ISSN: 1935-8156 Volume 16, Number 1, 2021, pp. 60–104 DOI: 10.3194/1935-8156-16.1.60

Teaching Data Joins: A Conceptual Approach Using SQL, Alteryx, and Tableau

Jie Li
Indiana University Bloomington, jieli@indiana.edu

Lorraine Lee
University of North Carolina Wilmington, leel@uncw.edu

Abstract

The many ways of joining data can be confusing for beginning, or even intermediate, data analytics students. Several accounting information system textbooks cover SQL and joins, but few supplemental resources exist that can reinforce and clarify join concepts. In this paper, we use a multi-tool approach in a three-part assignment for teaching data joins using three different software tools. We use four different join types (INNER JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN, and FULL OUTER JOIN) in Microsoft Access, Alteryx, and Tableau. We further demonstrate how the Tableau data model, using separate logical and physical layers, implements the various join types depending on the data needs of a particular visualization. Overall, thus three-part assignment can help students develop a comprehensive understanding of joins and how to implement them in widely-used software tools. The assignment also gives students practice in selecting, using, and validating joins in data analytics.

Keywords
Data joins, databases, SQL, Alteryx, Tableau, contextual join

Practice-focused organizations, such as the American Institute of Certified Public Accountants (AICPA), the National Association of State Boards of Accountancy (NASBA), and the Institute of Management Accountants (IMA) have noted the increasing importance of database skills in their recent frameworks. For example, the CPA evolution initiative (AICPA & NASBA, 2021) identifies the "ability to describe the key characteristics of a relational database such as database tables, records, fields, relationships, and queries" as a learning objective. The IMA (2019) includes "technology and analytics" as one of six relevant knowledge domains for accounting professionals. In the IMA framework, expected "information systems knowledge" specifically includes the elements and concepts of relational databases and the design of relational database tables (IMA, 2019).

Database management and data analytics extensively use queries written in structured query language (SQL) to manipulate the data storage structure and to access data. SQL is the primary language for interacting with traditional relational database management systems (RDBMS), such as SQL Server and MySQL. Big data storage systems (such as Hive, Spark, and Impala) use SQL to retrieve data from Hadoop and similar distributed file systems. Therefore, SQL remains a foundational skill for database and data analytics.

Despite the importance of databases and SQL queries in accounting curricula, few educational resources exist to support accounting instructors teaching these topics. Although textbooks such as Romney and Steinbart (2017) and Richardson et al. (2021) introduce SQL basics, few supplemental resources for teaching SQL exist. Resources that help students understand queries that join data from multiple tables are particularly helpful (Lee & Casterella 2021; Renaud and van Biljon, 2004).

To address this need, we provide a three-part assignment that can help students understand data joins better. We use the visual interfaces of Alteryx and Tableau, two widely-used data analytics tools, to provide a visual representation of the data joins. The assignment's first part explores the differences in join types using SQL examples in Microsoft Access. The second part has students use the JOIN and UNION tools in Alteryx to examine visually the results of various joins. In the third part, students learn how joins are implemented in the physical and logical layers of Tableau's data model. Through this assignment, students learn to solve similar problems using different tools and perspectives. We also include additional review questions to provide practice with joins, validate results, and to help students learn how to translate their questions into queries with appropriate joins. These review questions help students who might struggle with the concepts.

The next section includes a review of the literature on how databases and queries have been taught in accounting and business and identifies problems related to teaching joins. This review also summarizes how joins are discussed in current accounting information systems (AIS) textbooks and in non-accounting courses (including information systems and computer science) and explains why joins are complex, which makes them so difficult for students to learn and apply. The literature review concludes with notes on how data analytics software default settings can further complicate the learning and use of joins.

Literature Review

Coverage of database and SQL topics in accounting courses is usually broad and high-level, focusing on the practical application and benefits of using databases and queries (e.g. Alzubaidi, 2015), providing examples of specific accounting reports that can be generated using queries (Hall 2011; Heagy & Lehmann 2011), and discussing the advantages of using queries for decision-making (Romney & Steinbart, 2017; Gelinas et al., 2017). Other accounting education papers have explained how to use Query By Example (QBE) features, design grids, and query builders to generate queries easily (Bodnar & Hopwood 2013; Chang & Ingraham 2012; Perry & Newmark, 2012), and have provided more technical introduction of query elements such as functions, expressions and operators (Gaskin et al., 2011; Owen, 2012). Teaching cases in the AIS education literature cover the use of SQL queries to detect data anomalies (Lambert et al., 2016), clean dirty data before analysis (Lawson & Street, 2021), and to create a data source for Tableau visualizations (Igou & Coe, 2016).

AIS Textbook Coverage of Joins

Databases are covered in many AIS textbooks. For example, Romney and Steinbart (2017) devote four chapters to relational databases, query-building in Microsoft Access, the resources-events-agents (REA) data model, and both conceptual and logical data modeling. Richardson et al. (2021) has chapters devoted to data modeling and relational databases, and also uses Microsoft Access to introduce SQL.

Information Systems and Computer Science Coverage of Joins

Database and SQL topics are also taught in information systems and computer science courses. Matos et al. (2006), note that SQL is not easy to learn, despite having fewer statements than conventional programming languages. Some aspects of SQL are especially challenging, such as outer joins (Coleman 2003; Date 1986). Matos et al. (2006) formulate three different SQL statements for LEFT OUTER JOINS and evaluated students' responses to each alternative, finding students have difficulty understanding such joins, despite the conceptual scaffolding they set up. Matos et al. (2006) suggest teaching the LEFT JOIN syntax by demonstrating it with multiple datasets, but do not propose any specific techniques for improving student learning of outer joins.

Background Information About Joins

Over 90% of queries composed by experienced SQL users require two or more tables and thus need a join, a SQL operation used to combine columns from different relational database tables to produce a new table (Smelcer, 1995). A join is a horizontal merge of the tables. Joins can be nested so that multiple tables can be combined by successively joining each result with additional tables. Another way to combine data in two tables is to perform a union operation, which is a vertical concatenation of rows from two tables. A union requires that the tables have identical columns, which limits the use of union operations.

Most relational databases today implement three types of joins: cross joins, inner joins, and outer joins (Garcia-Molina et al., 2008). A cross join returns the Cartesian product of all rows from both tables; that is, the combination of each row in one table with every row in the other table. The total number of rows generated is the product of the numbers of rows in the input tables. Although the cross join is the easiest type of join to understand, it has few practical uses. The other two types (inner joins and outer joins) are used frequently in practice. Students must understand the nuances of inner joins and the variations of outer joins to interpret the returned results and to select the correct type of join to use in a given situation.

This assignment deals with inner joins and outer joins only. It also deals with joins of two tables only, with one column from the input table used as the key column on which to join the tables.

Conceptual Complexity of Joins

Joins can have multiple layers of complexity. First, both the columns and rows of the two input tables must be combined to form a result table. However, the number of rows created depends on whether the relationship between the two tables is one-to-one or one-to-many. If the two tables have a one-to-many relationship, the matching rows of one table must be duplicated to be combined with rows in the other. Second, there can be unmatched rows in the input tables. These unmatched rows are not included in the result of inner joins but are included in outer joins. LEFT, RIGHT and FULL OUTER JOINS are differentiated by which set of unmatched rows are included in the result table (in this paper, we use the terms LEFT OUTER JOIN and LEFT JOIN interchangeably, and the terms RIGHT OUTER JOIN and RIGHT JOIN interchangeably).

Null Values

If null values exist in the matching column or columns of one or both input tables, the rows containing the null values in one table do not have corresponding rows (in the other table) because a null value is not equal to any other value, including itself. Therefore, the type of join will determine whether the rows containing null values are returned or not.

Unfortunately, no matter what type of join is used, a result set is always returned. That is, the software tools used to join tables will often not signal errors when an incorrect join type is used, and errors associated with incorrect join types often go unnoticed. In the past, most data retrievals were done by data professionals who had rigorous training in join design and structure. Today, as data analytics becomes democratized, organizations need all employees (including accountants) to understand the subtleties of using joins in data queries (Marr, 2016).

Software Tools for Data Analysis: A Caveat

Software tools such as Alteryx and Tableau are designed to help more people in an organization do data analysis; however, such tools can exacerbate the problems associated with incorrect join types. These software tools extend the accessibility of data analytics with easy drag-and-drop and visual representations of both the process and results. However, a potential unintended effect arises when obtaining an answer becomes too easy. The potential reliance on and passive acceptance of a software tool's default join setting can lead to data analysis errors.

Although most software tools' join functions default to inner joins, there are scenarios in which an inner join does not generate the intended results. For example, users might not understand that some rows in one or both input tables will not be included in an inner join result. When students still learning about joins extract data using multiple join operations, they are often several steps removed from the underlying source data and can easily misinterpret or

misunderstand the results. Thus, students need a clear understanding of join types and how each join is implemented in the software tools they use for data analysis or visualization.

Tableau ameliorates some of these issues by adding a logical layer into its data model and incorporating the join type decision into the tool itself by implementing what it calls a "contextual join." In a contextual join, Tableau chooses the join based on the specific data fields included in a visualization. Thus, a user must understand the join type choices made by Tableau and which rows are matched (or not matched) in the visualization.

Learning Objectives

From the issues of join complexity identified and discussed in our literature review, we develop a three-part assignment that can help students learn complex join concepts using a simple dataset and SQL in Microsoft Access (the first part, shown in Appendix A), apply that knowledge using Alteryx (the second part, shown in Appendix B) and also apply that knowledge in Tableau (the third part, shown in Appendix C). By working through this three-part assignment set, students will develop the knowledge and skills described below.

Part 1: Demonstrate Conceptual Understanding of Join Types Using Microsoft Access SQL

Students follow provided examples of join types to implement similar joins using SQL commands in Microsoft Access (see Appendix A) and learn to do the following:

1. Explain the concept and purpose of SQL joins.
2. Create SQL query scripts to implement INNER JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN, and FULL OUTER JOIN types.
3. Identify and use primary keys and foreign keys correctly in join operations.
4. Use appropriate joins to formulate queries of data in multiple tables.

Part 2: Demonstrate Correct Use of Joins in Alteryx

Students follow join examples and instructions in Alteryx (see Appendix B) and learn to do the following:

1. Describe Alteryx tools and interface features for working with joins (including the JOIN tool and the UNION tool).
2. Create Alteryx workflows that implement INNER JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN, and FULL OUTER JOIN (using the UNION tool) types.
3. Practice results validation by examining output streams.
4. Compare and contrast Alteryx joins with SQL joins.
5. Create queries that include joins using Alteryx.

Part 3: Use the Tableau Data Model to Create Visualizations Based on Joins

Students follow examples and instructions in Tableau (see Appendix C) and learn to do the following:

1. Implement joins in Tableau's physical and logical model layers correctly.
2. Demonstrate how join type choices in the logical model layer affect visualizations.
3. Use Left-only Rows and Right-only Rows options in Tableau correctly.
4. Demonstrate correct use of fields to trigger various join types in Tableau.

Implementation Guidance

The assignment is suitable for use in AIS and accounting analytics classes that include database design and SQL. Many introductory AIS courses assume no prior database or SQL knowledge; the assignment can help reinforce join basics in those courses. Parts 2 and 3 of the assignment can help instructors introduce the use of joins in software tools like Alteryx and Tableau in either accounting analytics courses or AIS courses at any level. More advanced accounting analytics courses can use these assignments to help students review and refresh their understanding data join concepts (Part 1) as they learn data retrieval and integration topics (in Parts 2 and 3).

Software Requirements

Students will need Microsoft Access for Part 1, Alteryx for Part 2, and Tableau for Part 3 of the assignment. Instructors that use a relational database management system other than Microsoft Access could use it for Part 1. All three of these software tools can run on individual student laptops or on computers in an instructional lab.

Prerequisite Knowledge

The assignment assumes that students have some knowledge of basic relational database theory and SQL, topics often covered in introductory AIS classes. For example, the Richardson et al. (2021) and Romney et al. (2020) AIS

textbooks each include chapters on relational database concepts and SQL, topics that this assignment supplements and reinforces. Instructors or students that want to review database and SQL concepts before using the assignment can use the appendix of Richardson et al. (2021) or online data analytics training platforms such as Coursera (n.d.) or Kubicle (n.d.). Specifically, students should be familiar with relational database principles, including primary keys, foreign keys, one-to-one relationships, one-to-many-relationships, and querying terms (such as "select," "from," and "where").

Assignment Administration

Supplementary materials for the assignment include student handouts in Microsoft Word format (which are editable versions of the three-part assignment that appears in Appendices A, B, and C), instructor teaching notes, and copies of the database tables used in the assignment. Although full solutions are also available to adopting instructors, we do not recommend grading each question individually. Instead, we suggest that the instructor award points for overall completion of the assignments and then review the solutions in class to further reinforce student learning. The concept-based multiple choice questions in Part 1 can be used to assess student performance.

The entire assignment (including Parts 1, 2, and 3) can be covered in a two week period in an AIS or accounting analytics course. We recommend administering the assignment as shown in Table 1.

Table 1

Schedule for Classroom Use of the Assignment

Class period	Time (minutes)	Prerequisite knowledge	Assignment	Activities
1	75	Basic database concepts and basic SQL	Part 1: Join Concepts	Part 1 Problems 1-3: work in class Part 1 Problems 4-6: assign as homework
2	30	Basic database concepts and basic SQL	Part 1: Join Concepts	Review solutions to Part 1: Problems 4-6
2	45	Basic Alteryx skills	Part 2: Alteryx	Part 2 Alteryx Problems 1-2: work in class Part 2 Problems 3-6: assign as homework
3	45	Basic Alteryx skills	Part 2: Alteryx	Review Alteryx solutions to Part 2 Problems 3-6
4	45 - 75	Basic Tableau skills	Part 3: Tableau	Introduce Tableau data model and joins at the logical and physical layers (Part 3 Problems 1 and 2)
5	45 - 75	Basic Tableau skills	Part 3: Tableau	Part 3 Problems 3 and 4: Either assign as homework before class and review the solutions in class or work through Problems 3 and 4 together in class

During the first 75-minute class, the instructor can review database and SQL concepts, then work Problems 1 through 3 together with the class. After students complete Problems 4 through 6 on their own, the instructor can review those problems as part of the next class. Instructors that have only one week to devote to these topics can complete Part 1 of the assignment in that time. Parts 2 and 3 of the assignment allow the instructor to introduce two additional software tools and help students build on their knowledge of databases and SQL to accomplish data analytics and visualization tasks.

Efficacy

We used Part 1 of the assignment in a graduate-level accounting database class with 25 students and surveyed them about their experience in using it. We asked them to provide their agreement with six statements about what they might have learned by doing the assignment. Their ratings are summarized in Table 2.

Table 2
Student Agreement with Learning Statements After Completing Part 1 (n = 25)

Statement	*Mean*[a]
The data join assignments helped me understand	
… SQL better	6.44
… the overall relationship among data	6.32
… how tables are joined together	6.40
… the different types of joins	6.36
… the difference between a left join and a right join	6.52
… understand the row results returned from the various join types	6.40

[a]Scale: 1=strongly disagree to 7=strongly agree with the statements

The student feedback was overwhelmingly positive, with each question averaging greater than 6.32 (on a seven-point Likert scale). Our administration of Part 1 did teach us that one 75-minute session was insufficient to complete the activities and led us to recommend the split noted in Table 1. We have not used Parts 2 or 3 in a classroom yet; however, we did have a graduate assistant (who had completed an accounting database course) work through Parts 2 and 3 and provide feedback. We also had a faculty member with more than 10 years of database and SQL teaching experience review the assignment and provide feedback that focused on Parts 2 and 3. We have incorporated all of this feedback into the assignment. Thus, we believe that the assignment can help other accounting instructors teach this material.

Concluding Thoughts

Joins are an important operation in accounting database applications that practicing accountants use. Understanding join nuances is crucial to obtaining correct source data, which is the starting point of data analysis. The complexity of joins and the often limited time devoted to learning about them can lead less-experienced accountants to misinterpret data analytics results. We believe this assignment can unravel layers of join complexity and help students understand the differences between join types. We demonstrate the visual features of Alteryx that can clarify the differences between join types and we introduce the Tableau data model, explaining how it makes a contextual join decision as a visualization is built. The assignment helps students develop a thorough conceptual understanding of joins they can use to create joins in SQL, Alteryx, and Tableau.

References

American Institute of Certified Public Accountants & National Association of State Boards of Accountancy (2021). CPA evolution. https://www.evolutionofcpa.org

Alzubaidi, R. (2015). Query language application in accounting database, *Proceedings for the Northeast Region Decision Sciences Institute (NEDSI)*. 1–12.

Bodnar, G. H., & Hopwood, W. S. (2013). *Accounting information systems* (11th ed.). Pearson.

Chang, C., & Ingraham, L. R. (2012). *Modeling and designing accounting systems using Access to build a database* (2nd Ed.). Wiley.

Coleman, J. (2003). *Online assessment of SQL query formulation skills*, Australasian Computing Education Conference, Adelaide, Australia.

Coursera (n.d.). https://www.coursera.org/

Date, C.J. (1986). A critique of the SQL database language, *Relational database: Selected writings*, Addison-Wesley Longman.

Garcia-Molina, H., Ullman, J. D., & Widom, J. (2008). *Database systems: The complete book*. (2nd Ed.). Pearson.

Gaskin, S., McLellan C., & Graviett, N. (2011), *Access 2010 comprehensive*, Prentice Hall.

Gelinas, U. J., Dull, R. B., Wheeler, P., & Hill, M. (2017). *Accounting information systems* (11th ed.). Cengage Learning.

Hall, J. (2011). *Accounting information systems* (7th ed.). Cengage Learning.

Heagy, C., & Lehmann, C. M. (2011). *Accounting information systems: A practitioner emphasis* (7th ed.). Cengage Learning.

Igou, A., & Coe. M. (2016). Vistabeans coffee shop data analytics teaching case, *Journal of Accounting Education*, *36*, 75–86. https://doi.org/10.1016/j.jaccedu.2016.05.004

Institute of Management Accountants (2019). *IMA management accounting competency framework.* https://www.imanet.org/-/media/590889ef44ad401bb94d83cd43e584b8.ashx

Kubicle (n.d.). Building a community of data citizens.

Lambert, S. L, Holladay, J., & Drum, D. M. (2016, Fall). International AC: An education case on continuous monitoring SQL Server data with ODBC-Linked Tables in Microsoft Access, *Journal of Emerging Technologies in Accounting*, *13*(2), 195–213. https://doi.org/10.2308/jeta-51597

Lawson, J.G., & Street, D. A. (2021, June). Detecting dirty data using SQL: Rigorous house insurance case, *Journal of Accounting Education, 55*, 100714. https://doi.org/10.1016/j.jaccedu.2021.100714

Lee, L., & Casterella, G. (2021). A mental model approach to teaching database querying skills with SQL and Alteryx [Manuscript submitted for publication], Department of Accountancy and Business Law, University of North Carolina Wilmington.

Marr, B. (2016). *Big data in practice: How 45 successful companies used big data analytics to deliver extraordinary results*. Wiley. https://doi.org/10.1002/9781119278825

Matos, V. M., Grasser, R., & Jalics, P. (2006). The case of the missing tuple: Teaching the SQL outer-join operator to undergraduate information systems students. *Journal of Computing Sciences in Colleges 22*(1), 23–32.

Owen, G. (2012). *Using Excel & Access 2013 for accounting*, Cengage Learning.

Perry, J., & Newmark, R. (2012). *Building accounting systems using Access 2010*. (8th ed.). Cengage Learning.

Richardson, V., Chang, C., & Smith, R. (2021). *Accounting information systems.* (3rd ed.). McGraw-Hill.

Richardson, V. J., Teeter, R. A., & Terrell, K. L. (2021). *Data analytics for accounting*. McGraw-Hill.

Romney, M., & Steinbart, P. J. (2017). *Accounting information systems*. (14th ed.). Pearson.

Renaud, K., & van Biljon, J. (2004). Teaching SQL - Which pedagogical horse for this course? *Volume 3112 of Lecture Notes in Computer Science*, 244–256. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-27811-5_22

Smelcer, J. B. (1995). User errors in database query composition. *International Journal of Human-Computer Studies*, *42*(4), 353-381. https://doi.org/10.1006/ijhc.1995.1017

Appendix A

Part 1: Conceptual Understanding of Joins with SQL

Learning Objectives

1. Explain the concept and purpose of SQL joins.
2. Create SQL query scripts to implement INNER JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN, and FULL OUTER JOIN types.
3. Identify and use primary keys and foreign keys correctly in join operations.
4. Use appropriate joins to formulate queries of data in multiple tables.

Overview

This assignment includes six problems that lead students through a series of joins and concludes with a set of review questions as follows:

- Problem 1: INNER JOIN of the *Customer* Table and the *Customer_Extra* Table
- Problem 2: Joining the *Customer* Table with the *Order* Table
- Problem 3: Joining the *Order* Table with the *Customer* Table
- Problem 4: Joining the *Order* Table with the *Employee* Table
- Problem 5: Joining the *Employee* Table with the *Order* Table
- Problem 6: Sample Queries
- Review Questions

The assignment uses data about customers, employees, and orders stored in four tables (*Customer*, *Customer_Extra*, *Order*, and *Employee*), as shown in Figure A1:

Figure A1

Part 1 data tables

Order

OrderID	CustomerID	EmployeeID	RECEIVED	SHIPPED
1020	1111	1000	12/4/2015	12/15/2015
1021	1111	1000	12/5/2015	12/19/2015
1022	2222	1002	12/6/2015	12/14/2015
1023	3333	1000	12/7/2015	12/18/2015
1024	3333	1003	12/9/2015	12/21/2015
1825	4444		7/20/2016	
1826	5555	1002	7/21/2016	
1827	6666		7/22/2016	

Customer_Extra

CustomerID	Gender	Age
1111	M	46
2222	M	54
3333	F	23
4444	F	20
5555	M	34
6666	M	36
7777	M	60
8888	M	49
9999	M	63

Customer

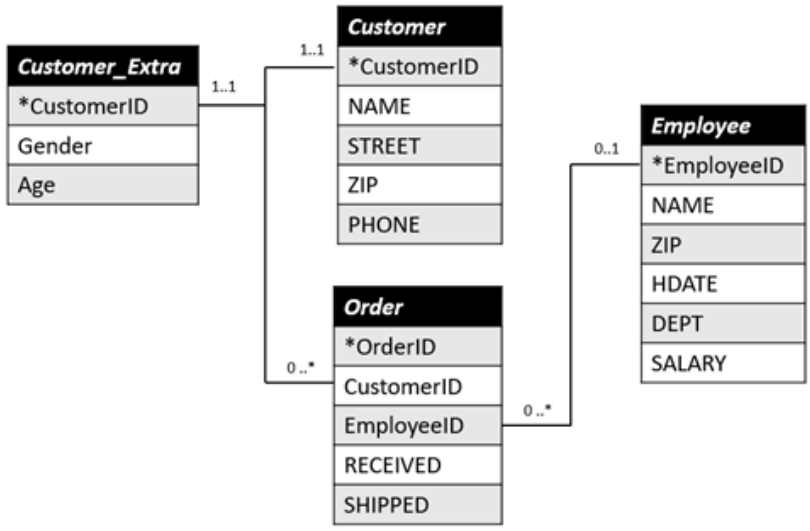
CustomerID	NAME	STREET	ZIP	PHONE
1111	Charles	123 Main St.	67226	316-636-5555
2222	Bertram	237 Ash Avenue	67226	316-689-5555
3333	Barbara	111 Inwood St.	60606	316-111-1234
4444	Bala	119 Indiana St.	60616	317-111-2345
5555	Dennis	333 Outwood Ave.	60606	316-222-1234
6666	Davis	444 Michigan St.	60616	317-333-3456
7777	Scott	777 State St.	60616	317-777-7456
8888	John	888 College Ave.	60606	317-888-7886
9999	Stephen	768 Mall Rd.	60606	317-223-7783

Employee

EmployeeID	NAME	ZIP	HDATE	DEPT	SALARY
1000	Jones	67226	9/1/2012	Sales	75000
1001	Smith	60606	10/15/2012	Marketing	74000
1002	Brown	50302	3/1/2013	Sales	48000
1003	John	50305	6/1/2013	Sales	44000
1004	Rachael	60606	7/15/2013	IT	59000
1005	Sue	67226	8/1/2013	Finance	58000
1006	Fred	67227	1/2/2014	Marketing	47000

The relationships among the tables are shown in the class diagram that appears in Figure A2:

Figure A2
Data class diagram



The data used in this assignment is available as an Excel file from the authors or this journal.

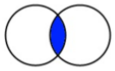
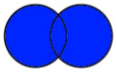
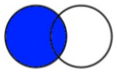
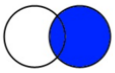
Terms Used in the Assignment

A primary key is a column or group of columns designated to uniquely identify each row in a table. A foreign key is a column or group of columns that provides a link between rows in two tables. The foreign key acts as a cross-reference between tables because it references the primary key of another table, thereby establishing a link between them. Although a primary key can exist on its own, a foreign key must reference the primary key of another table. The original table containing the primary key is the *parent* table (or referenced table), while the key can be referenced by multiple foreign keys from other tables, known as *child* tables. Two tables to be joined usually have a parent-child relationship. The primary key from the parent table and the foreign key from the child table are often the matching columns for join.

A join is a relational database operation that brings columns from two (or more) tables together as one result set based on a matching column (also called a field) in the tables. A comparison operator (usually "=") is used to identify identical values in the matching columns of the tables. In a two-table join, every row of the left table is compared to rows in the right table based on the value of the matching column.

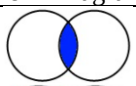
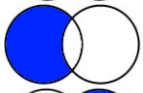
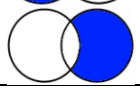
Joins may be an INNER JOIN or one of three types of outer joins (LEFT OUTER JOIN; RIGHT OUTER JOIN; FULL OUTER JOIN). Because the matching columns used in join are often the primary key and foreign key columns, the values in the matching column are often called the key values. Venn diagrams are commonly used to describe the four join types, as detailed in Table A1:

Table A1
Definition of Joins and Venn Diagram Representation

Join Type	Venn Diagram Representation	Definition
INNER JOIN		Returns the rows from both the left table and the right table where the key value of the left table is equal to the key value of the right table.
FULL OUTER JOIN		Returns the matching rows from the left and the right tables, as well as any unmatched rows from the left and right table with NULL values filled in columns where there are no values for the result set.
LEFT OUTER JOIN (LEFT JOIN)		Returns all the rows from the left table where the key value of the rows from the left table equals to the key value of the rows from the right table AND includes the rows from the left table for which there are no matching rows on the right table. The result set will contain NULL values in columns where there are no values for the result set.
RIGHT OUTER JOIN (RIGHT JOIN)		Returns all the rows from the right table where the key value of the rows from the right table equals to the key value of the rows from the left table AND includes the rows from the right table for which there are no matching rows on the left table. The result set will contain NULL values in columns where there are no values for the result set.

Another way to think about joins is to consider the three possible sets of rows in the two tables to be joined as shown in Table A2.

Table A2
Matching Rows, Left-Only Rows, and Right-Only Rows

Row Results	Venn Diagram	Definition
Matching Rows		Rows that have matching values on the key column
Left-Only Rows		Rows on the left table that do not match any rows on the right table
Right-Only Rows		Rows on the right table that do not match any rows on the left table

Problem 1: INNER JOIN of Customer and Customer_Extra

In this first example, you will implement an INNER JOIN of the Customer Table and the Customer_Extra Table. Answer the following questions first:

1. What is the primary key of the *Customer* table?
2. What is the primary key of the *Customer_Extra* table?
3. What column in the *Customer_Extra* table is the foreign key when relating the *Customer* table with the *Customer_Extra* table? Note that *Customer* is the parent table and *Customer_Extra* is the child table.
4. What is the SQL code for joining the *Customer* table with the *Customer_Extra* table?
5. Import the ***Customer*** table and the ***Customer_Extra*** table into a relational database, e.g. MS Access. Create a query based on the answer above. Save this query as "Q1_INNER_JOIN_ Customer_Customer_Extra"

The join result can be seen in Figure A3. Every row in the left table has a matching row in the right and vice versa.

Figure A3

INNER JOIN of Customer Table and Customer_Extra Table

Customer

	Customer. CustomerID	NAME	STREET	ZIP	PHONE
1	1111	Charles	123 Main St.	67226	316-636-5555
2	2222	Bertram	237 Ash Avenue	67226	316-689-5555
3	3333	Barbara	111 Inwood St.	60606	316-111-1234
4	4444	Bala	119 Indiana St.	60616	317-111-2345
5	5555	Dennis	333 Outwood Ave.	60606	316-222-1234
6	6666	Davis	444 Michigan St.	60616	317-333-3456
7	7777	Scott	777 State St.	60616	317-777-7456
8	8888	John	888 College Ave.	60606	317-888-7886
9	9999	Stephen	768 Mall Rd.	60606	317-223-7783

Customer_Extra

Customer_Extra. CustomerID	Gender	Age
1111	M	46
2222	M	54
3333	F	23
4444	F	20
5555	M	34
6666	M	36
7777	M	60
8888	M	49
9999	M	63

Key Takeaways:

- These two tables have a one-to-one relationship meaning every row in the parent table has a corresponding row in the child table and vice versa.
- Therefore, all rows in the parent and child tables have matching key values in the columns to be joined.
- The one-to-one relationship and all matching values ensure that the result is a simple horizontal merge of the two tables.
- The result has the same number of rows as that of the input tables. Note that this is only the case when the one-to-one relationship is mandatory. If the "extra" information is not required for each customer, then the number of rows may be less than the number in the Primary Key table.
- In this join of the *Customer* and the *Customer_Extra* table, INNER JOIN, OUTER JOIN, LEFT JOIN, and RIGHT JOIN will all provide the same results.

Problem 2: Joining the Customer Table with the Order Table

In Problem 2, you will be joining the Customer table with the Order table. The Customer table will be used as the left table and the Order table will be used as the right table. The Customer table has nine rows, the Order table has eight rows (Figure A4). These tables will be useful in illustrating LEFT JOINS and RIGHT JOINS in that there are customers who have not placed orders.

Figure A4

Customer Table and Order Table

Customer

	CustomerID	NAME	STREET	ZIP	PHONE
1	1111	Charles	123 Main St.	67226	316-636-5555
2	2222	Bertram	237 Ash Avenue	67226	316-689-5555
3	3333	Barbara	111 Inwood St.	60606	316-111-1234
4	4444	Bala	119 Indiana St.	60616	317-111-2345
5	5555	Dennis	333 Outwood Ave.	60606	316-222-1234
6	6666	Davis	444 Michigan St.	60616	317-333-3456
7	7777	Scott	777 State St.	60616	317-777-7456
8	8888	John	888 College Ave.	60606	317-888-7886
9	9999	Stephen	768 Mall Rd.	60606	317-223-7783

Order

	OrderID	CustomerID	EmployeeID	RECEIVED	SHIPPED
1	1020	1111	1000	12/4/2015	12/15/2015
2	1021	1111	1000	12/5/2015	12/19/2015
3	1022	2222	1002	12/6/2015	12/14/2015
4	1023	3333	1000	12/7/2015	12/18/2015
5	1024	3333	1003	12/9/2015	12/21/2015
6	1825	4444	Null	7/20/2016	Null
7	1826	5555	1002	7/21/2016	Null
8	1827	6666	Null	7/22/2016	Null

Answer the following questions, which are based on the structure of and the information included in the two tables that appear in Figure A4:

1. What is the primary key of the *Customer* table?
2. What is the primary key of the *Order* table?
3. Which table is the "parent" table?
4. Which table is the "child" table?

5. What is the foreign key in the child table that links the two tables? (The key column in which the matching values will be found)
6. What are the Matching Rows? Using the data provided in the Excel file, copy and paste the matching rows below. In other words, 1) copy all rows in *Customer* table that contain a matching row in the *Order* table; and 2) copy all rows in the *Order* Table that contain a matching row in the *Customer* Table.
7. Are the number of matching rows from the *Customer* Table and *Order* Table the same?
8. If the answer is no, which foreign key values appears multiple times (which specific values for Customer_ID appear multiple times in the *Order* Table)?

   How are matching rows combined to form the result set? The result set has columns from both tables. The foreign key values appearing multiple times makes the merging not as straight-forward as in Problem 1 when the relationship between the parent and child table is one-to-one. The rows identified in above from the parent table that match multiple rows in the child table should be duplicated in the result set.

9. Left-Only Rows: Are there any rows in the *Customer* table (left table) that have no matching rows in the *Order* table (right table)? If so, how many?
10. Right-Only Rows: Are there any rows in the *Order* table (right table) that have no matching rows in the *Customer* table (left table)? If not, why do you think this is the case?

Problem 2.1: INNER JOIN of Customer Table with the Order Table

Answer the following questions for the INNER JOIN of the two tables (Customer table and Order table)

1. Which possible set(s) of rows are included in the result set of an INNER JOIN? Select more than one options if necessary.
   a. Matching Rows
   b. Left-Only Rows
   c. Right-Only Rows
2. What is the SQL code for the INNER JOIN of the *Customer* table with the *Order* table?
3. Import the *Customer* table and the *Order* table into MS Access. Create a query based on the answer above. Save this query as "Q2_1_INNER_JOIN_Customer_Order"
4. How many rows are returned in the SQL join of the Customer table with the Order table?
5. If the numbers of matching rows from the *Customer* table and *Order* table are different, how does the match work? A solution is provided in Figure A5.

Figure A5

Solution to part 5 of Problem 2.1

Customer. CustomerID	NAME	STREET	ZIP	PHONE	OrderID	Order. CustomerID	EmployeeID	RECEIVED	SHIPPED
1111	Charles	123 Main St.	67226	316-636-5555	1020	1111	1000	12/4/2015	12/15/2015
1111	Charles	123 Main St.	67226	316-636-5555	1021	1111	1000	12/5/2015	12/19/2015
2222	Bertram	237 Ash Avenue	67226	316-689-5555	1022	2222	1002	12/6/2015	12/14/2015
3333	Barbara	111 Inwood St.	60606	316-111-1234	1023	3333	1000	12/7/2015	12/18/2015
3333	Barbara	111 Inwood St.	60606	316-111-1234	1024	3333	1003	12/9/2015	12/21/2015
4444	Bala	119 Indiana St.	60616	317-111-2345	1825	4444		7/20/2016	
5555	Dennis	333 Outwood Ave.	60606	316-222-1234	1826	5555	1002	7/21/2016	
6666	Davis	444 Michigan St.	60616	317-333-3456	1827	6666		7/22/2016	

The rows with customerID 1111 and 3333 in the parent table (Customer) are duplicated to match the rows with customerID 1111 and 3333 in the child table (Order). In Figure A5, columns with the white background are from the left table, while the columns with the yellow background color are from the right table.

Key Takeaways:

- The *Customer* table and *Order* table have a one-to-many relationship. A row in the parent table does not necessarily have a corresponding row in the child table. However, one row in the parent table can have more than one corresponding row in the child table. In other words, a customer can have zero or more orders. A row in the parent table is duplicated if multiple rows in the child table have matching foreign keys to its primary key. For example, the rows in *Customer* table with CustomerID 1111 and 3333 belong to this category and are duplicated in the INNER JOIN result.

- The number of matching rows in the INNER JOIN result (8 rows) is larger than the number of matching rows (6 rows) in the parent table (*Customer* Table) but is equal to the matching rows (8 rows) in the child table (*Order* Table).

Problem 2.2: LEFT OUTER JOIN of Customer Table with the Order Table

1. Which possible set(s) of rows are included in the result set of a *LEFT OUTER JOIN* of *Customer* table with *Order* table? Select more than one options if necessary.
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows
2. Left-Only Rows: Examining the data from *Customer* (left) table and *Order* (right) table, which rows from the *Customer* (left) table are NOT associated with rows from the *Order* (right) table? Copy the result below:
3. Write the SQL code for the LEFT OUTER JOIN of *Customer* and *Order* tables
4. Create a query of the LEFT OUTER JOIN between *Customer* and *Order* tables based on the answer above. Save this query as "Q2_2_LEFT_OUTER_JOIN_Customer_Order"
5. How many rows are returned from the LEFT OUTER JOIN query?
6. How many of the rows from the LEFT OUTER JOIN query are associated with Matching Rows?
7. How many of the rows are associated with Left-Only Rows?

Examine the following results (Figure A6) of the Customer table LEFT JOIN the Order table. Columns with yellow background are from the right table (the Order table). Rows with green background are the left-only Rows.

Figure A6

Solution to Problem 2.2

Customer. CustomerID	NAME	STREET	ZIP	PHONE	OrderID	Order. CustomerID	EmployeeID	RECEIVED	SHIPPED
1111	Charles	123 Main St.	67226	316-636-5555	1020	1111	1000	12/4/2015	12/15/2015
1111	Charles	123 Main St.	67226	316-636-5555	1021	1111	1000	12/5/2015	12/19/2015
2222	Bertram	237 Ash Avenue	67226	316-689-5555	1022	2222	1002	12/6/2015	12/14/2015
3333	Barbara	111 Inwood St.	60606	316-111-1234	1023	3333	1000	12/7/2015	12/18/2015
3333	Barbara	111 Inwood St.	60606	316-111-1234	1024	3333	1003	12/9/2015	12/21/2015
4444	Bala	119 Indiana St.	60616	317-111-2345	1825	4444		7/20/2016	
5555	Dennis	333 Outwood Ave.	60606	316-222-1234	1826	5555	1002	7/21/2016	
6666	Davis	444 Michigan St.	60616	317-333-3456	1827	6666		7/22/2016	
7777	Scott	777 State St.	60616	317-777-7456	Null	Null	Null	Null	Null
8888	John	888 College Ave.	60606	317-888-7886	Null	Null	Null	Null	Null
9999	Stephen	768 Mall Rd.	60606	317-223-7783	Null	Null	Null	Null	Null

Key Takeaways:

- When there are unmatched rows from the left table, they will be included in the result of LEFT OUTER JOIN.
- In this example, the result includes all customers, even those customers who are "prospective" and haven't placed any orders yet
- For these rows of prospective customers, the values in the columns from the right table will be NULL
- Since the OUTER JOIN result always includes the rows from the INNER JOIN, the number of rows in the LEFT OUTER JOIN will always be greater than or equal to that of the rows from the INNER JOIN.

Problem 2.3: RIGHT OUTER JOIN of Customer Table with the Order Table

1. Which possible set(s) of rows are included in the result set of a *RIGHT OUTER JOIN* of *Customer* table with *Order* table? Select more than one options if necessary.
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows

2. Right-Only Rows: Examining the data from the *Customer* (left) table and the *Order* (right) table, which rows from the *Order* (right) table are NOT associated with rows from the *Customer* (left) table?
3. Write the SQL code for the RIGHT OUTER JOIN of the Customer and Order tables
4. Create a query of the RIGHT OUTER JOIN between Customer and Order tables based on the answer above. Save this query as "Q2_3_RIGHT_OUTER_JOIN_Customer_Order"
5. How many rows are returned from the RIGHT OUTER JOIN query?
6. How many of the rows from the RIGHT OUTER JOIN query are associated with Matching Rows?
7. How many of the rows are associated with Right-Only Rows?

Key Takeaways:

- When there are unmatched rows from the right table, they will be included in the result of the RIGHT OUTER JOIN.
- For these rows, the values in the columns from the left table will be NULL.
- Since the OUTER JOIN result always includes the rows from the INNER JOIN, the number of rows in the RIGHT OUTER JOIN will be greater than or equal to that of the rows from INNER JOIN.
- In our example above, since there are zero unmatched rows from the right table, the results of the INNER JOIN are equivalent to that of the RIGHT OUTER JOIN.
- There are no unmatched rows because every order must be from an existing customer.

Problem 2.4: FULL OUTER JOIN of Customer Table with the Order Table

1. Which possible set(s) of rows are included in the result set of a *FULL OUTER JOIN* of *Customer* table with *Order* table? Select more than one options if necessary.
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows
2. Write the SQL code for the FULL OUTER JOIN of *Customer* and *Order* tables
3. Create a query of FULL OUTER JOIN between the *Customer* and *Order* tables based on the above. Save this query as "Q2_4_FULL_OUTER_JOIN_Customer_Order"
4. How many rows are returned from the FULL OUTER JOIN?

Key Takeaways:

- A FULL OUTER JOIN is used when both unmatched rows from the left and right tables should be returned, as well as the matching rows.
- If there are only unmatched rows from the left table, the FULL OUTER JOIN result is the same as that of the LEFT OUTER JOIN result. There will be NULL values in the columns of the right table for the unmatched rows.
- If there are only unmatched rows from the right table, FULL OUTER JOIN result is the same as that of the RIGHT OUTER JOIN result. There will be NULL values in the columns of the left table for the unmatched rows.
- If there are unmatched rows from both the left and right tables, the number of rows returned from FULL OUTER JOIN is greater than that of both the LEFT OUTER JOIN and RIGHT OUTER JOIN.

Problem 3: Joining the Order Table with the Customer Table

For this problem, the same two tables are used as in Problem 2. However, the ordering of the tables is switched so that the Order Table is now the left table and the Customer Table is the right table.

1. What are the **Matching Rows**? Using the data provided in the Excel file, copy and paste the matching rows below. In other words, 1) copy all of the rows from *Order* table that contain a matching row from the *Customer* table; and 2) copy all of the rows from the *Customer* Table that contain a matching row from the *Order* Table.
2. Left-Only Rows: Are there any rows in the *Order* table (left table) that have no matching rows in the *Customer* table (right table)? If so, how many?
3. Right-Only Rows: Are there any rows in the *Customer* table (right table) that have no matching rows in the *Order* table (left table)? If so, how many?

Problem 3.1: INNER JOIN of Order Table with the Customer Table

1. Write the SQL code for the INNER JOIN of the *Order* table and the *Customer* table
2. Create a query of INNER JOIN between *Order* and *Customer* tables based on the answer above. Save this query as "Q3_1_INNER JOIN Order_Customer"

3. How many rows are returned from the INNER JOIN?
4. The results of the INNER JOIN from above returns which of the following?
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows
5. Compare the results of this INNER JOIN (INNER JOIN of *Order* and *Customer*) with the results from Problem 2.1 (INNER JOIN of *Customer* and *Order*). Does the order of the joins matter in an INNER JOIN?

Problem 3.2: LEFT OUTER JOIN of Order Table with the Customer Table

1. Compare this problem (Problem 3.2) with Problem 2.2. Do you think the same results will be returned (ignoring the order of columns in the result)?
2. Write the SQL code for the LEFT OUTER JOIN of *Order* and *Customer* tables
3. Create a query of LEFT OUTER JOIN between *Order* and *Customer* tables based on the
4. answer above. Save this query as "Q3_2_LEFT_OUTER_JOIN_Order_Customer"
5. How many rows are returned in the LEFT OUTER JOIN from above?
6. Compare the LEFT OUTER JOIN results from this problem (Problem 3.2) with the LEFT OUTER JOIN from Problem 2.2. Describe any similarities or differences.
7. The results of the LEFT OUTER JOIN from above returns which of the following? Select more than one options if necessary.
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows
8. Is the number of rows returned from the LEFT OUTER JOIN the same or different from the INNER JOIN? Why?

Problem 3.3: RIGHT OUTER JOIN of Order Table with the Customer Table

1. Compare this problem (Problem 3.3) with Problem 2.3. Do you think the same results will be returned (ignoring the order of columns in the result)?
2. Write the SQL code for the RIGHT OUTER JOIN of the *Order* and *Customer* tables.
3. Create a query of RIGHT OUTER JOIN between Order and Customer tables based on the answer above. Save this query as "Q3_3_RIGHT_OUTER_JOIN_Order_Customer"
4. How many rows are returned in the RIGHT OUTER JOIN from above?
5. Compare the SQL RIGHT OUTER JOIN result from Problem 3.3 with Problem 2.3. Describe any similarities or differences.
6. The results of the RIGHT OUTER JOIN from above returns which of the following? Select more than one options if necessary.
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows

Key Takeaway:

- The RIGHT OUTER JOIN result of *Order* to *Customer* table is different from the INNER JOIN result of the *Order* to *Customer* table. This is because there are three customers who have not placed an order. The RIGHT OUTER JOIN results contain the matching rows (eight rows) plus the three right-only rows (customers with no orders) for a total of 11 rows returned. See the result shown in Figure A7. Columns with yellow background color are from the right table. Rows with salmon background color are the right-only rows.

Figure A7
Key Takeaway From Problem 3.3, Illustrated

OrderID	Order. CustomerID	EmployeeID	RECEIVED	SHIPPED	Customer. CustomerID	NAME	STREET	ZIP	PHONE
1020	1111	1000	12/4/2015	12/15/2015	1111	Charles	123 Main St.	67226	316-636-5555
1021	1111	1000	12/5/2015	12/19/2015	1111	Charles	123 Main St.	67226	316-636-5555
1022	2222	1002	12/6/2015	12/14/2015	2222	Bertram	237 Ash Avenue	67226	316-689-5555
1023	3333	1000	12/7/2015	12/18/2015	3333	Barbara	111 Inwood St.	60606	316-111-1234
1024	3333	1003	12/9/2015	12/21/2015	3333	Barbara	111 Inwood St.	60606	316-111-1234
1825	4444	Null	7/20/2016	Null	4444	Bala	119 Indiana St.	60616	317-111-2345
1826	5555	1002	7/21/2016	Null	5555	Dennis	333 Outwood Ave.	60606	316-222-1234
1827	6666	Null	7/22/2016	Null	6666	Davis	444 Michigan St.	60616	317-333-3456
Null	Null	Null	Null	Null	7777	Scott	777 State St.	60616	317-777-7456
Null	Null	Null	Null	Null	8888	John	888 College Ave.	60606	317-888-7886
Null	Null	Null	Null	Null	9999	Stephen	768 Mall Rd.	60606	317-223-7783

Problem 3.4: FULL OUTER JOIN of Order Table with the Customer Table

1. Compare this problem with the FULL OUTER JOIN in Problem 2.4. Do you think the same results will be returned (ignoring the order of columns in the result)?
2. Write the SQL code for the FULL OUTER JOIN of *Order* and *Customer* tables
3. Create a query of FULL OUTER JOIN between *Order* and *Customer* tables based on the answer above. Save this query as "Q3_4_FULL_OUTER_JOIN_Order_Customer"
4. How many rows are returned in the FULL OUTER JOIN?
5. Compare the SQL FULL OUTER JOIN result from Problem 3.4 with Problem 2.4. Are there any differences? If so, please elaborate.
6. The results of the FULL OUTER JOIN from above returns which of the following? Select more than one options if necessary.
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows

Key Takeaways:

- The INNER JOIN and FULL OUTER JOIN results remain the same regardless of the ordering of the table, i.e. which table is on the left versus on the right.
- The left and RIGHT OUTER JOIN result become the opposite when the left and right tables switch positions in the join operation.

Problem 4: Joining the Order Table with the Employee Table

In this problem, the Order table is the same table as in the previous problems, but the Employee table is a new table. Review the tables shown in Figure A8 below and answer the questions following the figure.

Figure A8

Order Table and Employee Table

Order

	OrderID	CustomerID	EmployeeID	RECEIVED	SHIPPED
1	1020	1111	1000	12/4/2015	12/15/2015
2	1021	1111	1000	12/5/2015	12/19/2015
3	1022	2222	1002	12/6/2015	12/14/2015
4	1023	3333	1000	12/7/2015	12/18/2015
5	1024	3333	1003	12/9/2015	12/21/2015
6	1825	4444	Null	7/20/2016	Null
7	1826	5555	1002	7/21/2016	Null
8	1827	6666	Null	7/22/2016	Null

Employee

	EmployeeID	NAME	ZIP	HDATE	DEPT	SALARY	ManagerID
1	1000	Jones	67226	9/1/2012	Sales	75000	NULL
2	1001	Smith	60606	10/15/2012	Marketing	74000	NULL
3	1002	Brown	50302	3/1/2013	Sales	48000	1000
4	1003	John	50305	6/1/2013	Sales	44000	1000
5	1004	Rachael	60606	7/15/2013	IT	59000	NULL
6	1005	Sue	67226	8/1/2013	Finance	58000	NULL
7	1006	Fred	67227	1/2/2014	Marketing	47000	1001

1. What is the primary key of the *Order* table?
2. What is the primary key of the *Employee* table?
3. What is the parent table?
4. What is the child table?
5. What is the foreign key in the child table that links the two tables? (The key column that matching values will be found)
6. Does every order have to be taken by an employee? How do you know?
7. What are the Matching Rows between the two tables? Examine the data provided in the Excel file to answer the next two questions.
8. Are there any Left-only Rows, rows in the *Order* table (left table) that have no matching rows in the *Employee* table (right table)? For example, these are orders that may have been placed online without the assistance of a specific employee. Copy the non-matching rows below:
9. Are there any Right-only Rows, rows in the *Employee* table that have no matching rows in the *Order* table? For example, these might be employees that are not associated with the sales process and therefore do not take orders.

Problem 4.1: INNER JOIN of Order Table with the Employee Table

1. Which possible set(s) of rows are included in the result set of an INNER JOIN? Select more than one options if necessary.
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows
2. Write the SQL code for the INNER JOIN of *Order* and *Employee* tables
3. Import the *Employee* table into MS Access. Create a query based on the answer above. Save this query as "Q4_1_INNER JOIN_Order_Employee"

The results of the SQL INNER JOIN for the Order and Employee tables are those identified as "matching Rows" and appear in Figure A9. However, note that the rows 1000 and 1002 from Employee table are duplicated to match those in the Order table. The columns with yellow background are from the right table.

Figure A9

Results of the SQL INNER JOIN for the Order and Employee tables

EmployeeID (Order)	OrderID	CustomerID	RECEIVED	SHIPPED	EmployeeID (Employee)	NAME	ZIP	HDATE	DEPT	SALARY
1000	1020	1111	12/4/2015	12/15/2015	1000	Jones	67226	9/1/2012	Sales	75000
1000	1021	1111	12/5/2015	12/19/2015	1000	Jones	67226	9/1/2012	Sales	75000
1000	1023	3333	12/7/2015	12/18/2015	1000	Jones	67226	9/1/2012	Sales	75000
1002	1022	2222	12/6/2015	12/14/2015	1002	Brown	50302	3/1/2013	Sales	48000
1002	1826	5555	7/21/2016	Null	1002	Brown	50302	3/1/2013	Sales	48000
1003	1024	3333	12/9/2015	12/21/2015	1003	John	50305	6/1/2013	Sales	44000

Problem 4.2: LEFT OUTER JOIN of Order Table with the Employee Table

1. Which possible set(s) of rows are included in the result set of a LEFT OUTER JOIN? Select more than one options if necessary.
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows
2. Write the SQL code for the LEFT OUTER JOIN of *Order* and *Employee* tables.
3. Create a query of LEFT OUTER JOIN between the *Order* and *Employee* tables based on the answer above. Save this query (shown in Figure A10) as "Q4_2_LEFT_OUTER_JOIN_Order_Employee."

The results of a LEFT OUTER JOIN contain Left-Only Rows, as well as Matching Rows (INNER JOIN results). The rows with green background are the Left-Only Rows.

Figure A10

Query results for LEFT OUTER JOIN of the Order and Employee tables

EmployeeID (Order)	OrderID	CustomerID	RECEIVED	SHIPPED	EmployeeID (Employee)	NAME	ZIP	HDATE	DEPT	SALARY
1000	1020	1111	12/4/2015	12/15/2015	1000	Jones	67226	9/1/2012	Sales	75000
1000	1021	1111	12/5/2015	12/19/2015	1000	Jones	67226	9/1/2012	Sales	75000
1000	1023	3333	12/7/2015	12/18/2015	1000	Jones	67226	9/1/2012	Sales	75000
1002	1022	2222	12/6/2015	12/14/2015	1002	Brown	50302	3/1/2013	Sales	48000
1002	1826	5555	7/21/2016	Null	1002	Brown	50302	3/1/2013	Sales	48000
1003	1024	3333	12/9/2015	12/21/2015	1003	John	50305	6/1/2013	Sales	44000
Null	1825	4444	7/20/2016	Null	Null	Null	Null	Null	Null	Null
Null	1827	6666	7/22/2016	Null	Null	Null	Null	Null	Null	Null

Problem 4.3: RIGHT OUTER JOIN of Order Table with the Employee Table

1. Which possible set(s) of rows are included in the result set of a RIGHT OUTER JOIN? Select more than one options if necessary.
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows
2. Write the SQL code for the RIGHT OUTER JOIN of the *Order* and *Employee* tables
3. Create a RIGHT OUTER JOIN query on the *Order* and *Employee* tables following the previous example. Save this query as "Q4_3_RIGHT_OUTER_JOIN_Order_Employee."

The results of a RIGHT OUTER JOIN contain the right-only Rows, as well as the matching rows (INNER JOIN results). The rows with the salmon background in Figure A11 are the right-only rows.

Figure A11

Query results for RIGHT OUTER JOIN of the Order and Employee tables

EmployeeID (Order)	OrderID	CustomerID	RECEIVED	SHIPPED	EmployeeID (Employee)	NAME	ZIP	HDATE	DEPT	SALARY
1000	1020	1111	12/4/2015	12/15/2015	1000	Jones	67226	9/1/2012	Sales	75000
1000	1021	1111	12/5/2015	12/19/2015	1000	Jones	67226	9/1/2012	Sales	75000
1000	1023	3333	12/7/2015	12/18/2015	1000	Jones	67226	9/1/2012	Sales	75000
1002	1022	2222	12/6/2015	12/14/2015	1002	Brown	50302	3/1/2013	Sales	48000
1002	1826	5555	7/21/2016	Null	1002	Brown	50302	3/1/2013	Sales	48000
1003	1024	3333	12/9/2015	12/21/2015	1003	John	50305	6/1/2013	Sales	44000
Null	Null	Null	Null	Null	1001	Smith	60606	10/15/2012	Marketing	74000
Null	Null	Null	Null	Null	1004	Rachael	60606	7/15/2013	IT	59000
Null	Null	Null	Null	Null	1005	Sue	67226	8/1/2013	Finance	58000
Null	Null	Null	Null	Null	1006	Fred	67227	1/2/2014	Marketing	47000

Problem 4.4: FULL OUTER JOIN of Order Table with the Employee Table

1. Which possible set(s) of rows are included in the result set of a FULL OUTER JOIN? Select more than one options if necessary.
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows
2. Write the SQL code for the FULL OUTER JOIN of *Order* and *Employee* tables
3. Create a query of FULL OUTER JOIN between *Order* and *Employee* tables based on the answer above. Save this query as "Q4_4_FULL_OUTER_JOIN_Order_Employee"

The results of a FULL OUTER JOIN contain the results from the INNER JOIN (matching rows), plus the left-only rows, and the right-only rows. The rows with a green background are left-only rows, and the rows with salmon background are right-only rows, all shown in Figure A12.

Figure A12
Query results for FULL OUTER JOIN of the Order and Employee tables

EmployeeID (Order)	OrderID	CustomerID	RECEIVED	SHIPPED	EmployeeID (Employee)	NAME	ZIP	HDATE	DEPT	SALARY
1000	1020	1111	12/4/2015	12/15/2015	1000	Jones	67226	9/1/2012	Sales	75000
1000	1021	1111	12/5/2015	12/19/2015	1000	Jones	67226	9/1/2012	Sales	75000
1000	1023	3333	12/7/2015	12/18/2015	1000	Jones	67226	9/1/2012	Sales	75000
1002	1022	2222	12/6/2015	12/14/2015	1002	Brown	50302	3/1/2013	Sales	48000
1002	1826	5555	7/21/2016	Null	1002	Brown	50302	3/1/2013	Sales	48000
1003	1024	3333	12/9/2015	12/21/2015	1003	John	50305	6/1/2013	Sales	44000
Null	1825	4444	7/20/2016	Null	Null	Null	Null	Null	Null	Null
Null	1827	6666	7/22/2016	Null	Null	Null	Null	Null	Null	Null
Null	Null	Null	Null	Null	1001	Smith	60606	10/15/2012	Marketing	74000
Null	Null	Null	Null	Null	1004	Rachael	60606	7/15/2013	IT	59000
Null	Null	Null	Null	Null	1005	Sue	67226	8/1/2013	Finance	58000
Null	Null	Null	Null	Null	1006	Fred	67227	1/2/2014	Marketing	47000

Problem 5: Joining the Employee Table with the Order Table
For this problem, the same two tables are used as in Problem 4. However, the table order is switched. Use the *Employee* table as the left table and the *Order* table as the right table.

Problem 5.1: INNER JOIN of Employee Table with the Order Table

1. Should the result of this INNER JOIN be the same as that of INNER JOIN between the *Order* table and *Employee* table (Problem 4.1)?
2. Write the SQL code for this join:
3. Create a query of INNER JOIN between *Employee* and *Order* tables. Save this query as "Q5_1_INNER_JOIN_Employee_Order"
4. Are the results the same as Problem 4.1? Describe the similarities or differences below.

Problem 5.2: LEFT OUTER JOIN of Employee Table with the Order Table

1. Should the result of this LEFT OUTER JOIN of the *Employee* table with the *Order* table be the same as that of the LEFT OUTER JOIN between the *Order* table and *Employee* table (ignoring the order of columns)?
2. Write the SQL code for this left join
3. Create a query of LEFT OUTER JOIN between *Employee* and *Order* tables. Save this query as "Q5_2_LEFT_OUTER_JOIN_Employee_ Order"
4. Are the results the same as Problem 4.2? Describe the similarities or differences.

Problem 5.3: RIGHT OUTER JOIN of Employee Table with the Order Table

1. Should the result of this RIGHT OUTER JOIN be the same as that of RIGHT OUTER JOIN between *Order* table and *Employee* table (ignore the order of columns)?
2. Write the SQL code for this join
3. Create a query of RIGHT OUTER JOIN between *Employee* and *Order* tables. Save this query as "Q5_3_RIGHT_ORJDR_JOIN_Employee_Order."
4. Are the results the same as Problem 4.3? Describe the similarities or differences.

Problem 5.4: FULL OUTER JOIN of Employee Table with the Order Table

1. Should the result of the FULL OUTER JOIN between the *Employee* table and the *Order* table be the same as that of FULL OUTER JOIN between *Order* table and *Employee* table (ignore the order of columns)?
2. Write the SQL code for this join.
3. Create a query of FULL OUTER JOIN between *Employee* and *Order* tables. Save this query as "Q5_4_FULL_OUTER_JOIN_Employee_ Order."
4. Are the results the same as Problem 4.4? Describe the similarities or differences.

Problem 6: Implementing Joins in Data Queries

For each of these queries, write the query in SQL. Save the query and the responses in the Access Database as "Query6_1," "Query6_2," "Query6_3," and so on.

1. Query6.1: For all customers, list their CustomerID, Name, Gender, and Age. Only include the result if there is a corresponding match in the "customer_extra" table. Hint: Use an INNER JOIN

2. Query 6.2: For all customers, list their CustomerID, Name, Gender, and Age. If there is no associated gender or age, then a NULL value should be returned for the unmatched field(s) in that particular row. Hint: Use a LEFT JOIN or a RIGHT JOIN
3. Query 6.3: Display the OrderID, CustomerID, and Customer "Name" associated with each of the orders in the *Order* table. If there is no associated Customer "Name," then a NULL value should be returned as the "Name" for that particular row.
4. Query 6.4: Display the OrderID, EmployeeId, and Employee "Name" associated with each of the orders in the *Order* table. If there is no associated Employee "Name," then a NULL value should be returned as the "Name" for that particular row.
5. Query 6.5: Display the customers (CustomerID and NAME) who have NOT placed any orders in the *Order* table.
6. Query 6.6: Display the employees (EmployeeID and NAME) who are NOT associated with any orders in the *Order* table.

Review Questions

1. An INNER JOIN is most closely associated with:
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows
    d. Matching Rows + Left-Only Rows + Right-Only Rows

2. A LEFT OUTER JOIN is most closely associated with:
    a. Matching Rows
    b. Left-Only Rows
    c. Matching Rows + Left-Only Rows
    d. Matching Rows + Right-Only Rows
    e. Matching Rows + Left-Only Rows + Right-Only Rows

3. A RIGHT OUTER JOIN is most closely associated with:
    a. Matching Rows
    b. Right-Only Rows
    c. Matching Rows + Left-Only Rows
    d. Matching Rows + Right-Only Rows
    e. Matching Rows + Left-Only Rows + Right-Only Rows

4. A FULL OUTER JOIN is most closely associated with:
    a. Matching Rows
    b. Right-Only Rows
    c. Matching Rows + Left-Only Rows
    d. Matching Rows + Right-Only Rows
    e. Matching Rows + Left-Only Rows + Right-Only Rows

Questions 5 to 12 use the data below:

Class Table

Class_ID	Instructor_ID	Description
ACG_201	001	Introduction to Financial Accounting
ACG_203	002	Introduction to Managerial Accounting
ACG_302	005	Intermediate Accounting 2

Instructor Table

Instructor_ID	Name	Class_ID
001	Corey Jones	ACG_201
002	Alexis Smith	ACG_203
003	Emily Lyons	ACG_301
004	Jack Stone	ACG_401

5. An INNER JOIN of the Class table and the Instructor table would result in how many total rows returned?
    a. 1 row
    b. 2 rows
    c. 3 rows
    d. 4 rows

6. A LEFT JOIN of the Class Table and the Instructor Table would result in how many total rows returned?
    a. 1 row
    b. 2 rows
    c. 3 rows
    d. 4 rows
    e. 5 rows
7. A RIGHT JOIN of the Class Table and the Instructor Table would result in how many total rows returned?
    a. 1 row
    b. 2 rows
    c. 3 rows
    d. 4 rows
    e. 5 rows
8. A FULL OUTER JOIN of the Class Table and the Instructor Table would return how many rows?
    a. 1 row
    b. 2 rows
    c. 3 rows
    d. 4 rows
    e. 5 rows
9. Using a LEFT JOIN, write the SQL code to return a list of all the classes and the associated instructors (if available). If a class doesn't have an instructor associated with it, the class should still be returned but with NULL values for fields related to the Instructor table.
10. Using a RIGHT JOIN, write the SQL code to return a list of all the classes and the associated instructors (if available). If a class doesn't have an instructor associated with it, the class should still be returned but with NULL values for fields related to the Instructor table
11. Using a LEFT JOIN, write the SQL code to return a list of all the INSTRUCTORS and their associated classes (if available). If an instructor doesn't have any class associated with them, the instructor should still be returned but with NULL values for fields related to the Class table.
12. Using a RIGHT JOIN, write the SQL code to return a list of all the INSTRUCTORS and their associated classes (if available). If an instructor doesn't have any class associated with them, the instructor should still be returned but with NULL values for fields related to the Class table.
13. The same query results can be achieved with either a LEFT JOIN OR a RIGHT JOIN.
    a. True
    b. False
14. The ordering of the tables in SQL join query will impact the results of both an INNER JOIN and a FULL OUTER JOIN.
    a. True
    b. False
15. The ordering of the tables in SQL join query will impact the result of a LEFT OUTER JOIN, but not for a RIGHT OUTER JOIN.
    a. True
    b. False
16. A LEFT OUTER JOIN returns the Left-Only rows but does NOT return the matching rows.
    a. True
    b. False
17. Briefly describe an accounting-related scenario in which a LEFT JOIN or a RIGHT JOIN would be particularly useful?

Appendix B

Part 2: Joins with Alteryx

Alteryx data analytics software can be used to explore, understand, integrate, aggregate and transform data, as well as to build machine learning models. Alteryx is a user-friendly tool that can join tables easily and presents the join process and results intuitively. This exercise introduces the tools and workflow used to join tables in Alteryx. For comparison and consolidation purposes, you will use the same set of tables as in Part 1.

Learning Objectives

1. Describe Alteryx tools and interface features for working with joins (including the JOIN tool and the UNION tool).
2. Create Alteryx workflows that implement INNER JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN, and FULL OUTER JOIN (using the UNION tool) types.
3. Practice results validation by examining output streams.
4. Compare and contrast Alteryx joins with SQL joins.
5. Create queries that include joins using Alteryx.

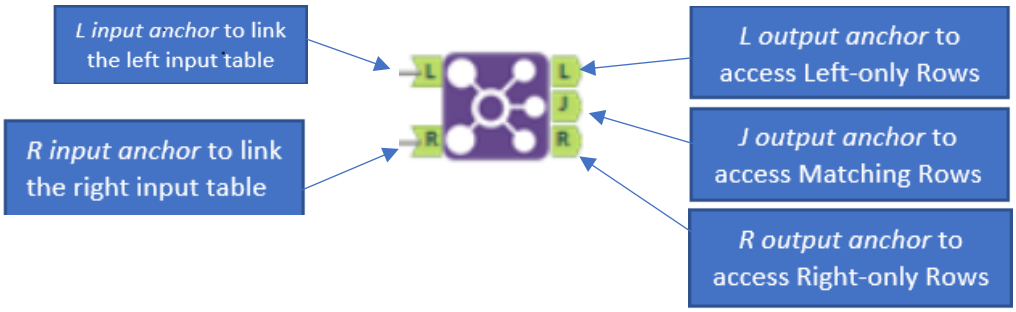
Tools Introduction

Joins are implemented with the JOIN tool and the UNION tool in Alteryx. External data can be imported into Alteryx using the INPUT DATA Tool. These three tools are introduced below.

The JOIN tool combines data from two input tables (also called input streams in Alteryx) to produce output streams. The two input streams should be linked to the input anchors marked as L and R on the left side of the tool icon. The L anchor links the left table to the JOIN tool and the R anchor links the right table to the JOIN tool. The three possible sets of rows generated from the JOIN tool are Left-only Rows, Matching (inner-join) Rows, and Right-only Rows. These row sets can be accessed from the three output anchors (L, J, and R) on the right side of the tool icon. The relationship between the input and output streams through the JOIN tool is illustrated in Figure B1.

Figure B1

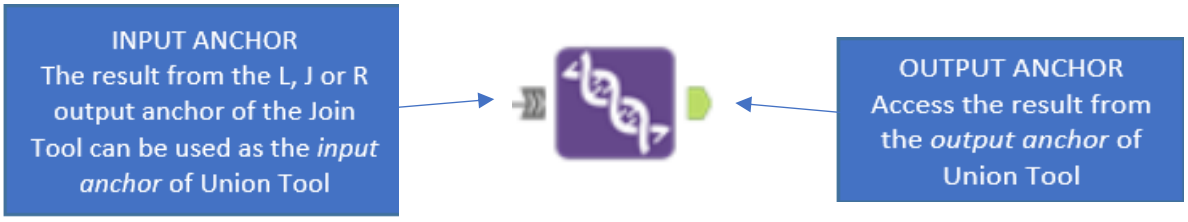
Alteryx JOIN Tool Icon



The UNION tool is used to combine rows vertically from different tables with similar columns. It functions much like the "Union" statement in MS Access SQL implements FULL OUTER JOINs. The UNION tool can be used to combine some or all of the three row sets (output streams) from the JOIN tool. When using the UNION tool, combine tables by linking them to the input anchor on the left side of the UNION tool. The combined result can be accessed through the output anchor on the right-hand side of the UNION tool (Figure B2).

Figure B2

Alteryx UNION Tool



The INPUT DATA tool is used to load data from external sources such as databases, files, etc. You can import our data into Alteryx from the Excel worksheets using the Input Tool. Table B1 provides a summary of the three basic Alteryx tools used in this exercise.

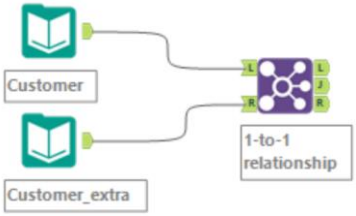
Table B1
Alteryx Tools Used in this Exercise

Tool	Icon	Description (from Alteryx website)
JOIN Tool		Use to combine two inputs based on common fields between the two tables. You can also Join two data streams based on record position.
UNION Tool		Use to combine two or more datasets on column names or positions. In the output, each column contains the rows from each input. You can configure how the columns stack or match up in the output.
INPUT DATA Tool		Use to add data to your workflow by connecting it to a file or database.

Problem 1: INNER JOIN of Customer and Customer_Extra
In Problem 1, you will be performing an INNER JOIN of the *Customer* table with the *Customer_Extra* table.

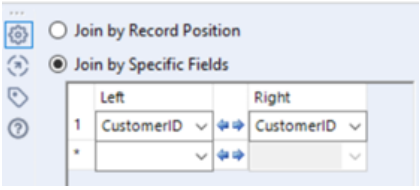
1. Use two INPUT DATA tools to import the *Customer* and *Customer_Extra* tables (Note: The location of the Excel data file has to be specified at this point)
2. Drag a JOIN tool to the design canvas to the right of the Input tools
3. Link the input tool of *Customer* to the L input anchor on the left-hand side of the JOIN tool
4. Link the input tool of *Customer_Extra* to the R input anchor on the right-hand side of the JOIN tool. The results of Steps 1-4 are displayed in Figure B3.

Figure B3
INNER JOIN of CUSTOMER and CUSTOMER_EXTRA table



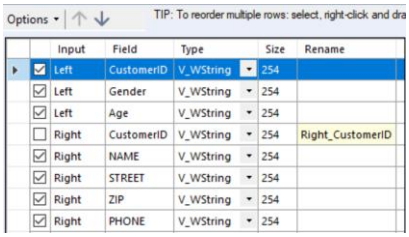
5. For the JOIN tool configuration, make sure to select "Join by Specific Fields" (see Figure B4).

Figure B4
JOIN Tool Configuration



6. All columns (Fields) from both the left and right tables are displayed in the configuration window. Note that Alteryx adds the prefix "Right_" for columns that share the same name in left and right tables. This can be checked or unchecked to be included or excluded from the result. CustomerID on the Right table is the foreign key and is unchecked in this example (Figure B5).

Figure B5
Column Selection

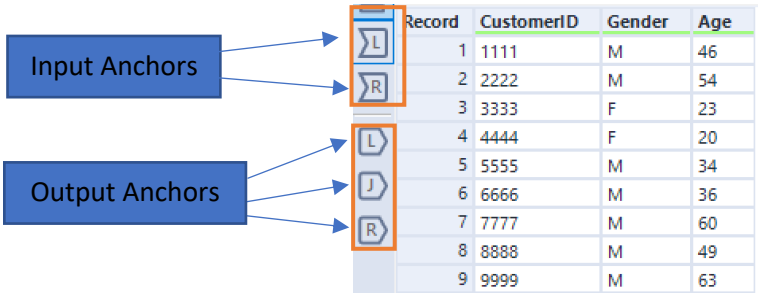


7. Run the workflow. The result is displayed in the result pane underneath the workflow pane. Note the result text for the JOIN tool in this pane. Nine (9) rows from both tables are joined. There are no unmatched rows on the left table or right table (Figure B6).

Figure B6
Workflow Results (Messages)

8. Next, examine the result pane. You can access the two input tables (input streams) and the three result row sets (output streams) by clicking the anchors to the left of the data view as indicated by the arrows. Note that the anchors for input streams are displayed at the top of the result pane and those for the output streams are displayed at the bottom (Figure B7).

Figure B7
Data View on the Result Pane



Record	CustomerID	Gender	Age
1	1111	M	46
2	2222	M	54
3	3333	F	23
4	4444	F	20
5	5555	M	34
6	6666	M	36
7	7777	M	60
8	8888	M	49
9	9999	M	63

9. Click on the various anchors to view the data and answer these questions.
   a. Click on input anchors L. What is this view?
   b. Click on input anchors R. What is this view?
   c. Click on the output anchor J. What is this view?
   d. Click on the output anchor L. What is this view?
   e. Click on the output anchor R. What is this view?
10. Compare the results with that of the results from Part 1 Problem 1 (SQL). Are the Alteryx results the same as the SQL results from Part 1?
11. What are some pros and cons of performing joins in Alteryx versus SQL?

Save this Alteryx work flow as "Problem 1.yxmd"

Problem 2: Joining the Customer Table with the Order Table

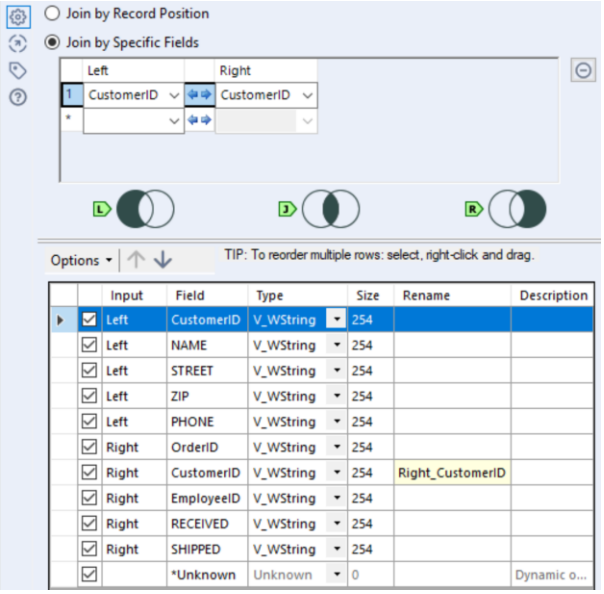
In this problem, you will learn how to work with the four join types (INNER JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN, and FULL OUTER JOIN) in Alteryx. Remember, the Alteryx "L" and "R" output streams do not correspond to a LEFT OUTER JOIN and a RIGHT OUTER JOIN respectively. Instead, the "L" output stream returns the "Left-Only Rows," while the "R" output stream returns the "Right-Only" rows. You might recall from Part 1 that a LEFT OUTER JOIN returns the matching rows plus "left-only rows," while a RIGHT OUTER JOIN returns the matching rows plus "right-only rows."

Problem 2.1: INNER JOIN of Customer Table with the Order Table

1. Start a new workflow and import the *Customer* table and *Order* table with the INPUT DATA tool.
2. Drag a JOIN tool to the design canvas to the right of the INPUT DATA tools.
3. Link the Customer input to the "L" input anchor on the left-hand side of the JOIN tool.
4. Link the Order input to the "R" input anchor on the left-hand side of the JOIN tool.
5. For the JOIN tool configuration, make sure to select "Join by Specific Fields." Ensure the primary key and foreign key fields of the parent and child tables are selected as the fields to join the tables with. Select (check) the output fields and ensure that the fields the tables are joined by are both returned. Your configuration pane should look like Figure B8.

Figure B8

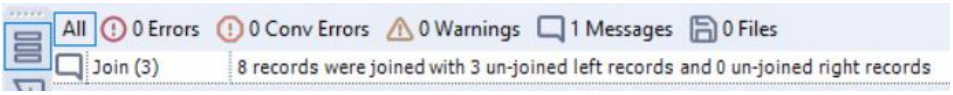
Configuration Pane for the INNER JOIN



6. Run the workflow, then click the result message icon in the result pane to obtain a result that resembles Figure B9.

Figure B9

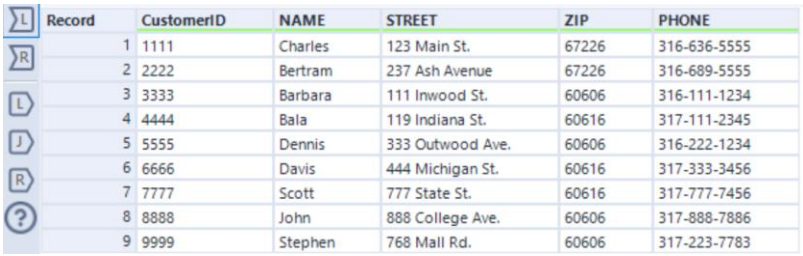
Workflow Result



7. Click on the various anchors to view the different input and output data streams, then click the input anchor "L." Your result should look like Figure B10.

Figure B10

Input Anchor "L" Result



Record	CustomerID	NAME	STREET	ZIP	PHONE
1	1111	Charles	123 Main St.	67226	316-636-5555
2	2222	Bertram	237 Ash Avenue	67226	316-689-5555
3	3333	Barbara	111 Inwood St.	60606	316-111-1234
4	4444	Bala	119 Indiana St.	60616	317-111-2345
5	5555	Dennis	333 Outwood Ave.	60606	316-222-1234
6	6666	Davis	444 Michigan St.	60616	317-333-3456
7	7777	Scott	777 State St.	60616	317-777-7456
8	8888	John	888 College Ave.	60606	317-888-7886
9	9999	Stephen	768 Mall Rd.	60606	317-223-7783

   a. Click on input anchor "R." Take a screenshot of your results.
   b. Click on output anchor "J." Take a screenshot of your results.
   c. Click on output anchors "L." Take a screenshot of your results and copy below.
   d. Click on output anchors "R." Take a screenshot of your results and copy below.

8. Which output stream contains the INNER JOIN result?
   a. The output stream under anchor L
   b. The output stream under anchor J
   c. The output stream under anchor R

9. The "J" output stream contains what type of results?
   a. Matching rows
   b. Left-only rows
   c. Right-only rows

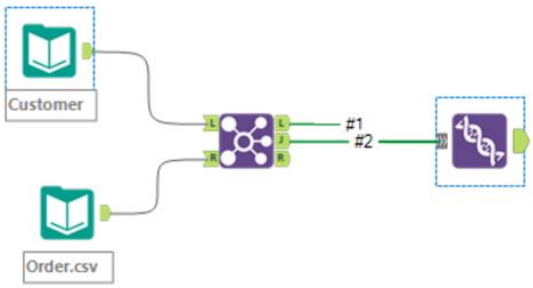
10. The "L" output stream contains what type of results?
    a. Matching rows
    b. Left-only rows
    c. Matching rows plus left-only rows
11. The "R" output stream contains what type of results?
    a. Matching rows
    b. Right-only rows
    c. Matching rows plus right-only rows
12. Take a screenshot of your workflow and the **INNER JOIN** results and paste below.

Problem 2.2: LEFT OUTER JOIN of Customer Table with the Order Table

From Problem 2.1 above, you see that the "L" output tab from the JOIN tool returns the "left-only" rows and that this is not the same as a "LEFT OUTER JOIN." Recall that a LEFT OUTER JOIN returns the matching rows plus left-only rows.

To create a LEFT OUTER JOIN in Alteryx, you will use both the JOIN tool and the UNION tool to return the combination of the matching rows + left-only rows. First, use the JOIN tool to create the INNER JOIN of the *Customer* and the *Order* table. Then, use the UNION tool to combine the "J" output stream with the "L" output stream to create the LEFT OUTER JOIN (Figure B11).

Figure B11
Left Join Workflow



You will now build on what you did in Problem 2.1 to create the LEFT OUTER JOIN.

1. Start with the workflow from Problem 2.1 with the Customer data as the "L" input and the Order data as the "R" input to the JOIN tool.
2. Drag a UNION tool to the design canvas to the right of the Join tool
3. Drag the L output stream of the Join tool to the input anchor of the UNION tool. A "#1" will be added to the line between the two anchors.
4. Drag the J output stream of the JOIN tool to the input anchor of the UNION tool. A "#2" will be added to the line between the two anchors. Your results should be similar to Figure B11.
5. Run the workflow and click on the UNION tool to view the input and output data streams in the result pane.
6. There are two input data streams to the UNION Tool. What is input stream #1 (which appears in Figure B12)?
    a. The stream of matching rows of the join result
    b. The stream of left-only rows of the join result
    c. The stream of right-only rows of the join result

Figure B12
Left Join Workflow

Record	CustomerID	NAME	STREET	ZIP	PHONE
1	OK: 100.00%	Scott	777 State St.	60616	317-777-7456
2	8888	John	888 College Ave.	60606	317-888-7886
3	9999	Stephen	768 Mall Rd.	60606	317-223-7783

7. What is input stream #2?
    1. The stream of matching rows of the join result
    2. The stream of left-only rows of the join result

3. The stream of right-only rows of the join result
8. Take a screenshot of input stream #2.
9. The output stream from the UNION contains which of the following?
    1. Matching rows
    2. Left-only rows
    3. Matching rows + left-only rows
10. Take a screenshot of the output stream from the Union.
11. How many rows are returned in input stream #1?
12. How many rows are returned in input stream #2?
13. How many rows are returned in the output stream? _
14. Is the sum of the number of rows in the two input streams equal to the number of rows in the output stream?
15. Take a screenshot of your workflow and paste it below. This should be similar to Figure 10.

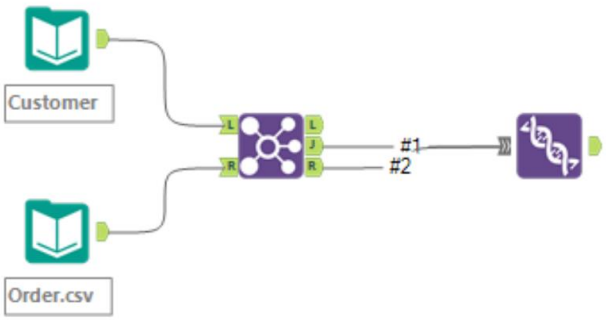
Problem 2.3: RIGHT OUTER JOIN of Customer Table with the Order Table

In Problem 2.2, you created a LEFT OUTER JOIN in Alteryx. A RIGHT OUTER JOIN is created in a similar manner, except the "R" output (instead of the "L") from the join is used as an input stream with the "J" output.

To create a RIGHT OUTER JOIN in Alteryx, you will use both the JOIN tool and the UNION tool to return the combination of the matching rows + right-only rows. First, use the JOIN tool to create the INNER JOIN of the Customer and the ORDER table. Then, use the UNION tool to combine the "J" output stream with the "R" output stream to create the RIGHT OUTER JOIN as shown in Figure B13.

Figure B13

RIGHT OUTER JOIN in Alteryx



1. Start with the workflow from Problem 2.2, which has the Customer data as the "L" input and the Order data as the "R" input to the JOIN tool.
2. Delete the previous input links to the UNION Tool, i.e. delete the "L" (#1) input stream and the "J"(#2) input stream.
3. Drag the "J" output stream of the JOIN tool to the input anchor of the UNION tool, "#1" will be added to the line between the two anchors.
4. Drag the R output stream of the JOIN tool to the input anchor of the UNION tool. "#2" will be added to the line between the two anchors. Your results should be similar to Figure 11.
5. Run the workflow and click on the UNION tool to view the input and output data streams in the result pane.
6. There are two input data streams to the UNION Tool. What is the input stream #1? (Your results should be similar to Figure B13).
    a. The stream of matching rows of the join result
    b. The stream of left-only rows of the join result
    c. The stream of right-only rows of the join result
7. What is input stream #2?
    a. The stream of matching rows of the join result
    b. The stream of left-only rows of the join result
    c. The stream of right-only rows of the join result
8. Take a screenshot of input stream #2.
9. The output stream from the UNION Tool above contains which of the following?
    a. Matching Rows
    b. Right-Only Rows

c. Matching Rows + Right-Only Rows

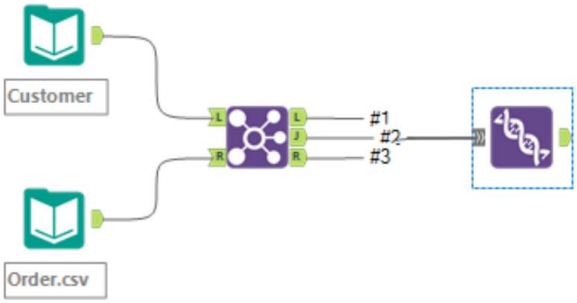
10. Take a screenshot of the output stream from the UNION and copy below.
11. How many rows are returned in input stream #1?
12. How many rows are returned in input stream #2?
13. How many rows are returned in the output stream?
14. Is the sum of the number of rows in the two input streams equal to the number of rows in the output stream?
15. Take a screenshot of this workflow and paste it below.

Problem 2.4: FULL OUTER JOIN of Customer Table with the Order Table

A FULL OUTER JOIN returns the Matching Rows + Left-Only Rows + Right-Only Rows. As such, a FULL OUTER JOIN is created by using the UNION tool to combine the "L," "J," and "R" output streams from the JOIN tool (Figure B14).

Figure B14

FULL OUTER JOIN Between the CUSTOMER and ORDER Tables



Modify your workflow from Problem 2.3 to create a FULL OUTER JOIN by following these steps:

1. Delete the previous links from Problem 2.2 from the JOIN tool to the UNION tool
2. Create three new links from the "L", "J", and "R" output streams from the JOIN tool as input streams into the UNION tool. Take a screenshot of the workflow and paste it here. (This should be similar to Figure 12).
3. View the output stream from the UNION tool. How many rows are returned?
4. Take a screenshot of the output stream from the UNION and save this Alteryx project as "Problem 2.yxmd."

Key Takeaways:

- With Alteryx, an INNER JOIN uses the JOIN tool; however, a left join, right join and FULL OUTER JOIN all require both the JOIN tool and the UNION tool.
- Alteryx makes the Matching Rows, Left-Only Rows and Right-Only Rows easily accessible through the output streams from the JOIN tool.
- In the Alteryx JOIN tool, the "L" output stream is NOT a LEFT OUTER JOIN. Instead, it returns the "Left-Only Rows." Similarly, the "R" output stream is NOT a RIGHT OUTER JOIN in that the "Right-Only Rows" are returned. Recall that a LEFT OUTER JOIN returns the Matching Rows + Left-Only Rows, while a RIGHT OUTER JOIN returns the Matching Rows + Right-Only Rows.
- The decision of whether LEFT, RIGHT or FULL OUTER JOIN is needed can be made after examining the L and R output streams. If the L output stream is empty, LEFT OUTER JOIN will not add any additional rows. If the R output stream is empty, RIGHT OUTER JOIN will not add any additional rows. On the other hand, if both streams have data, a FULL OUTER JOIN is needed to return all the rows from both tables.

Problems 3, 4, and 5: Joining Additional Tables in Alteryx

For Problems 3, 4, and 5, you will use Alteryx to perform the various joins for the different table combinations. As you perform each join, you can verify the output results with your responses from Part 1 (SQL exercises). For each combination, capture a screenshot of both the workflow and the output results, and copy these screenshots into the answer sheet below. Problem 3 has been completed for you. Build the workflows and run the query for each part of Problems 4 and 5.

Problem 3: Joining the Order Table with the Customer Table

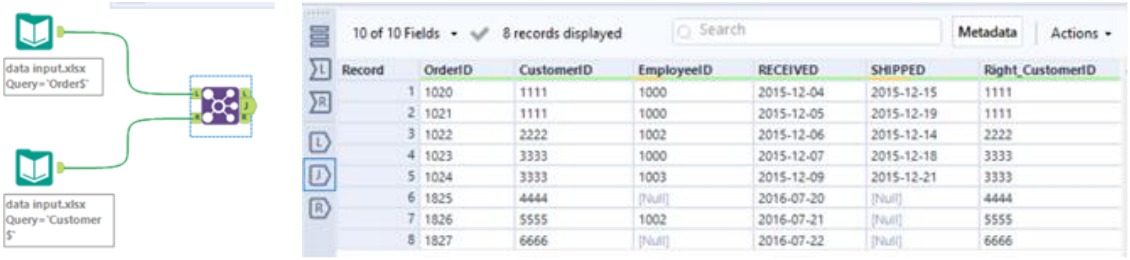
This problem is similar to Problem 2, but the Order table is now the LEFT input stream, while the Customer table is the RIGHT input stream.

Problem 3.1: Order table INNER JOIN Customer table

The workflow and query results for this problem appear below in Figure B15.

Figure B15

Workflow and Results for Order table INNER JOIN Customer table



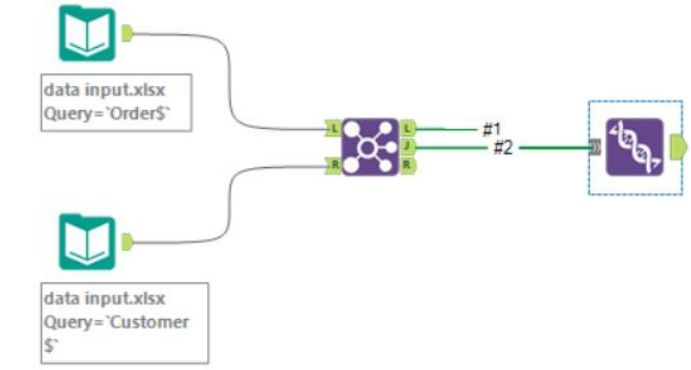
Record	OrderID	CustomerID	EmployeeID	RECEIVED	SHIPPED	Right_CustomerID
1	1020	1111	1000	2015-12-04	2015-12-15	1111
2	1021	1111	1000	2015-12-05	2015-12-19	1111
3	1022	2222	1002	2015-12-06	2015-12-14	2222
4	1023	3333	1000	2015-12-07	2015-12-18	3333
5	1024	3333	1003	2015-12-09	2015-12-21	3333
6	1825	4444	[Null]	2016-07-20	[Null]	4444
7	1826	5555	1002	2016-07-21	[Null]	5555
8	1827	6666	[Null]	2016-07-22	[Null]	6666

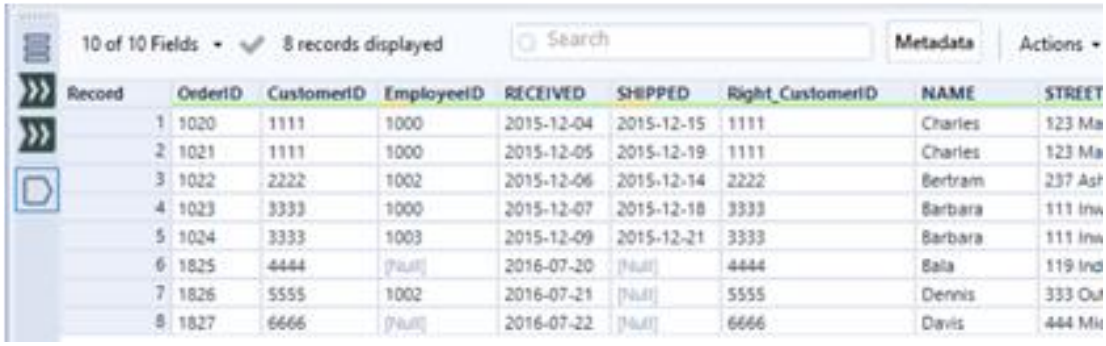
Problem 3.2: Order table LEFT JOIN Customer table

The workflow and query results for this problem appear in Figure B16.

Figure B16

Workflow and Results for Order table LEFT JOIN Customer table





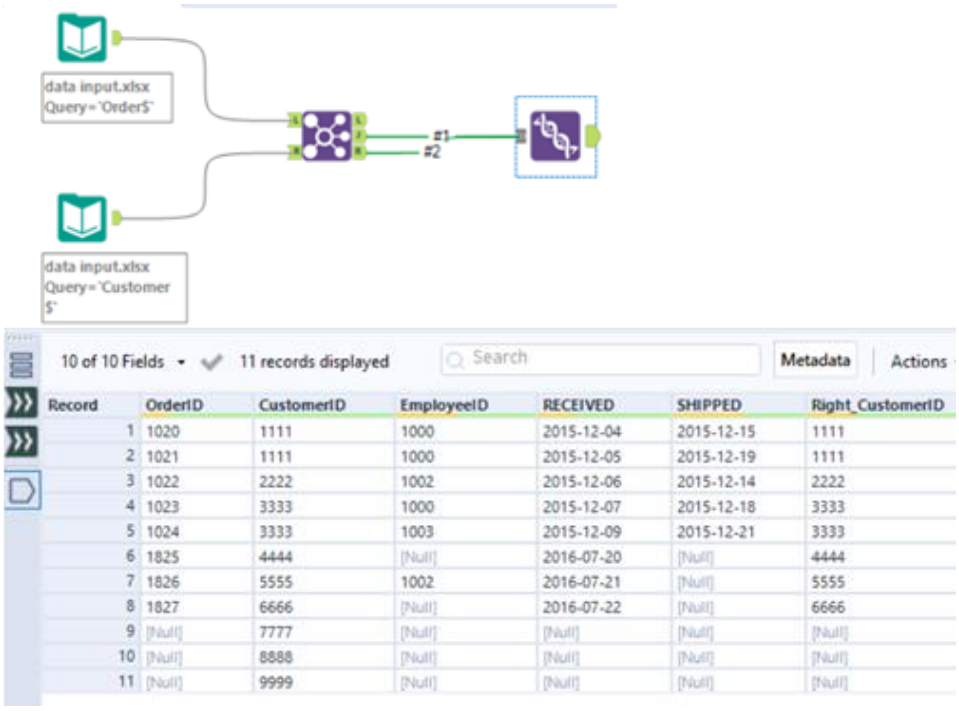
Record	OrderID	CustomerID	EmployeeID	RECEIVED	SHIPPED	Right_CustomerID	NAME	STREET
1	1020	1111	1000	2015-12-04	2015-12-15	1111	Charles	123 Ma
2	1021	1111	1000	2015-12-05	2015-12-19	1111	Charles	123 Ma
3	1022	2222	1002	2015-12-06	2015-12-14	2222	Bertram	237 Ash
4	1023	3333	1000	2015-12-07	2015-12-18	3333	Barbara	111 Inv
5	1024	3333	1003	2015-12-09	2015-12-21	3333	Barbara	111 Inv
6	1825	4444	[Null]	2016-07-20	[Null]	4444	Bala	119 Ind
7	1826	5555	1002	2016-07-21	[Null]	5555	Dennis	333 Ou
8	1827	6666	[Null]	2016-07-22	[Null]	6666	Davis	444 Mi

Problem 3.3: *Order* table RIGHT JOIN *Customer* table

The workflow and query results for this problem appear below in Figure B17.

Figure B17

Workflow and Results for Order table RIGHT JOIN Customer table



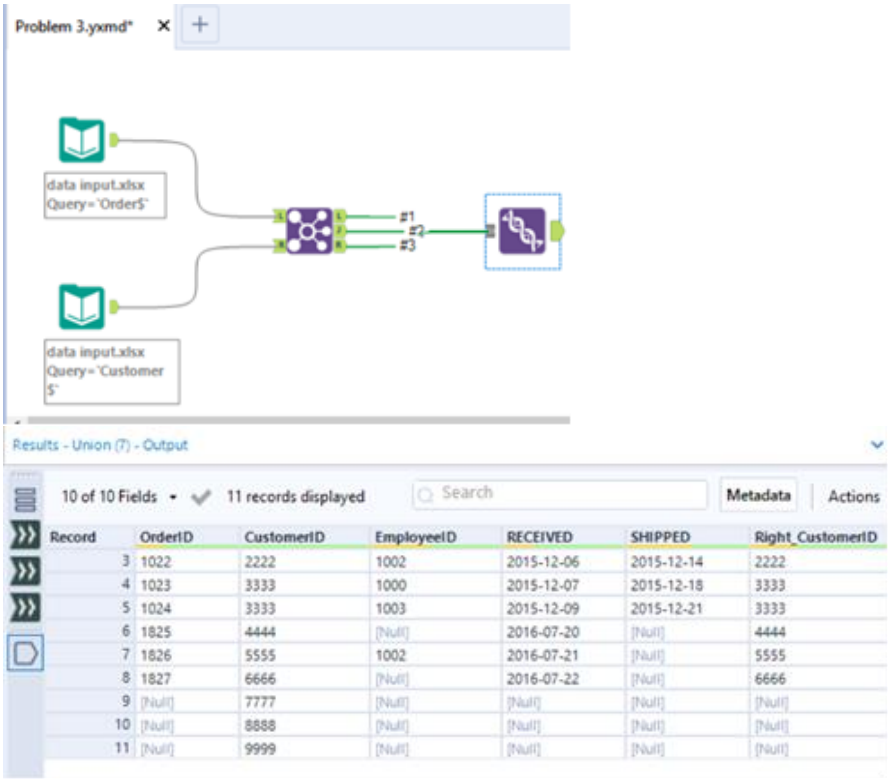
Record	OrderID	CustomerID	EmployeeID	RECEIVED	SHIPPED	Right_CustomerID
1	1020	1111	1000	2015-12-04	2015-12-15	1111
2	1021	1111	1000	2015-12-05	2015-12-19	1111
3	1022	2222	1002	2015-12-06	2015-12-14	2222
4	1023	3333	1000	2015-12-07	2015-12-18	3333
5	1024	3333	1003	2015-12-09	2015-12-21	3333
6	1825	4444	[Null]	2016-07-20	[Null]	4444
7	1826	5555	1002	2016-07-21	[Null]	5555
8	1827	6666	[Null]	2016-07-22	[Null]	6666
9	[Null]	7777	[Null]	[Null]	[Null]	[Null]
10	[Null]	8888	[Null]	[Null]	[Null]	[Null]
11	[Null]	9999	[Null]	[Null]	[Null]	[Null]

Problem 3.4: *Order* table FULL OUTER JOIN *Customer* table

The workflow and query results for this problem appear below in Figure B18.

Figure B18

Workflow and Results for Order table FULL OUTER JOIN Customer table



Record	OrderID	CustomerID	EmployeeID	RECEIVED	SHIPPED	Right_CustomerID
3	1022	2222	1002	2015-12-06	2015-12-14	2222
4	1023	3333	1000	2015-12-07	2015-12-18	3333
5	1024	3333	1003	2015-12-09	2015-12-21	3333
6	1825	4444	[Null]	2016-07-20	[Null]	4444
7	1826	5555	1002	2016-07-21	[Null]	5555
8	1827	6666	[Null]	2016-07-22	[Null]	6666
9	[Null]	7777	[Null]	[Null]	[Null]	[Null]
10	[Null]	8888	[Null]	[Null]	[Null]	[Null]
11	[Null]	9999	[Null]	[Null]	[Null]	[Null]

Problem 4: Joining the Order Table with the Employee Table

For Problem 4, you will use Alteryx to perform the various joins for the different table combinations. As you perform each join, you can verify the output results from your SQL responses in Part 1. For each combination,

capture a screenshot of both the workflow and the output results, and save this Alteryx project as "Problem 4.yxmd." The table combinations for each problem are as follows:

Problem 4.1: *Order* table INNER JOIN *Employee* table
Problem 4.2: *Order* table LEFT JOIN *Employee* table
Problem 4.3: *Order* table RIGHT JOIN *Employee* table
Problem 4.4: *Order* table FULL OUTER JOIN *Employee* table

Problem 5: Joining the Employee Table with the Order Table

This problem is similar to Problem 4, but the *Employee* table is now the LEFT input stream, while the *Order* table is the RIGHT input stream. Save this Alteryx project as "Problem 5.yxmd." The table combinations for each problem are as follows:

Problem 5.1: *Employee* table INNER JOIN *Order* table
Problem 5.2: *Employee* table LEFT JOIN *Order* table
Problem 5.3: *Employee* table RIGHT JOIN *Order* table
Problem 5.4: *Employee* table FULL OUTER JOIN *Order* table

Problem 6: Implementing Joins in Data Queries

For each of the following queries, use Alteryx to determine the solution. Make a copy of the workflow and the query results for each query and save this Alteryx project as "Problem 6.yxmd."

Query 6.1: For all customers, list their CustomerID, Name, Gender, and Age. Only include the result if there is a corresponding match in the *Customer_Extra* table.
Query 6.2: For all customers, list their CustomerID, Name, Gender, and Age. If there is no associated gender or age, then a NULL value should be returned for the unmatched field(s) in that particular row. Hint: Use a Left Join or a Right Join
Query 6.3: Display the OrderID, CustomerID, and Customer "Name" associated with each of the orders in the ORDER table. If there is no associated Customer "Name," then a NULL value should be returned as the "Name" for that particular row.
Query 6.4: Display the OrderID, EmployeeId, and Employee "Name" associated with each of the orders in the ORDER table. If there is no associated Employee "Name," then a NULL value should be returned as the "Name" for that particular row.
Query 6.5: Display the customers (CustomerID and NAME) who have NOT placed any orders in the ORDERS table.
Query 6.6: Display the employees (EmployeeID and NAME) who are NOT associated with any orders in the ORDERS table.

Review Questions

1. In the Alteryx JOIN tool, the "J" output anchor displays which rows?
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows
    d. Matching Rows + Left-Only Rows
    e. Matching Rows + Right-Only Rows
2. In the Alteryx JOIN tool, the "L" output anchor displays which rows?
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows
    d. Matching Rows + Left-Only Rows
    e. Matching Rows + Right-Only Rows
3. In the Alteryx JOIN tool, the "R" output anchor displays which rows?
    a. Matching Rows
    b. Left-Only Rows
    c. Right-Only Rows
    d. Matching Rows + Left-Only Rows
    e. Matching Rows + Right-Only Rows
4. How would you perform a LEFT OUTER JOIN in Alteryx?

   a. Join two input data streams with the JOIN tool and select the "L" output for a LEFT OUTER JOIN
   b. Use the "L" output and the "J" output of the JOIN tool as inputs to the UNION tool. The output anchor from the UNION tool will be a LEFT OUTER JOIN.
   c. Use the "L" output, the "J" output, and the "R" output as inputs to the UNION tool. The output anchor from the UNION tool will be a LEFT OUTER JOIN.
   d. A LEFT OUTER JOIN is not possible in Alteryx; it is only possible in SQL.

5. How would you perform an INNER JOIN in Alteryx?
   a. Join two input data streams with the JOIN tool and select the "L" output
   b. Join two input data streams with the JOIN tool and select the "J" output
   c. Join two input data streams with the JOIN tool and select the "R" output

6. How would you perform a FULL OUTER JOIN in Alteryx?
   a. Join two input data streams with the JOIN tool and select the "L" output for a FULL OUTER JOIN
   b. Use the "L" output and the "J" output of the JOIN tool as inputs to the UNION tool. The output anchor from the UNION tool will be a FULL OUTER JOIN.
   c. Use the "L" output, the "J" output, and the "R" output as inputs to the UNION tool. The output anchor from the UNION tool will be a FULL OUTER JOIN.
   d. A FULL OUTER JOIN is not possible in Alteryx; it is only possible in SQL.

7. Which tool (SQL or Alteryx) do you prefer using to join data together? Why?

8. What do you like better about SQL?

9. What do you like better about Alteryx?

Appendix C

Part 3: Joins Using Tableau

Tableau is primarily a data visualization software package. Data must be imported to serve as the source to create the visualizations. The data can be from a single table or a combination of multiple tables. The data structured or combined at the data source stage is referred to as the "data model." A data model contains two layers: a physical layer and a logical layer. The physical layer is the basis of the logical layer and is created by using the join operations you learned in Parts 1 and 2 of this assignment. It can also be created using a union operation.

If a specific type of join is required in the visualization, the join type can be implemented at the physical layer. The logical layer delays the decision on how data are joined until you create the visualization and the data needs are concrete. Visualizations based on different join types can be generated with the same logical layer data model. Data can be connected in the logical layer using relationships (what Tableau calls "noodles.")

This exercise introduces the Tableau data model with its two layers, studies the type of joins used based on the visualization at the logical layer, and reinforces the rationales behind the application of different types of joins. For the purposes of comparison and review, the same set of tables used in previous two parts will be used: *Customer*, *Customer_Extra*, *Order* and *Employee*. The tables are described in Part 1 (Appendix A) and contained in the accompanying Excel file "Tables.xlsx."

Learning Objectives

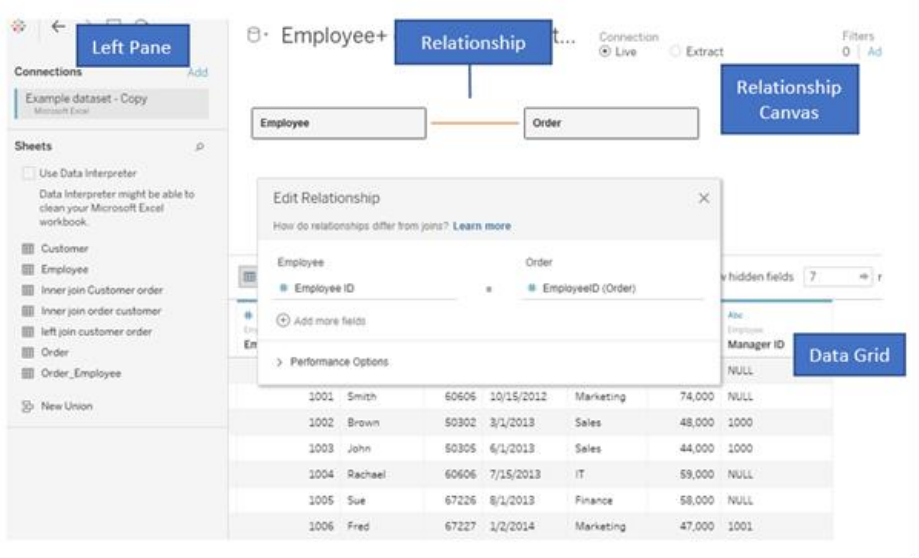
1. Implement joins in Tableau's physical and logical model layers correctly.
2. Demonstrate how join type choices in the logical model layer affect visualizations.
3. Use Left-only Rows and Right-only Rows options in Tableau correctly.
4. Demonstrate correct use of fields to trigger various join types in Tableau.

Background: Tableau Data Model

External data must be imported into Tableau as the Data Source before any visualizations can be created. As tables are added to the Data Source, a data model is created for the future visualizations. A single-table data model is straightforward. However, when multiple tables are needed, the data model should be designed carefully for accurate and flexible data needs. A data model has two layers: the logical layer and the physical layer. The logical layer is the default when you add tables to the Relationship Canvas of the Data Source interface. If more than one table is added, Tableau will try to guess the data fields (or columns) that should be used to connect the two tables. Tables are connected through the "relationships" ("noodles" in Tableau terminology) in the logical data model. See the orange line between the two tables in Figure C1.

Figure C1

Logical Layer of the Tableau Data Model



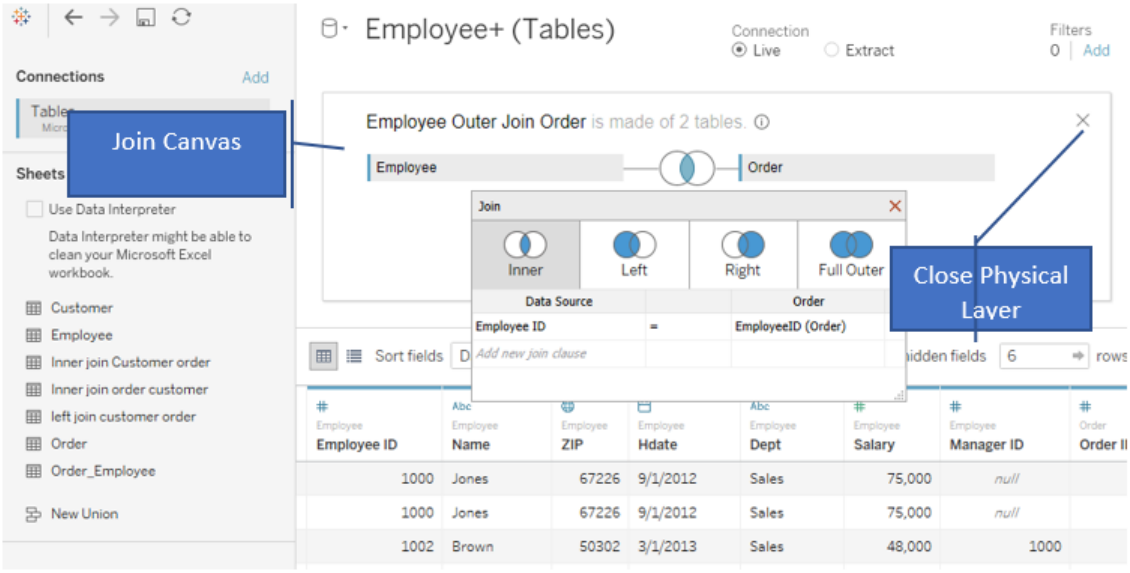
You can click the relationship line to display a popup window where the columns forming the relationship can be edited. Tables on this layer are called logical tables, which can contain multiple physical tables joined together at the physical layer of the logical table. Logical tables are related instead of being physically joined at this layer.

In the data grid, you can preview data by clicking the table name on the canvas. If you click Employee in the following figure, the Employee data is displayed in the data grid underneath the canvas. Similarly, if you click Order, the data from the Order table is displayed. There is no view of the two tables combined since the tables are not joined at the data source stage in the logical layer. In fact, the decision on what type of join should be used is made when a specific visualization is being built based on the exact data fields to be used and how those fields will be used. In other words, Tableau makes the decision on the type of join dynamically based on the specific visualization that is being built. Tableau calls this a "contextual join."

Creating a logical data model requires dragging and dropping tables to the relationship canvas of the Data Source page and confirming or editing the relationships between the two tables. The physical layer data model is similar to a join with SQL or Alteryx. The user decides on the type of join to combine the data, and the result is fixed once the join is completed. If a different result is needed, the join type must be changed at the data source (see Figure C2). Since all tables on the Physical Layer are joined together, a single flat table is created to support the logical table to which the physical layer belongs.

Figure C2

Physical Layer of the Data Model



To view the physical layer of the data model, double-click a logical table on the canvas of the Data Source page. The physical layer is displayed as a pane within the Join Canvas with the "X" button to exit the physical layer. If you drag tables to this pane, Tableau will physically join or union the two tables. If the tables are joined, a Venn diagram between the two tables shows the join type, with the joined result displayed in the data grid underneath the canvas. By default, an inner join is performed by Tableau if it finds matching columns in the two tables. To edit the join relationship, click the Venn diagram and a popup window will display in which you can change the matching columns to join by and the type of join performed. The join result preview is displayed in the data grid. Click the "X" of the Physical Layer pane to exit and return to the logical layer. Table C1 contains a summary of Tableau terminology for your reference.

Table C1
Tableau Terminology Summary

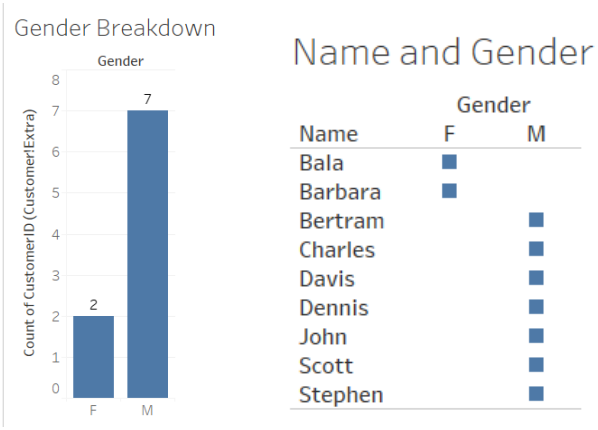
Term	Definition
Data Model	External data imported to Tableau and combined to form the data source.
Logical Layer	The top layer in the data model where the join relationships between tables are established by indicating the matching columns if there are more than one logical table. The join is not executed however.
Logical Table	Table objects on the logical layer. The Logical Table must be based on at least one physical table but could contain multiple tables that have been joined together at the physical layer.
Relationship (noodle)	The matching columns of the logical tables that will be joined upon at the time when a visualization is created. It is represented as a curve at the Logical Layer of a data model, implying the flexibility of a relationship or noodle.
Physical Layer	The layer underneath a Logical Table. At this layer, more than one physical table can be joined together to represent the physical table.
Join or Union	The operation used in the physical layer to combine multiple physical tables into a logical table. Join and union have the same meaning as that in SQL. Join and union are displayed as straight lines in contrast to curves used to represent relationships (or noodles).
Contextual Join	The join of logical tables happening after Tableau knows the specific columns used to build a visualization. The join type is decided by Tableau instead of the user.

Problem 1: Join of Customer and Customer_Extra

Recall that the *Customer* table contains the CustomerID, Name, Street, Zip, and Phone fields, while the Customer_Extra table includes the Gender and Age. Because you want to create visualizations with all of these fields, the data will need to be joined together.

The goal of this problem is to create two visualizations: one that will display the gender breakdown of customers; and a second that lists the name and gender of each customer. Although the gender breakdown can be created with data only from the *Customer_Extra* table, the list of customer names with their gender requires joining the data from two tables. Figure C3 shows the goal of this problem, to create two data visualizations.

Figure C3
The Goal: Two Visualizations

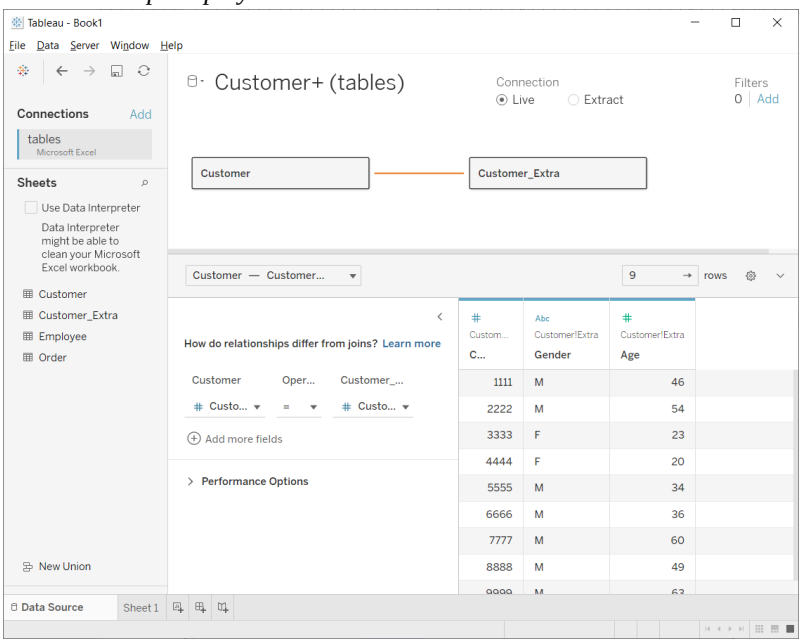


Problem 1.1: Solving with the logical layer

To create a visualization with data from the Customer table and the Customer_Extra table, you must input the data from both tables into Tableau.

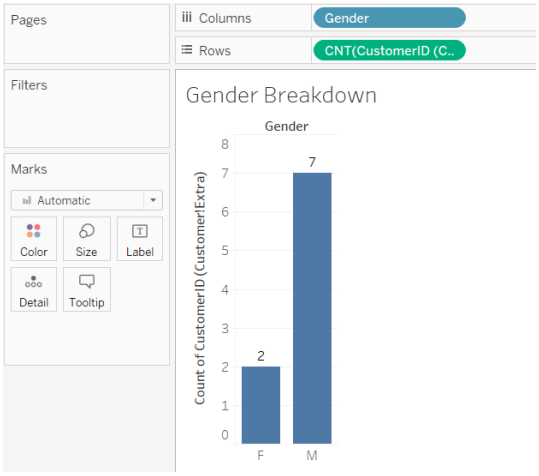
1. Open Tableau and "Connect" to a "Microsoft Excel" file.
2. Select the "tables.xlsx" file; by default, the logical layer interface is displayed.
3. Drag the *Customer* and the *Customer_Extra* table to the main canvas. Note the orange line between the two tables, indicating a relationship, and that the relationship can be defined in the data pane. In Figure C4 below, you can see that Tableau correctly guessed that the *Customer* and the *Customer_Extra* tables should be linked via the CustomerID field to create the goal visualization.

Figure C4
A Relationship Displayed in the Tableau Main Canvas



4. Now that you have defined the relationship at the logical layer, you can use data in "Sheet 1" to create the visualization showing the numbers of male and female customers.
    a. Move the "CustomerID" dimension to the Rows shelf
    b. Move the "Gender" measure to the Columns shelf
    c. Select the right arrow from the "CustomerID" dimension, then select "CNT" to count the number of customers with each gender type.
    d. Rename this visualization as "Gender Breakdown" to produce what you see in Figure C5.

Figure C5
The Gender Breakdown Visualization



5. To create the second visualization listing the names and gender of the customers, insert a new worksheet. Move the "Name" dimension to the rows shelf and the "Gender" dimension to the columns shelf. Then move another "Gender" dimension to the "Detail" marks card. Rename this worksheet as "Name and Gender" and export this workbook as "Problem 1 Tableau Logical.twbx."

Key Takeaways:

1. The two tables are indeed joined before the visualization is created even though the two tables are only connected with a relationship at the Logical Layer.

2. With the logical layer data model, accessing data fields from source tables is as easy as using a single-table data model.

Reviewing the Physical Layer

The visualization from above was created via a relationship defined at the logical layer between the Customer Table and the Customer_Extra table. To review the physical layer associated with the logical layer.

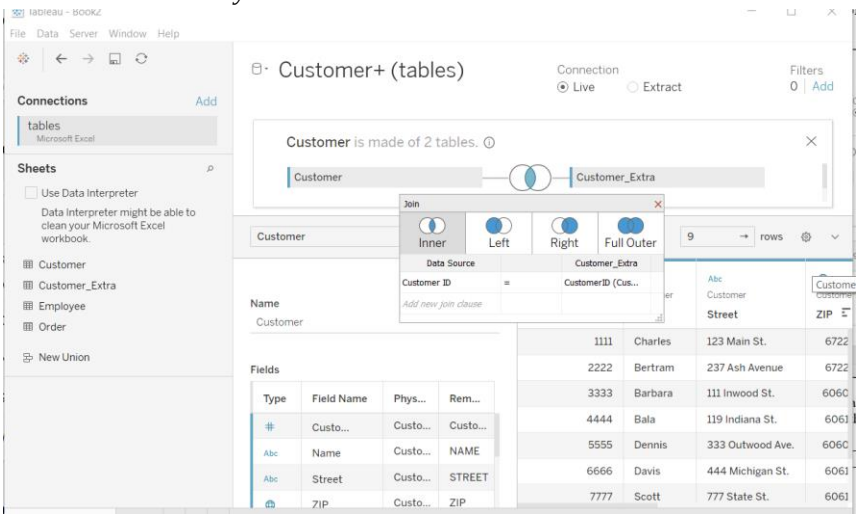
1. Go back to the "Data Source" tab.
2. Double-click on the logical "Customer" table to view the underlying physical table.

Note that the *Customer* and *Customer_Extra* tables are only logically connected to create the visualization. The underlying data is in two separate logical tables, the *Customer* and the *Customer_Extra* tables. You will next create a new Tableau workbook based on a physical join of the data from the *Customer* and *Customer_Extra* tables.

1. Close the previous Tableau workbook (Problem 1 Tableau Logical.twbx).
2. In a new Tableau workbook, go to the "Data Source" tab and connect to the Microsoft Excel file (tables.xlsx)
3. By default, the logical layer interface is displayed. This time, drag just the *Customer* table to the main pane.
4. Double-Click on the *Customer* table from the main pane of the logical layer to open up the interface to edit the physical layer
5. Next, you will create an inner join of the *Customer* table with the *Customer_Extra* table at this physical layer. Using this interface, drag the *Customer_Extra* table to the main pane. Once the *Customer_Extra* table is dragged to the right side of the main pane, Tableau automatically selects the "inner join" to join the two tables as shown below in Figure C6. At this point, you can change the join type or the fields that are joined together if desired. You have now created a physical join between the two tables and can use the joined data to create another visualization.

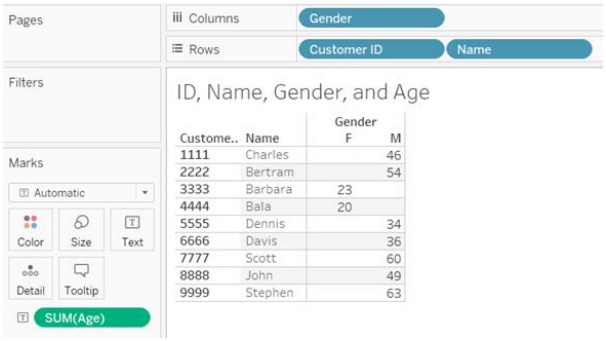
Figure C6

Tableau Automatically Selects an Inner Join



6. Go to Sheet 1 to create a visualization that lists the CustomerID, Name, Gender, and Age of all the customers. (For the age, you can move the "Age" measure to the "Detail" mark card.) Rename this worksheet as "ID, Name, Gender, and Age" as shown in Figure C7.

Figure C7
Renamed Worksheet in Tableau



Custome..	Name	Gender F	Gender M
1111	Charles		46
2222	Bertram		54
3333	Barbara	23	
4444	Bala	20	
5555	Dennis		34
6666	Davis		36
7777	Scott		60
8888	John		49
9999	Stephen		63

7. Export this Tableau Workbook as "Problem 1 Tableau Physical.twbx."

Problem 2: Join of the Customer Table and the Order Table

The join in Problem 1 is based on an INNER JOIN in that only matching rows in the two tables are returned. In Problem 2, you will analyze the case where there are matching rows, left-only rows, and right-only rows are needed.

As you may recall, there are customers in the Customer table that have not placed any orders. If your data visualizations are based on a physical layer, i.e. an INNER JOIN of the Customer table with the Order table, then you cannot illustrate the left-only rows or the right-only rows. (Recall that the inner join does not return any left-only rows or right-only rows.) In this problem, you will see the four types of joins using the logical layer; and joins based on the physical layer.

Creating the Logical Layer

Before you create the four joins, you must import the data. Because you will be creating visualizations from two tables, you will import the *Customer* and *Order* tables as logical tables.

1. Create the logical layer by dragging the *Customer* table to the left side and the *Order* table to the right side. Take the default relationship detected by Tableau.

Problem 2.1: Inner Join of Customer and Order Table

In this problem, you want to have an inner join of the *Customer* and *Order* tables. You will achieve this purpose by creating a visualization displaying the CustomerID (*Customer* table), the CustomerID (*Order* table), and the OrderID (*Order* table). Only matching rows should be returned. This confirmed that an inner join is performed by Tableau.

1. Create the visualization and rename it as "Customer Inner Join Order."

Problem 2.2: Customer LEFT OUTER JOIN Order Table

From Problem 2.1 above, you can see that when only dimension fields from two tables are used to build a visualization, an inner join of the tables is triggered and the matching rows are returned. The user does not specify the type of join. Instead, the join type is automatically triggered within Tableau. In a left outer join of the Customer and the Order table, you would like all of the matching rows returned, as well as any left-only rows (matching rows plus left-only rows). You would like all customers displayed, even those without orders.

Note that if a measure (numeric field) from a table is used in a visualization, the join type returning every row in that table will be used. In other words, a left outer join will be triggered if the measure is from the left table, and a right outer join will be triggered if the measure is from the right table.

You will therefore trigger Tableau to perform a LEFT OUTER JOIN by including a measure from the *Customer* (Left) table.

1. To create the visualization, add the "CustomerID" (*Customer*), "Customer ID" (*Order*), and "OrderID" (*Order*) as before on the Rows shelf. Then move the Customer (Count) field from the Customer table to the "Text" card. (Note that the green hashtag # indicates that this field is a continuous measure.) This will add the row count value of "1" to all of the rows from the *Customer* table (even those without a match.)
2. Save this visualization as "Customer LEFT OUTER JOIN Order."

Problem 2.3: Customer RIGHT OUTER JOIN Order Table

In a right outer join of the Customer and the Order table, you would like all of the matching rows returned, as well as any right-only rows (matching rows + right-only rows). You would like all orders displayed, even those without an associated customer.

1. To create this RIGHT OUTER JOIN, include the "CustomerID" (Customer), "Customer ID" (Order), and "OrderID" (Order) as before on the Rows shelf. Then move the Order (Count) field from the Order table to the "Text" card. The Order(Count) field is a numeric field, which will trigger Tableau to display all of the rows in the Order table. This adds the row count value of "1" to all of the rows from the Order table (even those without a match.) Recall though that in our data, all of the orders are associated with a customer. Therefore, the same rows will be returned as that of the INNER JOIN.
2. Save this visualization as "Customer RIGHT OUTER JOIN Order."

Problem 2.4: Customer FULL OUTER JOIN Order Table

In a full outer join of the Customer and the Order table, you would like all of the matching rows returned, as well as any left-only rows and right-only rows (matching rows + left-only rows + right-only rows).

1. Create a visualization that represents a full outer join of the Customer and the Order table. Save this visualization as "Customer FULL OUTER JOIN Order." Remember that to trigger all rows from both tables, you need to include a measure (numeric field) from each table, i.e., Customer (Count) and Order (Count).
2. Save this visualization as "Customer FULL OUTER JOIN Order
3. Export this Tableau workbook as "Problem 2 Logical Layer.twbx."

Physical Layer

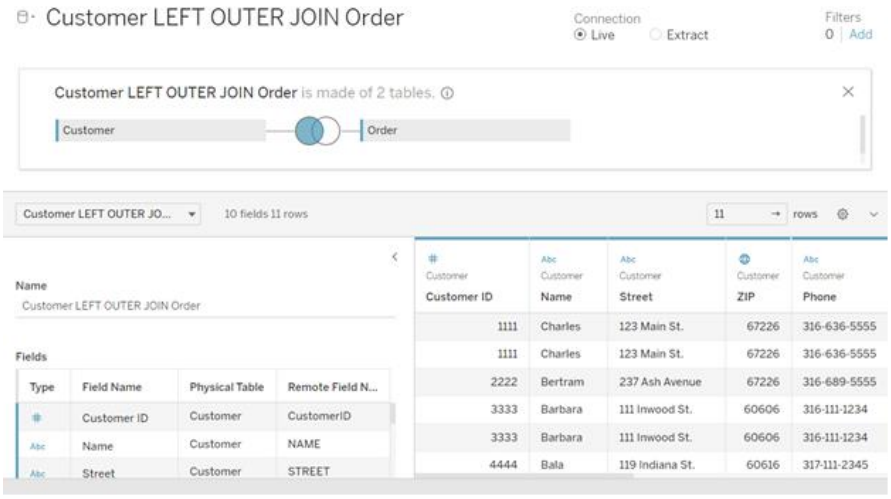
In Problems 2.1-2.4, you created the visualizations based on the logical representation. In Tableau, you can also create the visualization based on the specific data that you join as part of the physical layer.

You can rework the left outer join from Problem 2. Instead of using the logical layer, you will create a physical layer based on the Customer LEFT OUTER JOIN Order. Once this join is complete, you can then create visualizations based on that data.

1. Open a new Tableau Workbook.
2. Connect to the Microsoft Excel data file (tables.xlsx)
3. Drag the Customer table to the Main Pane
4. Double-click on the Customer table to open up the physical layer interface
5. On the physical layer interface, drag the Order table to the right side
6. Select the "LEFT" option to create a left outer join
7. Rename this logical Table as "Customer Left Outer Join Order." As you can see in Figure C8 below, the logical layer now consists of the *Customer* table LEFT OUTER JOIN *Order* table. If you browse through the data, you can see that there are 11 rows, which consists of the eight matching rows plus the three left-only rows.

Figure C8

Joining Tables in Tableau's Physical Layer



8. Create a visualization displaying the CustomerID (*Customer* table), the CustomerID (*Order* table), and the OrderID (*Order* table). Note that you do not have to add a measure field from the customer table to trigger the inclusion of the left-only rows. As the underlying data from the physical layer already is based on the LEFT OUTER JOIN, the three left-only rows are returned. However, you are no longer able to create a

visualization based on other join types of the Customer and the Order table (as the LEFT OUTER JOIN relationship has been hard-coded into the physical layer.)

9. Rename this visualization "Customer LEFT OUTER JOIN Order (Physical Layer)"
10. Export this Tableau workbook as "Problem 2 Physical Layer.twbx"

Key Takeaways:

- The data used to create visualizations can be represented at both the logical layer and the physical layer
- When importing data into Tableau, it is important to understand when and where the joins are physically occurring.
- Defining relationships at the logical layer gives the user the flexibility to build visualization requiring different join types with the same data source.
- Whenever a number is retrieved in a visualization, all rows of the table in which the number is stored will be returned. There is no need to worry about missing values inadvertently.
- As part of the Tableau Data Model, a user can still define a specific join type of the data at the Physical Layer as necessary. However, this could lead to limitations:
    - If a user defines an inner join at the physical layer, only the matching rows are available. This means that the left-only rows and right-only rows are no longer available to be used in visualizations as they are no longer included in the data set.
    - For example, it might not be obvious that the employees not associated with any orders are no longer included in the underlying data.

Problem 3: Joining the Employee Table and the Order Table

The join of the Employee Table and the Order Table is interesting because there are employees who are not associated with orders, as well as are orders that are not associated with employees.

Problem 3.1: Physical Layer Joins

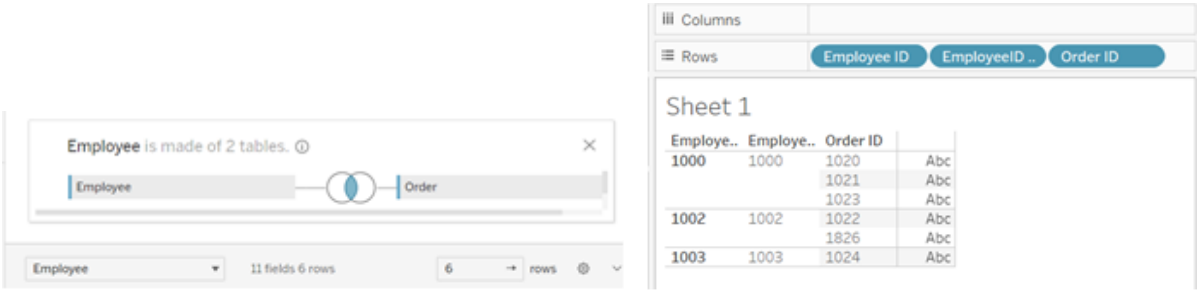
First, use the physical layer of Tableau to join the Employee and the Order tables. Then answer the following questions based on the Employee and the Order tables.

1. Which set of Employees (EmployeeID) have placed orders?
2. Which set of Employees (EmployeeID have NOT placed orders?
3. Which set of Orders (OrderID) are associated with employees?
4. Which set of Orders (OrderID) are not associated with employees?
5. Number of Matching Rows
6. Left-Only Rows (Employee as the "Left" table)
7. Right-Only Rows (Order as the "Right" table)
8. How many rows would you expect to be returned in an "Inner Join"?
9. How many rows would you expect to be returned in a LEFT JOIN of Employee (left table) and Order (right table)?
10. How many rows would you expect to be returned in a RIGHT JOIN of Employee (left table) and Order (right table)?
11. How many rows would you expect to be returned in a FULL OUTER JOIN?

In this part of the exercise, you will be creating a join in the Physical Layer for each of the four join types.

1. Create a new Tableau workbook. Using the physical layer interface, create the following types of joins in Tableau in the "Data Source" tab of the project. Create a simple visualization with the Employee ID (Employee), Employee ID (Order) and OrderID (Order). Note that you only have to create the visualization one time. As the physical layer is modified (e.g., changed from an "inner join" to a "left join"), the results of the visualization automatically change too since the visualization is based on the underlying data model. Screenshots of the physical layer model a visualization built on it for the Employee INNER JOIN Order appear in Figure C9 as an example for you to follow.

Figure C9
A Physical Layer Model and Visualization for Employee INNER JOIN Order



2. Using the example in Figure C9 as a guide, take screenshots of the physical layer model and a visualization built on it for Employee LEFT JOIN Order.

Problem 3.2: Relating the Employee and Order Table at the Logical Layer

Triggering an INNER JOIN
You will change the Data Source to create a Logical Link between the *Employee* table and the *Order* table.

1. Delete the physical layer relationships defined in the previous step.
2. Create a new logical layer between the Employee and the Order table. Rename this data source as "Employee and Order"
3. Once this data source has been changed to the logical relationship, what does your visualization in "Sheet1" look like?

Key Takeaways:

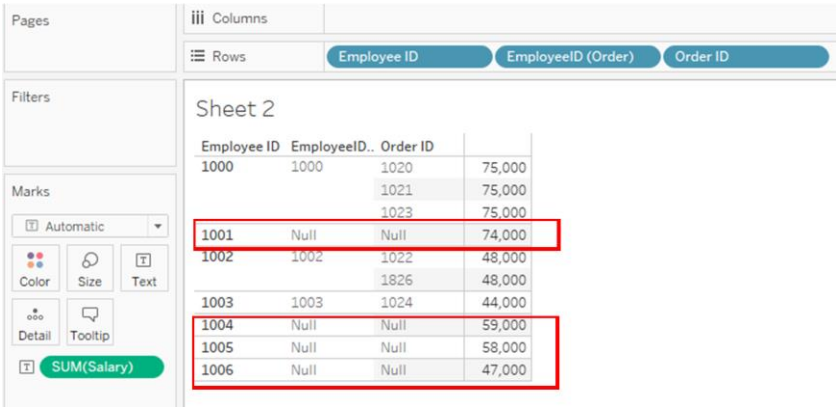
- The above visualization draws from the *Employee* Table (Employee ID), as well as from the *Order* Table (EmployeeID, OrderID)
- When dimensions from two tables are used to build a visualization, an inner join of the tables is triggered, and the Matching Rows are returned.
- The user does not specify the type of join. Instead, the join type is automatically triggered within Tableau.
- In the logical layer, each table in the relationship can be used independently to build visualizations as long as only fields in the same table are used to build a visualization. this added flexibility does not exist if the join is executed in the physical layer.

LEFT/RIGHT OUTER JOINS

In this step, you will build a visualization that triggers a left or right outer join. Remember that the tables have been connected at the logical layer and that you are not manually creating the joins.

1. Add another sheet. From the Employee table, drag Employee ID to the Rows shelf. From the Order table, drag EmployeeID (Order), and Order ID to the Rows Shelf. This results in an INNER JOIN of the two tables.
2. Now add a measure (numeric field) from the Employee table. You will add the "Salary" field, which is a numeric measure from the Employee table. Move "Salary" to the Text Marks card. The visualization should resemble Figure C10, which shows every row of the Employee table, including the four rows without a matching EmployeeID (outlined in red). This is an example of "Employee LEFT OUTER JOIN Order."

Figure C10
Employee LEFT OUTER JOIN Order



Remember, if a measure (such as this numerical field) from a table is used in the visualization, the join type returning every row in that table will be used. In other words, a left outer join will be triggered if the measure is from the left table, and a right outer join will be triggered if the measure is from the right table. Tableau does not differentiate between the left or right table.

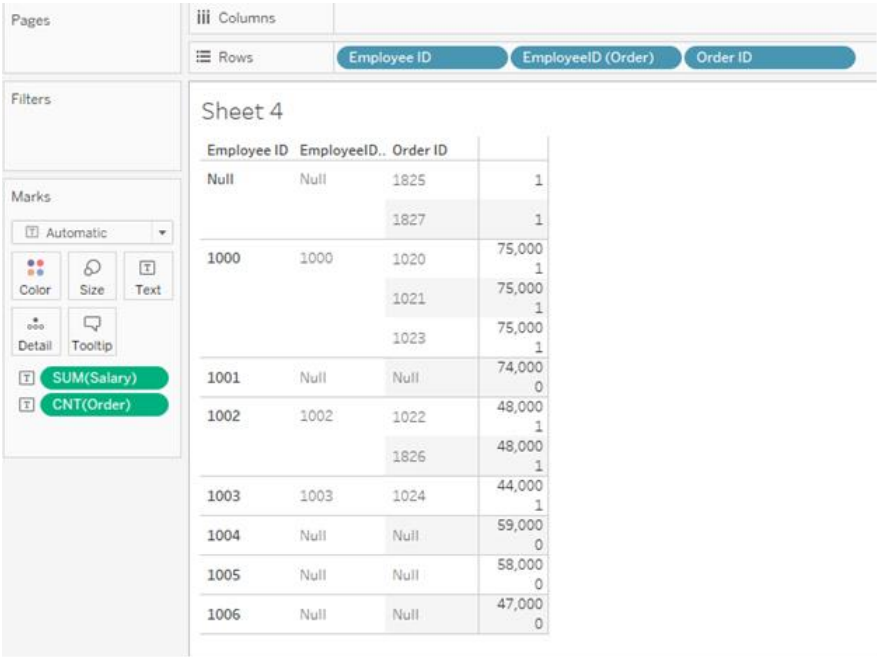
1. Next, add a value from the Order Table (instead of the Employee Table). Add another sheet (Sheet 3). From the Employee table, drag Employee ID to the Rows shelf. From the Order table, drag EmployeeID (Order), and Order ID to the Rows Shelf. This results in an INNER JOIN of the two tables.
2. Now add a measure field from the Order table. You will add the "Order (Count)" field, which is a continuous measure of the count of the number of records from the Order table. Move "Order (Count)" to the Text Marks card. Note that every row of the Order table is displayed, including the two rows without a matching EmployeeID. This is an example of "Employee RIGHT OUTER JOIN Order."

FULL OUTER JOIN

In this step, you will build a visualization that triggers a FULL OUTER JOIN. Remember that the data has been joined at the logical layer and that you are not manually creating the joins. Also remember that a FULL OUTER JOIN returns the "matching rows" + "left-only rows" + "right-only rows."

1. Add another sheet (Sheet 4). From the Employee table, drag Employee ID to the Rows shelf. From the Order table, drag EmployeeID (Order), and Order ID to the Rows Shelf. This results in an INNER JOIN of the two tables ("Matching Rows).
2. Now add a measure (numeric field) from the *Employee* table. You will add the "Salary" field, which is a numeric measure from the *Employee* table. Move "Salary" to the Text Marks card. This will add the "Left-Only" Rows.
3. Add a measure (numeric field) from the *Order* table. You will add the "Order (Count)" field, which is a continuous measure of the count of the number of records from the Order table. Move "Order (Count)" to the Text Marks card. This will add the "Right-Only" Rows. The visualization should match Figure C11.

Figure C11
Visualization that Triggers a FULL OUTER JOIN



Employee ID	EmployeeID..	Order ID	
Null	Null	1825	1
		1827	1
1000	1000	1020	75,000 1
		1021	75,000 1
		1023	75,000 1
1001	Null	Null	74,000 0
1002	1002	1022	48,000 1
		1826	48,000 1
1003	1003	1024	44,000 1
1004	Null	Null	59,000 0
1005	Null	Null	58,000 0
1006	Null	Null	47,000 0

4. Export this Tableau workbook as "Problem 3 – Logical Layer.twbx."

Key Takeaways:

1. If fields from different tables are used to build a visualization, a join will be triggered within Tableau.
2. When dimensions from two tables are used to build a visualization, an inner join of the tables are triggered.
3. If a measure (numerical field) from a table is used in the visualization, the join type that will return every row in that table will be used. In other words, a left outer join will be triggered if the measure is from the left table, and a right outer join will be triggered if the measure is from the right table. From Tableau visualization's perspective, it does not differentiate left or right table.
4. If matches are not found for dimension or dimensions from the other table, the values of these dimension fields are displayed as Null.

Problem 4 Query with Visualization

Create a Data Model based on the four tables to answer the six queries.

1. Create a new Tableau workbook "Problem 4.twbx"
2. Create a Logical Layer as illustrated below. By creating relationships among the four tables, you will be able to create a separate visualization for each query.
3. Create a separate visualization for each query. Provide a screenshot of your visualization in the space below. Save each visualization in a separate worksheet, e.g. "Query 4.1."

Query4.1: For all customers, list their CustomerID, Name, Gender, and Age. Only include the result if there is a corresponding match in the *Customer_Extra* table. *Hint: Remember that for these two tables all four joins return the same results.*

Query 4.2: For all customers, list their CustomerID, Name, Gender, and Age. If there is no associated gender or age, then a NULL value should be returned for the unmatched field(s) in that particular row. *Hint: Because all Customers have a corresponding row in the Customer_Extra table, the results will be the same as Query 4.1.*

Query 4.3: Display the OrderID, CustomerID, and Customer "Name" associated with each of the orders in the ORDER table. If there is no associated Customer "Name," then a NULL value should be returned as the "Name" for that particular row.

Query 4.4: Display all OrderID and EmployeeId, and Employee "Name" associated with each of the orders in the ORDER table. If there is no associated Employee "Name" for any order, then a NULL value should be returned as the "Name" for that particular row.

Query 4.5: Display the customers (CustomerID and NAME) and the order ID if they have made any orders. Include customers who have NOT placed any orders in the *Orders* table. Note: In Tableau since you are using the Logical Layer and not making the joins directly, you can include in the results all the Customers in the *Customer* table with NULLS associated with the OrderID for those that have not placed an order.

Query 4.6: Display all employees (EmployeeID and NAME) including those who are NOT associated with any orders in the *Orders* table. Note: In Tableau since you are using the Logical Layer and not making the joins directly, you can include all the Employees in the *Employee* table with NULLS associated with the OrderID for those that are not associated with an order.

Export this Tableau workbook (Problem 4.twbx) and submit the files as instructed.

AIS Educator Journal Editorial Board

The *AIS Educator Journal* is published by the AIS Educator Association:
President: Ann O'Brien, University of Wisconsin
Vice President and President Elect: Cynthia Frownfelter-Lohrke, Samford University
Secretary: Cheryl L. Dunn, Grand Valley State University
Treasurer: Kristian Mortenson, University of St. Thomas
Past-President: Dawna Drum, Western Washington University