



Hochschule
Albstadt-Sigmaringen

Albstadt-Sigmaringen University

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

Matr. Nr.: 107611

Table of Contents

Table of Contents	II
List of Figures	III
1. Question No. 1	1
2. Question No. 2	2
3. Question No. 3	3
3.1 Question No. 3.1	3
3.2 Question No. 3.2	3
3.3 Question No. 3.3	4
3.4 Question No. 3.4	4
3.5 Question No. 3.5	5
4. Question No. 4	6
4.1 Question No. 4.1	6
4.2 Question No. 4.2	6
4.3 Question No. 4.3	7
4.4 Question No. 4.4	7
4.5 Question No. 4.5	8
5. Question No. 5	9
5.1 Question No. 5.1	9
5.2 Question No. 5.2	11
6. Question No. 6	13
7. Question No. 7	14
8. Question No. 8	15
9. Question No. 9	16
10. Question No. 10	17
11. Question No. 11	18
12. Question No. 12	19
References	IV

List of Figures

Figure 1: Dimension Table before running stored procedure	1
Figure 2: Dimension Table after running stored procedure	1
Figure 3: Table before removing duplicates	2
Figure 4: Table after removing duplicates	2
Figure 5: Result set Q3.1.	3
Figure 6: Result set Q3.2.	3
Figure 7: Result set Q3.3.	4
Figure 8: Result set Q3.4.	4
Figure 9: Result set Q3.5.	5
Figure 10: Result Set Q4.1.....	6
Figure 11: Result Set Q4.2.....	6
Figure 12: Result Set Q4.3.....	7
Figure 13: Result Set Q4.4.....	7
Figure 14: Result Set Q.4.5.....	8
Figure 15: Result Set Q8.....	15
Figure 16: Hadoop vs. Spark.....	16
Figure 17: Example 1NF	19
Figure 18: Example 2NF	20
Figure 19: Example 3NF	21
Figure 20: Example and fix of Insertion Anomaly	22
Figure 21: Example and fix of Update Anomaly	23
Figure 22: Example and fix of Deletion Anomaly	23

1. Question No. 1

Dimension Table before running stored procedure:

	DimID	ID	Name	Value	StartDate	EndDate	IsCurrent
1	1	1	Item1	Value1	2024-01-01 00:00:00.000	NULL	1
2	2	2	Item2	Value2	2024-01-05 00:00:00.000	NULL	1
3	3	3	Item3	Value3	2024-01-05 00:00:00.000	NULL	1

Figure 1: Dimension Table before running stored procedure

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

	DimID	ID	Name	Value	StartDate	EndDate	IsCurrent
1	1	1	Item1	Value1	2024-01-01 00:00:00.000	NULL	1
2	2	2	Item2	Value2	2024-01-05 00:00:00.000	NULL	1
3	3	3	Item3	Value3	2024-01-05 00:00:00.000	2025-01-02 14:12:41.127	0
4	4	3	Item3	Value4	2025-01-02 14:12:41.127	NULL	1

Figure 2: Dimension Table after running stored procedure

SQL Query: See file *query_q1.sql*

2. Question No. 2

Table before removing duplicates:

	ID	Item	Value
1	1	Item1	Value1
2	2	Item2	Value2
3	3	Item3	Value3
4	4	Item3	Value3
5	5	Item1	Value1

Figure 3: Table before removing duplicates

Table after removing duplicates:

	ID	Item	Value
1	1	Item1	Value1
2	2	Item2	Value2
3	3	Item3	Value3

Figure 4: Table after removing duplicates

SQL Query: See file *query_q2.sql*

3. Question No. 3

3.1 Question No. 3.1.

	item_name	total_sales
1	Red Bull 12oz	1305700
2	K Cups Daily Chef Columbian Supremo	1245394
3	K Cups Original Donut Shop Med. Roast	1188843
4	K Cups Dunkin Donuts Medium Roast	1109760
5	Muscle Milk Protein Shake Van. 11oz	1050924
6	K Cups Folgers Lively Columbian	1042406
7	Honey Packets	1012995
8	K Cups - Starbuck's Pike Place	995456
9	K Cups -Organic Breakfast Blend	957516
10	K Cups - McCafe Premium Roast	956886

Figure 5: Result set Q3.1.

Query: See file *queries_q3.sql*

3.2 Question No. 3.2.

	item_name	total_quantity_sold
1	Pepsi - 12 oz cans	46837
2	Muscle Milk Protein Shake Van. 11oz	45665
3	Coke Classic 12 oz cans	45501
4	Diet Coke - 12 oz cans	45202
5	Sprite - 12 oz cans	45140
6	Diet Pepsi - 12 oz cans	23969
7	Nat.Valley PeanutButter Protein Bars	23958
8	Nabisco Classic Mix ccooki, cracker	23852
9	Red Bull 12oz	23740
10	Cascade Gel Packs Dishwasher	23648

Figure 6: Result set Q3.2.

SQL Query: See file *queries_q3.sql*

3.3 Question No. 3.3.

	man_country	total_quantity_sold	total_sales_amount
1	Lithuania	592105	72695487
2	poland	635611	69148989
3	India	730625	64332205
4	Germany	699848	61117160
5	China	362645	47526644

Figure 7: Result set Q3.3.

SQL Query: See file *queries_q3.sql*

3.4 Question No. 3.4.

	year	quarter	month	total_sales_amount
1	2014	1.00	1	2235320
2	2021	1.00	1	3874560
3	2014	1.00	2	4932895
4	2015	1.00	2	4999910
5	2020	2.00	6	5013817
6	2018	1.00	2	5114400
7	2017	1.00	2	5126175
8	2020	1.00	2	5208736
9	2018	3.00	9	5275388
10	2014	2.00	4	5287227
11	2019	2.00	6	5313940
12	2016	1.00	1	5323352
13	2017	2.00	6	5352373
14	2016	2.00	6	5354497
15	2016	4.00	11	5354805
16	2020	3.00	7	5366213
17	2019	1.00	2	5376138
18	2019	3.00	9	5381129
19	2017	3.00	9	5395294
20	2014	4.00	11	5447350

Figure 8: Result set Q3.4.

SQL Query: See file *queries_q3.sql*

3.5 Question No. 3.5.

	coustomer_key	name	count_used_trans_type
1	C000001	sumit	3
2	C000002	tammanne	3
3	C000003	kailash kumar	3
4	C000004	bhagwati prasad	3
5	C000005	ajay	3
6	C000006	silender	3
7	C000007	deepak	3
8	C000008	akhilesh	3
9	C000009	dipendra kumar	3
10	C000010	nitin	3
11	C000011	doodhnath pandit	3
12	C000012	aslam allam	3
13	C000013	rahul	2
14	C000014	jitender kumar	3
15	C000015	adnan	3
16	C000016	vijay	3
17	C000017	yogesh	3
18	C000018	kabir	3
19	C000019	sarvesh	3
20	C000020	rakesh sarkar	3

Figure 9: Result set Q3.5.

SQL Query: See file *queries_q3.sql*

4. Question No. 4

4.1 Question No. 4.1.

item_name	total_sales_amount
Red Bull 12oz	1305700
K Cups Daily Chef...	1245394
K Cups Original D...	1188843
K Cups Dunkin Don...	1109760
Muscle Milk Prote...	1050924
K Cups Folgers Li...	1042406
Honey Packets	1012995
K Cups Starbucks...	995456
K Cups Organic B...	957516
K Cups - McCafe P...	956886

Figure 10: Result Set Q4.1.

Pyspark Query: See file *pyspark_notebook.ipynb*

4.2 Question No. 4.2.

item_name	total_quantity_sold
Pepsi - 12 oz cans	46837
Muscle Milk Prote...	45665
Coke Classic 12 o...	45501
Diet Coke - 12 oz...	45202
Sprite - 12 oz cans	45140
Diet Pepsi - 12 o...	23969
Nat.Valley Peanut...	23958
Nabisco Classic M...	23852
Red Bull 12oz	23740
Cascade Gel Packs...	23648

Figure 11: Result Set Q4.2.

Pyspark Query: See file *pyspark_notebook.ipynb*

4.3 Question No. 4.3.

man_country	total_quantity_sold	total_sales_amount
Bangladesh	772031	13332668
India	730625	13151006
Lithuania	592105	11739725
poland	635611	10966500
Germany	699848	10948917

Figure 12: Result Set Q4.3.

Pyspark Query: See file *pyspark_notebook.ipynb*

4.4 Question No. 4.4.

year	quarter	month	total_sales_amount
2014	Q1	1	496549.25
2021	Q1	1	883772.25
2014	Q1	2	1122547.0
2020	Q1	2	1128964.0
2017	Q1	2	1144129.5
2018	Q1	2	1148761.5
2019	Q1	2	1149508.5
2015	Q1	2	1157159.75
2019	Q2	6	1158766.0
2016	Q2	6	1193281.75
2015	Q2	4	1193564.0
2018	Q2	4	1197510.0
2020	Q4	11	1200248.75
2019	Q2	4	1201298.0
2016	Q1	1	1202031.5
2019	Q4	12	1206605.5
2016	Q3	8	1208266.0
2020	Q2	4	1211492.0
2014	Q4	11	1211569.75
2016	Q4	10	1211723.5

Figure 13: Result Set Q4.4.

Pyspark Query: See file *pyspark_notebook.ipynb*

4.5 Question No. 4.5.

+-----+-----+-----+		
coustomer_key	name	count_used_trans_type
+-----+-----+-----+		
C000001	sumit	3
C000002	tammanne	3
C000003	kailash kumar	3
C000004	bhagwati prasad	3
C000005	ajay	3
C000006	silender	3
C000007	deepak	3
C000008	akhilesh	3
C000009	dipendra kumar	3
C000010	nitin	3
C000011	doodhnath pandit	3
C000012	aslam allam	3
C000013	rahul	2
C000014	jitender kumar	3
C000015	adnan	3
C000016	vijay	3
C000017	yogesh	3

Figure 14: Result Set Q.4.5.

Pyspark Query: See file *pyspark_notebook.ipynb*

5. Question No. 5

5.1 Question No. 5.1.

Pipeline Name:

The screenshot shows the configuration form for a pipeline named 'copy_from_adls_to_sql'. The 'Name' field is filled with 'copy_from_adls_to_sql' and has a 'Learn more' link. The 'Description' field is empty. The 'Activity state' is set to 'Activated' with a radio button. The 'Timeout' is set to '0.12:00:00'. The 'Retry' is set to '0'. The 'Retry interval (sec)' is set to '30'. The 'Secure output' and 'Secure input' checkboxes are both unchecked.

Name *	copy_from_adls_to_sql	Learn more
Description		
Activity state ⓘ	<input checked="" type="radio"/> Activated <input type="radio"/> Deactivated	
Timeout ⓘ	0.12:00:00	
Retry ⓘ	0	
Retry interval (sec) ⓘ	30	
Secure output ⓘ	<input type="checkbox"/>	
Secure input ⓘ	<input type="checkbox"/>	

Source Dataset:

The screenshot shows the configuration form for a source dataset named 'ADLS_to_SQL_source'. The 'Source dataset' dropdown is set to 'ADLS_to_SQL_source'. The 'File path type' is set to 'File path in dataset'. The 'Filter by last modified' section has 'Start time (UTC)' and 'End time (UTC)' fields. The 'Recursively' checkbox is checked. The 'Enable partitions discovery' checkbox is unchecked. The 'Max concurrent connections' and 'Skip line count' fields are empty. The 'Additional columns' section has a '+ New' button.

Source dataset *	ADLS_to_SQL_source	Open + New Preview data Learn more
File path type	<input checked="" type="radio"/> File path in dataset <input type="radio"/> Wildcard file path <input type="radio"/> List of files ⓘ	
Filter by last modified ⓘ	Start time (UTC)	End time (UTC)
Recursively ⓘ	<input checked="" type="checkbox"/>	
Enable partitions discovery ⓘ	<input type="checkbox"/>	
Max concurrent connections ⓘ		
Skip line count		
Additional columns ⓘ	+ New	

The screenshot shows the configuration form for a linked service named 'ADLS_to_SQL_Source_LS'. The 'File path' is set to 'source / Directory / DimCustomer.csv'. The 'Compression type' is set to 'No compression'. The 'Column delimiter' is set to 'Comma (,)'.

Linked service *	ADLS_to_SQL_Source_LS	Test connection Edit + New Learn more
File path	source / Directory / DimCustomer.csv Browse Preview data Detect format	
Compression type	No compression	
Column delimiter ⓘ	Comma (,)	
Row delimiter ⓘ	Default (\n, \r, or \r\n)	
Encoding ⓘ	Default(UTF-8)	
Quote character ⓘ	Double quote (")	
Escape character ⓘ	Backslash (\)	
First row as header ⓘ	<input checked="" type="checkbox"/>	
Null value ⓘ		

Sink Dataset:

Sink dataset *

ADLS_to_SQL_target

Open

New

Learn more

Write behavior

Insert

Upsert

Stored procedure

Bulk insert table lock ⓘ

Yes

No

Table option

Use existing

Auto create table ⓘ

Pre-copy script ⓘ

TRUNCATE TABLE dbo.dim_customer

Write batch timeout ⓘ

e.g. 00:30:00

Write batch size ⓘ

Max concurrent connections ⓘ

Linked service *

ADLS_to_SQL_target

Test connection

Edit

New

Learn more

Table

schema name

dim_customer

Preview data

Enter manually

Pipeline Runs:

<input type="checkbox"/>	pipeline_ADLS_to_SQL	1/2/2025, 4:29:56 PM	1/2/2025, 4:30:20 PM	24s	Manual trigger	✔ Succeeded
<input type="checkbox"/>	pipeline_ADLS_to_SQL	1/2/2025, 4:28:52 PM	1/2/2025, 4:29:11 PM	19s	Manual trigger	✔ Succeeded
<input type="checkbox"/>	pipeline_ADLS_to_SQL	1/2/2025, 4:25:40 PM	1/2/2025, 4:26:00 PM	20s	Manual trigger	✔ Succeeded

Table in Storage Container:

source

Container

Search

Upload

Add Directory

Refresh

Rename

Delete

Change tier

Acquire lease

Break lease

Give feedback

Overview

Diagnose and solve problems

Access Control (IAM)

Settings

Authentication method: Access key (Switch to Microsoft Entra user account)

Location: source

Search blobs by prefix (case-sensitive)

Name	Modified	Access tier	Archive status
DimCustomer.csv	12/19/2024, 4:00:12 PM	Hot (inferred)	

Table Copied in SQL Database:

Tables

dbo.dim_customer

CustomerKey (nvarchar, null)

GeographyKey (nvarchar, null)

CustomerAlternateKey (nvarchar, null)

Title (nvarchar, null)

FirstName (nvarchar, null)

MiddleName (nvarchar, null)

LastName (nvarchar, null)

NameStyle (nvarchar, null)

BirthDate (nvarchar, null)

MaritalStatus (nvarchar, null)

Suffix (nvarchar, null)

Gender (nvarchar, null)

EmailAddress (nvarchar, null)

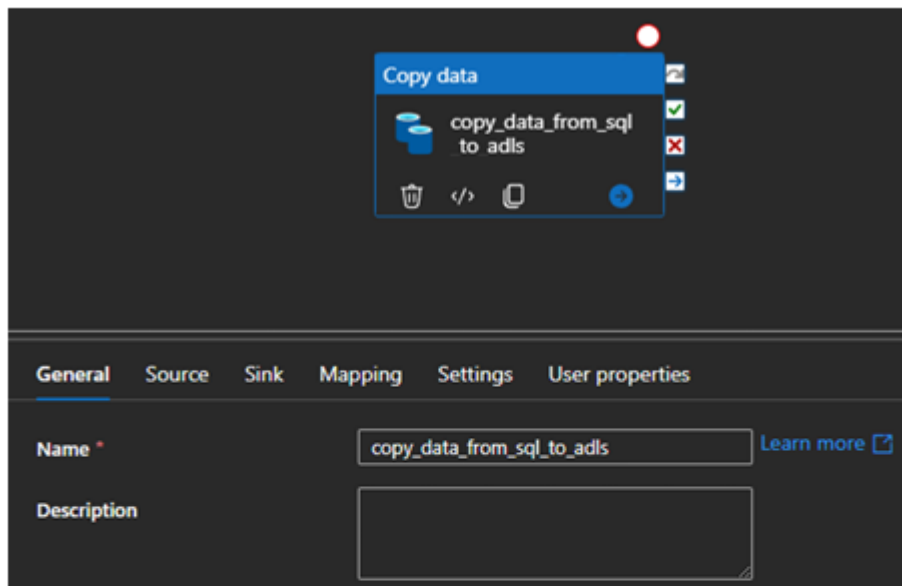
YearlyIncome (nvarchar, null)

TotalChildren (nvarchar, null)

NumberChildrenAtHome (nvarchar, null)

5.2 Question No. 5.2.

Pipeline Name:



Copy data

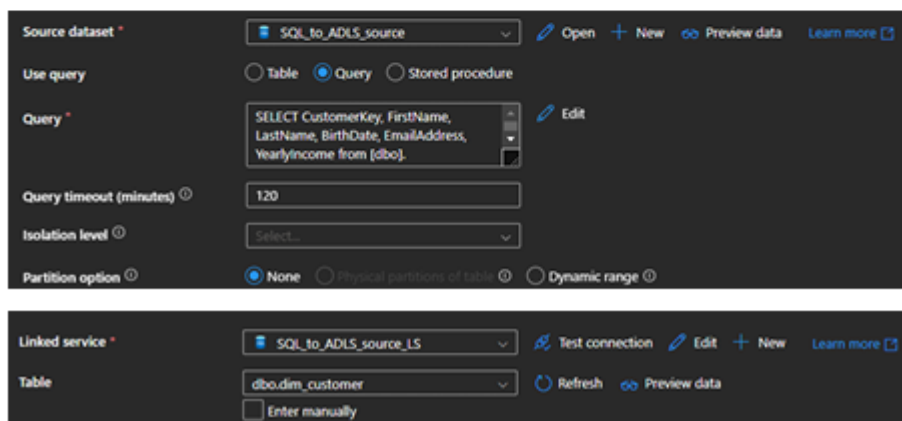
copy_data_from_sql_to_adls

General Source Sink Mapping Settings User properties

Name * copy_data_from_sql_to_adls [Learn more](#)

Description

Source Dataset:



Source dataset * SQL_to_ADLS_source [Open](#) [New](#) [Preview data](#) [Learn more](#)

Use query ☐ Table ☒ Query ☐ Stored procedure

Query * `SELECT CustomerKey, FirstName, LastName, BirthDate, EmailAddress, YearlyIncome from [dbo].` [Edit](#)

Query timeout (minutes) ⓘ 120

Isolation level ⓘ Select...

Partition option ⓘ ☒ None ☐ Physical partitions of table ⓘ ☐ Dynamic range ⓘ

Linked service * SQL_to_ADLS_source_LS [Test connection](#) [Edit](#) [New](#) [Learn more](#)

Table `dbo.dim_customer` [Refresh](#) [Preview data](#)

☐ Enter manually

SQL Query for incremental Load:

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

```
FROM [dbo].[dim_customer]
```

```
WHERE [YearlyIncome] > 60000
```

Sink Dataset

Sink dataset * SQL_to_ADLS_target Open

Copy behavior ^① Select...

Max concurrent connections ^①

Block size (MB) ^①

Linked service * SQL_to_ADLS_target_LS Test connection Edit New Learn more

File path target / Directory / DimCustomer.csv Browse

Compression type No compression

Column delimiter ^① Comma (,)

Row delimiter ^① Default (\r\n, or \r\n)

Encoding ^① Default(UTF-8)

Quote character ^① Double quote (")

Escape character ^① Backslash (\)

First row as header ^① ☒

Null value ^①

Pipeline Runs:

<input type="checkbox"/>	pipeline_SQL_to_ADLS	1/2/2025, 4:46:51 PM	1/2/2025, 4:50:07 PM	18s	Manual trigger	✓ Succeeded	Original
<input type="checkbox"/>	pipeline_SQL_to_ADLS	1/2/2025, 4:43:57 PM	1/2/2025, 4:44:11 PM	14s	Manual trigger	✓ Succeeded	Original

Table in SQL Database:

Query 1

```

1 SELECT CustomerKey, FirstName, LastName, BirthDate, EmailAddress, YearlyIncome FROM [dw].[dim_customer]
2 WHERE [YearlyIncome] < 50000

```

Results

CustomerKey	FirstName	LastName	BirthDate	EmailAddress	YearlyIncome
11000	Jon	King	10/6/1971	jonking@adventure-works.com	90000
11003	Christy	Zhou	8/14/1973	christy13@adventure-works.com	70000
11004	Elizabeth	Johnson	8/5/1975	elizabeth@adventure-works.com	80000
11005	John	Roos	8/1/1976	john1@adventure-works.com	70000
11006	Janet	Alonso	10/3/1976	janet@adventure-works.com	70000

Table in Target Storage Container

Upload Add Directory Refresh Rename Delete Change tier Acquire lease Break lease Give feedback

Authentication method: Access key (Switch to Microsoft Entra user account)

Location: target

Search blobs by prefix (case-sensitive)

Name	Modified	Access Tier
<input type="checkbox"/> DimCustomer.csv	1/2/2025, 4:50:07 PM	Hot (inferred)

6. Question No. 6

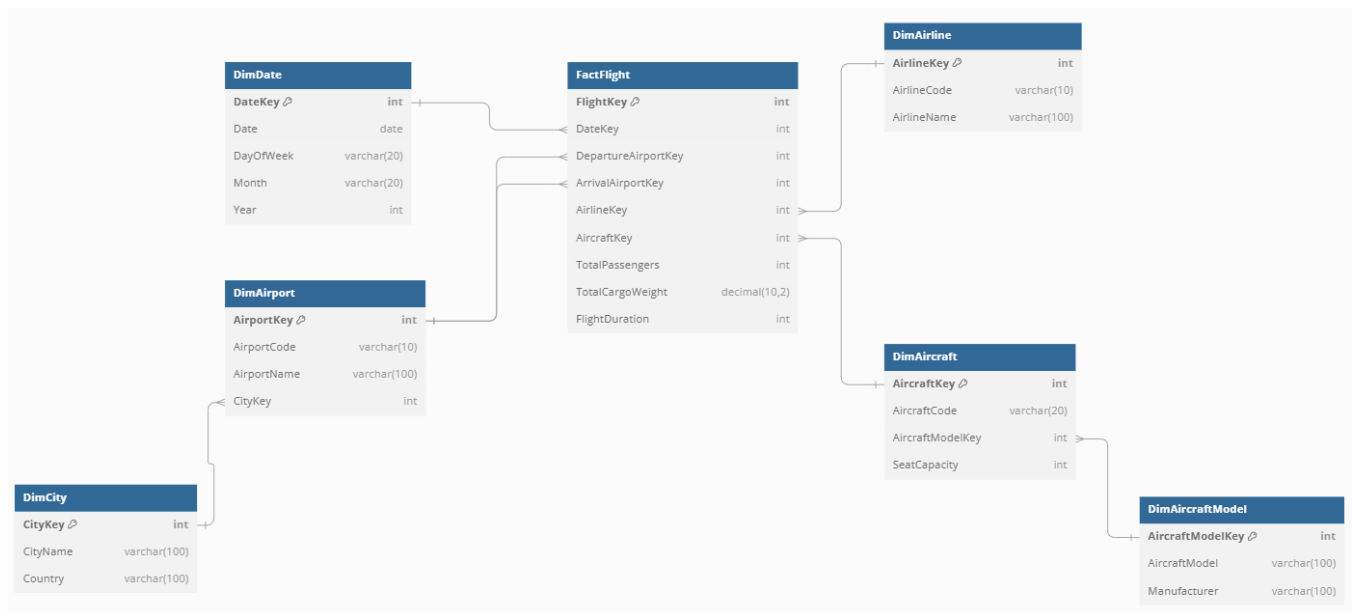


Diagram can be opened here: https://dbdiagram.io/d/Data_etl_pipelines_project-671917c897a66db9a305f05b

7. 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).

8. Question No. 8

	MaritalStatus	SalesTerritoryCountry	total_sales_amount	total_orders
1	M	Australia	4219833	7056
2	S	Australia	4841168	6289
3	M	Canada	1078215	4611
4	S	Canada	899630	3009
5	M	France	1406578	2712
6	S	France	1237440	2846
7	M	Germany	1588371	2902
8	S	Germany	1305941	2723
9	M	United Kingdom	2028760	3869
10	S	United Kingdom	1362952	3037
11	M	United States	4865619	12123
12	S	United States	4524171	9221

Figure 15: Result Set Q8

SQL Query: See file *query_q8.sql*

9. 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. Spark¹

¹ Own figure based on Amazon Web Services (2024)

10. 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

2. 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

3. 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

11. 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)

12. 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)

Example not in 1NF:

CustomerID	Name	PhoneNumbers
1	Alice	123-456, 789-012
2	Bob	345-678

Example in 1NF:

CustomerID	Name	PhoneNumber
1	Alice	123-456
1	Alice	789-012
2	Bob	345-678

Figure 17: Example 1NF²

² Own figure

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)

Example in 1NF but not in 2NF

OrderID	ProductID	ProductName	Quantity
101	1	Pen	10
102	2	Notebook	5

Normalized to 2NF:

- Orders Table:

OrderID	ProductID	Quantity
101	1	10
102	2	5

- Products Table:

ProductID	ProductName
1	Pen
2	Notebook

Figure 18: Example 2NF³

³ Own figure

Third Normal Form - 3NF:

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

Example in 2NF but not in 3NF:

EmployeeID	DepartmentID	DepartmentName
1	10	HR
2	20	IT

➔ DepartmentName depends on DepartmentID, not directly on EmployeeID

Normalized to 3NF:

- **Employee Table:**

EmployeeID	DepartmentID
1	10
2	20

- **Department Table:**

DepartmentID	DepartmentName
10	HR
20	IT

Figure 19: Example 3NF⁴

⁴ Own figure

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:

EmployeeID	Name	DepartmentName
1	Alice	HR
2	Bob	IT
3	-	Marketing

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

DepartmentID	DepartmentName
10	HR
20	IT
30	Marketing

Figure 20: Example and fix of Insertion Anomaly⁵

⁵ Own figure based on Javatpoint, 2025

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

EmployeeID	Name	DepartmentName
1	Alice	HR → Human Resources
2	Bob	HR → Human Resources

With a normalized Schema, the Department Table can simply be updated:

DepartmentID	DepartmentName
10	HR → Human Resources
20	IT

Figure 21: Example and fix of Update Anomaly 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:

OrderID	ProductID	ProductName	Quantity
101	1	Pen	10
102	2	Notebook	5

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

ProductID	ProductName
1	Pen

Figure 22: Example and fix of Deletion Anomaly ⁷

⁶ Own figure based on Javatpoint, 2025

⁷ Own figure based on Javatpoint, 2025

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/>