

**Faculty of Engineering**

Course of Study Data Engineering & Consulting

**Topic**

Data & ETL Pipelines

**Final Report**

Submitted

in the winter semester 24/25

By

Daniel Saiger

# Table of Contents

[Table of Contents II](#_Toc188742450)

[List of Figures IV](#_Toc188742451)

[1. Question No. 1 1](#_Toc188742452)

[2. Question No. 2 2](#_Toc188742453)

[3. Question No. 3 3](#_Toc188742454)

[3.1 Question No. 3.1. 3](#_Toc188742455)

[3.2 Question No. 3.2. 3](#_Toc188742456)

[3.3 Question No. 3.3. 4](#_Toc188742457)

[3.4 Question No. 3.4. 4](#_Toc188742458)

[3.5 Question No. 3.5. 5](#_Toc188742459)

[4. Question No. 4 6](#_Toc188742460)

[4.1 Question No. 4.1. 6](#_Toc188742461)

[4.2 Question No. 4.2. 6](#_Toc188742462)

[4.3 Question No. 4.3. 7](#_Toc188742463)

[4.4 Question No. 4.4. 7](#_Toc188742464)

[4.5 Question No. 4.5. 8](#_Toc188742465)

[5. Question No. 5 9](#_Toc188742466)

[5.1 Question No. 5.1. 9](#_Toc188742467)

[5.2 Question No. 5.2. 11](#_Toc188742468)

[6. Question No. 6 13](#_Toc188742469)

[7. Question No. 7 14](#_Toc188742470)

[8. Question No. 8 15](#_Toc188742471)

[9. Question No. 9 16](#_Toc188742472)

[10. Question No. 10 17](#_Toc188742473)

[11. Question No. 11 18](#_Toc188742474)

[12. Question No. 12 19](#_Toc188742475)

[References IV](#_Toc188742476)

# 

# List of Figures

[Figure 1: Dimension Table before running stored procedure 1](#_Toc188742477)

[Figure 2: Dimension Table after running stored procedure 1](#_Toc188742478)

[Figure 3: Table before removing duplicates 2](#_Toc188742479)

[Figure 4: Table after removing duplicates 2](#_Toc188742480)

[Figure 5: Result set Q3.1. 3](#_Toc188742481)

[Figure 6: Result set Q3.2. 3](#_Toc188742482)

[Figure 7: Result set Q3.3. 4](#_Toc188742483)

[Figure 8: Result set Q3.4. 4](#_Toc188742484)

[Figure 9: Result set Q3.5. 5](#_Toc188742485)

[Figure 10: Result Set Q4.1. 6](#_Toc188742486)

[Figure 11: Result Set Q4.2. 6](#_Toc188742487)

[Figure 12: Result Set Q4.3. 7](#_Toc188742488)

[Figure 13: Resul Set Q4.4. 7](#_Toc188742489)

[Figure 14: Result Set Q.4.5. 8](#_Toc188742490)

[Figure 15: Result Set Q8 15](#_Toc188742491)

[Figure 16: Hadoop vs. Spark 16](#_Toc188742492)

[Figure 17: Example 1 NF 19](#_Toc188742493)

[Figure 18: Example 2NF 20](#_Toc188742494)

[Figure 19: Example 3NF 21](#_Toc188742495)

[Figure 20: Example and fix of Insertion Anomaly 22](#_Toc188742496)

[Figure 21: Example and fix of Update Anomaly 23](#_Toc188742497)

[Figure 22: Example and fix of Deletion Anomaly 23](#_Toc188742498)

# Question No. 1

Dimension Table before running stored procedure:

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Figure 1: Dimension Table before running stored procedure

Dimension Table after running stored procedure and **changing the value of item 3**:

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Figure 2: Dimension Table after running stored procedure

**SQL Query:** See file *query\_q1.sql*

# Question No. 2

Table before removing duplicates:

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Figure 3: Table before removing duplicates

Table after removing duplicates:

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Figure 4: Table after removing duplicates

**SQL Query:** See file *query\_q2.sql*

# Question No. 3

## Question No. 3.1.

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Figure 5: Result set Q3.1.

Query: See file *queries\_q3.sql*

## Question No. 3.2.

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Figure 6: Result set Q3.2.

**SQL Query:** See file *queries\_q3.sql*

## Question No. 3.3.

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Figure 7: Result set Q3.3.

**SQL Query:** See file *queries\_q3.sql*

## Question No. 3.4.

Ein Bild, das Text, Screenshot, Zahl, Karte Menü enthält.

Automatisch generierte Beschreibung

Figure 8: Result set Q3.4.

**SQL Query:** See file *queries\_q3.sql*

## Question No. 3.5.

Ein Bild, das Text, Screenshot, Zahl, Karte Menü enthält.

Automatisch generierte Beschreibung

Figure 9: Result set Q3.5.

**SQL Query:** See file *queries\_q3.sql*

# Question No. 4

## Question No. 4.1.

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Figure 10: Result Set Q4.1.

**Pyspark Query:** See file *pyspark\_notebook.ipynb*

## Question No. 4.2.

Ein Bild, das Text, Screenshot, Zahl, Schrift enthält.

Automatisch generierte Beschreibung

Figure 11: Result Set Q4.2.

**Pyspark Query:** See file *pyspark\_notebook.ipynb*

## Question No. 4.3.

Ein Bild, das Text, Quittung, Schrift, Screenshot enthält.

Automatisch generierte Beschreibung

Figure 12: Result Set Q4.3.

**Pyspark Query:** See file *pyspark\_notebook.ipynb*

## Question No. 4.4.

Ein Bild, das Text, Screenshot, Dokument, Schrift enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Quittung, Schrift, weiß enthält.

Automatisch generierte Beschreibung

Figure 13: Resul Set Q4.4.

**Pyspark Query:** See file *pyspark\_notebook.ipynb*

## Question No. 4.5.

Ein Bild, das Text, Screenshot, Zahl, Karte Menü enthält.

Automatisch generierte Beschreibung

Figure 14: Result Set Q.4.5.

**Pyspark Query:** See file *pyspark\_notebook.ipynb*

# Question No. 5

## Question No. 5.1.

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Elektronik, Screenshot, Software enthält.

Automatisch generierte Beschreibung

## Question No. 5.2.

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung

**SQL Query for incremental Load:**

*SELECT CustomerKey, FirstName, LastName, BirthDate, EmailAddress, YearlyIncome*

*FROM [dbo].[dim\_customer]*

*WHERE [YearlyIncome] > 60000*

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

# Question No. 6

Ein Bild, das Text, Diagramm, Screenshot, Design enthält.

Automatisch generierte Beschreibung

**Diagram can be opened here:** <https://dbdiagram.io/d/Data_etl_pipelines_project-671917c897a66db9a305f05b>

# Question No. 7

**Spark Driver**

Program which declares the SparkContext. Its role is to split the user program into a series of tasks which can be distributed across the cluster. It also coordinates the execution of tasks and communicates with the Cluster Manager to allocate resources for the application (Chambers & Zaharia, 2015, pp. 20–24).

* Spark Driver is the central coordinating entity of the Spark Application

**Spark Executor**

Process which task it is to run the tasks in parallel on worker nodes in the cluster →The Driver launches the Executor process, which runs the tasks that are assigned to him by the driver (Chambers & Zaharia, 2015, pp. 20–24).

Responsible for processing the data and executing the code in parallel → Runs the user defined Spark Code and performs the calculations and transformations on the data (Medium, 2024)

**Worker Node**

Worker nodes are the physical or virtual machines in the Spark cluster that host executors. They communicate with the cluster manager to allocate resources for executors. They track and report the status of executors and tasks back to the driver (Medium, 2024).

# Question No. 8

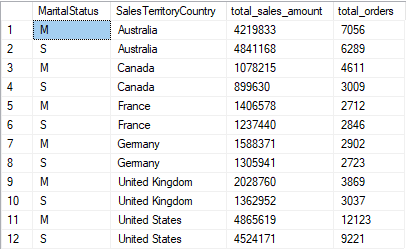


Figure 15: Result Set Q8

**SQL Query:** See file *query\_q8.sql*

# Question No. 9

Spark enhances Hadoop's MapReduce by keeping data in memory for processing, unlike MapReduce, which writes data to the disks between steps. This makes Spark much faster especially for small data amounts (IBM, 2021a).

Instead of the two-stage execution process used in MapReduce, Spark uses a Directed Acyclic Graph (DAG) to manage task scheduling and node coordination in a Hadoop cluster. The DAG also helps with fault tolerance by replaying saved operations to recover data to a previous state if needed (IBM, 2021a).

|  |  |
| --- | --- |
| **Hadoop** | **Spark** |
| MapReduce for Batch oriented processing | Resilient distributed Datasets (RDDs) for both batch and stream processing |
| Writes back the data to the hard drives for every data processing step 🡪 Slower | Uses in-memory computation 🡪 faster |
| Batch processing only | Batch, real-time and interactive processing |
| Has to use the Hadoop distributed file system (HDFS) | Can use the HDFS but is storage agnostic 🡪 can also use e.g. AWS S3 |

Figure 16: Hadoop vs. Spark[[1]](#footnote-1)

# Question No. 10

1. **Broadcast Join:**

Used when one of the datasets is small enough to fit in memory on each worker node. Spark broadcasts the smaller dataset across all nodes, so each partition of the larger dataset can join with the smaller dataset locally without any need for shuffling data across the network (Medium, 2024b).

**Use Case:**

Joining a small dataset with a large dataset

1. **Shuffle Merge Join:**

Default join type in Spark for large datasets. Data gets shuffled across the cluster to group matching keys together → all rows with the same key are on the same partition. After that the datasets on each partition are merged to complete the join (Medium, 2024b).

**Use Case:**

Joining two large datasets → None of them can be broadcasted

1. **Sort Merge Join:**

Both datasets are first sorted by the join key. After that, the datasets are merged. It is efficient when both datasets are already sorted or can be efficiently sorted (Medium, 2024b).

**Use Case:**

* When both datasets are already sorted
* Optimal for large datasets → sorting reduces the complexity of the join

# Question No. 11

A surrogate key is an artificially generated key. It is used when the data has no natural key for unique identification. While the business key is generated in the source system of the data, a surrogate key has no relationship to the source system and is generated e.g. by the ETL process (IBM, 2021b)

**Benefits:**

* Stable and does not change over time, business key might change over time (Sisense, 2024)
* Essential for implementing SCDs. They allow differentiation between multiple historical versions of a record while maintaining a unique identifier for each (Sisense, 2024)
* Numeric, small in size (e.g., integers), business keys can be large and complex (e.g. composite keys) (Sisense, 2024)
* Ensures uniqueness within the database, business keys may not be unique across all systems (Sisense, 2024)

# Question No. 12

**First Normal Form - 1NF:**

* A single cell holds only one value
* Primary Key for identification
* No duplicated rows or columns
* Each column has only value for each row in the table (Visual Paradigm, 2023)

Ein Bild, das Text, Zahl, Schrift, Screenshot enthält.

Automatisch generierte Beschreibung

Figure 17: Example 1 NF [[2]](#footnote-2)

**Second Normal Form - 2NF:**

* It is already in 1NF
* No partial dependency → All non-key attributes are fully dependent on a primary key (Visual Paradigm, 2023)

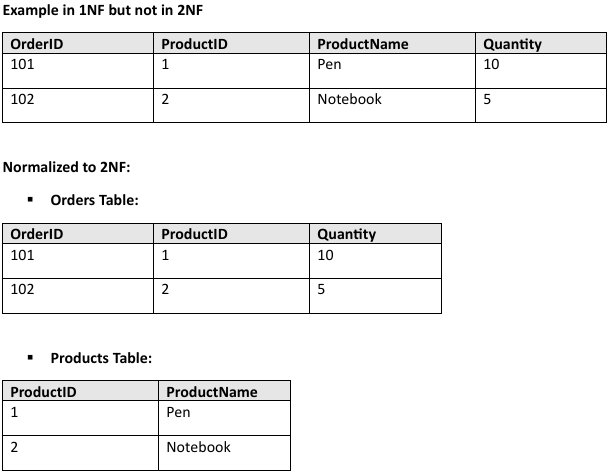


Figure 18: Example 2NF [[3]](#footnote-3)

**Third Normal Form - 3NF:**

* Is already in 2NF
* No transitive partial dependency(Visual Paradigm, 2023)

Ein Bild, das Text, Screenshot, Quittung, Schrift enthält.

Automatisch generierte Beschreibung

Figure 19: Example 3NF [[4]](#footnote-4)

**Insertion Anomaly**

Occurs when adding new data requires unnecessary information (Javatpoint, 2025)

**Example:** We want to add a new department (Marketing) without any employees. We can’t do so without assigning an employee in a non-normalized schema:

Ein Bild, das Text, Reihe, Screenshot, Zahl enthält.

Automatisch generierte Beschreibung

With a normalized Schema, the Department Table can simply be updated, without affecting the other tables:

Ein Bild, das Text, Screenshot, Reihe, Schrift enthält.

Automatisch generierte Beschreibung

Figure 20: Example and fix of Insertion Anomaly[[5]](#footnote-5)

**Update Anomaly**

Occurs when updating a value requires changes in multiple rows (Javatpoint, 2025)

**Example:** We want to change the name of the HR department to “Human Resources”. In a non-normalized schema multiple rows have to be updated → risk of inconsistencies

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Figure 21: Example and fix of Update Anomaly [[6]](#footnote-6)

**Deletion Anomaly**

Occurs when deleting a row inadvertently removes necessary data (Javatpoint, 2025)

**Example:** Deleting the only order of “Pen” would result in a complete loss of the product data in a non-normalized schema:

Ein Bild, das Text, Reihe, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

With a normalized schema the product information is stored separate from the orders. A deletion of orders therefore does not affect the product information:

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

Figure 22: Example and fix of Deletion Anomaly [[7]](#footnote-7)

# References

Amazon Web Services. (2024). *The difference between Hadoop and Spark.* Retrieved January 16, 2025, from <https://aws.amazon.com/de/compare/the-difference-between-hadoop-vs-spark/>

Chambers, B. & Zaharia, M. (2018). *Spark: The definitive guide: Big data processing made simple.* O'Reilly Media

IBM. (2021a). Hadoop vs. Spark: What’s the difference? IBM. Retrieved January 16, 2025, from <https://www.ibm.com/think/insights/hadoop-vs-spark>

IBM. (2021b). Surrogate Keys. Retrieved January 13, 2025, from <https://www.ibm.com/docs/en/ida/9.1.2?topic=keys-surrogate>

Javatpoint. (2025). *Anomalies in DBMS.* Retrieved January 02, 2025, from <https://www.javatpoint.com/anomalies-in-dbms>

Medium. (2024a). *What are the specific roles of Spark Driver and Executor?* Retrieved January 16, 2025, from <https://medium.com/sparkbyexamples/what-are-the-specific-roles-of-spark-driver-and-executor-a17f90c7e7fd>

Medium. (2024b). *Understanding Broadcast Join and Normal Shuffle-Sort-Merge Join in Apache Spark.* Retrieved January 05, 2025, from <https://sachin-s1dn.medium.com/understanding-broadcast-join-and-normal-shuffle-sort-merge-join-in-apache-spark-22f60cb1a7f0>

Sisense. (2023). *When (and how) to use surrogate keys.* Retrieved January 04, 2025, <https://www.sisense.com/blog/when-and-how-to-use-surrogate-keys/>

Visual Paradigm. (2023). *A Comprehensive Guide to Database Normalization with Examples.* Retrieved January 05, 2025, from <https://guides.visual-paradigm.com/a-comprehensive-guide-to-database-normalization-with-examples/>

1. Own figure based on Amazon Web Services (2024) [↑](#footnote-ref-1)
2. Own figure [↑](#footnote-ref-2)
3. Own figure [↑](#footnote-ref-3)
4. Own figure [↑](#footnote-ref-4)
5. Own figure based on Javatpoint, 2025 [↑](#footnote-ref-5)
6. Own figure based on Javatpoint, 2025 [↑](#footnote-ref-6)
7. Own figure based on Javatpoint, 2025 [↑](#footnote-ref-7)