the multi-cloud context in which data is delivered according to data consumers expectations by profiting from previous integration results instead of launching the expensive data integration process from the first step.

3.3 State of the Art

The state of the art comprehend four topics, which are discussed in the following sections: (i) data integration and data quality in the database domain; (ii) data integration approaches in the cloud and in service-oriented contexts; (iii) query rewriting approaches; and (iv) service level agreements for cloud computing.

3.3.1 Data integration and data quality in the database domain

The data integration problem intends to offer to data consumers an integrated view of data build by combining and merging several databases *known-in-advance*. This problem has been widely examined in the database theory. [Lenzerini 2002] discussed theoretical aspects in data integration including modeling applications, query evaluation, dealing with inconsistencies and reasoning queries. The core of data integration is the rewriting process. Many authors have reported algorithms for this purpose. For instance, [Levy 1996] proposed the *bucket algorithm*, which generates combinations of the different *buckets* (views mapped to a given subgoal), and checks whether each one is a rewriting of the query. [Duschka 1997] introduced the *inverse-rules algorithm*, which produces a set of *inverse rules* from local views to the global view. Then, a rewriting is obtained by unfolding the query in terms of the *inverse rules*. [Pottinger 2001] presented the *MiniCon algorithm* funded in ideas of [Duschka 1997]. Although *MiniCon* has shown a more efficient implementation, these kind of algorithm share the same performance problem while combining views to produce a rewriting. Several query rewriting approaches were reviewed in [Halevy 2001].

Data quality issues in data integration have been explored in many researches. [Gertz 1998] designed a framework for modeling quality aspects (such as timeliness, accuracy and completeness) in databases integration. The framework stores quality aspects as meta-data in the integration level, and based on this information delivers a high-quality data from local databases. [Scannapieco 2004] presented a data quality broker that allows to submit queries with associated quality requirements over a global schema and to provide results according to them. [Batista 2007] proposed a quality criteria (such as reputation, availability, response time, completeness, among others) analysis of data integration elements (sources, schemas and data) to improve quality on query execution. [Abdel-Moneim 2015] proposed a method to improve quality in data integration solutions by adding and storing a set of data quality measures in data sources. Then, given a user query and quality preferences, the most relevant sources are selected to answer the query. [Monem 2016] introduced a framework (called DIRA), which produces the *top-k* query answers using an algorithm that selects data sources based on quality aspects stored as meta-data

in a data source such as accuracy, validity, completeness, among others. Others data integration issues and quality aspects in data integration systems are tackled in [Batini 2006, Angeles 2009, Boufares 2012].

3.3.2 Data integration in the cloud and in service-oriented domains

The data integration has also been addressed in cloud and service-oriented contexts. For instance, [Yau 2008] introduced a repository in which based on user integration requirements, the repository retrieves and integrates the data collected from different services focusing on data privacy issues. [Tian 2010] proposed an inter-cloud data integration system which considers privacy requirements and the cost for protecting and processing data. According to the user privacy requirements, the system decides where is the best location to execute the query while meeting privacy and cost constraints: direct on service providers or on its own cloud repository. [Benslimane 2013] designed a composition-based approach for data integration taking into consideration the privacy of the data manipulated by web services and compositions. [Yau 2008, Tian 2010, Benslimane 2013] considered privacy and cost while integrating data, but other quality aspects associated to data itself (such as veracity, type, freshness, among others) and to the infrastructure are still missing.

Some authors have addressed data integration in service-oriented contexts taking into consideration the requirement of computing resources for integrating data across several data service. For instance, [Correndo 2010] presented a method for data integration using SPARQL on Linked Data. The objective is to solve the entity co-reference problem and to exploit ontology alignments interested in data manipulation. [Thor 2011] introduced an mashup-based integration system called *CloudFuice*. It provides a language for specifying dataflows for performing parallel data integration tasks. [Alsubaiee 2012] introduced a system which allows to store, consume, integrate and analyze social and wed data in a scalable manner. [Mubeen 2012] proposed a new architecture for web-scale data integration in which the data extracted from services are stored in a single cloud data store. [ElSheikh 2013] designed the *SODIM* which combines data integration, service oriented architecture and distributed processing applying MapReduce techniques. The system works on a pool of collaborative services and it can process a large number of databases represented as web services. [Hong 2014] introduced a cloud data service system to integrate data from distributed data services considering three level of data security focusing on data privacy. In general, [Correndo 2010, Thor 2011, Alsubaiee 2012, Mubeen 2012, ElSheikh 2013, Hong 2014] exploited parallel settings for implementation costly data integration processes.

3.3.3 Query rewriting approaches

In traditional databases theory, query rewriting activities are essential to data integration solutions. On cloud and service-oriented context query rewriting issues are commonly referred as a service matching and composition problem in which

given a query, the objective is to match and compose services that can contribute to produce a result. [Barhamgi 2010] proposed a query rewriting approach which processes queries on data provider services. The query and data services are modeled as RDF views. Then, a rewriting answer is a service composition in which the set of data service graphs fully satisfy the query graph. Inspired by [Barhamgi 2010], [Benouaret 2011] introduced a service composition framework to answer preference queries which ranks the compositions based on previously computed scores. [Costa 2013] presented a refinement approach of web service compositions in which given an abstract query specification, a set of concrete service compositions are produced. [Ba 2014] extended [Costa 2013] introducing the notion of user preferences and scores used to rank services and compositions while rewriting the query specification. As in the database domain, the algorithms discussed above deal with performance issues while rewriting depending on complexity of the query and on the number of available services. Although [Benouaret 2011, Ba 2014] have considered preferences and scores to produce rewritings, the multi-cloud context introduces new requirements and constraints to the integration process. Currently, the approaches are not sufficient to cover the new challenges. Thus, they should be revisited and adapted in order to make the integration efficient in this environment.

3.3.4 Service level agreements for cloud computing

Service level agreements (SLA) state what a *consumer* can expect from a system or system behavior. Several researchers have reported studies on SLAs in different domains [Alhamad 2011]. On cloud computing, contributions associated to SLAs manly comprehend: *(i)* management, negotiation and matching of SLAs; *(ii)* security issues; and *(iii)* resource allocation.

For instance, [Redl 2012] introduced a method for matching of SLA elements exported by different cloud providers. [Mavrogeorgi 2013] presented a SLA management framework responsible to negotiate and enforce customized SLAs enriched with the information of the cloud federation and renegotiation rules. [Son 2014] introduced SLA negotiation strategies for a multi-cloud broker and discussed design issues. [Falasi 2016] designed an SLA negotiation model to based on the game-theory to allow cloud services to specify, negotiate and establish SLAs.

Other proposals intends to assure security aspects in the cloud by defining and including security requirements on contracts. For instance, [Rak 2013] introduced an approach to specify, assess and integrate security requirements to cloud services. [Rojas 2016] proposed a framework for management of SLA security requirements and enforcing them during the entire SLA lifecycle. [Casola 2016] specified a catalogue of security services that can be monitored and negotiated through SLAs.

Finally, several works address SLA monitoring to detect and avoid violations. For example, [Brandic 2010] designed a model to identify and propagate SLA violation to the cloud providers. [Leitner 2010] introduced a event-based system to predict SLA violations and take necessary measures before they have impacted the provider SLA. [Emeakaroha 2012] proposed a system to monitor SLA in order to detect violation

Database	Amount	Include	Excluded
IEEE	658	56	602
ACM	649	31	618
Science Direct	106	6	100
CiteSeerX	419	21	398
Total	1832	114	1718

Table 3.1: Systematic Mapping: sumary of paper per databse.

and provide tools for resource allocation and scheduling. [Hussain 2015] described a profile-based SLA prediction model, which predicts the necessary resources based on user reputation history.

To the best of our knowledge, related work on SLA models does not seem to address data integration in multi-cloud architectures. Probably, because the contributions are mainly interested in specifying and monitoring performance aspects (for example, response time, memory, availability, among others) in the SLA rather than quality issues regarding the data properties (such as data type, veracity, freshness, provenance and others). Thus, we strongly believe that SLAs can be used to explicitly introduce the notion of quality in the current data integration solutions. In this sense, the use of SLAs to guide the entire data integration in a multi-cloud context seems original and promising for providing new perspectives to the data integration problem.

3.4 Synthesis of the work

3.4.1 First year

During the first year of project, we have been working on the state of the art. The idea is to be aware of all types of publications close related to the thesis proposal. To reach this, we proceeded with a literature analysis using a systematic mapping methodology.

Briefly, the methodology consists in retrieving papers from scientific databases using the same search string. These papers are filtered according to an inclusion and exclusion criteria that should be defined based on the research interests. The papers will be classified in different categories (called facets) and for each facet in a specific dimension. The facets and dimension are defined based on the authors' knowledge and interests. Taking the final papers collection, the abstracts should be read in order to classify each paper into the dimensions for each facet. This methodology allowed us to identify trends and open issues regarding our research topic and proposing an approach that fills some gaps and proposes an original data integration solution according to current trends in the area. The table 3.1 presents our results. We retrieved 1832 papers. 114 papers were selected based on our inclusion and exclusion criteria. As result of this work, we have published a paper on DEXA 2015 (See annex A).