

SENG 550: Scalable Data Analytics

Group Assignment 3 (Research Part) - Project & Group Presentation: Scalable Data Engineering and Analytics

Rafee Al Ahsan

Instructor & Ph.D. Researcher

Department of Electrical and Software Engineering

University of Calgary (UCalgary)

Date: 1st November, 2025

Project Submission Due Date: 1st December, 2025

Total Marks: 50

Project Overview (40 Marks)

This **Group Assignment 3 (Research Part) – Project** challenges you to design, implement, and analyze an end-to-end **data analytics and engineering pipeline** aligned with the learning outcomes of **SENG 550: Scalable Data Analytics**. Your project must integrate the core concepts of the course: *sources and characteristics of large-scale data, data modeling and storage design, scalable query and analytics workflows, and applications across real-world domains* (e.g., business intelligence, social media, IoT, or text analytics).

Projects should combine **technical rigor, architectural understanding, and practical coding demonstrations**. Your work must integrate key elements such as:

- Big-data characteristics (**5Vs**: Volume, Velocity, Variety, Veracity, and Value);
- End-to-end **data engineering** using Python, PostgreSQL, Spark, Hadoop and/or MongoDB for ingestion, transformation, and storage;
- Implementation of scalable analytics through **ETL/ELT workflows**, SQL joins, NoSQL aggregations, and machine learning or text-processing modules where relevant;
- **Evaluation of performance, optimization, and scalability**, supported by metrics such as query latency, throughput, and storage efficiency;
- Clear visualization of analytical outcomes and interpretation of results through professional dashboards or reports.

(See Minimum Required Standards for more information).

Use of AI Tools: Generative AI (e.g., ChatGPT, Gemini) may assist with brainstorming, code refactoring, and documentation. However, you **MUST**:

- Clearly **cite any portion of the report influenced or supported by an AI tool** (code/comments/text).
- In your individual **Contribution Statement**, explain how the AI tool was used, for what purpose, and how it helped you reach your goals.

- Note: Unacknowledged AI assistance violates academic integrity and will incur severe penalties.

Learning Outcomes

By completing this project, you will be able to:

- Identify and model diverse large-scale data sources using Python-based ingestion and transformation tools.
- Design efficient database architectures and schemas in **PostgreSQL** and **MongoDB** for structured and semi-structured data.
- Implement complete **ETL/ELT pipelines** that integrate Python scripts, SQL queries, and NoSQL aggregations for scalable data analytics.
- Apply analytical techniques such as joins, aggregations, text mining, and predictive modeling using Python and database-native operations.
- Evaluate query optimization, indexing, and storage design strategies to improve system performance and scalability.
- Compare results with distributed frameworks (e.g., Spark, Kefka, Hadoop) to analyze scalability trade-offs across different architectures.
- Develop dashboards and visual analytics in Python (Dash, Matplotlib, Plotly, or Streamlit) to communicate insights clearly and effectively.
- Demonstrate teamwork, documentation, and reproducibility in the design, execution, and presentation of large-scale data analysis projects.

Approved Project Topic Titles

Each group must select **one** topic from the list below and confirm by **November 4th, 2025**. Topics are assigned on a **first-come, first-served basis**. Each project must:

- Implement a reproducible and/or distributed data analytics systems using **Python, PostgreSQL, Spark, Hadoop, Kefka and/or MongoDB**.
- Demonstrate practical data engineering (ETL/ELT), analytical querying, and visualization.

If no selection is received by the deadline, a topic will be automatically and randomly assigned.

1. Customer Analytics Dashboard using Python and PostgreSQL

- Design an end-to-end analytics system for customer purchase data.
- Build ETL pipelines in Python and store relational data in PostgreSQL.
- Implement SQL-based insights and create dashboards using Plotly, Dash or any appropriate software.

2. Text Classification System with Python and MongoDB

- Collect, clean, and store unstructured text data in MongoDB.
- Implement text preprocessing, TF-IDF, and classification using scikit-learn.
- Compare MongoDB queries vs. Python in-memory dataframes for performance.

3. E-Commerce Data Warehouse using PostgreSQL

- Design star/snowflake schemas for sales, customers, and transactions.
- Perform multi-dimensional OLAP queries and aggregations.
- Analyze query optimization and indexing for analytical workloads.

4. NoSQL–SQL Hybrid Analytics Platform

- Integrate MongoDB (semi-structured) and PostgreSQL (structured) data.
- Implement synchronization via Python connectors.
- Compare aggregation pipelines vs. SQL joins for analytics.

5. Real-Time Stock Market Analytics using Python APIs, Spark and PostgreSQL

- Stream financial data via APIs and load into PostgreSQL tables.
- Use Python for moving averages, volatility, and correlation analysis.
- Deploy a lightweight Spark job for historical batch comparison.

6. Social Media Sentiment Analysis and Visualization

- Collect Twitter or Reddit data with Python's Tweepy/PRAW libraries.
- Store results in MongoDB and analyze sentiment trends over time.
- Build dashboards or word clouds using Matplotlib, Dash or any appropriate software.

7. Healthcare Data Analytics using PostgreSQL and Python

- Create an anonymized healthcare dataset (patients, diagnoses, prescriptions).
- Use SQL joins and Python visualizations for trends and outcome prediction.
- Discuss scalability and compliance (HIPAA/PIPEDA) considerations.

8. Data Cleaning and Quality Validation Framework

- Build a Python-based ETL pipeline that detects and corrects data quality issues.
- Store clean data in PostgreSQL or MongoDB.
- Implement automated validation reports and logging.

9. Movie Recommendation System using Python and SQL

- Build a relational database of users, ratings, and movies.
- Implement collaborative filtering or cosine similarity in Python.
- Compare SQL-based aggregation vs. in-memory computations.

10. IoT Sensor Data Management using MongoDB and Spark

- Simulate sensor streams and store time-series data in MongoDB.
- Use Python to analyze device reliability and detect anomalies.
- Parallelize large data analysis using Spark.

11. Retail Sales Forecasting using Python and PostgreSQL

- Import retail data and apply regression or ARIMA forecasting models.
- Use SQL queries for feature extraction and join operations.
- Visualize forecasts and evaluate prediction accuracy.

12. Fraud Detection on Transactional Databases

- Design a relational model for transactional logs in PostgreSQL.
- Use Python ML algorithms to classify anomalies or fraudulent behavior.
- Measure model latency and scalability for growing datasets.

13. Document-Based Knowledge Retrieval using MongoDB

- Store technical documents and reports in MongoDB collections.
- Implement text search and ranking using n -gram or embedding similarity.
- Compare retrieval speed vs. SQL full-text search.

14. Real-Time Event Tracking with Python, PostgreSQL and Spark/Kefka

- Build an event ingestion pipeline (e.g., website or sensor logs).
- Store events in PostgreSQL and aggregate for analytics.
- Integrate Kafka or Spark Streaming for real-time alerts.

15. ETL Workflow Automation using Airflow and Python

- Design data ingestion DAGs for multiple data sources.
- Load processed data into PostgreSQL and MongoDB targets.
- Compare Airflow orchestration vs. manual Python scripting.

16. Student Performance Analytics using Python and SQL

- Create an academic dataset (students, grades, activities).
- Query learning patterns and correlations using SQL joins and Python plots.
- Evaluate normalization, indexing, and visualization best practices.

17. Business Intelligence Dashboard for Sales Insights

- Build an analytical data mart in PostgreSQL.
- Use Python libraries (Dash, Streamlit or Matplotlib) for interactive dashboards.
- Benchmark performance on Spark and SQL for large datasets.

18. Geospatial Data Analytics with PostgreSQL/PostGIS

- Store location-based data and perform spatial joins/queries.
- Use Python for map visualizations and proximity analysis.
- Discuss indexing and optimization of spatial data.

19. Text-Based Question Answering System using Python

- Build a small-scale QA system using NLP and MongoDB for storage.
- Implement retrieval-based or transformer-based approaches.
- Measure latency, accuracy, and scalability trade-offs.

20. Benchmarking PostgreSQL vs. MongoDB for Data Analytics

- Load identical analytical datasets into both databases.
- Compare query performance, storage efficiency, and scalability.
- Provide recommendations for hybrid data architectures.

Final Project Report Requirements (IEEE Format)

Submit a professional technical report in the IEEE conference paper format attached in D2L. Your report must include the following and **must not be below 5 pages (excluding the Appendix Section)**:

1. **Title** – Select one of the approved project titles. **Your entire report should focus on the chosen Python, PostgreSQL, Spark, Hadoop and/or MongoDB-based analytics topics, where mentioned.**
2. **Abstract** – Concisely summarize your data sources, database architecture, data-processing workflow, analytical methods, key results, and real-world relevance.
3. **Introduction** – Introduce the motivation, context, and problem statement. Explain the business or domain value of scalable data analytics in your selected area.
4. **Background and Theory** – Present relevant concepts from large-scale data management, relational and NoSQL systems, schema design, and query optimization. Briefly include theoretical foundations of ETL/ELT, indexing, and data modeling principles.
5. **Methods and Pipeline** – Describe your implemented architecture in detail:
 - Data ingestion and preprocessing in **Python**.
 - Data storage and transformation in **PostgreSQL** and/or **MongoDB**.
 - Analytical methods (e.g., SQL joins, aggregation pipelines, text or predictive analytics).
 - Comparison with distributed frameworks such as **Spark** or **Hadoop** for scalability analysis where mentioned.
6. **Results and Discussion** – Present quantitative results such as query runtime, storage efficiency, accuracy, or system performance. Discuss scalability, fault-tolerance, and trade-offs between SQL and NoSQL designs.
7. **Visual Elements** – Include ER diagrams, data flow and pipeline charts, query execution plans, and plots of runtime, latency, or cost comparisons. Visualize analytical outcomes through dashboards or charts.
8. **Conclusions and Future Work** – Summarize findings, practical implications, and lessons learned. Suggest future extensions such as larger-scale deployment, cloud migration, or integration with streaming pipelines.
9. **References** – Cite at least five scholarly or technical references (academic papers, documentation, or books) in IEEE citation format.
10. **Appendix** – Provide detailed **Group Contribution Statements** for each member, clearly disclosing any use of AI tools and describing their purpose and contribution.

Expectations & Individual Contribution Statement

- This is a **group project**. All design, implementation, analysis, and writing are collaborative; however, each member must submit an **Individual Contribution Statement** in the Appendix Section.
- The report must demonstrate **collective mastery** of scalable data engineering/analytics and clear division of responsibilities.
- Each member's **Individual Contribution Statement** must describe:
 - The pipeline components or algorithms they implemented (ingestion, ETL, modeling, evaluation, ops).
 - How they validated results (tests, benchmarks, ablations).
 - Challenges faced and lessons learned.
 - Whether/how AI tools were used, including purpose and contribution.
- Conceptual discussion among groups is permitted, but **code, experiments, and writing must be your own group's work**.

Minimum Required Standards for this Project Report

- Integration of **at least four to five SENG 550 course topics** across your implemented analytics pipeline (e.g., data source characterization, Python-based ETL/ELT design, schema modeling, indexing, query optimization, aggregation, data visualization, NoSQL–SQL integration, text analytics, predictive modeling, data governance, scalability evaluation, cost/performance benchmarking, and reproducibility practices).
- The final report (excluding the appendix) must be a minimum of **5 full pages** and follow the IEEE paper format provided on D2L.
- Projects must be original — **no duplication of code or examples from tutorials, assignments, or other online repositories. Any violation of this would result in an immediate zero (0) for the whole Assignment 3 Research Part.**
- The report must be clearly organized, technically rigorous, and professionally formatted in IEEE style, with all figures, tables, and references properly labeled and cited.
- Include an **experimental evaluation** demonstrating analytical performance — e.g., query execution time, indexing speedup, or storage efficiency. Comparisons with Spark or Hadoop for scalability are required where mentioned in the approved topic.

- Provide at least **five (5) scholarly or technical references**, including database documentation, research papers, or technical reports, cited in IEEE format. Make sure to also include references of those sites/platforms/papers that helped you develop certain codes that you used to implement your Data Analytics scheme.
- Clearly acknowledge any assistance or code generation from AI tools (e.g., ChatGPT, Gemini, Copilot) within your **Group Contribution Statements**.

Presentation Overview (10 marks)

Each **group** must present their project and submit both the presentation materials and report by the deadline. This component assesses your ability to communicate an end-to-end scalable analytics solution with technical clarity and professional polish.

Presentation Requirements

- Each group must prepare and deliver an **in-person 15-minute presentation** during the scheduled presentation period between **December 2nd and December 5th, 2025**. The exact date, time, and order of group presentations will be announced closer to the presentation week via D2L.
- All group members are expected to participate and present in their allotted time or they will end up getting a zero for the presentation. Each presenter should clearly explain their individual contributions, technical responsibilities, and findings.
- Presentations must cover the data domain, system architecture (Python/ PostgreSQL/ MongoDB/ Spark/ Hadoop pipeline), implemented analytics or algorithms, evaluation results, and key insights.
- Slides must be professionally prepared with clear structure, consistent formatting, and legible visuals (e.g., ER diagrams, query plans, performance graphs, and dashboards).
- Each group will be evaluated on:
 - Clarity and depth of technical explanation.
 - Demonstration of analytical rigor and understanding of scalable data processing.
 - Visual organization, teamwork, and presentation delivery.
 - Ability to answer follow-up questions confidently and accurately.
- **Presentations exceeding 15 minutes will incur a time penalty.** Groups are encouraged to rehearse beforehand to ensure smooth delivery within the allotted time.

Required Presentation Structure (15-Minute Duration + 3–5 Minute Q/A)

1. **Opening (1 min)** Introduce your team, project title, and briefly state the motivation behind your topic. Highlight the real-world problem, its relevance, and the type of data you are analyzing.
2. **Introduction & Problem Definition (2 min)** Describe the data domain and clearly define the analytical or engineering problem being addressed. Summarize the dataset characteristics, its source, and the specific challenges (e.g., volume, variety, or data quality).
3. **Objectives & Scope (2 min)** Present your main objectives, key performance metrics, and any assumptions or constraints. Mention any ethical, governance, or privacy considerations relevant to your dataset.
4. **System Design / Methodology (5 min)** Explain your end-to-end pipeline using **Python, PostgreSQL, Spark, Hadoop and/or MongoDB**. Describe the stages of data ingestion, storage, transformation, and analytics (e.g., queries, aggregations, text or predictive modeling). Include visual elements such as ER diagrams, schema designs, and data flowcharts. If applicable, briefly mention optional scalability experiments using Spark or Hadoop.
5. **Results & Discussion (3–4 min)** Present your key findings with supporting figures, dashboards, or query results. Interpret outcomes such as query performance, model accuracy, or visualization insights. Discuss trade-offs, optimization techniques, and lessons learned.
6. **Conclusion & Future Work (1–2 min)** Summarize the main takeaways, highlight technical achievements, and propose possible extensions (e.g., deployment on the cloud, real-time analytics, or integration with other data systems).
7. **Q/A Session (3–5 min)** Address questions from the instructor and audience. Each group member should be ready to answer questions related to their individual contributions and implementation responsibilities.

Technical and Professional Expectations

- Each group will have a total timeslot of approximately **18–20 minutes**, consisting of a **15-minute presentation** followed by a **3–5-minute Q/A session**.
- All group members must be present and ready to answer questions regarding their individual and collective contributions.
- Ensure your slides are finalized, consistent, and match the version used during the in-person presentation.

- Verify that all figures, diagrams, and visual elements are clear, labeled, and properly cited.
- Maintain a professional and confident speaking pace — avoid reading slides verbatim.
- Use precise technical and analytical terminology when describing your methods, results, and database implementation.
- Coordinate transitions between speakers smoothly to ensure balanced time allocation across team members.
- Respect the time limit: **15 minutes for presentation** and **3–5 minutes for Q/A**. Exceeding the allotted time will result in a deduction of marks.

Submission Deadline for both Project & Presentation

- **Deadline: 1 December 2025 (Monday) by 11:59 PM via D2L.**
- **Required Submissions:**
 - **Project Report in IEEE Format including Group Contribution Statements in the Appendix (PDF)** – Upload to the “*Assignment 3 - Research Part*” Dropbox on D2L.
 - **Presentation Slides (PowerPoint)** – Upload to the “*Assignment 3 - Research Part*” Dropbox on D2L.
- Late submissions will not be accepted without prior approval and will also result in a penalty.
- All files based on whether they are either an IEEE project report or presentation slides **MUST** be properly named using the format shown below:
Group_Number_SENG550_(Project/Presentation)_ResearchPart_ProjectTitle.

—— The END —— Good Luck! ——