Moteur d’analyse de données des principales Blockchains

# Cas particulier d’une Blockchain bitcoin

##### Références : Rapport final de projet industriel

##### Odile BAIMA/Pierre NOMO, 14/03/17, version : 01

Table des matières

[Cas particulier d’une Blockchain bitcoin 1](#_Toc477081508)

[Remerciements 4](#_Toc477081509)

[Fiche d’identité et présentation du secteur d’activités de l’entreprise Mubiz 5](#_Toc477081510)

[Contexte 6](#_Toc477081511)

[Problématique 6](#_Toc477081512)

[Méthodologie et technologies 7](#_Toc477081513)

[Choix de la méthodologie 7](#_Toc477081514)

[Choix des technologies et outils 8](#_Toc477081515)

[Planning 8](#_Toc477081516)

[Travail réalisé 10](#_Toc477081517)

[Rédaction du cahier des charges 10](#_Toc477081518)

[Analyse 10](#_Toc477081519)

[Design 13](#_Toc477081520)

[Implémentation/Codage 13](#_Toc477081521)

[Dépôt Git 13](#_Toc477081522)

[Code source Java 15](#_Toc477081523)

[Scripts SQL 16](#_Toc477081524)

[Tests 16](#_Toc477081525)

[Résultats 17](#_Toc477081526)

[Conclusion 18](#_Toc477081527)

[Documents de référence 19](#_Toc477081528)

[Glossaire 20](#_Toc477081529)

# Remerciements

Ce rapport est le résultat d’un travail de recherche de près de 6 mois. En préambule, nous voulons adresser tous nos remerciements à toute personne, qui de loin ou de près nous a aidés - que ce soit par un conseil, par une information, ou par une suggestion - dans l’accomplissement de nos tâches.

Nous remercions Monsieur Albin CAUDERLIER, notre tuteur de projet et fondateur de l'entreprise MUBIZ pour la confiance qu’il nous a accordé, en nous confiant la réalisation de ce projet enrichissant.

Merci à Monsieur Marc PASQUET notre professeur principal, pour sa disponibilité et ses conseils dans l’accomplissement de la formation MTS.

Enfin nous remercions Monsieur Mohammed Achemlal, notre tuteur école pour son efficacité dans la communication avec Mubiz.

# Fiche d’identité et présentation du secteur d’activités de l’entreprise Mubiz

Le site de Mubiz (http://mubiz.com/) présente l’entreprise comme suit :

« … *Mubiz est une start up qui conseille et accompagne des entreprises de toutes tailles dans la découverte, l'expérimentation et l'utilisation de la technologie Blockchain. Elle est un prestataire technique facilitant l'accès aux réseaux décentralisés, dont bitcoin et les Blockchains. Mubiz fonctionne comme un Gmail ou un hotmail de e-commerce.  
Celle-ci commissionne 5% sur les projets ayant réussi un financement participatif et facture la mise en avant de produits. Mubiz n'enregistre pas les données des utilisateurs et ne réalise donc aucune revente ni analyse de ces données. Mubiz se différencie par son utilisation de*[*Bitcoin*](http://fr.mubiz.com/faq)*et des avantages des Blockchains. Ainsi, Mubiz est mondial et met en relation des porteurs de projets et des contributeurs présents à travers le monde, sans restriction d'accès.   
De plus, les transactions de Mubiz sont entièrement transparentes et publiées dans la Blockchain Bitcoin. Chacun peut consulter l'utilisation qui est faite des fonds et suivre l'avancement des projets. La réalisation de transactions en Bitcoin étant bien moins onéreuse que celles bancaires, les coûts d'utilisation de Mubiz sont inférieurs à ceux des autres plates formes de financement participatif. Ainsi, la commission perçue par Mubiz est de 5%, bien inférieure à celles d'autres plate formes, qui perçoivent entre 9% et 10% de commissions.   
De plus, les transactions de Mubiz sont entièrement transparentes et publiées dans la Blockchain Bitcoin. Chacun peut consulter l'utilisation qui est faite des fonds et suivre l'avancement des projets. Enfin, Mubiz est instantané. Les contributions sont visibles dans la seconde sur le compte du projet sans délai. Pour bénéficier des progrès apportés par les Blockchains, Mubiz sélectionne les projets afin qu'ils respectent*[*une déontologie*](http://fr.mubiz.com/faq/#control)*et collabore à la lutte contre le blanchiment d'argent* … »

En résumé, Mubiz est une entreprise spécialisée dans la technologie Blockchain plus particulièrement dans les Blockchains bitcoin et ethereum. Elle propose ses services dans le domaine et se finance par les commissions sur ceux-ci.

# Contexte

Les Blockchains offrent des données ouvertes et structurées pour les moteurs de data mining. Au-delà, le data mining et l’intelligence artificielle permettent de définir des modèles prévisionnels de l'activité des Blockchains. Mubiz souhaite mettre en place pour un client un moteur d’analyse des données des principales Blockchains. Il s’agit ici de croiser les données collectées de la Blockchain, qui seront au préalable stockées dans une base de données. On pourrait s’imager la création de vues statistiques sur les données récoltées. Le chapitre suivant présentera sous forme détaillée un exemple d’architecture possible.

# Problématique

Dans ce projet il s’agit d’implémenter une architecture d’un moteur d’analyse des données des Blockchains, notamment des Blockchains bitcoin et ethereum. Mubiz souhaite récupérer ces données que l’on stockera dans une base de données, modélisée au préalable. Dans notre cas, le modèle de la Blockchain est implémenté dans des web services, fournis par Mubiz. Les données seront donc collectées par un client HTTP à implémenter, afin d’être chargées dans la base de données à modéliser. La figure ci-dessous récapitule le contexte expliqué ci-dessus.

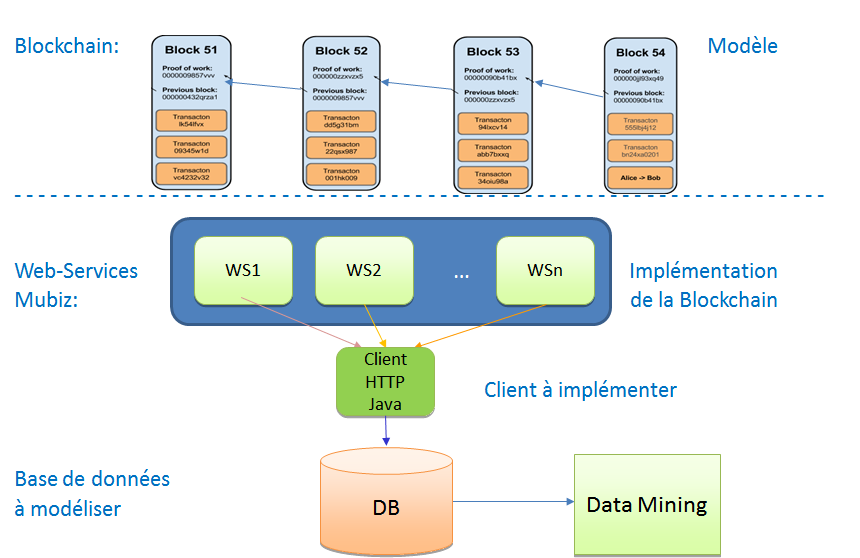


Figure 1 : Schéma d’ensemble du projet à réaliser

Pour récapituler, le travail à faire comprendra les tâches suivantes :

* Implémenter un client qui se chargera d’envoyer des requêtes HTTP vers les web-services Mubiz. Celui-ci se collectera d’autre part les données de la réponse afin de les charger dans la base de données
* Modéliser la base de données qui stockera les données récoltées.

# Méthodologie et technologies

## Choix de la méthodologie

* Pour la conduite du projet, nous avons opté pour une combinaison de méthodologies en V et en spirale. La méthodologie en spirale nous est apparu la mieux adaptée pour la modélisation de la base de données. N’ayant pas une vue d’ensemble claire sur les données produites par la Blockchain, il nous était difficile d’avoir un modèle conceptuel clair de données. Nous sommes donc partis sur une modélisation empirique, d’où la méthode en spirale.
* En ce qui concerne l’implémentation du client Java http nous avons choisi la méthodologie en V. En d’autres termes, pour la complexité relativement moyenne de l’implémentation de celui-ci, ainsi que de ses modules associés, un modèle classique de développement en V nous semblait évident.

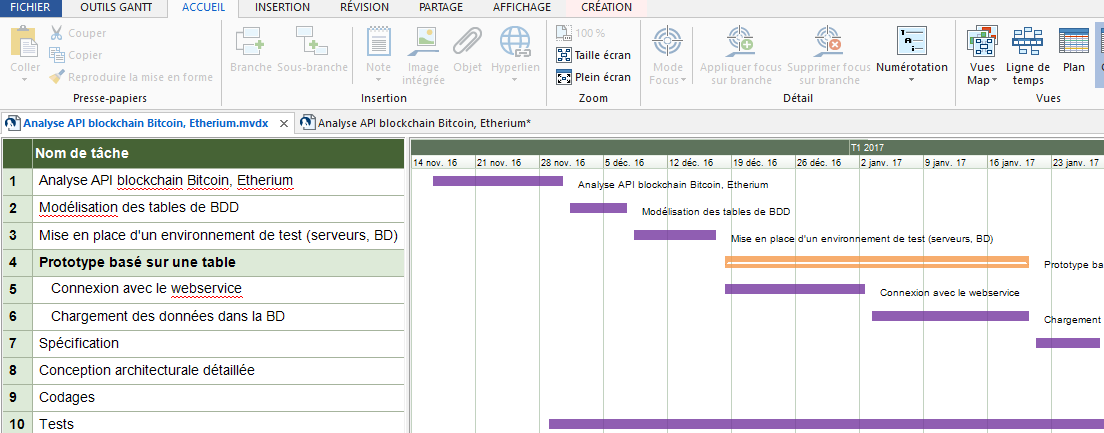
## Choix des technologies et outils

Pour la réalisation du projet, nous avons opté pour les choix suivants :

* Le client HTTP et ses modules associés, seront programmés en Java
* L’outil de programmation utilisé est Eclipse
* Le SGBDR qui porte la base de données est MySQL
* Le code Java, SQL ainsi que les documents inhérents au projet seront sous contrôle de version dans la plate-forme GitHub
* Les modules associés au client java utilisent des bibliothèques externes, notamment la bibliothèques GSON de Google pour le parseur JSON, ainsi que de la technologie Java Database Connectivity (JDBC)

# Planning

Pour la réalisation de notre projet, nous sommes convenus sur un planning comme sur la figure ci-dessous :



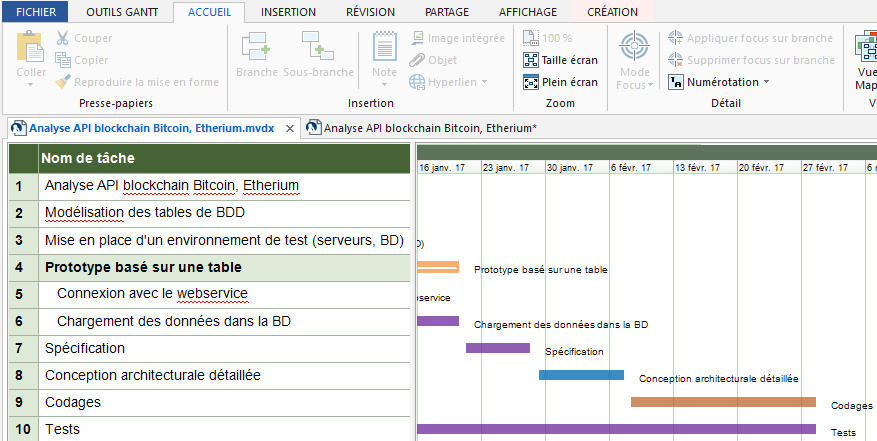


Figure 2 : Diagrammes de GANTT pour le projet

Notre analyse nous mena à un Planning estimé à 54 jours hommes pour un découpage du projet comme suit:

* Analyse et compréhension des API Blockchain bitcoin et ethereum (***11 jours***)
* **La modélisation de la base de données** et donc des tables qui récupèrent les données des différentes Blockchain. Pour rappel l’effort du travail fut concentré sur la Blockchain bitcoin et en particulier sur la table « ***Block*** », puisque priorité lui fut accordée pour une première implémentation (***1 semaine***).
* **La mise en place d’un environnement de test à distance** mis à disposition par MUBIZ. Nous n’avons finalement plus utilisé cet environnement, car les tests se sont fait directement sur nos machines. Par contre cet environnement sera utilisé en production compte tenu de la masse de données à récolter (***1*** ***semaine***).
* **La création d’un prototype** basé sur la table « ***Block*** » (***25 jours***). Cette étape peut être assimilée à un « Proof of Concept » c’est-à-dire à vérifier que le concept d’ensemble est juste, auquel cas il peut être généralisé aux autres tables. Il se décompose en 2 sous étapes :
  + Implémentation et test de la connexion avec les web services (***11 jours***)
  + Implémentation et test du chargement des données dans la base de données (***14 jours***)
* La spécification (***5 jours***)
* La conception architecturale détaillée (***1*** **semaine**)
* Les codages (***2 semaines***)
* Les tests (***66 jours***)

# Travail réalisé

## Rédaction du cahier des charges

Pour la définition de nos besoins, la rédaction d’un cahier de charge fut nécessaire.

## Analyse

Dans un premier temps, une compréhension de la théorie sur la Blockchain fut nécessaire. Ensuite, nous avons dû nous familiariser avec la documentation Bitcoin et Ethereum sur le site (<https://www.blockcypher.com/dev/bitcoin/>) afin de s’imprégner de la structure et des différends éléments logiques de la Blockchain. Riches de nos informations tirées de la documentation, nous avons entrepris l’analyse des comportements des Web-services Mubiz.

Pour la réalisation de notre base de données, nous avons opté pour le SGBDR MySQL, celui-ci étant libre. Sur demande expresse de Mubiz, la suite du travail consistait à faire une étude sur les monnaies virtuelles bitcoin et ethereum et particulièrement sur les données de celles-ci. De par leur similarité, nous sommes arrivés au constat qu’elles avaient des données communes. Il s’agissait donc pour nous par la suite, de modéliser les tables qui en collecteraient les données. Nous parvînmes à un modèle à trois tables à savoir :

* Une Table Block, comportant les données de chaque bloc de la chaîne
* Une Table Transaction, comportant les données transactionnelles de la chaîne
* Une Table Adresse, comportant les données adresse des clients de la chaine

Ce modèle conceptuel fut validé par Mubiz. Pour une première implémentation dudit modèle, la priorité fut donnée à la table « Block » ; ceci sur décision de notre tuteur entreprise. La suite du travail consistait à modéliser la table. Notre analyse nous mena à une table avec les champs comme sur la figure de la page suivante :

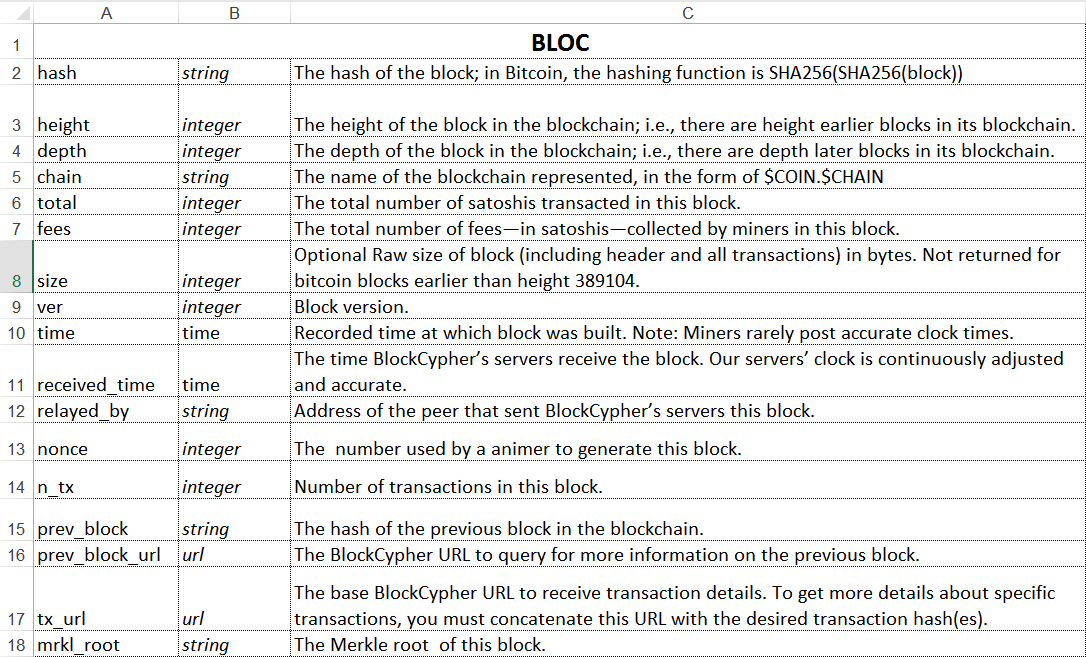


Figure 3 : Structure de la table Block

La Blockchain présente la caractéristique particulière de produire des données en masse. Il s’est donc posé pour nous la question des différents types à assigner aux différents champs de la base de données. Et surtout de trouver les types de données équivalentes MySQL à celles spécifiées dans la documentation bitcoin et ethereum. En parcourant la documentation MySQL, nous avons trouvé les types appropriés des champs.

Pour un bon contrôle qualité logicielle de notre produit, nous sommes convenus, de commun accord avec Mubiz, d’utiliser la plateforme de gestion de version GitHub. Une lecture assidue de la documentation Git, nous a permis de configurer les dépôts Git locaux et distants du projet.

Avant la phase de design des classes java, une analyse préalable fut nécessaire, notamment de la structure et de l’interaction entre les différentes classes. D’autre part nous avons dû nous imprégner des techniques de programmation de la technologie JDBC. Les web-services Mubiz produisant des réponses sous le format JSON, il était logique de s’informer sur les techniques de parsage JSON. Après une recherche intensive sur internet, nous avons eu recours à la librairie externe Gson de parsage du format JSON, implémentée par Google.

## Design

Cette phase est constituée principalement du design des classes java conséquentes au design du client java http. Pour rappel (voir figure 1) le client java est chargé de collecter les données de la Blockchain. Dans notre cas ces données proviennent des web services Mubiz qui implémentent la Blockchain. Ensuite le client stocke les données collectées dans la base de données. Il s’ensuit que le client doit s’appuyer sur des classes apportant les fonctionnalités suivantes :

* La connectivité avec la base de données. Comme indiqué plus haut nous nous sommes aidés de la technologie JDBC.
* La connectivité avec les web services. Le client communique avec ceux-ci sous le protocole http en envoyant des requêtes http simples. Pour ce faire, nous avons eu recours aux librairies externes httpclient-4.5.2 et httpcore-4.4.6.
* Un module de parsage JSON.
* Un module qui encapsule les données provenant des web services.

## Implémentation/Codage

### Dépôt Git

Pour débuter la phase de codage, nous avons configuré un dépôt Git distant ainsi que local ; ceci sur demande du tuteur entreprise pour assurer une bonne qualité logicielle du produit. La structure du dépôt Git se présente comme sur la figure de la page suivante.

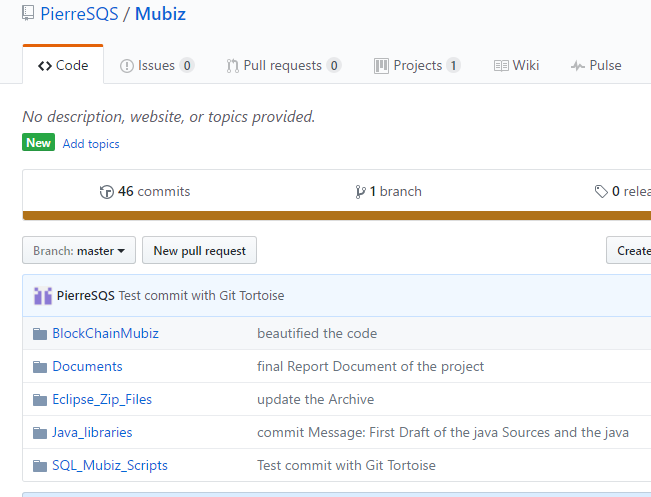


Figure 4 : Structure du dépôt Git distant

Le dépôt distant Git se présente comme suit :

* Un répertoire nommé « **BlockChainMubiz** » code source
* Un répertoire nommé « **Documents** » comportant les documents inhérents au projet
* Un répertoire nommé « **Eclipse\_Zip\_Files** » comportant un fichier Jar du projet java Eclipse
* Un répertoire nommé « **Java\_librairies** » comportant les librairies externes
* Un répertoire nommé « **SQL\_Mubiz\_Scripts** » comportant les différents scripts SQL, à savoir le script d’initialisation, de création, d’insertion, et celui de la base de donnée.

### Code source Java

Le code source java du projet développé avec l’outil Eclipse, se compose de 9 classes réparties dans 5 packages et 6 bibliothèques externes. Les 6 librairies Java externes fournissent les fonctionnalités suivantes :

* Connectivité avec la base de donnée
* Fonctionnalités pour le protocole http
* Parsage du format JSON

Les classes Java réparties dans 5 packages de fonctionnalité sont structurées comme suit :

* 3 classes encapsulant les données de la Blockchain c’est à dire des web services
* 2 grandes classes qui assurent l’interaction avec le SGDBR MySQL
* 1 classe fournissant les fonctionnalités du Protocole http
* 1 classe constituant le parseur JSON
* 2 classes comportant d’une part le client Java en soi et une classe client de tests

Le code source comporte aussi des fichiers de configuration, par exemple le port de communication avec la base de données, le nom de la base de données ou le nom du serveur MySQL. Ceci assure la portabilité du programme ainsi que la flexibilité dans son déploiement sur l’infrastructure d’exécution.

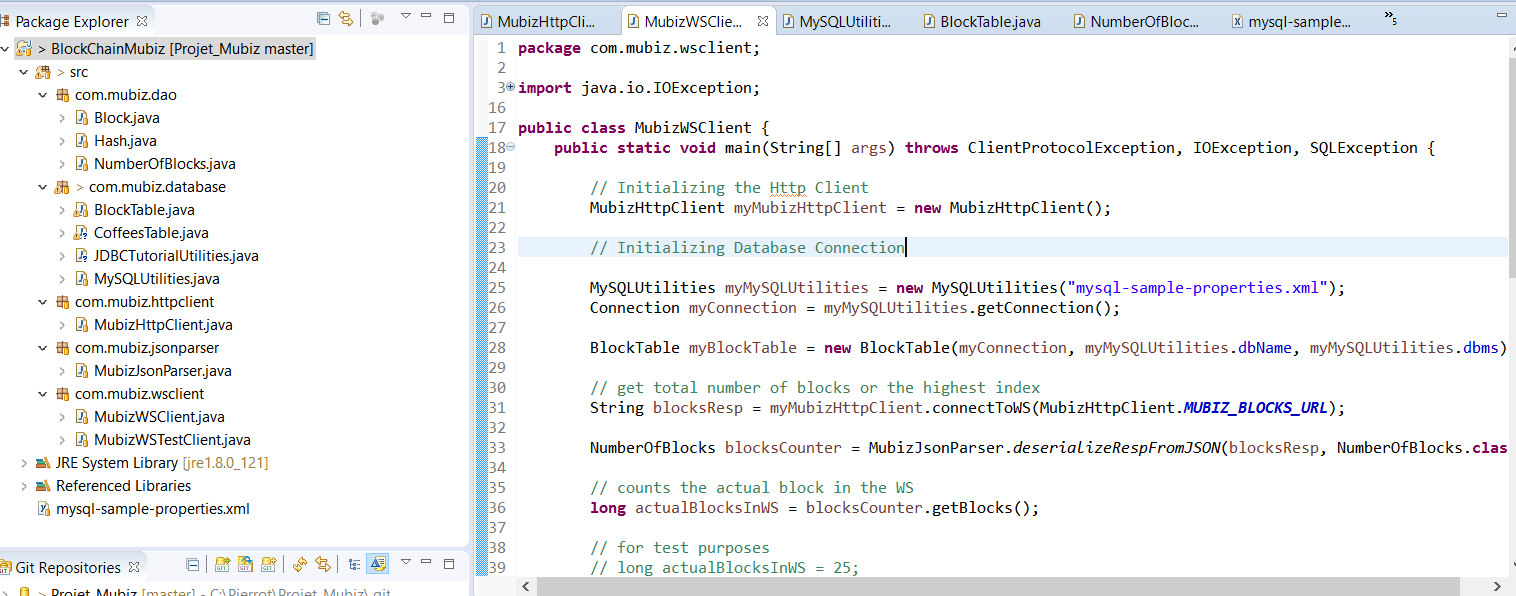


Figure 4 : Code source dans Eclipse.

### Scripts SQL

Le Script d’initialisation initialise la base de données en permettant le mode test, qui efface la base de donnée si elle existe, la recrée ainsi que la table « *Block* » chargée de collecter les données des blocs de la Blockchain. Ceci permet d’avoir une base de données propre à chaque fois qu’on fera une modification de celle-ci. Ce script sera exécuté afin que les nouvelles modifications soient considérées.

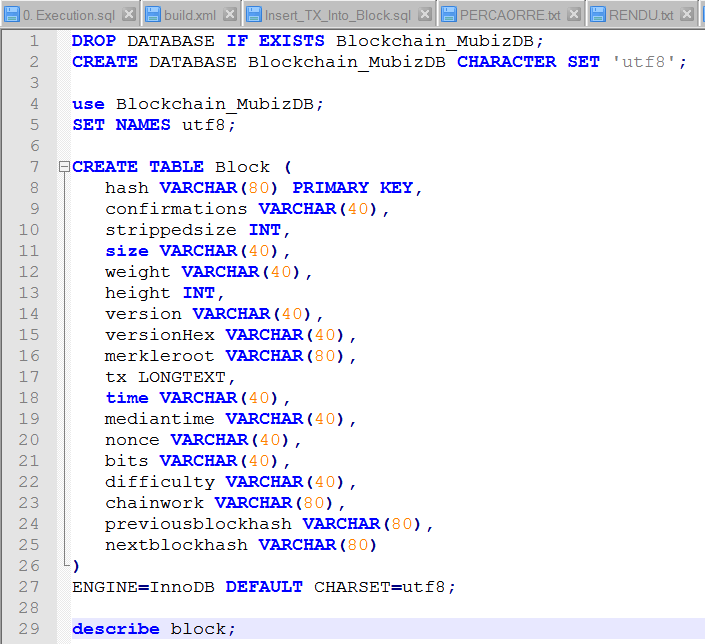


Figure 5 : Extrait du script d’initialisation

Le Script d’insertion nous sert à insérer manuellement de nouvelles données dans la table « *Block* ». Ce script sert essentiellement aux tests d’insertions manuels des données de la Blockchain. Ceci est nécessaire, pour pouvoir tester à chaque fois le modèle conceptuel de la table du fait qu’on n’a pas une vue d’ensemble sur les données globales fournies par la Blockchain.

## Tests

La phase de tests était essentiellement constituée de tests du code Java ainsi que du perfectionnement des scripts SQL.

Les Test des scripts SQL, ont débuté par une phase intensive et laborieuse de préparation des données de tests. Ensuite il fut nécessaire d’affiner les scripts. Ce perfectionnement fut surtout justifié par la recherche des types appropriés à assigner aux différents champs. Pour rappel, dans la modélisation de la base de données, nous avons attribués les types aux champs de façon empirique dans un premier temps ; les tests étant effectués sur un échantillon de données. Nous avons par la suite affiné les types au fur et à mesure de l’avancement des tests. Pendant cette phase, le champ récoltant les transactions des blocs, est celui qui nous posa le plus de difficulté, du fait de la quantité de données qu’il transporte. Spécialement pour ce champ dénommé « *tx* », il fut effectivement difficile de trouver le type adapté.

Le code Java fut essentiellement testé par une série de tests unitaires par modules c’est-à-dire les modules fonctionnels d’interaction avec la base de données, de parsage JSON JSON et enfin le module qui gère les requêtes HTTP.

# Résultats

Après une phase de tests intensifs, on peut constater que les premiers résultats sont satisfaisants car :

* Le code Java est bien traçable sur le dépôt distant Git
* Le modèle conceptuel et logique de la base de données s’avère être correct. Les données se chargent sans difficulté. La collaboration entre les différentes classes ne présente pas d’anomalies particulières
* Ceci confirme que les modules java collaborent correctement. Les requêtes HTTP sont bien transmises aux web-services, la communication avec la base de données se passe de façon stable, le module de parsage JSON ne présente lui aussi aucun dysfonctionnement notable jusqu’à présent

# Conclusion

Notre travail pose les premiers jalons, pour une généralisation du concept sur d’autres Blockchain notamment la Blockchain ethereum. En effet comme signalé plus haut, le client HTTP joue son rôle sans problème majeur de performance, malgré la masse des données. Les données collectées ne présentent aucune inconsistance notoire. Pour Mubiz nous avons ainsi réalisé la première étape du Moteur d’analyse de données. En effet, l’objectif final du projet est de relier notre base de données à un outil de Data Mining. On peut s’imaginer des statistiques sur les données récoltées. On peut par exemple faire des statistiques sur les transactions les plus fréquentes, des statistiques sur les produits les plus achetés, des statistiques sur une période bien définie, le choix des traitements statistiques est illimité.

Ce projet nous a permis d’approfondir nos connaissances sur des sujets divers compte tenu de nos profils si divergents. Ainsi nous avons appris entre autre la création et l’administration d’un dépôt Git, de monter en compétence sur les bases de données, de se perfectionner dans le langage SQL. Nous avons pris connaissance des méthodes utilisées dans la réalisation des tests. Nous sommes d’avis que par ce projet, nous avons pu constater en partie le contexte en entreprise. Nous pouvons conclure que l’expérience a été bénéfique autant pour l’entreprise que pour nous.

# Documents de référence

<http://mubiz.com/>

<https://www.blockcypher.com/dev/bitcoin/#block>

<https://blockchainfrance.net/decouvrir-la-blockchain/c-est-quoi-la-blockchain/>

<https://github.com/>

<https://fr.wikipedia.org/wiki/Service_web>

<https://www.jmdoudoux.fr/java/dej/chap-gson.htm>

<https://fr.wikipedia.org/wiki/Dépôt_(informatique>)

<https://fr.wikipedia.org/wiki/Client_HTTP>

<http://www.base-plus.fr/qqoqccp-du-data-mining/>

<http://www.journaldunet.com/solutions/pratique/dictionnaire-du-webmastering/technologies-langages/19528/script-definition.html>

<https://fr.wikipedia.org/wiki/Java_Database_Connectivity>

# Glossaire

|  |  |
| --- | --- |
| **Termes** | **Définition** |
| **Blockchain**  source : https://blockchainfrance.net/decouvrir-la-blockchain/c-est-quoi-la-blockchain/ | La Blockchain est une technologie de stockage et de transmission d’informations, transparente, sécurisée, et fonctionnant sans organe central de contrôle. |
| **Bitcoin**  Source : https://bitcoin.org/fr/ | Bitcoin est une Blockchain. La gestion des transactions et la création de bitcoins est prise en charge collectivement par le réseau. Bitcoin est libre et ouvert. Sa conception est publique, personne ne possède ni ne contrôle Bitcoin et [tous peuvent s'y joindre](https://bitcoin.org/fr/supporter-bitcoin). Grâce à plusieurs de ses propriétés uniques, Bitcoin rend possible des usages prometteurs qui ne pourraient pas être couverts par les systèmes de paiement précédents. |
| **Web services**  Source : https://fr.wikipedia.org/wiki/Service\_web | Un service web  est un protocole d'interface [informatique](https://fr.wikipedia.org/wiki/Programme_informatique) de la famille des [technologies web](https://fr.wikipedia.org/wiki/Technologies_Web) permettant la communication et l'échange de données entre applications et systèmes hétérogènes dans des [environnements distribués](https://fr.wikipedia.org/wiki/Architecture_distribuée). |
| **Data Mining**  Source : | Le data mining regroupe l’ensemble des technologies susceptibles d’analyser les informations d’une base de données pour y trouver des informations utiles à l’action voulue et d’éventuelles corrélations signifiantes et utilisables entre les données. |
| **Script** | Un langage de script est un [langage de programmation](https://fr.wikipedia.org/wiki/Langage_de_programmation) interprète. Il s'agit d'une suite d'instructions simples et souvent peu structurées qui permettent l'automatisation de certaines tâches. De façon plus directe, le script manipule les fonctionnalités d'un système informatique. |

|  |  |
| --- | --- |
| **Termes** | **Définition** |
| **GitHub** | GitHub est une plateforme open source de gestion de versions et de collaboration destinée aux développeurs de logiciels |
| **Web services**  Source : https://fr.wikipedia.org/wiki/Service\_web | Un service web  est un protocole d'interface [informatique](https://fr.wikipedia.org/wiki/Programme_informatique) de la famille des [technologies web](https://fr.wikipedia.org/wiki/Technologies_Web) permettant la communication et l'échange de données entre applications et systèmes hétérogènes dans des [environnements distribués](https://fr.wikipedia.org/wiki/Architecture_distribuée). |
| **Data Mining**  Source : http://www.base-plus.fr/qqoqccp-du-data-mining/ | Le data mining regroupe l’ensemble des technologies susceptibles d’analyser les informations d’une base de données pour y trouver des informations utiles à l’action voulue et d’éventuelles corrélations signifiantes et utilisables entre les données. |
| **Parseur JSON**  Source : | Un parseur est un composant logiciel d’analyse syntaxique. Un parseur JSON permet d’analyser la syntaxe des données sur le format JSON |
| **JDBC**  Source **:**  https://www.jmdoudoux.fr/java/dej/chap-jdbc.htm | JDBC est l'acronyme de Java DataBase Connectivity et désigne une API pour permettre un accès aux bases de données avec Java. |
| **Client HTTP**  Source :https://fr.wikipedia.org/wiki/Client\_HTTP | Un client HTTP est un [logiciel](https://fr.wikipedia.org/wiki/Logiciel) conçu pour se connecter à un [serveur HTTP](https://fr.wikipedia.org/wiki/Serveur_HTTP). |
| **GSON**  Source :  https://www.jmdoudoux.fr/java/dej/chap-gson.htm | GSON est une bibliothèque open source développée par Google pour convertir un objet Java dans sa représentation JSON et vice versa. |