



Université Hassan II de Casablanca  
Ecole Normale Supérieure de l'Enseignement Technique, Mohammedia

## **Rapport de Stage**

**Présenté en vue d'obtenir le diplôme d'Ingénieur d'Etat**

**SPECIALITE**

**Ingénierie Informatique – Big Data et Cloud Computing**

**II-BDCC**

**Elaboré par**

**Mohamed BOUKHLIF**

*Oracle PGX Graph Size Calculator*

*&*

*Data Studio Load Tests*

**Réalisé au sein de**

**Oracle Labs Morocco**

**Encadré par**

**Encadrants universitaires**

Mme. Nezha BENMOUSSA  
M. Mohammed QBADOU

**Encadrante professionnelle**

Mme. Rania MEDRAOUI

Année universitaire  
**2019 - 2020**



## Dédicaces

*Je dédie ce travail, comme preuve de respect et de reconnaissance à :*

*Ma chère mère & mon cher père (Que  
Dieu ait pitié de lui)*

*Pour l'éducation et le grand amour dont ils m'ont  
entouré depuis ma naissance et pour leurs patiences  
et leurs sacrifices.*

*Tous mes enseignants*

*Pour leurs temps et leurs connaissances afin de  
satisfaire nos interrogations.*

*Mes meilleurs amis*

*Pour leurs aides et leurs encouragements.*

# **Remerciements**

Nous tenons à remercier toutes les personnes qui ont participé de différentes façons à la réussite de ce stage.

Nous remercions Mme Rania MEDRAOUI : Ingénieur Sénior chez la société Oracle, de m'avoir assuré une meilleure intégration au sein de l'équipe Data Studio, et pour le mentorat qui ma donner durant cette période de stage.

Nous tenons également à remercier l'ensemble des membres de l'équipe Data Studio de m'avoir si bien accueilli, et nous remercions l'équipe PGX, l'équipe Oracle JET et mon équipe pour l'aide qui m'ont offert. Grâce à tous, j'ai pu effectuer les tâches qui m'incombait dans des conditions optimales.

Nous tenons aussi à remercier toute l'équipe pédagogique de l'ENSET Mohammedia, pour les connaissances acquises durant ces trois ans.

# Table des matières

Dédicaces.....	i
Remerciements .....	ii
Table des matières .....	iii
Liste des figures.....	vi
Liste des abréviations .....	vii
Introduction générale.....	1
Chapitre I. Présentation de l'entreprise .....	2
<b>I.1 INTRODUCTION.....</b>	<b>2</b>
<b>I.2 HISTOIRE DE L'ENTREPRISE .....</b>	<b>2</b>
I.2.1 Oracle .....	2
I.2.2 Oracle Labs .....	3
I.2.2.1 Stratégie de recherche.....	3
I.2.2.2 Projets en vedette.....	4
I.2.2.2.1 GraalVM.....	4
I.2.2.2.2 Parallel Graph AnalytiX (PGX).....	5
I.2.3 Oracle Labs Morocco .....	6
<b>I.3 L'EQUIPE D'ACCUEIL .....</b>	<b>6</b>
<b>I.4 CONCLUSION .....</b>	<b>7</b>
Chapitre II. Contexte du projet .....	8
<b>II.1 INTRODUCTION .....</b>	<b>8</b>
<b>II.2 LE PROJET DATA STUDIO.....</b>	<b>8</b>
II.2.1 Qu'est-ce que c'est que Data Studio.....	8
II.2.2 Architecture Data Studio .....	9
II.2.2.1 Le client .....	10
II.2.2.2 Le serveur .....	10
II.2.2.3 L'agent.....	10
II.2.2.4 Les interpréteurs .....	10
II.2.2.5 Le serveur PGX .....	10
<b>II.3 LE PROCESSUS DE DEVELOPPEMENT.....</b>	<b>10</b>
II.3.1 La phase 10 – Idée.....	11
II.3.2 La phase 11 – collecte des besoins .....	11
II.3.3 La phase 20 - Spécification .....	12
II.3.4 La phase 30 - prêt à signer.....	13
II.3.5 La phase 40 – Fonctionnalité spécifiée.....	13
II.3.6 La phase 50 – en progrès.....	13
II.3.7 La phase 60 – en revue (équipe).....	14
II.3.8 La phase 61 – en revue (Management).....	14
II.3.9 La phase 65 – Assurance qualité .....	15
II.3.10 La phase 70 - Master.....	16
<b>II.4 CONCLUSION.....</b>	<b>17</b>

Chapitre III. Présentation du projet .....	18
<b>III.1 INTRODUCTION</b> .....	<b>18</b>
<b>III.2 LE COMPOSANT « GRAPH SIZE CALCULATOR »</b> .....	<b>18</b>
III.2.1 Problématique .....	18
III.2.2 Cahier des charges .....	18
III.2.2.1 Objectif .....	18
III.2.2.2 Besoins fonctionnels .....	18
III.2.2.3 Besoins non fonctionnels .....	19
<b>III.3 LE BENCHMARK DE DATA STUDIO AVEC DES GRAPHS PGX</b> .....	<b>19</b>
III.3.1 Problématique .....	19
III.3.2 Objectif .....	19
<b>III.4 CONCLUSION</b> .....	<b>19</b>
Chapitre IV. Analyse et conception .....	20
<b>IV.1 INTRODUCTION</b> .....	<b>20</b>
<b>IV.2 DIAGRAMME DE CAS D'UTILISATION</b> .....	<b>20</b>
<b>IV.3 DIAGRAMMES DE CLASSES</b> .....	<b>21</b>
IV.3.1 Module de configuration .....	21
IV.3.2 Module de formatage des résultats .....	22
IV.3.3 Module principale .....	23
<b>IV.4 DIAGRAMME DE SEQUENCE</b> .....	<b>23</b>
<b>IV.5 CONCLUSION</b> .....	<b>24</b>
Chapitre V. Environnement de développement .....	25
<b>V.1 INTRODUCTION</b> .....	<b>25</b>
<b>V.2 ENVIRONNEMENT DE TRAVAIL</b> .....	<b>25</b>
V.2.1 Les logiciels et les langages .....	26
V.2.1.1 IntelliJ IDEA .....	26
V.2.1.2 Apache JMeter .....	26
V.2.1.3 Git .....	27
V.2.1.4 HTML .....	27
V.2.1.5 CSS .....	28
V.2.1.6 SASS .....	28
V.2.1.7 TypeScript .....	29
V.2.1.8 JQuery .....	29
V.2.1.9 Oracle JET .....	30
V.2.1.10 Jest .....	31
V.2.1.11 Jenkins .....	32
V.2.1.12 SonarQube .....	32
V.2.1.13 La stack Atlassian .....	33
V.2.1.13.1 Jira .....	33
V.2.1.13.2 Confluence .....	33
V.2.1.13.3 Bitbucket .....	34
<b>V.3 CONCLUSION</b> .....	<b>34</b>
Chapitre VI. Réalisation .....	35
<b>VI.1 INTRODUCTION</b> .....	<b>35</b>
<b>VI.2 LE « GRAPH SIZE CALCULATOR »</b> .....	<b>35</b>
VI.2.1 La maquette de l'interface utilisateur .....	35

VI.2.2	Calcul du taux d'erreur .....	35
VI.2.3	Captures d'écran .....	36
<b>VI.3</b>	<b>LE BENCHMARK DE DATA STUDIO AVEC DES GRAPHES PGX.....</b>	<b>37</b>
VI.3.1	Le premier plan de test.....	39
VI.3.2	Le deuxième plan de test.....	41
VI.3.3	Résultats du benchamark .....	44
<b>VI.4</b>	<b>CONCLUSION.....</b>	<b>45</b>
	Conclusion générale .....	46
	Bibliographie .....	47
	Annexes .....	48

## Liste des figures

Figure 1 - Logo Oracle.....	2
Figure 2 - Logo Oracle Labs .....	3
Figure 3 - L'équipe Data Studio.....	6
Figure 4 - Architecture Data Studio .....	9
Figure 5 - Exemple de ticket Jira .....	11
Figure 6 - Cycle de vie de ticket (type feature).....	11
Figure 7 - Exemple de liste des reviseurs.....	13
Figure 8 - Exemple de Pull Request.....	14
Figure 9 - Exemple de Pull Request prête à être fusionner .....	14
Figure 10 - Le pipeline "Test And Merge".....	15
Figure 11 - Résultats de SonarQube affichés dans une Pull Request.....	16
Figure 12 - Aperçu des analyses de SonarQube pour une branche .....	16
Figure 13 - Diagramme de cas d'utilisation.....	20
Figure 14 - Diagramme de classes - Module de configuration .....	21
Figure 15 - Diagramme de classes - Module de formatage.....	22
Figure 16 - Diagramme de classes - Module principal .....	23
Figure 17 - Diagramme de séquence.....	24
Figure 18 – Logo IntelliJ IDEA .....	26
Figure 19 - Logo Apache JMeter .....	26
Figure 20 - Logo Git .....	27
Figure 21 - logo HTML.....	27
Figure 22 - Logo CSS 3 .....	28
Figure 23 - Logo SASS.....	28
Figure 24 - Logo TypeScript.....	29
Figure 25 - Logo JQuery .....	29
Figure 26 - Logo Oracle JET.....	30
Figure 27 - Logo Jest.....	31
Figure 28 - Logo Jenkins.....	32
Figure 29 - Logo SonarQube.....	32
Figure 30 - Logo Jira.....	33
Figure 31 - Logo Confluence .....	33
Figure 32 - Logo Bitbucket .....	34
Figure 33 - Une partie des résultats de calcul d'erreur .....	36
Figure 34 - Le "Graph Size Calculator" (mode normal) .....	36
Figure 35 - Le "Graph Size Calculator" (mode avancé).....	37
Figure 36 - plan de test 2 - évolution des threads d'exécution .....	37
Figure 37 - plan de test de chargement du graphe.....	39
Figure 38 - Plan de test 2 - Investigation des fraudes .....	41
Figure 39 - JMeter - Random Variable .....	43
Figure 40 - Les résultats généraux de l'exécution .....	44
Figure 41 – Benchmark - Le throughput (débit req/s).....	44
Figure 42 - Benchmark - La consommation mémoire .....	45



## **Liste des abréviations**

UML	Unified Modeling Language
PGX	Parallel Graph AnalytiX
PGQL	Property Graph Query Language
API	Application Programming Interface
R&D	Recherche et Développement
REST	Representational State Transfer
QA	Quality Assurance
JDBC	Java Database Connectivity
FTP	File Transfer Protocol
LDAP	Lightweight Directory Access Protocol
JMS	Java Messaging Services
HTTP	Hyper Text Transfer Protocol
TCP	Transmission Control Protocol
DOM	Document Object Model
W3C	World Wide Web Consortium

# Introduction générale

Chaque jour, l'humanité produit près de 2,5 trillions d'octets de données. A tel point que 90% des données dans le monde ont été créés au cours des deux dernières années seulement. Avec ce grand flux de données, les entreprises se retrouvent confrontées à des nouveaux problématiques. Leurs opérations quotidiennes génèrent des données en volume croissant qu'elles se doivent d'exploiter pleinement au risque de perdre rapidement, voire définitivement, leur avantage concurrentiel.

Ainsi, la science de données présente un intérêt pour les entreprises. Un intérêt sur quasi-toutes leurs activités : détection d'opportunités, conception et introduction de nouveaux produits et services, choix d'implantation géographique, optimisation de la relation client et des flux logistiques, maîtrise des risques opérationnels, détection de fraude, sécurisation et recrutement...

Le scientifique de données (Data Scientist) a besoin d'un outil accessible, intuitif et maniable qui lui permettra de faire facilement des modifications et des adaptations de code, vérifier le contenu de données, valider les hypothèses, tester différents algorithmes, présenter les résultats sous forme de tableaux de bord, et les partager en un clic. Les notebooks répondent parfaitement à ce besoin, en proposant un modèle web qui offrent toutes les avantages mentionnés et plus.

Oracle Data Studio est l'un des notebooks qui existe sur le marché, est c'est le projet que j'avais la chance de participer à son développement durant mon stage.

Ce manuscrit reporte l'expérience de mon stage de fin d'études au sein de la société Oracle, une expérience riche en termes d'environnement de travail, nouveaux moyens de communication et atouts relationnels.

# Chapitre I. Présentation de l'entreprise

## I.1 Introduction

Dans ce chapitre nous allons présenter l'entreprise Oracle qui représente l'entreprise d'accueil de mon stage PFE, nous allons aussi détailler les différentes équipes qui constitue Oracle Labs, ainsi que l'équipe d'accueil de mon stage.

## I.2 Histoire de l'entreprise

### I.2.1 Oracle



Figure 1 - Logo Oracle

Oracle (Oracle Corporation), ayant le logo dans la figure 1, est une entreprise américaine créée en 1977 par Larry Ellison. Ses produits phares sont Oracle Database (un système de gestion de base de données), Oracle Weblogic Server (un serveur d'applications), Oracle E-Business Suite (un progiciel de gestion intégré) et Oracle Cloud Infrastructure (une offre de Cloud Computing). En 2019, Oracle était la deuxième plus grande entreprise de logiciels en termes de chiffre d'affaires et de capitalisation boursière.

Tableau 1 – Fiche descriptive de l'entreprise Oracle

<b>Nom</b>	Oracle
<b>Date de création</b>	16 juin 1977
<b>Fondateurs</b>	<ul style="list-style-type: none"><li>• Larry Ellison</li><li>• Bob Miner</li><li>• Ed Oates</li></ul>
<b>Forme juridique</b>	Corporation

<b>Slogan</b>	Information driven
<b>Siège social</b>	Redwood Shores (Californie), États-Unis
<b>Activité</b>	Logiciel et programmation
<b>Effectif</b>	137 000 (en 2018)
<b>Site web</b>	<a href="https://oracle.com">https://oracle.com</a>
<b>Capitalisation</b>	186 milliards USD (en 2019)
<b>Chiffre d'affaires</b>	39,83 milliards USD (en 2018)
<b>Résultat net</b>	3,82 milliards USD (en 2018)

### I.2.2 Oracle Labs



**Figure 2 - Logo Oracle Labs**

La mission d'Oracle Labs, ayant le logo dans la figure 2, est simple : identifier, explorer et transférer de nouvelles technologies susceptibles d'améliorer considérablement les activités d'Oracle.

L'engagement d'Oracle envers la R&D est un facteur moteur dans le développement de technologies qui ont maintenu Oracle à la pointe de l'industrie informatique. Bien que bon nombre des technologies de pointe d'Oracle proviennent de ses organisations de développement de produits, Oracle Labs est la seule organisation d'Oracle qui se consacre exclusivement à la recherche.

#### I.2.2.1 Stratégie de recherche

Les chercheurs d'Oracle Labs recherchent de nouvelles approches et méthodologies, prenant souvent en charge des projets à haut risque ou incertitude, ou qui sont difficiles à gérer au sein d'une organisation de développement de produits. La recherche d'Oracle Labs se concentre sur les résultats du monde réel : nos chercheurs visent à développer des

technologies qui joueront un jour un rôle important dans l'évolution de la technologie et de la société. Par exemple, le multithreading de puces et le langage de programmation Java sont issus du travail effectué dans Oracle Labs.

Oracle Labs maintient un portefeuille de recherche équilibré avec quatre approches principales :

- Recherche exploratoire : Apporter les meilleurs et les plus brillants dans leurs domaines pour poursuivre leurs idées dans des domaines pertinents pour Oracle.
- Recherche dirigée : Travailler en collaboration avec les équipes de produits sur des problèmes difficiles et tournés vers l'avenir, en dehors du cadre du cycle de vie de la version du produit, mais en fonction des exigences du produit.
- Consultant : Fournir une expertise unique qui est utile dans les petits engagements dans de nombreuses organisations de produits.
- Incubation de produits : Fournir un endroit pour développer de nouveaux produits issus de la recherche Oracle Labs. L'incubation est nécessaire lorsque les résultats de la recherche n'ont pas de domicile naturel (comme c'est souvent le cas pour la recherche dans les différents domaines de produits), ou lorsque davantage de risques doivent être éliminés du travail pour démontrer sa valeur.

### **I.2.2.2 Projets en vedette**

#### ***I.2.2.2.1 GraalVM***

Les machines virtuelles de production actuelles (VM) fournissent une exécution de programmes à hautes performances uniquement pour une langue spécifique ou un très petit ensemble de langues. La compilation, la gestion de la mémoire et l'outillage sont maintenus séparément pour différentes langues, violant le principe de ne pas se répéter (DRY). Cela entraîne non seulement une charge plus importante pour les implémenteurs de VM, mais aussi pour les développeurs en raison de caractéristiques de performances, d'outils et de configuration incohérents. De plus, la communication entre des programmes écrits dans différentes langues nécessite une logique de sérialisation et de désérialisation coûteuse.

Enfin, les machines virtuelles hautes performances sont des processus lourds avec une empreinte mémoire élevée et une intégration difficile.

GraalVM est un projet de recherche pour explorer une nouvelle architecture pour les machines virtuelles. Notre vision est de créer une machine virtuelle unique qui offre des performances élevées pour tous les langages de programmation, facilitant ainsi la communication entre les programmes. Cette architecture prend en charge un outil unifié indépendant du langage pour une meilleure maintenabilité et son intégration pourrait rendre la machine virtuelle omniprésente dans la pile.

GraalVM prend en charge les langages basés sur JVM comme Java, Scala, Groovy ou Kotlin, JavaScript (y compris node.js), les langages se compilant en bitcode LLVM comme C, C++ ou Rust et les versions expérimentales de Ruby, R et Python.

#### ***1.2.2.2 Parallel Graph AnalytiX (PGX)***

Les graphiques sont une abstraction puissante pour permettre la découverte de connaissances à partir de relations dans de grands ensembles de données, grâce à leur représentation explicite des relations sous forme d'arêtes. L'analyse graphique révèle des informations latentes qui sont codées, non pas comme des champs dans les données, mais comme des relations directes et indirectes entre les éléments des données - des informations qui ne sont pas évidentes à l'œil nu, mais qui peuvent avoir une valeur énorme une fois découvertes.

PGX est une boîte à outils pour l'analyse de graphiques qui prend en charge à la fois les algorithmes en cours d'exécution tels que le PageRank sur les graphiques et la mise en correspondance de modèles de type SQL sur les graphiques, en utilisant les résultats des analyses algorithmiques. Les algorithmes sont parallélisés pour des performances extrêmes. La boîte à outils PGX comprend à la fois un moteur en mémoire à nœud unique et un moteur distribué pour les graphiques extrêmement volumineux. Les graphiques peuvent être chargés à partir de diverses sources, notamment des fichiers plats, des bases de données SQL et NoSQL et Apache Spark et Hadoop ; les mises à jour incrémentielles sont prises en charge.

PGX est à la fois déjà disponible en option dans les produits Oracle et un projet de recherche actif chez Oracle Labs, avec une équipe de chercheurs de classe mondiale qui fait progresser les capacités de la boîte à outils.

### I.2.3 Oracle Labs Morocco

Fondée en 2017, Oracle Labs Morocco représente une nouvelle branche des laboratoires d'Oracle. Oracle Labs a des bureaux à Redwood Shores, en Californie, à Burlington, au Massachusetts, à Cambridge au Royaume-Uni, à Brisbane en Australie, à Vienne en Autriche, à Zurich en Suisse et à Casablanca au Maroc.

Située à Casa Near Shore, Shore 14, 2ème étage, qui représentait le local d'Oracle au Maroc. Avant la création d'Oracle Labs, ce locale avait seulement les commerciaux d'Oracle, après la création, les développeurs et commerciales partagent ce locale en attendant la fin des travaux dans le nouveau local situé aussi à Casa Near Shore, Shore 23, 1er étage, ce nouveau local sera dédié aux collaborateurs d'Oracle Labs seulement.

### I.3 L'équipe d'accueil

Oracle Labs se constitue de plusieurs équipes, divisé selon les projets, voici quelques-uns :

- Equipe GraalVM
- Equipe PGX
- Equipe Data Studio
- Equipe DevOps

Durant notre stage, nous faisons partie de l'équipe Data Studio.

Vous trouverez une image des membres de l'équipe Data Studio dans la figure 3.

The Team (In Alphabetical Order)


















 @Achrif El Khandouli (intern)	 @Alexander Weld	 @Alexandra Fritzen	 @Claudio Loureiro (intern)	 @Daniel Langerenken	 @David Tichy
 @Felix Schwarzmeier	 @Florian Morath (intern)	 @Hamza Fawzi	 @Hamza Ihizan	 @Hassan Chafi	 @Julia Kindelsberger
 @Linus Handschin	 @Martijn Dwars	 @Matthias Brantner	 @Michiel Haisma	 @Mohamed Boukhliif (intern)	 @Mosab Atchane
 @Nabila Hamdaoui	 @Oussama Charafi (intern)	 @Rania Medraoui	 @Riva Nathans	 @Sabrina Senna	 @Youssef El Abbassi
 @Zakaryae Mazouzi					

Figure 3 - L'équipe Data Studio

## **I.4 Conclusion**

Dans ce chapitre, nous avons présenté la société Oracle, ainsi que les différentes équipes d'Oracle Labs. Nous constatons une grande diversité dans la culture d'Oracle due au fait qu'il est une multinationale, et cela représente un défi qui consiste à créer une homogénéité entre des collaborateurs ayant plusieurs nationalités, religions, traditions, etc.



---

# Chapitre II. Contexte du projet

---

## II.1 Introduction

Le projet Data Studio suit une méthodologie agile Scrum, en utilisant la stack Atlassian (détaillé dans le chapitre V). Oracle Labs utilise la version payante de la stack Atlassian, ce qui nous permet de profiter des toutes les fonctionnalités de ces outils. Dans ce chapitre nous allons voir en quoi consiste Data Studio, et nous allons détailler le processus de développement au sein du projet Data Studio.

## II.2 Le projet Data Studio

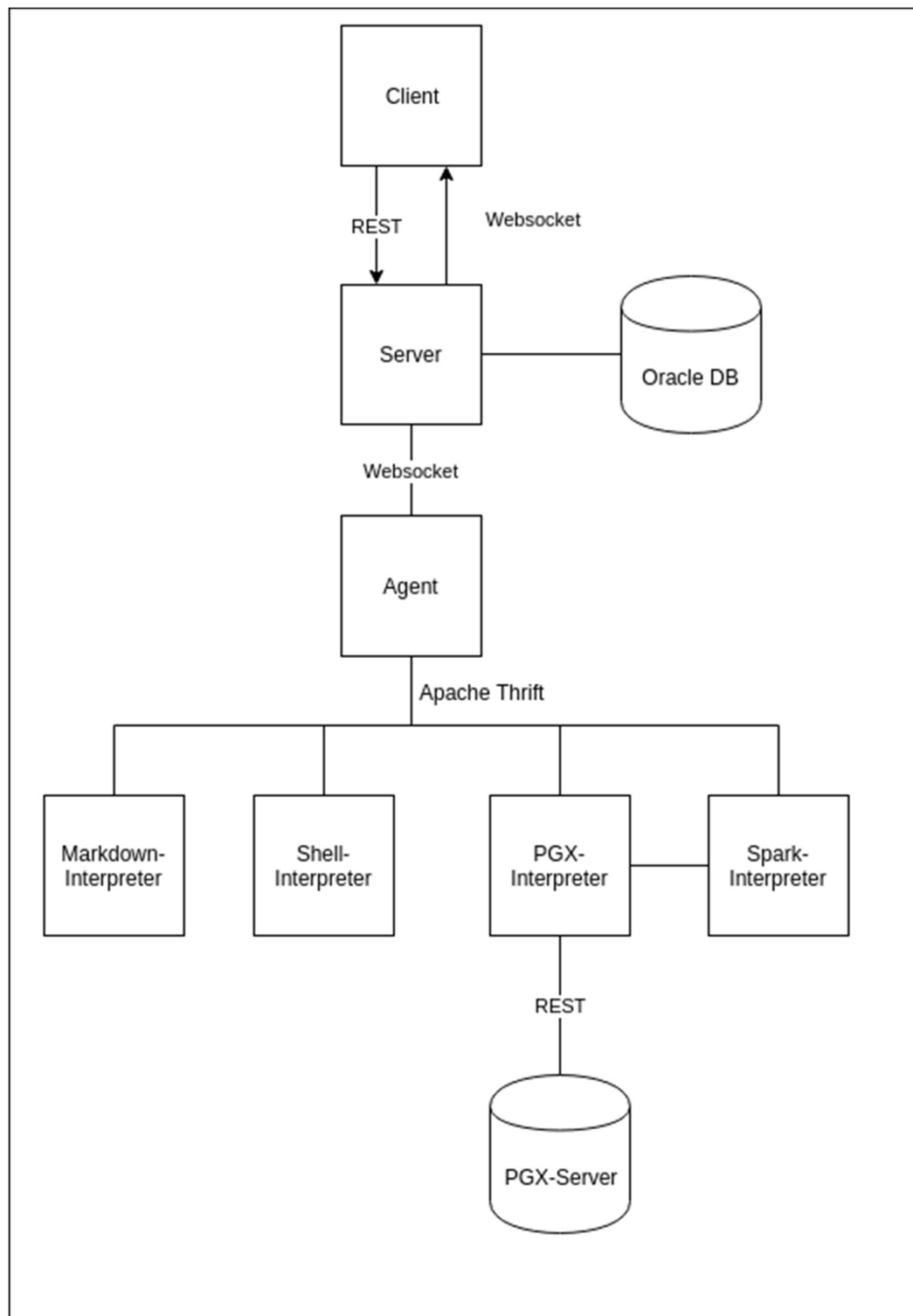
### II.2.1 Qu'est-ce que c'est que Data Studio

Oracle Labs Data Studio est une plate-forme de bloc-notes (notebooks) web pour les scientifiques des données. En combinant la collaboration de code en direct dans plusieurs langages de programmation avec des analyses de graphiques et des visualisations riches et interactives, Data Studio accélère le processus d'exploration et d'acquisition d'informations de vos données.

Les données peuvent être importées sur la plate-forme à partir de diverses sources (à partir de HDFS / Spark, de bases de données ou de fichiers) et analysées avec des environnements d'interpréteur pour une gamme de langages de programmation (Python, R, Shell, Spark et autres). Pour les données de graphique (pensez aux réseaux sociaux ou aux transactions financières), Data Studio est livré avec l'outil d'analyse graphique (PGX) et le langage de requête de graphe de propriétés (PGQL) d'Oracle Labs, ajoutant une couche visuelle interactive qui prend en charge le filtrage des graphiques, la mise en évidence des éléments, la visualisation géographique données, et élargir / contracter la vue, permettant aux utilisateurs d'explorer intuitivement de grands graphiques.

Les composants Data Studio constituent une base réutilisable pour les produits logiciels d'entreprise adaptés à des secteurs spécifiques. Les exemples d'utilisation incluent la détection et la conformité des délits financiers, l'apprentissage automatique pour les sciences de la santé et la segmentation du marché pour la vente au détail.

## II.2.2 Architecture Data Studio



**Figure 4 - Architecture Data Studio**

Le projet Data Studio se compose principalement des composants suivants (vue globale dans la figure 4) :

### **II.2.2.1 Le client**

Représente le client JavaScript du projet, il se repose sur le framework Oracle JET, ainsi que d'autres librairies tel que : JQuery, Knockout, etc. Ce composant offre des visualisations de graphes en plusieurs forme, toute en assurant une meilleur expérience utilisateur et en se concentrant sur l'accessibilité.

### **II.2.2.2 Le serveur**

Ce composant contient principalement la logique métier de Data Studio, il est aussi le point d'entrée du backend, donc il se charge de toutes les communications REST et WebSocket avec le client. Ce composant est un projet Spring boot, et il utilise une base de données Oracle.

### **II.2.2.3 L'agent**

Ce composant représente un intermédiaire entre le serveur et les interpréteurs, son but c'est juste de séparé la logique métier des interpréteurs dans un autre composant. Ce composant est un projet Spring boot avec Apache Thrift, le but de ce dernier est d'écrire un seul code et permettre la communication avec différents autres types d'interfaces (dans notre cas différent interpréteurs).

### **II.2.2.4 Les interpréteurs**

Data Studio utilise plusieurs interpréteurs afin d'assurer la compilation et l'interprétation de différent langages, ces interpréteurs offrent une API à utiliser pour assurer la communication. Parmi les interpréteurs, un est spécial, c'est l'interpréteur PGX, cet interpréteur ne peut pas fonctionner tout seul, mais il a besoin d'un serveur pour sauvegarder les graphes et faire les opérations nécessaires dessus.

### **II.2.2.5 Le serveur PGX**

Ce composant représente le serveur PGX, qui est essentiel pour la sauvegarde, exécution des requêtes sur les graphes. Il offre une API REST pour la communication, cet API est utilisé par l'interpréteur PGX.

## **II.3 Le processus de développement**

Le projet Data Studio est divisé en des tâches, sous forme de tickets Jira (détail dans le Chapitre V), chaque ticket à un type (feature, bugfix, build failure, task, etc), et il a une

description, avec possibilité de mettre des commentaires. Dans la figure 5 nous avons un exemple de ticket Jira qui représente le ticket de la fonctionnalité sujet de mon PFE.

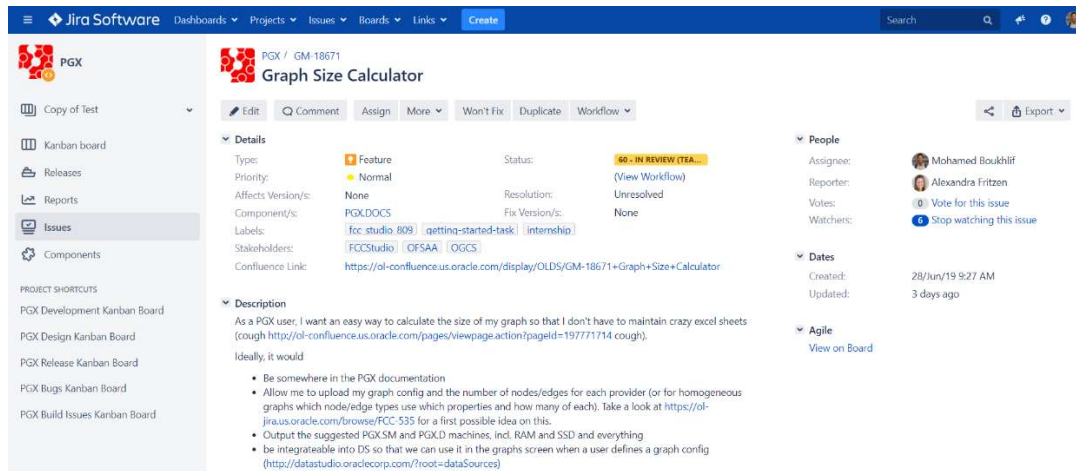


Figure 5 - Exemple de ticket Jira

Nous allons voir le cycle de vie d'un ticket (Jira), d'une nouvelle fonctionnalité (feature), dès sa création jusqu'à ce qu'il soit fermé, en d'autres termes, nous allons détailler le cycle de vie affiché dans la figure 6.

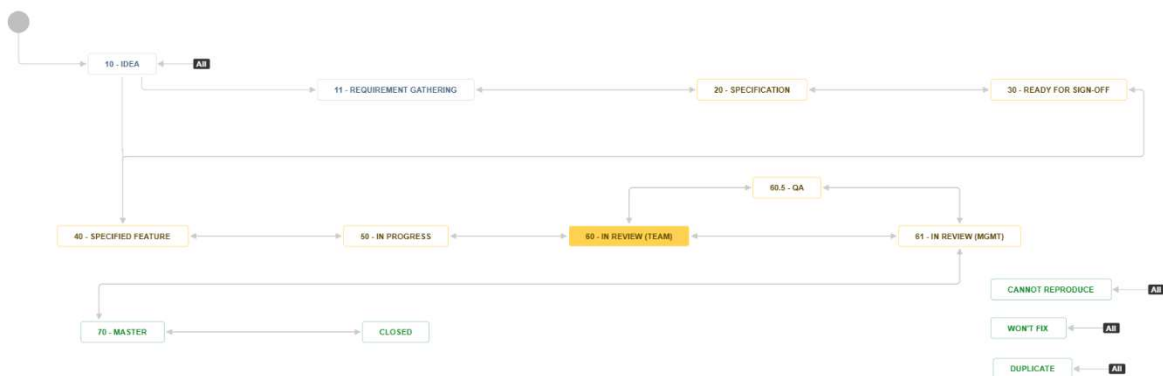


Figure 6 - Cycle de vie de ticket (type feature)

### II.3.1 La phase 10 – Idée

Le ticket commence avec l'état « IDEA », idée en français, dans cette l'étape il y a juste une petite description de la fonctionnalité que nous voulons ajouter.

### II.3.2 La phase 11 – collecte des besoins

Puis, le ticket passe vers l'étape suivante intitulé « REQUEREMENT GATHERING », en français : collecte des besoins, dans cette étape, la personne a qui le ticket est assigné, doit créer une page Confluence (voir le chapitre V pour plus de détails) lié à ce ticket, cette page doit être rempli avec tous les détails à savoir pour commencer le

développement de la tâche, en quelque sort, cette étape représente la phase de la création du cahier des charges de la fonctionnalité. (Voir l'annexe numéro 8 pour la page Confluence du Graph Size Calculator).

### II.3.3 La phase 20 - Spécification

Ensuite, on passe à la phase intitulée « SPECIFICATION », dans cette phase on doit ajouter des réviseurs (reviewers) pour commencer la révision de la fonctionnalité, dans cette phase, les réviseurs commencent à poser des questions sur l'ensemble des sections de la fonctionnalité, si quelque chose n'est pas claire, ou doit être justifié, ou ne doit pas être fait de cette façon, et éventuellement ils marquent dans la page qu'ils ont fait la révision.

La page doit contenir les sections suivantes :

- Spécification fonctionnelle
  - Le résultat attendu
  - Maquettes d'interface utilisateur
  - Description de l'API
- Spécifications de conception
  - Frontend
  - Backend
  - Testing
- User stories
- Questions en relation avec la sécurité

Après avoir eu la confirmation de tous les réviseurs, comme on peut voir dans la figure 7, la phase suivante c'est « READY FOR SIGN-OFF ».

## Reviewers

- ☒ @Alejandro De Gante
- ☐ @Alexander Weld
- ☒ @Alexandra Fritzen
- ☒ @Daniel Langerenken
- ☒ @Iraklis Psaroudakis
- ☒ @Michiel Haisma
- ☒ @Rania Medraoui
- ☒ @Sabrina Senna
- ☐ Hassan

Figure 7 - Exemple de liste des reviseurs

### II.3.4 La phase 30 - prêt à signer

Dans cette phase le responsable du ticket doit faire une réunion avec M. Hassan Chafi, qui représente le conseil de management, cette réunion se fait chaque mercredi à 16h:00. Comme tous les reviseurs, M. Hassan va consulter la page Confluence de la fonctionnalité, et il va discuter tout ce qui n'est pas claire ou a besoin de clarification. À la fin de la réunion, s'il y n'y a pas de changement demandé, le ticket passe à la phase « SPECIFIED FEATURE », sinon, le responsable part pour faire les changements demandés et revient le mercredi prochain pour les valider.

### II.3.5 La phase 40 – Fonctionnalité spécifiée

Cette phase est une phase transitoire, lorsque le responsable est prêt à commencer le développement, il va faire passer le ticket à la phase « IN PROGRESS »

### II.3.6 La phase 50 – en progrès

Dans cette phase le responsable du ticket, commence le développement de la fonctionnalité, tout en suivant les détails décrit dans la page Confluence.

Lorsque le développement du ticket est terminé, le responsable crée une nouvelle « Pull Request » pour fusionner la branche de la fonctionnalité avec la branche principale de Data Studio, le ticket passe à la phase suivante.

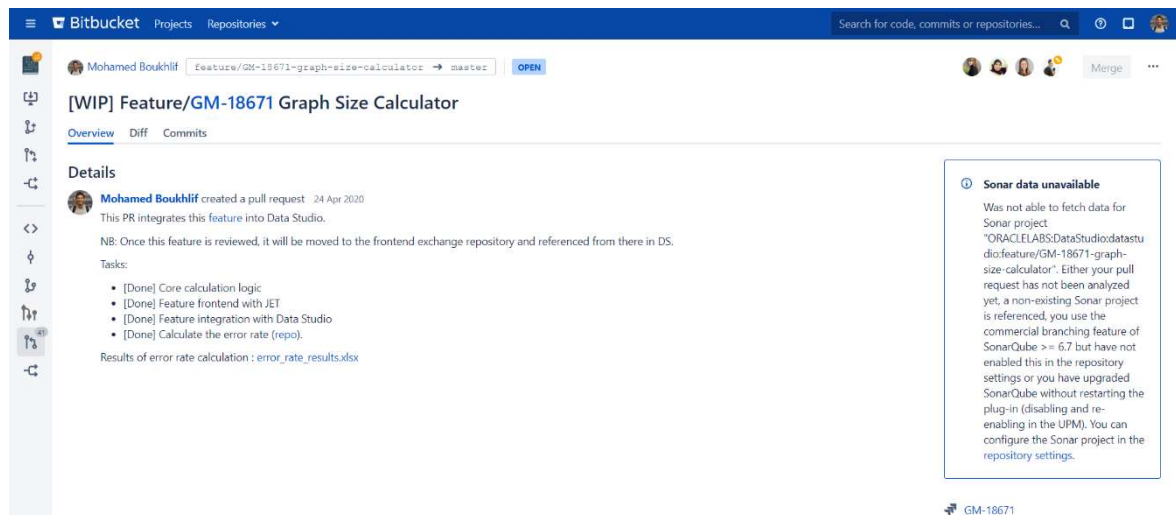


Figure 8 - Exemple de Pull Request

### II.3.7 La phase 60 – en revue (équipe)

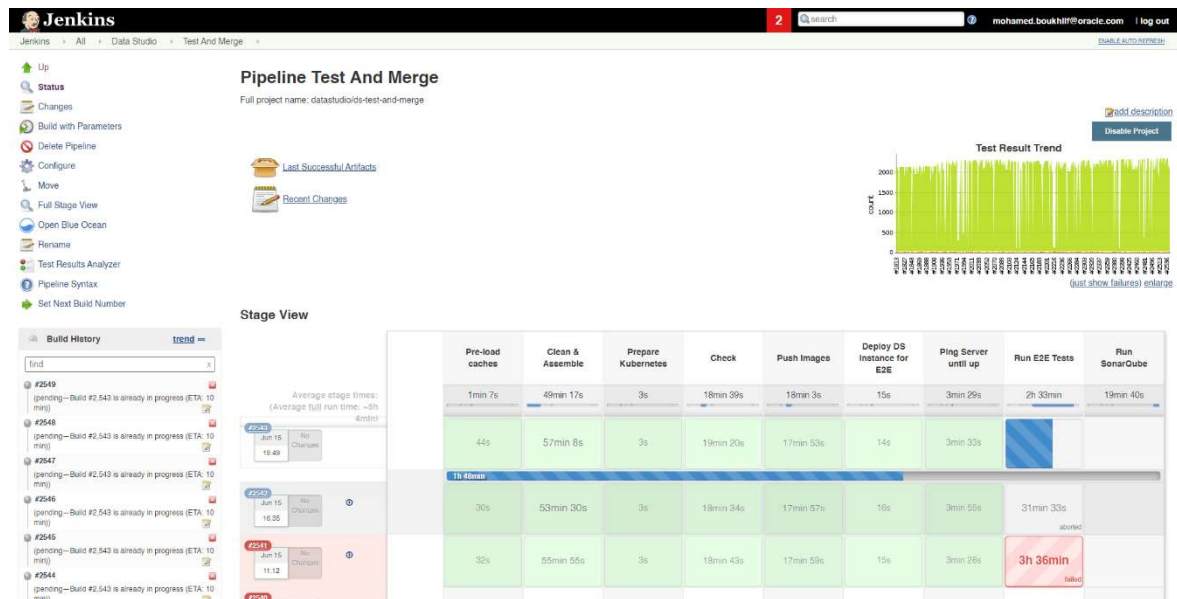
Dans cette phase « IN REVIEW (TEAM) », la même équipe qui a fait la révision dans la page Confluence, maintenant, cette équipe va faire la révision du code dans la « Pull Request », un exemple de Pull Request est capturé dans la figure 8. Les reviseurs vont commencer à demander des changements, le responsable fait ces changements et mis à jour la « Pull Request ». Le ticket reste dans cette phase jusqu'à ce que le toute l'équipe marque la révision.

### II.3.8 La phase 61 – en revue (Management)

La phase suivante c'est « IN REVIEW (MGMT) », où le responsable va ajouter M. Matthias Brantner, et M. Hassan Chafi, qui représentent le conseil de management, afin de faire une dernière révision du code avant de le fusionner avec la branche principale. Lorsqu'ils vont marquer la Pull Request comme révisée (comme dans la figure 9), en fin le responsable du ticket peut lancer le processus de fusion avec la branche principale de Data Studio.



Figure 9 - Exemple de Pull Request prête à être fusionner



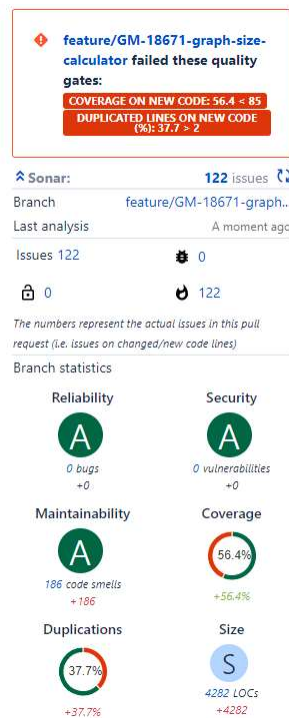
**Figure 10 - Le pipeline "Test And Merge"**

Le pipeline de Jenkins qui fait cette action est appelé « TEST & MERGE » (affiché dans la figure 10), il commence par créer une version de Data Studio à partir de la Pull Request, et il exécute les tests unitaires, les tests d'intégrations, les end-to-end tests, et en fin fait une analyse avec SonarQube, et si tous ces étapes été vert, Jenkins fusionne la branche du ticket avec la branche principale de Data Studio, et il fait aussi passer le ticket vers l'étape « MASTER ».

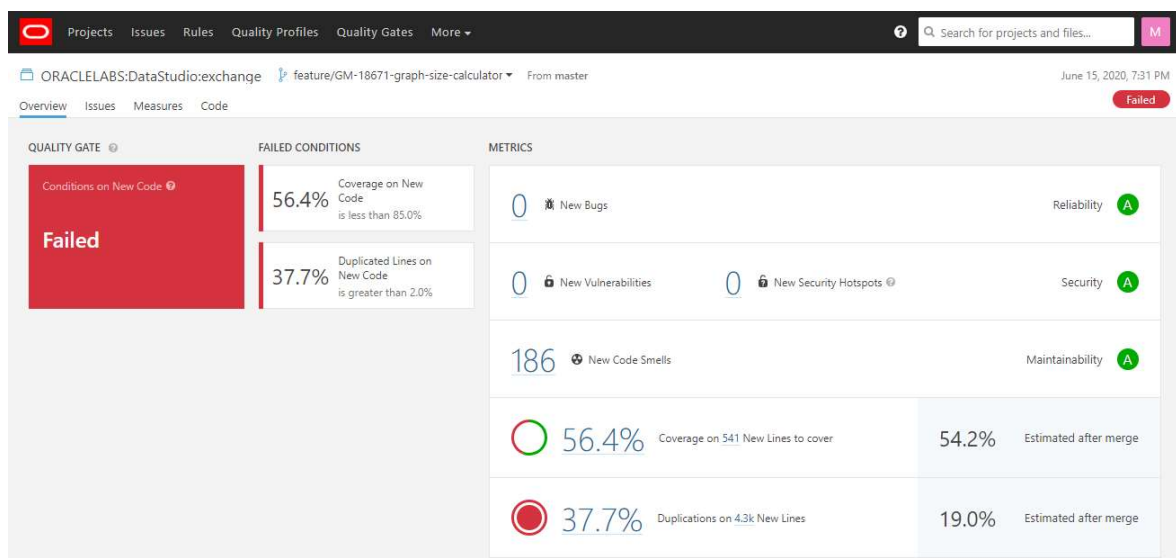
### II.3.9 La phase 65 – Assurance qualité

Après la phase du « IN REVIEW (TEAM) », le ticket peut passer à une phase optionnelle, qui est la phase « QA » pour « Quality Assurance », ou l'assurance qualité, dans cette phase le responsable peut ajouter des tests unitaires qu'il n'a pas fait dans la phase de développement, ou régler des problèmes signaler par SonarQube (comme présenté dans les figures 11 et 12).





**Figure 11 - Résultats de SonarQube affichés dans une Pull Request**



**Figure 12 - Aperçu des analyses de SonarQube pour une branche**

### II.3.10 La phase 70 - Master

Cette représente un ticket qui est fusionné avec la branche principale « master », rien ne se passe dans cette phase.

## II.4 Conclusion

Dans ce chapitre nous avons découvert Data Studio, ainsi que son processus de développement, ça peut apparaître long, et c'est vrai, mais c'est la seule façon à assurer une qualité logicielle, et assurer une homogénéité entre les membres des équipes dans le monde entier.

---

# Chapitre III. Présentation du projet

---

## III.1 Introduction

Notre projet est divisé en deux parties : la première (la grande) partie consiste à concevoir et développer un nouveau composant de Data Studio appelé « Graph Size Calculator », la deuxième partie consiste à créer un benchmark de Data Studio en utilisant des graphes PGX.

## III.2 Le composant « Graph Size Calculator »

### III.2.1 Problématique

Pour estimer la taille d'un graph, l'équipe PGX et Data Studio, utilisait un fichier Excel, où il faut pour chaque graphe insérer manuellement tous les caractéristiques (graphe partitionné/non partitionné, nombre de sommets, nombre d'arêtes, type de chaque propriété...)

#### Inconvénients

- Pour chaque graphe il faut un fichier Excel
- Saisie manuelle des caractéristiques du graphe
- Difficile de mettre le graphe à l'échelle
- Difficile à maintenir les formules de calcul entre les fichiers

### III.2.2 Cahier des charges

#### III.2.2.1 Objectif

Le but est de créer un outil capable d'estimer la taille mémoire des graphes PGX.

L'outil va prendre en entrée une configuration d'un graphe, ainsi que le nombre de nœuds et arêtes, et donne en sortie l'estimation de la mémoire On-Heap, Off-Heap et le Total.

#### III.2.2.2 Besoins fonctionnels

- Estimer la taille mémoire OnHeap/OffHeap d'un graphe PGX

- Un taux d'erreur inférieur à 1%

### III.2.2.3 Besoins non fonctionnels

- Fiabilité : L'outil doit fonctionner de façon cohérente sans erreurs.
- Exceptions/Erreurs : Les ambiguïtés doivent être signalées par des messages d'erreurs bien organisés pour bien guider l'utilisateur et le familiariser avec l'outil
- Une meilleure expérience utilisateur
- Aptitude à la maintenance et la réutilisation : L'outil doit être conforme à une architecture standard et claire permettant sa maintenance et sa réutilisation.

## III.3 Le benchmark de Data Studio avec des graphes PGX

### III.3.1 Problématique

Dans le cadre de la « Task Force : Brace Yourselves, Users Are Coming »<sup>1</sup>, nous voulons savoir l'impact de l'exécution de plusieurs de graphes PGX sur Data Studio.

### III.3.2 Objectif

Créer un cas d'utilisation (use case) en utilisant principalement 2 notebooks, le premier va générer un graphe avec PGX qui représente les transactions d'une banque donnée, et le deuxième va faire des investigations sur ces graphes afin de détecter des fraudes.

## III.4 Conclusion

Dans ce chapitre nous avons présenté les piliers de notre projet ainsi que les objectifs à atteindre durant notre stage PFE de 6 mois.

---

<sup>1</sup> Une task force est une collection de tickets ayant le même but. Par exemple nous avons une task force de benchmark et performance, une task force du frontend, une task force de sécurité, etc.

# Chapitre IV. Analyse et conception

## IV.1 Introduction

Dans ce chapitre, nous allons présenter la conception que nous avons menée, afin d'avoir une vision claire du chemin à suivre durant la phase de réalisation. Vue qu'il n'y a pas d'analyse et conception nécessaire pour la tâche du benchmark, ce chapitre est dédié au « Graph Size Calculator ».

## IV.2 Diagramme de cas d'utilisation

Comme présenté dans la figure 13, pour un utilisateur de Data Studio ou un utilisateur de PGX, il peut, soit faire une estimation de la taille du graphe, ou bien basculer vers le mode avancé.

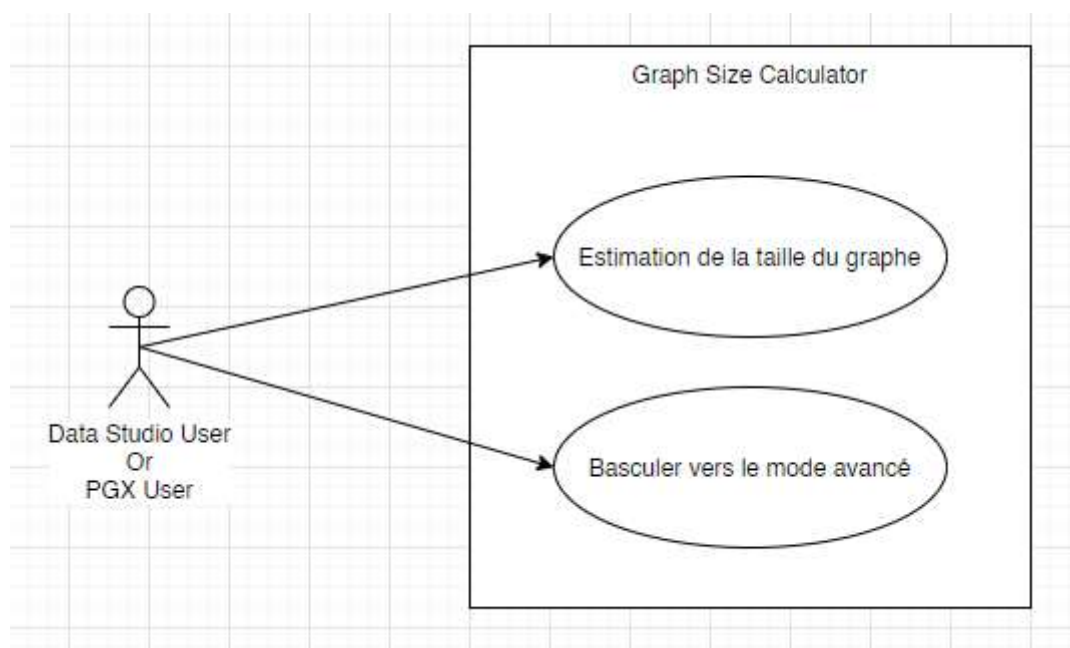
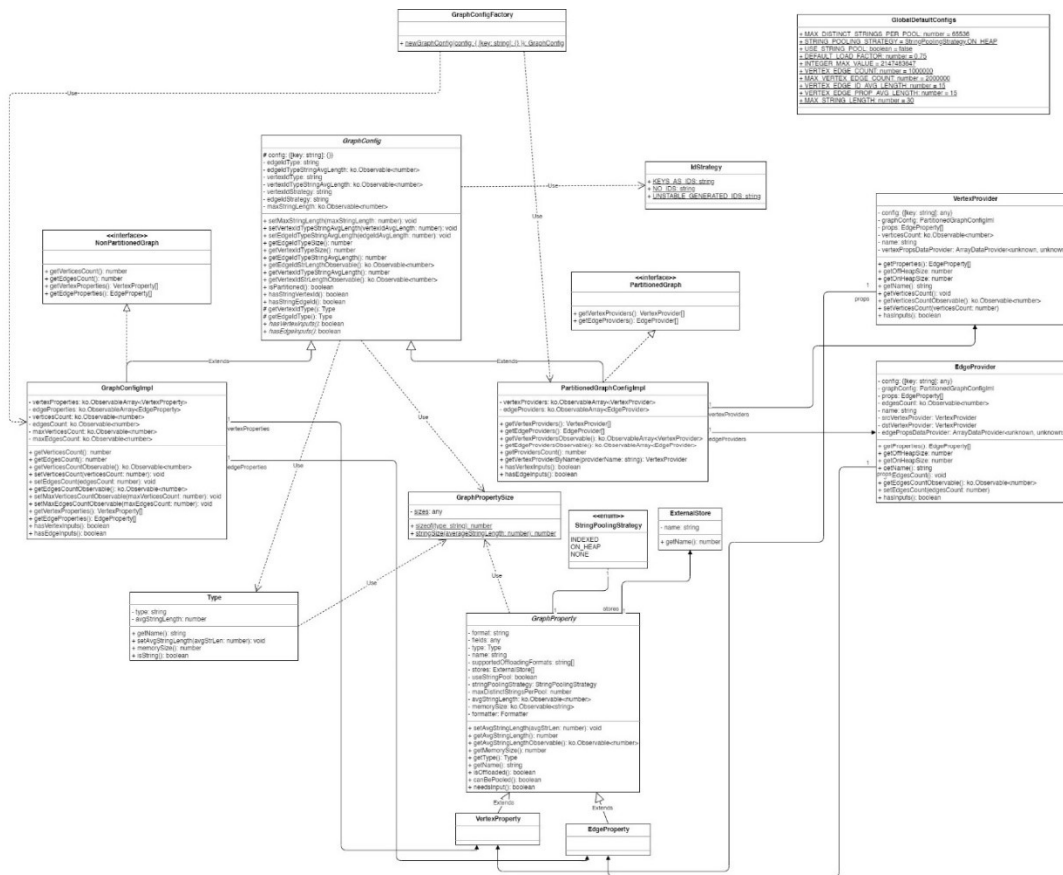


Figure 13 - Diagramme de cas d'utilisation

### IV.3 Diagrammes de classes

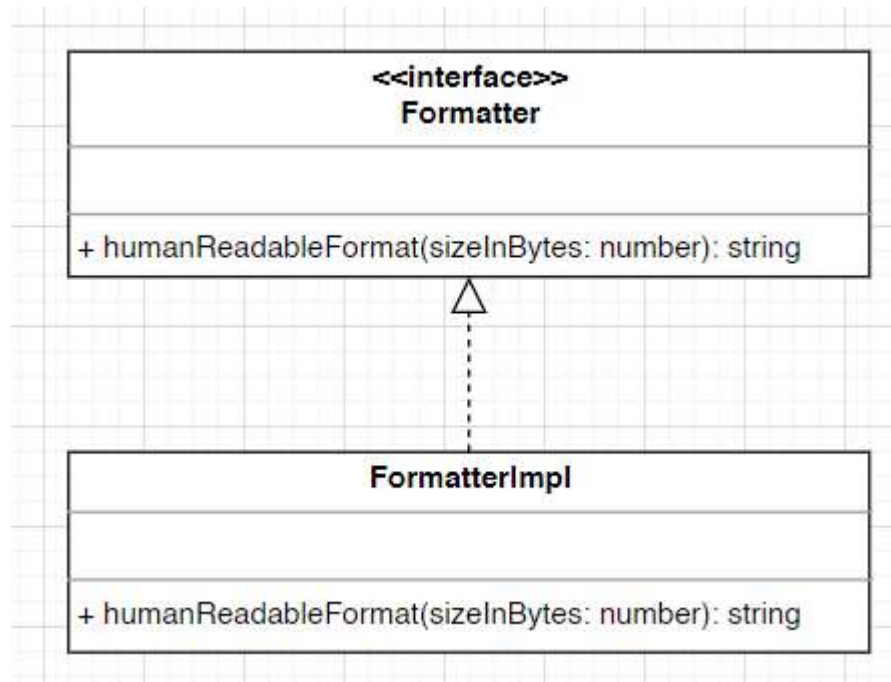
### IV.3.1 Module de configuration



**Figure 14 - Diagramme de classes - Module de configuration**

Ce module, ayant le diagramme de classes dans la figure 14, est responsable principalement du parsing de la configuration JSON.

### IV.3.2 Module de formatage des résultats



**Figure 15 - Diagramme de classes - Module de formatage**

Ce module, présenté dans la figure 15, est responsable du formatage des résultats.





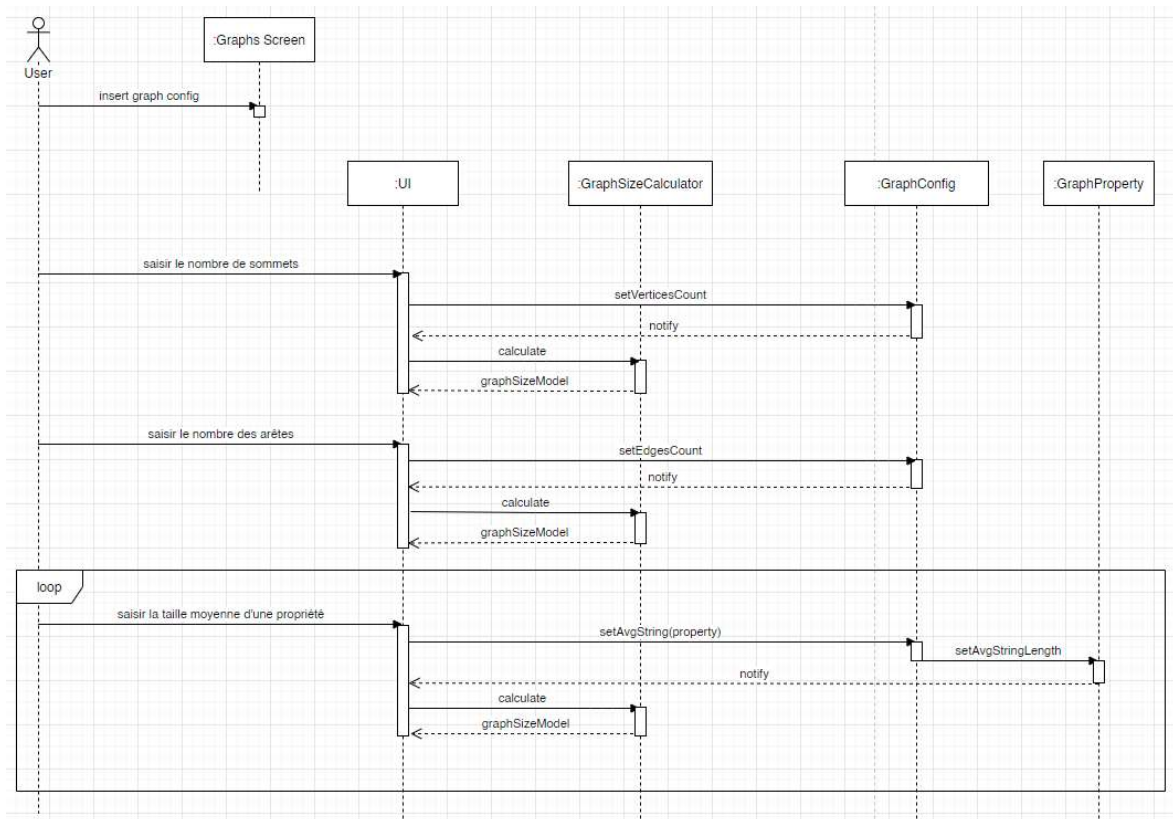


Figure 17 - Diagramme de séquence

## IV.5 Conclusion

Nous avons vu dans ce chapitre les trois phases principales de la conception orienté objet, à savoir : le diagramme de cas d'utilisation, le diagramme de classes, et le diagramme de séquence. Nous avons divisé notre composant en trois modules à savoir un module de configuration, un module de formatage des résultats, et un module principal qui représente l'interface d'utilisation du composant (à ne pas confondre avec l'interface graphique).

# Chapitre V. Environnement de développement

---

## V.1 Introduction

Dans ce chapitre nous allons voir l'environnement de travail à savoir, le matériel utilisé durant ce stage mais le aussi les logiciels et langages utilisés afin de réussir ce stage.

## V.2 Environnement de travail

Au niveau de cette partie, nous allons énumérer le matériel que nous avons utilisé pour mener au bout notre projet.

Dès le premier jour chez Oracle Labs, nous avons eu la chance de faire le « unboxing » du matériel nous-même, ce matériel consiste à :

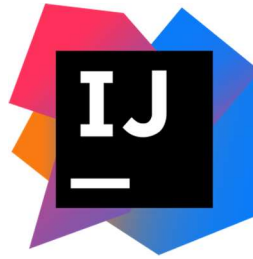
- Une unité centrale
- RAM : 16 Go
- CPU : Intel i7 6<sup>ème</sup> génération
- Disque dur : 1 To HDD
- Ecran Lenovo ThinkVision 23''
- Clavier mécanique (azerty) Lenovo
- Souris Lenovo
- Système d'exploitation : Oracle Linux

Tous ces composants nous en été livré à la maison lors du commencement du confinement, afin de continuer notre travail dans les meilleures conditions.

Nous avons aussi accès à une connexion haut débit de 12 Mb/s, et après le confinement nous avons commencé à utiliser notre propre internet, et déclarer toutes les factures à Oracle pour le remboursement.

## V.2.1 Les logiciels et les langages

### V.2.1.1 IntelliJ IDEA



**Figure 18 – Logo IntelliJ IDEA**

IntelliJ IDEA également appelé « IntelliJ », « IDEA », ayant le logo dans la figure 18, est un environnement de développement intégré (en anglais Integrated Development Environment - IDE) de technologie Java destiné au développement de logiciels informatiques. Il est développé par JetBrains (anciennement « IntelliJ ») et disponible en deux versions, l'une communautaire, open source, sous licence Apache 2 et l'autre propriétaire, protégée par une licence commerciale. Tous deux supportent les langages de programmation Java, Kotlin, Groovy et Scala.

### V.2.1.2 Apache JMeter



**Figure 19 - Logo Apache JMeter**

Apache JMeter, ayant le logo dans la figure 19, est un projet Apache qui peut être utilisé comme un outil de test de charge pour analyser et mesurer les performances d'une variété de services, en mettant l'accent sur les applications Web.

JMeter peut être utilisé comme outil de test unitaire pour les connexions à la base de données JDBC, FTP, LDAP, services Web, JMS, HTTP, connexions TCP génériques et OS-processus natifs. Nous pouvons également configurer JMeter en tant que moniteur, bien qu'il soit généralement utilisé comme solution de surveillance de base plutôt que comme surveillance avancée. Il peut également être utilisé pour certains tests fonctionnels. De plus, JMeter prend en charge l'intégration avec Selenium, ce qui lui permet d'exécuter des scripts d'automatisation parallèlement aux tests de performances ou de charge.

JMeter prend en charge le paramétrage des variables, les assertions (validation des réponses), les cookies par thread, les variables de configuration et une variété de rapports.

L'architecture JMeter est basée sur des plugins. La plupart de ses fonctionnalités "prêtes à l'emploi" sont implémentées avec des plugins.

### V.2.1.3 Git



**Figure 20 - Logo Git**

Git, ayant le logo dans la figure 20, est un logiciel de gestion de versions décentralisé. C'est un logiciel libre créé par Linus Torvalds, auteur du noyau Linux, et distribué selon les termes de la licence publique générale GNU version 2. En 2016, il s'agit du logiciel de gestion de versions le plus populaire qui est utilisé par plus de douze millions de personnes.

### V.2.1.4 HTML



**Figure 21 - logo HTML**

HTML5 (HyperText Markup Language 5), ayant le logo dans la figure 21, est la dernière révision majeure du HTML (format de données conçu pour représenter les pages web). Cette version a été finalisée le 28 octobre 2014. HTML5 spécifie deux syntaxes d'un modèle abstrait défini en termes de DOM : HTML5 et XHTML5. Le langage comprend également une couche application avec de nombreuses API, ainsi qu'un algorithme afin de pouvoir traiter les documents à la syntaxe non conforme. Le travail a été repris par le W3C en mars 2007 après avoir été lancé par le WHATWG.

### V.2.1.5 CSS



**Figure 22 - Logo CSS 3**

Les feuilles de style en cascade, ayant le logo dans la figure 22, généralement appelées CSS de l'anglais Cascading Style Sheets, forment un langage informatique qui décrit la présentation des documents HTML et XML. Les standards définissant CSS sont publiés par le World Wide Web Consortium (W3C). Introduit au milieu des années 1990, CSS devient couramment utilisé dans la conception de sites web et bien pris en charge par les navigateurs web dans les années 2000.

### V.2.1.6 SASS



**Figure 23 - Logo SASS**

Sass (abréviation de Syntactically awesome style sheets), ayant le logo dans la figure 23, est un langage de feuille de style initialement conçu par Hampton Catlin et développé par Natalie Weizenbaum.

Sass est un langage de script de préprocesseur qui est interprété ou compilé dans des feuilles de style en cascade (CSS). SassScript est le langage de script lui-même. Sass se compose de deux syntaxes. La syntaxe d'origine, appelée "la syntaxe en retrait", utilise une syntaxe similaire à Hamlet. Il utilise l'indentation pour séparer les blocs de code et les caractères de nouvelle ligne pour séparer les règles. La nouvelle syntaxe, "SCSS" (Sassy

CSS), utilise un formatage de bloc comme celui de CSS. Il utilise des accolades pour désigner les blocs de code et les points-virgules pour séparer les règles au sein d'un bloc. La syntaxe en retrait et les fichiers SCSS reçoivent traditionnellement les extensions .sass et .scss, respectivement.

#### V.2.1.7 TypeScript



**Figure 24 - Logo TypeScript**

TypeScript, ayant le logo dans la figure 24, est un langage de programmation libre et open source développée par Microsoft qui a pour but d'améliorer et de sécuriser la production de code JavaScript. C'est un sur-ensemble de JavaScript (c'est-à-dire que tout code JavaScript correct peut être utilisé avec TypeScript). Le code TypeScript est transcompilé en JavaScript, et peut ainsi être interprété par n'importe quel navigateur web ou moteur JavaScript. TypeScript a été cocréé par Anders Hejlsberg, principal inventeur de C# 2,3,4,5,6.

TypeScript permet un typage statique optionnel des variables et des fonctions, la création de classes et d'interfaces, l'import de modules, tout en conservant l'approche non-contraignante de JavaScript. Il supporte la spécification ECMAScript 6

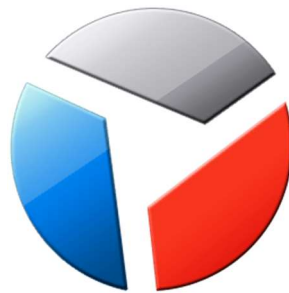
#### V.2.1.8 JQuery



**Figure 25 - Logo JQuery**

jQuery, ayant le logo dans la figure 25, est une bibliothèque JavaScript libre et multiplateforme créée pour faciliter l'écriture de scripts côté client dans le code HTML des pages web. La première version est lancée en janvier 2006 par John Resig. Depuis sa création en 2006 et notamment à cause de la complexification croissante des interfaces Web, jQuery a connu un large succès auprès des développeurs Web et son apprentissage est aujourd'hui un des fondamentaux de la formation aux technologies du Web. Il est à l'heure actuelle la librairie front-end la plus utilisée au monde (plus de la moitié des sites Internet en ligne intègrent jQuery).

#### V.2.1.9 Oracle JET



**Figure 26 - Logo Oracle JET**

Oracle JavaScript Extension Toolkit (Oracle JET), ayant le logo dans la figure 26, est une boîte à outils de développement JavaScript complète mais modulaire qui aide les développeurs à créer des interfaces utilisateur attrayantes. Basé sur les normes de l'industrie et les frameworks open source populaires, Oracle JET ajoute des fonctionnalités et des services avancés pour aider les développeurs à créer de meilleures applications plus rapidement.

##### **Avantages :**

- Cadre de développement JavaScript complet
- Exploite les technologies open source populaires
- Prise en charge intégrée du développement d'applications mobiles
- Gestion complète du style de vie pour le SPA basé sur un modèle
- Prise en charge intégrée de l'accessibilité
- Prise en charge de l'internationalisation (28 langues et 190+ paramètres régionaux)

- Ensemble riche de composants d'interface utilisateur
- Liaison bidirectionnelle avancée avec une couche de modèle commune
- Système de routage puissant prenant en charge la navigation d'une seule page
- Gestion intelligente des ressources
- Pour les développeurs JS intermédiaires et avancés
- Libre et open source

#### V.2.1.10 Jest



**Figure 27 - Logo Jest**

Jest, ayant le logo dans la figure 27, est un charmant framework de test JavaScript axé sur la simplicité. Il fonctionne avec des projets utilisant : Babel, TypeScript, Node, React, Angular, Vue et autres.

##### **Avantages :**

- Zéro config : Jest a pour objectif de fonctionner immédiatement, sans configuration, sur la plupart des projets JavaScript.
- Instantanés : Faites facilement des tests qui permettent de suivre les gros objets. Les instantanés vivent soit à côté de vos tests, soit intégrés en ligne.
- Isolé : Les tests sont parallélisés en les exécutant dans leurs propres processus pour maximiser les performances.
- API simple et intuitive : Jest a l'ensemble de la boîte à outils en un seul endroit. Bien documenté, bien entretenu.



### V.2.1.11 Jenkins



**Figure 28 - Logo Jenkins**

Jenkins, ayant le logo dans la figure 28, est un outil open source d'intégration continue, fork de l'outil Hudson après les différends entre son auteur, Kohsuke Kawaguchi, et Oracle. Écrit en Java, Jenkins fonctionne dans un conteneur de servlets tel qu'Apache Tomcat, ou en mode autonome avec son propre serveur Web embarqué. Il s'interface avec des systèmes de gestion de versions tels que CVS, Git et Subversion, et exécute des projets basés sur Apache Ant et Apache Maven aussi bien que des scripts arbitraires en shell Unix ou batch Windows.

### V.2.1.12 SonarQube



**Figure 29 - Logo SonarQube**

SonarQube (précédemment Sonar), ayant le logo dans la figure 29, est un logiciel libre permettant de mesurer la qualité du code source en continu.

#### **Fonctionnalités**

- Support de plus de vingt-cinq langages (Java, C, C++, Objective-C, C#, PHP, Flex, Groovy, JavaScript, Python, PL/SQL, COBOL...), dont certains sont sous licence commerciale.
- Reporting sur :

- Identification des duplications de code
- Mesure du niveau de documentation
- Respect des règles de programmation
- Détection des bugs potentiels
- Evaluation de la couverture de code par les tests unitaires
- Analyse de la répartition de la complexité
- Analyse du design et de l'architecture d'une application
- Évolution dans le temps et vues différentielles
- Analyses entièrement automatisées : intégration avec Maven, Ant, Gradle et serveurs d'intégration continue (Atlassian Bamboo, Jenkins, Hudson...).
- Intégration avec l'environnement de développement Eclipse
- Intégration avec des outils externes : Jira, Mantis, LDAP, Fortify Software (en)...
- Extensible par des plugins. Cela signifie qu'il est possible d'étendre ce cœur afin d'augmenter les fonctionnalités (ajout d'un nouveau langage, calcul d'une nouvelle métrique, ajout de règles de programmation...). Le portail des plugins SonarQube permet d'accéder à la liste des extensions existantes.

### V.2.1.13 La stack Atlassian

#### V.2.1.13.1 *Jira*



Figure 30 - Logo Jira

Jira, ayant le logo dans la figure 30, est un système de suivi de bugs, un système de gestion des incidents, et un système de gestion de projets développé par Atlassian.

#### V.2.1.13.2 *Confluence*



Figure 31 - Logo Confluence

Confluence, ayant le logo dans la figure 31, est un logiciel de wiki, utilisé comme logiciel de travail collaboratif. Confluence est un logiciel commercial, développé par Atlassian.

#### **V.2.1.13.3      *Bitbucket***



**Figure 32 - Logo Bitbucket**

Bitbucket, ayant le logo dans la figure 32, est un service web d'hébergement et de gestion de développement logiciel utilisant le logiciel de gestion de versions Git (et par le passé également le logiciel Mercurial). Il s'agit d'un service freemium dont la version gratuite permet déjà de créer jusqu'à un nombre illimité de dépôts privés, accessibles par cinq utilisateurs au maximum.

## **V.3 Conclusion**

Nous constatons qu'Oracle donne une très grande importance au confort de son personnel, en leur garantissant les meilleures conditions de travail et en utilisant les dernières tendances des technologies dans ce projet. Ceci se voit concrètement lors du confinement ou l'équipe Oracle nous a livré à domicile notre bureau, afin d'assurer la continuité du travail dans les meilleures conditions.

---

# Chapitre VI. Réalisation

---

## VI.1 Introduction

Dans ce chapitre nous allons présenter le résultat que nous avons pu attendre jusqu’au moment de la rédaction de ce rapport, une réalisation divisée en 2 parties, premièrement le nouveau composant de Data Studio appelé « Graph Size Calculator », et deuxièmement les résultats du benchmark fait sur Data Studio en utilisant des graphes PGX.

## VI.2 Le « Graph Size Calculator »

### VI.2.1 La maquette de l’interface utilisateur

Dans la phase du ticket « Phase 20 – Spécification » (voir le chapitre II), nous étions menés à créer la maquette de l’interface utilisateur pour cet outil, pour cette tâche nous avons choisi d’utiliser l’outil Figma<sup>2</sup>. Les résultats sont dans les annexes 6 et 7.

### VI.2.2 Calcul du taux d’erreur

Pour le calcul du taux d’erreur, nous avons choisi de charger plusieurs graphes PGX et capturer leur taille réelle, et comparer les résultats avec les estimations faites en utilisant le « Graph Size Calculator ». Pour cela, nous avons utilisé un cluster<sup>3</sup> d’Oracle, et nous avons créé un script Java (code source dans l’annexe 2) qui prend en paramètre le chemin d’un graphe (Captures d’écran dans l’annexe 3, 4 et 5), et commence à charger le graphe dans le serveur PGX, lorsqu’il est chargé, il exécute des requêtes PGQL pour calculer la longueur moyenne des propriétés, et en fin il capture la taille mémoire utilisé grâce à l’admin API de PGX (voir les résultats dans la figure 33).

---

<sup>2</sup> Figma est une application de conception d’interface qui s’exécute dans le navigateur.

<sup>3</sup> Un cluster est une grappe de serveurs sur un réseau, appelé ferme ou grille de calcul.

Config	Vertices	Edges	Actual String length	Experimental size	String length used	Estimated size
Social cloud	2880870	12658205	11.088	<ul style="list-style-type: none"> <li>Total : 926 MB</li> <li>On-Heap : 210 MB</li> <li>Off-Heap : 716 MB</li> </ul>	11	<ul style="list-style-type: none"> <li>Total : 916.21 MB</li> <li>On-Heap : 175.83 MB</li> <li>Off-Heap : 740.38 MB</li> </ul>
Customers Transactions	56856811	94453354	14.872	<ul style="list-style-type: none"> <li>Total : 11.14 GB</li> <li>On-Heap : 3.93 GB</li> <li>Off-Heap : 7.21 GB</li> </ul>	15	<ul style="list-style-type: none"> <li>Total : 11.17 GB</li> <li>On-Heap : 4.21 GB</li> <li>Off-Heap : 6.96 GB</li> </ul>
Page links	8637721	165049964	10.862	<ul style="list-style-type: none"> <li>Total : 5.93 GB</li> <li>On-Heap : 2.81 GB</li> <li>Off-Heap : 3.13 GB</li> </ul>	11	<ul style="list-style-type: none"> <li>Total : 6.26 GB</li> <li>On-Heap : 2.20 GB</li> <li>Off-Heap : 4.06 GB</li> </ul>
Frappe RDBMS	5018583	63819065	15.418	<ul style="list-style-type: none"> <li>Total : 4.57 GB</li> <li>On-Heap : 989.91 MB</li> <li>Off-Heap : 3.60 GB</li> </ul>	15	<ul style="list-style-type: none"> <li>Total : 5.28 GB</li> <li>On-Heap : 1.23 GB</li> <li>Off-Heap : 4.05 GB</li> </ul>

Figure 33 - Une partie des résultats de calcul d'erreur

Les résultats finaux sont :

- Total d'erreur : 1%
- Erreur on-heap : 5%
- Erreur off-heap : 0.1%

## VI.2.3 Captures d'écran

Le graphe à deux modes : le mode normal où nous ne nous affichons pas les propriétés du graphes (voir la figure 34).

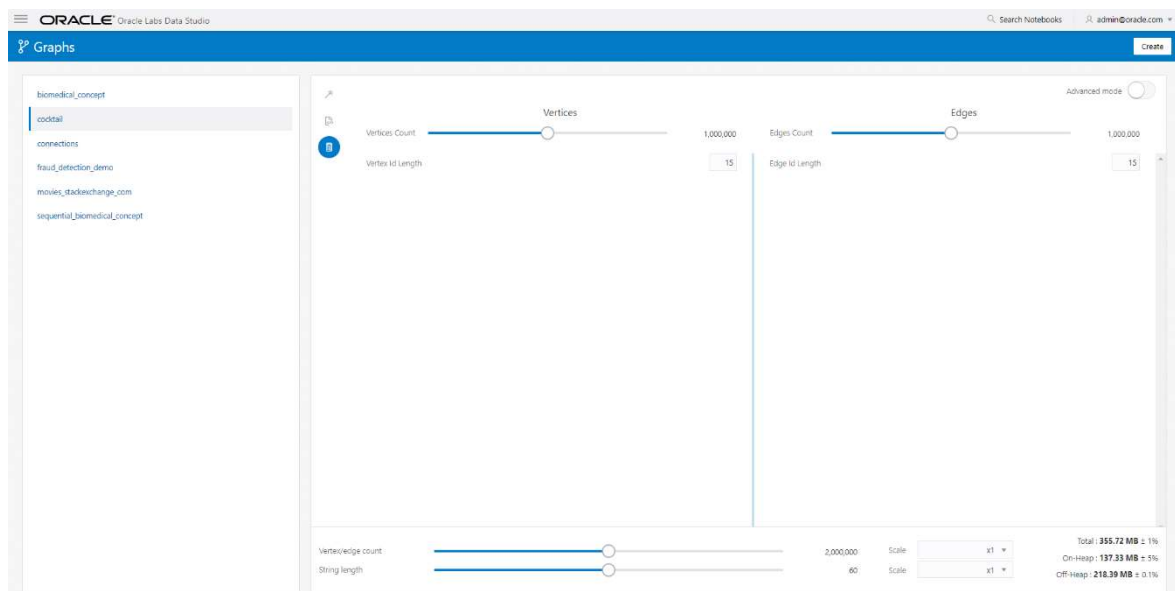


Figure 34 - Le "Graph Size Calculator" (mode normal)

L'autre mode est le mode avancé, dans ce mode, l'utilisateur a la possibilité de mettre des valeurs personnalisées pour les longueurs moyennes de chaînes de caractères des propriétés (voir la figure 35).

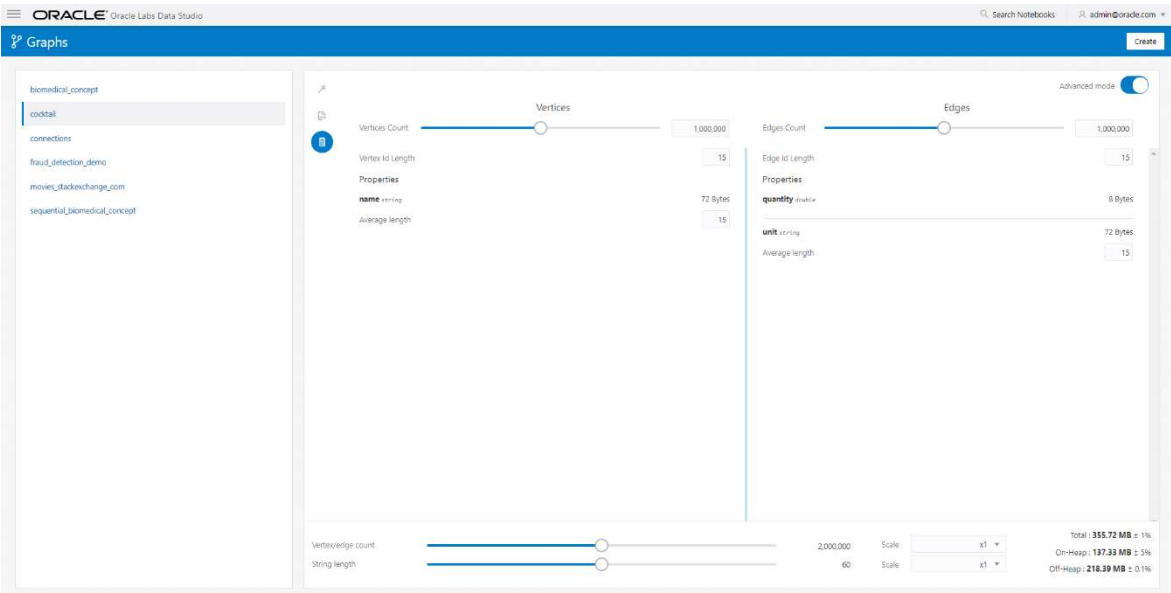


Figure 35 - Le "Graph Size Calculator" (mode avancé)

VI.3 Le benchmark de Data Studio avec des graphes PGX

Nous avons créé deux plans de tests, le premier va s’exécuter une seule fois, son rôle principal c’est de charger un très grand graphe de transactions bancaires et créer des corrélations entre les nœuds. Le deuxième va faire des investigations afin de découvrir toutes sorte de fraudes dans les données. Ce dernier plan va s’exécuter avec 100 utilisateurs simultanées pour une durée de 30 minutes.

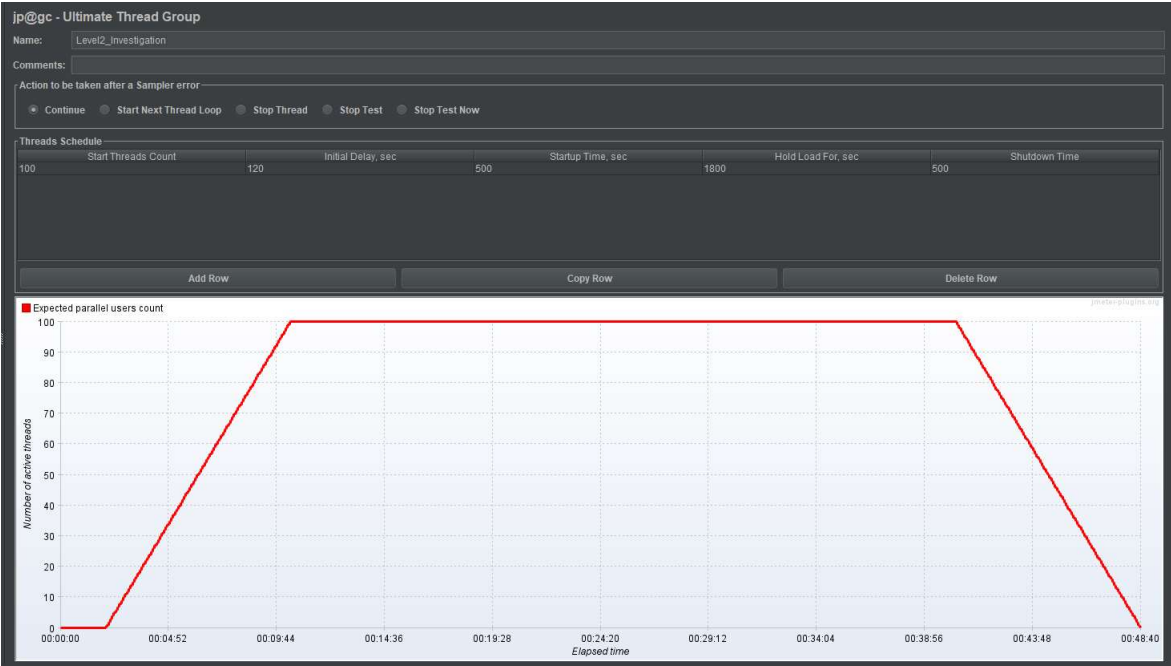
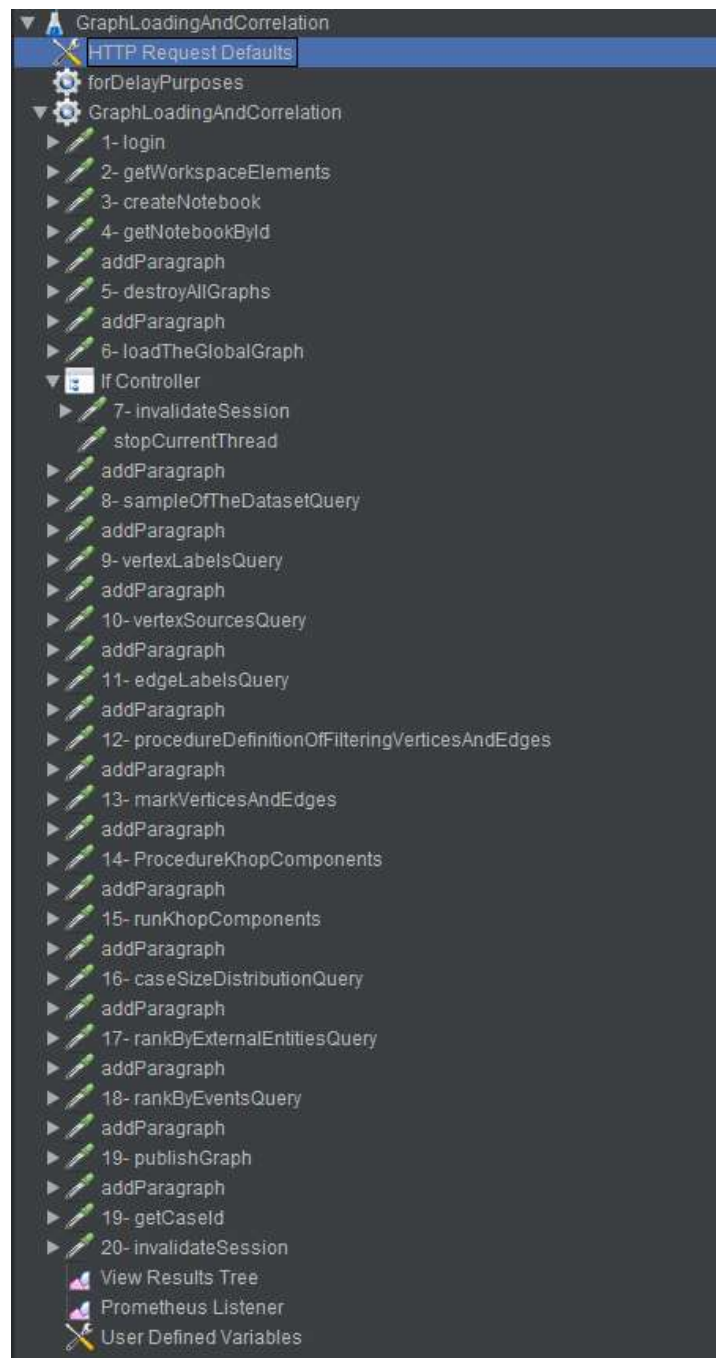


Figure 36 - plan de test 2 - évolution des threads d'exécution

Dans la figure 36, nous constatons que ce thread groupe est paramétrable à l'aide des arguments suivants :

- « Start Threads Count » : il représente le nombre de thread qui vont être créés.
- « Initial Delay, sec » : il représente un retard initial en secondes, il peut servir dans des cas où il faut attendre à ce que le système démarre.
- « Startup Time, sec » : il représente la durée, en secondes, que le thread groupe va prendre pour arriver au nombre planifié dans le « Start Threads Count ». Par exemple si nous voulons démarrer un thread toutes les 5 secondes et nous voulons atteindre 100 threads, cet argument doit avoir une valeur de 500.
- « Hold Load For, sec » : il représente la durée en secondes pour laquelle JMeter va essayer de garder le même nombre de threads. Par exemple si nous mettons 1800 (30 minutes), JMeter va essayer de garder les 100 threads en travail pendant cette durée, c'est-à-dire, si un thread termine, un autre va commencer, dans le but de garder la même charge.
- « Shutdown Time » : La durée en secondes, que JMeter va consacrer pour mettre fin aux threads. Si cette valeur est de 500, et nous avons 100 threads, chaque 5 secondes JMeter va mettre fin à un thread.

### VI.3.1 Le premier plan de test



**Figure 37 - plan de test de chargement du graphe**

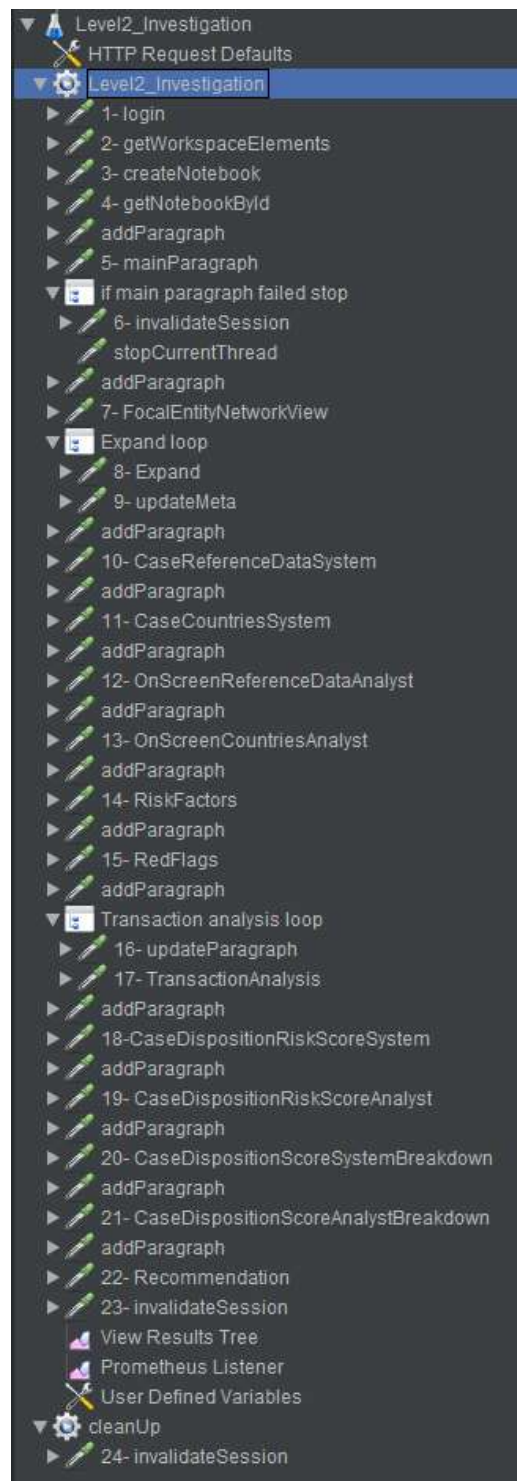
Comme présenter dans la figure 37, le premier plan de test suit les étapes suivantes

- Login : Fait appel à la route du login avec les identifiants, et il extrait le « authToken » de la réponse, afin de l'utiliser dans les requêtes suivantes.



- 
- Récupérer les éléments de l'espace de travail : il appelle la route convenable afin d'imiter le chargement de éléments de l'espace de travail qui se fait après l'authentification.
  - Créer un notebook : il fait appel à la route convenable pour créer un notebook qui sera rempli après avec des paragraphes.
  - Récupérer le notebook par son id : cette requête ne change rien dans le notebook, mais son objectif c'est d'imiter le chargement d'un notebook après ce qu'il soit créé.
  - Créer un paragraphe et exécuter son contenu : le paragraphe crée dans cette requête fait vider la session de tous les graphes créés préalablement, afin d'éviter tout problème de sorte : « Un graphe avec le nom X existe déjà ».
  - Créer un paragraphe et exécuter son contenu : Le paragraphe créé dans cette requête est le paragraphe qui charge le graphe sujet de notre étude.
  - Contrôleur Si : un contrôleur est un composant de JMeter, il offre plusieurs contrôleurs parmi eux il y a le contrôleur Si, en utilise ce contrôleur ici pour s'assurer que la requête précédente est exécutée avec succès, sinon nous arrêtons le thread.
  - Les requêtes restantes sont des requêtes simples qui essaie de faire des corrélations entre les nœuds du graphe et afficher des résultats à l'aide des requêtes PGQL.

### VI.3.2 Le deuxième plan de test



**Figure 38 - Plan de test 2 - Investigation des fraudes**

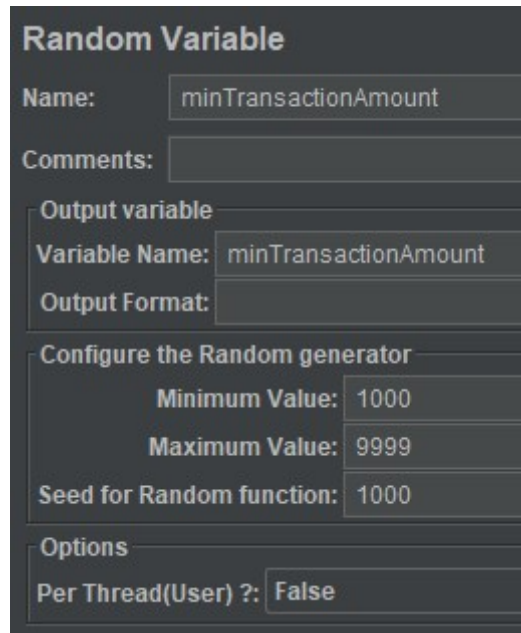
Comme présenter dans la figure 38, le deuxième plan de test suit les étapes suivantes :

- Commence avec les requêtes nécessaires pour l'initialisation (ils sont expliqués dans la partie précédente) : l'authentification, le chargement des éléments de l'espace de travail, création d'un notebook et la récupération de ce notebook.

- 
- Création d'un paragraphe qui constitue le cœur de ce plan : ce paragraphe crée le code qui fait l'investigation dans le graphe, le paragraphe est de type « pgx-java », c'est un paragraphe Java mais qui utilise l'API PGX, il est destiné à être exécuter dans le serveur PGX. Ce paragraphe prend en paramètre un « case\_id » ou numéro de cas, nous utilisons pour cela un autre composant de JMeter qui s'appelle « CSV Dataset Config », qui nous permet d'utiliser un fichier CSV des id des cas, et pour chaque thread créer, JMeter prend un id de cas, et le passe à la requête, cela donne que chaque notebook créé est responsable d'un cas.
  - Contrôleur si : son but est comme dans le plan précédent : être sûr que le paragraphe précédent s'est exécuté avec succès.
  - Un paragraphe qui visualise le résultat de l'investigation sous forme de graphe.
  - Loop contrôleur : Celui-ci est un autre composant de JMeter, il sert, comme son nom l'indique, à créer une boucle de requête. Nous l'utilisons pour boucler (10 fois) sur deux requêtes qui imite un utilisateur qui fait un étendage<sup>4</sup> du graphe plusieurs fois.
  - Nous utilisons une autre boucle dans ce plan : Une boucle qui imite une suite de mise à jour, l'exécution sur un paragraphe. Ce paragraphe affiche les résultats d'un des analyses sur des transactions bancaires, il prend deux paramètres, le montant minimum, et le montant maximum de ces transactions.

---

<sup>4</sup> Un étendage est une fonctionnalité dans les visualisations des graphes dans Data Studio, où l'utilisateur peut étendre un nœud pour voir les propriétés de ce nœud.



**Random Variable**

Name: minTransactionAmount

Comments:

Output variable

Variable Name: minTransactionAmount

Output Format:

Configure the Random generator

Minimum Value: 1000

Maximum Value: 9999

Seed for Random function: 1000

Options

Per Thread(User) ?: False

**Figure 39 - JMeter - Random Variable**

Pour générer ces montants, on utilise un autre composant de JMeter qui s'appelle : « Random Variable » (voir figure 39), comme son nom l'indique, il nous permet de générer des variables numériques aléatoires qui varient dans une range, ces variables sont utilisées au sein du paragraphe de mise à jour afin d'offrir une sorte de réalité au plan de test.

Dans cette boucle nous avons deux requêtes, une met à jour le paragraphe, et l'autre exécute le paragraphe.

- À la fin du plan de test, nous avons un thread groupe qui s'appelle « TearDown Thread Group », les requêtes de ce groupe sont exécutées en dernier, à la fin du premier thread groupe, au sein de groupe nous avons une seule requête qui appelle une route qui va invalider la session d'un thread donné (nous avons ajouté cette requête après avoir constaté que les pods<sup>5</sup> ne sont pas détruits après la fin d'une session (un thread)). Donc son rôle c'est de détruire la session, et par conséquent détruire le pod.

---

<sup>5</sup> Un pod représente la plus petite unité déployable et exécutable au sein du cluster. C'est la brique de base de Kubernetes qui encapsule un ou plusieurs conteneurs.

Ce groupe s'exécute autant de fois que le thread groupe principale, afin de s'assurer que toutes les sessions sont détruites. Chaque fois, nous passons comme paramètre le ID du notebook (session), qui a été capturer lors de la création de ce dernier.

### VI.3.3 Résultats du benchamark

Pour afficher les résultats nous avons utilisé l'outil Grafana<sup>6</sup>, les résultats de cette partie représente les résultats d'exécution de plan de test numéro 2, puisqu'il s'exécute plusieurs fois.

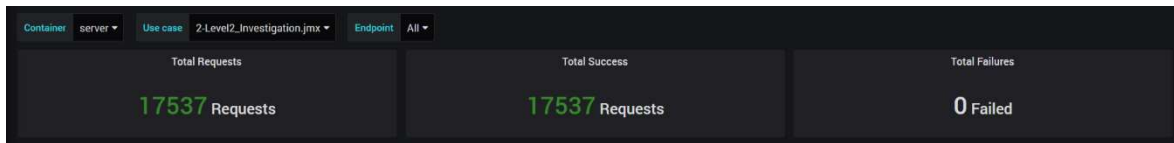


Figure 40 - Les résultats généraux de l'exécution

Le throughput en anglais, ou le débit en français, représente le nombre de requête passé par seconde. En voici les résultats dans la figure 41.

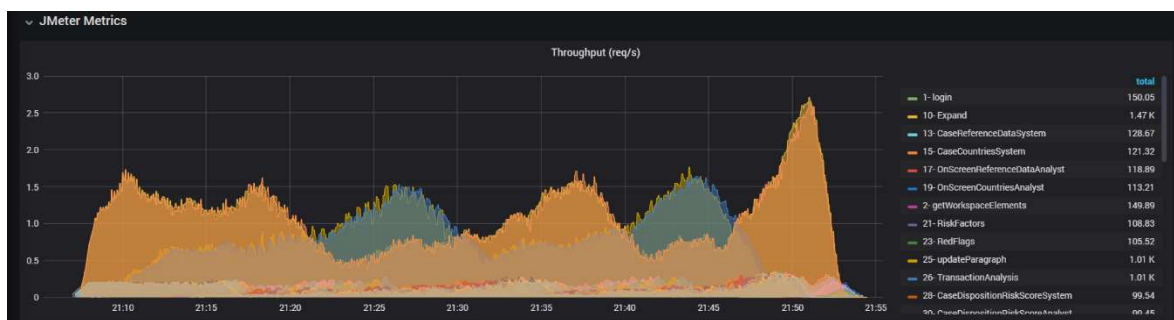
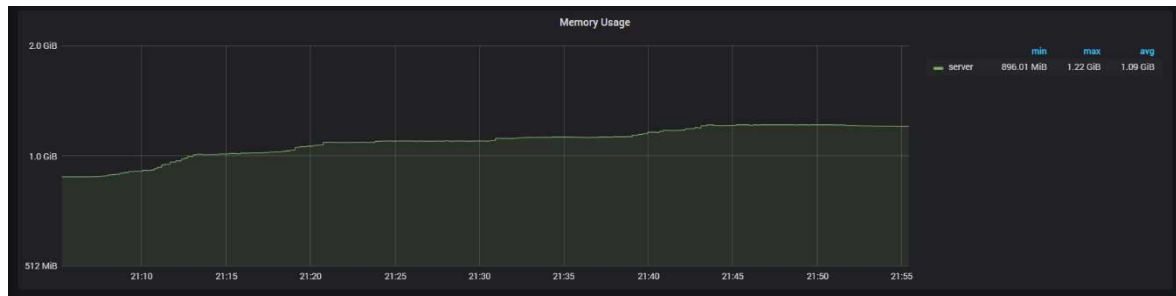


Figure 41 – Benchmark - Le throughput (débit req/s)

La consommation mémoire nous donne une idée sur la l'évolution de l'allocation mémoire du serveur, ainsi voir si, à moment donnée il a une évolution exponentielle, ou linéaire (voir la figure 42).

<sup>6</sup> Grafana est un logiciel libre qui permet la visualisation de données. Il permet de réaliser des tableaux de bord et des graphiques depuis plusieurs sources dont des bases de données temporelles comme Graphite, InfluxDB et OpenTSDB.



**Figure 42 - Benchmark - La consommation mémoire**

## VI.4 Conclusion

Dans ce chapitre nous avons vu le progrès fait dans chacune des parties de notre PFE, il faut mentionner que le stage est toujours en cours, donc ses résultats vont surement être améliorer dans la durée restante de notre stage.

## Conclusion générale

Ce présent rapport décrit le travail réalisé lors de notre stage de fin d'étude effectué au sein de la société Oracle. Ce stage concerne la conception et la réalisation d'une composant web qui sert à estimer la taille mémoire d'un graphe PGX, ainsi que le benchmark de Data Studio avec des graphes PGX.

L'objectif pour le composant web est d'éviter d'utiliser un grand fichier Excel, et avoir plutôt un outil simple à utiliser et facilement intégrable dans n'importe quelle application web. L'objectif pour le benchmark c'est de créer un cas d'utilisation de Data Studio avec des graphes PGX, afin de savoir l'impact de l'exécution de ces graphes sur la performance de Data Studio, ainsi que savoir le nombre maximum d'utilisateurs simultanées que Data Studio peut supporter.

Pour mener à bien ce stage, nous avons commencé par une analyse des besoins, la rédaction d'un cahier de charge. Ensuite nous avons entamé la phase de l'analyse et de conception à travers l'utilisation du langage UML. Lors de la phase de réalisation, nous avons utilisé le Framework JavaScript Oracle JET vu les avantages qu'il offre ainsi que d'autres technologies de pointe.

Pour conclure, ce stage a été très enrichissant pour nous car il nous a permis de découvrir l'environnement de travail dans une société multinationale, En effet, nous avons pu bénéficier de nouvelles acquisitions qui ont enrichi nos connaissances et compétences. Sur le plan professionnel, nous avons pu travailler dans une équipe qualifiée, multinational et conviviale, sur un projet réel et réglementé, des réunions d'avancements et bien d'autres tâches dont nous étions chargés de faire.

## Bibliographie

Oracle. Oracle PGX Documentation, [en ligne]. Disponible sur : [https://docs.oracle.com/cd/E56133\\_01/latest/index.html](https://docs.oracle.com/cd/E56133_01/latest/index.html)

Oracle PGQL. Property Graph Query Language, [en ligne]. Disponible sur : <https://pgql-lang.org>

Oracle. Oracle JavaScript Extension Toolkit (JET) API Reference, [en ligne]. Disponible sur : <https://docs.oracle.com/en/middleware/developer-tools/jet/8.2/reference-api/index.html>

Oracle. Oracle JET Cookbook, [en ligne]. Disponible sur : <https://www.oracle.com/webfolder/technetwork/jet/jetCookbook.html>

Knockout. KnockoutJs, [en ligne]. Disponible sur : <https://knockoutjs.com>

Figma. Figma : the collaboration design tool, [en ligne]. Disponible sur : <https://www.figma.com>

Wikipédia, Wikipédia : l'encyclopédie libre, [en ligne]. Disponible sur : <https://fr.wikipedia.org>

SchedMD. Slurm: workload manager, [en ligne]. Disponible sur : <https://slurm.schedmd.com>

Memory usage of Java Strings and string-related objects. In : JavaMex, [en ligne]. Disponible sur : [https://www.javamex.com/tutorials/memory/string\\_memory\\_usage.shtml](https://www.javamex.com/tutorials/memory/string_memory_usage.shtml)

Apache. Apache JMeter, [en ligne]. Disponible sur : <https://jmeter.apache.org>

Linkedin Learning. JMeter: Performance and load testing, [en ligne]. Disponible sur : <https://www.linkedin.com/learning/jmeter-performance-and-load-testing>



# Annexes

## Annexe 1 Journal du stage

Nous allons suivre une structure par semaine, où chaque semaine, nous allons mentionner un résumé divisé en 4 parties :

- Les réalisations (Accomplishments) : pour le travail fait durant la semaine
- Les priorités (Priorities) : pour les tâches qui ont une priorité pour la semaine d'après
- Les incidents (Incidents) : pour toutes incidents qui s'est passé durant la semaine, exemple : Malade le mardi.
- Préoccupations / bloqueurs (Concerns/Blockers) : Pour tout ce qui bloque pour attendre une réalisation.



Les captures d'écran sont prises depuis notre page Confluence de rapport hebdomadaire (weekly report).

### Semaine du 07 février 2020

#### Accomplishments:

- (done) Attend the Oracle Labs introduction meeting
- (done) Environment Setup
- (done) Attend Data Studio Full Team Meeting
- (done) Finish the compliance courses
- (done) Attend PFE Interns Topic Intro Meeting
- (done) Translate PGX Notebook into PGX-Java Notebook (<http://datastudio.oraclecorp.com/?root=notebooks~ebook=dskrMWikVvY>)

#### Priorities:

- (in progress) Add translated notebook as a builtin notebook  **PN-5045** - PGX-Java-fy notebook Deep Dive **CLOSED**
- (in progress) Understanding how graph size calculator works  **GM-18671** - Graph Size Calculator **60 - IN REVIEW (TEAM)**

#### Incidents:

- 

#### Concerns/Blockers:

-

## Semaine du 14 février 2020

### Accomplishments:

- (code review) ✓ **PN-5015** - PGX-Java-fy notebook Deep Dive **CLOSED**
- (done) Run PGX locally. Needed for **GM-20759** - Graph Size Calculator Simple Script **CLOSED**
- (done) Attend Benchmark TF Meeting
- (done) Setup Data Studio locally for ✓ **PN-5015** - PGX-Java-fy notebook Deep Dive **CLOSED**
- (done) Setup JMeter locally for ✓ **PN-4098** - PGX use case test **IN PROGRESS**

### Priorities:

- (in progress) ✓ **PN-4098** - PGX use case test **IN PROGRESS**
- (in progress) **GM-20759** - Graph Size Calculator Simple Script **CLOSED**

### Incidents:

- 

### Concerns/Blockers:

- 

## Semaine du 21 février 2020

### Accomplishments

- (done) ✓ **PN-5015** - PGX-Java-fy notebook Deep Dive **CLOSED**
- (done) Organize meeting with **@Iraklis Psaroudakis** and **@Damien Hilloulin**
  - Subject: In depth understanding of the graph size calculation
- (in progress) **GM-20759** - Graph Size Calculator Simple Script **CLOSED**
- Misc:
  - Created: **GM-20856** - Broken link: Download PGX Server **OPEN**
  - Created: **PN-6212** - Test & Merge failed due to agent restarting **CANNOT REPRODUCE**

### Priorities

1. (in progress) **GM-20759** - Graph Size Calculator Simple Script **CLOSED**
  - a. Repo: [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif\\_oracle.com/repos/graph-size-calculator](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif_oracle.com/repos/graph-size-calculator)
2. (in progress) **GM-18671** - Graph Size Calculator **60 - IN REVIEW (TEAM)**

### Incidents






- 

### Concerns/Blockers




-

## Semaine du 28 février 2020

### Accomplishments

- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  GM-20759 - Graph Size Calculator Simple Script CLOSED
  - repo: [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif\\_oracle.com/repos/graph-size-calculator](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif_oracle.com/repos/graph-size-calculator)
- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (done)  GM-20973 - Typo in "Memory Consumption" page in Docs CLOSED
- (done)  PN-6274 - Add "fcc\_graph\_test" as builtin graph CLOSED

### Priorities

1. (in progress)  PN-4098 - PGX use case test IN PROGRESS
2. (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
3. (in progress)  GM-20759 - Graph Size Calculator Simple Script CLOSED

### Incidents





- 

### Concerns/Blockers




- 

## Semaine du 06 mars 2020

### Accomplishments

- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  GM-20759 - Graph Size Calculator Simple Script CLOSED
  - repo: [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif\\_oracle.com/repos/graph-size-calculator](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif_oracle.com/repos/graph-size-calculator)
- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (in progress) Resolving comments in proposal page
  - <https://ol-confluence.us.oracle.com/display/OLDS/GM-18671+Graph+Size+Calculator>
- (created) (TF task)  PN-6374 - Load Graph Configuration Timed out WON'T FIX
- (done) Attend Task force meeting
- (done) Attend DS Full Team Meeting

### Priorities

- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  GM-20759 - Graph Size Calculator Simple Script CLOSED
- (in progress) Resolving comments in proposal page
  - <https://ol-confluence.us.oracle.com/display/OLDS/GM-18671+Graph+Size+Calculator>

### Incidents



- 

### Concerns/Blockers




-

## Semaine du 13 mars 2020

### Accomplishments

- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (in progress) Resolving comments in proposal page
  - <https://ol-confluence.us.oracle.com/display/OLDS/GM-18671+Graph+Size+Calculator>
- (done) Attend "TF: Brace Yourselves users are coming" meeting
- (done) Organize meeting with reviewers of my proposal
  - proposal: <https://ol-confluence.us.oracle.com/display/OLDS/GM-18671+Graph+Size+Calculator>

### Priorities

- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  GM-20759 - Graph Size Calculator Simple Script CLOSED
- (in progress) Resolving comments in proposal page
  - <https://ol-confluence.us.oracle.com/display/OLDS/GM-18671+Graph+Size+Calculator>

### Incidents



- 

### Concerns/Blockers




- 

## Semaine du 20 mars 2020

### Accomplishments

- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (in progress) Resolving comments in proposal page
  - <https://ol-confluence.us.oracle.com/display/OLDS/GM-18671+Graph+Size+Calculator>
- (done) Attend "TF: Brace Yourselves users are coming" meeting
- (done) Organize meeting with reviewers of my proposal
  - proposal: <https://ol-confluence.us.oracle.com/display/OLDS/GM-18671+Graph+Size+Calculator>
- (done) Attend "Interns Progress Review" meeting

### Priorities

- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  GM-20759 - Graph Size Calculator Simple Script CLOSED

### Incidents

- 


### Concerns/Blockers

-






## Semaine du 27 mars 2020

### Accomplishments

- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress) Resolving comments in proposal page
  - <https://ol-confluence.us.oracle.com/display/OLDS/GM-18671+Graph+Size+Calculator>
- (done) Attend "TF: Brace Yourselves users are coming" meeting
- (done) Attend "Feature Sign Off" meeting

### Priorities

- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  GM-20759 - Graph Size Calculator Simple Script CLOSED

### Incidents




- 

### Concerns/Blockers




- 

## Semaine du 03 avril 2020

### Accomplishments

- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
  - [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhliif\\_oracle.com/repos/graph-size-calculator-ts](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhliif_oracle.com/repos/graph-size-calculator-ts)
  - [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhliif\\_oracle.com/repos/graph-size-calculator-example](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhliif_oracle.com/repos/graph-size-calculator-example)
  - [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhliif\\_oracle.com/repos/graph-size-calculator-ojet](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhliif_oracle.com/repos/graph-size-calculator-ojet)
- (in progress)  PN-4098 - PGX use case test IN PROGRESS
  - Investigate graph loading timeout error
- (done) Attend "TF: Brace Yourselves users are coming" meeting
- (done) Update UI Mocks for sign off (  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM) )
- (done) Attend "feature sign off" meeting
- (done) Join "[oraclejetcommunity.slack.com](https://oraclejetcommunity.slack.com)" to ask questions
- (done) Learning Oracle JET

### Priorities

- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  GM-20759 - Graph Size Calculator Simple Script CLOSED

### Incidents

- 

### Concerns/Blockers





-

## Semaine du 10 avril 2020

### Accomplishments

- (done) UI of the Graph Size Calculator Component
- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
  - [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif\\_oracle.com/repos/graph-size-calculator-ojet](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif_oracle.com/repos/graph-size-calculator-ojet)
- (done) Attend Data Studio Full Team meeting

### Priorities

- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  GM-20759 - Graph Size Calculator Simple Script CLOSED
- (in progress)  GM-20761 - Graph Size Calculator Integration With Data Studio IN PROGRESS

### Incidents

- 





### Concerns/Blockers

## Semaine du 17 avril 2020

### Accomplishments

- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
  - [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif\\_oracle.com/repos/graph-size-calculator-ojet](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif_oracle.com/repos/graph-size-calculator-ojet)
- (in progress)  GM-20761 - Graph Size Calculator Integration With Data Studio IN PROGRESS

### Priorities

- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  GM-20759 - Graph Size Calculator Simple Script CLOSED
- (in progress)  GM-20761 - Graph Size Calculator Integration With Data Studio IN PROGRESS

### Incidents





- 

### Concerns/Blockers





- Setup DS in my personal computer

## Semaine du 24 avril 2020

### Accomplishments

- (done) Open PR For  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (done) Attend "TF Brace yourselves users are coming" meeting  PN-4098 - PGX use case test IN PROGRESS
- (done) Find case\_id that gives visual representation of the investigation  PN-4098 - PGX use case test IN PROGRESS
- (done) Integrate Graph Size Calculator Component into Data Studio
- (done) Organize meeting with my mentor to validate feature  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)

### Priorities

- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  GM-29759 - Graph Size Calculator Simple Script CLOSED
- (in progress)  GM-20761 - Graph Size Calculator Integration With Data Studio IN PROGRESS

### Incidents

- 




### Concerns/Blockers

## Semaine du 01 mai 2020

### Accomplishments

- (done) Address comments in PR
  - <https://ol-bitbucket.us.oracle.com/projects/OLDS/repos/datastudio/pull-requests/2929/overview>

### Priorities

- (in progress) Calculate the error rate for the graph size calculator
- (in progress)  PN-4098 - PGX use case test IN PROGRESS
- (in progress)  GM-18671 - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  GM-20761 - Graph Size Calculator Integration With Data Studio IN PROGRESS


### Incidents

- 



### Concerns/Blockers

## Semaine du 08 mai 2020

### Accomplishments

- (done)  [GM-20759](#) - Graph Size Calculator Simple Script CLOSED
- (done) Address comments in PR
  - <https://ol-bitbucket.us.oracle.com/projects/OLDS/repos/datastudio/pull-requests/2929/overview>
- (done) Graph Size Calculator
  - Enhance UI/UX
  - Add advanced mode
  - Calculate memory estimation for FCC graphs using Graph Size Calculator, and compare it with Jonas's experimental results
- (done) Meetings
  - Progress meeting with mentor
  - Collaboration meetings with Jonas
  - Full team meeting

### Priorities

- (in progress) Determine error rate for Graph Size Calculator
  - [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif\\_oracle.com/repos/gsc-accuracy-tests/browse](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif_oracle.com/repos/gsc-accuracy-tests/browse)
- (open) Open PR to 'frontend-exchange' repository
- (in progress)  [GM-18671](#) - Graph Size Calculator 60 - IN REVIEW (TEAM)
- (in progress)  [GM-20761](#) - Graph Size Calculator Integration With Data Studio IN PROGRESS

### Incidents

•



### Concerns/Blockers

## Semaine du 15 mai 2020

### Accomplishments

- (done) Graph Size Calculator
  - Address comments in PR (UI/UX related issues)
- (done)  [PN-4815](#) - 'Add' Button in Share Dialog Cut Off in German CLOSED
- (done)  [PN-6248](#) - Initial code in a new paragraph is not syntax highlighted CLOSED

### Priorities

- (in progress)  [GM-18671](#) - Graph Size Calculator 60 - IN REVIEW (TEAM)
  - Calculate error rate experimentally
- (in progress)  [GM-20761](#) - Graph Size Calculator Integration With Data Studio IN PROGRESS

### Incidents

•

### Concerns/Blockers



## Semaine du 22 mai 2020

### Accomplishments

- (done) Graph Size Calculator Component
  - PR created : <https://ol-bitbucket.us.oracle.com/projects/OLDS/repos/exchange/pull-requests/11/overview>
- (done)  **PN-6994** - Builtin notebooks have no template **CLOSED**
- (done) TF - Brace Yourselves Users Are Coming - meeting

### Priorities

- (in progress)  **GM-18671** - Graph Size Calculator **60 - IN REVIEW (TEAM)**
  - Calculate error rate experimentally
- (in progress)  **GM-20760** - Graph Size Calculator Component **IN PROGRESS**
- (in progress)  **GM-20761** - Graph Size Calculator Integration With Data Studio **IN PROGRESS**
- (in progress)  **PN-6901** - Re-importing existing notebook with same "link" property leads to console error **IN REVIEW (TEAM)**

### Incidents

- 

### Concerns/Blockers





- Waiting for access to Horde

## Semaine du 29 mai 2020

### Accomplishments

- (in progress) Graph Size Calculator: Error rate
  - Results so far: [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif\\_oracle.com/repos/gsc-accuracy-tests/browse](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhelif_oracle.com/repos/gsc-accuracy-tests/browse)
- (done) Meetings
  - Progress review meeting
  - TF meeting
  - Meeting about design changes
  - Meeting with mentor

### Priorities

- (in progress)  **PN-4098** - PGX use case test **IN PROGRESS**
- (open)  **PN-7975** - Run markdown benchmark on Apache Zeppelin **CLOSED**
- (in progress)  **GM-20760** - Graph Size Calculator Component **IN PROGRESS**
- (in progress)  **GM-20761** - Graph Size Calculator Integration With Data Studio **IN PROGRESS**

### Incidents

- 




### Concerns/Blockers

## Semaine du 05 juin 2020

### Accomplishments

- (done) Update Graph Size Calculator PR according to follow-up meeting
- (done) Create Confluence page with results of benchmark on Apache Zeppelin
  - Page : [Run markdown benchmark on Apache Zeppelin](#)
- (done) Load graphs on horde and get their sizes
  - Results : [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhliif\\_oracle.com/repos/gsc-accuracy-tests/browse](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhliif_oracle.com/repos/gsc-accuracy-tests/browse)
- (done) Meetings
  - Full team meeting
  - Meetings with mentor

### Priorities

- (in progress)  **PN-4098** - PGX use case test **IN PROGRESS**
- (in progress)  **PN-7975** - Run markdown benchmark on Apache Zeppelin **CLOSED**
- (in progress)  **GM-20760** - Graph Size Calculator Component **IN PROGRESS**

### Incidents

- 




### Concerns/Blockers

## Semaine du 12 juin 2020

### Accomplishments

- (done) Meeting: Task force: Brace yourselves, users are coming
  - Presented the results of running markdown benchmark on Zeppelin
  - Presented the results of "pgx use case" benchmark, after adding requests for expanding the graph
- (done)  **PN-7975** - Run markdown benchmark on Apache Zeppelin **CLOSED**

### Priorities

- (in progress)  **GM-18671** - Graph Size Calculator **60 - IN REVIEW (TEAM)**
  - Finish error rate calculation. Results : [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhliif\\_oracle.com/repos/gsc-accuracy-tests/browse](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhliif_oracle.com/repos/gsc-accuracy-tests/browse)
  - Open PR to exchange repo
  - Resolve comments in PR
  - Add unit tests/improve code coverage
- (in progress)  **PN-4098** - PGX use case test **IN PROGRESS**
  - Updated the use case by adding requests for expanding the graph
- (in progress)  **GM-20760** - Graph Size Calculator Component **IN PROGRESS**

### Incidents

- 

### Concerns/Blockers



---

## Semaine du 19 juin 2020

### Accomplishments

- (done) Meetings
  - TF - Users Are Coming meeting
  - Meeting with mentor
  - Meeting with Jonas about error rate calculation and PR review.
- (done) Address comments in Pull Request
  - <https://ol-bitbucket.us.oracle.com/projects/OLDS/repos/exchange/pull-requests/11/overview>

### Priorities

- (in progress) Investigate Data Studio benchmark results (based on PN-7975)
  - <https://ol-confluence.us.oracle.com/display/OLDS/Run+markdown+benchmark+on+Apache+Zeppelin>
- (in progress)  GM-18671 - Graph Size Calculator **60 - IN REVIEW (TEAM)**
- (in progress)  GM-20760 - Graph Size Calculator Component **IN PROGRESS**

### Incidents

- 

### Concerns/Blockers

## Annexe 2

### Le script de calcul de la taille expérimentale d'un graphe PGX

```
package com.oracle.labs.gscaccuracytests;

import java.net.InetAddress;
import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;

import com.fasterxml.jackson.databind.JsonNode;
import oracle.pgx.api.PgqlResultSet;
import oracle.pgx.api.Pgx;
import oracle.pgx.api.PgxGraph;
import oracle.pgx.api.PgxSession;
import oracle.pgx.api.ServerInstance;

public class Main {
    private static final String PGX_CONF_PATH = "/scratch_user/mboukhli/apps/pgx-20.0.2/conf/pgx.conf";
    private static final String GRAPH_CONFIG = "lubm8k_corrupt.pgb.json";
    private static final ServerInstance instance = Pgx.getInstance(Pgx.EMBEDDED_URL);

    private static PgxGraph graph;

    public static void main(String[] args) throws Exception {
        instance.startEngine(PGX_CONF_PATH);

        PgxSession session = instance.createSession("my-session");
        graph = session
            .readGraphWithProperties(Main.class.getClassLoader().getResource(GRAPH_CONFIG).getPath(), "testgraph");

        // important to get accurate results from the admin api
        System.gc();

        printResult();
    }

    private static void printResult() throws Exception {
        JsonNode serverState = instance.getServerState();
        JsonNode graphAdminApi = serverState.get("graphs").get(0);
        JsonNode memoryAdminApi = serverState.get("memory");
        long onHeapMemory = onHeapMemory(memoryAdminApi);
        long offHeapMemory = offHeapMemory(memoryAdminApi);

        System.out.println("=====");
        System.out.println("Vertices count : " + graphAdminApi.get("vertices_num"));
        System.out.println("Edges count : " + graphAdminApi.get("edges_num"));
        System.out.println("Total : " + humanReadableFormat(totalMemory(onHeapMemory, offHeapMemory)));
        System.out.println("On-Heap : " + humanReadableFormat(onHeapMemory));
        System.out.println("Off-Heap : " + humanReadableFormat(offHeapMemory));
        System.out.println("Average string length : " + getAverageStringLength());
        System.out.println("=====");
    }

    private static double getAverageStringLength() {
        List<String> vertexValues = new ArrayList<>();
        List<String> edgeValues = new ArrayList<>();

        graph.getVertexProperties().stream()
            .filter(prop -> prop.getType().getTypeClass().getName().equals("java.lang.String"))
            .forEach(prop -> {
                try {
                    PgqlResultSet result = graph.executePgql("SELECT DISTINCT n." + prop.getName() + " FROM " + graph.getName() + " MATCH (n)");
                    while(result.next()) {
                        vertexValues.add(result.getString("n." + prop.getName()));
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                }
            });

        graph.getEdgeProperties().stream()
            .filter(prop -> prop.getType().getTypeClass().getName().equals("java.lang.String"))
            .forEach(prop -> {
                try {
                    PgqlResultSet result = graph.executePgql("SELECT DISTINCT e." + prop.getName() + " FROM " + graph.getName() + " MATCH (n) -[e]-> (m)");
                    while(result.next()) {
                        edgeValues.add(result.getString("n." + prop.getName()));
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                }
            });

        double vertexAvg = vertexValues.stream()
            .map(String::length)
            .collect(Collectors.averagingInt(Integer::intValue));

        double edgeAvg = edgeValues.stream()
            .map(String::length)
            .collect(Collectors.averagingInt(Integer::intValue));
    }
}
```

```
        if (vertexAvg == 0.0) {
            return edgeAvg;
        } else if (edgeAvg == 0.0) {
            return vertexAvg;
        } else {
            return (vertexAvg + edgeAvg) / 2;
        }
    }

    private static long offHeapMemory(JsonNode memoryAdminApi) throws Exception {
        return memoryAdminApi.get(InetAddress.getLocalHost().getHostName()).get(1).get("used_off_heap_mb").asLong()
            * 1_000_000;
    }

    private static long onHeapMemory(JsonNode memoryAdminApi) throws Exception {
        return memoryAdminApi.get(InetAddress.getLocalHost().getHostName()).get(0).get("used_heap_mb").asLong() * 1_000_000;
    }

    private static long totalMemory(long onHeap, long offHeap) {
        return offHeap + onHeap;
    }

    private static String humanReadableFormat(double sizeInBytes) {
        String[] units = { "kB", "MB", "GB", "TB", "PB", "EB", "ZB", "YB" };
        int thresh = 1024;

        if (Math.abs(sizeInBytes) < thresh) {
            return sizeInBytes + " B";
        }

        int u = -1;
        do {
            sizeInBytes /= thresh;
            ++u;
        } while (Math.abs(sizeInBytes) >= thresh && u < units.length - 1);

        return String.format("%.2f", sizeInBytes) + " " + units[u];
    }
}
```

## Annexe 3

### Exécution du script dans le cluster d'Oracle (Horde)

```

-bash-4.2$ srun gradle run
srun: job 3695141 queued and waiting for resources
srun: job 3695141 has been allocated resources
Starting a Gradle Daemon, 4 busy and 1 incompatible and 1 stopped Daemons could not be reused, use --status for details
> Task :compileJava
> Task :processResources
> Task :classes
> Task :run
ERROR StatusLogger No Log4j 2 configuration file found. Using default configuration (logging only errors to the console), or user programmatically provided configurations. Set system property 'log4j2.debug' to show Log4j 2 internal initialization logging. See https://logging.apache.org/log4j/2.x/manual/configuration.html for instructions on how to configure Log4j 2

```

## Annexe 4

### Horde - Résultats pour un graphe donnée

```

-bash-4.2$ srun gradle run
srun: job 3697598 queued and waiting for resources
srun: job 3697598 has been allocated resources
Starting a Gradle Daemon, 4 busy and 1 incompatible and 1 stopped Daemons could not be reused, use --status for details
> Task :compileJava
> Task :processResources
> Task :classes
> Task :run
ERROR StatusLogger No Log4j 2 configuration file found. Using default configuration (logging only errors to the console), or user programmatically provided configurations. Set system property 'log4j2.debug' to show Log4j 2 internal initialization logging. See https://logging.apache.org/log4j/2.x/manual/configuration.html for instructions on how to configure Log4j 2
=====
Vertices count : 186943565
Edges count : 397070701
Total : 27.33 GB
On-Heap : 3.18 GB
Off-Heap : 24.16 GB
Average string length : 11.078226740565892
=====

```

## Annexe 5

### Horde - Erreur de mémoire insuffisante

```

-bash-4.2$ srun gradle run
Starting a Gradle Daemon, 12 busy Daemons could not be reused, use --status for details
> Task :compileJava UP-TO-DATE
> Task :processResources UP-TO-DATE
> Task :classes UP-TO-DATE
> Task :run
ERROR StatusLogger No Log4j 2 configuration file found. Using default configuration (logging only errors to the console), or user programmatically provided configurations. Set system property 'log4j2.debug' to show Log4j 2 internal initialization logging. See https://logging.apache.org/log4j/2.x/manual/configuration.html for instructions on how to configure Log4j 2
Exception in thread "main" Exception in thread "Thread pool #10" Exception in thread "Thread pool #14" Exception in thread "Thread pool #12" in thread "Thread pool #17" java.lang.OutOfMemoryError: GC overhead limit exceeded
Exception in thread "Thread pool #16" java.util.concurrent.ExecutionException: java.lang.OutOfMemoryError: GC overhead limit exceededException
in thread "Thread pool #15" at java.util.concurrent.CompletableFuture.reportGet(CompletableFuture.java:357)
    at java.util.concurrent.CompletableFuture.get(CompletableFuture.java:1895)
    at oracle.pgx.api.PgxFuture.get(PgxFuture.java:99)
Exception in thread "Thread pool #13" at oracle.pgx.api.PgxSession.readGraphWithProperties(PgxSession.java:1919)
    at com.oracle.labs.gscaccuracytests.Main.main(Main.java:27)
Caused by: java.lang.OutOfMemoryError: GC overhead limit exceeded
java.lang.OutOfMemoryError: GC overhead limit exceeded
java.lang.OutOfMemoryError: GC overhead limit exceeded
java.lang.OutOfMemoryError: GC overhead limit exceeded
java.lang.OutOfMemoryError: GC overhead limit exceeded
java.lang.OutOfMemoryError: GC overhead limit exceeded
05:21:23.637 [Thread pool #12] ERROR oracle.pgx.runtime.util.RtsLoggingAdapter - 10788628 ms [rts thread 12] ERROR - GetJavaVM failed
> Task :run FAILED
FAILURE: Build failed with an exception.

* What went wrong:
Execution failed for task ':run'.
> Process 'command "/cm/shared/apps/java/jdk1.8.0_201/bin/java"' finished with non-zero exit value 134

* Try:
Run with --stacktrace option to get the stack trace. Run with --info or --debug option to get more log output. Run with --scan to get full insights.
* Get more help at https://help.gradle.org

BUILD FAILED in 3h 0m 55s
3 actionable tasks: 1 executed, 2 up-to-date
srun: error: horde041: task 0: Exited with exit code 1
-bash-4.2$

```

## Annexe 6

## Maquette du « Graph Size Calculator » - graphe non partitionné

**Graph Size Calculator**

**Vertices**

Count:

Vertex Id String Average Length:

**PROPERTIES**

**my-vertex-property (string)**  
Average length:

**my-fifth-vertex-property (string)**  
Average length:

**my-seventh-vertex-property (string)**  
Average length:

**Edges**

Count:

Edge Id String Average Length:

**PROPERTIES**

**my-edge-property (string)**  
Average length:

Scale vertex/edge count:

Scale string length:

Total : 569.24 MB +/- X%  
On-heap : 498.10 MB +/- X%  
Off-heap : 71.14 MB +/- X%

## Annexe 7

## Maquette du « Graph Size Calculator » - graphe partitionné

**Graph Size Calculator**

**Vertices**

Vertex Id String Average Length:

**Account provider**

Count:

**PROPERTIES**

**Name (string)**  
Average length:

**Country (string)**  
Average length:

**Customer provider**

Count:

**PROPERTIES**

**Given Name (string)**  
Average length:

**Email (string)**  
Average length:

**Edges**

Edge Id String Average Length:

**HasAccount provider**

Count:

**PROPERTIES**

**Label (string)**  
Average length:

**Currency (string)**  
Average length:

**Another property (string)**  
Average length:

Scale vertex/edge count:

Scale string length:

Total : 569.24 MB +/- X%  
On-heap : 498.10 MB +/- X%  
Off-heap : 71.14 MB +/- X%

## Annexe 8

### Page Confluence du « Graph Size Calculator »

The screenshot shows a Confluence page for 'Oracle Labs Data Studio'. The page title is 'GM-18671 Graph Size Calculator', created by Alexandra Fritzen and last modified by Mohamed Boukhil on Mar 30, 2020. It is currently in 'IN REVIEW (TEAM)' status. The page content includes sections for Stakeholders (FCC Studio, OGCS), Current Problem and Proposed Solution, Use Cases (Basic - Must Have - Graph Size), Acceptance Criteria, Feature Owner (@Mohamed Boukhil), and Functional Specification.

**Stakeholders**  
FCC Studio, OGCS

**Current Problem and Proposed Solution**  
As a DS user, I want to know how large my PGX graph is so that I can estimate the hardware that I need to load it.  
Calculating graph size manually can get very cumbersome and introduces lots of potential for error > See [Graph & Machine Sizing](#) for what FCC Studio is using/doing.  
We want to provide a well-tested framework to do this for the user. We can then use this framework for our benchmarking to verify that our components behave as expected for graphs of certain sizes.

**Use Cases**  
**Basic - Must Have - Graph Size**

1. Assumption: PGX.SM
2. User adds graph configuration in Graphs screen (<http://datastudio.oraclecorp.com/?root=graphs>)
3. User clicks on graph size calculator's icon
4. Tab opens where user can
  - a. Specify number of items per edge / vertex provider (e.g. 3million customer vertices, 20 million transaction edges)
  - b. OR scale up the graph specified by a factor (current size, x20, x5000)
5. And are provided with a size estimation of the graph when loaded into memory (on and off heap)

**Acceptance Criteria**

- ☐ User provides number of vertex / edge provider items and system calculates size of the graph
- ☐ User provides loaded graph and can click on scaling factor to calculate scaled-up size of the graph
- ☐ **Error margin for size calculation is less than 0.1%**
- ☐ Size estimation is given in human-readable format with error rate (e.g. 64GB +/- X%)
- ☐ Graph size calculation is well-tested for very large graphs (millions, billions of nodes)
- ☐ Test various different approaches to make it as easy as possible for the user
- ☐ Graph size calculator is integrated in the Graphs screen in DS
- ☐ Graph size calculator is integrateable into the PGX documentation [https://docs.oracle.com/cd/E56133\\_01/latest/whatsnew/whatsnew.html](https://docs.oracle.com/cd/E56133_01/latest/whatsnew/whatsnew.html)

**Feature Owner**  
[@Mohamed Boukhil](#)

**Functional Specification (What to implement?, UI Mocks, API Description)**  
The user should be able to estimate the size of a graph based on the graph configuration created in this page.

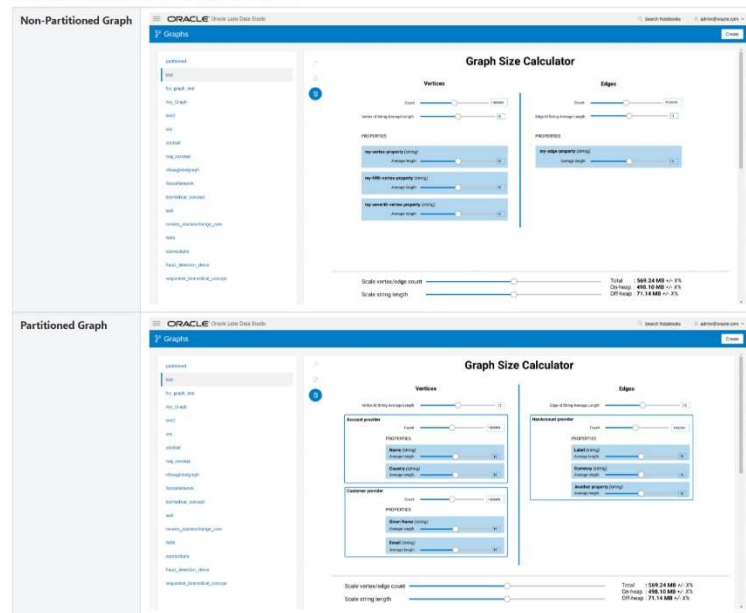
1. The user clicks a button and a tab opens up
2. The tab contains all the inputs needed for the calculation.
3. The inputs are pre-filled with random numbers
4. The user changes the inputs and the result changes accordingly
5. The result is shown, below the form, in human readable format with an error rate.



## UI Mocks



1. The user clicks the icon for graph size estimation (calculator icon)
2. The configuration tab shows up. (depending on the graph type)



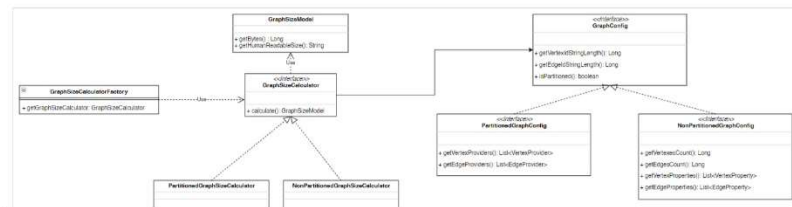
## Design Specification (How to implement?)

## Frontend

- We will have a third icon, under the plain configuration icon, this icon will have this css class "oj-tabbar-item-icon fa fa-calculator"

## Backend

- This feature will be a standalone JET library.
- There will be no persistence of the submitted graph config estimation
- Graph Size Calculator logic: [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhilf\\_oracle.com/repos/graph-size-calculator-ts/browse/README.md](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhilf_oracle.com/repos/graph-size-calculator-ts/browse/README.md)
- Example of usage: [https://ol-bitbucket.us.oracle.com/users/mohamed.boukhilf\\_oracle.com/repos/graph-size-calculator-example/browse](https://ol-bitbucket.us.oracle.com/users/mohamed.boukhilf_oracle.com/repos/graph-size-calculator-example/browse)
- Basic class diagram



## Testing

Number	Type	Prerequisite	Scenario	Expected Result
1	unit		Test string size depending on user entry	pass
2	unit		Test with some pre-known graphs config	pass

## User Stories (that speak in solving spirit of these problem areas)

Priority (1 lowest, 5 highest)	Story (As a < type of user >, I want < some goal > so that < some reason >.)
5	As a DS user, I want to be able to load a configuration file and get the estimated size of my graph, so that I can estimate the hardware that I'll need to load it.
5	As a PGX Documentation visitor, I want to be able to load a configuration file and get the estimated size of my graph, so that I can estimate the hardware that I'll need to load it.

Non-Goals

None

Prerequisites / Dependencies

- Graph Size Calculator Library

Open questions

- How this feature will be shipped?
  - ☐ Java Library
  - ☒ JS/let Library

Decisions that need resolution

Fundamentals

Introduces new external dependencies

No

Effects on related projects

Core Technology Package

- Waiting for the decision to be taken
- Explanation:

Includes breaking changes / affects backwards compatibility

No

Security Review Qualification Questions

Answer each of the following questions with yes or no. Our security SPOC to review. Source of the questions: Security Review Qualification Questions.

NB: the goal here is not to avoid a security review as much as possible, but to think proactively about security and make sure we are delivering secure software.

If you answer "yes" to any of the following questions (PGX-specific process)

1. Create a sub-page using the "Security Architecture / Design Review" template and fill it up
  - a. Click "..." next to "Create", select the "Security" space, scroll down and select the "Security Architecture / Design Review" template
2. Add a row in the table on the Security Review Log and Plan page and let @Sungpack Hong and @Davide Bartolini know, so that the review can be scheduled
3. Add our SPOC (@Roxana Bradescu) as reviewer to this page
4. Be ready to present the security aspects of this feature in one of the recurring meetings with our SPOC (you'll be invited)

	Question	Answer (yes / no + short explanation as appropriate)
1	Will this bug fix/enhancement/new feature be included in a shipped product? If no, security review is probably not required.  Note - you can optionally request a security you think would like to get early input on security design.	Yes
2	Does this bug fix/enhancement/new feature address security requirements/use cases? If yes, please provide a couple sentences to determine level of review required.	No
3	Are there any obvious security concerns for this feature? If yes, please provide a couple sentences to determine level of review required.	No
4	Does your bug fix/enhancement/new feature process sensitive or regulated data? If yes, please provide a couple sentences to determine level of review required.  Note sensitive data includes encryption keys, secrets, credentials, etc.	No
5	Does your bug fix/enhancement/new feature write any OS files system files or fork any process? If yes, please provide a couple sentences to determine level of review required.	No
6	Does your bug fix/enhancement/new feature access do any network communications? If yes, please provide a couple sentences to determine level of review required.	No
7	Does your bug fix/enhancement/new feature make a change in access control of any kind? If yes, please provide a couple sentences to determine level of review required.	No
8	Has this area had any previous security issues? If yes, please provide a couple sentences to determine level of review required.	No

Resources

- PGX Memory Consumption
- String Memory Usage in Java
- Non-Partitioned Graph Configuration
- Partitioned Graph Configuration

Reviewers

- ☒ @Alejandro De Gante
- ☐ @Alexander Weld
- ☒ @Alexandra Fritzen
- ☒ @Daniel Langerenken
- ☒ @Iraklis Psaroudakis
- ☒ @Michiel Haisma
- ☒ @Rania Medraoui
- ☒ @Sabrina Senna
- ☐ Hassan

**Rapport de Stage ENSET, Mohammedia 2020**

---

**RESUME**

Ce projet a eu lieu au sein de la société Oracle à Casablanca. Il concerne la conception et implémentation d'un composant web qui estime la taille mémoire des graphes PGX. Nous avons aussi travaillé sur la création d'un plan de test des charges sur Oracle Data Studio en utilisant des graphes PGX.

Nous avons travaillé dans un environnement SCRUM, en utilisant principalement la stack Atlassian. Lors de la phase de réalisation, nous avons utilisé le Framework Oracle JET, et l'outil JMeter pour les tests de charges.

**Mots clés : graphes PGX, composant web, tests de charges**

---

**SUMMARY**

This project took place within the Oracle corporation in Casablanca. It concerns the design and implementation of a web component which estimates the memory size of PGX graphs. We also worked on creating a load test plan on Oracle Data Studio using PGX graphs.

We worked in a SCRUM environment, mainly using the Atlassian stack. During the implementation phase, we used the Oracle JET Framework, and the JMeter tool for load testing.

**Keywords: PGX graphs, web component, load tests**