.
- lexicon-derived relationships: the annotation phase is exploited to translate relationships holding at the lexical level into relationships to be added to the Common Thesaurus. These relationships may be inferred from lexical knowledge (e.g. by querying WordNet for relationships between senses).
- designer-supplied relationships: new relationships can be supplied directly by the designer, to capture specific domain knowledge.
- inferred relationships: Description Logics (DL) techniques of ODB-Tools [Bergamaschi et al., 1997], are exploited to infer new relationships.

Global Schema Generation The Global virtual Schema (GS) consists of a set of classes (called Global Classes), plus mappings to connect the global attributes of each Global Class and the local source attributes. The MOMIS methodology allows identifying similar ODL$_{I^3}$ classes (i.e. classes that describe the same or semantically related concept in different sources) and mappings to connect the global attributes of each global class to the local source attributes. To this end, affinity coefficients are evaluated for all possible pairs of ODL$_{I^3}$ classes, based on the relationships in the Common Thesaurus properly strengthened. Affinity coefficients determine the degree of matching of two classes based on their names (*Name Affinity coefficient*) and their attributes (*Structural Affinity coefficient*) and are fused into the *Global Affinity coefficient*, calculated by means of the linear combination of the two coefficients [Castano et al., 2001]. Global affinity coefficients are then used by a hierarchical clustering algorithm, to cluster ODL$_{I^3}$ classes according to their degree of affinity. For each cluster C, a Global Class G, with a set of Global Attributes $GA_1, \ldots GA_N$, and a Mapping Table MT,

expressing mappings between local and global attributes, are defined. The Mapping Table is a table whose columns represent the local classes which belong to the Global Class and whose rows represent the global attributes. An element $MT[GA][LC]$ is a function which represents how local attributes of the Local Class LC are mapped into the global attribute GA:

$$MT[GA][LC] = f(LAS)$$

where LAS is a subset of the local attributes of LC. More precisely, the integration designer can define, for each not null element $MT[GA][LC]$, a Data Conversion Function, denoted by $MTF[GA][LC]$, which represents the mapping of local attributes of LC into the global attribute GA. $MTF[GA][LC]$ is a function that must be executable/supported by the class LC local source. For example, for relational sources, $MTF[GA][LC]$ is an SQL value expression. $T(LC)$ denotes L transformed by the Data Conversion Functions. Further details about MOMIS data conversion functions can be found in [Beneventano et al., 2007].

GS lexical annotation To annotate a GS means to assign a name and a set (eventually empty) of meanings to each global element (class or attribute). MOMIS automatically annotates each global element proposing the broadest meaning extracted from the annotations of the local sources, based on the relationships included in the Common Thesaurus. Names and meanings have then to be confirmed by the ontology designer. This annotation step is a significant result, since these metadata may be exploited by external users and applications.

2.2 MOMIS as a Data Fusion System

After the Global Schema generation process, each global class G is associated to a Mapping Table. Starting from the Mapping Table of G, the Integration Designer defines its mapping query $\mathcal{MQ}^G$; as said before, $\mathcal{MQ}^G$ is a query over $L(G)$ which defines the instance of G starting from the instances of its local classes.

In this thesis we are interested at the case where the mapping query $\mathcal{MQ}^G$ is defined to make a global class perform *Data Fusion* among its local class instances [Bleiholder and Naumann, 2008]: multiple *local tuples* coming from local classes and representing the same real-world object are fused into a single and consistent *global tuple* of the global class. To identify multiple local tuples coming from local classes and representing the same real-world object, we assume that error-free and shared object identifiers exist among different sources (this is the situation where the step of Object Identification has been already performed and its result is the assignment of an object identifier to each record): two local tuples with the same object identifier indicate the same object in different sources.

Mapping Table of G_1

G_1	L_1	L_2
ID	ID	ID
A	A	
B		B
C	C	C

Mapping Table of G_2

G_2	L_1	L_2	L_3
ID	ID	ID	ID
A	A		
B		B	
C	C	C	C

Figure 2.2: Examples of Mapping Tables

As said before, both the global and the local schemas are expressed in the ODL_{I^3} language [Bergamaschi et al., 2001]. However, for the scope of this thesis, we consider both the GS and the LS_i as relational schemas, but we will refer to their elements respectively as global and local classes to comply with the MOMIS terminology. As an example, in Figure 2.2, we consider three local classes with schema $L_1(ID, A, C)$, $L_2(ID, B, C)$ and $L_3(ID, C)$, respectively, and we show the Mapping Table of the global classes G_1, with schema $G_1(ID, A, B, C)$ and with local classes $L(G_1) = \{L_1, L_2\}$, and G_2 with schema $G_2(ID, A, B, C)$ and with local classes $L(G_2) = \{L_1, L_2, L_3\}$. A global attribute GA such as there is only a not null element $MT[GA][L]$ is called *one-to-one*, otherwise is called *one-to-many*.

As said before, *GAV* mapping assertions are expressed by specifying for each global class G a query over $L(G)$, called *mapping query* and denoted by $\mathcal{MQ}^G$, which defines the instance of G starting from the instances of its local classes. The mapping query $\mathcal{MQ}^G$ can be defined to make a global class perform *Data Fusion* among its local class instances [Bleiholder and Naumann, 2008]: multiple *local tuples* coming from local classes and representing the same real-world object are fused into a single and consistent *global tuple* of the global class. Merging data from different sources requires different instantiations of the same real world object to be identified; this process is called object identification [Bleiholder and Naumann, 2008]. In the MOMIS system, to identify instances of the same object and fuse them we introduce Join Conditions among pairs of local classes belonging to the same global class. Given two local classes L_1 and L_2 belonging to G, a Join Condition between L_1 and L_2, denoted with $JC(L_1, L_2)$, is an expression over $L_1.A_i$ and $L_2.A_j$ where A_i (A_j) are local attributes with a not null mapping in L_1 (L_2). In this thesis, to identify multiple local tuples coming from local classes and representing the same real-world object, we assume that error-free and shared object identifiers exist among different sources (this is the situation where the step of Object Identification has been already performed and its result is the assignment of an object identifier to each record): two local tuples with the same object identifier indicate the same object in different sources. In

our example, we assume ID as an object identifier.

Data fusion may involve *Data Reconciliation*, i.e. the resolution of possible conflicts between data coming from different sources; several high level strategies to handle inconsistent data have been described and classified in [Bleiholder and Naumann, 2008]. As we highlighted in [Beneventano et al., 2012b], the MOMIS system supports: *conflict avoiding* strategies (such as the *trust your friends* strategy which takes the value of a preferred source), *conflict ignoring* strategies (such as the *pass it on* strategy, which presents all values deferring conflict resolution to the user) and *resolution* strategies (such as the *meet in the middle* strategy which takes an average value). These strategies are implemented by means of *Resolution functions* [Naumann and Bleiholder, 2006]: for each GA such that there are more than one non empty element $MT[GA][L]$, a *Resolution Function* (*RF*) is defined to obtain, starting from the transformed local classes, a *unique value* for GA. Examples of *RFs* are *aggregation functions* (for numerical attributes: SUM, AVG, MIN, MAX, etc;), *Precedence function* (the highest informational quality value on the basis of an information quality model) *Coalesce function* (the first not null value among the local attributes values) and *Random function* (this function results in having the value of one of the local attribute randomly chosen).In our example an `AVG` resolution function is defined on `C`.

In order to carry on the *Data Fusion* operations described so far, the mapping query $\mathcal{MQ}^G$ is defined by means of the *full outerjoin-merge operator* proposed in [Naumann et al., 2004] and adapted to the MOMIS framework in [Beneventano et al., 2010].Here, we consider an informal formulation of $\mathcal{MQ}^G$ in SQL. Intuitively, defining $\mathcal{MQ}^G$ by means of a full outerjoin-merge corresponds to the following two operations: (1) Computation of the *Full Outer Join*, on the basis of the shared object identifiers, of the *local classes* of G; (2) Application of the resolution functions. Thus $\mathcal{MQ}^G$ can be formulated by standard SQL, with the exception of (some) resolution functions. As an example, for the global class G_1 of Figure 2.2, we have the following $\mathcal{MQ}^{G_1}$:

```
MQ^G_1 : SELECT ID,
                L1.A AS A,
                L2.B AS B,
                AVG(L1.C,L2.C) AS C
           FROM L1 FOJ L2 USING (ID)
```

where `FOJ` is an abbreviation for the SQL full outer join operator[5] , `COALESCE` is the standard SQL function which returns its first non-null parameter value and `AVG` is a (non-standard SQL) function to compute the average value.

[5]As shown in [Beneventano et al., 2010] if we use a join condition based on the shared object identifier ID, the order of local classes in the full outer join evaluation is not relevant.

L_1

ID	A	C
1	3	24
2	NULL	20
3	9	NULL
4	8	25
5	NULL	20

L_2

ID	B	C
1	3	24
2	NULL	30
3	NULL	20
5	NULL	30

L_3

ID	C
5	25

$G_1 = G_2$

ID	A	B	C
1	3	3	24
2	NULL	NULL	25
3	9	NULL	20
4	8	NULL	25
5	NULL	NULL	25

Figure 2.3: Instances of the local classes and of the global classes computed with the full outer join merge operator

For the global class G_2 (with more than two local classes) of Figure 2.2, we have the following $\mathcal{MQ}^{G_2}$:

```
MQˆG_2: SELECT ID,
              L1.A AS A, L2.B AS B,
              AVG(L1.C,L2.C,L3.C) AS C
        FROM L1 FOJ L2 USING (ID)
                FOJ L3 USING (ID)
```

As an example, in Figure 2.3, we show the instances of local classes [6] and the corresponding instance of the global class computed by means of $\mathcal{MQ}^G$. As it can be easily verified, for this example, the instances of G_1 coincide with the instances of G_2.

2.3 MOMIS Architecture

In this section we briefly describe the architecture of the MOMIS system, shown in Figure 2.4. Further details can be found in [Orsini, 2009].

The main components of MOMIS are:

- **Wrappers:** Wrappers are software modules with the role of managing the interactions with the data sources. The MOMIS wrappers translate the source data structures into ODL_{I^3}. Their role is to deal with the diversity of the data sources thus allowing MOMIS to pay no attention to the language details of the different data sources. Wrappers are available for different kind of data sources, ranging from different database management systems

[6] Let C denote either a local class L or a global class G. An instance of C is a set of tuples conforming with the schema of C; we conflate the notation and use the same symbol C for both the class C and an instance of C.

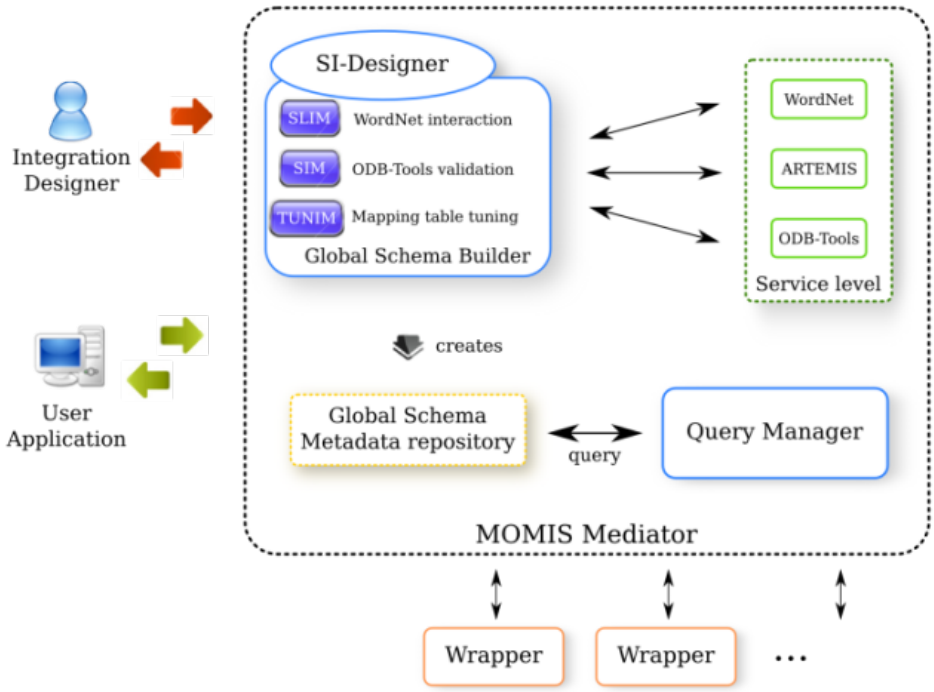


Figure 2.4: MOMIS Architecture

to semi-structured data like XML, RDF, OWL formats. Wrappers logically guarantee two main operations:

- *getschema()* translates the schema from the original format into ODL_{I^3}, dealing with the necessary data type conversions
- *runquery()* executes a query on the local source. The MOMIS Query Manager translates a query on the GS (a global query) into a set of local queries to be locally executed by means of wrappers.

- **Global Schema Builder:** The Global Schema Builder is the main module that interacts with the wrappers and the Service tools to generate the Global Schema. It is composed of different components that implements the different steps of the integration process described in Section 2.1.2.

- **Service tools:** the Service tools are important modules exploited by the Global Schema Builder and the Query Manager. These include the software module in charge of managing the WordNet lexical database used for the annotation phase, the Artemis tool implementing the clustering algorithm, and the ODB-Tools reasoner used to calculated inferred relationships.

- **Query manager:** the Query Manager is the component in charge of solving a user query: it generates the local queries queries for wrappers, starting from a global query formulated by the user on the global schema. In more details, the MOMIS Query Manager allows the user to pose a query expressed in OQL [Beneventano et al., 2003b, Orsini, 2009] over the ontology and to obtain a unified answer from all the data sources integrated in the GS. When the MOMIS Query Manager receives a query, it rewrites the global query as an equivalent set of queries expressed on the local schemas (local queries); this query translation is carried out by considering the mapping between the GS and the local schemata. The query translation is thus performed by means of query unfolding, i.e. by expanding a global query on a global class G of the GS according to the definition of the mapping query q_G. The local queries are then executed on the sources, and MOMIS performs the fusion of the local answers into a consistent and concise unified answer, and present the global answer to the user.

Chapter 3

An integrated Ontology for genotypic and phenotypic data: the CEREALAB database V2.0

This chapter presents the release of the CEREALAB database V2.0: A database that contains phenotypic and genotypic data for three species of cereals: wheat, barley and rice. The realization of this project was done in collaboration between the CropSci group of the Department of Agricultural and Food Sciences and the DBGroup of the Department of Information Engineering of the University of Modena and Reggio Emilia in the context of the SITEIA [1] project funded by the Regional Government of Emilia Romagna in Italy.
The objective of this work is the improvement of the first release of the CEREALAB database by making several extensions derived from the real needs of the end-user. First, to offer to the breeders new significant data, the CEREALAB database content was extended. Second, to improve and simplify the access to the database, a new Breeder-friendly Graphic User Interface (GUI) was developed. Third, to maximize and optimize the accessibility of the available information, new functionalities and additional tools were realized; in particular, to offer the breeders an effective tool for the analysis of the data, the possibility to obtain structured reports was introduced. Finally, to insert new data into the database, a new data entry module was implemented. The CEREALAB database V2.0 can be accessed at this link: http://www.cerealab.org.
The chapter is organized as follows: section 3.1 presents the problem definition and motivates the development of the second version of the CEREALAB database; section 3.2 defines the problem domain to facilitate the comprehension of the terms used in the rest of the chapter; section 3.3 describes the data sources integrated to create the CEREALAB database; section 3.4 presents an example of the creation of the global class Germplasm and describes the new ontology of the CEREALAB database ; section 3.5 presents the new CEREALAB web interface with its functionalities, report and data entry module. Finally in section 3.6 we discuss the improvements made with respect to the first version.
The work presented in this chapter was published in [Dannaoui et al., 2012c], [Dannaoui et al., 2012b], [Dannaoui et al., 2012a] [2] and [Beneventano et al., 2012a].

3.1 Problem Definition

Biotechnology has opened an area of great interest and great potential through the analysis of the genome based on the detection of molecular markers. The principle on which is based the application of molecular markers derives from the fact that its disclosure is not directly attributable to the activity of specific genes but is based directly on the identification of differences in the nucleotide sequence of

[1] See `http://www.siteia.it`
[2] Submitted to Database /`http://database.oxfordjournals.org`

DNA (insertions, deletions, inversions, etc.) which constitutes the genome of each individual. These markers have many advantages with respect to morphological markers. They, being the expression of the genotype, do not suffer interferences from the environment, can be detected in the early stages of development of the plant and are recordable with only small amounts of biological samples.
Various types of markers are now used in the plant breeding, each with characteristics more or less suited to different purposes which aims molecular analysis at the DNA level. Among the molecular markers used today in the genetic improvement are the RAPD (Rapid Amplified Length Polymorphism), the AFLP (Amplified Fragment Length Polymorphism), the CAPS (Cleaved Amplified Polymorphic Sequences), the SCAR (Sequence Characterized Amplified Regions), the STS (Sequence Tagged Site), the SSR (Simple Sequence Repeat) and the SNP (Single Nucleotide polymorphism).
The use of molecular markers for assisted selection (i.e. in the early selection for the character of interest, no longer at the level of phenotype using morphological markers, but at the level of the genotype through the analysis of molecular markers) has attracted considerable interest. This possibility, which immediately seemed very attractive to those working in the field of genetic improvement, has led to develop the so called molecular breeding or molecular marker assisted selection (MAS, Marker Assisted Selection). The selection for genetic markers closely associated with the gene to be selected, relatively easier to detect, can facilitate the selection of traits of interest, more difficult to determine. In practice, the presence of a gene of interest is detected (resistance, morphology of the plant, flower, fruit, etc.) through the presence of the marker closely linked to the gene, as associated genes are transmitted together. The selection for markers associated may thus be more effective than the direct determination of the trait. The selection assisted by molecular markers can be screened directly on the DNA of plants at an early stage of development, without waiting for specific phenological stage in which this character is expressed thus speeding up time and reducing the space required for the work of selection, which often also requires the analysis of thousands of different genotypes. In fact, after having identified one or more markers associated with the character to be selected, the screening is performed on small quantities of DNA extracted from various tissues of the plant, without determining either the plant destruction or the use of large quantity of seeds and this allows the simultaneous analysis of multiple characters and perform multiple cycles of selection in the same year.
Currently assisted selection is used for obtaining superior genotypes both for resistance to biotic stress, and for yield and quality traits. In the first case, MAS clearly shows its advantages when the character is expressed in the final stages of development of the plant, when the expression of the character is recessive and when is requested complex work before the gene is discoverable.

With the rapid development of sequencing techniques and its prices decreasing, research laboratories and seed companies have begun to produce a large amount of data regarding molecular markers, genes and alleles of resistance in various species such as maize, wheat, barley, rice, tomato ...etc, and the associations between them. This amount of data required its storage in a data store to make it available at any time to those who need it and this led to the birth of different biological databases public and private. Some of these databases are relational databases while others are simple Excel file. In addition, some databases contain data regarding associations between molecular markers, genes and alleles in the same germplasm others contain different trait studies for different varieties in different atmospheric conditions such as resistance to cold, the morphology of the plant and more.

For the plant breeding process, breeders need different type of information that can be found in various data sources so they have to access different databases to retrieve the information they need and to do a cross search to get the final result. This process is very complicated for users who do not have enough biological preparation or informatics skills and when the data are repeated in different data sources, so it may have the opposite effect making them spending too much time to retrieve the data of interest rather than exploit the existing information in order to accelerate the results achievement. To overcome this problem, it is necessary to create specific tools for breeders to give them the possibility to access a single data source that contains the information they need and to implement user friendly interfaces to simplify the search for users with low IT and biology expertise.

The CEREALAB and SITEA projects were funded by the Emilia Romagna regional government to increase the competitiveness of regional seed companies through the use of the modern MAS (Marker-Assisted Selection) technologies. One of the objectives of these two projects was the creation of the CEREALAB database. The CEREALAB database was designed to store genotypic and phenotypic data obtained by the project, and to integrate them with already existing data sources, in order to create a tool for plant breeders and geneticists. Plant breeders obtain new types of cultivated plants named varieties, i.e. populations of genotypes of a given species, uniform and distinct from other varieties. The procedure of plant breeding necessarily passes through creation of new genetic variation, generally by crossing parental genotypes, and selection of desired genotypes. In a modern view, operators need to know the genetics of economically important phenotypic traits, to identify and tag molecular markers associated to such key traits, to choose genetically consciously the desired parentals for specific breeding programs, as well as to select genotypically the plants derived from initial crosses. In this view the database was created, to provide the breeders with a comprehensive tool to help them to use in the breeding process molecular markers associated to the agronomically most important traits. The

database has been divided into three sub-schemas corresponding to major cereal species of interest: wheat, barley and rice; each sub-schema has been then divided into two sub-ontologies, regarding genotypic and phenotypic data, respectively [Milc et al., 2011]. The data contained in the CEREALAB database are the result of the integration of different open source databases that contain either phenotypic or genotypic data, with the data obtained by the genotyping activity of the CEREALAB project. A mediator system called MOMIS (Mediator envirOnment for Multiple Information Sources) was used to integrate these data. MOMIS is a framework [Bergamaschi et al., 2001] that performs the integration in a semi-automatic way and gives the users the possibility to interrogate the sources regardless of the specific language or the structure of each.
Integration of the two different data types, phenotypic (records of several agronomic traits for a given genotype) and genotypic (mainly molecular markers, genes, and other DNA-based information for a given genotype), together with the possibility to carry out combined queries for the two sub-ontologies is the advantage of CEREALAB for its use as a tool for modern plant breeding. In fact, even if many public databases exist for the plant science community, in most cases they are not designed for marker-assisted breeding. Among the many open source databases for plant biological data, Gramene [Liang et al., 2008] is a comparative genome mapping database for grasses, Graingenes [Carollo et al., 2005] is a database for Triticeae (wheat, barley, rye, triticale) and oats that contains genotypic and phenotypic data, MetaCrop [Schreiber et al., 2012] is a database for Biochemical pathways and enzymes in crop plants, PlantGDB [Duvick et al., 2008] is a database of plant genomic sequences (ESTs), Expressed Sequence Tags, that correspond to fragments of genes that are actively transcribed under particular conditions. Recently, a few public databases were created to store both genotypic and phenotypic data, and among them can be cited PhenomicDb [Kahraman et al., 2005], Germinate [Lee et al., 2005], Panzea [Zhao et al., 2006], AppleBreed [Antofie et al., 2007], PlantDB [Exner et al., 2008] and Sol Genomics Network [Mueller et al., 2005]. The PhenomicDb is a phenotype-genotype database that allows to browse, from different organisms, known phenotypes for a given gene. Germinate is a database that integrates phenotypic and genotyping information for different species, although access to it is not completely public. Built specifically for maize crops, Panzea permits advanced Genomic Diversity and Phenotype Connection (GDPC) search. Although designed for perennial crops, where it supports apple breeders and geneticists in their genetic studies and in their exploration of germplasm collections also for trait/marker associations, AppleBreed is sufficiently generic to allow the use on other perennial crops and the management of pluri-annual data on the same individual plants. PlantDB, which focuses on experimental and research purposes, allows the management of plant genetic resource collections by using a simple

and versatile MS Access-based downloadable database. The Sol Genomics Network (SGN) is specific for the Solanaceae and Astreid plants, in particular pepper, potato, tomato and petunia and is a clade oriented database. While the CEREALAB database is designed to be a comprehensive tool to support the breeders in their activities, the SGN database contains just a very basic tool for breeders that allows them to do some simple query like the association between traits and QTL and none of the others databases is suitable for the breeding activity.
The growth of the web-based applications used by the bioinformatics community to share information imposes the need to take into account their usability to ensure better use by the researchers [Javahery, 2004]. The community also acknowledges that the Human Computer interaction can be very important to improve the utility and usability of these applications [Hochheiser and Shneiderman, 2003, Bartlett and Toms, 2005]. The usability of a system can be critical for the achievement of results by researchers, giving them the possibility to exploit the information more effectively and efficiently in a time saving manner, may enable them to gain improved insights into biological processes with the potential of producing new scientific results. In [Bolchini et al., 2009] specific design principles to improve the usability of web bioinformatics applications were proposed; these specifically concern the design of the navigation and search mechanisms available to the user. Most of such usability design principles have been applied to implement the new web interface; in particular the access structure were designed following the breeders way of reasoning. From the perspective of the plant breeders community, a very important need is searching a genotype that satisfies more than one condition, for example a variety that is resistant to more than one disease, and contemporarily contains genes determining superior grain quality. To perform a query like that, as a first possibility users could know well the database structure and do a complex query manually. A second solution that can greatly improve usability of the database is to offer the breeders a ready to use query, implemented in an easy to use graphic interface, with a guideline of how to compound the interrogations by inserting/selecting the query parameters in the interface.
The first release of the CEREALAB database [Milc et al., 2011] has been completely renovated from a content and usability point of view. Regarding usability, in the first version of the CEREALAB database, a Graphical User Interface (GUI) was available as a Java Web Start application [Sala and Bergamaschi, 2009] A significant extension to this first version, was the development of a web-based, user-friendly GUI. This new web interface follows the design principles proposed in [Bolchini et al., 2009]. Moreover, structured reports were introduced to provide breeders with an effective tool for the data analysis. Finally, a data entry module was created in the interface, to directly insert new data by the CEREALAB researchers. Regarding the content of the CEREALAB database, its integrated schema was modified to include new tables in the integration process, like 'Lo-

cus' and 'Allele', in order to offer additional interesting data for the breeders.

3.2 Problem Domain

In this section I provide a brief description of some main terms used in the biological domain and in the CEREALAB database to facilitate their comprehension in the rest of the chapter. The main entities used in the CEREALAB domain are:

- **Gene**: it is the unit of inheritance in living organisms. The genes are contained in the genome of an organism, which may be composed of DNA or RNA, and directing the physical and behavioural development of the organism. Most of the genes encode proteins, which are the macromolecules more involved in biochemical and metabolic processes of the cell. Many genes do not encode proteins but produce non-coding RNA, which can play a key role in the biosynthesis of proteins and gene expression.
- **Marker**: it is a polymorphic sequence (with multiple allelic variants) that is located in a specific locus within the chromosome. It identifies a chromosomal regions in which may be localized important gene or QTL whose segregation can be difficult or impossible to follow.
- **Allele**: it is meant any form of DNA coding for the same gene: in other words, the allele is responsible for the particular manner in which it manifests the inherited trait controlled by that gene. For example, a gene that controls the character (eye color) can exist in two alleles (i.e. in two alternative forms): the allele (clear eye) allele and the (dark eye).
- **QTL**: (quantitative trait locus) it is a genomic sequence that has a role in the expression of a quantitative trait. If it is associated with the expression of a trait, it probably contains a gene that is involved in the expression of that trait. They identify a marker contiguous / associated to it, since this could be used for the selection of the desired character simply evaluating the presence / absence of specific allelic variants of the marker.
- **Locus**: A unique genomic location within a chromosome that allows to define the position of a gene or of any DNA sequence.
- **Germplasm**: it represents the total genetic variability available for a specific population of individuals. It is represented by seeds, tissues or cells that can restore an entire organism. In the case of cultivated species, the conservation of germplasm is particularly important as it allows to maintain a high biodiversity of a given species and represents the basis for the insertion of new useful traits

- **Trait**: is the expression of the gene / QTL in a visible way. An example of a trait could be the length of a plant that is determined by the presence/absence of gene / QTL that determines the length of the plant.

Each of these entities has its own attributes, for example the Gene and QTL entities has a chromosome (Structural unit of nuclei which carries the genes arranged in a linear fashion. Also considered as a set of genes organized according to a linear succession that tend to be inherited together) attribute that determine where a gene/ QTL is located in the genome.

3.3 Data Sources

As mentioned before, the purpose of the CEREALAB database is to provide the breeders with a unique tool where they can find data of interest without looking in different data sources and to provide them with a User-friendly interface to facilitate users with less IT and biology expertise. The CEREALAB database is specific for three species of cereals: wheat, barley and rice, thus the selected data sources to be integrated treat at least one of this three species. The selected data sources can be divided in two categories, one with the molecular data and the other with phenotypic evaluations.

Data sources for molecular data: Among the available databases for molecular data we choose the most relevant databases for the wheat, barley and rice: the Graingenes [3] database as a source for wheat and barley, and the Gramene [4] database as a source for rice.
In the 1992 the US Department of Agriculture created the graingenes database to collect and distribute information about wheat, barley, rye, oat and their wild relatives [Matthews et al., 2003]. the Graingenes database is an international database for genetic and genomic information that constitutes the main repository for information about genes, QTLs, markers, primers and alleles for these species [Carollo et al., 2005].
Gramene is a comparative genome mapping database for grasses. Genome assembly and annotations, genetic and physical maps/markers, genes, QTLs, proteins, ontologies, genetic diversity data from rice, maize and wheat and orthologous gene sets compound its core [Liang et al., 2008].
These two data sources are integrated with a relational database that contains molecular data obtained from the experiment done within the CEREALAB

[3] http://wheat.pw.usda.gov/GG2/index.shtml/
[4] http://www.gramene.org/

project[5].

Data sources with phenotypic evaluations: Phenotypic evaluations for some traits are also required for the purpose of the MAS process. Data regarding this type of information is available in different databases; Also in this case the data was chosen from some relevant data sources. These data sources are the following:

- **GRIN** [6] (the Germplasm resources Information Network) for barley and wheat where data is available as CSV files.
- The public data of **CRA** [7] (the Agricultural Research Council) for wheat, barley and rice.
- The public data of **Ente Risi** [8] for rice.
- The public data of the **Emilia Romagna variety field trials** for wheat and barley.

Before integrating these data sources we asked the curator of Graingenes to send us a copy to install it locally on our server, while for the Gramene database we downloaded the last release and installed it on the server (The Gramene database is available to download from the database site).We made this choice to avoid bottleneck caused by a remote data source. These two data sources have been persevered without doing any modification and we integrate them with the other data sources presents on the same sever to create the CEREALAB database. Because the content of Gramene and Graingenes is always increasing, we decided to periodically update the copy of these two databases installed locally in our server to offer to the breeders up to date data.

3.4 The CEREALAB data integration process

In the first version of the CEREALAB database, CEREALAB V1.0, we exploited the MOMIS system [Milc et al., 2011]. MOMIS is a mediator system that performs the integration of heterogonous data sources in a semi-automatic way and create a global schema that can be seen as an ontology emerging from the source

[5] http://www.cerealab.org/
[6] http://www.ars-grin.gov/
[7] http://sito.entecra.it/portale/index2.php/
[8] http://www.enterisi.it

schemas [Bergamaschi et al., 2011]. Details about the ontology of the CEREALAB database V1.0 can be found in [Sala and Bergamaschi, 2009]; in particular, in [Sala and Bergamaschi, 2009] the MOMIS semi-automatic approach to build the GS of the CEREALAB Data Integration Application is described.

In subsection 3.4.1 we focus on one of the main classes of this Global Schema and we will discuss in details its mapping table; this example will be also used in the rest of this thesis. In subsection 3.4.2 The CEREALAB Ontology, we introduce the Ontology and we will discuss how some global classes, automatically generated in the integration phase, were modified by the integration designer, to take into account the feedbacks from the end-users.

3.4.1 An example: The Germplasm global class

In this section, we focus on the *Germplasm* global class (see Figure 3.1) obtained by integrating two local classes `A` and `B` which store data about germplasms coming from two different data sources: `A` stores data obtained by experimental results within the CEREALAB project, `B` stores data coming from other public data sources (see3.3 for a fully description of these sources).

Local classes

`A (GermplasmA)`

GPN	yield	FHB	type
Eureka	18	MR	
Fortuna	7	MR	
Mentana		S	line
Kenora	20	MR	landrace
Oasis	21	MR	cultivar

`B (GermplasmB)`

GPN	yield	FHB	type
Eureka	6	S	cultivar
Fortuna	15	S	landrace
Mentana	20	MR	line
Kenora			cultivar

Mapping Table of the global class `GERMPLASM`

GERMPLASM	A	B
GPN	GPN	GPN
Yield	yield	yield
FHB	FHB	FHB
Type	type	type

Figure 3.1: Example: two local classes with two conflicting attributes

The attributes of these tables represent the following information:

- `GPN`: the name of a variety;

- FHB: the resistance of the germplasm to the *FHB* disease (Fusarium Head Blight) (as values are S=susceptible, MR= Moderately Resistant and R= Resistant);

- Type: the variety's type;

- yield : the *grain yield* expressed in tons/hectare.

Figure 3.1 shows the Mapping Table of the global class GERMPLASM, with schema GERMPLASM(GPN,Yield,FHB,Type), obtained by integrating A and B. As discussed in [Sala and Bergamaschi, 2009] GERMPLASM plays the role of performing *data fusion*. To identify multiple local tuples coming from local classes and representing the same real-world object, we assume that error-free and shared object identifiers exist among different sources: two local tuples with the same object identifier ID indicate the same object in different sources; thus we can use L^{id} to denote the tuple t of a local class L with ID equal to id, i.e. $t[ID] = id$. In our example, we assume GPN as an object identifier; then the first tuple of the local class A, i.e. the tuple with GPN $= Eureka$, will be denoted with A$(Eureka)$. For conflicting attributes the following strategies are used:

- FHB (*trust your friends* strategy): FHB=COALESCE(A.FHB,B.FHB).
  Conflicts are solved preferring the data coming from the A Italian source to offer the Italian breeders information from Italian studies and so nearest to their needs;

- Type (*pass it on* strategy): Type= ALLVALUES(A.type, B.type)
  A germplasm type is unique, so when we find two different types for the same germplasm we know that uncertain data are present. In the case of *Kenora*: we have two types but we do not know which value is correct. To avoid choosing the wrong value all values are maintained and conflict resolution is deferred to the user;

- Yield (*meet in the middle* strategy): Yield=AVG(A.yield, B.yield) represents the average of grain yield in the local classes.

Finally, the mapping query of the global class GERMPLASM (and then the instance of GERMPLASM) is defined by means of the *full outerjoin-merge operator*. Intuitively, as shown in Figure 3.2, this corresponds to the following two operations:
(1) Computation of the *full outerjoin*, on the basis of the shared object identifier, of the *local classes* of GERMPLASM;
(2) Application of the *Resolution Functions*. [9].

[9] COALESCE is the (standard SQL) function which returns its first non-null value;

Mapping Query of GERMPLASM

```
SELECT  GPN = GPN,
        Yield=AVG(A.yield, B.yield)
        FHB = COALESCE(A.FHB,B.FHB),
        Type = ALLVALUES(A.type, B.type)
FROM    A FULL OUTER JOIN B
        USING (GPN)
```

Instance of GERMPLASM

GPN	Yield	FHB	Type
Eureka	12	MR	cultivar
Fortuna	11	MR	landrace
Mentana	20	S	line
Kenora	20	MR	landrace,cultivar
Oasis	21	MR	cultivar

Figure 3.2: Mapping Query and Instance of the global class GERMPLASM

3.4.2 The CEREALAB Ontology

The CEREALAB database is composed of three sub-databases, one for wheat, one for barley and one for rice. Each section is divided in two sub-ontologies: 'Genotypic data' and 'Phenotypic data'. In the new version two global classes: 'Allele' and 'Locus', omitted in the initial integration phase, were added to the Global Schema. In this activity, the integration designer was supported by methodologies and tools for Global Schema evolution provided by the MOMIS system, as described in [Beneventano et al., 2003a]. As a consequence, the Genotypic sub-ontology has been modified and two global classes were added: 'Allele' and 'Locus'.
The 'Allele' class stores the following information:

- Allele: the name/symbol of the allele variants.
- Comment: a comment about the allele.
- Synonym: a synonym for the allele.
- Reference: the reference name from which this data was obtained.
- Trait_affected: trait affected by the allele.

AVG is a (non standard SQL) function to compute the average value;
ALLVALUES is a (non standard SQL) function which returns all non-null values.

The 'Locus' class stores the following information:

- Locus: the locus name/symbol.
- Synonym: the synonym of the locus.
- Type: the type of locus, that can be, gene, marker etc.
- Chromosome: the information on chromosome where the locus is located.
- Map: an external link to a map where we can see the locus location on the respective chromosome.
- Comment: a comment about the locus.
- Near_by_loci: information about loci near the searched locus.
- Reference: reference for the data.

Further formation about which allele is for gene, marker or germplasm can be retrieved from the relationships between this new global class 'Allele' and the global classes 'Gene', 'Marker' and 'Germplasm' created before. Other informations about which is the locus for a marker, gene, allele and QTL can be retrieved from the relationships between this new global class 'Locus' and the global classes 'Marker', 'Gene', 'QTL' created before and the new global class 'Allele'.
Regarding the Phenotypic sub-ontology, in the first version of the database, the data were divided into super-classes and then each super-class was divided into several sub-classes. In the new version, we cancelled the super classes and we put all traits together for two reasons: the first one is the traits number; since we have a few traits, users can see it all without spending time to search trait/s in superclass/es. The second, is to avoid ambiguity. Because the subdivision is not always unique; for example the "kernel color" subclass was in the "growth and development" super-class but it can also be in the "Anatomy and morphology" super-class and vice versa. The new schema of the CEREALAB integrated ontology is presented in Figure 3.3.

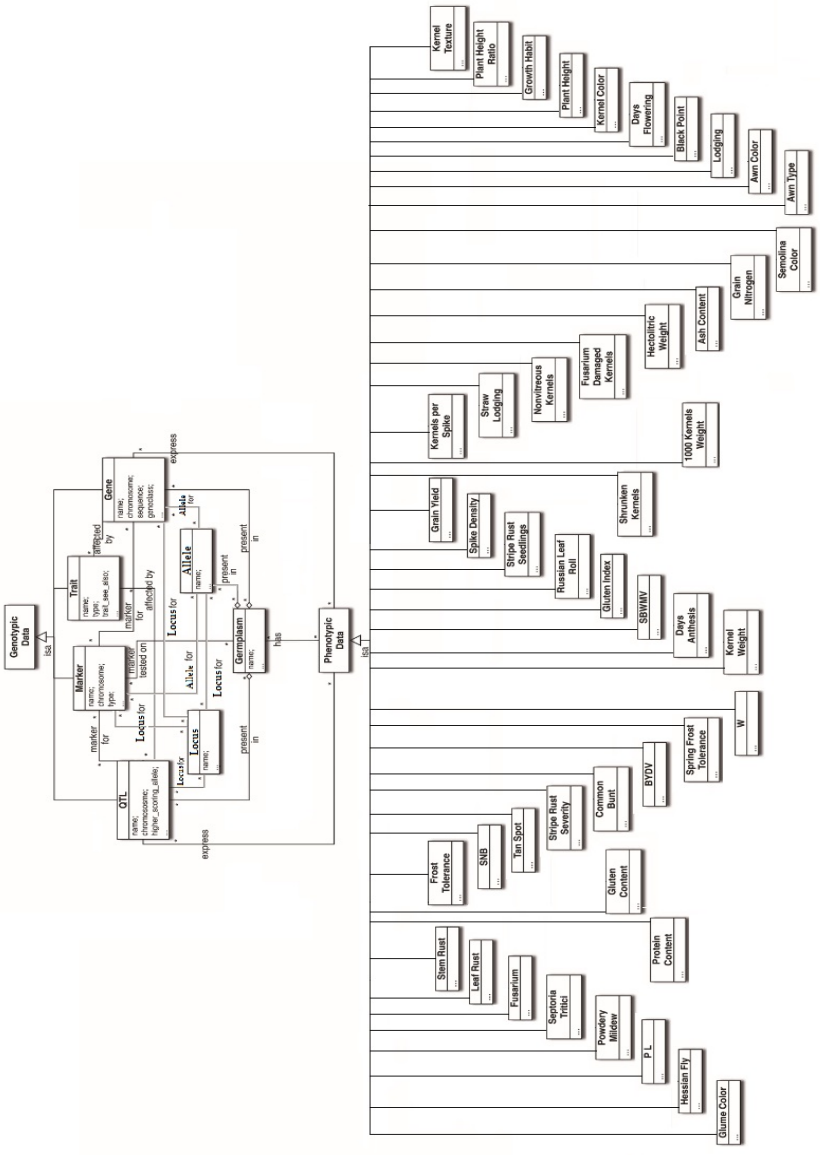


Figure 3.3: The new CEREALAB Ontology

3.5 The CEREALAB web interface

The CEREALAB database V2.0 is accessible using a common web browser like Internet Explorer, Mozilla Firefox, Chrome, etc. Hidden, ready to use queries are implemented in the interface, to simplify the use of the system by the less skilled users, and the final queries are presented like a simple questions that the user just needs to complete using one of the visualized options. The results are presented like a simple structured report that contains molecular and/or phenotypic information about the searched object (e.g. Gene, Marker, Germplasm) and can be saved by the final user. The interface gives to the final user the possibility to perform a simple vs. complex query, browse the data, save the final result, and also insert new data following the procedure described below. The user can send feedbacks by contacting the curator at the e-mail address presented in the interface. The interface is very user-friendly and contains a guideline about how to use each of its sections and about the meaning of each output. This can help to formulate queries and analyse the results. To query the CEREALAB database, a Graphical User Interface (GUI) realized using PhP (Hypertext Preprocessor), is available as a web-based application. In the 'Database' section at www.cerealab.org, Choosing 'Access the Database' then "Enter the Database" opens the application and users have to select one of the following schema: 'Wheat', 'Barley' and 'Rice' to start querying. Once the schema has been selected, the user can use the different functionalities present in the menu to retrieve the information of interest in a very simple way and without the necessity to have any informatics skills to do that.

3.5.1 Functionalities

As the requirement for a new GUI, also the new functionalities were derived from both the use of the database by the developers and project partners, and the interviews with the industrial partners, plant breeders of SIS (Società Italiana Sementi) and PSB (Produttori Sementi Bologna) seed companies. The following new functionalities were added:

- **Browse**: it allows users to browse the most important tables of the database and to visualize their content. Selecting, for example, 'Gene' the list of all available genes is obtained, and by clicking a gene from the list, the user retrieves a structured report containing all the molecular and phenotypic information present in the database for the selected gene.

- **Search**: it allows the users to find an object with a known name or find objects that contain one or more keywords anywhere in their text, in both genotypic and phenotypic data, and access its report. A very simple guide-

line is implemented in the correspondent page to help users during their search through the listboxes.

- **Querying Germplasm**: The functionality allows to search through germplasm either for genotypic, or phenotypic data, or genotypic plus phenotypic data, by selecting one of the three sub-tabs under the tab 'Germplasm'. A very simple guideline to explain how to use this page is implemented in the corresponding section. The Genotypic query allows users to find genotypes that contain a certain gene, allele, or QTL. For example, to search a germplasm that contains a specific QTL, users need to select the radio button 'QTL', and then or to choose the QTL from the listbox or to type in the in the text box. Selecting a QTL from the listbox, retrieves a list of germplasm that harbor that QTL, while typing in the textbox, a list of all QTLs that start with the inserted text will appear together with germplasm accessions. If the user selects a QTL from the listbox and inserts a text in the textbox, the search will take into account only the listbox choice. The Phenotypic query allows the users to find accessions that satisfy one or more phenotypic conditions (i.e. levels for Traits). This query gives the breeder the possibility to find the germplasm resistant both to some diseases and to some abiotic stresses, like e.g. frost. When the user chooses to search the phenotypic data, a list of traits is proposed with a listbox, next to each of them, that contains conditions the users can select to compose their queries. For example, if users want to search a variety that has a black awn color and is resistant to BYDV, they just need to select 'Black from the conditions in the listbox next to 'Awn color' and to select 'resistant' in the listbox next to 'BYDV' to see the list of all the accessions that satisfy these conditions. Selecting a variety retrieves a report containing all the relative information available in the database. The genotypic plus phenotypic query is a very important query for breeders, it allows them to search accessions resistant to some disease, with some searched phenotypic feature like 'Awn color' and have a precise gene, allele or QTL; in other word it combines the two queries cited above in one query. When users select the 'Phenotypic and genotypic data' from the sub-menu of 'Gemrplasm', a page with two columns appears; the first contains the same part of the Phenotypic data' query while the second one contains the same part of the 'Genotypic data' query. For example, if breeders want to search a wheat accession characterized by < 2 % difference between the genotype and the field average value for the ash content trait, white awn color and resistant to the black point and contains the 'Glu-A1c' allele, they can satisfy this request by selecting 'resistant black point' from the listbox next to 'Black point' , select < 2 % difference' from the listbox next to 'Ash content', 'white' from the listbox

next to ‘Awn color’ from the left column and select the radio bottom next to ‘Allele’ and select the ‘Glu-A1c’ allele from the listbox from the right column to see the varieties that satisfy the selected conditions, and by clicking a variety, for example ‘Campodoro’ from the list they can see all the information present in the database that concern the selected variety. The described query is visualized in Figure 3.4 and Figure 3.5.

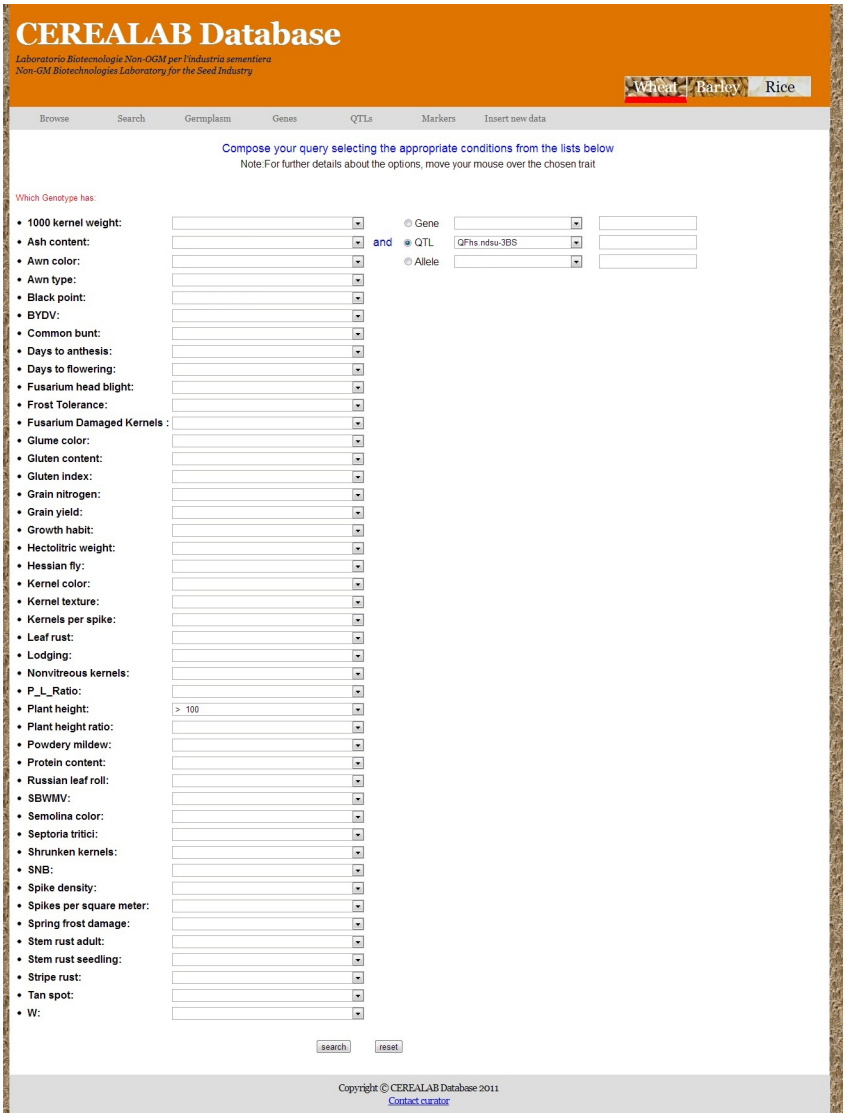
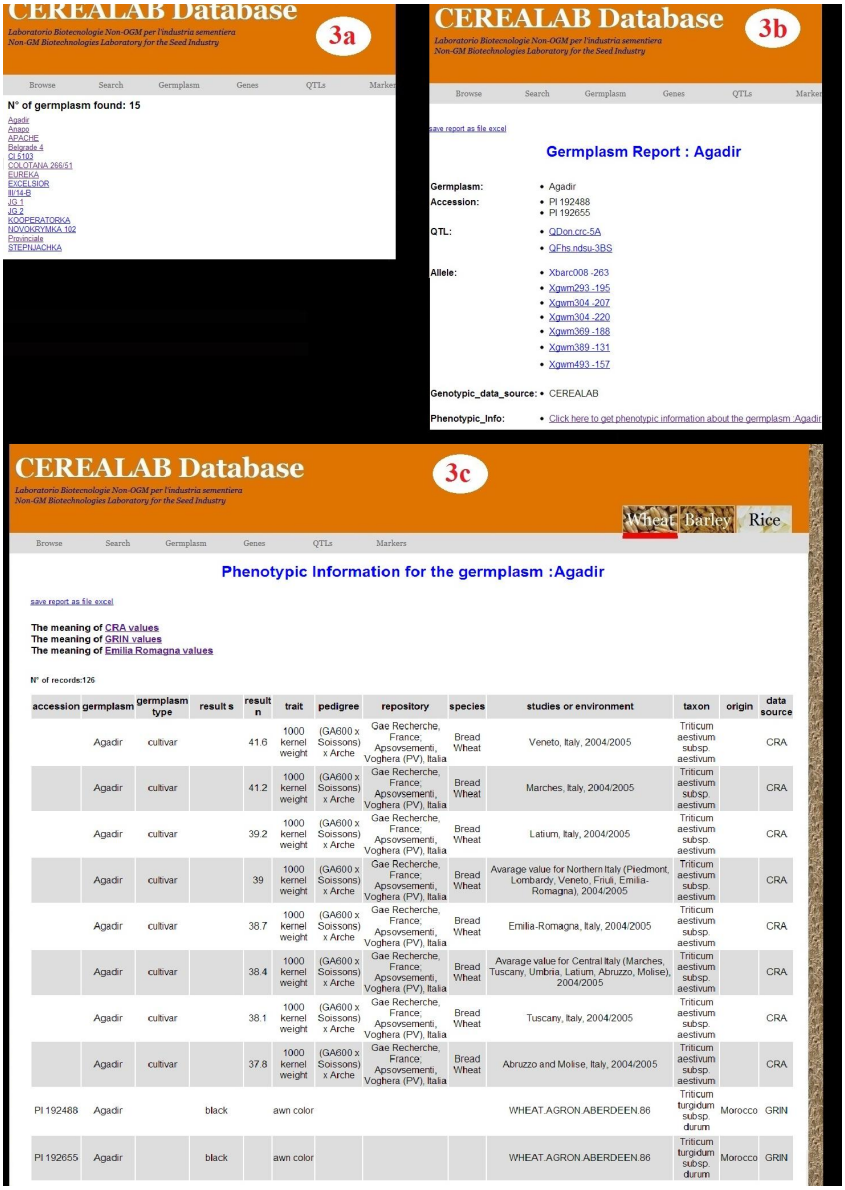


Figure 3.4: Screenshots of the query example



accession	germplasm	germplasm type	result s	result n	trait	pedigree	repository	species	studies or environment	taxon	origin	data source
	Agadir	cultivar		41.6	1000 kernel weight	(GA600 x Soissons) x Arche	Gae Recherche, France; Apsovsementi, Voghera (PV), Italia	Bread Wheat	Veneto, Italy, 2004/2005	Triticum aestivum subsp. aestivum		CRA
	Agadir	cultivar		41.2	1000 kernel weight	(GA600 x Soissons) x Arche	Gae Recherche, France; Apsovsementi, Voghera (PV), Italia	Bread Wheat	Marches, Italy, 2004/2005	Triticum aestivum subsp. aestivum		CRA
	Agadir	cultivar		39.2	1000 kernel weight	(GA600 x Soissons) x Arche	Gae Recherche, France; Apsovsementi, Voghera (PV), Italia	Bread Wheat	Latium, Italy, 2004/2005	Triticum aestivum subsp. aestivum		CRA
	Agadir	cultivar		39	1000 kernel weight	(GA600 x Soissons) x Arche	Gae Recherche, France; Apsovsementi, Voghera (PV), Italia	Bread Wheat	Avarage value for Northern Italy (Piedmont, Lombardy, Veneto, Friuli, Emilia-Romagna), 2004/2005	Triticum aestivum subsp. aestivum		CRA
	Agadir	cultivar		38.7	1000 kernel weight	(GA600 x Soissons) x Arche	Gae Recherche, France; Apsovsementi, Voghera (PV), Italia	Bread Wheat	Emilia-Romagna, Italy, 2004/2005	Triticum aestivum subsp. aestivum		CRA
	Agadir	cultivar		38.4	1000 kernel weight	(GA600 x Soissons) x Arche	Gae Recherche, France; Apsovsementi, Voghera (PV), Italia	Bread Wheat	Avarage value for Central Italy (Marches, Tuscany, Umbria, Latium, Abruzzo, Molise), 2004/2005	Triticum aestivum subsp. aestivum		CRA
	Agadir	cultivar		38.1	1000 kernel weight	(GA600 x Soissons) x Arche	Gae Recherche, France; Apsovsementi, Voghera (PV), Italia	Bread Wheat	Tuscany, Italy, 2004/2005	Triticum aestivum subsp. aestivum		CRA
	Agadir	cultivar		37.8	1000 kernel weight	(GA600 x Soissons) x Arche	Gae Recherche, France; Apsovsementi, Voghera (PV), Italia	Bread Wheat	Abruzzo and Molise, Italy, 2004/2005	Triticum aestivum subsp. aestivum		CRA
PI 192488	Agadir		black		awn color				WHEAT.AGRON.ABERDEEN.86	Triticum turgidum subsp. durum	Morocco	GRIN
PI 192655	Agadir		black		awn color				WHEAT.AGRON.ABERDEEN.86	Triticum turgidum subsp. durum	Morocco	GRIN

Figure 3.5: The result (a), the report (b) and the phenotypic data (c) for the query example

- **Querying Genes**: allows users to find the genes that underlie a selected trait. In this section, users find a list of radio buttons associated to all the studied traits. For example selecting the Septoria tritici radio button retrieves a list of genes that underlie the resistance to this disease; then the user can select one gene to see its report. If no results are returned, the system will suggest user to move to the QTL section, because the trait may be regulated by a QTL instead of a gene.

- **Querying QTLs**: allows users to find the QTLs that underlie a selected trait. The query is executed in the same way as for the 'Genes' section.

- **Querying Markers**: allows users to find markers for a gene, a QTL or a trait. In this section, users find three radio buttons, one for gene, one for QTL and one for trait with the relative associated listboxes and textboxes. To search a marker for a specific QTL, users need to select the radio button 'QTL', to select a QTL from the listbox or to type in in the text box. Selecting a QTL from the listbox, retrieves a list of associated markers, while typing in the textbox a list of all QTLs that start with the inserted text, they will appear together with their associated markers. The user can then select one QTL or associated marker from the obtained list, to see its report. When both a QTL is selected from the listbox and the text is typed in the textbox, the search will take into account just the selected value from the listbox. The query is practically identical for markers associated to genes, as well as for traits. A very simple guideline to explain how to use this page is implemented in the corresponding section.

The sample queries for all the main functionalities of the database are presented in the Figure 3.6.

3.5.2 Structured Reports

A number of useful reports are generated on database content according to user's needs. This list of available reports is likely to grow as users request new reports which might be of use to others. The reports currently available are: Allele, Gene, Germplasm, Locus, Marker, QTL, and Trait. Each of these reports contains all the molecular and/or the phenotypic data present in the database and concerns the searched object. These reports, structured to facilitate understanding and analysis of results by the end users, can also be saved as MS Excel files. An example of the report for the 'Campodoro' variety is shown in Figure 3.5 section 3b and 3c.

Query 1. Retrieve all available phenotypic data for the trait "Frost tolerance" in barley
Barley schema → Browse → Phenotypic classes → Frost tolerance

Query 2. Retrieve data regarding wheat Pm31 gene (resistance to Powdery mildew)
Wheat schema → Search → Genotypic data → Select class: gene → Select column: name → insert: Pm31 → click the search botton

Query 3. Retrieve rice germplasm genotyped with markers associated to Pita gene (resistance to Pyricularia grisea)
Rice schema → Germplasm → Genotypic data → Select radio button 'Gene' → Choose 'Pita' from the listbox or Type 'Pita' in the texbox

Query 4. Retrieve rice germplasm characterized by high amylose content, resistant to leaf blast and to lodging
Rice schema → Germplasm → Phenotypic data → Choose from the listbox: (Amylose content: >25%) (Leaf blst: 1 no infection) (Lodging: < 10%)

Query 5. Retrieve wheat genes for 'Frost tolerance'
Wheat schema → Genes → Frost tolerance

Query 6 Retrieve barley QTLs for 'Beta glucan' content
Barley schema → QTLs → Beta glucan

Query 7. Retrieve molecular markers that can be used in marker assisted selection (MAS) for resistance to Fusarium spp. in wheat
Wheat schema → Markers → Select radio button 'Trait' → Chooses 'Fusarium head blight' from the listbox or type 'Fusarium' in the textbox

Figure 3.6: Examples of possible queries deploying the new functionalities

3.5.3 A Data Entry Module

Another extension to the CEREALAB database v1.0, was the development of a data entry module. In the CEREALAB database v1.0, the data obtained by the CEREALAB project were all entered manually in a relational database, called CEREALAB local source - CEREALAB LS. In the CEREALAB database v2.0, an "insert new data" module was implemented for the CEREALAB local source, to help the CEREALAB project members in the process of insert the data resulting from the new experiments. With the virtual approach offered by MOMIS system, the new inserted data are immediately available at the level of the CEREALAB database v2.0 so when the new data is inserted, it will automatically appear in the new queries; in fact, queries are executed directly on the data sources that are integrated in the CEREALAB database, thus providing always up-to-date information.
First of all, the 'Insert new data' module implements an user authentication method based on username/password. The main functionality of the 'Insert new data' module is a list of links, each of them allowing the insertion of the new data in a specific table of the CEREALAB LS. Selecting a link opens a form composed of columns of the selected table where the user can insert the new data. At the time of data insertion into the database the system controls if the inserted data are already present in the table. If data are present, a message notifying the presence will appear and suggest the user to see the report of the object he tries to insert. The data entry module allows users to insert genotypic or phenotypic data manually and directly in the corresponding table, or they can be imported from a CSV file.Figure 3.7 shows the operation of inserting a gene in the database.

3.6 Discussion

The CEREALAB database v2.0 was created to improve the previous version; our objectives were to extend of the database structure to include new interesting data; to resolve the usability problem of the interface by making it breeder friendly and to implement a data entry module to insert the new data by the CEREALAB research group.
The new version as it is realized offers new interesting data. Breeders can found information about 'Allele' and 'Locus'. Also, they can found the relationship between these new two classes and the others classes to have a complete vision of the biological data of the searched variety. In this way the breeder is able to find directly a variety that harbours a certain allele of a gene or marker. For example knowing that higher scoring allele of Sumai3 of the SSR marker Xgwm493 gives the amplification product of 194bp he can retrieve from our database the list

and all information about varieties that harbour this allele, and use one of them in his marker assisted breeding program using the PCR protocol for the marker Xgwm493 that he finds thanks to the relationship between the Global Classes 'Allele' and 'Marker'.
The new interface has been designed following the suggestions that were given to the database creators by the breeder end-users trying to resolve the most common problems arose during the use of the database to make it more user friendly. Breeders do not need to have advanced informatics skills to deal with it. It contains prepacked queries that cover the majority of the breeders needs. The results are presented in a structured way (reports) and contain molecular and/or phenotypic data and can be saved as Excel files that can be consulted also offline. The interface is a web application that can be used with common browsers, without the necessity to install any other programs on the own pc. It also contains many help sections that can support the user while doing the interrogations, some of this help sections are accessible by clicking a link as the help section of the 'Genotypic data' sub-menu, other are accessible just by moving the mouse cursor over the label present in the interface as the case of the 'Phenotypic Data' and 'Genotypic and Phenotypic Data' sub-menu, users have to move the mouse courser on the label to see the meaning of the listboxes values. A comparison between the sample query described for the CEREALAB database v1.0 [Milc et al., 2011] and the same query available in the new version can be used as an example of the usability improvement that was obtained for the database. To execute the query allows the breeder to find both the phenotypic and genotypic regarding germplasm accessions of his interest the user needed to select first the subontology PHENOTYPIC DATA, then to add to the selection the 'Referenced Class' Marker_Tested_on_Germplasm. After adding in the 'condition' panel the desired condition for the selected attribute, clicking 'add condition' and the 'execute query' button, all the phenotypic data available for cultivars that met the criteria were retrieved together with the molecular data. In the present version of the database the user just has to choose the "Germplasm" class form the main menu, then the "Genotypic and Phenotypic data" choose the condition for the trait he is interested in from the left column listbox to retrieve the phenotypic data he looks for. Moreover choosing "Gene", "QTL", or "Allele" from the relative listboxes or typing directly in the textboxes in the right column will retrieve all the molecular and genotyping data available. Performing such a combined phenotypic-genotypic data is now much more user friendly respect to the previous database version.
Apart from being more user-friendly the new version of the database can also be implemented in a more easy way as it gives the CEREALAB research group members the possibility to insert the new data obtained by their experiment directly from the interface. In the previous version the new data was entered into

Spread Sheet formats by each researcher and then loaded by the curator. In this new version no intermediates are necessary; indeed any authorized researcher can access the 'CEREALAB LS' database with his credentials to insert the new data. They can select to import data from a CSV file with the same structure of the corresponding table, that can see by clicking a link in the interface, or to insert it manually by selecting the corresponding table from the list. The approach used in the MOMIS system allows them to see the new data in the new queries.

The CEREALAB database v2.0 has been created by integrating different open-source databases using the MOMIS data integration system. MOMIS allows the integration of infinite number of heterogeneous sources so in the future other data sources may be incorporate in the integration process to enrich the available information. Moreover, other sub-databases regarding other species can be created meeting any possible breeders need. Also, the creation of new modules and the implementation of new functionalities to answer the breeders need may be implemented in the future.

Figure 3.7: An example of inserting two genes, one present in the database and the other is new.

Chapter 4

On Provenance of Data Fusion Queries

Data Lineage is an open research problem. This is particularly true in data integration systems, where information coming from different sources, potentially uncertain or even inconsistent with each other, is integrated. In this context, having the possibility to trace the lineage of certain data can help unravelling possible unexpected or questionable results.
In this chapter, we describe our work about this problem in the context of the MOMIS data fusion system. We discuss and compare the use of *lineage* and *why*-provenance for the data fusion operator used in the MOMIS system; in particular we evaluate how the computation of the *why*-provenance should be extended to deal with *Resolution Functions* used in our data fusion system. The remainder of the chapter is organized as follows. In section 4.1 we will present the problem of data provenance in the data integration systems; In section 4.2 we will informally discuss the use of *lineage* and *why*-provenance for data fusion queries and we will present the Perm system, then in section 4.3 we formally define these concepts. Finally, conclusions and future works are sketched in section 4.4. The work presented in this chapter was published in [Beneventano et al., 2011b, Beneventano et al., 2011a],

4.1 Problem Definition

Lineage, sometimes called *provenance*, in its most general definition, describes where data came from, how it was derived and how it was modified over time. Lineage provides valuable information that can be exploited for many purposes, ranging form simple statistical resumes presented to the end-user, to more complex applications such as managing data uncertainty or identifying and correcting data errors. For these reasons, in the last few years the research activity in the Information Management System area has been increasingly focused on this topic. In Particular, lineage has been studied extensively in data warehouse systems [Cui et al., 2000, Cui and Widom, 2003]. However, in *Data Integration* systems, lineage is still considered as a open research problem [Halevy et al., 2006a, Halevy and Li, 2003]. Data Integration systems deal with information coming from different sources, potentially uncertain or even inconsistent with each other. In this context, collecting lineage information becomes a necessity. Lineage information helps the integration process by improving the system capability to introspect about the sources reliability and the certainty of the data.
An interesting challenge presented in [Moreau, 2009, Jagadish et al., 2007] regards human-computer interactions. Many operations performed by Integration Systems, ranging from schema-mapping design to querying, involve humans in the loop, who impact on decisions and processes. This scenario clearly delineates

the necessity to include data lineage in human-computer interactions. By explaining how data are integrated, lineage plays an important role in this process and improves the quality of the human-computer interaction itself. Moreover, lineage has been used to debug schema mappings that may be imprecise or incorrect. The debugging tool developed in [Chiticariu and Tan, 2006] describes the relationship between source and target data with the schema mapping, allowing designers to see how 'bad' data has been produced.
As explained in section 1.2 the *data fusion* process is a fundamental task in data integration. This process involves the resolution of possible conflicts between data coming from different data sources [Bleiholder and Naumann, 2008].A recent tutorial [Dong and Naumann, 2009] listed data lineage as one of the open problems and desiderata for data fusion systems. Merging data implies a partial loss of the original values of the local sources. For this reason database administrators and data owners are notoriously hesitant to merge data. Data lineage can help explaining merging decisions by tracking which original values were involved and how they have been fused together.
In this chapter, we describe our work on the application of data lineage techniques in the Data Fusion framework of the MOMIS Integration System. By implementing a data lineage support in the MOMIS framework, we want to realize a new component with these new capabilities:

- *Explanation of merging decision:* while querying the system, users can inspect lineage information together with the query results. Thanks to the lineage, the integration process is no more seen as a 'black box', and the users have a better understanding of how the final results were produced and where they came from.

- *Error correction at the data level through user feedback:* by exploiting lineage information, the system performances can be improved in many directions. An important application is data error correction. Since MOMIS cannot modify the data on the local sources, error correction can only happen at the global virtual view level. Data errors are identified by the end-users who check the query results and find out which local sources caused the error using the lineage. The system keeps track of the errors indicated by the end-users, and automatically corrects the results in the next queries.

- *Error correction at the schema mapping level:* lineage is also useful for debugging the schema mapping. When users identify errors that happens constantly, they can use the lineage information to inspect the transformation and reconciliation functions involved in the merging process. For instance, the integration designer may decide to add other functions (for example fil-

ter functions for NULL values), or to change existing functions previously set.

The final result is an improved human-computer interaction. Through lineage and user feedback, MOMIS provides high quality information to the user and improves its integration process over time.
In the rest of this chapter, we discuss and compare the use of *lineage* and *why*-provenance for the data fusion operator used in the MOMIS system; in particular we evaluate how the computation of the *why*-provenance should be extended to deal with *Resolution Functions* used in our data fusion system.

4.2 Provenance for Data Fusion Queries in MOMIS

In this chapter we refer to the definition of *lineage* and *why*-provenance as formulated in [Cheney et al., 2009] for relational databases. The concept of *lineage* for relational databases was introduced in [Cui et al., 2000]:the *lineage* of an output record is based on identifying a subset of input records relevant to the output record; intuitively, an input record is relevant to an output record if it contributed to the existence of that output record. As in [Cheney et al., 2009], we consider *lineage* as a subinstance of the input, whereas in [Cui et al., 2000] lineage is defined as a vector of subsets of the input relations; this is a minor difference in presentation.
Buneman et al. [Buneman et al., 2001] defined the notion of *why*-provenance in terms of a deterministic semistructured data model and query language. *Why*-provenance is based on identifying subinstances of the input that "witness" a part of the output, i.e., *why*-provenance encodes all the *possible different derivations* of an output tuple in the query result by storing a set of input tuples *for each derivation*. In [Cheney et al., 2009], *why*-provenance is reformulated in terms of the relational model and relational algebra query language.
Both these concepts of *lineage* and *why*-provenance are defined for unions of conjunctive queries, and are then insufficient to capture provenance for data fusion queries based on the full outer join merge operator, which involves a form of negation. For this reason, we first informally discuss how the concepts of *lineage* and *why*-provenance should be extended to this new data fusion operator (section 4.2.1).Then we introduce their formal definitions in section 4.3.

4.2.1 Lineage and Why-Provenance for Projection Queries

For better reading, in this section we report the example illustrated in section 2.2 in Fig. 4.1 . We use q with subscripts to denote queries and Q to denote the

L_1

ID	A	C
1	3	24
2	NULL	20
3	9	NULL
4	8	25
5	NULL	20

L_2

ID	B	C
1	3	24
2	NULL	30
3	NULL	20
5	NULL	30

L_3

ID	C
5	25

$G_1 = G_2$

ID	A	B	C
1	3	3	24
2	NULL	NULL	25
3	9	NULL	20
4	8	NULL	25
5	NULL	NULL	25

Figure 4.1: Instances of the local classes and of the global classes computed with the full outer join merge operator

relation that results from evaluating q. Let G be a global class and let $\mathcal{L}(G) = \{L_1, \ldots, L_n\}$ be the set of its local classes. Given a query on G, we want to express the lineage of a global tuple $t \in Q$ in terms of the local classes of G,thus:

- the *lineage* of t is a set of local tuples, i.e. an element of $\mathcal{P}(L_1 \cup \ldots \cup L_n)$.
- the *why-provenance* (or *witness basis*) of t is a set of sets of local tuples, i.e., is an element of $\mathcal{P}(\mathcal{P}(L_1 \cup \ldots \cup L_n))$.

The *lineage* of a (global) tuple in the result of a query is the set of (local) tuples that were involved in some derivation of that result tuple. We use L^{id} to denote the tuple t of a local class L with object identifier ID equal to id, i.e. $t[ID] = id$. In the same way, for a query q, Q^{id} denotes the tuple $t \in Q$ with object identifier ID equal to id.
We start our discussion by considering Projection Queries on a single global class, where only *one-to-one* attributes (besides the object identifier ID) are selected. A first example is shown in Figure 4.2; in the derivation of Q_1^1 either L_1^1 and L_2^1 are involved, thus the lineage of Q_1^1 is $\{L_1^1, L_2^1\}$. In other words, if a global tuple is derived from the fusion of two local tuples, one from a local class, one from another local class, the lineage shows both these two local tuples.

Q_1

ID	A	B	*Lineage*	*Why-provenance*	*Minimal Why-provenance*
1	3	3	$\{L_1^1, L_2^1\}$	$\{\{L_1^1, L_2^1\}\}$	$\{\{L_1^1, L_2^1\}\}$
2	NULL	NULL	$\{L_1^2, L_2^2\}$	$\{\{L_1^2\}, \{L_2^2\}, \{L_1^2, L_2^2\}\}$	$\{\{L_1^2\}, \{L_2^2\}\}$
3	9	NULL	$\{L_1^3, L_2^3\}$	$\{\{L_1^3, L_2^3\}, \{L_1^3\}\}$	$\{\{L_1^3\}\}$
4	8	NULL	$\{L_1^4\}$	$\{\{L_1^4\}\}$	$\{\{L_1^4\}\}$
5	NULL	NULL	$\{L_1^5, L_2^5\}$	$\{\{L_1^5\}, \{L_2^5\}, \{L_1^5, L_2^5\}\}$	$\{\{L_1^5\}, \{L_2^5\}\}$

Figure 4.2: Lineage for q1=select ID,A,B from G1

Why-provenance [Buneman et al., 2001] encodes all the *possible different derivations* of a global tuple in the query result by storing a set of local tuples *for each derivation*. In our example, there is a unique derivation of Q_1^1, which involve both the tuples L_1^1 and L_2^1, thus its why-provenance is $\{\{L_1^1, L_2^1\}\}$. For the tuple Q_1^3 there are two possible derivations, $\{L_1^3, L_2^3\}$ and $\{L_1^3\}$, thus its why-provenance is $\{\{L_1^3, L_2^3\}, \{L_1^3\}\}$. For the tuple Q_1^2 there are three possible derivations, thus its why-provenance is $\{\{L_1^2, L_2^2\}, \{L_1^2\}, \{L_2^2\}\}$. As it can be seen in the example, when only *one-to-one* attributes (besides the ID) are selected, different derivations are due to the presence of the *NULL* value into the local tuples.

In all the previous examples, we discussed the lineage and the why-provenance of global tuples obtained fusing two local tuples, one from a local class, and the other from a different local class. We now consider the tuple Q_1^4 coming from a tuple of $L1$ that has no corresponding tuple in $L2$. How can we then define the lineage of such a kind of tuple?

Recently, in Perm [Glavic and Alonso, 2009], the concept of *PI-CS*-provenance (*Perm Influence Contribution Semantics*) is introduced to produce more precise provenance information for outer joins. As observed in [Glavic and Alonso, 2009], the lineage of Q_1^4 would contain all tuples from relation L_2, but in fact none of them contributed to Q_1^4. We agree with [Glavic et al., 2010] claiming that a better semantics for the provenance of the tuple Q_1^4 would indicate that L_1^4 paired with no tuples from L_2 influences Q_1^4 (rather than saying every tuple of L_2 is in the provenance of this tuple). On the other hand, though, the *PI-CS* provenance as defined in TRAMP [Glavic et al., 2010] is not able to represent *each derivation*, in fact for tuple Q_1^3 should show only the witness basis $\{\langle L_1^3, L_2^3 \rangle\}$.

In [Glavic et al., 2010] is presented TRAMP (TRAnsformation Mapping Provenance) as an extension to the Perm system to develop a suite of tools supporting the debugging and tracing of schema mappings and transformation queries.

These considerations lead us to define the lineage and *why*-provenance of Q_1^4 in a straightforward way as $\{L_1^4\}$ and $\{\{L_1^4\}\}$, respectively.

Besides the concept of *why*-provenancee, in [Buneman et al., 2001] the concept of *minimal why-provenance* is introduced to represent the *minimal witness basis* which consists of all the minimal witnesses in the witness basis, where a witness is minimal if none of its proper subinstances is also a witness in the witness basis. For example, $\{\{L_1^3\}\}$ is a minimal witness for Q_1^3 whereas $\{\{L_1^3, L_2^3\}, \{L_1^3\}\}$ is not.

We conclude this initial discussion about lineage for the data fusion operator considering the same query posed on the global class G_2 obtained from all the three local classes: we denote this query as q_2. As it can be easily verified, the result of q_2 is the same as the result of q_1, and the lineage of tuples that

are not present in L_3 does not change. On the other hand, lineage of tuple Q_2^5 is $\{L_1^5, L_2^5, L_3^5\}$, while the *why*-provenance is $\mathcal{P}(\{L_1^5, L_2^5, L_3^5\}) \setminus \emptyset$ (i.e., all the possible non-empty subsets of the set $\{L_1^5, L_2^5, L_3^5\}$), the minimal why-provenance is $\{\{L_1^5\}, \{L_2^5\}, \{L_3^5\}\}$.
We will now discuss and compare what *lineage* and *why*-provenance produce for query with resolution functions, that represent the most significant aspect in our data fusion framework. To this end, we will consider the query q_3 posed on the global class G_2 (see Figure 4.3)which selects the *one-to-many* global attribute C defined by the AVG resolution function.

Q_3

ID	C	*Lineage*	*Why-provenance*	*Minimal Why-provenance*
1	24	$\{L_1^1, L_2^1\}$	$\{\{L_1^1\}, \{L_2^1\}, \{L_1^1, L_2^1\}\}$	$\{\{L_1^1\}, \{L_2^1\}\}$
2	25	$\{L_1^2, L_2^2\}$	$\{\{L_1^2, L_2^2\}\}$	$\{\{L_1^2, L_2^2\}\}$
3	20	$\{L_1^3, L_2^3\}$	$\{\{L_1^3, L_2^3\}, \{L_2^3\}\}$	$\{\{L_2^3\}\}$
4	25	$\{L_1^4\}$	$\{\{L_1^4\}\}$	$\{\{L_1^4\}\}$
5	25	$\{L_1^5, L_2^5, L_3^5\}$	$\{\{L_1^5, L_2^5, L_3^5\}, \{L_1^5, L_2^5\}, \{L_3^5\}\}$	$\{\{L_1^5, L_2^5\}, \{L_3^5\}\}$

Figure 4.3: Lineage for q3 = select ID,C from G2

The computation of the AVG resolution function corresponds to performing a grouping on the object identifier and to the computation of the AVG aggregation function. On the other hand, either *lineage* and *why*-provenance defined in [Cheney et al., 2009] are limitated to Selection–Projection–Join–Union operations and thus are not defined for grouping with aggregation.
Lineage for grouping with aggregation is defined in TRIO [Benjelloun et al., 2006] and in TRAMP [Glavic et al., 2010]. *Trio*-lineage is similar to *why*-provenance, but derivations involving the same set of tuples are represented separately, i.e., with the *Trio*-lineage, *bag* of sets of input tuples are represented, each of which corresponds to one derivation. For grouping with aggregation, the *Trio*-lineage of a tuple t in the set of all tuples in the group that corresponds to t, and the same holds for the *PI-CS*-provenance introduced in TRAMP [Glavic et al., 2010].
This way to define the lineage for grouping with aggregation can be applied in the case of the lineage for our data fusion operator containing resolution functions, obtaining the straightforward result shown in Figure 4.3; as an example for tuple Q_3^1 the lineage is $\{L_1^1, L_2^1\}$, i.e. the two local tuples which are fused to obtain Q_3^1.
On the other hand, either *Trio*-lineage [Benjelloun et al., 2006] and *PI-CS*-provenance [Glavic et al., 2010] for grouping with aggregation are not appropriate to define *why*-provenance for our data fusion operator containing

resolution functions, since with the *why*-provenance we want to encode all the *possible different derivations*. In other words, we believe a better semantics for the *why*-provenance of the tuple Q_3^1 would be $\{\{L_1^1\}, \{L_2^1\}, \{L_1^1, L_2^1\}\}$, where each set of local tuples which corresponds to one derivation is represented. Another significant example is Q_3^5. In this way, for a global tuple obtained by means of an `AVG` resolution function, the *why*-provenance produces *all* the possible derivations, with a behaviour that is homogeneous with the case of one-to-one attributes discussed before. This is true in the case of the average function, but the application of the *why*-provenance need to be further analyzed for different types of resolution functions.

In [Naumann and Bleiholder, 2006], the properties of the resolution functions are examined; in particular, resolution functions are subdivided into *mediating* and *deciding* functions. A function is *mediating* if $RF(v_1, \ldots, v_n) = y$, meaning that a new value is created by the resolution function. Intuitively, for some mediating functions, such as `AVG` and `MEDIAN`, the *why*-provenance provides several witnesses , while for other ones, such as `SUM` and `CONCAT`, a unique witness is produced.

Deciding functions choose among the already present values, e.g., `COALESCE` or `SHORTEST`, where $RF(v_1, \ldots, v_i, \ldots, v_n) = v_i, i \in \{1, \ldots, n\}$. As observed in [Naumann and Bleiholder, 2006], if we assume that ties (e.g., two shortest values) are broken by a secondary criterion, e.g., the order of the values, we always get a defined result. Intuitively, for deciding functions, the *why*-provenance provides a unique witness with only the local tuple whose value is chosen by the resolution function.

In [Cheney et al., 2009], the relationships between lineage, why-provenance (i.e., the witness basis) and the minimal witness basis are discussed. In particular, the authors show that both lineage and the minimal witness basis can be computed from why-provenance; however, neither lineage, nor why-provenance can be obtained from the minimal witness basis, as it can be verified also in our examples.

Another consideration about the relationship between why-provenance and minimal why-provenance that is peculiar of the data fusion operator is the following: both the second and the third tuple of q_1 of Figure 4.2, i.e., Q_1^3 and Q_1^4, have a null value for the attribute B, but in the case of Q_1^3 the null value is coming from a data source (the local class L_1), while in the case of Q_1^4 the null value is obtained because the attribute B has no mapping on a local source. In other words, the results from query q_1 show no differences between a null value coming from the data sources (tuple Q_1^3) and a null value obtained because the attribute has no mapping on a local source (tuple Q_1^4).

The difference between these two cases can be described in terms of *why*-provenance, while the *minimal why*-provenance remains the same. In other

words, having the possibility to query the *why*-provenance, we can have different results in these two cases, while it is not possible with the *minimal why*-provenance (neither with the *lineage*).

4.2.2 The Perm system

[1]

The Perm system (Provenance Extension of the Relational Model) is a system that provides provenance information for relational databases. It generates provenance information using the query rewriting method (i.e. for a query q, it rewrites the query to produce the same result of q extended with attributes that contain provenance information). The transformed query is also a relational algebra statement so the provenance information can also be queried by means of other sql queries. The result of the provenance computation is presented as relation that contains the result of the original query and the provenance information; in other words, the original result tuples are extended with the base relations tuples, accessed by the original query, that contribute to the output of the original query to give the provenance Information. The provenance attributes are compounded from a prefix and the relation name they are derived from, this is to distinguish between original attributes and provenance attributes. Perm covers a wide range of relational algebra that are not covered by other systems like queries with the full outer join operator.
Perm is implemented as an extension of the PostgresSQL DBMS (see figure 4.4). The postgres query analyzer passed its output to the *Perm rewrite* module that transforms the query (or a part of it) into a provenance query by applying the provenance query rewrite rules. Perm benefits from the query optimization techniques incorporated in the postgres by passing the rewritten query (output of the *Perm rewrite* module) to the Postgres query optimizer.
Perm uses the SQL-PLE (provenance-language-extension) to control and trigger the Provenance computation. The provenance computation query is marked with the new keyword PROVENANCE:

SELECT **PROVENANCE** * FROM relation;

The language extension not influence the SQL features provided by PostgreSQL. Indeed they may be used in combination with the provenance computation and then user can receive provenance information and also query provenance information.

[1] `http://permdbms.sourceforge.net/`

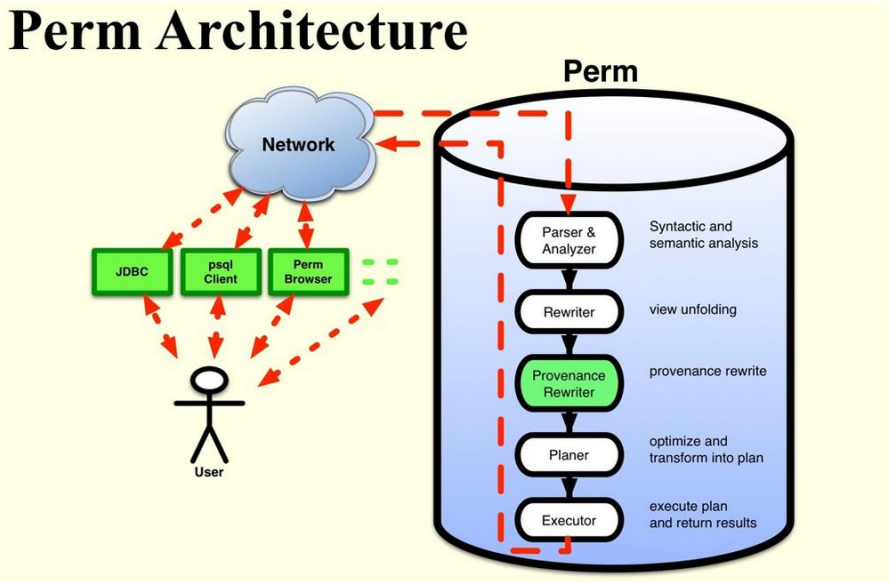


Figure 4.4: Perm Architecture

In the rest of this section I will show the result of using the Perm system. First I will start with a simple example with two queries one with a non-conflicting attribute and the other with a conflicting attribute, then i will show a complex example with the provenance computation for two queries, one with non conflicting attribute and the other one with an aggregate function to show the output result of the Perm system and compare it with the result of *why - provenance* we need to obtain and presented in fig 4.2 and fig 4.3.
For the first example we will refer to the three classes in figure 4.5

C_1

ID	A
1	3
2	3
3	12
4	12

C_2

ID	B
1	4
2	NULL
3	3

C_3

ID	A
1	4
2	NULL
3	3

Figure 4.5: Instances of the three classes C1, C2 and C3

The following query:

```
 Select Distinct A
 From C1 full join C2 Using (id)
 where C1.A > 3 and C2.B > 3;
```

highlights the case of non conflicting attribute because the column A comes just from the class C1 and the result is the following:

```
 a
----
  3
 12
```

If I want to visualize the provenance information, the query to pose is the following:

```
Select provenance * from (
Select Distinct A
 From C1 full join C2 Using (id)
 where C1.A > 3 and C2.B > 3 ) TMP;
```

the result is the following:

```
a   | prov_public  |  prov_public  | prov_public  | prov_public
    | _c1_id       | _c1_a         | _c2_id       | _c2_b
----+--------------+---------------+--------------+--------------
  3 |            1 |             3 |            1 |            4
 12 |            3 |            12 |            3 |            3
 12 |            4 |            12 |              |
```

As we can see in the query result, the provenance information in the Perm system is represented in a relational form so each record is represented by all its attributes, and to these attribute the keyword "prov_public" is added, then the result contains the attributes prov_public_c1_id, prov_public_c1_a ... etc. In presence of tuple identifier (id) we can limit the provenance information on the tuple identifier to simplify the result so we pose the following query:

```
select a, prov_public_c1_id as c1_id,
prov_public_c2_id as c2_id  from
(select provenance a from
  (select distinct a
```

```
  from C1 full join C2 using (id)
  where C1.a > 3 or C2.b >3)tmp
)tmp1;
```

and the final result is:

```
 a  | c1_id | c2_id
----+-------+---------
  3 |     1 |       1
 12 |     3 |       3
 12 |     4 |
```

The provenance information of the output result A = 3 indicates that this result can be obtained by using both record with id = 1 from the two classes C1 and C2 while the provenance information for A = 12 indicates that this output record can be obtained by using both record with id = 3 from the two classes C1 and C2 or by using just the record with id = 4 from the class C1.
In this example we can see how the Perm system represents the witness lists in a relational form, in fact each witness list is represented by a single record.

For the second case (conflicting attribute) i will use the following query:

```
 SELECT DISTINCT COALESCE(C1.A,C3.A)
FROM C1 FULL JOIN C3 USING (ID)
WHERE C1.A > 3 OR C3.A  > 3
```

that returns the following result:

```
  a
-----
  3
  12
```

and in the same way I will pose the following query to see the provenance information:

```
select a, prov_public_c1_id as c1_id,
 prov_public_c3_id as c3_id from
(
```

```
  select provenance *
 from ( select distinct coalesce (c1.a, c3.a)
           from c1 full join c3 using (id)
           where c1.a > 3 or c3.a >3
          )tmp
)tmp1;
```

and the final result is:

```
 a   | c1_id | c3_id
-----+-------+-------
   3 |     1 |     1
  12 |     3 |     3
  12 |     4 |
```

The result indicates how to derive the output result from C1 and C3 but it does not cover all the possibile derivation of an output tuple and this can be verified in A = 12 which can also be obtained by using just the record with id = 3 from C1.

Now I will discuss the provenance obtained in the Perm system for the global class G_1 (see figure 4.1) in the case of conflicting and non conflicting attribute.
For the first case (query with non conflicting attribute) the query to pose is the following:

```
 q2 = Select ID,A,B,prov_public_luno_id as luno,
prov_public_ldue_id as ldue from
(select Provenance ID, A, B from G1) Tmp;
```

and the result is the following:

```
 id | a  | b |luno | ldue
----+--- +---+-----+-----
 1  | 3  | 3 | 1   | 1
 2  |    |   | 2   | 2
 3  | 9  |   | 3   | 3
 4  | 8  |   | 4   |
 5  |    |   | 5   | 5
```

As we said in the previous subsection the *PI-CS* provenance is not able to represent *each derivation* and this can be verified in the previous result where

for the tuple with id = 3 the provenance information is $\{\{L_1^3\},\{L_2^3\}\}$ while the provenance with all possible derivation must be $\{\{L_1^3, L_2^3\},\{L_2^3\}\}$.

For the second case (query with aggregate functions), the query to pose is the following:

```
q3 = Select Provenance ID,C from G1;
```

where the aggregate function used is the AVG applied on the column C (see section 2.2). Also in this case to simplify the interpretation of the result and to see just provenance information we pose the following query:

```
q4 = Select ID,C,prov_public_luno_id as luno,
prov_public_ldue_id as ldue, prov_public_ltre_id as
ltre from (select Provenance ID, C from G1) Tmp;
```

The final result from the Perm system is the following:

```
 id | c  | luno |ldue | ltre
----+----+------+-----+-----
 1  | 24 | 1    | 1   |
 2  | 25 | 2    | 2   |
 3  | 20 | 3    | 3   |
 4  | 25 | 4    |     |
 5  | 25 | 5    | 5   | 5
```

Also in this case the *PI-CS* provenance is not able to represent *each derivation* and this can be verified in the previous result where for the tuple with id = 5 the provenance information is $\{\{L_1^5, L_2^5, L_3^5\}\}$ while the provenance with all possible derivation must be $\{\{L_1^5, L_2^5, L_3^5\},\{L_1^5, L_2^5\},\{L_3^5\}\}$.

To avoid this problem we need to extend the definition of the *PI-CS* provenance to represent all the possible derivations of a tuple in both cases, when resolution functions are used or not, and to apply this extension to the full outer join operator, thus we provide provenance information for the full outer join merge operator used in the MOMIS system and this will be discussed in the next section.

4.3 *Lineage* and *Why*-Provenance: a formal definition

Let G be a global class with schema $S(G)$ and let $\mathcal{L}(G) = \{L_1, \ldots, L_n\}$ be the set of its local classes, with schema $S(L_i)$, for each i. As in our previous examples, we assume ID as a share object identifier among all local classes, i.e. $ID \in \cap_i S(L_i)$ and we assume $S(G) = \cup_i S(L_i)$, i.e. global and local attribute names are the same (we will use A_i to denote global and local attributes).

Given a set of global attributes $S = \{A_1, \ldots, A_k\}$ we consider the query

$$q_S = \texttt{SELECT DISTINCT } A_1, \ldots, A_k \texttt{ FROM } G$$

i.e., we consider the set semantics.
Given a query q_S, we add to the select list the object identifier ID, i.e. we consider the query with attributes $S \cup \{ID\}$, denoted by q_S^*; we first define *lineage* and *why*-provenance for this query q_S^* and then we will give the definitions for a generic query q_S, i.e. for a query where the select list S doesn't necessarily contain the object identifier ID. *Lineage* and *why*-provenance for q_S^* will be respectively denoted by IDLIN and IDWHY;*lineage* and *why*-provenance for q_S will be respectively denoted by LIN and WHY.

The query q_S^* is rewritten w.r.t. local classes as follows:

```
SELECT
  COALESCE(L1.ID,L2.ID, L3.ID) AS ID,
  Li.A AS A -- if  A is mapped only in Li
  RF(L1.A, ... Lk.A) AS A -- if A is  mapped in L1, ... Lk
FROM L1 FOJ L2 ON (L2.ID=L1.ID)
     FOJ L3 ON (L3.ID=L1.ID OR L3.ID=L2.ID )
     ... FOJ Ln ON (Ln.ID=L1.ID OR ... OR Ln.ID=Ln-1.ID)
```

Let $u(id) \in q_S^*$ be the *unique* tuple of q_S^* with ID equal to id, i.e. $u(id)[ID] = id$; given $u(id) \in q_S^*$, we define the set $\mathbf{LC}(id) = \{L_i^{id} \mid \exists L_i \in \mathcal{L}(G), \exists L_i^{id} \in L_i\}$ of local tuples with ID equal to id; $\mathcal{P}(\mathbf{LC}(id))$ denotes the set of all subset of $\mathbf{LC}(id)$.

Given $u(id) \in q_S^*$, its lineage is defined by:

$$\text{IDLIN}(q_A^*, u(id)) = \mathbf{LC}(id)$$

Thus, lineage for $u(id) \in q_S^*$, is independent from the selected global attributes, as swhon in the examples of figures 4.2 and 4.3.
To define *why*-provenance, we first consider a single global attribute A; given $u(id) \in q_A^*$, $\text{IDWHY}(q_A^*, u(id))$ is a subset of $\mathcal{P}(\mathbf{LC}(id))$ defined as follows:

- if $A = ID$ is the object identifier: $\text{IDWHY}(q^*_A, u(id)) = \mathcal{P}(\mathbf{LC}(id))$.
- if A is one-to-one, mapped only on $L.A$:
  - if $L^{id}[A]$ is $NULL$
    then $\text{IDWHY}(q^*_A, u(id)) = \mathcal{P}(\mathbf{LC}(id))$
    else $\text{IDWHY}(q^*_A, u(id)) = \{S \in \mathcal{P}(\mathbf{LC}(id)) \mid L^{id} \in S\}$.
- if A is a one-to-many, mapped into $k \leq n$ local classes and defined by the resolution function RF:

  $\text{IDWHY}(q^*_A, u(id))$ = { { L^{id}_1,...,L^{id}_q } | $q \leq k$, $L^{id}_i \in L_i, 1 \leq i \leq q$ and $RF(L^{id}_1[A], \ldots, L^{id}_k[A]) = u(id)[A]$ }

The first rule is trivial. The second rule takes into account that, due to the full outer join operation, a $NULL$ value for $u(id)[A]$, can come either from a $NULL$ value from the local class where A is mapped (i.e. $L^{id}[A]$) or it can be obtained from any local class where A is not mapped. The last rule takes into account a one-to-many global attribute A; in this case the value for A is computed by means of a resolution function: $FJ^{id}[A] = RF(v_1, v_2, \ldots, v_k)$, with $v_i = L^{id}_i[A]$, $1 \leq i \leq k$. The rules generate a witness { L^{id}_1,...,L^{id}_q } for each subset $v_1, v_2, \ldots, v_q$, $q \leq k$, such that $RF(v_1, v_2, \ldots, v_q) = RF(v_1, v_2, \ldots, v_k)$.

The function $\text{IDWHY}(q^*_S, u(id))$ is extended to a subset $S = \{A_1, \ldots, A_k\}$ with $k \geq 1$, as follows:

$$\text{IDWHY}(q^*_S, u(id)) = \bigcap \Big\{ \text{IDWHY}(q^*_S, u(id)) \mid A \in S \Big\}$$

As an example, for the query in Figure 4.2 we have $S = \{ID, A, B\}$; $\text{IDWHY}(q^*_S, u(3))$ is computed as:

1. $\text{IDWHY}(q^*_{ID}, u(3))$ = $\mathcal{P}(\{L^3_1, L^3_2\}) \setminus \emptyset$
2. $\text{IDWHY}(q^*_A, u(3))$ = $\{\{L^3_1\}, \{L^3_1, L^3_2\}\}$
3. $\text{IDWHY}(q^*_B, u(3))$ = $\{\{L^3_1\}, \{L^3_2\}, \{L^3_1, L^3_2\}\}$

Then $\text{IDWHY}(q^*_S, u(3)) = \{\{L^3_1\}, \{L^3_1, L^3_2\}\}$.

Finally, we give the definitions for a generic query q_S, i.e. for a query where the select list S doesn't necessarily contain the object identifier ID. The *Lineage* and the *why*-provenance of $t \in q_S$ are, respectively, defined as follows:

$$\text{LIN}(q_S, t) = \bigcup \Big\{ \text{IDLIN}(q^*_S, u(id)) \mid u(id)[S] = t \Big\}$$

$$\text{WHY}(q_S, t) = \bigcup \Big\{ \text{IDWHY}(q^*_S, u(id)) \mid u(id)[S] = t \Big\}$$

4.4 Discussion

In this chapter we presented our work to apply data provenance techniques in the Data Fusion framework of the MOMIS Integration System. We focused our attention on the extension of the concept of *why*-provenance to deal with Resolution Functions. First we discussed different types of provenance: *Lineage*, *why*- provenance and *minimal why*-provenance in the context of our data fusion system MOMIS and then we presented the *PI-CS* provenance that deal with the provenance for the full outer join operator and we presented the Perm system that implement this type of provenance. Finally, through an example in Perm, we showed the limitation of the *PI-CS* provenance and in the last section we presented a formal definition for provenance that we want to obtain in our system.

Chapter 5

Provenance of Cereal Genotypic and Phenotypic Data

This chapter presents the ongoing research on the design and development of a Provenance Management component, PM_{MOMIS}, for the MOMIS Data Integration System. PM_{MOMIS} aims to provide the provenance management techniques supported by two of the most relevant data provenance systems, the *Perm* and *Trio* systems, and extends them by including the data fusion and conflict resolution techniques provided by MOMIS.
PM_{MOMIS} functionalities have been studied and partially developed in the domain of genotypic and phenotypic cereal-data management within the CEREALAB project. Users of CEREALAB played a major role in the emergence of real needs of provenance management in their domain.
The remainder of the chapter is organized as follows. In section 5.1 we will present and motivate the problem of data provenance in the CEREALAB database; In section 5.2 we will present the result of using the Perm system as the SQL engine of the query manager of MOMIS to compute provenance in the CEREALAB database, then in section 5.3 we present a problem encountered during the study of provenance for biological data and then present a possible solution for this problem by using the Trio system in section 5.4. Finally, conclusions and future works are sketched in section 5.5.
The work presented in this chapter was published in [Beneventano et al., 2012b, Beneventano et al., 2012c],

5.1 Problem Definition

As described in 3.1 the CEREALAB Data Integration Application has been developed to create a powerful tool for plant breeders and geneticists. It stores genotypic and phenotypic cereal-data collected within the CEREALAB project and integrates them with already existing well known public data sources. The integrated database can help breeders and geneticists in: unravelling the genetics of economically important phenotypic traits, identifying and choosing molecular markers associated to key traits, and in choosing the desired parentals for breeding programs. The CEREALAB Data Integration Application development was one of the objectives of the CEREALAB and SITEIA projects and of the BIOGEST-SITEIA laboratory (www.biogest-siteia.unimore.it), funded by Emilia-Romagna (Italy) regional government, and aims to increase the competitiveness of Regional seed companies through the use of modern selection technologies, i.e. the Marker-Assisted Selection (MAS). Data integration is obtained by using the MOMIS system (Mediator envirOnment for Multiple Information Sources), a framework to perform integration of structured and semi-structured data sources [Beneventano et al., 2003a, Beneventano et al., 2010]. MOMIS is characterized by a classical wrapper/mediator architecture: the local data sources

contain the real data, while a Global Schema (GS) provides a *reconciled, integrated, read-only view* of the underlying sources. The GS and the mappings between the GS and the local sources are semi-automatically defined at design time by the Integration Designer component of the system [Beneventano et al., 2003a]. After GS creation end-users can pose queries over this GS in a transparent way w.r.t. the local sources.
As discussed in 2.2 the global classes of the *CEREALAB GS* plays the role of performing *data fusion*. Data fusion may involve the resolution of possible conflicts between data coming from different sources; several high level strategies to handle inconsistent data have been described and classified in [Bleiholder and Naumann, 2008]. MOMIS supports: *conflict avoiding* strategies (such as the *trust your friends* strategy which takes the value of a preferred source), *conflict ignoring* strategies (such as the *pass it on* strategy, which presents all values deferring conflict resolution to the user) and *resolution* strategies (such as the *meet in the middle* strategy which takes an average value). These strategies are implemented by means of *Resolution functions* in the *full outerjoin-merge operator* proposed in [Naumann et al., 2004] and adapted to the MOMIS System in [Beneventano et al., 2010].
A requirement emerging from CEREALAB users, the breeders, was that in many cases they would prefer to give a look at the data coming from the local sources, i.e. they need *provenance*. A need to support detailed data provenance is one of the main requirements of biological data management identified in [Jagadish and Olken, 2004][1].
Lineage, or *provenance*, in its most general definition, describes where data came from, how it was derived and how it was modified over time. Lineage provides valuable information that can be exploited for many purposes, ranging from simple statistical resumes presented to the end-user to more complex applications, such as, managing data uncertainty or identifying and correcting data errors. Lineage has been studied extensively in data warehouse systems [Cui et al., 2000], but it is still an open research problem in Data Integration systems [Halevy et al., 2006b, Halevy and Li, 2003].

5.2 Provenance in the CEREALAB database

In this section we present the provenance computation in the cerealab database. First we present the use of the PERM system in the MOMIS system to compute provenance then we present the provenance computation obtained in the MOMIS

[1]This article is an extract from the Report of the NSF Workshop on Data Management for Molecular and Cell Biology, edited by H. V. Jagadish and Frank Olken he workshop was held at the National Library of Medicine, Bethesda, MD, Feb. 2-3,2003)

system using the PERM system. For better reading, in this section we report the example illustrated in section 3.4.1 in Fig. 5.1 and Fig. 5.2.

Local classes

A (GermplasmA)

GPN	yield	FHB	type
Eureka	18	MR	
Fortuna	7	MR	
Mentana		S	line
Kenora	20	MR	landrace
Oasis	21	MR	cultivar

B (GermplasmB)

GPN	yield	FHB	type
Eureka	6	S	cultivar
Fortuna	15	S	landrace
Mentana	20	MR	line
Kenora			cultivar

Mapping Table of the global class GERMPLASM

GERMPLASM	A	B
GPN	GPN	GPN
Yield	yield	yield
FHB	FHB	FHB
Type	type	type

Figure 5.1: Example: two local classes with two conflicting attributes

Mapping Query of GERMPLASM

```
SELECT  GPN = GPN,
        Yield=AVG(A.yield, B.yield)
        FHB = COALESCE(A.FHB,B.FHB),
        Type = ALLVALUES(A.type, B.type)
FROM    A FULL OUTER JOIN B
        USING (GPN)
```

Instance of GERMPLASM

GPN	Yield	FHB	Type
Eureka	12	MR	cultivar
Fortuna	11	MR	landrace
Mentana	20	S	line
Kenora	20	MR	landrace,cultivar
Oasis	21	MR	cultivar

Figure 5.2: Mapping Query and Instance of the global class GERMPLASM

5.2.1 PM_{MOMIS} (Provenance Management component for the MOMIS system)

In chapter 4 we introduced the notion of provenance into MOMIS, by defining the provenance for the *full outerjoin-merge operator*; this definition is based on the concept of *PI-CS*-provenance (Perm Influence Contribution Semantics) proposed in *Perm* (Provenance Extension of the Relational Model) [Glavic and Alonso, 2009] to produce more precise provenance information for outerjoins. Another important reason behind the choice of using the *PI-CS*-provenance, was that it is fully implemented in an open-source provenance management system that is capable of computing, storing and querying provenance for relational databases. At present, we are using the *Perm* system as the SQL engine of MOMIS, so that to obtain the provenance in our CEREALAB Application Fig. 5.3

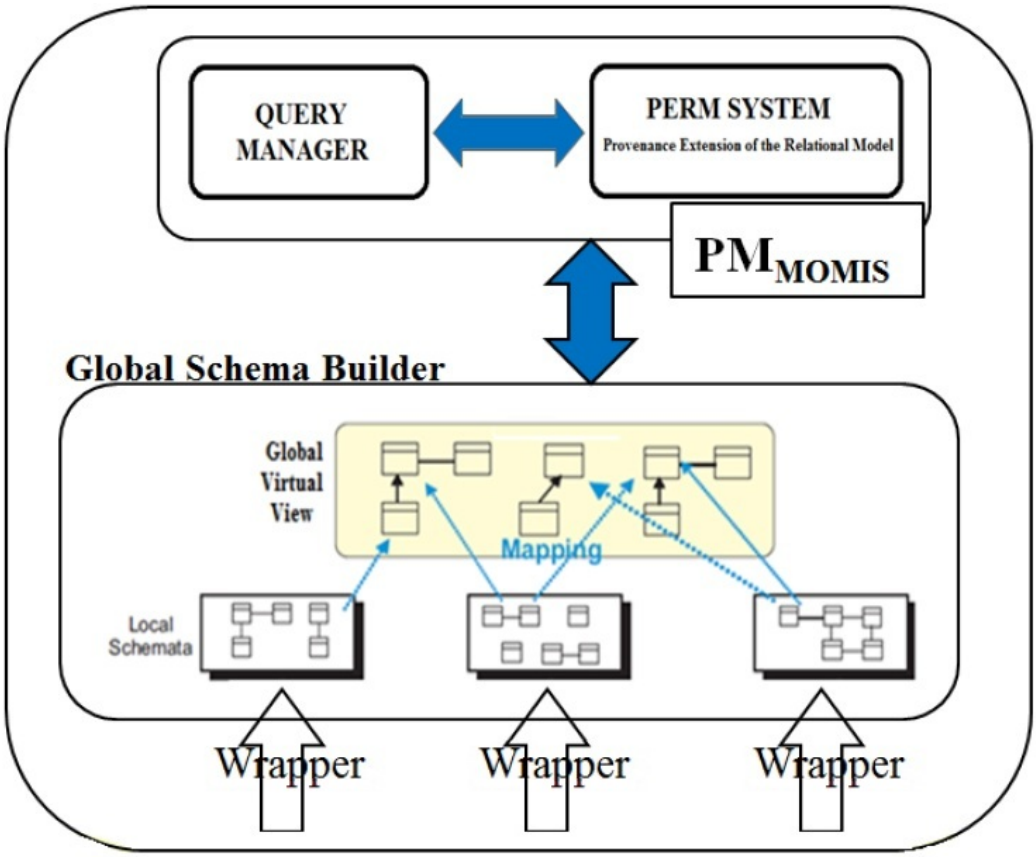


Figure 5.3: PM_{MOMIS}

5.2.2 An example of Provenance Computation in CEREALAB

We remember that GERMPLASM (See 5.2) is a global class derived from the fusion of two local classes, GermplasmA and GermplasmB, denoted respectively by A and B. We consider the following query:

```
Select Provenance GPN, Yield from GERMPLASM;
```

The result is very long and is difficult to interpret (as we said before the provenance information in the PERM system is represented in a relational form so the record A(Eureka) is represented by all its attributes, and to these attribute the keyword "prov_public" is added, then the result contains the attributes prov_public_germplasma_gpn, prov_public_germplasma_yield ... etc.) so to facilitate the result visualization and interpretation we show just the tuple identifier to see the origin of the tuples that contibute in the result by posing the following query:

```
 Select GPN, Yield, prov_public_germplasma_gpn,
prov_public_germplasmb_gpn from (Select
Provenance GPN, Yield from GERMPLASM) tmp;
```

To facilitate the result visualization, we also replaced Prov_public_germplasma by A. and Prov_public_germplasmb by B. (we will use this notation in the rest of this chapter) so the result is the following:

```
  GPN     | yield  | A.GPN    | B.GPN
----------+--------+----------+----------
 Eureka   | 12     | Eureka   | Eureka
 Fortuna  | 11     | Fortuna  | Fortuna
 Kenora   | 20     | Kenora   | Kenora
 Menatna  | 20     | Mentana  | Mentana
 Oasis    | 21     | Oasis    |
```

If we consider the first tuple, *GPN = Kenora*, the provenance presented as a set of witness lists is $\{ \langle \mathtt{A}(Kenora), \mathtt{B}(Kenora) \rangle \}$. The obtained result is not the desired result because the output tuple can be obtained also using just the tuple with *GPN= Kenora* from the relation *GermplasmA*, so the set of witness list must be $\{ \langle \mathtt{A}(Kenora), \mathtt{B}(Kenora) \rangle, \langle \mathtt{A}(Kenora), \perp \rangle \}$.
To resolve this problem and to obtain the *why - provenance* presented in section 4.2.1, we need to implement the definition of provenance presented in section 4.3 and this will be a future work.

5.3 Problem of provenance at present implemented in the MOMIS system

In chapter 4 the concept of *PI-CS*-Provenance was studied and extended to the full outerjoin-merge operator. As an example, let us consider a query on GERMPLASM (the user is searching for the types of the varieties that are resistant to the fusarium head blight):

```
TYPE_MR = SELECT  DISTINCT Type
            FROM  GERMPLASM
            WHERE FHB = 'MR'
```

The *PI-CS*-Provenance of an output tuple is a set of *witness lists*, where each witness list represents one combination of local tuples that were used together to derive the output tuple; a witness list contains a local tuple from each local class or the special value $\perp$, indicating that no tuple from a local class was used to derive the output tuple (useful in modelling outerjoins). For example, the *PI-CS*-Provenance of the output tuple cultivar (see Figure 5.4) is a set of two *witness lists*, where the second one $\langle \mathtt{A}(Oasis), \perp \rangle$ indicates that $\mathtt{A}(Oasis)$ paired with no tuples of B is a possible derivation of the output tuple.

Type	*PI-CS* Provenance as a set of witness lists
landrace	$\{ \langle \mathtt{A}(Fortuna), \mathtt{B}(Fortuna) \rangle \}$
cultivar	$\{ \langle \mathtt{A}(Eureka), \mathtt{B}(Eureka) \rangle, \langle \mathtt{A}(Oasis), \perp \rangle \}$
landrace,cultivar	$\{ \langle \mathtt{A}(Kenora), \mathtt{B}(Kenora) \rangle \}$

Relational Representation of the *PI-CS* Provenance

Type	A.GPN	A.yield	A.FHB	A.type	B.GPN	A.yield	B.FHB	B.type
landrace	Fortuna	7	MR		Fortuna	15	S	landrace
cultivar	Eureka	18	MR		Eureka	6	S	cultivar
cultivar	Oasis	21	MR	cultivar				
landrace,cultivar	Kenora	20	MR	landrace	Kenora			cultivar

Figure 5.4: Example: *PI-CS* Provenance for the query TYPE_MR

Witness lists are represented in a relational form, as shown in Figure 5.4: each witness list of an output tuple is represented by a single tuple.
The main drawback of this solution is that often conflicting values represent *alternative*. For example if we want to select other germplasms of the same type of *Kenora*, the two values landrace and cultivar must be considered as *alternative values*; if not, we might obtain Fortuna and Eureka as germplasms of the

same type. Our proposal to overcome this drawback is to consider the output of the full outer join merge operator as an *uncertain relation* and then manage it with a system that supports uncertain data and data lineage, the *Trio* system [Agrawal et al., 2006, Benjelloun et al., 2006].

5.4 Provenance based Conflict Handling Strategies

The *Trio* system is based on the *ULDB* data model [Benjelloun et al., 2006], which extends the relational model with:

- *Alternatives*, representing uncertainty about the contents of a tuple. ULDB uncertain relations have a set of *certain* attributes and a set of *uncertain* attributes; each tuple in a ULDB relation has one value for each certain attribute, and a set of possible values for the uncertain attributes.
- *maybe* ('?') annotations, representing uncertainty about the presence of a tuple.
- *Lineage*, connecting tuple-alternatives to other tuple-alternatives from which they were derived: *Trio*-Provenance is a boolean formula λ over tuple-alternatives.
- *Confidences*: numerical confidence values optionally attached to alternatives.

(A) Global class GERMPLASM

GPN	Yield	FHB	Type	
Eureka	12	MR	cultivar	?
Fortuna	11	MR	landrace	?
Mentana	20	S	line	?
Kenora	20	MR	landrace || cultivar	?
Oasis	21	MR	cultivar	

(B) query TYPE_MR

Type	
landrace	?
cultivar	
landrace || cultivar	?

Figure 5.5: Global class GERMPLASM and query TYPE_MR as *uncertain relations*

Intuitively, these concepts might be applied to our data fusion context as follows. As shown in Figure 5.5.A, the global class GERMPLASM is considered as an *uncertain relation* where *non-conflicting* attributes and *solved-conflicting* attributes (like FHB and Yield) are modelled as *certain* attributes and *not-solved-conflicting* attributes (like Type) are modelled as *uncertain* attributes. The global tuple identified by GPN $= Kenora$, denoted by $Kenora$, has two

alternatives, $(Kenora, 1)$ and $(Kenora, 2)$. A global tuple coming from local tuples with conflicting values (solved or not solved) is annotated with '?' ; in the example, all global tuples are *maybe* tuples '?', with the exception for $Oasis$. The *Trio*-Provenance of a global tuple-alternative is a boolean formula over the local tuples from which the alternatives were derived; for example: $\lambda(Kenora, 1) = \texttt{A}(Kenora)$: this alternative derives from a local tuple in A; $\lambda(Kenora, 2) = \texttt{A}(Kenora) \wedge \texttt{B}(Kenora)$: this alternative derives from the conjunction of two local tuples. At present *Confidences* are not considered in our framework.

How can we implement the three above TRIO concepts in PM_{MOMIS}? Can we obtain the GERMPLASM *uncertain relation* shown in the example by means of the *Trio* system?

GERMPLASM computation is based on outerjoins and uses resolution functions, thus such computation is not simple as in *Trio* outerjoins are not allowed[2] and resolution functions have to be implemented in *Trio* (as we already did with a strong effort in *Perm*). For this reason, our choice is to use PERM for computing global classes and Trio for querying global classes. The computation of the full outer join merge operator is extended to obtain global classes including conflicts as *uncertain relations* with their related *TRIO*-Provenance; this computation is discussed in [Benjelloun et al., 2006] and its result is intuitively shown in Figure 5.5.A. Thus, the *Trio* system[3] may be used to execute queries on global classes (i.e. exploiting the uncertain database theory); intuitively, the uncertain relation resulting from query TYPE_MR_S is shown in Figure 5.5.B. In this way, the user knows that only cultivar (the second tuple) is coming from non-conflicting local tuples, since it is not a *maybe* tuples; then he may obtain the provenance of this tuple either[4] at the level of global classes, $(Eureka, 1) \vee (Oasis, 1)$, and at the level of local classes, $(\texttt{A}(Eureka) \wedge \texttt{B}(Eureka)) \vee \texttt{A}(Oasis)$.

A relevant application and extension of the *Trio* system to query conflicting data, is to use the possible instances of a query to provide the user with different *search strategies* for querying the Global Schema. As an example (the user is searching for the types of the varieties with a yield value greater than 11):

[2] In a *TRIO* database U, *base tables* are uncertain relations and the result of a relational query Q on U is an uncertain relation: in [Ikeda and Widom, 2009] the problem of incorporating outerjoins into uncertain databases is considered but the authors argued that standard possible-worlds semantics may be inappropriate for outerjoins.

[3] The *Trio* source code is freely available and the system is based also on PostgreSQL.

[4] *TRIO*-Provenance is a multilevel (transitive) relationships: formulas specify direct derivations, but when the alternatives in a formula are themselves derived from other alternatives, it is possible to recursively expand a formula until it specifies local tuples only.

```
HIGH_PROD_TYPE =
   SELECT DISTINCT Type
   FROM GERMPLASM
   WHERE Yield > 11
```

HIGH_PROD_TYPE
as an *uncertain* relation

Type	
line	?
cultivar	
landrace || cultivar	?

The user, by interacting with the framework, might obtain: At first, only *consistent global tuples* coming from non-conflicting local tuples are viewed : { cultivar }. Then, after the application of conflict handling strategies (`yield` solved with `AVG`, `Type` considered as an uncertain attribute) the uncertain relation `HIGH_PROD_TYPE`, provides two different answers: (1) tuples in *every* possible instance: { cultivar,line }; (2) tuples in *some* possible instance: { cultivar,line,landrace}.

5.5 Discussion

In this chapter we presented the ongoing research on the design and development of a Provenance Management component, PM_{MOMIS}, for the MOMIS System. PM_{MOMIS} functionalities have been studied and partially implemented in the domain of genotypic and phenotypic cereal-data management within the CEREALAB project.

Several notions of provenance for database queries have been proposed and studied in the past years, see [Cheney et al., 2009] for a survey. Among these approaches, one of the most expressive ones is the *Provenance Semiring* [Green et al., 2007]; an extension of the *Provenance Semiring* to schema mappings is used in ORCHESTRA data sharing system [Ives et al., 2008]; as in our framework, provenance in ORCHESTRA is also used to perform reconciliation based on user preferences on the sources the data come from; moreover, the ORCHESTRA system is being prototyped in applications with biological collaborators [Ives, 2009].*Provenance Semirings* and ORCHESTRA are then important references for our future work.

Another important references for our future work is the Open Provenance Model (OPM) [Moreau et al., 2008], which aims to define a generic and comprehensive representation of data provenance and which is becoming a standard to share provenance information. OPM represents provenance through a graph; a graphical representation of provenance graph may be used to formulate queries from a visual and intuitive interface, and then enabling end-users (plant scientists) to be able to directly query the provenance information.

An interesting idea of provenance application in the biology community is

sketched in [Jagadish and Olken, 2004]: mechanisms similar to the bibliographic citation index for articles and authors are needed to acknowledge ”publication” of datasets in shared databases, so as to encourage rapid, effective sharing of data; data management support for tracking data provenance can provide the analog of citations. Following this idea we are ”ranking” the local sources integrated in the CEREALAB application on the basis of users’ queries.

Chapter 6

Conclusions

This thesis focuses on data integration and data provenance in the context of the MOMIS data integration system that was used to create the CEREALAB database. Its main contribution is the creation of the CEREALAB database V2.0 with new functionalities derived from the needs of the end users and the study of different data provenance models to finally create a new component for the MOMIS system in order to offer data provenance support for the CEREALAB users.
In chapter 3 I described the new version of the CEREALAB database obtained from a real application of the MOMIS data integration system with all the new extensions made to solve an actual problem for plant breeders to find the data of interest.
In chapter 4 I presented the problem of data provenance in the context of data integration systems and I presented three different models of data provenance *Lineage, why-provenance and PI-CS*. Then I compared their results and I argued the difference between these results. Also in chapter 4 I presented the Perm system that implement the *PI-CS provenance*, how it works and how presents the final result and I showed the provenance result in the case of non conflicting attribute and in aggregate functions are used to ague why this system don't provide a fully solution for our problem. Finally, I presented a formal definition of the provenance model we want to obtain in our framework; In particular I focused my attention on the extension of the concept of *why*-provenance to deal with Resolution Functions.
In chapter 5 I presented the realization of the PM_{MOMIS} component, obtained by the use of the Perm system as the SQL engine of the query manager of MOMIS, and I showed the result of using this component in the CEREALAb database. Then I described an encountered problem during the use of this component in the context of biological data derived from the particularity of this data and I finally presented a possible solution for it by using the *Trio* system. The PM_{MOMIS} is yet underdevelopment and the next step will be the application of the extension studied and formally presented in chapter 4.
Some future directions of this Phd research are the following:

- *Querying Data Lineage*.
  In a data fusion scenario, the data lineage can be useful to understand the relation between the results we obtain querying a global class G and the local classes G is mapped on. This is particularly important for example to evaluate how the data we obtain from a data integration system can be affected when one or more local sources become unavailable.

  It is thus necessary to allow querying data lineage, providing an appropriate method to express conditions in our queries to consider tuples with lineage from certain local classes.

For example, the results from query q_1 (figure 4.2) show no differences between a `NULL` value coming from the data sources (first tuple) and a `NULL` value obtained because the attribute has no mapping on a local source (second tuple). Having the possibility to query the data lineage, we can have different results in these two cases.

- *Where*-Provenance.
  Our preliminary work on data lineage started with analyzing *lineage* and *why*-Provenance; the next step will be analyzing also the *Where*-Provenance, with particular regards to resolution functions. The starting point will be the observation in [Naumann and Bleiholder, 2006]: mediating resolution functions does not allow evaluating the *Where*-Provenance, while it is possible to assign it with deciding resolution functions.

- *Other Provenance Models*.
  Beyond *Lineage* and *Why*-provenance, several other concepts of provenance (or provenance models) were proposed in literature, and, among these, the most informative form of provenance is the semiring of provenance polynomials [Green et al., 2007]. On the other hands, these provenance models are defined for unions of conjunctive queries, and then need to be extended to capture provenance for queries based on the full outer join merge operator, which involves a form of negation.

- *Complexity Analysis*.
  In the comparison between *lineage* and *why*-provenance, also the computational costs must be considered: *why*-provenance provides more information than *lineage*, but, intuitively, its computational cost is higher. It is thus important to discuss this precision/cost tradeoff.

- *Implementation*.
  In the MOMIS system, to answer a query over a global class G, the query must be rewritten as an equivalent set of queries expressed on the local schemas (local queries); this query translation performs some query optimization techniques, such as predicate push down (to push a constraint on local queries) and full join simplification (to reduce full join to left/right/inner join). Thus, our idea is to extend our query processing to include also the provenance computation in the query rewriting, in order to be able to provide the user with lineage information when obtaining the results.

Appendix A

The ODL$_{I^3}$ language syntax

The following is the BNF description of the ODL$_{I^3}$ description language. This object-oriented language, with an underlying Description Logic, is introduced for information extraction. The ODL$_{I^3}$ language is presented in [Bergamaschi et al., 2001], in the following I include the syntax fragments which differ from the original ODL grammar, referring to it for the remainder.

⟨interface_dcl⟩ ::= ⟨interface_header⟩
{[⟨ interface_body⟩]};
[**union** ⟨identifier⟩ { ⟨interface_body⟩ }**;**]

⟨interface_header⟩ ::= **interface** ⟨identifier⟩
[⟨inheritance_spec⟩]
[⟨type_property_list⟩]

⟨inheritance_spec⟩ ::= : ⟨scoped_name⟩
[,⟨inheritance_spec⟩]

Local schema pattern definition: the wrapper must indicate the kind and the name of the source of each pattern.

⟨type_property_list⟩ ::= **(** [⟨source_spec⟩]
[⟨extent_spec⟩]
[⟨key_spec⟩] [⟨f_key_spec⟩] [⟨c_key_spec⟩] **)**

⟨source_spec⟩ ::= **source** ⟨source_type⟩
⟨source_name⟩

⟨source_type⟩ ::= **relational** | **nfrelational**
| **object** | **file**
| **semistructured** | **multimedia**
| **glso**

⟨source_name⟩ ::= ⟨identifier⟩

⟨extent_spec⟩ ::= **extent** ⟨extent_list⟩

⟨extent_list⟩ ::= ⟨string⟩ | ⟨string⟩,⟨extent_list⟩

⟨key_spec⟩ ::= **key**[**s**] ⟨key_list⟩

⟨f_key_spec⟩ ::= **foreign_key** (⟨f_key_list⟩)
references ⟨key_list
⟩ [⟨f_key_spec⟩]

⟨c_key_spec⟩ ::= **candidate_key** ⟨identifier⟩
(⟨key_list⟩**)**

Global pattern definition rule, used to map the attributes between the global definition and the corrisponding ones in the local sources.

⟨attr_dcl⟩ ::= **[readonly] attribute**
[⟨domain_type⟩]
⟨attribute_name⟩ **[*]**
[⟨fixed_array_size⟩]
[⟨mapping_rule_dcl⟩]
⟨mapping_rule_dcl⟩ ::= **mapping_rule** ⟨rule_list⟩
⟨rule_list⟩ ::= ⟨rule⟩ | ⟨rule⟩**,**⟨rule_list⟩
⟨rule⟩ ::= ⟨local_attr_name⟩ |
'⟨identifier⟩**'**
⟨and_expression⟩ |
⟨union_expression⟩
⟨and_expression⟩ ::= **(** ⟨local_attr_name⟩ **and**
⟨and_list⟩ **)**
⟨and_list⟩ ::= ⟨local_attr_name⟩
| ⟨local_attr_name⟩ **and**
⟨and_list⟩
⟨union_expression⟩ ::= **(** ⟨local_attr_name⟩ **union**
⟨union_list⟩ **on** ⟨identifier⟩ **)**
⟨union_list⟩ ::= ⟨local_attr_name⟩
| ⟨local_attr_name⟩ **union**
⟨union_list⟩
⟨local_attr_name⟩ ::= ⟨source_name⟩**.**⟨class_name⟩**.**
⟨attribute_name⟩

...

Terminological relationships used to define the Common Thesaurus.

⟨relationships_list⟩ ::= ⟨relationship_dcl⟩**;** |
⟨relationship_dcl⟩**;**
⟨relationships_list⟩
⟨relationships_dcl⟩ ::= ⟨local_name⟩
⟨relationship_type⟩
⟨local_name⟩
⟨local_name⟩ ::= ⟨source_name⟩**.**
⟨local_class_name⟩
[**.**⟨local_attr_name⟩]
⟨relationship_type⟩ ::= **SYN** | **BT** | **NT** | **RT**

...

Extended base type definition for multimedia objects.

⟨BaseTypeSpec⟩ ::= ⟨FloatingPtType⟩ |
⟨IntegerType⟩ |
⟨CharType⟩ |
⟨BooleanType⟩ |
⟨OctetType⟩ |
⟨RangeType⟩ |
⟨AnyType⟩ |
⟨MultiMediaType⟩

⟨MultiMediaType⟩ ::= **Text** |
Image

OLCD integrity constraint definition: declaration of rule (using *if then* definition) valid for each instance of the data; mapping rule specification (*or* and *union* specification rule).

⟨rule_list⟩ ::= ⟨rule_dcl⟩**;** | ⟨rule_dcl⟩**;** ⟨rule_list⟩
⟨rule_dcl⟩ ::= **rule** ⟨identifier⟩ ⟨rule_spec⟩
⟨rule_spec⟩ ::= ⟨rule_pre⟩ **then** ⟨rule_post⟩ |
{ ⟨case_dcl⟩ }
⟨rule_pre⟩ ::= ⟨forall⟩ ⟨identifier⟩ **in** ⟨identifier⟩ **:**
⟨rule_body_list⟩
⟨rule_post⟩ ::= ⟨rule_body_list⟩
⟨case_dcl⟩ ::= **case of** ⟨identifier⟩ **:** ⟨case_list⟩
⟨case_list⟩ ::= ⟨case_spec⟩ | ⟨case_spec⟩ ⟨case_list⟩
⟨case_spec⟩ ::= ⟨identifier⟩ **:** ⟨identifier⟩ **;**

⟨rule_body_list⟩ ::= **(** ⟨rule_body_list⟩ **)** |
⟨rule_body⟩ |
⟨rule_body_list⟩ **and**
⟨rule_body⟩ |
⟨rule_body_list⟩ **and**
(⟨rule_body_list⟩ **)**

⟨rule_body⟩ ::= ⟨dotted_name⟩
⟨rule_const_op⟩
⟨literal_value⟩ |
⟨dotted_name⟩
⟨rule_const_op⟩
⟨rule_cast⟩ ⟨literal_value⟩ |
⟨dotted_name⟩ **in**
⟨dotted_name⟩ |
⟨forall⟩ ⟨identifier⟩ **in**
⟨dotted_name⟩ **:**
⟨rule_body_list⟩ |
exists ⟨identifier⟩ **in**
⟨dotted_name⟩ **:**
⟨rule_body_list⟩

⟨rule_const_op⟩ ::= = | ≥ | ≤ | > | <

⟨rule_cast⟩ ::= **(**⟨simple_type_spec⟩**)**

⟨dotted_name⟩ ::= ⟨identifier⟩ | ⟨identifier⟩**.**
⟨dotted_name⟩

⟨forall⟩ ::= **for all** | **forall**

Bibliography

[DBL, 2009] (2009). Data integration. In Liu, L. and Özsu, M. T., editors, *Encyclopedia of Database Systems*, page 585. Springer US.

[Agrawal et al., 2006] Agrawal, P., Benjelloun, O., Sarma, A. D., Hayworth, C., Nabar, S., Sugihara, T., and Widom, J. (2006). Trio: a system for data, uncertainty, and lineage. In *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*, pages 1151–1154. VLDB Endowment.

[Antofie et al., 2007] Antofie, A., Lateur, M., Oger, R., Patocchi, A., Durel, C. E., and Van de Weg, W. E. (2007). A new versatile database created for geneticists and breeders to link molecular and phenotypic data in perennial crops. *Bioinformatics*, 23(7):882–891.

[Bartlett and Toms, 2005] Bartlett, J. C. and Toms, E. G. (2005). Developing a protocol for bioinformatics analysis: An integrated information behavior and task analysis approach: Research articles. *J. Am. Soc. Inf. Sci. Technol.*, 56(5):469–482.

[Beneventano et al., 2012a] Beneventano, D., Bergamaschi, S., Dannaoui, A., Milc, J., Pecchioni, N., and Sorrentino, S. (2012a). The cerealab database: Ongoing research and future challenges. In Dodero, J., Palomo-Duarte, M., and Karampiperis, P., editors, *Metadata and Semantics Research*, Communications in Computer and Information Science, pages 336–341. Springer Berlin Heidelberg.

[Beneventano et al., 2012b] Beneventano, D., Bergamaschi, S., and Dannaoui, A. R. (2012b). Integration and provenance of cereals genotypic and phenotypic data. In *SEBD*, pages 91–98.

[Beneventano et al., 2012c] Beneventano, D., Bergamaschi, S., and Dannaoui, A. R. (2012c). Integration and provenance of cereals genotypic and phenotypic data. In *DILS (Data Integration in the Life Sciences)*.

[Beneventano et al., 2010] Beneventano, D., Bergamaschi, S., Guerra, F., and Orsini, M. (2010). Data integration. In Embley, D. and Thalheim, B., editors, *Handbook of conceptual modelling*. Springer-Verlag. To appear. Available at http://dbgroup.unimo.it/SSE/SSE.pdf.

[Beneventano et al., 2003a] Beneventano, D., Bergamaschi, S., Guerra, F., and Vincini, M. (2003a). Synthesizing an integrated ontology. *IEEE Internet Computing*, pages 42–51.

[Beneventano et al., 2003b] Beneventano, D., Bergamaschi, S., and Sartori, C. (2003b). Description logics for semantic query optimization in object-oriented database systems. *ACM Trans. Database Syst.*, 28:1–50.

[Beneventano et al., 2007] Beneventano, D., Bergamaschi, S., Vincini, M., Orsini, M., and Nana Mbinkeu, R. C. (2007). Getting through the thalia benchmark with momis. In *Proceedings of the Third International Workshop on Database Interoperability (InterDB 2007) held in conjunction with the 33rd International Conference on Very Large Data Bases, VLDB 2007, Vienna, Austria, September 24*.

[Beneventano et al., 2011a] Beneventano, D., Dannaoui, A. R., and Sala, A. (2011a). On provenance of data fusion queries. In *SEBD*, pages 84–94.

[Beneventano et al., 2011b] Beneventano, D., Dannoui, A., and Sala, A. (2011b). Data lineage in the momis data fusion system. In *Data Engineering Workshops (ICDEW), 2011 IEEE 27th International Conference on*, pages 53 –58.

[Beneventano et al., 2008] Beneventano, D., Guerra, F., Orsini, M., Po, L., Sala, A., Gioia, M. D., Comerio, M., de Paoli, F., Maurino, A., Palmonari, M., Gennaro, C., Sebastiani, F., Turati, A., Cerizza, D., Celino, I., and Corcoglioniti, F. (2008). Detailed design for building semantic peer. *Networked Peers for Business, Deliverable D.2.1, Final Version, available at* $http://www.dbgroup.unimo.it/publication/d2\_1.pdf$, pages 52–57.

[Beneventano and Lenzerini, 2005] Beneventano, D. and Lenzerini, M. (2005). Final release of the system prototype for query management. sewasie, deliverable d.3.5, final version. http://www.dbgroup.unimo.it/prototipo/paper/D3.5Final.pdf.

[Benjelloun et al., 2006] Benjelloun, O., Sarma, A. D., Halevy, A., and Widom, J. (2006). Uldbs: databases with uncertainty and lineage. In *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*, pages 953–964. VLDB Endowment.

[Bergamaschi et al., 2011] Bergamaschi, S., Beneventano, D., Guerra, F., and Orsini, M. (2011). Data integration. In Embley, D. W. and Thalheim, B., editors, *Handbook of Conceptual Modeling*, pages 441–476. Springer Berlin Heidelberg.

[Bergamaschi et al., 1999] Bergamaschi, S., Castano, S., and Vincini, M. (1999). Semantic integration of semistructured and structured data sources. *Special Interest Group on Management of Data, SIGMOD Record*, 28(1):54–59.

[Bergamaschi et al., 2001] Bergamaschi, S., Castano, S., Vincini, M., and Beneventano, D. (2001). Semantic integration of heterogeneous information sources. *Data Knowl. Eng.*, 36(3):215–249.

[Bergamaschi et al., 2007] Bergamaschi, S., Po, L., and Sorrentino, S. (2007). Automatic annotation in data integration systems. In Meersman, R., Tari, Z., and Herrero, P., editors, *OTM Workshops (1)*, volume 4805 of *Lecture Notes in Computer Science*, pages 27–28. Springer.

[Bergamaschi et al., 2008] Bergamaschi, S., Po, L., and Sorrentino, S. (2008). Automatic annotation for mapping discovery in data integration systems. In Gaglio, S., Infantino, I., and Saccà, D., editors, *SEBD*, pages 334–341.

[Bergamaschi et al., 1997] Bergamaschi, S., Sartori, C., Beneventano, D., and Vincini, M. (1997). Odb-tools: A description logics based tool for schema validation and semantic query optimization in object oriented databases. In Lenzerini, M., editor, *AI*IA*, volume 1321 of *Lecture Notes in Computer Science*, pages 435–438. Springer.

[Bleiholder and Naumann, 2008] Bleiholder, J. and Naumann, F. (2008). Data fusion. *ACM Comput. Surv.*, 41(1).

[Bolchini et al., 2009] Bolchini, D., Finkelstein, A., Perrone, V., and Nagl, S. (2009). Better bioinformatics through usability analysis. *Bioinformatics*, 25(3):406–412.

[Buneman et al., 2001] Buneman, P., Khanna, S., and Tan, W. C. (2001). Why and where: A characterization of data provenance. In *ICDT*, pages 316–330.

[Cali et al., 2002] Cali, A., Calvanese, D., Giacomo, G. D., and Lenzerini, M. (2002). Data integration under integrity constraints. In *Information Systems*, pages 262–279. Springer.

[Carollo et al., 2005] Carollo, V., Matthews, D. E., Lazo, G. R., Blake, T. K., Hummel, D. D., Lui, N., Hane, D. L., and Anderson, O. D. (October 2005).

Graingenes 2.0. an improved resource for the small-grains community. *Plant Physiology*, 139(2):643–651.

[Castano et al., 2001] Castano, S., Antonellis, V. D., and di Vimercati, S. D. C. (2001). Global viewing of heterogeneous data sources. *IEEE Trans. Knowl. Data Eng.*, 13(2):277–297.

[Cheney et al., 2009] Cheney, J., Chiticariu, L., and Tan, W. C. (2009). Provenance in databases: Why, how, and where. *Foundations and Trends in Databases*, 1(4):379–474.

[Chiticariu and Tan, 2006] Chiticariu, L. and Tan, W.-C. (2006). Debugging schema mappings with routes. In *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*, pages 79–90. VLDB Endowment.

[Cui and Widom, 2003] Cui, Y. and Widom, J. (2003). Lineage tracing for general data warehouse transformations. *The VLDB Journal*, 12(1):41–58.

[Cui et al., 2000] Cui, Y., Widom, J., and Wiener, J. L. (2000). Tracing the lineage of view data in a warehousing environment. *ACM Trans. Database Syst.*, 25(2):179–227.

[Dannaoui et al., 2012a] Dannaoui, A. R., Beneventano, D., Milc, J., Caffagni, A., Bergamaschi, S., and Pecchioni, N. (2012a). Release of the cerealab database v 2.0. In *Database Journal*.

[Dannaoui et al., 2012b] Dannaoui, A. R., Sala, A., Beneventano, D., Milc, J., Caffagni, A., and Pecchioni, N. (2012b). Release of the cerealab database v 2.0. In *56 ANNUAL CONGRESS SOCIETA ITALIANA DI GENETICA AGRARIA*.

[Dannaoui et al., 2012c] Dannaoui, A. R., Tumino, G., Beneventano, D., Milc, J., Caffagni, A., and Pecchioni, N. (2012c). Release of the cerealab database v 2.0 for rice marker-assisted selection. In *Crop Improvement in a Changing Environment: the RISINNOVA Project for sustainable rice production in Italy*.

[Dong and Naumann, 2009] Dong, X. L. and Naumann, F. (2009). Data fusion: resolving data conflicts for integration. *Proc. VLDB Endow.*, 2(2):1654–1655.

[Duvick et al., 2008] Duvick, J., Fu, A., Muppirala, U., Sabharwal, M., Wilkerson, M. D., Lawrence, C. J., Lushbough, C., and Brendel, V. (2008). Plantgdb: a resource for comparative plant genomics. *Nucleic Acids Research*, 36(suppl 1):D959–D965.

[Exner et al., 2008] Exner, V., Hirsch-Hoffmann, M., Gruissem, and Wilhelm; Hennig, L. (2008). Plantdb - a versatile database for managing plant research. *Plant Meth.*

[Fagin et al., 2005] Fagin, R., Kolaitis, P. G., Miller, R. J., and Popa, L. (2005). Data exchange: semantics and query answering. *Theor. Comput. Sci.*, 336(1):89–124.

[Fellbaum, 1998] Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. Bradford Books.

[Glavic and Alonso, 2009] Glavic, B. and Alonso, G. (2009). Perm: Processing provenance and data on the same data model through query rewriting. In *Proceedings of the 2009 IEEE International Conference on Data Engineering*, pages 174–185, Washington, DC, USA. IEEE Computer Society.

[Glavic et al., 2010] Glavic, B., Alonso, G., Miller, R. J., and Haas, L. M. (2010). Tramp: Understanding the behavior of schema mappings through provenance. *PVLDB*, 3(1):1314–1325.

[Green et al., 2007] Green, T. J., Karvounarakis, G., and Tannen, V. (2007). Provenance semirings. In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, PODS '07, pages 31–40, New York, NY, USA. ACM.

[Haas, 2007] Haas, L. M. (2007). Beauty and the beast: The theory and practice of information integration. In Schwentick, T. and Suciu, D., editors, *ICDT*, volume 4353 of *Lecture Notes in Computer Science*, pages 28–43. Springer.

[Halevy and Li, 2003] Halevy, A. and Li, C. (2003). Information integration research: Summary of nsf idm workshop breakout session. *NSF IDM Workshop*.

[Halevy et al., 2006a] Halevy, A., Rajaraman, A., and Ordille, J. (2006a). Data integration: the teenage years. In *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*, pages 9–16. VLDB Endowment.

[Halevy et al., 2006b] Halevy, A. Y., Rajaraman, A., and Ordille, J. J. (2006b). Data integration: The teenage years. In Dayal, U., Whang, K.-Y., Lomet, D. B., Alonso, G., Lohman, G. M., Kersten, M. L., Cha, S. K., and Kim, Y.-K., editors, *VLDB*, pages 9–16. ACM.

[Heimbigner and McLeod, 1985] Heimbigner, D. and McLeod, D. (1985). A federated architecture for information management. *ACM Trans. Inf. Syst.*, 3(3):253–278.

[Hochheiser and Shneiderman, 2003] Hochheiser, H. and Shneiderman, B. (2003). Dynamic querying for pattern identification in microarray and genomic data. In *IEEE International Conference on Multimedia and Expo, 2003. Reference 124 Insight-Based Studies for Pathway and Microarray Visualization Tools*.

[Ikeda and Widom, 2009] Ikeda, R. and Widom, J. (2009). Outerjoins in uncertain databases. In *Management of Uncertain Data (MUD)*. Stanford InfoLab.

[Inmon, 2002] Inmon, W. H. (2002). *Building the Data Warehouse,3rd Edition*. John Wiley & Sons, Inc., New York, NY, USA.

[Ives, 2009] Ives, Z. (2009). Data integration and exchange for scientific collaboration. In Paton, N., Missier, P., and Hedeler, C., editors, *Data Integration in the Life Sciences*, volume 5647 of *Lecture Notes in Computer Science*, pages 1–4. Springer Berlin Heidelberg.

[Ives et al., 2008] Ives, Z. G., Green, T. J., Karvounarakis, G., Taylor, N. E., Tannen, V., Talukdar, P. P., Jacob, M., and Pereira, F. (2008). The orchestra collaborative data sharing system. *SIGMOD Rec.*, 37(3):26–32.

[Jagadish et al., 2007] Jagadish, H. V., Chapman, A., Elkiss, A., Jayapandian, M., Li, Y., Nandi, A., and Yu, C. (2007). Making database systems usable. In *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 13–24, New York, NY, USA. ACM.

[Jagadish and Olken, 2004] Jagadish, H. V. and Olken, F. (2004). Database management for life sciences research. *SIGMOD Rec.*, 33(2):15–20.

[Javahery, 2004] Javahery, Seffah, K. (2004). Beyond power: Making bioinformatics tools usercentric. communications of the acm. *Special Issue on Bio-Informatics*, 47:5863.

[Kahraman et al., 2005] Kahraman, A., Avramov, A., Nashev, L. G., Popov, D., Ternes, R., Pohlenz, H.-D., and Weiss, B. (2005). Phenomicdb: a multi-species genotype/phenotype database for comparative phenomics. *Bioinformatics*, 21(3):418–420.

[Kolaitis, 2005] Kolaitis, P. G. (2005). Schema mappings, data exchange, and metadata management. In *PODS '05: Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 61–75, New York, NY, USA. ACM.

[Lee et al., 2005] Lee, J. M., Davenport, G. F., Marshall, D., Ellis, T. N., Ambrose, M. J., Dicks, J., van Hintum, T. J., and Flavell, A. J. (2005). GERMINATE. A Generic Database for Integrating Genotypic and Phenotypic Information for Plant Genetic Resource Collections. *Plant Physiol.*, 139(2):619–631.

[Lenzerini, 2002] Lenzerini, M. (2002). Data integration: A theoretical perspective. In Popa, L., editor, *PODS*, pages 233–246. ACM.

[Liang et al., 2008] Liang, C., Jaiswal, P., Hebbard, C., Avraham, S., Buckler, E. S., Casstevens, T., Hurwitz, B., McCouch, S., Ni, J., Pujar, A., Ravenscroft, D., Ren, L., Spooner, W., Tecle, I., Thomason, J., Tung, C.-w., Wei, X., Yap, I., Youens-Clark, K., Ware, D., and Stein, L. (2008). Gramene: a growing plant comparative genomics resource. *Nucleic Acids Research*, 36(suppl 1):D947–D953.

[Matthews et al., 2003] Matthews, D. E., Carollo, V. L., Lazo, G. R., and Anderson, O. D. (2003). Graingenes, the genome database for small-grain crops. *Nucleic Acids Research*, 31(1):183–186.

[Milc et al., 2011] Milc, J., Sala, A., Bergamaschi, S., and Pecchioni, N. (2011). A genotypic and phenotypic information source for marker-assisted selection of cereals: the cerealab database. *Database*, 2011.

[Moreau, 2009] Moreau, L. (2009). The foundations for provenance on the web.

[Moreau et al., 2008] Moreau, L., Freire, J., Futrelle, J., Mcgrath, R. E., Myers, J., and Paulson, P. (2008). The open provenance model: An overview.

[Mueller et al., 2005] Mueller, L. A., Solow, T. H., Taylor, N., Skwarecki, B., Buels, R., Binns, J., Lin, C., Wright, M. H., Ahrens, R., Wang, Y., Herbst, E. V., Keyder, E. R., Menda, N., Zamir, D., and Tanksley, S. D. (July 2005). The sol genomics network. a comparative resource for solanaceae biology and beyond. *Plant Physiology*, 138(3):1310–1317.

[Naumann and Bleiholder, 2006] Naumann, F. and Bleiholder, J. (2006). Conflict handling strategies in an integrated information system. In *Proceedings of the WWW Workshop in Information integration on the Web (IIWEB)*.

[Naumann et al., 2004] Naumann, F., Freytag, J. C., and Leser, U. (2004). Completeness of integrated information sources. *Inf. Syst.*, 29(7):583–615.

[Noy, 2004] Noy, N. F. (2004). Semantic integration: A survey of ontology-based approaches. *SIGMOD Record*, 33(4):65–70.

[Orsini, 2004] Orsini, M. (2003/2004). Interoperabilitá tra ontologie eterogenee: i traduttori OWL - ODLI3. Master's thesis, University of Modena and Reggio Emilia.

[Orsini, 2009] Orsini, M. (2009). *Query Management in Data Integration Systems: the MOMIS approach*. PhD thesis, International Doctorate School in Information and Communication Technologies of the University of Modena and Reggio Emilia.

[Po, 2009] Po, L. (2009). *Automatic Lexical Annotation: an effective technique for dynamic data integration*. PhD thesis, International Doctorate School in Information and Communication Technologies of the University of Modena and Reggio Emilia.

[Sala and Bergamaschi, 2009] Sala, A. and Bergamaschi, S. (2009). A mediator-based approach to ontology generation and querying of molecular and phenotypic cereals data. *IJMSO*, 4(1/2):85–92.

[Schreiber et al., 2012] Schreiber, F., Colmsee, C., Czauderna, T., Grafahrend-Belau, E., Hartmann, A., Junker, A., Junker, B. H., Klapperstck, M., Scholz, U., and Weise, S. (2012). Metacrop 2.0: managing and exploring information about crop plant metabolism. *Nucleic Acids Research*, 40(D1):D1173–D1177.

[Sheth and Larson, 1990] Sheth, A. P. and Larson, J. A. (1990). Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22(3):183–236.

[Wiederhold, 1992] Wiederhold, G. (1992). Mediators in the architecture of future information systems. *IEEE Computer*, 25(3):38–49.

[Zhao et al., 2006] Zhao, W., Canaran, P., Jurkuta, R., Fulton, T., Glaubitz, J., Buckler, E. S., Doebley, J., Gaut, B., Goodman, M., Holland, J., Kresovich, S., McMullen, M. D., Stein, L., and Ware, D. (2006). Panzea: a database and resource for molecular and functional diversity in the maize genome. *Nucleic Acids Research*, 34(Database-Issue):752–757.