CHAPTER-5

INTEGRATION OF RESULTS

5.0 Introduction

In SIQS, multiple query resolutions are generated, as discussed in Chapter-4, for a user query. Each query resolution generated picks up only one intended aspect of the SMN that is part of the SELECT clause i.e. the SMN either represents a relation or an attribute. This is due to the fact that a SMN may represent more than one relation or attribute present in the component databases registered with the global system and therefore the user query may result in fetching data, corresponding to the SMN, from more than one component database. These query resolutions are transformed to the generated queries, which are executed to give multiple results to the user. Since the user while posing a query specifies certain intent, he/she may show interest in some of the results and not all of them. It may be possible that the information of interest to the user may be present in more than one result. The user then may find these results difficult to interpret and, therefore, may want to view these results in a way that makes them easy to interpret. One way to make these results interpretable is by integrating them. A single result arrived at after integrating these multiple results
would be more desirable as related information about a particular concept will be present in a single tuple instead of being split across multiple tuples in multiple results. Consider the following example.

Suppose the user poses the following user query

```
Select Company Department
Where Cloc = 'BANGALORE' or Dname = 'COMPUTER'
```

The query is executed and the user saves the results of interest to him/her. Suppose the user saves the following results

Result-1 - (C#, Cname, Cloc, Department)
Result-2 - (D#, Dname, Company, Cloc)
Result-3 - (Company, Department)

It can be seen from the above that each of these results store some information, which is not present in the others. The user, therefore, may have decided to save these results as all the information in these results is of interest to him/her. Since, the information is split across many results, the user may find these results difficult to interpret. The user has to go through each of these results in order to find information about a single concept. This is an unnecessary burden on the user. The user, therefore, may wish to see these results as a single result so that these multiple results are simpler to interpret. This may require integration of these results so that a single meaningful result can be presented to the user. This integrated result may be more desirable from the user's point of view.
Also, in SIQS, the user is allowed to pose multiple interrelated queries in addition to a single query in SIQL. This may be possible due to the fact that the user may not be aware of the actual requirements initially while posing the query. Based on the results for a user query, the user may pose another query and at the same time may save the current result for future reference. That is, the result of the user query motivates the user to gather more information from the component databases.

Consider the following example

Suppose the user is interested in finding the information about the companies that have a department named Computer. He/She poses the following query in SIQL

```
Select Company

Where Dname = 'Computer'
```

The query is executed and the user saves the results of interest to him/her. These results are then integrated. Let us assume that the integrated result has the following attributes

```
IR_1 - C#, Cname, Cloc, Company
```

The user on viewing the values in the integrated result may feel inclined to seek more information. The user, also, at the same time may wish to store this integrated result for further reference. Suppose the user now, based on the results, is interested in finding information about the departments in the companies located at Bangalore. The user may pose the following query
Select Department
Where Cloc = 'Bangalore'

The query is executed and the user saves the relevant results for integration. Let the attributes of the integrated result, after the relevant results are integrated, be the following

IR_2 - D#, Dname, Department, C#, Company, Cloc

The user has now two results IR_1 and IR_2. He/She may decide to integrate these results in order to view them as one. So the user who initially was interested in posing a single query lands up posing two queries. This was due to the fact that the location information about the company present in the result of the first query motivated the user to pose another query in order to gather information about the Departments of the Company located at Bangalore. This may have resulted from the fact that the user may not have been sure about his/her requirements while posing the first query. If the user would have known the actual requirements initially, he/she may have posed a single query in SIQL combining the above two queries in SIQL. The user query in SIQL then would have been the following

Select Company Department
Where Cloc = 'BANGALORE' or Dname = 'COMPUTER'

The relevant result, for the user, arrived at after execution of the above user query is integrated and shown to the user. The integration of results for a single query in SIQL is termed as Intra-Query Result Integration whereas integration of results of
multiple queries in SIQL is called Inter-Query Result Integration. They are discussed next.

5.1 Integration

The pictorial representation of the Intra-Query and the Inter-Query result integration is shown in Figure-1. In this figure

- $UQ_i$ denotes the $i^{th}$ user query in SIQL.
- $R_{ij}$ denote the $j^{th}$ result for integration in the $i^{th}$ user query.
- $IR_{ij}$ denote the $j^{th}$ integrated result in the $i^{th}$ user query.
- $R_i$ denote the $i^{th}$ integrated result for integration for the multiple user query.
- $IR_i$ denotes $i^{th}$ integrated result for the multiple user query.

The following are the two ways in which the user can specify his/her intent:

- The user may specify his/her intent by posing a single query in SIQL. Let the user query be $UQ_1$ and number of results, relevant to the user, for integration be $j$ then the Integrated result $IR_{1j-1}$ is the final integrated result.

- The user may specify his/her intent by posing multiple interrelated queries in SIQL. Let the number of user queries be $m$ then $IR_{m-1}$ is the final integrated result.

The Intra-Query result integration and Inter-Query result integration is discussed next.
Figure 1 – Pictorial Representation - Intra and Inter Query Result Integration
5.1.1 Intra-Query Result Integration

In SIQS, the results, for the user query are shown, through a GUI as shown in Figure-5 of Chapter-4, one at a time, to the user. The user is given a choice to save results of interest to him/her. The user on viewing the results may show interest in some or all of the results. The user saves the relevant results, which are then saved, by the GUI, as tables in the database server. Once all the results for the user query are generated and the user has saved the results of interest to him/her, the user in SIQS thereafter is shown all the saved results together, through the GUI, so that he/she may integrate some or all of the results. The GUI is presented to the user in the form of a Window as shown in Figure-2.

Figure-2 – Intra-Query Result Integration Interface
The Window shows the relevant results saved by the user. Each of these results has a check box attached to it. The user checks the box by clicking on it in order to choose the result for integration. There are two buttons in the Window namely Integrate and Exit. Clicking on the Exit button results in exiting from the system. On clicking the Integrate Button, the chosen results are sent to the Result Integrator for integration. After the results are integrated, the integrated result is shown to the user, through the GUI. The GUI, which is in the form of a Window, is shown in Figure-3.

![Figure-3 - Intra-Query Integrated Result Interface](image)

The Window shows the integrated result. The Window has four buttons namely Save, Coalesce, Next and Exit. The Exit button is clicked to exit the system. On clicking the Save button the integrated result is saved in the database server in order to be used for future integration. Clicking the Next button opens a Window, shown
in Figure-2 of Chapter-4, for posing another query in SIQL. On clicking the Coalesce button, a Window for inter-query result integration is presented to the user.

5.1.2 Inter-Query Result Integration

The user in SIQS can also express his/her intent by posing multiple SIQL queries. This may be due to the following reasons

• It may be possible that the user may like to specify his/her intent by posing a sequence of interrelated queries in SIQL instead of posing a single query in SIQL. The results of each of these SIQL queries are integrated at the end to give the final result for the user query.

• It may also be possible that, the user, based on the result for a user query, may pose another query in SIQL. That is, the results for a user query may motivate the user to seek more information and hence the user may feel inclined to pose another query in SIQL. The results for each of these SIQL queries are collected and later integrated to give the final result for the user query.

On clicking the Coalesce button in the Window shown in Figure-3, a new window is opened as shown in Figure-4. The Window shows all the integrated results saved by the user. A check box is attached with each of these results. The user chooses the results by checking on the check box. The Window has two buttons namely Integrate and Exit. The Exit button is used to exit the system. Clicking on the Integrate button opens a new Window, shown in Figure-5, which presents to the
user, the integrated result after integrating the integrated results of each of the SIQL queries. The Window has an Exit button that is clicked to exit the system.

Figure-4 – Inter-Query Result Integration Interface

Figure-5 – Inter-Query Integrated Result Interface
5.2 Result Integration

In SIQS, the results chosen by the user are integrated. This should be integrated in such a manner that the integrated result should be meaningful to the user. Therefore, while integrating the results we need to take into consideration the fact that the results that are integrated are join consistent i.e they should agree on common attributes so that no information is lost. The information about a particular concept in reality should appear in a single tuple in the final integrated result. One way to achieve this is to take an outer join or a natural outer join between the results, which are saved as tables in the database server.

An outer join is an extended form of a regular or inner join where tuples in one relation, having no counterparts in the other relation, appear in the result with nulls, for other attributes, instead of being ignored. The outer join [Date1983] preserves the information of unmatched tuples in the participating relation. A left or a right outer join preserves only one of the participating relations whereas the full outer join preserves both of the participating relations. The natural outer join is an equi-outer join on attributes that are common, with one of the set of common attributes preserved in the resultant relation. There are many approaches proposed to process queries involving outer joins and natural outer joins [Munakata1999] [Legaria1994] [Lee1994][Chen1990][Rajaraman1996]. The outer join and natural outer join are not associative and hence for more than two relations, different ordering of relations for outer joining or natural outer joining may give different results. This can be shown with the help of the following example.
Consider results R1, R2 and R3 shown in Figure-6.

Let us consider the following natural outer join ordering for the results shown in Figure-6.

1. \( (RI \bowtie R2) \bowtie R3 \)
2. \( (R2 \bowtie R3) \bowtie R1 \)
3. \( (R1 \bowtie R3) \bowtie R2 \)

The results based on the above ordering are shown in Figure-7.
Figure-7 – Results based on different Natural Outer Join ordering

It can be seen that the result from natural outer join ordering (1) is meaningful because there is no loss of information as information about each of the company is
present in one tuple. The result from natural outer join ordering (2) is not meaningful, as C002 appears in two tuples in the integrated result. That is, the tuples (C002, ~, ~, ~, MUMBAI) and (C002, ~, D002, ~, ~) make no sense for the user. This is due to the fact that the tuple (C002, MUMBAI) in result R3 has no corresponding match in result R2 and therefore Cname, D#, Dname has a null value in the result, obtained after the natural outer join is performed between result R2 and result R3. This tuple that contains the value for D# as null has no match in result R1.

The result based from natural outer join ordering (3) is also not meaningful to the user. This is due to the fact that C004 appears in two tuples in the integrated result. That is, the tuples (C004, ~, ~, ~, BANGALORE) and (C004, HP, D004, ~, MARKETING) make no sense to the user. This is due to the fact that the tuple (C004, BANGALORE) in result R3 has no corresponding match in R1 and therefore D# has null value in the result, obtained after natural outer join is performed between result R1 and R3. This tuple for which D# has null value has no match in the result and hence for a single concept in reality two tuples appears in the relation.

The result from natural outer join ordering (1), which gives meaningful results, is full disjunction as defined by [Rajaraman1996].

The definition of full disjunction as defined by [Rajaraman1996] is the following

Let R = R₁, R₂, ..., Rₙ be relations whose tuples do not have nulls. FD is said to be a full disjunction for R if the following conditions hold

1. No tuple in FD subsumes any other tuple in FD.
2. Tuples of FD comes from connected pieces of R.
3. All the connections are represented
Although [Legaria1994] has discussed the problem of full disjunction, but only for non-natural outer joins. [Rajaraman1996] has studied this problem for natural-outer joins.

As can be seen from the above example not all orderings of natural outer join leads to full disjunction. Therefore, we need to define an appropriate natural outer join ordering strategy for the results to be integrated in order to achieve the full disjunction. We propose a 'connectedness' criterion [Parimala2002b] for ordering results for integration. This criterion defines the order in which the results need to be integrated in order to be meaningful to the user. The 'connectedness' criterion is based on two principles 'Commonality' and 'Variance', which are discussed next.

5.2.1 The Two Principles

The 'connectedness' is defined based on two concepts namely 'Commonality' and 'Variance'. Two results, which are more 'connected', are integrated before those which are less 'connected'. These are defined below

Commonality is a measure of the number of similar attributes between a pair of results. It is defined as follows

Let \( R_i \) and \( R_j \) be two results of a user query which are to be integrated. Let \( A_{i1}, A_{i2}, \ldots, A_{im} \) be the attributes of result \( R_i \) and \( B_{i1}, B_{i2}, \ldots, B_{in} \) be the attributes of result \( R_j \). Let \( C_{ij} \) be the commonality between them. Then

\[
C_{ij} = k \text{ where } k \text{ is the number of common attributes between } R_i \text{ and } R_j.
\]
Variance is the measure of the number of dissimilar attributes between a pair of results. If \( R_i \) and \( R_j \) are defined as above then variance \( V_{ij} \) between \( R_i \) and \( R_j \) is defined as

\[
V_{ij} = m + n - 2k
\]

Suppose the following results are to be integrated.

\( R_i(A, B), R_2(A, C, E), R_3(A, C, E, D), R_4(C, E) \)

The Commonality and Variance for the above results are shown in Figure-8

<table>
<thead>
<tr>
<th>Pair of Results</th>
<th>Commonality</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R_i(A, B) ) &amp; ( R_2(A, C, E) )</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>( R_i(A, B) ) &amp; ( R_3(A, C, E, D) )</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>( R_i(A, B) ) &amp; ( R_4(C, E) )</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

Figure-8- Commonality and Variance

It must be noted that the same commonality can have different variance and vice versa.

Using the concepts of ‘Commonality’ and ‘Variance’, an appropriate ordering of results for intra-query and inter-query result integration is defined. The result ordering for integration is discussed next.

**5.2.2 Result Integration Ordering**

The pair of results, which have maximum number of attributes in common, are considered more ‘connected’ and hence are integrated before others. This is due to the fact that more attributes will participate in the join condition and therefore more
matching tuples can be found between the pair of results. Within the pairs of results that have the same number of attributes in common, the ones which have the minimum number of differing attributes, are preferred over the others. This is due to the fact that if the number of differing attributes is less, the size of the intermediate result will be small. The smaller the size of the intermediate result, the lesser the amount of storage required.

The ordering of the results is done in the following manner

1. First integrate those pairs of results which have maximum Commonality

2. Within pairs of results having the same maximum Commonality, integrate that pair having the minimum Variance.

The pair of results, which have maximum Commonality and minimum variance, is considered the most 'connected' result.

The algorithm that integrates results with maximum Commonality first and within these the ones which have minimum Variance is discussed next.

5.2.3 Algorithm

Let \( R_1, R_2 \ldots R_n \) be the different results that have to be integrated. Compute \( C_{ij} \) and \( V_{ij} \) for every pair of results

Step-1 : If the number of results to be integrated are two i.e. \( n=2 \) then go to Step-4
Step-2: If the number of results to be integrated are more than two i.e. \( n > 2 \) then consider the pairs of results which have the maximum commonality. If there is only one such pair then go to Step-4.

Step-3: If there are more than one pair of results having maximum commonality, then consider the pairs which have minimum variance. If there is only one such pair go to Step-4 else choose any one of them.

Step-4: Let the pair of results chosen for integration be \( R_i \) and \( R_j \). If the results are union compatible then take a Union. If not perform the Natural Outer Join between the pair of results. The results are integrated and let the integrated result be \( \text{IR}_{ij} \). The set of results left for integration is

\[
\{ R_i, R_j, \ldots, R_n \} - \{ R_i, R_j \} + \text{IR}_{ij}
\]

If the number of results to be integrated is more than one then go to Step-1 else Stop.

Using this algorithm, a natural outer join ordering for results is defined that leads to full disjunction. The proof is discussed next.

**5.2.4 Proof for Full Disjunction**

The integrated results can be represented as an hypergraph\[\text{Fagin1982}\] \[\text{Ullman1989}\][\text{Bernstein1981}\]. The attributes of the result are represented as the hypernodes and the results are represented as hyperedges in the hypergraph. The hypergraph for the example discussed in Section 5.2.1 is shown in Figure-9.
It has been shown that if the set of relational schemes forms a connected $\gamma$ acyclic hypergraph then there exists a natural outer join ordering that leads to full disjunction [Rajaraman1996]. An hypergraph is connected if we can partition its edges into two non-empty sets such that there exists nodes that appear in members of both sets. A hypergraph is $\gamma$ acyclic if and only if it has no pure cycle and $\gamma$ - 3-cycle[Rajaraman1996] otherwise it is $\gamma$-cyclic.

In order to show that the ordering based on our algorithm leads to full disjunction, we make the use of the proof [Rajaraman1996 ] for a sound natural outer join ordering. We show that the order produced by the proposed algorithm discussed above is one of the orders produced by the algorithm in [Rajaraman1996]. In addition we will show that our algorithm reduces the size of the intermediate results.

In [Rajaraman1996] a sound natural outer join ordering strategy has been discussed that leads to full disjunction. Here, a Bachman Diagram is constructed for the hypergraphs. This Bachman Diagram for the hypergraph (BD(H)) is then decomposed based on a defined strategy to arrive at the natural outer join ordering of relations. The order of decomposition is the order of the natural outer join that leads to full disjunction. While decomposing, the node that is picked up is the one
which is minimal i.e no proper subset of this node is a node in the Bachman Diagram. The BD(H) and the corresponding hypergraph gets decomposed by removing this node and in the sound outer join order an outer join using this minimal node is performed last. If there are two hypergraphs then the common node in the corresponding BD(H) denotes the commonality between these hypergraphs. Therefore, the hypergraphs, which have minimum commonality, are to be joined last. This is exactly the situation in our proposed algorithm. The first outer join in the algorithm proposed by [Rajaraman1996], is made on the node, which is used for the decomposition last. This when translated in terms of our proposed algorithm implies that maximum commonality should be joined first. This is exactly the situation in our proposed algorithm. Thus the natural outer join based on our algorithm leads to full disjunction.

The order produced by our algorithm takes into account not only the commonality but also the variance. In [Rajaraman1996], if there are two minimal nodes which can be used in the decomposition, then any one can be picked up. In our algorithm, the corresponding case is the one where the commonality between the hypergraphs is the same. Here, we have proposed the one with minimum variance i.e the nodes in which the two corresponding hypergraphs differ is minimum, should be considered first. This leads to reduction in the size of the intermediate results. We consider an example to demonstrate this aspect.

Consider three results R₁, R₂ and R₃. Let the commonality between these be equal i.e C₁₂ = C₁₃ = C₂₃. Let the variance between results R₁ and R₂ be less than the variance between results R₁ and R₃ i.e. V₁₂ < V₁₃. Let p, q and r be the number of attributes in
R₁, R₂ and R₃ respectively. Let the number of attributes in common between these pair of results be k as the commonality is the same. Then, the number of attributes in the integrated result IR₁₂ = p + q - k and the number of attributes in the integrated result IR₁₃ = p + r - k. Since V₁₂ < V₁₃, the number of attributes in IR₁₂ < IR₁₃. That is, the total number of columns at the end of the natural outer join in the result is less in the former case. This leads to reduction in the size of the intermediate results.

As shown above, the algorithm proposed ensures full disjunction only in the case when the set of results, for integration, form a connected γ-acyclic hypergraph.

5.3 An Example

Consider the following user query

Select Company Department

Where Cloc = ‘Bangalore’ OR Dname = ‘Computer’

There are seven non-redundant results for the above user query. These seven results are shown in Figure-10

<table>
<thead>
<tr>
<th>Results</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>D# Dname Company Cloc</td>
</tr>
<tr>
<td>R2</td>
<td>Company Department</td>
</tr>
<tr>
<td>R3</td>
<td>C# Cname Cloc D# Dname</td>
</tr>
<tr>
<td>R4</td>
<td>C# Cname Cloc D# Dname</td>
</tr>
<tr>
<td>R5</td>
<td>C# Cname Cloc Department</td>
</tr>
<tr>
<td>R6</td>
<td>Company Department</td>
</tr>
<tr>
<td>R7</td>
<td>C# Cname Cloc Department</td>
</tr>
</tbody>
</table>

Figure-10 – Results for the user query
Suppose the user chooses all the results for integration purposes. These results need to be integrated in a defined order based on ‘connectedness’ of results.

From Figure-10, it can be seen that the result R3 and R4 have maximum number of attributes in common between them and hence are considered the most ‘connected’ among the results chosen.

The commonality between R3 and R4 is

$$K = \text{Number of similar attributes in R3 and R4} = 5$$

The variance between R3 and R4 is

$$V = m + n - 2k = (5 + 5) - 2 \times 5 = 0$$

The integrated result IR34, obtained after integration of result R3 and R4, will have the following attributes

IR34 - C# Cname Cloc D# Dname

The results left for integration after integrating R3 and R4 are shown in Figure-11

<table>
<thead>
<tr>
<th>Results</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>D# Dname Company Cloc</td>
</tr>
<tr>
<td>R2</td>
<td>Company Department</td>
</tr>
<tr>
<td>R5</td>
<td>C# Cname Cloc Department</td>
</tr>
<tr>
<td>R6</td>
<td>Company Department</td>
</tr>
<tr>
<td>R7</td>
<td>C# Cname Cloc Department</td>
</tr>
<tr>
<td>IR34</td>
<td>C# Cname Cloc D# Dname</td>
</tr>
</tbody>
</table>

Figure-11 – Results after Integrating R3 & R4

From Figure-11 we find that the results R5 and R7 are the most ‘connected’ as they have the maximum number of attributes similar and hence maximum commonality.
i.e 4 and therefore are integrated. The integrated result IR57 will have the following attributes

IR57 - C# Cname Cloc Department

The results left for integration after integrating R5 and R7 are shown in Figure-12

<table>
<thead>
<tr>
<th>Results</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>D# Dname Company Cloc</td>
</tr>
<tr>
<td>R2</td>
<td>Company Department</td>
</tr>
<tr>
<td>R6</td>
<td>Company Department</td>
</tr>
<tr>
<td>IR34</td>
<td>C# Cname Cloc D# Dname</td>
</tr>
<tr>
<td>IR57</td>
<td>C# Cname Cloc Department</td>
</tr>
</tbody>
</table>

Figure-12 – Results after integrating R5 & R7

From Figure-12, we find that the results R1 and IR34 have the same commonality as the results IR34 and IR57. The commonality in both the case is 3. We now find the variance in both the cases

Variance between R1 and IR34 is

\[ V = m + n - 2k = 5 + 4 - 2\times 3 = 3 \]

Variance between IR34 and IR57 is

\[ V = m + n - 2k = 5 + 4 - 2\times 3 = 3 \]

Since the Variance is also same, we can choose either of the pairs for integration. Suppose we choose IR34 and IR57 then the integrated result IR3457 will have the following attributes

IR3457 - C3 Cname Cloc D# Dname Department

The results left for integration after integrating IR34 and IR57 are shown in Figure-13
Similarly, we can integrate the remaining results based on our ordering criterion. The final integrated result is shown in Figure-14.

The natural outer join ordering, for the results in the above example, based on the proposed algorithm is

\[((((R3 ▷ R4) ▷ (R5 ▷ R7)) ▷ R1) ▷ (R2 ▷ R6)))\]

The pictorial representation of the integration process for the above example is shown in Figure-15.