SCHOOL OF ARCHITECTURE, COMPUTING & ENGINEERING

Submission instructions

- Cover sheet to be attached to the front of the assignment when submitted
- Question paper to be attached to assignment when submitted
- All pages to be numbered sequentially
- All work has to be presented in a ready to submit state upon arrival at the ACE Helpdesk. Assignment cover sheets or stationery will **NOT** be provided by Helpdesk staff

Module code	CN7031		
Module title	Big Data Analytics		
Module leader	Amin Karami		
Assignment tutor	A Karami, F Jafari and the lab tutors		
Assignment title	Big Data Analytics: Coursework		
Assignment number	1		
Weighting	100%		
Handout date	Week 6 (5th November 2025)		
Submission date	Presentation: Week 12 (15th-19th Dec 2025) Turnitin Submission: 15th December 2025, 22:00		
Learning outcomes assessed by this assignment	1-8		
Turnitin submission requirement	Yes	**Turnitin GradeMark feedback used?**	No
UEL Plus Grade Book submission used?	No	**UEL Plus Grade Book feedback used?**	No
Other electronic system used?	Yes	**Are submissions / feedback totally electronic?**	Yes
Additional information			

Form of assessment:

☐ Individual work ☒ Group work

For **group work** assessment which requires members to submit both individual and group work aspects for the assignment, the work should be submitted as:

☒ Consolidated single document ☐ Separately by each member

Number of assignment copies required:

☒ 1 ☐ 2 ☐ Other

Assignment to be presented in the following format:

☒ On-line submission
☐ Stapled once in the top left-hand corner
☐ Glue bound
☐ Spiral bound
☐ Placed in a A4 ring bound folder (not lever arch)

Note: To students submitting work on A3/A2 boards, work has to be contained in suitable protective case to ensure any damage to work is avoided.

Soft copy:

☐ CD (to be attached to the work in an envelope or purpose made wallet adhered to the rear)
☐ USB (to be attached to the work in an envelope or purpose made wallet adhered to the rear)
☒ Soft copy not required

CN7031 - Big Data Analytics
Group assignment 2025-26 Academic Year

The CRWK must be completed in **groups of 3 or 4 students**. It consists of two sections: (1) Big Data analytics on a real case study and (2) a group presentation. All group members must participate in the presentation, which will be conducted online via Microsoft Teams. Failure to attend the presentation with a video call will result in module failure.

- **In the main sit**, the overall mark is determined by two activities:

1. Big Data Analytics report (approximately 3,000 words, with a tolerance of ± 10%) in the HTML format (60%)
2. Presentation (40%)

- In the re-sit, the overall mark is based solely on the report:

1. Big Data Analytics report (approximately 3,000 words, with a tolerance of ± 10%) in the HTML format (100%). Failed students must propose new solutions and avoid duplicating their original contribution from the main sit.

Good Luck with your coursework!

Big Data Analytics using PySpark

CN7031 – Big Data Analytics (60%)

This project requires students to complete a full Big Data ETL (Extract, Transform, Load) and Analysis cycle using PySpark (RDD or DataFrame API). **The collaboration focuses on knowledge sharing and peer learning, but the marking is strictly individual.**

(1) Data Requirements

- **Source**: Data must be sourced from **HuggingFace Datasets**. If your data comes from other sources, you will be failed this assessment, and we will not mark your work. Then, you will need to attend resit.
- **Size Constraint**: The dataset (or the relevant subset used) must be **at least 200MB** to necessitate the use of parallel processing techniques and expose performance issues like Data Skew and Shuffling.
- **Type:** Structured (e.g., large CSV/Parquet), Unstructured (e.g., text corpus), or Semi-structured (e.g., large nested JSON/XML).

(2) Individual Contribution

To ensure individual contribution and minimize reliance on external generative tools for direct answers, the following rules are non-negotiable:

- **Non-Similar Code/Analysis**: Each student must select 4 non-similar tasks from the pool of 20 below.
- **Dissimilar Contribution:** Even if two students in a group use the same dataset, their chosen tasks must be demonstrably non-similar in coding methodology, theoretical application, and final analysis/insight.
    - Example: If Student A chooses "Broadcast Join Optimization," their implementation and analysis must be distinct from Student B, even if Student B chooses "Shuffle Partitioning Strategy." Their RDD/DF pipelines, while operating on the same data, must pursue different analytical goals and apply different core transformations.
- **Similarity between groups**: If two or more groups come with similar datasets and similar chosen tasks, it is considered plagiarism, and everyone involved will fail.

(3) Pool of Advanced Tasks (Choose 4 Non-Similar Tasks)

Each student must choose 4 tasks from this pool to form the core of their technical contribution.

ID	Theme	Task Description	Focus Areas
1	**Nested/Complex RDD/DF**	**Hierarchical Data Parsing & Flattening:** Process a large semi-structured dataset (e.g., JSON or XML). Design and implement a PySpark pipeline (using either RDD operations or advanced DataFrame functions like explode, struct, or custom UDFs) to flatten the data into an analytic structure.	Advanced DF/RDD, UDFs, Schema Inference.
2	**Optimization / Joins**	**Broadcast Join Implementation & Justification:** Identify a suitable smaller lookup table within your data/analysis. Implement both a standard inner join and a **Broadcast Join**. Quantify the performance improvement using the Spark UI and explain the underlying mechanism of why broadcasting is effective in this scenario.	Optimization, Spark UI Analysis, Shuffling.
3	**Data Skew / Partitioning**	**Skew Mitigation Strategy (Salting):** Identify a feature that leads to Data Skew. Implement and test a strategy to mitigate this skew using **key salting**. If you can, show the job execution stages before and after mitigation using the Spark UI to prove the reduction in task time variation.	Data Skew, Shuffling Issues, Spark Performance.
4	**Unstructured Data / REGEX**	**Advanced Text Feature Extraction:** Apply complex **REGEX patterns** within a PySpark pipeline (via RDD map/filter or DF UDFs) to extract sophisticated features from an unstructured text column (e.g., extracting date formats, custom identifiers, or nested tags). Your code must showcase non-trivial regular expression usage.	REGEX, UDFs, Unstructured Data Handling.
5	**DF Optimization / Windowing**	**Window Function Optimization:** Use the DataFrame API to implement two non-trivial **Window Functions** (e.g., cumulative sum, moving average, or rank based on multiple partitions). Analyze the execution plan and discuss how Spark manages the data movement (shuffling) required for these complex aggregations.	Advanced DF, Spark Optimization, Shuffling.
6	**RDD Optimization / Caching**	**Persistence Strategy Comparison:** Compare the performance of three different RDD/DataFrame persistence levels (MEMORY_ONLY, DISK_ONLY, MEMORY_AND_DISK) for a heavy iterative task (e.g., calculating a large number of descriptive statistics). Use code to demonstrate the correct use of cache() or persist() and analyze the resultant DAG.	RDD/DF Caching, Memory Management, Optimization.
7	**Data Imbalance / RDD**	**Imbalanced Sampling with RDD:** If your primary analytic goal involves predicting a rare event or processing imbalanced data, use RDD transformations (e.g., sample or custom filtering logic based on keys) to implement a **sampling or oversampling strategy**. Justify your chosen approach statistically and show the before/after effect on the data distribution.	Imbalance Data, RDD Complexity, Statistical Sampling.
8	**Spark Configuration**	**Custom Spark Configuration Tuning:** Use the Google Colab environment to demonstrate tuning key Spark parameters (spark.executor.memory, spark.default.parallelism, or spark.sql.shuffle.partitions). Systematically test two different	Spark Optimization, Configuration Tuning.

		configuration settings and report the impact on overall job runtime and resource utilization.	
9	**LSEPI & Critical Thinking**	**Legal, Social, Ethical, and Professional Issues (LSEPI) Analysis:** Write a detailed critical analysis (500-1000 words, must be open-ended and highly original) on the LSEPI implications of using *your specific dataset* for Big Data Analysis (e.g., data bias, privacy concerns in the ETL stage, legal compliance for data provenance).	LSEPI, Critical Thinking, Subject Understanding.
10	**RDD vs. DF Comparison**	**Code Efficiency and Performance Comparison:** Implement the exact same data transformation logic using **both RDD API and DataFrame API**. Analyze the resulting DAG and execution plans, discussing the Catalyst Optimizer's role and the trade-offs between the two approaches in terms of code verbosity and performance.	RDD/DF Comparison, Optimization, DAG/Execution Plan.
11	**Descriptive Stats / Aggregation**	**Statistical Profile Generation with Aggregations:** Generate a comprehensive statistical profile (kurtosis, skewness, quantile calculations, mode for categorical features) using **optimized DF aggregate functions**. Analyze the cost of these distributed calculations.	Descriptive Stats, Aggregation, Performance.
12	**Optimization / Bucketing**	**Optimized Data Bucketing:** Implement **data bucketing** (saving to disk) based on a high-cardinality column. Demonstrate and quantify the query performance improvement (read speed) when reading and joining against the bucketed dataset versus the non-bucketed dataset for filtering/joining operations.	Bucketing, Read Optimization, File Layout.
13	**Optimization / Pandas UDF**	**Pandas UDF (Vectorized) Implementation:** Implement a complex calculation (e.g., iterative scoring or complex mapping) first as a standard Python UDF and then refactor it as a **Pandas UDF (Vectorized UDF)**. Quantify the performance difference using timestamps and explain the optimization achieved via Apache Arrow.	UDF Optimization, Vectorization, Apache Arrow.
14	**Partitioning Strategy**	**Range Partitioning for Search Efficiency (RDD/DF):** Apply a **Range Partitioner** based on a numerical or time-series key (either RDD partitionBy or DF options). Justify why range partitioning is superior to hash partitioning for ordered lookups or range queries on your specific dataset.	Partitioning Strategy, RDD/DF Control, Search Optimization.
15	**Advanced DF Syntax**	**Advanced Array/Map Manipulation for Feature Engineering:** Apply advanced DataFrame functions (array_except, map_concat, transform, aggregate) to generate a new, high-impact feature (e.g., a complex risk score or categorical index). The task requires at least two layers of nested DF function calls.	Advanced DF Syntax, Feature Engineering, Complex Output.
16	**MLlib/Feature Engineering**	**One-Hot Encoding and Vector Assembly**: Use Spark MLlib transformers (specifically StringIndexer, OneHotEncoderEstimator, and VectorAssembler) to process a mix of categorical and numerical features, preparing the data for a Machine Learning model.	MLlib, Pipelines, Feature Engineering, Data Preparation.
17	**Custom Aggregation**	**Implementing a Custom Aggregator (User Defined Aggregate Function - UDAF)**: Define and implement a simple, business-specific aggregation logic (e.g., a custom trimmed mean or weighted average) using a UDAF. Contrast the complexity and performance with an equivalent approach using built-in functions.	UDAF, Custom Logic, Advanced Aggregation.

18	**Data Quality / Checkpointing**	**Data Validation and Checkpointing**: Implement a data quality check pipeline that verifies schema compliance and checks for critical data constraints (e.g., non-null, value range). Integrate checkpointing for an iterative process (like a while loop reading from a stream or a recursive function on RDDs) and explain its role in fault-tolerance.	Data Quality, Checkpointing, Fault Tolerance.
19	**File Format Optimization**	**Parquet vs. ORC Columnar Format Comparison:** Load a large dataset and write it out using both Parquet and ORC formats with Snappy compression. Compare the on-disk size, and then quantify the read-performance difference for a query that only selects a few columns (demonstrating the effect of columnar storage and predicate pushdown).	File Formats, Columnar Storage, I/O Optimization.
20	**Shuffling Optimization**	**Tuning** spark.sql.shuffle.partitions: Select a wide transformation operation (like a large groupBy or a standard join). Experimentally test three distinct values for the spark.sql.shuffle.partitions configuration (e.g., 5, 50, and 200). Use the Spark UI to compare the resulting number of shuffle files, the execution time, and the Task Time variance for the final stage. Justify which setting is optimal for your cluster and dataset.	Shuffling, Configuration, Spark UI Analysis, Performance Tuning.

(4) Marking Scheme: Individual CRWK Report (100 Marks)

This rubric assesses the depth of technical skill, critical thinking, and the clarity of individual contribution based on the four chosen tasks.

Type	Criteria & Focus	Marks
Data Acquisition & Cleaning	- Successful loading of the **200MB+** HuggingFace dataset. Demonstrably robust data cleaning/pre-processing pipeline. Effective use of PySpark to handle the data format (JSON/XML/Text). Evidence of handling missing values and data type conversion.	10
Implementation of Technical Task 1	Code implementation of the 1st technical task is correct, efficient, and addresses the technical challenge, using meaningful visualization and/or numerical outputs. **Non-similarity** to group members' code is clearly evident.	20
Implementation of Technical Task 2	Code implementation of the 2nd technical task is correct, efficient, and addresses the technical challenge, using meaningful visualization and/or numerical outputs. **Non-similarity** to group members' code is clearly evident.	20
Implementation of Technical Task 3	Code implementation of the 3rd technical task is correct, efficient, and addresses the technical challenge, using meaningful visualization and/or numerical outputs. **Non-similarity** to group members' code is clearly evident.	20
Implementation of Technical Task 4	Code implementation of the 4th technical task is correct, efficient, and addresses the technical challenge, using meaningful visualization and/or numerical outputs. **Non-similarity** to group members' code is clearly evident.	20
Academic Style & Professionalism	Use the CN7031 CRWK template. Individual contributions are clearly well-structured, professionally formatted, and adheres to academic conventions. Clear, error-free language and technical terminology.	10
TOTAL INDIVIDUAL REPORT MARKS		**100**

THE FORMAT OF FINAL SUBMISSION

1- We ONLY accept the HTML format for report submission. Use the ipynb template in the Moodle site to complete this CRWK for assessment.

2- Upload **ONLY one single HTML file per group** into Turnitin in Moodle. One member of each group must submit the work, **NOT** all members. The name of the file must be in the format of "`Your_Group_ID_CN7031`", such as ***Group200_CN7031.html*** if you are belonging to the group 200.

3- The submission link will be available from week 8, and you are free to amend your submitted file several times before submission deadline. Your last submission will be saved in the Moodle database for marking.

PLAGIARISM

In the case of **copied codes** from external sources or other individuals or **high similarity between groups**, it will result in a zero mark for the affected students. Moreover, a plagiarism flag will be placed on their transcript, and they will be required to attend a committee focused on addressing breaches of regulations. This committee will seek further explanations from the students and may impose additional penalties as deemed necessary.

FEEDBACK TO STUDENTS

Feedback is central to learning and is provided to students to develop their knowledge, understanding, skills and to help promote learning and facilitate improvement.

- Feedback will be provided as soon as possible after the student has completed the assessment task.
- Feedback will be in relation to the learning outcomes and assessment criteria.
- It will be offered via Turnitin GradeMark or Moodle post.

As the feedback (including marks) is provided before Award & Field Board, marks are:

- Provisional
- available for External Examiner scrutiny
- subject to change and approval by the Assessment Board

ASSESSMENT FORM FOR PRESENTATION (main sit)

CN7031 – Big Data Analytics (40%)

Students are required to accurately complete this section.

Group No: ....

1st Student (full name and ID):

2nd Student (full name and ID):

3rd Student (full name and ID):

4th Student (full name and ID):

Assessment Criteria:

Criteria	1st	2nd	3rd	4th	Mark
Clarity of the presentation and demonstration of group-based and individual contributions: - Clearly explains the individual contribution - Shows understanding of the project and dataset - Explains Spark configurations					**20**
Understand Spark ecosystem - Explain RDD/DF operators, transformations and actions					**30**
Answers technical questions about Spark ecosystem from the chosen tasks (e.g., optimization, DAG, SQL, UDFs, descriptive stats)					**30**
Verbal delivery and engagement during the presentation					**20**
Overall mark					**100**

Date & Time: ………………………….

Assessors’ signature and comments: