Collaborative Business Intelligence on the Cloud

Mai Kasem
Information Systems Department
Faculty of Computers and Information
Cairo University, Egypt

Ehab E. Hassanein
Information Systems Department
Faculty of Computers and Information
Cairo University, Egypt

Hesham H. M. Hassan
Computer Science Department
Faculty of Computers and Information
Cairo University, Egypt

ABSTRACT
Business Intelligence (BI) solutions help organizations make strategic, informed, and effective decisions via analysing and reporting the organisation's data; the better the data quality, the more accurate and informative the reports are. However, in the real-world complex business landscape, the data from a single organization may not be enough to generate an informed and effective strategic decision. Furthermore, most of the traditional BI solutions are built to serve a single organization; they use the organization's local data to generate decisions, which could lead to incomplete or non-holistic results leading to non-accurate decisions. Collaborative Business Intelligence (CBI) solutions resolve this challenge by extending the decision-making process beyond the organization's boundaries. One of the technologies that can make the CBI solutions more accessible is Cloud Computing. Binding two technologies, such as Business Intelligence and Cloud Computing, helps in extending the CBI solutions to reach more users via cloud accessible services. It could also simplify the way different organizations are connected, and the way the data sharing is governed. This paper introduces a new framework called Collaborative Business Intelligence on the Cloud (CBIC). It utilizes the use of CBI solutions and Cloud Computing service to enable users – who have a data sharing agreement – to connect their data warehouses through a cloud BI service and help them run business intelligence functionalities on their data.

General Terms
Business Intelligence, Cloud Computing, Software-as-a-Service.

Keywords
Cloud Business Intelligence, Collaborative Business Intelligence.

1. INTRODUCTION
Collaboration is considered one of the most important means for enhancing companies' efficiency, competitiveness, and their ability to cope with the changing market [1][2][3]. It is about merging more than one experience, business, and infrastructure to get more effective in the problem-solving process [4][5][6] and extend the companies capabilities beyond their boundaries. Collaboration could be made possible by using collaborative platforms, such as Collaborative Business Intelligence (CBI) platforms [7][8][9]. In CBI, the decisions can be obtained not only from the local information but also from the information outside the company's boundaries [10][11][12]. It can be enabled by using integration approaches that could be investigated along with three main directions: warehousing approaches, federative approaches, and peer-to-peer approaches. Rizzi S. discussed the data integration approaches as follows [1]:

- Warehousing approaches: Warehousing is responsible for data cleaning and integration [13][14][15]. In the warehousing data-integration approaches, the integrated data need to be stored physically in the data warehouse. These approaches assume that all integrated components have the same schema or there is a global schema given, and that's why it supports the static scenarios only. Warehousing approaches are convenient for organizations that have the same business or share their business view.
- Federated approach: It can be achieved by setting a global schema that represents the common business features of the organization. The integrated data does not need to be stored physically in the data warehouse, which makes the query management process more complex. However, it enables a flexible architecture model where dynamically inserting data warehouses is possible. In the federative approach, there are two basic ways for specifying the mapping in a data integration system, local-as-view (LAV), and global-as-view (GAV) [16][17][18]. In the LAV approach, the content of each source is characterised in terms of a view over the global schema, which leads to the complexity of the query processing in the LAV approach. On the other hand, GAV approach models the global schema as a set of views over the heterogeneous organizations' data sources. In the GAV approach, the mapping of each element in the global schema is represented as a query over the organization's data source and most GAV systems' query-answering is based on a simple unfolding strategy.
- Peer-to-Peer approach: It is different from the previous approaches as the integration between the data warehouses does not need a global schema. In this approach, the data is distributed over different peers rather than centralized. Each peer can add, edit, or delete its information and can join or leave the system at any time. Mapping the local schema at each peer is considered one of the challenges of this approach.

Cloud computing is another technology that provides companies with business agility and helps them to deal with the changing business need in an effective way [19][20][21][22]. This paper discusses the concept of binding the cloud computing technology with the CBI, introducing a new concept that will integrate the benefits of the cloud and the benefits of collaboration in the CBI tools. Combining those two technologies could enable organizations to focus on improving their future by understanding the market and taking the best decisions based on inside and outside information.

Following is an outline of this paper. Section 2 analyses one

of the CBI frameworks that are based on the peer-to-peer integration approach. It also highlights its benefits and open issues. Section 3 proposes the Collaborative Business Intelligence on the Cloud (CBIC) framework. It describes the components of the proposed framework with their design details, and it provides some examples to show how it handles different scenarios. The section ends with a comparison between the CBIC framework and the Business Intelligence Network (BIN), showing the commonalities and differences between them. Section 4 validates the CBIC framework against the BIN framework. It runs three experiments that aim at comparing the performance of both frameworks over a different number of network's layers (network depth), organizations, and returned results. Section 5 concludes the research, and section 6 highlights the future work topics.

2. BUSINESS INTELLIGENCE NETWORK

The CBI approaches - that are discussed in the previous section - have demonstrated various data sharing techniques that can help organizations gain more collaboration benefits; they work on broadening the data used for the decision-making process beyond the organization's boundaries. Golfarelli M. et al., 2012 described one of the frameworks that apply those collaborative data sharing techniques using business intelligence capabilities over a network of peers. This framework is called Business Intelligence Network (BIN) [23][24]. The BIN consists of a network of peers; each peer represents a participating company and has its BI platform that represents the company's functionalities and shares business information to support the decision-making process. Although different companies could be in different locations or have different business specialties, they can have mutual benefits by working in agreed upon way. Since the BIN is based on the Peer-to-Peer approach, it does not rely on a shared schema; each peer has full control over setting or editing its schema and the information it shares.

The BIN works as follows: Firstly, the user accesses the local multidimensional schema exposed by her peer p for formulating OLAP query q. During processing the query q locally on the data warehouse of peer p, peer p will get the mapping of all the neighbouring peers, reformulate the user's query q to match the neighbouring peers schemas using the peers' mapping, and then passes the reformulated query to them. The next peers will do the same thing by executing the formulated query on themselves then transform it and pass it over to their neighbouring peers, and so on. Finally, the queries results get passed back from one peer to another until they reach the user.

2.1 BIN Architecture

The BIN architecture, illustrated in figure 1 [23], shows its main components:

- User interface: This is a web-based component that is responsible for communicating with users. Using this interface, users can send OLAP queries to the local multidimensional schema and get the required results.
- Query Handler: This component is responsible for receiving the OLAP query from the user interface or the other peers on the network and then sending it to the OLAP adapter to get an answer locally. Finally, reformulates the answers and send them back to the asked peers.
- Data Handler: This component is responsible for collecting the query results from the OLAP adapter and the source peers. If the query formulation has been done locally, the data handler integrates the results and sends them back to the user interface. If the query formulation is done on other peers, the data handler collects the query results from the OLAP adapter and sends them over to the target peer.
- Multidimensional Engine: This component is responsible for managing the local data warehouse.

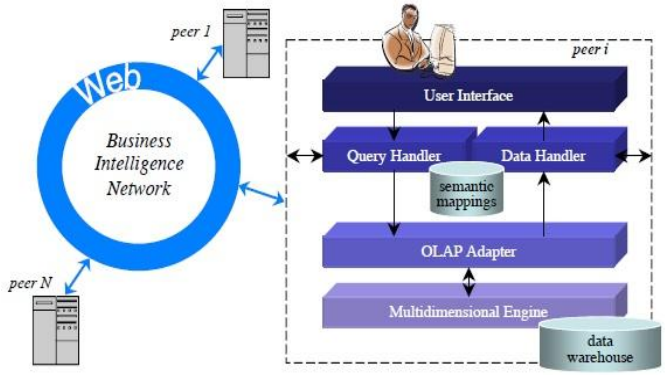

Figure 1: BIN Architecture

2.2 Challenges of the BIN Approach

Although there are many benefits of the BIN approach, there are some open issues and challenges. This study aims at resolving the following challenges of the BIN:

- There is no unified and integrated view for the different business information.
- Query results may not comply with the user's schema, because the BIN is a network of heterogeneous peers with different schemas.
- The BIN does not have techniques for assuring the data origin and quality in order to give the users reliable information.
- The BIN needs advanced security techniques to secure the information of the participating companies as well as approaches for keeping the undesired information away.
- The high network overhead and delays of exchanging messages between peers could lead to a performance impact.
- The need to have the BI solution installed at each peer's endpoint.
- The network peers can be connected like a chain which could cause more delays in the response time. Figure 2 shows an example of the BIN chain peer connection.

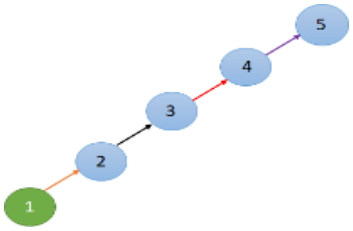

Figure 2: BIN Chain Connection

Also, the network may be connected in a way that has two peers connected to the same peer, which causes redundant data requests, as shown in figure 3.

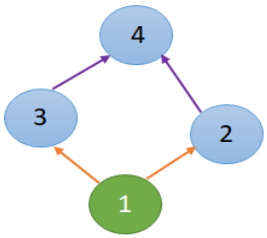

Figure 3: BIN Redundant Request Example

The network may also have some peers connected to each other in a way that could cause an infinite loop of data requests, as shown in figure 4. To solve this problem, each peer will need to keep a record of all the executed queries to prevent running the same query again and prevent the infinite loops of data requests.

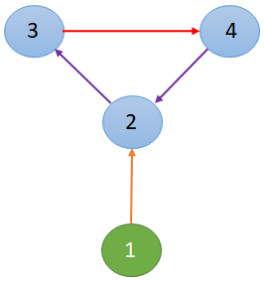

Figure 4: BIN Infinite Loop Example

The next section introduces a new cloud-based framework, called Collaborative Business Intelligence on the Cloud, aiming at resolving the above BIN challenges.

3. COLLABORATIVE BUSINESS INTELLIGENCE ON THE CLOUD FRAMEWORK

The Collaborative Business Intelligence on the Cloud (CBIC) is a new framework – introduced in this section – that provides an easy way for organisations to share their data with other organisations, via a software-as-a-service, decentralised, scalable, and fully autonomous subscription paradigm. It aims at resolving some of the BIN challenges, with more focus on enhancing the way peers are connected to each other. It adopts a hybrid data integration approach between the federative and peer-to-peer approaches, which allows the CBIC framework to benefit from the best features of each data integration approach. The framework acts as a star network (single layer network), where the cloud-based CBI system is responsible for connecting the peers and reducing the mapping overhead with every single peer, as every peer needs to map once and integrate with any. It also reduces the network delay required for hopping from one peer to another as the peers are no longer connecting directly to each other.

3.1 CBIC Framework Design

The CBIC framework adopts a similar data-sharing idea to the BIN framework. In the BIN flow, the original peer will have to wait for all its neighbouring peers to return their results, and each of the neighbouring peers will have to wait for its own neighbouring peers, and so on, which indicates that the more peer depth we have (the number of peer network layers) the more time it will take to get the results back to the caller. The increasing time will be equal to the network communication delay between the peers.

There is also the extra service calling overhead that each peer has to do, that does not exist in the CBIC system. The CBIC system connects directly to the organization's database; the BIN peer, on the other hand, passes the query request to the next peer as a service/remote-procedure call and then the peer will connect to its local DB, execute the query, and return the response to the caller-peer.

The CBIC framework will follow the idea of the schema mapping and query reformulation from the BIN framework, but with some enhancements. Instead of each peer having to map its schema to the neighbouring peers, the CBIC framework will enable the peer to map its schema once with a global mediator schema; and instead of each peer having to reformulate the user's query to match each neighbouring peer schema, the CBIC framework will perform the query reformulation/transformation in a centralized cloud-based system. Doing so will help reduce the mapping overhead from the peers' side, as they do not need to know about other peers' schemas. It will also get the cloud-based framework to utilize the elasticity of the cloud infrastructure, providing resources at the right time when needed. The cloud-based infrastructure allows the framework to scale up and down, providing the required resources as needed when the system load is high, and switching them back off when the system load goes down, covering all the processing-resource needs. Also, having the framework on the cloud removes the need to have the BI system installed at each peer's site, as the users can access the framework over the internet.

Moving forward, we will start calling peers as organizations/subscribers when talking about the CBIC framework because the organizations do not have a direct relation with each other.

3.2 Global Schema usage in CBIC

The framework is designed to provide a unified querying-interface to access the shared data through a global schema. The global schema represents the key features of the business context and allows the information to be retrieved from the different subscribed organizations' databases. As the different organizations' schemas are usually heterogeneous with different business views, the CBIC design provides the ability for each organization to integrate with multiple global schemas, covering different business domains. For example, each schema could be generically designed to fit a business domain, e.g. sales, communications, etc. Figure 5 shows an example of having multiple global schemas in the CBIC framework.

The global schemas in the CBIC framework are not meant to contain any data themselves; they act as mapping descriptions to help unifying the way all organizations access their local data. Metadata needs also to be attached to the global schema fields and act as a dictionary for the users and queries, helping the users to understand what each field is representing during the mapping process.

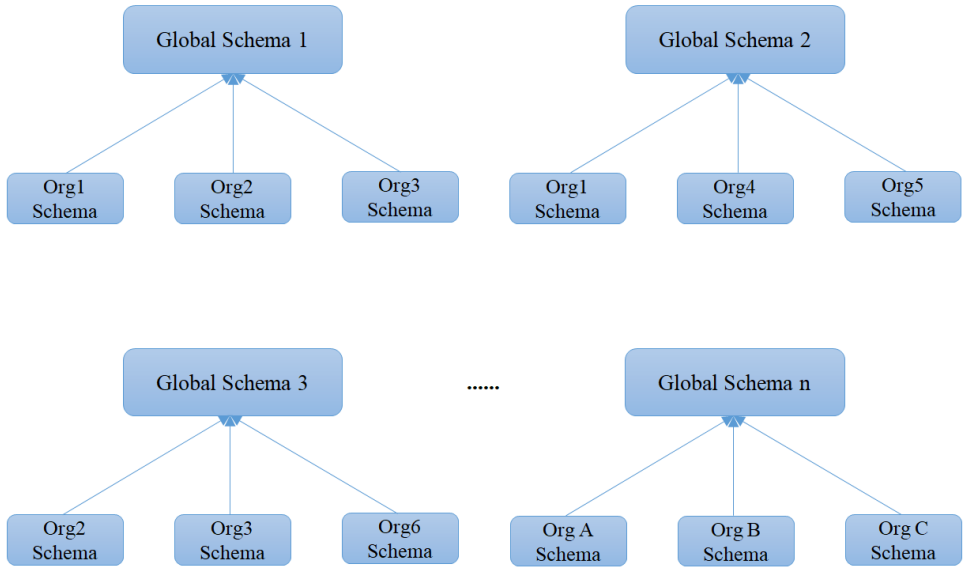

Figure 5: CBIC Global Schema

3.3 Data Sharing in CBIC

The cloud and federative nature of the framework enable simple data sharing between the different users. With the right implementation, the CBIC framework could enable the organization to request access to other organizations data, or it could enable it to grant the other organizations access to its shared data. This process can be done seamlessly without the need to understand the other organization's schema structure, as each organization only needs to work on the mapping of their own schema to the global one.

One of the real-life data-sharing agreement examples is Loyalty Programs [25][26][27], which aims at encouraging shoppers to return to the stores where they frequently make purchases, by providing the shoppers with some incentives. The Loyalty Programs usually involve the participation of multiple companies with data coalition agreement between them. The CBIC framework can help these companies benefit from accessing each other's shared data. One of the loyalty programs examples is a petrol company having a data coalition agreement with a travel agency. When the customers opt-in their loyalty program, the customers use the loyalty program card every time they fill in petrol and every time they use the travel agency for services. This coalition could allow the travel agency to find out their customer location based on the last petrol pump station they used, which allows the travel agency to use this data to better target the customer with an offer that they are more likely to accept.

The framework flexibility could also enable commercial data sharing use cases. For example, marketing companies selling their reports or data in the market to be used for a specific period of time, which is an approach similar to the one used by the Data Management Platforms for advertising campaigns. Another example could be selling a number of query executions on a pay per use model.
Unlike the BIN where the user has no control over the peers they are getting the data from, the CBIC framework could be implemented in a way that enables the user to choose where to run their query. The framework design can execute the query on the user's local mapped schema, on a combination of other organization's schema, or all organizations' schemas that the user can access.

3.4 Framework Architecture

The CBIC architecture consists of four main components, shown in figure 6.

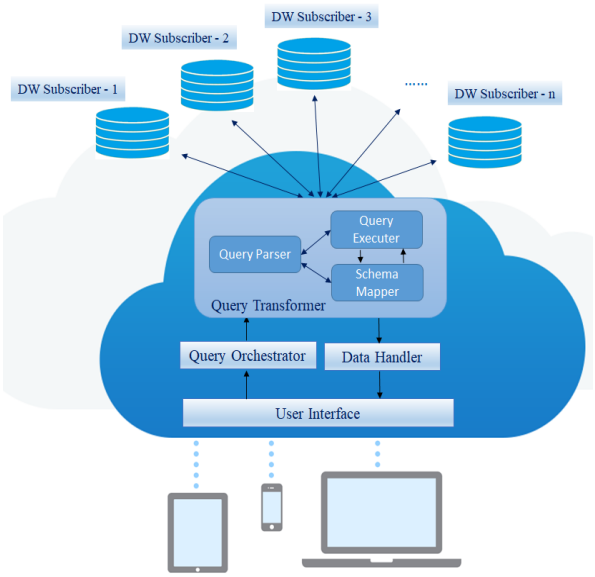

Figure 6: CBIC Architecture

1. **User Interface**: This component manages the interaction with the users and helps them to set their mapping with one of the global schemas, write the query syntax, and explore the query results. The User Interface component acts as a visual representation that can be implemented in any user experience technology. For example, the framework could have a web-based user interface and a mobile app interface that both access the framework over the internet and extends the reach of the framework to more users regardless of their physical locations.
2. **Query Orchestrator**: This component receives the query from the user interface and passes it over to the Query Transformer. It enables the framework to

operate in an omnichannel approach, where the different types of user interfaces could be hooked to the system and provide the user with the same experience.

3. **Query Transformer**: This component is responsible for transforming the user query to the format that fits each organization's schema structure. It then executes the transformed query on the target organizations' schemas and returns the results to the Data Handler component. The Data Handler then integrates or merges the query results and returns it to the user interface. Because all the query results are returned in the structure of the global schema, the User Interface can easily represent the data in a visual graph. The Query Transformer consists of three components:

   - **Query Parser**, which is responsible for parsing the query received from the Query Orchestrator, getting the mapping details from the Schema Mapper component, and executing the query transformation logic.
   - **Schema Mapper**, which is responsible for receiving the table and column names from the Query Parser and returning the mapped values for the target organization's schema.
   - **Query Executer**, which takes the transformed query from the Query Parser and sends it to the subscribed data warehouses to be executed. It then returns the query results to the Data Handler component. The Query Executer has been designed to fit multiple database management systems (DBMS), which means different organizations does not have to have the same DBMS running their local schemas; the framework will do the translation for them. However, the queries will have to be written in the global schemas DBMS specifications.

4. **Data Handler**: This component is responsible for taking the results from the Query Transformer component and prepares them in the way the user wants to see them. The Data Handler component is the place where data integration should be implemented. The framework does not dictate a specific data integration technique. Instead, it leaves it open as an implementation choice. However, we used the global schema to integrate between different organizations using the concept of the GAV integration technique.

The following is a possible interaction sequence with the CBIC framework, from the user's point of view:

1. The user accesses the front-end interface, provided by the cloud-based services, to formulate and submit the queries.
2. The CBIC framework takes the query from the user and transforms it to match the subscribed organizations' schemas.
3. The framework then executes the query on all the selected organizations, integrates the results and returns it to the user.

3.5 Proof of Concept

We implement a simple Proof of Concept (PoC) for the CBIC framework, to demonstrate the principle of the framework and highlight some of its potential usages. The PoC will implement the framework in a way that gives each organization the ability to choose the global schemas of their interest and map them to their local schemas. After the mapping is done, the organization will be able to run the default queries that come with the global schema or construct their own queries. The PoC will also implement the data-sharing part that enables the organizations to run their queries on other organization schemas. The PoC is also designed in a way that makes adding new DBMS easy; DBMS query translators can be implemented as modules and get attached to the framework. Once a new DBMS module is added, the PoC user interface will display it automatically to the organizations allowing them to map their local schemas that match the available DBMS.

The query transformation requires a mapping technique to run the queries on multiple organizations' schemas. Although the CBIC framework could work with any mapping technique, we have introduced a simple mapping technique that helps us in building the PoC and validating the CBIC framework against the BIN framework. The mapping technique we proposed uses three types of mapping: Direct Mapping, Query Mapping, and Static Mapping.

- **Direct Mapping**: in this type, a column from the local schema table can be directly mapped to one column in the global schema table. This mapping type requires that both the global and the local columns have the same datatype. It also assumes that both have the same semantic meaning.

The semantic mapping addresses the problem of resolving semantic conflicts between heterogeneous data sources. Some of the semantic integration research resolve this problem by using ontology-based data integration [28][29][30] to explicitly define the schema terms and thus help to resolve semantic conflicts. This topic is a very active area of research and beyond the scope of this research.

- **Query Mapping**: In some cases, the direct mapping type cannot be applied to the column mapping. Some of the examples are: when the global and local columns do not have the same data type, when the local column requires some processing to match the global column format, or when the global column needs to be constructed from multiple local schema columns. In these cases, the column from the global schema table can be mapped to a user-written query that constructs the data of the global column. This type of mapping can be thought of as a view written for a single column.

Because this type of views is not traditional, there are some points that need to be taken into consideration while constructing this column query. The query needs to return the target column as the same name as the global column it is trying to map itself to. It also needs to return all the primary keys of the global schema table even if it is not the mapped column so that the query could be easily joined with the rest of the mapping types.

- **Static Mapping**: In this option, the organization maps the selected column from the global schema table to a fixed value. This type of mapping could be useful when the organization wants to hide any of its data.

Figure 7 shows a mapping example that covers the three mapping types — assuming that an organization wants to map their `Customer_Details` local table to the `Customer` global table in the global schema.

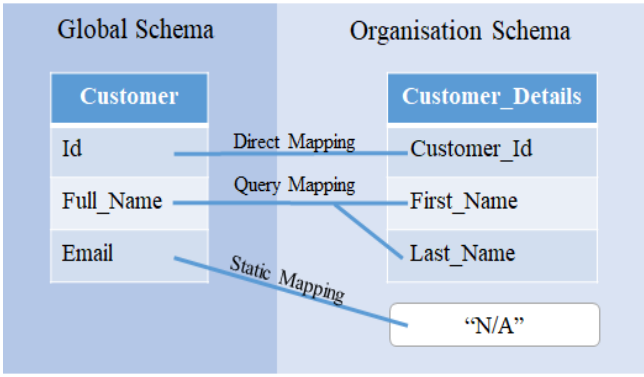

Figure 7: Columns Mapping

The `Customer_Details.Customer_Id` column could be a good candidate for direct mapping; it can be mapped directly to the `Customer.Id` column.

The Query Mapping can be used with the `Customer.Full_Name` column from the global schema as there is no field in the organization schema that matches its value. Table 1 shows the `Customer.Full_Name` query mapping.

Table 1. Query-Mapping Example

```
SELECT Customer_Id AS Id, CONCAT
(First_Name, ' ', Last_Name) AS
Full_Name FROM Customer_Details
```

If the organization does not want to have actual data mapping for the `Customer.Email` field for privacy reasons, or if the organization does not have any value for this field, a static mapping could be used. The `Customer.Email` column in the global schema table can be set to a fixed value "`N/A`".

To be able to run a query using the above mapping technique, the tables referenced by the direct column mappings need to have a queryable relationship to each other. This relationship could be obtained in different ways, such as:

1. Fully Manual: In this way, the tables relationships are obtained manually from the users where the users provide the tables they want to use upfront and provide the information on how the tables should be linked together.
2. Partially Automated: In this way, the user manually provides the tables they want to use for each global schema table, but the relationship between the tables will be detected automatically via the existing foreign key relationships.
3. Fully Automated: Detecting all the involved tables automatically during the direct column mapping process, and automatically detect the relationship between them via foreign keys.

The foreign keys relationship between the tables does not have to be direct. For example, two tables could still be related to each other via an intermediate table that both have a foreign key relationship to. The graph algorithm [31][32] could be used to facilitate the automated detection of the shortest foreign-key-relationship path between the selected tables. Figure 8 shows a graph example with multiple nodes. Each node represents a table, and each arrow (edge) represents a relation between the two tables. Using the graph algorithm, we can easily detect if two nodes/tables have a direct or indirect relationship to each other, and whether they can both be used in the direct mapping of a single global schema table.

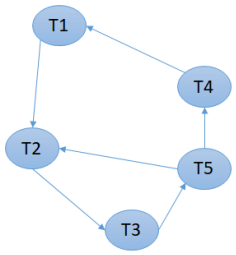

Figure 8: Graph Example

The following examples show the different relationships that need to exist between the tables that are involved in the direct column mapping type.

- **Example 1** (All direct column mapping exists in one table): Figure 9 is an example of when the global columns are fetched from a single organization's table. In this example, all the columns of the `customer` global table exist in the `customer_Details` organization's table.

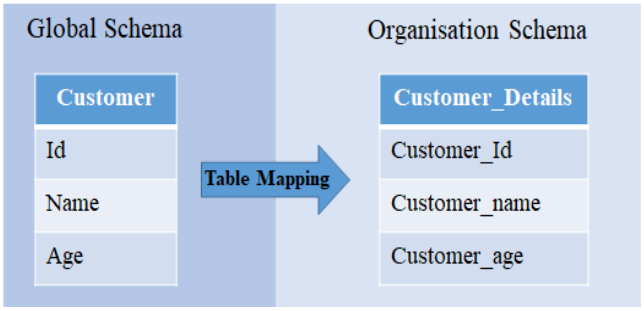

Figure 9: Single Table Mapping

- **Example 2** (Mapped tables are directly connected with a foreign key): In figure 10, the user wants to map the `customer` table from the global schema to the `customers` and `customers_contactInfo` tables from the organization schema. The two tables "`customers` and `customers_contactInfo`" are related by a direct foreign key relationship. Hence, the mapping can be done successfully.

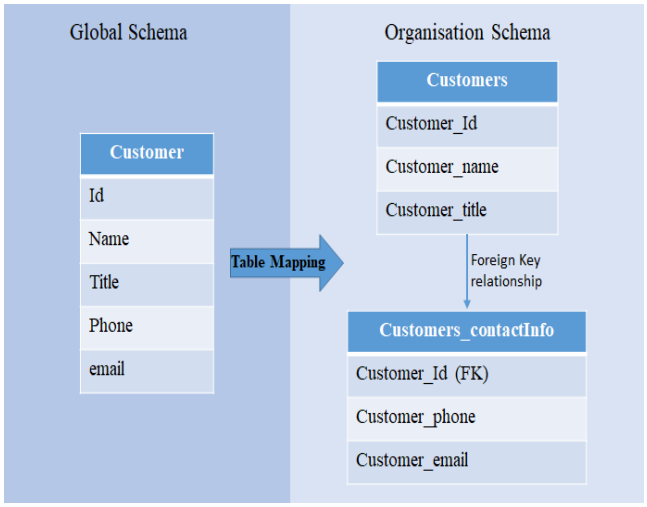

Figure 10: Multiple Table Direct Mapping

- **Example 3** (Indirect Table Mapping): In figure 11, the user wants to map the customer table from the global schema to the personal_details and shipping_address tables from the organization schema. The two tables, "personal_details and shipping_address", are not directly related by a foreign key relationship. However, they are indirectly related via the customers table. Hence, for the mapping to be possible, the customers table will need to be added as one of the mapping tables.

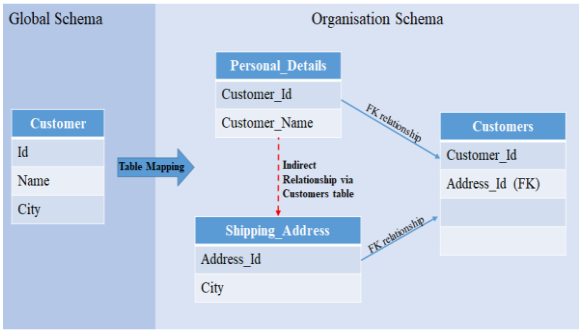

Figure 11: Indirect Table Mapping

- **Example 4** (Mapping is not possible): In figure 12, the user wants to map the `customer` table from the global schema to the `customers` and `region` tables. In this case, the `customers` and `region` tables are neither directly nor indirectly related to each other. Hence, the mapping cannot be done between the `customer` table from the global schema and the "`customers` and `region`" tables from the organization schema.

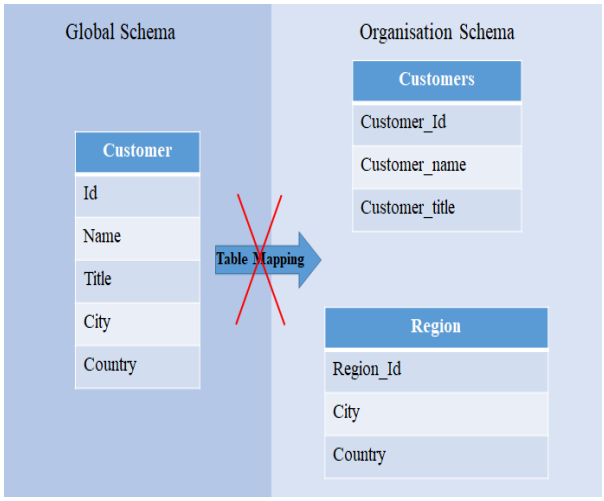

Figure 12: Unable to map between unrelated tables

3.6 Comparison between CBIC and BIN

Table 2 compares the CBIC framework and the BIN framework.

Table 2: CBIC vs. BIN

CBIC	BIN
Cloud service	Network of peers
Organizations do not need to have the BI solution locally installed, as it is a cloud-based service.	
Can merge the query results for all the involved organizations	The query results from other peers will often return results that are not matched to the schema of that query
Has the feature to buy a specific number of query execution on other organizations or sell data reports for a period of time	Does not have the feature of buying/selling data reports
Has a set of global schemas that cover most of the business areas	Does not have a unified view of the different business information
Provide a data sharing agreement between connected organizations to assure that the shared information can be relied on	Does not have techniques for assuring the data source and quality
Connects directly to the organizations' databases	Passes the query request to the next peer as a service/remote-procedure call and then the peer will connect to its local DB, execute the query, and return the response to the caller-peer
Enables the user to choose where to run their query. Users can choose to run the query on their local mapped schema, on a combination of other organization's schema, or on all organizations' schemas that the user can access.	Users have no control over the peers they are getting the data from.

4. FRAMEWORK VALIDATION

This section validates the concept of the Collaborative Business Intelligence framework by comparing its performance – in execution time – to the performance of the Business Intelligence Network. The comparison will focus on the areas with the main design differences, such as:

1. The number of network layers between the peers (network depth) in the BIN framework, and how it impacts the performance of the query; compared to the CBIC single layer network structure.
2. The impact of the number of organizations on both the CBIC and the BIN frameworks.
3. The impact of the number of fetched records on the CBIC and the BIN frameworks.

Each of those three factors has been examined in separate experiments, where the experiments' hypothesis is described, and the experiment results are mentioned and analysed. The experiments output has also been tested with a statistical significance test, and the results between the different frameworks and conditions are confirmed to be statistically significant.

4.1 Selecting the Query-Mapping and the Table Structure

According to the CBIC and the BIN framework designs, both systems are using the same query transformation and mapping techniques. Hence, a simple table with a simple direct mapping will be used in this chapter, to keep the focus on the areas with the main design difference. Table 3 shows the simple table that has been used in the experiments.

The CBIC experiments used the `PRODUCTS` table as the global schema table and used the `GOODS` table as the organizations' table.

Table 3: Experiment Tables & Mapping

PRODUCTS		GOODS
proID int(11)	→	goods_id int(11)
proPrice int(11)	→	Price int(11)
proName varchar(100)	→	goods_name varchar(100)
proBrand varchar(50)	→	Brand varchar(50)
proColour varchar(50)	→	Colour varchar(50)

To make sure the BIN system works under the same circumstances, we used the same table structures between the peers, in a way that forces the query transformation to jump between the `PRODUCTS` table and the `GOODS` table. Figure 13 shows how the tables are being set interchangeably between the peers.

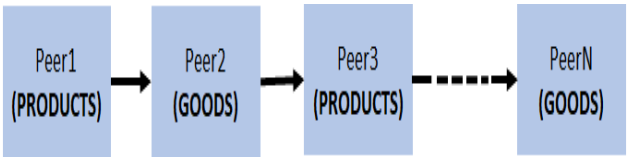

Figure 13: Peers Interchangeable Table Structure

4.2 Selecting the Experiments Query

Both frameworks – the CBIC and the BIN frameworks – are using the same Database Management System, which leads to the same query execution time. Hence, the experiments in this section will not be focusing on the query complexity; instead, a simple query will be used to keep the focus on the other aspects of the experiments. Table 4 shows the simple query that has been used in all the experiments of this chapter.

Table 4: Query Example

```
SELECT productID, productName, colour,
  brand, productPrice FROM products
```

4.3 Machine Preparation

The experiments have been done on two virtual machines, simulating the back and forth connection between the different peers in the BIN network and simulating the host and organizations in the CBIC system. Each virtual machine has been configured with 8GB of RAM, to make sure the operating systems run all the experiments on RAM and never need to use the SWAP memory. The 8GB memory size has been selected based on the memory usage of one of the highest resource consuming experiments (running a 20,000 records query on 25 peers) in the BIN system, which used around 5.8GB of memory. Also, the virtual machines have been configured with two virtual CPUs, running Centos 7 64-bit Linux operating system. The reason for adding 2 CPUs is to allow multi-threading to work properly on the machines.

As an attempt to standardize the experimentation setup on all the frameworks and scenarios, the following machine preparation steps have been performed:

- Switching the machine to the powered mode to prevent any energy saving mode from being activated.
- Connecting the machine to a router with no internet connection to prevent any automatic system updates from taking place.
- Disconnecting all other machines from the router, and only keeping the ones under the test.
- Restarting the machine for a fresh start and leaving it idle for 15 minutes to make sure all the Operating System's start-up processes are completed. The 15 minutes wait is an experimental number based on the time taken for the testing machine to complete all the start-up processes.

4.4 Anomaly Detection

During the execution of the experiments, we noticed that in some rare occasions a request execution could take a very long time to complete; it could take quadruple the time of normal requests, which impacts the accuracy of the experiments. Whenever any of these anomalies are detected, we ignore the whole session and start another one.

4.5 Experiment 1 – Number of Network Layers

One of the main differences between the CBIC framework and the BIN framework is in the way the query is executed. The CBIC performs the query transformation on a centralized cloud system, connects to the organization's database directly, and executes the query. The flow is executed for all the organizations at the same time as a multi-thread execution. Assuming that the cloud system will have enough resources to handle the query execution on all the organizations concurrently, the time taken to run the query will be equal to the time of the slowest organization.

Based on the design difference between the CBIC and the BIN systems, the hypothesis in this experiment is

- The CBIC system will outperform the BIN system even when the BIN peers are directly connected to each other (i.e. a single network layer between the peers), due to the service calling overhead
- The BIN system's performance will degrade as the number of network layers between the peers, increases.

In this experiment we worked on a static number of organizations/peers (10 organizations/peers) and a static

number of records (1000 records); the variable that we will be testing in this experiment is the number of network layers between peers and how it impacts the performance of the BIN network compared to the performance of the CBIC system on the same number of organizations and table records.

We will be testing the BIN performance on 1 Layer, 3 Layers, 5 Layers, 7 Layers, and 9 Layers. Each experiment will be executed 100 times in independent measurement session. In each session, the execution time will be calculated and stored in the output file. The 100 execution time records will then be used to generate the average execution time of the experiment and the standard deviation.

We choose 10 organizations and 1000 records as we had to start from somewhere. However, the impact of changing the number of organizations and records will be examined and analyzed in the later sub-sections.

As shown in table 5 and figure 14, the CBIC system is showing a better performance than the BIN system over all the peer network layers. Also, the performance of the BIN system is degrading as we increase the number of network layers between peers. These results prove the validity of the experiment hypothesis.

Table 5: Experiment 1 Execution Time

System	Execution Time (ms)	Percentage increase from the CBIC system
CBIC	718.18	
BIN 1 Layer	797.21	9.91%
BIN 3 Layers	849.06	15.41%
BIN 5 Layers	1027.8	30.12%
BIN 7 Layers	1218.28	40.81%
BIN 9 Layers	1388.43	48.27%

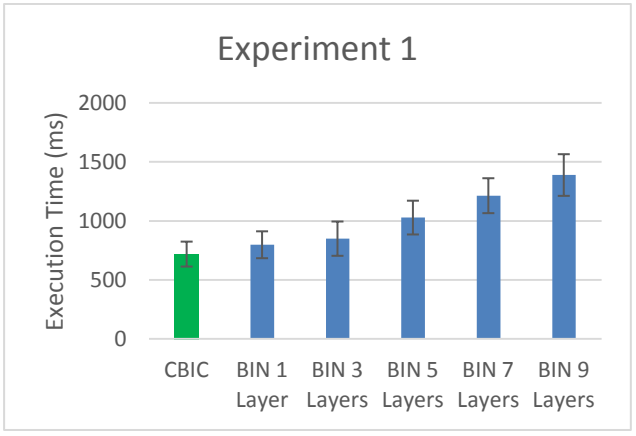

Figure 14: CBIC vs BIN with different layers

4.6 Experiment 2 – Number of Organizations/Peers

In this experiment, the variable that will be tested is the number of organizations/peers and how they impact the performance of the CBIC and the BIN frameworks. Hence, both the number of table-records and the number of network layers between the peers in the BIN framework will be made static.

The number of network layers between BIN peers will be set to 4 layers, as this is almost half the number of layers tested in experiment 1, so it could be considered as the average number of layers tested. It will also enable us to test the performance of the CBIC and BIN framework on a lower number of organizations/peers, such as 5 organizations/peers. The number of records in each table will be set to 1000 records, following the same number used in the first experiment. However, the impact of changing the number of records will be examined and analyzed in the later sub-sections. The BIN and the CBIC performance will be tested on 5, 10, 15, 20, and 25 organizations.

Due to the extra service calling overhead that each peer has to do in the BIN framework - which does not exist in the CBIC system - this experiment hypothesis is set as follows:

- The BIN system's performance will degrade at a higher rate than the CBIC system as the number of organizations increases.

As shown in table 6, the CBIC system is showing a better performance than the BIN system over all different number of organizations, due to the extra service calling overhead that each peer has to do in the BIN framework. Figure 15 and 16 shows that the BIN system's performance degrades in a higher rate than the CBIC system as the number of organizations/peers increases, making the CBIC system even better as the number of organizations increases. These results prove the validity of the experiment hypothesis.

Table 6: Experiment 2 Execution Time

Number of Organizations/Peers	CBIC Execution Time (ms)	BIN Execution Time (ms)
5 Organizations/Peers	433.41	701.74
10 Organizations/Peers	718.18	928.24
15 Organizations/Peers	880.51	1333.04
20 Organizations/Peers	1031.43	1821.48
25 Organizations/Peers	1301.81	2239.72

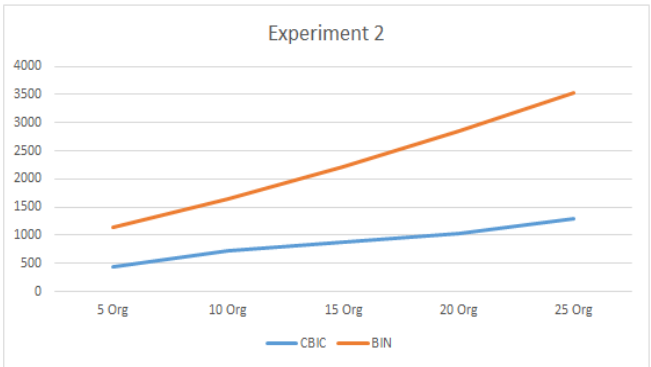

Figure 15: CBIC vs BIN Stacked Line Chart

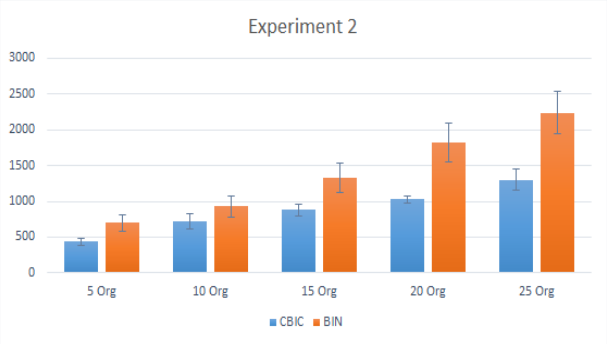

Figure 16: CBIC vs BIN Clustered Column Chart

4.7 Experiment 3 – Number of Records

In this experiment, the variable that will be tested is the number of records and how it impacts the performance of the CBIC and the BIN frameworks. Hence, both the number of organizations/peers and the number of network layers between the peers in the BIN framework will be made static.

The number of network layers between BIN peers will be set to 4 layers, as this is almost half the number of layers tested in experiment 1, so it could be considered as the average number of layers tested. The number of organizations/peers will be set to 10, following the same number used in the first experiment. The BIN and the CBIC performance will be tested in 1000, 5000, 10000, 15000, 20000 records.

The CBIC framework connects to the organization's DB directly while executing the query, using the DBMS connection protocol. The BIN framework relies on retrieving the records via calling service at the neighboring peers, which in turn runs the query on themselves and return it to the caller. This BIN connection mechanism adds an extra service calling overhead and uses different connection protocol than the CBIC framework.

Due to the different connection protocols used by the two frameworks and due to the extra service calling overhead that each peer has to do in the BIN framework, this experiment's hypothesis is set as follows:

- The BIN system's performance will degrade at a higher rate than the CBIC system as the number of table records increase.

As shown in table 7, the CBIC system is showing a better performance than the BIN system over all the different table record numbers, due to the extra service calling overhead that each peer has to do in the BIN framework and the different networking protocols between the CBIC and BIN systems. Figure 17 and 18, show that the BIN system's performance degrades in a higher rate than the CBIC system as the number of fetched table records increases, making the CBIC system even better as the number of records go higher. These results prove the validity of the experiment hypothesis.

Table 7: Experiment 3 Execution Time

Number of Result Records	CBIC Execution Time (ms)	BIN Execution Time (ms)
1000 records	718.18	928.24
5000 records	1096.3	1726.8
10,000 records	1633.73	2293.9
15,000 records	2082.06	3066.68
20,000 records	2531.96	3932.67

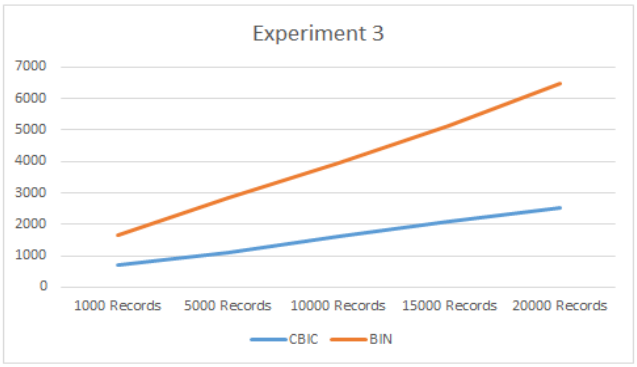

Figure 17: CBIC vs BIN Stacked Line Chart

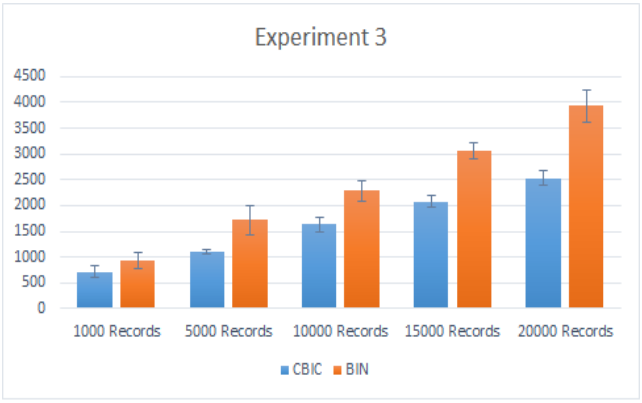

Figure 18: CBIC vs BIN Clustered Column Chart

5. CONCLUSION

Organizations should not depend only on their local data while making business decisions; instead, to make better decisions, they should consider utilizing data from outside the organization boundaries, when possible. Most of the existing CBI solutions focus on collaboration between the organization's departments and how to encourage collaboration between team members. This research focused on how CBI can be extended to cover different organizations, and how CBI could be greatly enhanced through the use of cloud computing technology. The cloud computing technology can make the CBI more accessible and can help in extending the CBI solutions to reach more users via cloud accessible services. It could also simplify the way different organizations are connected, and the way the data sharing is governed.

One of the frameworks that implement CBI between organizations is the Business Intelligence Network. It supports

collaboration between different organizations through a peer-to-peer network using business intelligence capabilities. Although the BIN helps in building the BI collaboration between physically dislocated peers with different business specialties, it could have some integration, connectivity, query-results integrity, and performance challenges.

This research introduced the Collaborative Business Intelligence on the Cloud framework. It enabled organizations to share their data with other organizations through a cloud-based BI service. The framework design resolved some of the BIN challenges, with a focus on enhancing the way peers are connected to each other. The framework acts as a star network (single layer network), where the cloud-based CBI system is responsible for connecting the organizations. The star network design reduced the network overhead of exchanged messages between organizations and reduced the network delay required for hopping from one peer to another as the peers are no longer connecting to each other. The framework also reduced the schema mapping overhead between organizations, as organizations only need to map their schema once to benefit from automatic integration with other organizations. The CBIC framework also resolved the challenge of the lack of a unified and integrated view for the different business information that faces the BIN by offering global schemas that cover different business contexts.

As the design difference between the CBIC and the BIN frameworks could have a performance impact, the research examined the performance of both frameworks. Based on the validation experiments, the research proved that the CBIC framework has higher performance than the BIN framework over a different number of organizations, records, and BIN peer layers. The performance difference is due to the extra service calling overhead that each peer has to do in the BIN framework, which does not exist in the CBIC framework. The different networking protocols between the CBIC and BIN framework is another factor for the performance difference. The experiments also proved that the BIN framework's performance degrades at a higher rate than the CBIC framework, as the number of network layers, organizations, or records increase.

6. FUTURE WORK

The following points outline some of the future work areas.

- Extending the CBIC framework to integrate with an existing cloud-based BI solution.
- Extending the CBIC framework to include the semantic mapping and resolve the semantic conflicts between the heterogeneous data sources.
- Extending the CBIC framework to work on different types of data sources, such as the non-structured databases.
- Researching the possible ways of resolving the data transfer challenge that comes with querying large organizations' schemas over the internet.
- Researching the different information security techniques that could be used to secure the information of the participating organizations.
- Extending the framework to incorporate machine learning capabilities that could make the BI much more effective at identifying hidden insights.
- Extending the framework to use data intelligence capabilities, which could form a better understanding of the collected information and analyzing the operations to make better decisions in the future.

7. REFERENCES

[1] Rizzi, S., 2012. Collaborative business intelligence. In Business Intelligence (pp. 186-205). Springer Berlin Heidelberg.

[2] Matei, G., 2010. "A collaborative approach of Business Intelligence systems", Journal of Applied Collaborative Systems, Vol. 2, No. 2.

[3] Morgan, J., 2012. *4 Reasons Your Company Needs A Collaboration Upgrade*, viewed July 2016, <http://www.fastcompany.com/1842473/4-reasons-your-company-needs-collaboration-upgrade-stat>.

[4] Nixon, N., 2014. *5 Reasons Why Collaboration Is Essential in Today's Business Environment*, viewed July 2016, <http://www.inc.com/natalie-nixon/5-reasons-why-collaboration-is-essential-in-today-s-business-environment.html>.

[5] Steelcase WorkSpace Futures, 2010, How the workplace can improve collaboration.

[6] Scholten, K. and Schilder, S., 2015. The role of collaboration in supply chain resilience. Supply Chain Management: An International Journal.

[7] Mihaela Muntean, 2012, "Theory and Practice in Business Intelligence", West University of Timisoara, Faculty of Economics and Business Administration, Department of Business Information Systems.

[8] Kaufmann, J. and Chamoni, P., 2014, January. Structuring collaborative business intelligence: A literature review. In 2014 47th Hawaii International Conference on System Sciences (pp. 3738-3747). IEEE.

[9] Rabelo, R.J., 2008. Advanced collaborative business ICT infrastructures. In Methods and Tools for collaborative networked organizations (pp. 337-370). Springer, Boston, MA.

[10] Stefanovic, N., 2015. Collaborative predictive business intelligence model for spare parts inventory replenishment. Comput. Sci. Inf. Syst., 12(3), pp.911-930.

[11] Berthold, H., Rösch, P., Zöller, S., Wortmann, F., Carenini, A., Campbell, S., Bisson, P. and Strohmaier, F., 2010, March. An architecture for ad-hoc and collaborative business intelligence. In Proceedings of the 2010 EDBT/ICDT Workshops (pp. 1-6).

[12] "Collaborative Business Intelligence: The Road to Better Decision Making", https://plastergroup.com/collaborative-business-intelligence/, viewed June 2020.

[13] Wrembel, R. ed., 2006. Data Warehouses and OLAP: Concepts, Architectures and Solutions: Concepts, Architectures and Solutions. Igi Global.

[14] Gatziu, S., 1999. Data Warehousing: concepts and mechanisms. In Wirtschaftsinformatik als Mittler zwischen Technik, Ökonomie und Gesellschaft (pp. 61-69). Vieweg+ Teubner Verlag.

[15] Bhatia, P., 2019. Data mining and data warehousing: principles and practical techniques. Cambridge

University Press.

[16] Lenzerini, M., 2002, June. Data integration: A theoretical perspective. In Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (pp. 233-246). ACM.

[17] Halevy, A.Y., 2001. Answering queries using views: A survey. The VLDB Journal, 10(4), pp.270-294.

[18] Levy, A.Y., Mendelzon, A.O., Sagiv, Y. and Srivastava, D., 1995, May. Answering queries using views. In PODS (Vol. 95, pp. 95-104).

[19] Yuvraj Singh Gurjar & Vijay Singh Rathore, 2013, "Cloud Business Intelligence – Is What Business Need Today", International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-1, Issue-6.

[20] Kasem, M. and Hassanein, E.E., 2014. "Cloud Business intelligence survey". *International Journal of Computer Applications*, *90*(1).

[21] Rajagopalan, V. and Jayasingh, S., 2019. Business Intelligence and Cloud Computing: Benefits, Challenges, and Trends. In Global Virtual Enterprises in Cloud Computing Environments (pp. 1-18). IGI Global.

[22] Rajagopalan, V. and Jayasingh, S., 2019. Business Intelligence and Cloud Computing: Benefits, Challenges, and Trends. In Global Virtual Enterprises in Cloud Computing Environments (pp. 1-18). IGI Global.

[23] Golfarelli, M., Mandreoli, F., Penzo, W., Rizzi, S. and Turricchia, E., 2012. "Business Intelligence Networks".

[24] Golfarelli, M., Mandreoli, F., Penzo, W., Rizzi, S. and Turricchia, E., 2010, October. Towards OLAP query reformulation in peer-to-peer data warehousing. In Proceedings of the ACM 13th international workshop on Data warehousing and OLAP (pp. 37-44). ACM.

[25] Liu, Y., 2007. The long-term impact of loyalty programs on consumer purchase behavior and loyalty. Journal of marketing, 71(4), pp.19-35.

[26] Reinartz, W.J., 2006. Understanding customer loyalty programs. In Retailing in the 21st Century (pp. 361-379). Springer, Berlin, Heidelberg.

[27] Kucklick, J.P., Kamm, M., Schneider, J. and Vom Brocke, J., 2020, January. Extending Loyalty Programs with BI Functionalities. In Proceedings of the 53rd Hawaii International Conference on System Sciences.

[28] Pinto, C.S., Jayadianti, H., Nugroho, L.E. and Santosa, P.I., 2012. Solving problems of data heterogeneity, semantic heterogeneity and data inequality: an approach using ontologies. In MCIS2012-The 7th Mediterranean Conference on Information Systems, In Knowledge and Technologies in Innovative Information Systems.

[29] Ram, S. and Park, J., 2004. Semantic Conflict Resolution Ontology (SCROL): An ontology for detecting and resolving data and schema-level semantic conflicts. IEEE Transactions on Knowledge and Data engineering, 16(2), pp.189-202.

[30] Ismail, W.S., Sultan, T.I., Nasr, M.M. and Khedr, A.E., 2013. Semantic Conflicts Reconciliation as a Viable Solution for Semantic Heterogeneity Problems. IJACSA) International Journal of Advanced Computer Science and Applications, 4(4).

[31] Mehlhorn, K., 2013. Data structures and algorithms 1: Sorting and searching (Vol. 1). Springer Science & Business Media.

[32] Mehlhorn, K., 2012. Data structures and algorithms 2: graph algorithms and NP-completeness (Vol. 2). Springer Science & Business Media.