IMT Atlantique
Technopôle de Brest-Iroise - CS 83818
29238 Brest Cedex 3
Téléphone : +33 (0)2 29 00 13 04
Télécopie : +33 (0)2 29 00 10 12
URL : **www.imt-atlantique.fr**

IMT Atlantique & CleverCloud Project : Performance evaluation and testing of Openhalo, a MySQL-to-PostgreSQL merger to simplify the world of databases

Statement of Work

Nélia Fedele, Co-project Leader
Sei Bayle, Co-project Leader
François-Xavier Collot, Production Manager
Juliette Faurie, Deliverables Manager
Hugo Bentata, Team Cohesion Officer
Mohamed Montadhar Ettaieb, Collaborative tools and software Manager
Houda Daouairi, Tutor and client communication manager

Gregory Smits, School Tutor
Clever Cloud, Client
Aurélien Hébert, Client's Tutor
Tanguy Mazé, Client's Tutor

Date d'édition : 8 octobre 2025
Version : 1.0

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Sommaire

1. Introduction

Nowadays, relational SQL databases revolve around different technologies such as : MySQL, PostgreSQL, MongoDB, Redis and Elasticsearch. Having 5 types only adds work for Clever Cloud employees. Our project can be divided in three main points and one optional point. First, we have to provide a user guide for all (documentation and test protocols). Then, they ask for a compliance test tool capable of verifying the exact compatibility between MySQL and PostgreSQL. And also a performance benchmarks that is able to evaluate the performance of openhalo versus MySQL. And if we have enough time we can try to configurate the migration, write a code in order to convert the settings to openhalo format.

2. Context and project presentation

2.1. Presentation of Clever Cloud

Clever Cloud is a French company founded in 2010 that offers a *Platform as a Service* (PaaS). It provides a reliable, scalable, and secure infrastructure, allowing teams to focus on application development rather than server management. Deployment is done through a simple *git push-to-deploy*, with pricing proportional to usage (*pay-as-you-scale*). The platform offers a catalog of managed databases, including PostgreSQL, MySQL, MongoDB, Redis, and Elasticsearch.

TABLE 1 – Clever Cloud — Main Features

Dimension	**Description**
Foundation	French company founded in 2010
Positioning	European PaaS for application deployment and operation
Deployment method	*Git push-to-deploy*, with no server management on the user side
Pricing	*Pay-as-you-scale*, costs aligned with actual usage
Focus	Emphasis on development (*coding over infrastructure*)
Databases	Multi-engine DBaaS offering : PostgreSQL, MySQL, MongoDB, Redis, Elasticsearch
Operations	Fleet of approximately 7,000 servers, 100 emergency interventions per month, dedicated team of 6 people

2.2. Project context

Clever Cloud aims to explore the possibility of simplifying and standardizing the management of its databases by limiting the diversity of environments used. Currently, many applications rely on MySQL, which complicates their migration to PostgreSQL due to differences in protocol and SQL dialect.

In this context, the company relies on openHalo, an open-source software derived from PostgreSQL, capable of interpreting the MySQL network protocol and SQL syntax. This compatibility makes it possible to migrate MySQL applications to PostgreSQL without modifying the application code (zero code change), while benefiting from PostgreSQL's performance, reliability, and ecosystem.

The project assigned to the student team involves studying and tooling this approach to determine whether it can provide a viable solution for reducing operational complexity and promoting code standardization within Clever Cloud environments.

2.3. Challenges for Clever Cloud

The experimentation around *openHalo* raises several technical and organizational challenges for Clever Cloud. The objective is to assess the feasibility of a transparent migration from MySQL to PostgreSQL and to measure its impacts in terms of compatibility, performance, and operations.

The main work axes assigned to the student team are as follows :

TABLE 2 – Project Challenges and Development Axes

Challenge	Description
Functional Compatibility	Define a methodology for reproducible tests to evaluate the level of compatibility between openHalo and MySQL (SQL queries, functions, data types). An automated testing tool should generate a compatibility score.
Performance	Set up benchmarks comparing openHalo and MySQL : response times, throughput under load, CPU and memory usage, in order to quantify the differences.
Configuration Migration	Develop scripts to automatically translate MySQL configuration files to openHalo, identifying unsupported parameters.
Documentation and Usage	Write installation, configuration, and testing documentation. Produce a user guide accessible to technical teams.

2.4. Stakeholders

The project involves various actors, each having a specific role in defining requirements, monitoring, and execution.

TABLE 3 – Stakeholders Table

Stakeholder	Role	Contact
Client : Aurélien Hébert	Requirement framing, strategic decisions, validation of deliverables	Email
Client : Tanguy Mazé	Requirement framing, strategic decisions, validation of deliverables	Email
School Supervisor	Pedagogical follow-up, feedback, arbitration if necessary	Email
Project Team	Production of deliverables (documentation, tests, benchmarks, configuration migration)	Internal tools (Drive, chat, Git) and WhatsApp

Project Team Composition : Sei Bayle, Nélia Fedele, Hugo Bentata, Mohamed Montadhar Ettaieb, François-Xavier Collot, Juliette Faurie and Houda Daouairi.

3. Specifications

3.1. Reformulation of the need

The main requirement of this project is to provide a reliable and comprehensive solution for migrating MySQL applications to PostgreSQL through OpenHalo. Our contribution lies in developing and designing an automated testing software to evaluate the system's functional compatibility and performance, while also producing detailed documentation and support tools to facilitate deployment, configuration, and adoption of the solution in real-world environments.

3.2. Service and Constraint Functions

TABLE 4 – Specifications – Main, Secondary, and Constraint Functions for OpenHalo

Function	Criteria	Level(s)	Tolerance	Priority
FP1 – Documentation				
FP1.1 – Getting Started Guide	In English, easy to understand	-	Low	1
FP1.2 – Installation Guide	Adaptable to different environments, including CleverCloud deployment	Local and cloud test	Low	2
FP2 – Automated Testing Software				
FP2.1 – Test Methodology	Simple and complex scenarios	-	Medium	1
FP2.2 – Cover Simple Queries	Verify MySQL → OpenHalo translation	Multiple query types	Low	1
FP2.3 – Cover Complex Queries	Complex joins, heavy transactions	Large data sets	Medium	2
FP2.4 – Test Documentation	Document scenarios, results, and best practices for the community	Complete documentation	Low	1
FP3 – Performance Analysis and Evaluation				
FP3.1 – Measurement by Data Size	Compare performance on small and large databases	Different data volumes	Medium	2
FP3.2 – Benchmark Evaluation	Measure transaction time, memory usage, CPU load	Repeated benchmarks	Medium	2
FP3.3 – Robustness and Reliability	Test stability with complex queries	Repeated tests	Medium	2
Secondary Functions – Bonus				
FS1 – Configuration Migration	Parse MySQL config and generate report with recommendations	Test on real configurations	High	3
FS2 – Automated Configuration	Automatically convert settings to OpenHalo format	Automatic .config file generation	High	3
FC – Constraint Functions				
FC1 – Environment Adaptability	Configuration and deployment on multiple OS and cloud environments	Multiple tests / On the CleverCloud platform	Medium	1

Continued on next page

Table 4 – continued

Function	Criteria	Level(s)	Tolerance	Priority
FC2 – Test Automation	Automatic execution without manual intervention	Automated tests	Low	2
FC3 – Data Integrity	No data loss or corruption during migration or tests	Post-test validation	Low	2
FC4 – Security and Isolation	Secure and isolated test environments	Best practices compliance	Medium	3

Legend : Priority levels : 1 = High, 2 = Medium, 3 = Low.

3.3. Validation/Final acceptance criteria

TABLE 5 – Summary of validation criteria for OpenHalo requirements

Function	Validation Criteria	Accepted Difference / Tolerance
FP1.1 – Getting Started Documentation	User can install and run OpenHalo in <30 minutes without external help on an targeted environment.	±15 minutes
FP1.2 – Installation Documentation	Operational in ≥ 90% of environments (Linux, Windows, CleverCloud).	±5% variation acceptable
FP2.1 – Test Methodology	At least 95% of scenarios run without error and results are reproducible.	±5% of scenarios may fail
FP2.2 – Simple Query Coverage	Data consistency ensured with zero loss or corruption.	No tolerance allowed
FP2.3 – Complex Query Coverage	90% of complex queries translated correctly. Execution time within PostgreSQL range.	±15% execution time
FP2.4 – Test Documentation	Structured document describing tests, results, and best practices is available.	Minor editorial errors tolerated
FP3.1 – Performance by Data Size	Performance degradation remains close to PostgreSQL across datasets.	±20% compared to PostgreSQL
FP3.2 – Benchmark Evaluation	Additional resource usage ≤25% compared to PostgreSQL (CPU, memory, transaction time).	±5% extra usage
FP3.3 – Robustness and Reliability	No crashes or critical errors during 24h stress test with concurrent queries.	Occasional slowdowns tolerated
FS1 – Configuration Migration (Bonus)	≥95% of common MySQL parameters parsed and translated, with clear report.	±5% unsupported options
FS2 – Automated Configuration (Bonus)	Auto-generated `.config` runs OpenHalo correctly in ≥80% of cases.	±10% manual adjustments
FC1 – Environment Adaptability	Installation and execution successful on Linux, Windows, MacOS, and CleverCloud.	One environment failure tolerated

Continued on next page

Table 5 – continued

Function	Validation Criteria	Accepted Difference / Tolerance
FC2 – Test Automation	≥ 95% of the test suite runs automatically via CI/CD pipeline.	±5% manual reruns
FC3 – Data Integrity	Checksums confirm no corruption or loss before and after migration/tests.	No tolerance allowed
FC4 – Security and Isolation	Tests run in isolated environments without unauthorized access or data leakage.	No tolerance allowed

4. Objectives and scientific/technical Methodology

4.1. Technical and scientific objectives

The goal of our project is clear : to confirm and validate the use of OpenHalo, developed by CleverCloud, as a solution to bridge the gap between the MySQL and PostgreSQL ecosystems. To achieve this, we must provide complete and accessible documentation covering the installation, configuration, and use of OpenHalo, so that the tool can be immediately deployed by a client or any user wishing to adopt it. At the same time, we will rigorously evaluate its performance and compatibility through a structured testing protocol, and draw conclusions about its applicability in a professional setting.

From a technical perspective, the objective is to design a reproducible deployment environment, with an architecture capable of installing and running OpenHalo easily, for instance by means of a ready-to-use configuration file. We must also develop an automated testing framework that will apply different queries to the databases, in order to compare results and performance between a query executed directly in MySQL and the same query translated by OpenHalo into PostgreSQL. For this development, several programming languages may be considered, including Python, C, or even Rust, depending on available resources and expertise.

From a scientific perspective, it is essential to provide detailed documentation on the use of OpenHalo, in order to contribute to the community and facilitate its dissemination. The tests we conduct will help assess the compatibility and robustness of the tool across multiple scenarios : executing simple queries to check syntactic and semantic compliance, testing MySQL-specific queries to analyze their translation and effectiveness, or experimenting with different data volumes to observe their impact on performance. Finally, we will analyze OpenHalo's behavior under demanding operations such as complex joins or heavy transactions, in order to determine whether such workloads affect its performance and stability.

Through this dual technical and scientific approach, our project therefore aims to provide both a concrete feedback on the use of OpenHalo and a contribution to the broader discussion on the unification of relational database management systems. This is not only about evaluating an emerging tool, but also about questioning its ability to meet real-world business needs and to contribute to the simplification of the technological landscape of database systems.

4.2. Chosen scientific methodology

The methodology adopted for this project combines a pragmatic technical approach with a rigorous scientific process, in order to evaluate OpenHalo under conditions close to those encountered in a professional environment.

First, an exploratory study will be conducted to analyze the available documentation on OpenHalo and similar projects. This step aims to understand the internal architecture of the tool, its integration with PostgreSQL, and its mechanisms for MySQL protocol compatibility. It will also help identify known limitations and provide the basis for clear and enhanced documentation that is accessible to any user.

Next, a reproducible deployment infrastructure will be designed. This infrastructure can be implemented locally but also in a cloud environment, for example on the CleverCloud platform, to benefit from real deployment conditions and scalability. The use of a standardized configuration file (.config) will allow automated environment setup and ensure test reproducibility. This step aims to make OpenHalo easy to use and portable, regardless of the execution context.

The experimental phase will rely on the construction of a precise and rigorous testing protocol. This protocol will cover different scenarios, ranging from basic queries to more complex and MySQL-specific

queries. Experiments will be conducted on various types of datasets : artificial databases designed to test specific cases, as well as more realistic datasets. In this context, it may be possible to request access from CleverCloud to representative client databases, in order to confront OpenHalo with real-world use cases and better evaluate its relevance in production.

To automate these experiments, dedicated testing software will be developed. This software will systematically execute the queries defined in the protocol, compare the results obtained between MySQL and OpenHalo, and measure performance according to various indicators (execution time, resource consumption, stability). The tool can be developed in Python, for its simplicity and extensive database- and benchmarking-oriented libraries, but alternatives such as C or Rust could also be considered to increase performance and measurement precision.

Finally, the data collected during the tests will be subject to in-depth analysis. The results will allow us to evaluate the accuracy of the translation between MySQL and PostgreSQL, identify potential differences in behavior, and measure the impact of data size and query complexity on performance. The use of real client databases, if feasible, will provide an additional dimension by allowing OpenHalo to be validated under conditions very close to those encountered in companies.

In addition to this methodology, a supplementary component of the project will focus on configuration migration. We plan to develop a tool capable of automatically parsing existing MySQL configuration files, converting the parameters to the format expected by OpenHalo, and flagging options that are unsupported or whose equivalence is not direct. This tool could also generate a detailed migration report indicating successfully converted parameters, unsupported parameters, and practical recommendations for adapting or replacing certain options. The goal is to simplify the transition from MySQL to OpenHalo, supporting users not only in deploying the tool but also in progressively migrating their existing environments.

This methodology thus ensures a dual objective : producing reliable and reproducible scientific results, and providing a concrete evaluation that can be directly exploited by the community and industrial stakeholders interested in OpenHalo.

5. Organization and planning

5.1. Work Breakdown Structure (WBS)

The diagram below illustrates the Work Breakdown Structure (WBS) for the project with Clever Cloud. It provides a clear overview of the main work packages and how the tasks are organized to achieve the project objectives and deliverables.

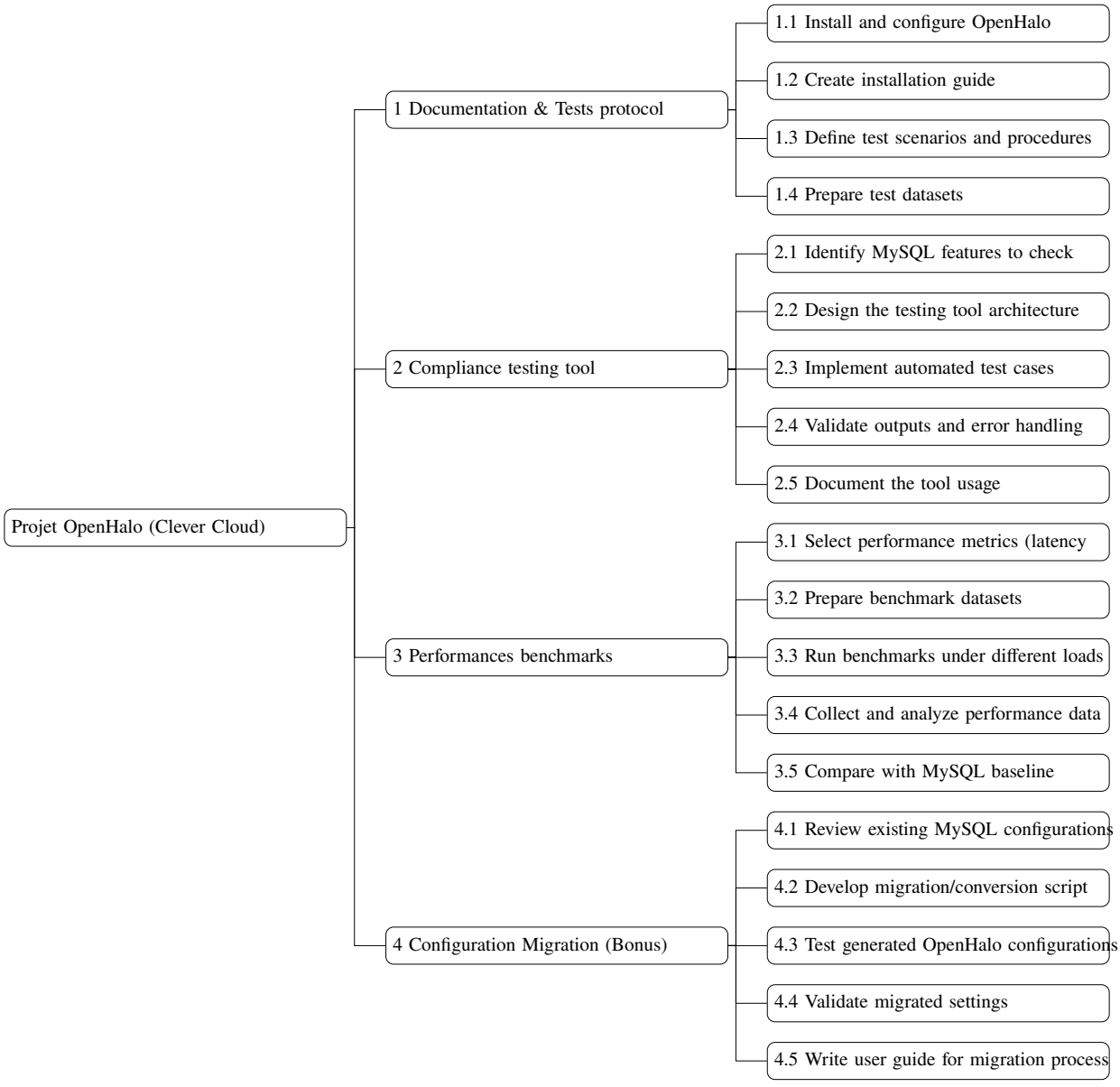


Figure 1 – Work Breakdown Structure of the project

The project is divided into four main packages : Documentation & Test Protocol, Compliance Testing Tool, Performance Benchmarks, and Configuration Migration (optional). Each package is broken down into specific, actionable tasks. For instance, the Documentation & Test Protocol package includes installing and configuring OpenHalo, preparing an installation guide, and defining test scenarios. The Compliance Testing Tool package covers identifying MySQL features to test, designing and implementing the testing tool, and validating its outputs. Performance Benchmarks involve preparing benchmark datasets, measuring response times under load, and comparing the results with MySQL. Finally, the Configuration Migration package

focuses on generating OpenHalo configurations automatically from existing MySQL setups.

This WBS allows the team to plan and track the project effectively. Each package is associated with one or more deliverables agreed with Clever Cloud :

Package 1 → Installation guide and test protocol documentation (20%)
Package 2 → Compliance testing tool (50%)
Package 3 → Performance measurement report (30%)
Package 4 (optional) → Automatic configuration script (bonus)

By defining the work packages upfront, the team can assign responsibilities, estimate effort for each member, identify dependencies between tasks, and anticipate potential risks. For example, delays in preparing the test protocol could directly impact the development of the testing tool. The WBS also supports coordination of sprints and reviews with Clever Cloud, ensuring feedback is received early and deliverables remain aligned with expectations.

Overall, the WBS serves as a roadmap for the project, showing not only what needs to be done but also how each task fits into the broader project context, from preparing the testing environment to delivering a fully functional tool and a comprehensive performance analysis.

5.2. Organization Chart and Responsibilities

Following the definition of work packages in the WBS, it is important to make the team's roles and responsibilities clear. With seven members collaborating on coding, documentation, testing, and performance evaluation, a structured approach ensures that tasks are properly assigned, decisions are taken efficiently, and no activity is overlooked.

The organizational chart provides a visual overview of the team structure, highlighting leadership, task managers, and supporting roles. It shows who supervises each area, who makes key decisions, and who serves as the main point of contact for questions or guidance.

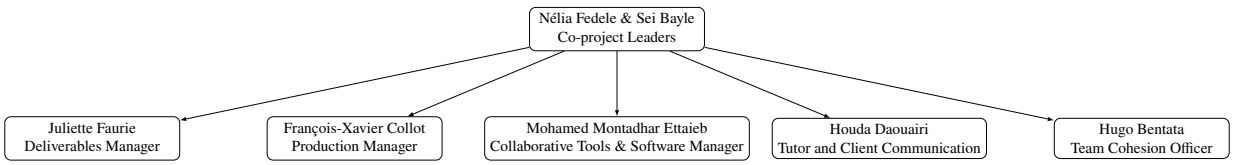


FIGURE 2 – Hierarchical organization chart of the project team

Complementing this, the RACI matrix specifies responsibility for each activity, indicating who executes the task, who validates or co-decides, and who contributes. It covers both technical work—such as developing the testing tool, running benchmarks, or implementing migration scripts—and coordination tasks, including communication with the client, management of collaborative tools, and team cohesion. Clearly defining these roles helps prevent overlaps, ensures accountability, and supports efficient collaboration across all project areas.

Project Task/Activity	Nélia	Sei	Juliette	François-Xavier	Montadhar	Houda	Hugo	Grégory	Clever Cloud
1 Documentation & Test Protocol									
1.1 Install and configure OpenHalo	A/C	A/C	C	R	C	C	C	A/C	A/C
1.2 Create installation guide	A/C	A/C	R	C	C	C	C	A/C	A/C
1.3 Define test scenarios and procedures	A/C	A/C	C	C	C	C	R	A/C	A/C
1.4 Prepare test datasets	C	C	R	C	C	C	C	A/C	A/C
2 Compliance Testing Tool									
2.1 Identify MySQL features to check	A/C	A/C	C	R	C	C	C	A/C	A/C
2.2 Design testing tool architecture	A/C	A/C	C	R	C	C	C	A/C	A/C
2.3 Implement automated test cases	C	C	C	R	C	C	C	A/C	A/C
2.4 Validate outputs and error handling	C	C	C	R	C	C	C	A/C	A/C
2.5 Document the tool usage	C	C	R	C	C	C	C	A/C	A/C
3 Performance Benchmarks									
3.1 Select performance metrics	A/C	A/C	C	C	C	C	R	A/C	A/C
3.2 Prepare benchmark datasets	C	C	C	R	C	C	C	A/C	A/C
3.3 Run benchmarks under different loads	C	C	C	R	C	C	C	A/C	A/C
3.4 Collect and analyze performance data	C	C	C	R	C	C	C	A/C	A/C
3.5 Compare with MySQL baseline	C	C	C	R	C	C	C	A/C	A/C
4 Configuration Migration (Bonus)									
4.1 Review existing MySQL configurations	C	C	C	R	C	C	C	A/C	A/C
4.2 Develop migration/conversion script	A/C	A/C	C	R	R	C	C	A/C	A/C
4.3 Test generated OpenHalo configurations	C	C	C	R	C	C	C	A/C	A/C
4.4 Validate migrated settings	C	C	C	R	C	C	C	A/C	A/C
4.5 Write user guide for migration process	C	C	R	C	C	C	C	A/C	A/C
5 Communication/Coordination									
5.1 Client coordination / feedback	A/C	A/C	C	C	C	R	C	A/C	A/C
5.2 Reporting intermediate results	A/C	A/C	C	C	C	R	C	A/C	A/C
5.3 Align deliverables with Clever Cloud	A/C	A/C	R	C	C	R	C	A/C	A/C
6 Development Tools & Automation									
6.1 Configure Build & Deployment Pipelines	C	C	C	C	R	C	C	A/C	A/C
6.2 Configure collaboration tools	C	C	C	C	R	C	C	A/C	A/C
6.3 Maintain repository	C	C	C	C	R	C	C	A/C	A/C
7 Team Cohesion/Well-being									
7.1 Organize team check-ins	C	C	C	C	C	C	R	A/C	A/C
7.2 Monitor workload	C	C	C	C	C	C	R	A/C	A/C
7.3 Support team morale	C	C	C	C	C	C	R	A/C	A/C

TABLE 6 – RACI table for project tasks

Legend : R= Responsible, A/C= Authority/Co-decider, C= Contributor

Together, the organizational chart and RACI matrix provide both a high-level view of the team and a detailed assignment of responsibilities. This combination facilitates coordination, highlights critical decision points, and ensures that the project can progress smoothly while maintaining quality and accountability.

5.3. Gantt chart

FIGURE 3 – Gantt chart of the project

The Gantt chart below presents the main phases of the project, the key deliverables, and the associated milestones. It covers the entire project timeline from the first meeting with the team and CleverCloud on September 22 to the final delivery on December 18.

The chart highlights the four main work packages identified in the WBS : Documentation & Test Protocol, Compliance Testing Tool, Performance Benchmarks, and the optional Configuration Migration. Each phase is represented with colored bars for clarity, and client-facing milestones are marked with diamonds to indicate key points of delivery and review.

This visual overview allows to quickly understand the project schedule, the sequencing of tasks, and the timing of major deliverables. It also helps the team coordinate work internally and ensures that deadlines and review points are clearly communicated.

6. Deliverables and resources

6.1. Deliverables Table

The table below summarizes the main deliverables of the project, with a short description, the team member responsible for coordination, and the scheduled delivery date to the client. Each deliverable corresponds to a task or milestone identified in the Gantt chart and shows how responsibilities are distributed within the team.

Deliverable	Description	Responsible	Scheduled Date
Initial project summary	Overview of project goals, scope, and planned milestones	Nélia & Sei	Sep 22
Installation guide	Step-by-step instructions to install and set up OpenHalo	Juliette	Week 3 (Oct 6)
Test protocol documentation	Description of test scenarios, expected results, and methods	Juliette	Week 4 (Oct 13)
Test datasets	Sample data used for compliance and performance tests	François-Xavier	Week 4 (Oct 13)
Test scenarios summary	Short table summarizing test cases for client review	François-Xavier	Week 4 (Oct 13)
Compliance testing tool	Software to check MySQL queries in OpenHalo	François-Xavier	Week 8 (Nov 3)
Tool validation report	Results of testing the compliance tool on defined scenarios	Nélia & Sei	Week 8 (Nov 3)
Performance benchmark data	Raw data collected during performance tests	François-Xavier	Week 10 (Nov 17)
Performance analysis report	Summary and analysis of OpenHalo performance vs MySQL	Nélia & Sei	Week 11 (Nov 24)
Performance comparison charts	Charts showing key metrics for the client	François-Xavier	Week 11 (Nov 24)
Configuration migration script (optional)	Script to convert MySQL configurations to OpenHalo	Montadhar	Week 12 (Dec 1)
Sample migrated configuration	Example configuration generated for client review	Montadhar	Week 12 (Dec 1)
Final delivery package	Full set of documentation, tools, and reports	Nélia & Sei	Week 12 (Dec 1)
Client feedback/Retrospective	Session to review results and get client feedback	Houda	Week 13 (Dec 8)

TABLE 7 – Project deliverables, responsible team members, and scheduled delivery dates

6.2. Table of input data expected from the client and scheduled dates

The table below lists the data and information expected from the client to carry out the project, along with the team member responsible for its integration and the expected delivery date.

Data / Information	Description / Use	Responsible	Expected Date
Existing MySQL configuration	To set up a reference environment and test migration	François-Xavier	Sep 29
Sample datasets	To run compliance and performance tests on OpenHalo	François-Xavier	Sep 29
Access credentials to Clever Cloud PaaS Databases	Required to deploy OpenHalo and MySQL instances	Montadhar	Sep 29
Client-specific test scenarios	Additional queries or edge cases to check for compliance	Juliette	Oct 2
Prioritized list of performance metrics	Define which metrics are most critical for benchmarking	Nélia & Sei	Oct 2
Configuration examples for migration (optional)	Sample MySQL configurations for automatic migration script	Montadhar	Oct 6
Feedback on interim deliverables	Comments on documentation, tool, and reports during project	Houda	Throughout project

TABLE 8 – Input data expected from the client, responsible team members, and expected dates of reception

6.3. Team skills and self-training needs

The team has a diverse set of skills, covering database management, software development, data analysis, and project coordination. However, some areas required for the project, such as OpenHalo deployment, advanced PostgreSQL queries, and performance benchmarking, are not fully mastered by all members. Each member will dedicate time to self-training in order to strengthen these skills and ensure the project objectives are fully met.

In particular, the team members bring the following backgrounds : Nélia specializes in Mathematical and Computational Engineering at IMT Atlantique, Sei in Distributed Systems Software Engineering, Juliette, Houda, and François-Xavier in Data Science, and Montadhar and Hugo in Collaborative Software Development. These backgrounds provide a strong foundation in both technical and analytical skills.

The diagram below illustrates the current proficiency levels and planned learning areas for key skills. It highlights both the strengths each member brings from their formation and the areas that will be enhanced through self-training. This approach ensures that the team will be able to address all aspects of the project efficiently and deliver high-quality results.

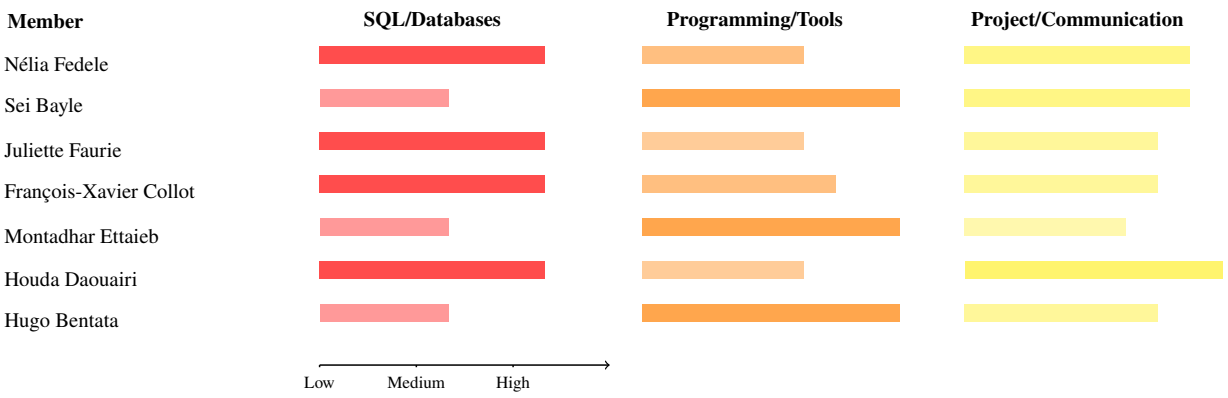


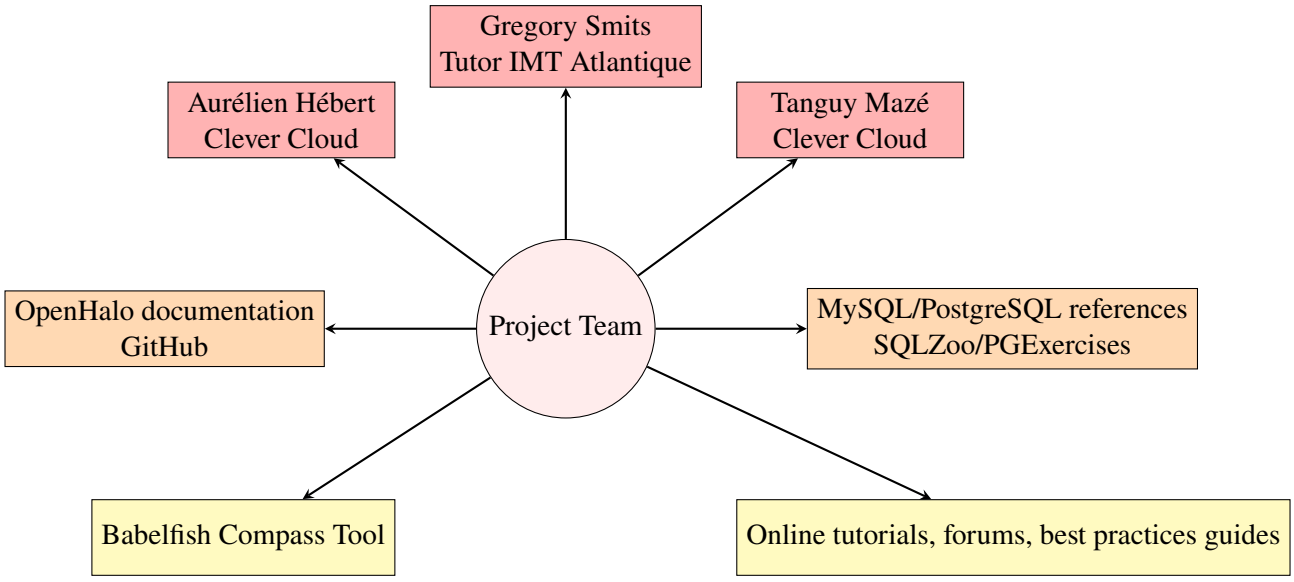
FIGURE 4 – Team Skills

The team will also spend time on self-training to cover areas that are less familiar. This includes

improving skills in PostgreSQL queries and performance optimization, learning the deployment and compatibility features of OpenHalo, and getting more practice with benchmarking and analyzing results. Members will also work on automating compliance tests, refining collaborative development practices, and producing clear documentation. These efforts will help the team fill knowledge gaps and handle the technical challenges of the project more effectively.

6.4. External resources/Experts mobilized

The project team relies on a variety of external and internal resources to ensure technical guidance, knowledge support, and best practice compliance. The diagram below illustrates the main experts, references, and tools available for the project.



These resources provide guidance on database deployment, OpenHalo usage, compliance testing, and performance benchmarking. They also help fill knowledge gaps in SQL and PostgreSQL, support tool validation, and ensure that project deliverables meet both client and academic expectations.

7. Communication and monitoring

7.1. Internal communication

Internal communication between team members is of paramount importance for making progress and keeping track of the various tasks assigned to us.

All group members meet twice a week to either work on the project or meet with the client/tutor. We've also created a group chat that has proven to be very helpful even in the early stages. So far, we've used it to share minutes of meetings, to collaborate on reports and to discuss the upcoming meetings in an effort to save time and to stay more organized.

7.2. Communication with CleverCloud

We've entrusted Houda with the task of communicating with our clients at CleverCloud (M. Hébert and M. Mazé) who requested weekly/biweekly email updates as well as in-person meetings at IMT where all members would be present.

We were also asked to deliver intermediate versions of our code as we deem a step finalized.

7.3. Document and code sharing

For code and document management, we initially set up a Github repository to ensure proper version control and collaborative development as well as a Google Drive workspace to easily access shared documents.

Since we are still in the early stages of the project, we remain flexible and are prepared to migrate the repository to GitHub if the client expresses such a preference.

Github link : https://github.com/fxcollot/openHaloDocumentation

7.4. Reporting procedure

Progress will be reported regularly to make sure work remains on the right track. Meeting minutes are also shared among group members.

Combined with everything mentioned above we hope that this reporting procedure will ensure that both the team and the client will remain informed and engaged.

8. Risk management and summary

8.1. Risk table

#	Risk description	Responsible	Identification date	W4	W5	W6	W7	W8	W9
1	Incomplete compatibility between MySQL and PostgreSQL (queries not supported or errors in translation)	François-Xavier	28/09						
2	Performance degradation on complex queries (joins, heavy transactions)	François-Xavier	28/09						
3	Data integrity issues during migration or testing (loss or corruption)	Sei	28/09						
4	Deployment issues on multi-environments (local, cloud, OS differences)	Montadhar	28/09						
5	Incomplete automation of testing scripts and configuration	Nelia	28/09						
6	Documentation unclear or not adapted to different users	Juliette	28/09						
7	Workload underestimated (development + documentation)	Houda	28/09						
8	Lack of technical expertise in some languages/tools (Python, C, Rust, PostgreSQL)	Sei	28/09						
9	No access to real client databases from CleverCloud (limiting realistic tests)	Hugo	28/09						
10	Benchmarks biased or not representative (unrealistic datasets)	Houda	28/09						

TABLE 9 – Risk tracking table for the OpenHalo project

- **Incomplete compatibility between MySQL and PostgreSQL :**
  Define a systematic testing protocol that covers both simple and complex queries to ensure comprehensive validation of the system's capabilities. Identify and document any unsupported queries, providing equivalent alternatives whenever possible to maintain functional continuity. Actively collaborate with the CleverCloud team to implement fixes or suggest improvements, fostering continuous enhancement and proactive issue resolution.

- **Performance degradation on complex queries :**
  Optimize indexes and query execution plans to improve performance and ensure efficient resource usage across different scenarios. When necessary, propose fallback strategies—such as using direct PostgreSQL queries—to guarantee reliability and maintain service continuity in cases where higher-level abstractions fall short.

- **Data integrity issues during migration or testing :**
  Implement validation checks both before and after the migration process to ensure data consistency

and correctness. Maintain secure backups of the original MySQL datasets as a safety measure, allowing recovery in case of unexpected issues. Automate integrity verification using mechanisms such as row counts and checksums to streamline the process and detect discrepancies efficiently.

- **Deployment issues on multi-environments :**
  Use containerization technologies like Docker to ensure environment consistency across development, testing, and production stages. Maintain standardized deployment scripts to streamline setup and reduce configuration errors. Test the solution across multiple platforms—including Linux, Windows, and various cloud environments—to ensure broad compatibility and reliability.

- **Incomplete automation of testing scripts and configuration :**
  Prioritize the automation of core test scenarios to ensure reliability and accelerate the validation process. Leverage CI/CD pipelines to enable automated execution and integration of tests within the development workflow. Additionally, document manual fallback steps to provide clear guidance in case of automation failure or unexpected issues.

- **Documentation unclear or not adapted to different users :**
  Prioritize the automation of core test scenarios to ensure reliability and accelerate the validation process. Leverage CI/CD pipelines to enable automated execution and integration of tests within the development workflow. Additionally, document manual fallback steps to provide clear guidance in case of automation failure or unexpected issues.

- **Workload underestimated (development + documentation) :**
  Assign clear responsibilities and define milestones to ensure accountability and structured progress throughout the project. Apply an agile methodology with weekly sprints and regular reviews to maintain momentum and adapt to evolving needs. In case of delays, re-scope or postpone non-essential features to keep the project on track and focused on critical deliverables.

- **Lack of technical expertise in some languages/tools :**
  Prioritize Python as the primary language for initial development to leverage its simplicity and strong ecosystem. Organize training sessions or pair programming initiatives to accelerate onboarding and promote knowledge sharing within the team. Rely on official libraries and documentation to ensure reliability, maintainability, and alignment with best practices.

- **No access to real client databases from CleverCloud :**
  Generate synthetic datasets that closely replicate real-world workloads to enable meaningful and realistic testing early in the process. When possible, request anonymized datasets from CleverCloud to further align test scenarios with production environments. Design tests to be flexible and easily adaptable, ensuring a smooth transition to real data when it becomes available.

- **Benchmarks biased or not representative :**
  Utilize a combination of artificial and semi-realistic datasets to cover a broad spectrum of testing scenarios. Compare the results against standard MySQL and PostgreSQL benchmarks to evaluate performance and accuracy effectively. Clearly document any limitations of the datasets used to provide context and transparency in the test outcomes.

8.2. Arbitration methods and internal team operation

To ensure smooth collaboration and avoid delays, the team has agreed on several principles and methods for working together. These practices aim to structure discussions, clarify responsibilities, and maintain a

positive and productive atmosphere throughout the project.

- **Regular coordination**
  A weekly meeting is held to review progress, reassign tasks when necessary, and anticipate potential risks. In addition, ad hoc meetings or quick online calls are organized whenever an urgent issue arises in order to prevent bottlenecks.

- **Conflict resolution and facilitation**
  Hugo Bentata acts as a neutral facilitator when tensions or disagreements occur, helping to structure discussions and reach a fair compromise. Disputes are first addressed directly between the concerned members, and if no agreement is reached, the issue is escalated to the co-leads for final arbitration.

- **Tracking and transparency**
  A shared online spreadsheet and a version-controlled repository document all tasks, deadlines, and progress, ensuring that everyone can see the project status at any time. Key decisions and meeting notes are also logged in order to keep track of commitments and rationales.

- **Leadership and motivation**
  Nélia Fedele and Sei Bayle provide leadership and set priorities, ensuring that goals remain aligned with the client's expectations. Team members rotate minor responsibilities, such as meeting facilitation or note-taking, in order to encourage engagement and shared ownership. Efforts are also made to balance workload and recognize contributions, thereby maintaining motivation and a collaborative spirit.

- **Decision-making principles**
  Decisions affecting the project scope or deliverables are taken collectively during meetings, with the co-leads having final approval. Technical choices are discussed openly, and where possible, alternatives are prototyped before adoption.

9. Conclusion

9.1. Next step

The next phase of the project consists in extending the scope of testing and validation. Functional compatibility tests should be broadened to cover a larger range of MySQL use cases, including complex queries, stored procedures, and edge cases in data types. Performance benchmarks must also be run on production-like workloads to evaluate scalability under realistic conditions. In parallel, further development of configuration migration scripts and enhanced documentation will be required to support seamless adoption by Clever Cloud teams and customers. Finally, the integration of the testing framework into the company's continuous deployment pipelines would represent a key milestone, enabling continuous validation of compatibility and performance over time.

9.2. Next actions of each member

Member	Next actions
Nélia	Ensure project management, track deadlines, organize weekly follow-up meetings
Sei	Coordinate tasks across the team, install and configure OpenHalo
Juliette	Draft and refine installation documentation for OpenHalo, including trouble-shooting guides and user-friendly onboarding material
François-Xavier	Lead deployment and adaptation of OpenHalo in multiple environments (Linux, Windows, cloud)
Montadhar	Maintain and improve the shared working environment, manage the Git repository, and ensure proper collaboration workflows (branches, commits, merges).
Hugo	Lead deployment and adaptation of OpenHalo in multiple environments (Linux, Windows, cloud)
Houda	Ensure communication with the client and the academic supervisor, provide regular updates, and produce clear documentation for deliverables and results

TABLE 10 – Next planned actions for each member in the OpenHalo project

OUR WORLDWIDE PARTNERS UNIVERSITIES - DOUBLE DEGREE AGREEMENTS

3 CAMPUS

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

IMT Atlantique Bretagne–Pays de la Loire – **http://www.imt-atlantique.fr/**

Campus de Brest
Technopôle Brest-Iroise
CS 83818
29238 Brest Cedex 3
France
T +33 (0)2 29 00 11 11
F +33 (0)2 29 00 10 00

Campus de Nantes
4, rue Alfred Kastler
CS 20722
44307 Nantes Cedex 3
France
T +33 (0)2 51 85 81 00
F +33 (0)2 99 12 70 08

Campus de Rennes
2, rue de la Châtaigneraie
CS 17607
35576 Cesson Sévigné Cedex
France
T +33 (0)2 99 12 70 00
F +33 (0)2 51 85 81 99