Les meilleures pratiques pour créer des classeurs Tableau efficaces

Tableau 10

Alan Eldridge
Tableau Software

À propos de ce document

Je souhaiterais rappeler que j'ai principalement cherché à combiner et condenser les écrits de nombreux auteurs dans un document unique. Certains lecteurs reconnaîtront peut-être leur production dans des sections, voire des passages entiers. Je tiens à remercier tous ces auteurs, qui ont eu par ailleurs la bonté de ne pas engager des poursuites pour violation de leurs droits d'auteur, sans qui ce livre blanc n'existerait pas.

Je remercie également tous ceux qui ont relu ce document et contribué à en améliorer la précision et la clarté. Je n'aurais pas été en mesure de produire un texte de cette qualité sans votre souci du détail et sans la patience dont vous avez fait preuve dans vos commentaires.

Ce livre blanc a été mis à jour de manière à refléter les fonctionnalités de Tableau 10. Certaines recommandations pourront changer en fonction des nouveautés des futures versions de Tableau.

Merci pour votre attention.

Alan Eldridge
Juin 2016

TL;DR

Ceux qui ont lu ce livre blanc ou qui le recommandent me disent souvent qu'il est trop long. Je leur réponds que cela est nécessaire pour aborder en détail un sujet aussi vaste.

Voici cependant un résumé des principaux points abordés.

- Il n'existe pas de solution miracle pour les classeurs qui manquent d'efficacité. Commencez par consulter l'enregistrement des performances pour comprendre les causes des problèmes. Vos requêtes sont-elles trop nombreuses et mettent-elles beaucoup de temps à s'exécuter ? Vos calculs s'effectuent-ils lentement ? Le rendu est-il complexe ? Utilisez les informations ainsi obtenues pour mieux orienter vos efforts.
- Même si ce document propose une série de meilleures pratiques, il ne s'agit que de recommandations qu'il vous appartient de tester pour vérifier qu'elles permettent d'améliorer les performances dans votre situation particulière. Les performances dépendent en effet souvent de la structure de vos données et de la source de données que vous utilisez (fichiers à plat, bases de données relationnelles ou extraits de données).
- Les extraits sont un moyen simple et rapide d'accélérer la plupart des classeurs.
- Plus vos données sont épurées et mieux elles reflètent la structure de vos questions, autrement dit moins de préparation et de manipulation sont nécessaires, plus vos classeurs seront rapides.
- La plupart du temps, la lenteur d'un tableau de bord est le résultat d'une mauvaise conception. C'est le cas par exemple s'il contient trop de graphiques ou si vous cherchez à afficher trop de données à la fois. Privilégiez la simplicité. Laissez les utilisateurs accéder progressivement aux détails plutôt que d'essayer de tout afficher et de leur permettre de filtrer ensuite.
- Utilisez uniquement les données dont vous avez besoin, aussi bien pour les champs auxquels vous faites référence qu'au niveau de la granularité des enregistrements retournés. Résultat : Tableau génère moins de requêtes, qui sont alors plus efficaces et plus rapides, et la quantité de données à déplacer de la source de données vers le moteur est moins importante. Les classeurs sont aussi plus petits, et donc plus faciles à partager et plus rapides à ouvrir.
- Lorsque vous réduisez la quantité de données utilisée, vous devez penser à utiliser les filtres de manière efficace.
- Les chaînes et les dates sont plus lentes que les données numériques et booléennes.

Pour finir, certaines des recommandations présentées ici n'auront un impact que si vous travaillez avec des ensembles de données complexes ou volumineux. Ce que l'on entend par là dépend de nombreux facteurs, mais il est toujours utile de suivre ces recommandations pour tous vos classeurs, au cas où le volume de données augmenterait. C'est en forgeant que l'on devient forgeron.

Sommaire

Introduction

Quels sont les points forts de Tableau ?

Chez Tableau, nous cherchons à changer la façon de voir et comprendre les données, ainsi que la manière d'interagir avec elles. Par conséquent, nous n'avons pas pour but d'offrir le même type d'expérience que sur les plates-formes BI traditionnelles. Tableau est l'outil idéal si vous souhaitez créer des classeurs avec les caractéristiques suivantes.

- **Visuels** : il est plus qu'évident qu'une représentation visuelle constitue le moyen le plus efficace de comprendre des ensembles de données volumineux et complexes. Par défaut, Tableau présente les données sous forme de graphiques, de diagrammes et de tableaux de bord. Les tables et les tableaux croisés sont toujours utilisés et pris en charge. Nous expliquerons plus loin comment les utiliser au mieux.
- **Interactifs** : les documents Tableau sont conçus pour fournir l'interactivité sur un ordinateur, sur Internet ou sur un appareil mobile. Contrairement à d'autres outils d'aide à la décision, qui produisent des résultats destinés à être imprimés, que ce soit sur du papier ou dans un fichier PDF, l'accent est mis sur la création d'expériences riches et interactives qui permettent aux utilisateurs d'explorer et d'interroger les données.
- **Itératifs** : le processus de découverte est cyclique par définition. Tableau est conçu pour accélérer le cycle question-réponses-question afin d'aider les utilisateurs à formuler rapidement une hypothèse, la tester avec les données disponibles, la rectifier, la tester à nouveau, et ainsi de suite.
- **Rapides** : le processus d'aide à la décision a toujours été caractérisé par une certaine lenteur, que ce soit dans l'installation et la configuration du logiciel, la mise à disposition des données à analyser, et la conception et la mise en œuvre des documents, des rapports ou des tableaux de bord. Tableau permet aux utilisateurs d'installer le logiciel, de se connecter aux données et de développer des documents plus rapidement que jamais. Très souvent, le délai d'obtention des réponses passe de plusieurs mois ou semaines à quelques heures ou minutes.
- **Simples** : les outils d'aide à la décision traditionnels sont souvent hors de portée de la plupart des utilisateurs métier en termes de coût et de complexité. Bien souvent, les utilisateurs ont besoin de l'IT ou d'un expert pour créer les requêtes et les documents souhaités. Tableau propose une interface intuitive pour permettre aux néophytes de créer des requêtes et d'analyser des données complexes sans avoir à devenir des experts en base de données ou en feuilles de calcul.
- **Attrayants** : on dit que la beauté est dans l'œil de celui qui regarde, mais en matière de communication visuelle, il faut suivre les meilleures pratiques. Grâce à des fonctionnalités comme l'outil Montre-moi, Tableau permet à des personnes sans connaissances techniques de créer des graphiques efficaces et clairs avec les données utilisées.
- **Compatibles avec plusieurs plates-formes** : les utilisateurs créent de plus en plus souvent des documents destinés à plusieurs plates-formes. Ils doivent en effet afficher les données et interagir avec elles sur leur ordinateur de bureau, sur Internet, sur des appareils mobiles, ou encore à l'intérieur d'autres applications ou documents. Tableau permet de publier un document unique et de l'utiliser sur toutes ces plates-formes sans avoir à l'adapter ou le recréer.

Dans quels cas Tableau n'est-il pas l'outil adéquat ?

Tableau est un outil riche et puissant, mais vous devez être conscient dès le départ que pour certains problèmes, il ne constitue pas forcément la solution optimale. Vous pouvez tout de même utiliser le logiciel Tableau, car il permet d'effectuer de nombreuses tâches pour lesquelles il n'a pas été développé à l'origine. Sachez toutefois que comme il n'a pas été conçu pour ces types de problématique, le résultat risque de ne pas être aussi efficace que vous le souhaitiez.

Vous devriez peut-être reconsidérer vos attentes ou opter pour une approche différente dans les situations suivantes.

- Vous avez besoin d'un document conçu pour un support papier, pas un écran. Autrement dit, vous devez contrôler une mise en page complexe ou créer des en-têtes et pieds de page pour des pages, des sections ou des groupes, ou vous avez besoin d'un format WYSIWYG précis. Tableau peut produire des rapports de plusieurs pages, mais vous avez moins de contrôle sur la mise en page qu'avec des outils de reporting dédiés.
- Vous recherchez un mécanisme de remise push complexe pour vos documents personnalisés (envois périodiques) que vous envoyez de plusieurs manières. Tableau Server propose un service d'abonnement aux rapports qui permet à un utilisateur de s'abonner (ou d'abonner d'autres utilisateurs dans la version 10) pour recevoir des rapports par e-mail, mais les clients recherchent parfois une solution plus flexible. Tableau peut être utilisé dans la création de systèmes de remise push, mais il ne s'agit pas d'une fonctionnalité native. Il faut alors développer une solution personnalisée basée sur l'utilitaire TABCMD ou recourir en sus à des solutions tierces comme *VizAlerts* (https://community.tableau.com/thread/191632) ou *Push Intelligence for Tableau* de Metric Insights (http://bit.ly/1HACxul).
- L'utilisateur exportera principalement les données dans un format différent (fichier CSV ou Excel). Il s'agit souvent d'un rapport tabulaire contenant de nombreuses lignes de données détaillées. Toutefois, Tableau ne permet pas d'exporter les données d'une vue ou d'un tableau de bord vers Excel, que ce soit à un niveau de détail ou de synthèse. Lorsque l'objectif principal est d'effectuer une exportation de données, il s'agit d'un processus d'extraction, de transformation et de chargement (ETL). Il existe pour cela d'autres solutions plus efficaces qu'un outil de reporting.
- Vous avez besoin de documents très complexes avec des tableaux croisés, qui ressemblent peut-être à des feuilles de calcul existantes comprenant des sous-totaux ou des références croisées complexes. Il peut s'agir de rapports financiers sur les pertes et profits, de bilans, etc. Vous avez peut-être également besoin de modéliser un scénario, d'effectuer des simulations ou de réinjecter des données d'hypothèses. Si les données granulaires sous-jacentes ne sont pas disponibles ou si la logique du rapport se base sur des cellules de référence plutôt que sur des récapitulatifs sous forme de totaux, il est peut-être préférable de continuer à utiliser une feuille de calcul pour les rapports de ce type.

Bien comprendre la notion d'efficacité

Qu'est-ce qu'un classeur « efficace » ?

Plusieurs facteurs font qu'un classeur est efficace. Certains sont techniques, d'autres sont plus du fait de l'utilisateur, mais en général, un classeur efficace est :

- **Simple** : le classeur est-il facile à créer et ensuite à gérer sur le long terme ? S'appuie-t-il sur les principes de l'analyse visuelle pour présenter clairement les données et le message de son auteur ?
- **Flexible** : le classeur permet-il de répondre à plusieurs questions ou à une seule ? Invite-t-il l'utilisateur à l'interaction ou s'agit-il simplement d'un rapport statique ?
- **Rapide** : le classeur réagit-il rapidement aux actions des utilisateurs ? Il peut s'agir notamment de la rapidité d'ouverture, d'actualisation ou de réponse aux interactions. Cette mesure est subjective, mais en général nous souhaitons qu'un classeur affiche les informations et réagisse aux interactions en quelques secondes seulement.

Divers facteurs, énumérés ci-dessous, ont un impact sur les performances d'un tableau de bord.

- La conception visuelle au niveau du tableau de bord et des feuilles de calcul, par exemple le nombre d'éléments et de points de données, l'utilisation des filtres et des actions, etc.
- Les calculs, notamment leur type, et où ils sont effectués
- Les requêtes, par exemple la quantité de données renvoyées ou l'utilisation d'expressions SQL personnalisées
- Les connexions de données et les données originales
- Certaines différences entre Tableau Desktop et Tableau Server
- D'autres facteurs, comme la configuration et la capacité du matériel

En quoi l'efficacité est-elle importante ?

Elle est importante pour différentes raisons.

- Réaliser des analyses ou créer des classeurs efficacement permet d'obtenir des réponses plus rapidement.
- Travailler efficacement vous permet de rester concentré sur l'analyse. Autrement dit, vous réfléchissez sur les questions et les données, et non sur la manière de manipuler l'outil pour obtenir les résultats souhaités.
- Concevoir un classeur modulable vous évite d'avoir à créer et gérer plusieurs classeurs pour répondre à des besoins similaires.
- Utiliser une conception simple permet aux utilisateurs d'exploiter votre classeur et de réaliser plus facilement des itérations à partir de vos découvertes.
- Concevoir des classeurs rapides améliore la satisfaction des utilisateurs, dans la mesure où leur perception de la réactivité des rapports ou tableaux de bord joue un rôle déterminant dans leur réussite.

Par expérience, nous avons pu constater que la plupart des problèmes de performances rencontrés par les clients sont dus à des erreurs dans la conception des classeurs. Si ces erreurs peuvent être corrigées, ou évitées grâce à de bons conseils, alors ces problèmes peuvent être résolus avec Tableau.

Si vous travaillez avec de petits volumes de données, la plupart de ces recommandations ne sont pas cruciales. Vous pouvez vous contenter de techniques qui ne sont pas optimisées. Néanmoins, si vous traitez des centaines de millions d'enregistrements et de nombreux classeurs, ou si plusieurs personnes

produisent des documents, les effets d'une mauvaise conception sont amplifiés. Vous devez alors prendre en considération les recommandations présentées dans ce livre blanc.

Bien entendu, ces conseils sont utiles pour tous vos classeurs, et la pratique vous aidera à vous perfectionner. Gardez à l'esprit que vous devez tester votre classeur dans des conditions de production réelles dans le cadre de sa conception.

Notez que même si nous faisons référence à Tableau Server dans ce document, la plupart de ces recommandations sont également valables pour Tableau Online si vous préférez une solution hébergée. Les seules exceptions concernent le réglage des paramètres et la configuration du serveur, et l'installation et la mise à jour du logiciel sur le serveur. Ces tâches incombent au prestataire de services dans un modèle SaaS.

Les lois de la physique

Avant de nous plonger dans les détails techniques pour décrire les effets des diverses fonctionnalités sur les performances des classeurs, voici quelques principes élémentaires qui vous permettront de créer des tableaux de bord et des rapports efficaces.

Si la source de données est lente, l'analyse dans Tableau le sera également

Si votre classeur Tableau est basé sur une requête qui s'exécute lentement, le classeur sera lent lui aussi. Dans les sections suivantes, nous allons vous donner des astuces pour améliorer vos bases de données et réduire le temps nécessaire à l'exécution de vos requêtes. Par ailleurs, nous expliquerons comment le moteur de données rapide de Tableau permet d'améliorer les performances des requêtes.

Si les performances sont lentes dans Tableau Desktop, elles le seront très certainement aussi dans Tableau Server

Un classeur lent dans Tableau Desktop ne sera pas plus rapide si vous le publiez sur Tableau Server. Les utilisateurs s'imaginent souvent que leur classeur sera plus rapide dans Tableau Server parce que le serveur a plus de CPU, de RAM, etc. que leur ordinateur. En fait, les classeurs sont souvent un peu plus lents dans Tableau Server, pour les raisons suivantes.

- De nombreux utilisateurs se partagent les ressources du serveur pour créer des classeurs simultanément, bien que vous puissiez paradoxalement constater que le classeur réagit plus rapidement lorsque vous le partagez, en raison des mécanismes de mise en cache de Tableau Server.
- Le rendu des tableaux de bord et des graphiques s'effectue sur le serveur, et non sur les ordinateurs clients.

Vous devez en premier lieu optimiser votre classeur dans Tableau Desktop avant de chercher à régler les performances dans Tableau Server.

La seule exception à cette règle concerne les situations où Tableau Desktop n'a pas assez de ressources, ce qui n'existe pas sur le serveur. C'est le cas par exemple si la RAM est insuffisante sur votre ordinateur pour gérer le volume de données que vous analysez, ou si le serveur dispose d'une connexion plus rapide ou d'une latence plus faible pour se connecter à la source de données. Certains utilisateurs subissent des performances dégradées ou même des problèmes de mémoire insuffisante lorsqu'ils utilisent un ensemble de données sur un ordinateur peu puissant, alors que les performances sont meilleures avec le classeur publié, car le serveur dispose de davantage de mémoire et de puissance de calcul.

Privilégiez les versions les plus récentes

Nos développeurs cherchent constamment à améliorer les performances et l'ergonomie de notre produit. L'utilisation de la version la plus récente de Tableau Desktop ou Tableau Server permet parfois d'améliorer nettement les performances, sans que vous ayez à modifier le classeur. Plusieurs clients ont signalé qu'entre la version 8 et la version 9, l'amélioration a été de 300 %, voire davantage. L'optimisation des performances reste au cœur de nos préoccupations pour Tableau 10 et pour les prochaines versions. Bien entendu, ce genre de problème disparaît si vous utilisez Tableau Online, puisque ce logiciel est systématiquement mis à jour.

C'est valable aussi bien pour les versions de maintenance que pour les versions majeures ou mineures. Pour connaître la liste de toutes les versions produites et accéder à des informations détaillées sur chacune d'elles, consultez les notes de version :

http://www.tableau.com/fr-fr/support/releases

Par ailleurs, dans le domaine des bases de données, les prestataires cherchent à améliorer la qualité de leur offre. Assurez-vous également que vous utilisez bien la version la plus récente du pilote de source de données en vérifiant sur cette page Web :

http://www.tableau.com/fr-fr/support/drivers

La sobriété est la clé de l'efficacité

Comme pour tout, l'excès peut être nuisible. N'essayez pas de tout mettre dans un classeur unique et monolithique. Bien qu'un classeur Tableau *puisse effectivement* contenir 50 tableaux de bord, chacun comportant 20 graphiques et communiquant avec 50 sources de données différentes, il sera vraisemblablement lent.

Il est préférable de le décomposer en plusieurs fichiers. Pour ce faire, il suffit de copier les tableaux de bord d'un classeur à l'autre, et Tableau se charge d'importer toutes les feuilles de calcul et sources de données associées. Si vos tableaux de bord sont extrêmement complexes, vous pouvez les simplifier et ajouter des interactions pour guider les utilisateurs d'un rapport à l'autre. Souvenez-vous que le logiciel n'est pas facturé à la page, alors n'hésitez pas à répartir vos données dans plusieurs documents.

L'extensibilité n'est pas synonyme de performances

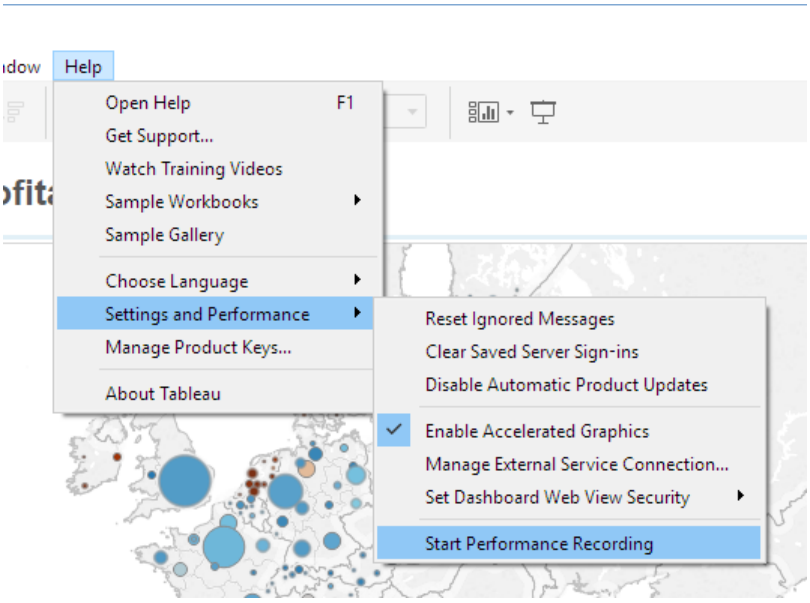
L'extensibilité garantit que nous pouvons prendre en charge de nombreux utilisateurs qui consultent des classeurs partagés. Les performances garantissent qu'un classeur donné fonctionnera aussi rapidement que possible. Bien que bon nombre des recommandations proposées ici soient utiles pour l'extensibilité des classeurs publiés sur Tableau Server, ce document est axé principalement sur l'amélioration des performances.

Une panoplie d'outils

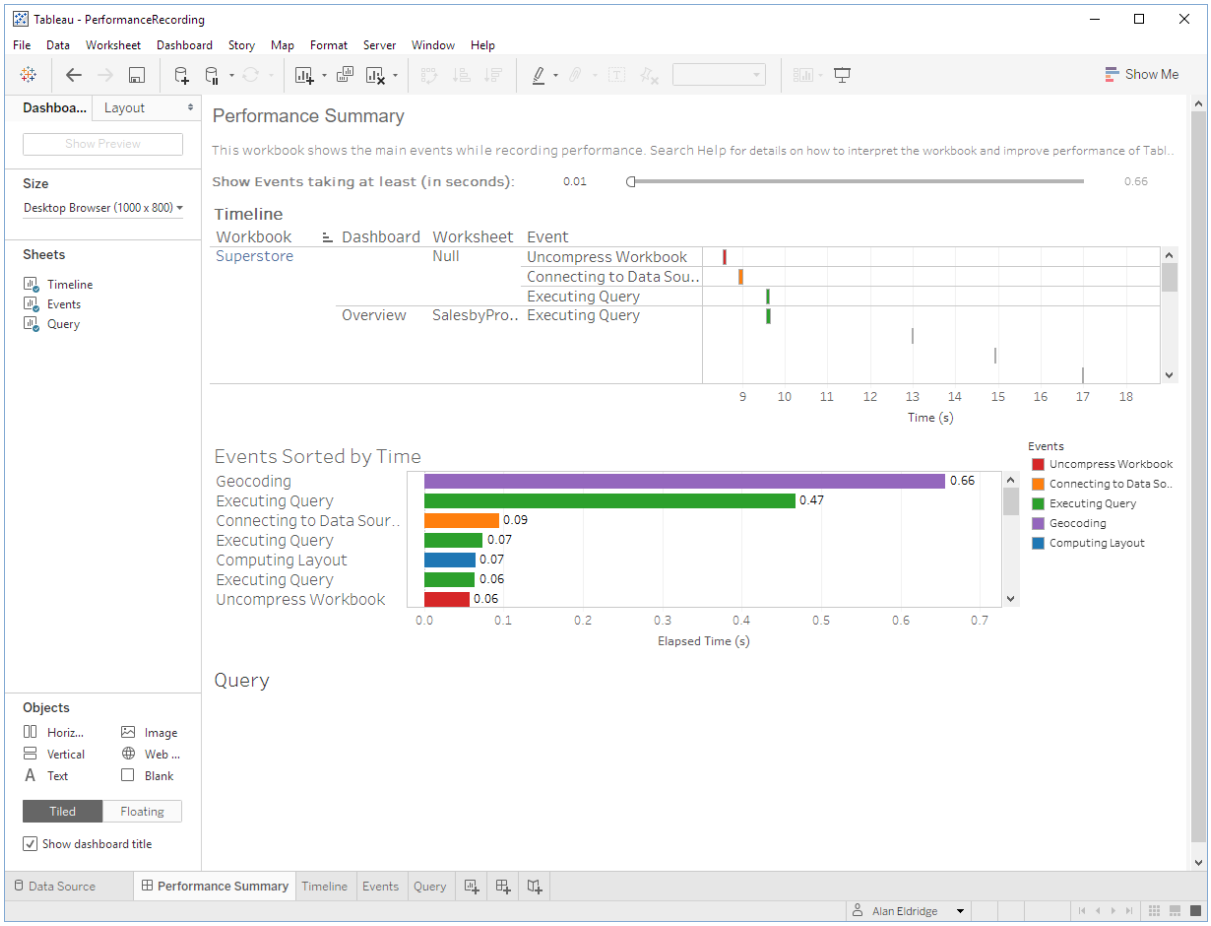
Pour bien comprendre les performances de vos classeurs, vous devez savoir ce qui se passe et combien de temps prend chaque opération. Ces informations proviennent de différents endroits selon l'emplacement où vous exécutez le classeur, par exemple Tableau Desktop ou Tableau Server, et avec des niveaux de détails différents. Passons en revue les différentes possibilités.

Exploitez l'enregistrement des performances

En premier lieu, pensez à consulter l'enregistrement des performances de Tableau Desktop ou Tableau Server pour obtenir des informations sur les performances. Dans Tableau Desktop, cette fonctionnalité est accessible depuis le menu Aide.



Lancez Tableau Desktop, démarrez l'enregistrement des performances, puis ouvrez votre classeur. Il est recommandé de ne pas ouvrir plusieurs classeurs pour éviter qu'ils utilisent les mêmes ressources. Interagissez comme le ferait un utilisateur final, et lorsque vous estimez avoir collecté suffisamment de données, revenez dans le menu Aide pour arrêter l'enregistrement. Une autre fenêtre de Tableau Desktop s'ouvre alors avec les données recueillies.



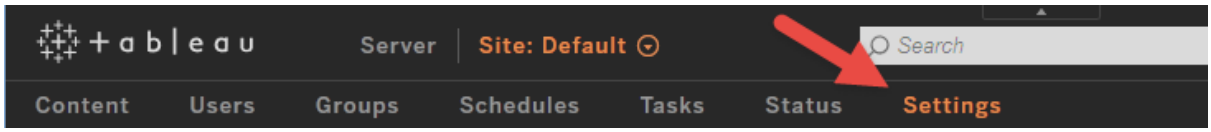
Vous pouvez maintenant identifier les actions du classeur qui prennent le plus de temps. Par exemple, dans l'image ci-dessus, la requête sélectionnée dans la feuille « Timeline » s'exécute en 30,66 secondes. Cliquez sur la barre pour afficher le texte de la requête exécutée. Comme les résultats de l'enregistrement des performances sont affichés dans un classeur Tableau, vous pouvez créer des vues supplémentaires pour explorer ces informations autrement.

> *Remarque : par défaut, les évènements qui prennent moins de 0,1 seconde ne sont pas affichés. Si vous voulez les inclure, modifiez le filtre en haut du tableau de bord. Il est toutefois préférable de se focaliser sur les tâches qui durent le plus longtemps. Régler le filtre sur 1 seconde est généralement la meilleure solution.*

Vous pouvez également créer des enregistrements des performances sur Tableau Server pour identifier les problèmes survenant après la publication du classeur. Par défaut, l'enregistrement des performances n'est pas activé sur Tableau Server. Cette fonctionnalité peut être contrôlée pour chaque site.

Un administrateur peut choisir de l'activer site par site.

1. Accédez au site pour lequel vous souhaitez enregistrer les performances.
2. Cliquez sur **Paramètres** :



3. Sous Mesures de performance du classeur, sélectionnez **Enregistrer les mesures de performance du classeur**.

4. Cliquez sur **Enregistrer**.

Lorsque vous souhaitez créer un enregistrement des performances :

1. Ouvrez la vue pour laquelle vous souhaitez enregistrer les performances. Lorsque vous ouvrez une vue, Tableau Server ajoute :iid=<*n*> à la fin de l'URL. Il s'agit d'un ID de session. Par exemple :

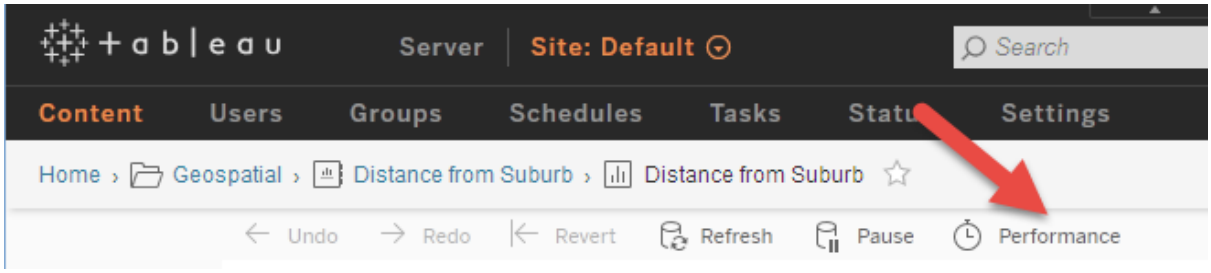
   ```
   http://<tableau_server>/#/views/Coffee_Sales2013/USSalesMarginsByAreaCode?:iid=1
   ```

2. Ajoutez `:record_performance=yes` après l'URL de la vue, juste avant l'ID de session. Par exemple :

   ```
   http://<tableau_server>/#/views/Coffee_Sales2013/USSalesMarginsByAreaCode?:record_performance=yes&:iid=1
   ```

3. Chargez la vue.

L'option **Performances** affichée dans la barre d'outils confirme que l'enregistrement des performances a commencé.



Une fois l'enregistrement des performances terminé, vous pouvez accéder aux résultats :

1. Cliquez sur **Performances** pour ouvrir le classeur des performances. Il s'agit d'un instantané des données de performance à la minute près. Vous pouvez continuer à prendre des instantanés pendant que vous travaillez sur la vue. Les données recueillies se cumulent.

2. Pour arrêter l'enregistrement, passez à une autre page ou supprimez `:record_performance=yes` de l'URL.

Ces informations permettent d'identifier les sections d'un classeur qui méritent le plus d'être révisées et améliorées, pour réduire le temps passé. Rendez-vous sur la page suivante pour en savoir plus sur l'interprétation de ces enregistrements :

> http://onlinehelp.tableau.com/current/pro/desktop/fr-fr/help.htm#perf_record_create_desktop.html

Les journaux

Dans Tableau, vous pouvez trouver le texte complet des requêtes dans le fichier journal. Par défaut, il se trouve ici : `C:\Users\<username>\Documents\My Tableau Repository\Logs\log.txt`
Ce fichier relativement prolixe est au format JSON. Il est conseillé de l'ouvrir avec un bon éditeur de texte comme Notepad++ ou Sublime. Recherchez les chaînes "begin-query" ou "end-query" pour

accéder au texte de la requête envoyée à la source de données. L'enregistrement "end-query" vous donne également des informations sur le temps nécessaire à l'exécution
de la requête et sur le nombre d'enregistrements renvoyés à Tableau :


```
{"ts":"2015-05-
24T12:25:41.226","pid":6460,"tid":"1674","sev":"info","req":"-","sess":"-
","site":"-","user":"-","k":"end-query",
"v":{"protocol":"4308fb0","cols":4,"query":"SELECT
[DimProductCategory].[ProductCategoryName] AS
[none:ProductCategoryName:nk],\n 14

[DimProductSubcategory].[ProductSubcategoryName] AS
[none:ProductSubcategoryName:nk],\n SUM(CAST(([FactSales].[ReturnQuantity])
as BIGINT)) AS [sum:ReturnQuantity:ok],\n SUM([FactSales].[SalesAmount]) AS
[sum:SalesAmount:ok]\nFROM [dbo].[FactSales] [FactSales]\n INNER JOIN
[dbo].[DimProduct] [DimProduct] ON ([FactSales].[ProductKey] =
[DimProduct].[ProductKey])\n INNER JOIN [dbo].[DimProductSubcategory]
[DimProductSubcategory] ON ([DimProduct].[ProductSubcategoryKey] =
[DimProductSubcategory].[ProductSubcategoryKey])\n INNER JOIN
[dbo].[DimProductCategory] [DimProductCategory] ON
([DimProductSubcategory].[ProductCategoryKey] =
[DimProductCategory].[ProductCategoryKey])\nGROUP BY
[DimProductCategory].[ProductCategoryName],\n
[DimProductSubcategory].[ProductSubcategoryName]","rows":32,"elapsed":0.951}
```


Pour Tableau Server, les journaux se trouvent dans le dossier
`C:\ProgramData\Tableau\TableauServer\data\tabsvc\vizqlserver\Logs`

Les vues des performances de Tableau Server

Dans Tableau Server, plusieurs vues permettent aux administrateurs de surveiller l'activité sur le serveur. Ces vues se trouvent dans la table Analyses de la section Maintenance du serveur.

Analysis

Dashboards that monitor Tableau Server activity.

Dashboard	Analysis
Traffic to Sheets	Usage and users for published sheets.
Traffic to Data Sources	Usage and users for published data sources.
Actions by All Users	Actions for all users.
Actions by Specific User	Actions for a specific user, including items used.
Actions by Recent Users	Recent actions by users, including last action time and idle time.
Background Tasks for Extracts	Completed and pending extract task details.
Background Tasks for Non Extracts	Completed and pending background task details (non-extract).
Stats for Load Times	Sheet load times and performance history.
Stats for Space Usage	Space used by published workbooks and data sources, including extracts and live connections.
Server Disk Space	Current and historical disk space usage, by server node.
Tableau Desktop License Usage	Summary of usage for Tableau Desktop licenses
Tableau Desktop License Expirations	Expiration information for Tableau Desktop licenses

Pour en savoir plus sur ces vues, rendez-vous sur cette page :

http://onlinehelp.tableau.com/current/server/fr-fr/adminview.htm

Par ailleurs, vous pouvez également créer des vues administratives personnalisées en vous connectant à la base de données PostgreSQL, stockée dans le référentiel Tableau. Rendez-vous sur cette page pour en savoir plus :

http://onlinehelp.tableau.com/current/server/fr-fr/adminview_postgres.htm

Surveillance et tests

TabMon

TabMon est un moniteur de cluster open source pour Tableau Server qui permet de collecter des statistiques sur les performances au fil du temps. TabMon bénéficie du soutien de la communauté d'utilisateurs, et nous publions le code source complet sous licence MIT.

TabMon est prêt à l'emploi et enregistre des informations sur l'état du système et les applications. Il collecte des mesures intégrées comme Windows Perfmon, Java Health et Java Mbean (JMX) sur les machines Tableau Server du réseau. Vous pouvez l'utiliser pour surveiller l'utilisation des ressources physiques (CPU, RAM), du réseau et du disque dur. Vous pouvez suivre le ratio des données lues en mémoire par rapport aux données trouvées sur le disque, la latence des requêtes, les sessions actives et bien plus encore. Cet outil affiche les données dans un format propre et structuré, ce qui permet de les visualiser facilement dans Tableau Desktop.

TabMon vous donne le contrôle total sur les mesures à collecter et les machines à surveiller, sans utiliser de scripts ni recourir au codage. Il suffit de connaître le nom de la machine et de la mesure. TabMon peut s'exécuter à distance et indépendamment de votre cluster. Vous pouvez surveiller, agréger et analyser les données sur l'état de vos clusters à partir de n'importe quel ordinateur se trouvant sur le réseau. De plus, cela n'a presque pas d'impact sur la charge de travail de vos machines de production.

Pour en savoir plus sur TabMon, reportez-vous à la page suivante :

http://bit.ly/1ULFelf

TabJolt

TabJolt est un outil de test de charge et de performances conçu pour fonctionner facilement avec Tableau Server. À l'inverse des outils de test de charge traditionnels, TabJolt peut automatiquement gérer la charge pour Tableau Server, sans nécessiter de développement ou de maintenance de scripts. Comme TabJolt est capable de reconnaître le modèle de présentation de Tableau, il peut automatiquement charger des visualisations et interpréter les interactions possibles pendant l'exécution des tests.

Ainsi, tout ce que vous avez à faire c'est de désigner un ou plusieurs classeurs stockés sur votre serveur. TabJolt les charge automatiquement et exécute des interactions sur les vues Tableau. TabJolt collecte également des mesures clés comme le temps de réponse moyen, le débit et la valeur du 95e centile, ainsi que des mesures de performance Windows pour établir des corrélations.

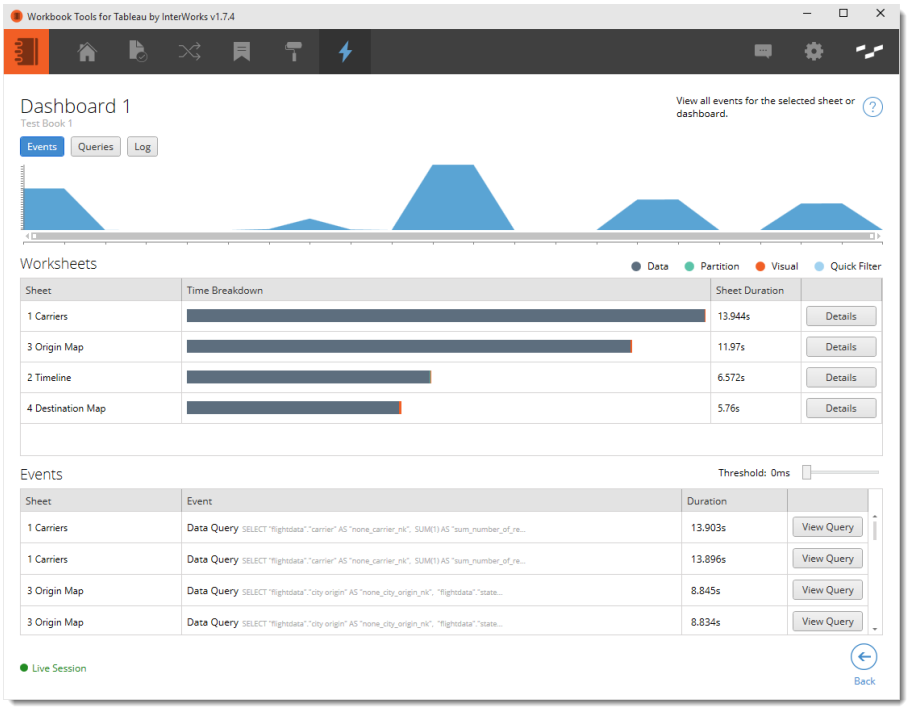
Bien entendu, même en utilisant TabJolt, les utilisateurs doivent connaître suffisamment l'architecture de Tableau Server. Considérer Tableau Server comme une boîte noire pour vos tests de charge n'est pas une bonne idée, car vous risquez d'obtenir des résultats qui ne correspondent pas à vos attentes. TabJolt est un outil automatisé qui ne peut facilement reproduire toute la panoplie des interactions humaines. Vous devez donc déterminer si les résultats que vous obtenez sont conformes à la réalité.

Pour en savoir plus sur TabJolt, reportez-vous à la page suivante :

http://bit.ly/1ULFtgi

Autres outils

Il existe d'autres outils tiers pour vous aider à identifier les caractéristiques de performance de vos classeurs. Power Tools for Tableau d'Interworks inclut notamment un outil d'analyse des performances, similaire à notre fonction d'enregistrement, qui vous permet d'explorer les informations et d'identifier les feuilles et les requêtes qui prennent le plus de temps.



Palette Insight, un produit de Palette Software, capture dans Tableau Server les informations sur les performances et vous permet de planifier la capacité, d'identifier les utilisateurs et les classeurs qui ont le plus besoin de ressources, de contrôler les accès des utilisateurs et de créer des modèles de répartition des coûts.

Par ailleurs, la plupart des plates-formes de gestion de bases de données modernes incluent des outils d'administration permettant de suivre et d'analyser les requêtes qui s'exécutent. L'administrateur de base de données peut vous aider si l'enregistrement de vos performances indique des temps d'exécution de requêtes problématiques.

Si vous pensez que l'interaction entre le serveur et le navigateur client pose problème, vous pouvez également utiliser Telerik Fiddler ou les outils de développeur de votre navigateur pour analyser plus en détail le trafic entre le serveur et le client.

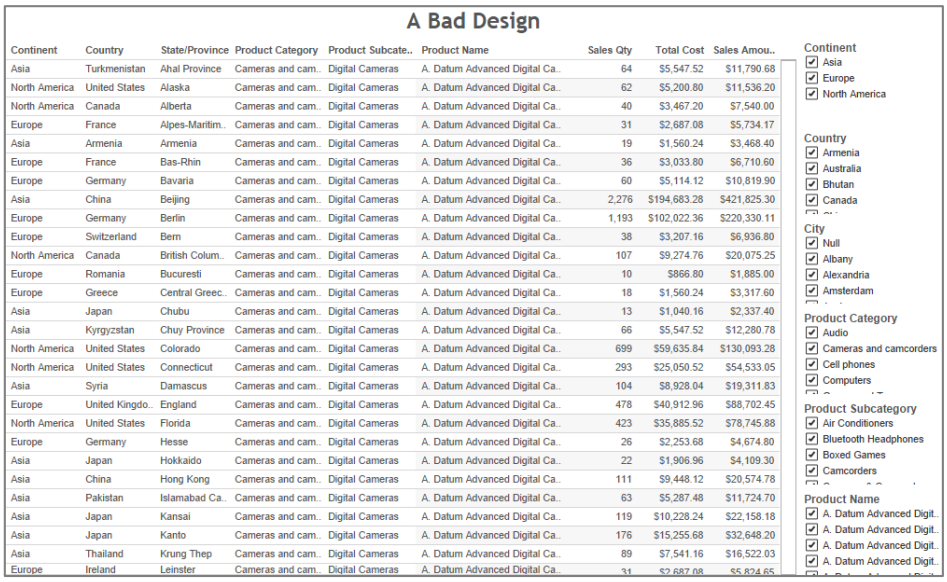
La conception du classeur est-elle en cause ?

De nombreux utilisateurs découvrent Tableau et doivent apprendre les techniques et les meilleures pratiques qui permettent de créer des classeurs efficaces. Nous constatons néanmoins que beaucoup tentent d'appliquer des approches obsolètes dans Tableau et obtiennent par conséquent des résultats peu satisfaisants. Dans cette section, nous allons aborder certains principes de conception à mettre en pratique.

La bonne manière de créer un tableau de bord

Grâce à Tableau, vous offrez l'interactivité à vos utilisateurs finaux. Le résultat produit par Tableau Server est une application interactive qui permet aux utilisateurs d'explorer les données, et pas simplement de les observer. Vous ne créez pas un tableau de bord efficace en adoptant la même méthode que pour un rapport statique.

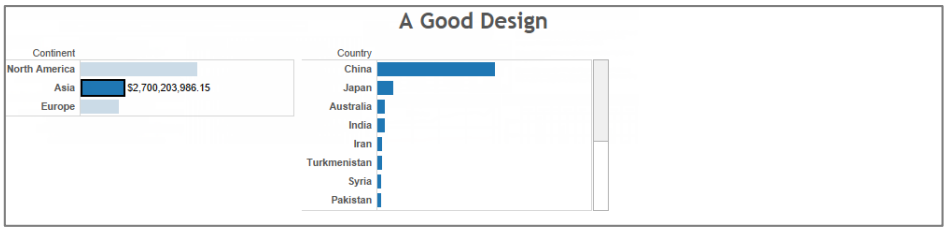
Voyez par exemple le type de tableau de bord créé par de nombreux nouveaux utilisateurs, en particulier s'ils travaillaient auparavant avec des outils tels qu'Access ou Excel, ou s'ils sont habitués aux outils de reporting traditionnels. Au départ, un rapport tabulaire montre l'ensemble des données et une série de filtres permet d'affiner le tableau jusqu'à ce qu'il affiche les enregistrements souhaités.



Ce n'est pas ce que nous pouvons appeler un tableau de bord Tableau bien conçu. En réalité, il ne s'agit pas d'un bon tableau de bord, tout simplement. Il s'agit au pire d'une simple extraction de données, car l'utilisateur souhaite exporter ces données vers un outil comme Excel pour les analyser et créer des graphiques. Au mieux, ce tableau de bord démontre que son auteur ne comprend pas vraiment comment l'utilisateur final souhaite explorer les données, alors il choisit de montrer l'ensemble des données et propose quelques filtres pour réduire la vue et trouver les données souhaitées.

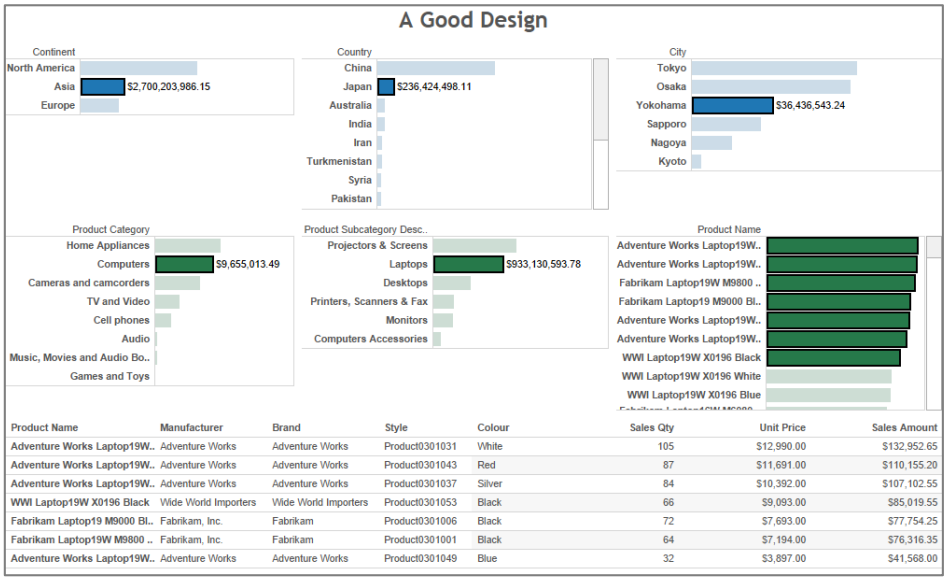
Observons maintenant cette nouvelle version, avec exactement les mêmes données. Nous partons du plus haut niveau d'agrégation.

En sélectionnant un ou plusieurs éléments, nous accédons au niveau de détail suivant.



Nous pouvons continuer ainsi et afficher de plus en plus de détails.

Nous atteignons finalement le dernier niveau, qui présente les mêmes données que dans le tableau croisé du début.



Product Name	Manufacturer	Brand	Style	Colour	Sales Qty	Unit Price	Sales Amount
Adventure Works Laptop19W..	Adventure Works	Adventure Works	Product0301031	White	105	$12,990.00	$132,952.65
Adventure Works Laptop19W..	Adventure Works	Adventure Works	Product0301043	Red	87	$11,691.00	$110,155.20
Adventure Works Laptop19W..	Adventure Works	Adventure Works	Product0301037	Silver	84	$10,392.00	$107,102.55
WWI Laptop19W X0196 Black	Wide World Importers	Wide World Importers	Product0301053	Black	66	$9,093.00	$85,019.55
Fabrikam Laptop19 M9000 Bl..	Fabrikam, Inc.	Fabrikam	Product0301006	Black	72	$7,693.00	$77,754.25
Fabrikam Laptop19W M9800 ..	Fabrikam, Inc.	Fabrikam	Product0301001	Black	64	$7,194.00	$76,316.35
Adventure Works Laptop19W..	Adventure Works	Adventure Works	Product0301049	Blue	32	$3,897.00	$41,568.00

Ne vous focalisez pas sur la présentation des données. Nous y reviendrons plus tard. Réfléchissez plutôt à l'expérience utilisateur que procure ce tableau de bord. Vous pouvez constater qu'il suit une progression naturelle, de gauche à droite et de haut en bas. Les données sous-jacentes peuvent être nombreuses, mais le tableau de bord guide l'utilisateur pour lui faire découvrir progressivement les détails, jusqu'à ce qu'il trouve ce qu'il cherche.

La différence essentielle entre ces deux exemples réside dans la manière d'orienter l'utilisateur dans le processus analytique. Le premier exemple part d'une vue générale, avec tous les enregistrements

possibles, et pousse l'utilisateur à réduire l'affichage en appliquant des filtres. Cette technique est problématique.

- En effet, la première requête à exécuter avant de présenter quoi que ce soit à l'utilisateur consiste essentiellement à tout afficher. L'exécution d'une telle requête et le renvoi des informations vers le moteur Tableau prennent énormément de temps. Pour l'expérience client, le « premier contact » est déterminant dans la perception de la solution. S'il faut plusieurs secondes avant qu'une action produise un résultat, la perception sera négative.
- La création d'une vue contenant des centaines de milliers, voire des millions, de repères (ce sont les cellules dans un tableau croisé) nécessite une puissance de calcul et une quantité de mémoire importantes. Par ailleurs, cela prend du temps, et le système est de plus en plus perçu comme peu réactif. Dans Tableau Server, le fait que plusieurs utilisateurs génèrent tous des tableaux croisés volumineux entraîne un ralentissement des performances, et le système peut même manquer de mémoire. Cela peut se traduire par des problèmes de stabilité, des erreurs et une expérience médiocre pour les utilisateurs finaux. Bien évidemment, vous pouvez simplement ajouter de la mémoire au serveur, mais il serait préférable de s'attaquer à la cause du problème plutôt qu'aux symptômes.
- Enfin, rien n'indique visuellement dans le contexte si l'ensemble de filtres initial est trop vaste ou trop réduit. Comment un utilisateur peut-il savoir que s'il sélectionne toutes les catégories disponibles, la requête renverra des dizaines de milliers d'enregistrements et consommera toute la RAM disponible sur le serveur ? Il ne peut pas. Seule une mauvaise expérience peut le renseigner.

À l'inverse, dans la seconde approche, la requête initiale présente uniquement le plus haut niveau d'agrégation.

- La requête initiale à exécuter est très agrégée et renvoie par conséquent uniquement un nombre réduit d'enregistrements. C'est très efficace pour une base de données bien conçue. Comme les délais de réponse sont réduits, l'utilisateur a une perception positive du système lors du premier contact. Au fil de l'exploration des données, chaque requête suivante est à la fois agrégée et limitée du fait des sélections effectuées au niveau le plus élevé. L'exécution reste rapide, de même que le renvoi des résultats au moteur Tableau.
- Même si nous nous retrouvons avec plusieurs vues une fois le tableau de bord entièrement terminé, chacune d'elles contient seulement quelques dizaines de repères. Les ressources nécessaires pour générer chacune de ces vues, même lorsque plusieurs utilisateurs sont actifs sur le système, restent dérisoires et le système risque beaucoup moins d'être à court de mémoire.
- Enfin, vous pouvez constater que pour les niveaux de « navigation » supérieurs, nous avons choisi d'afficher le volume des ventes de chaque catégorie. Avec un peu de contexte, l'utilisateur peut savoir si la sélection en question contient peu d'enregistrements ou beaucoup. Nous avons également utilisé de la couleur pour indiquer la rentabilité de chaque catégorie. C'est très utile, car cela permet de déterminer les domaines qui nécessitent une attention particulière, au lieu de naviguer en aveugle.

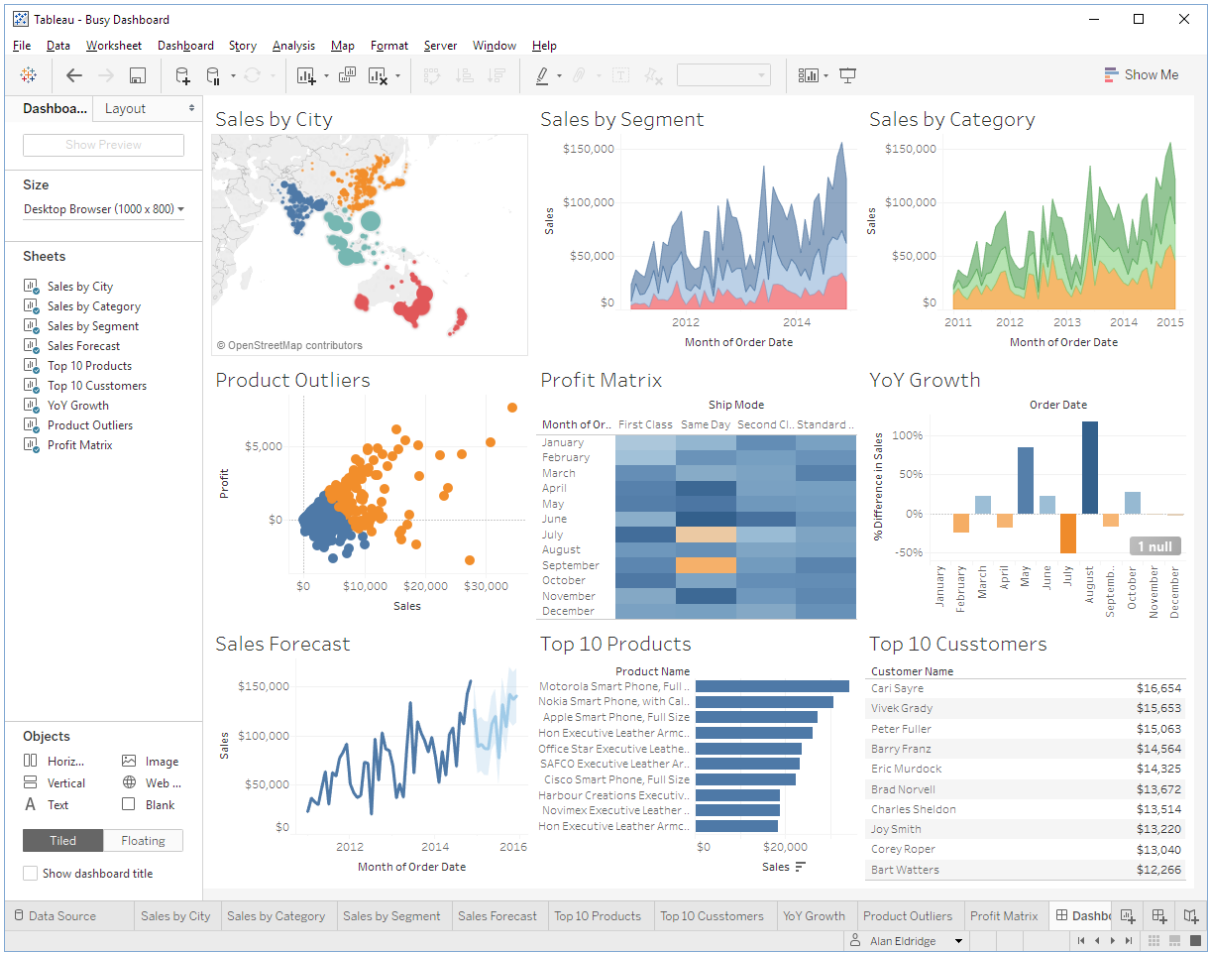
Privilégiez la simplicité

Souvent, les débutants font l'erreur de créer des tableaux de bord trop complexes. Ils cherchent peut-être à recréer un document qu'ils ont utilisé dans un outil différent ou à créer quelque chose qui soit conçu spécialement pour un rapport imprimé. Au final, ils obtiennent un classeur lent et peu efficace.

Voyons ce qui contribue à cette complexité excessive.

Trop de feuilles dans chaque tableau de bord

Les nouveaux utilisateurs commettent souvent l'erreur de tenter de placer le maximum de graphiques et/ou de feuilles dans le même tableau de bord.



Sachez que chaque feuille va exécuter une ou plusieurs requêtes sur les sources de données. Par conséquent, plus le nombre de feuilles est élevé, plus le rendu du tableau de bord est long. Tirez parti du fait que Tableau a été conçu pour proposer des tableaux de bord interactifs aux utilisateurs finaux et répartissez vos données sur plusieurs tableaux de bord ou pages.

Trop de filtres

Les filtres affichés sur les vues sont puissants, car ils permettent de proposer aux utilisateurs finaux des tableaux de bord riches et interactifs. Néanmoins, chaque filtre peut nécessiter une requête pour afficher les différentes options. Par conséquent, s'ils sont trop nombreux, le rendu du tableau de bord sera plus lent. Par ailleurs, lorsque vous décidez d'afficher les valeurs pertinentes uniquement, une requête doit mettre à jour les valeurs affichées chaque fois que les autres filtres changent. Utilisez cette fonctionnalité avec parcimonie.

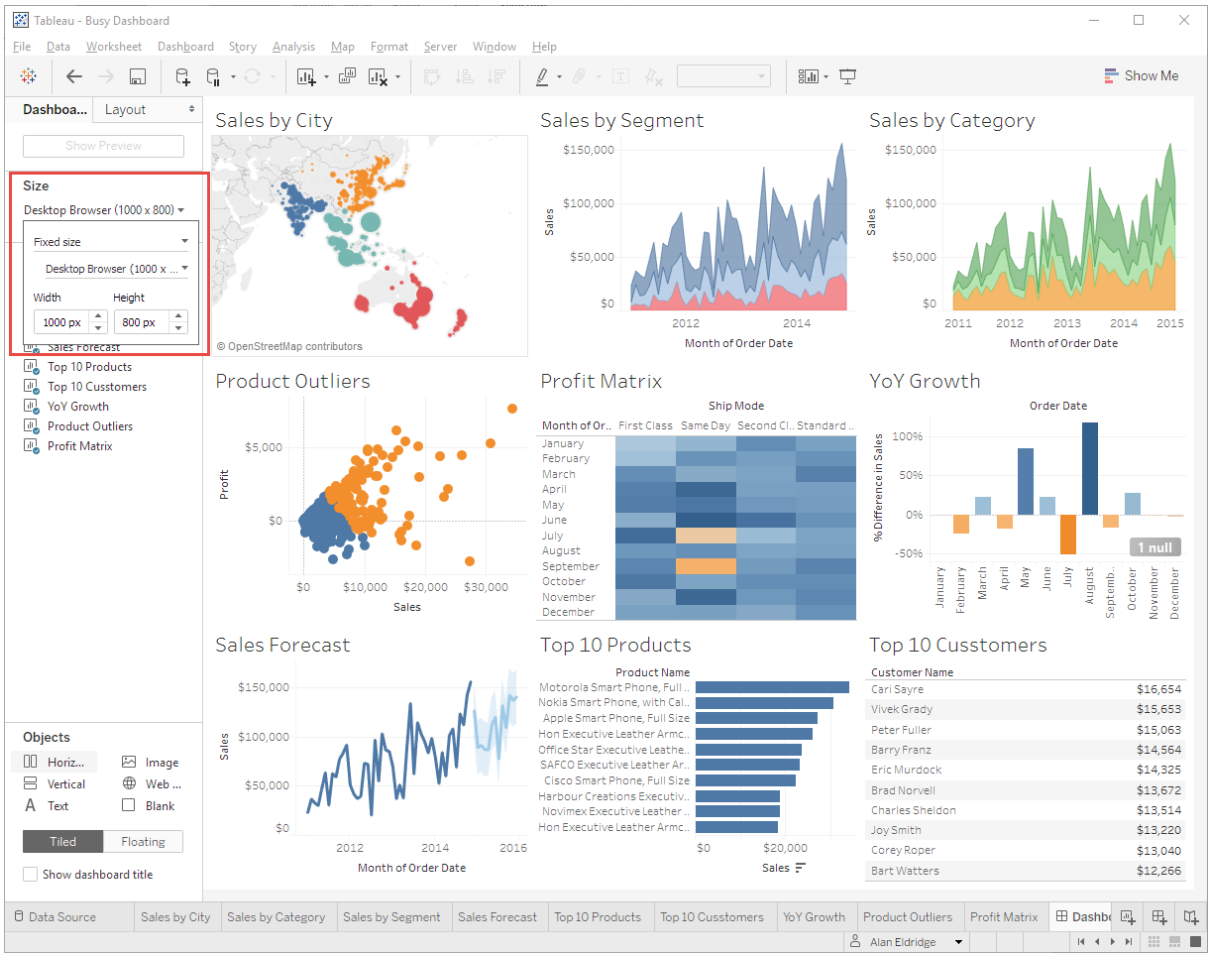
Si des filtres s'appliquent à plusieurs feuilles de calcul, gardez à l'esprit que chaque modification déclenchera plusieurs requêtes, car toutes les feuilles visibles seront mises à jour. Celles qui ne sont pas visibles ne sont pas concernées. Si cette opération nécessite plusieurs secondes, l'expérience utilisateur peut s'en ressentir. Si vous prévoyez que les utilisateurs changeront plusieurs fois un type de filtre à sélection multiple, vous pouvez proposer un bouton « Appliquer ». Ainsi, ils pourront déclencher l'actualisation une fois toutes les sélections effectuées.

Quelques ajustements pour optimiser les performances

Une fois que vous êtes sûr que votre tableau de bord est aussi simple que possible, vous pouvez rectifier la conception et exploiter la mise en cache pour optimiser les performances.

Utilisez des tableaux de bord à taille fixe

Pour améliorer les performances, le plus facile est de vérifier que votre tableau de bord a une taille fixe.



Une partie du processus de rendu dans Tableau consiste à créer une disposition : le nombre de lignes et de colonnes à afficher pour les séries de petits graphiques et les tableaux croisés, le nombre de graduations, les intervalles et le nombre de lignes à tracer pour les axes, ou encore le nombre d'étiquettes de repère et leur emplacement. C'est la taille de la fenêtre dans laquelle le tableau de bord s'affichera qui détermine cette disposition.

Si le même tableau de bord doit être affiché sur des fenêtres de différentes tailles, il faudrait créer une disposition pour chacune d'elles. En choisissant une taille fixe pour le tableau de bord, il suffit de créer une seule disposition qui sera réutilisée pour tous les cas. Cette pratique est d'autant plus importante pour le rendu côté serveur qu'elle permet de mettre en cache et partager les bitmaps dont le rendu est effectué sur le serveur, ce qui améliore à la fois les performances et l'extensibilité.

Des tableaux de bord adaptés aux différents appareils

Dans Tableau 10, une nouvelle fonctionnalité permet de créer des tableaux de bord propres à chaque type d'appareil. Vous pouvez ainsi créer des dispositions personnalisées qui sont sélectionnées automatiquement en fonction de l'appareil utilisé.

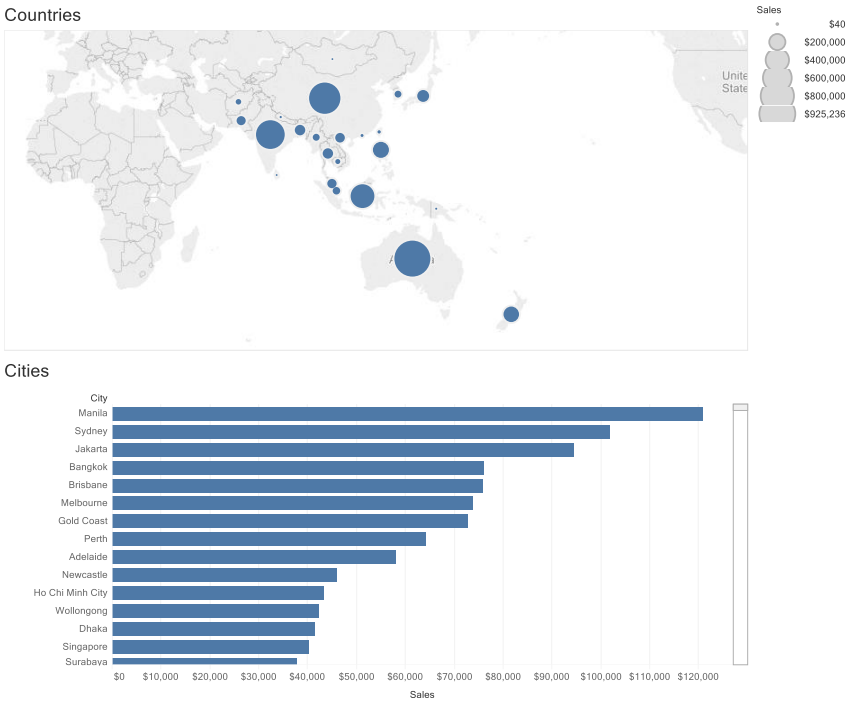
La disposition à utiliser est sélectionnée en fonction de la taille de l'écran :

- Téléphone : <= 500 pixels en largeur
- Tablette : <= 800 pixels en largeur
- Ordinateur de bureau : > 800 pixels

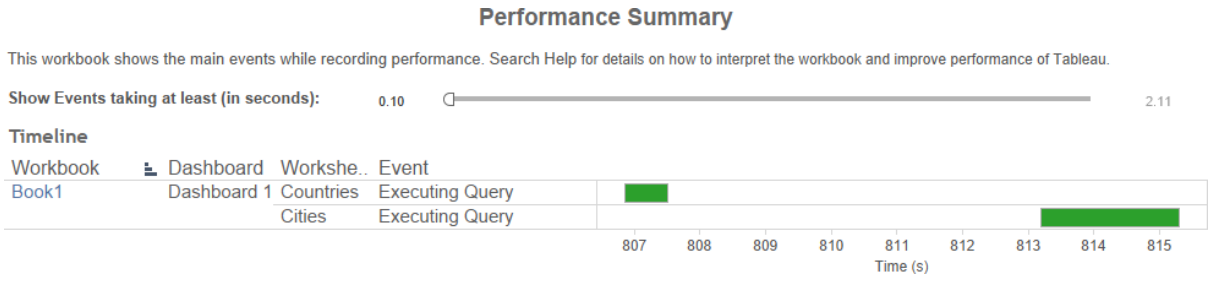
Étant donné que les différents appareils ont des tailles d'écran différentes dans chacune de ces catégories, et que l'affichage peut pivoter, il est préférable de configurer un redimensionnement automatique des dispositions pour téléphone et tablette. Cela permet d'optimiser la consultation, mais il y aura un impact sur la réutilisation des caches (celui des modèles de présentation et celui de la juxtaposition des images pour le rendu côté serveur). En général, les avantages d'un redimensionnement adapté à chaque appareil éclipsent l'impact sur la mise en cache, mais cela vaut la peine d'y réfléchir. Dès que les utilisateurs commenceront à utiliser le classeur, vous finirez par remplir tous les modèles et bitmaps destinés aux résolutions les plus communes, ce qui se traduira par une amélioration des performances.

Tirez profit du niveau de détail de la visualisation pour réduire le nombre de requêtes

Bien qu'il soit généralement préférable d'utiliser uniquement les champs dont vous avez besoin pour chaque feuille, vous pouvez parfois améliorer les performances en mettant davantage d'informations sur une feuille particulière pour éviter de générer des requêtes sur les autres. Observez le tableau de bord suivant.



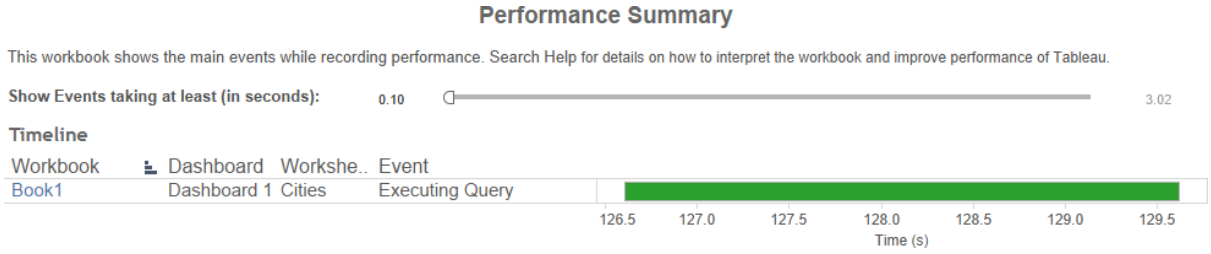
S'il est conçu comme prévu, le plan d'exécution produira deux requêtes, une pour chaque feuille :



```
SELECT [Superstore APAC].[City] AS [City],
  SUM([Superstore APAC].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[Superstore APAC] [Superstore APAC]
GROUP BY [Superstore APAC].[City]

SELECT [Superstore APAC].[Country] AS [Country],
  SUM([Superstore APAC].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[Superstore APAC] [Superstore APAC]
GROUP BY [Superstore APAC].[Country]
```

Si vous modifiez la conception du tableau de bord et ajoutez le champ Pays à la feuille des villes (sur l'étagère Détail), Tableau peut renseigner le tableau de bord avec une seule requête. Le logiciel est capable d'exécuter d'abord la requête pour la feuille Cities, puis d'utiliser le cache des résultats de cette requête pour fournir les données destinées à la feuille Countries. C'est ce que nous appelons le regroupement de requêtes.

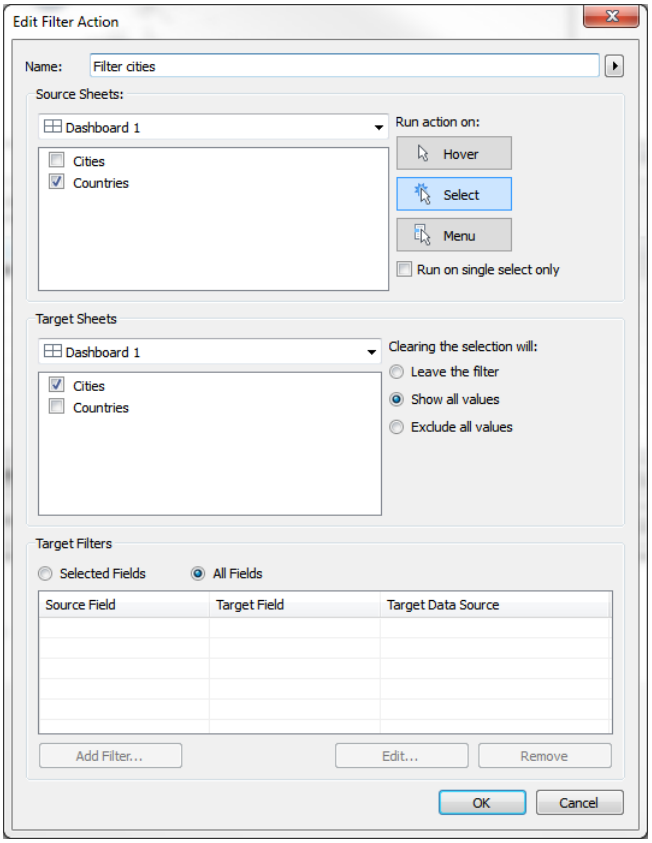



```
SELECT [Superstore APAC].[City] AS [City],
  [Superstore APAC].[Country] AS [Country],
  SUM([Superstore APAC].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[Superstore APAC] [Superstore APAC]
GROUP BY [Superstore APAC].[City],
  [Superstore APAC].[Country]
```

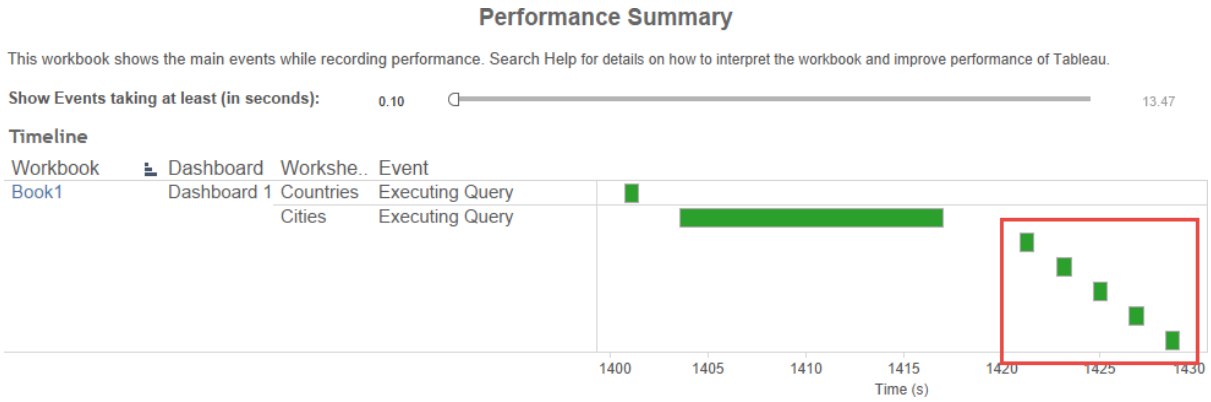
Il est clair que cela ne peut pas être effectué dans tous les cas, car l'ajout d'une dimension à la visualisation modifie le niveau de détail, ce qui peut entraîner l'affichage d'un nombre plus important de repères. Néanmoins, lorsque vos données comportent une relation hiérarchique comme dans l'exemple ci-dessus, cette technique est utile car elle ne change pas le niveau de détail visible.

Utilisation du niveau de détail de la visualisation pour optimiser les actions

Vous pouvez utiliser une approche similaire avec les actions pour réduire le nombre de requêtes à exécuter. Reprenons le même tableau de bord que ci-dessus (avant optimisation), mais ajoutons cette fois une action de filtre, de la feuille Countries vers la feuille Cities.



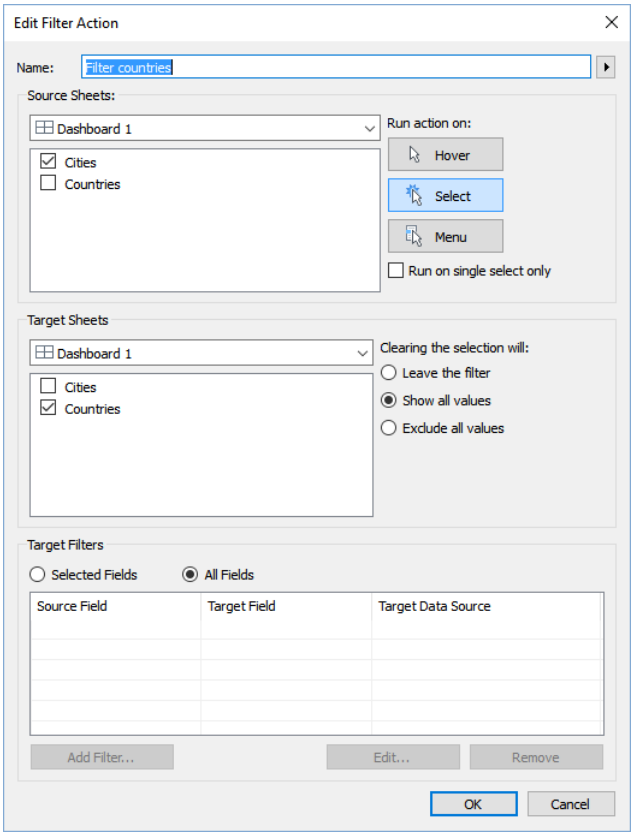
Lorsque vous déclenchez cette action en cliquant sur un repère de carte, vous pouvez voir que Tableau doit exécuter une requête pour déterminer les valeurs de la feuille Cities, car le cache des résultats de requête ne comporte pas de données sur la relation ville-pays.




```
SELECT [Superstore APAC].[City] AS [City],
  SUM([Superstore APAC].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[Superstore APAC] [Superstore APAC]
WHERE ([Superstore APAC].[Country] = 'Australia')
GROUP BY [Superstore APAC].[City]
```

Si vous ajoutez le champ Country à la feuille Cities, vous avez alors suffisamment d'informations dans le cache des résultats de requête pour utiliser ces filtres sans revenir à la source de données.

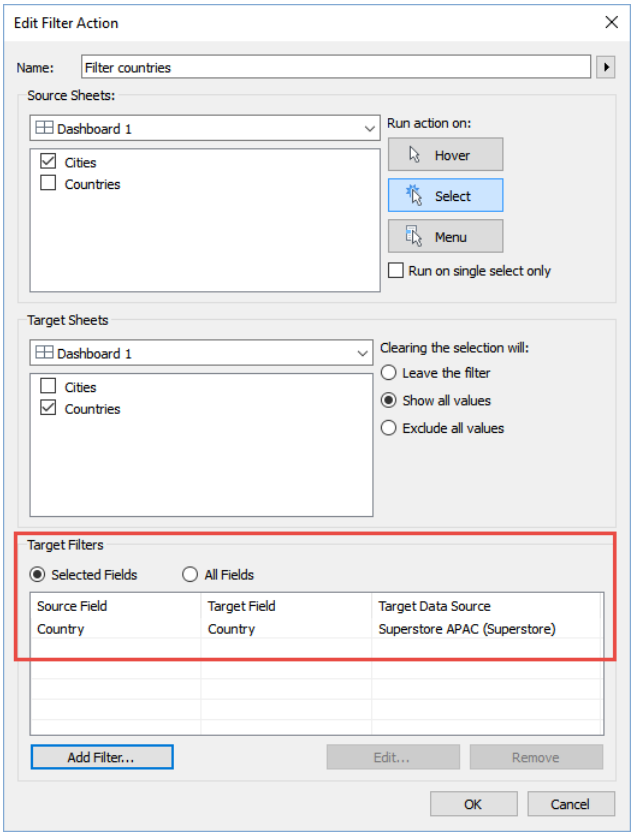
Vous pouvez effectuer une optimisation similaire lorsque la feuille source est plus détaillée que la feuille cible. Prenons la définition de l'action par défaut, où le filtrage est fait avec l'option Tous les champs :



Le classeur exécute une requête pour chaque action, car la clause du filtre fait référence au pays et à la ville, et ces informations ne peuvent être obtenues depuis le cache des résultats de requête pour la feuille Countries.

```
SELECT [Superstore APAC].[Country] AS [Country],
  SUM([Superstore APAC].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[Superstore APAC] [Superstore APAC]
WHERE (([Superstore APAC].[City] = 'Sydney') AND ([Superstore APAC].[Country] =
'Australia'))
GROUP BY [Superstore APAC].[Country]
```

Si vous changez l'action pour filtrer uniquement d'après le pays :



Vous pouvez désormais obtenir les informations nécessaires dans le cache des résultats de requête. Il n'est donc pas nécessaire de renvoyer une requête vers la source de données. Comme dans l'exemple précédent, vous devez prendre en compte l'impact d'un changement du niveau de détail sur la conception de votre feuille de calcul. S'il n'y a aucun impact, cette technique peut être utile.

La bonne manière de créer une feuille de calcul

Les feuilles de calcul se trouvent juste en dessous des tableaux de bord dans la hiérarchie. Dans Tableau, la conception des feuilles de calcul est intrinsèquement liée aux requêtes exécutées. Chaque feuille génère une ou plusieurs requêtes. Il faut donc s'assurer que ces dernières sont optimisées.

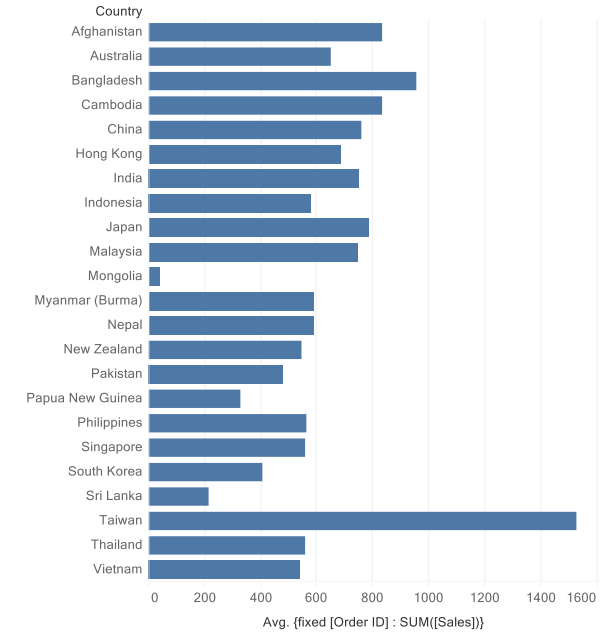
Incluez uniquement les champs dont vous avez besoin

Supprimez de l'étagère Détail tous les champs qui ne sont pas directement utilisés dans la visualisation, pas nécessaires dans l'infobulle ou pas requis pour accéder au niveau de détail de repère souhaité. Cela permet d'accélérer la requête sur la source de données et de renvoyer moins de données dans les résultats. Il existe quelques exceptions à cette règle, en effet le regroupement de requêtes évite d'en utiliser de similaires dans les autres feuilles de calcul, mais elles sont plutôt rares et impliquent de ne pas modifier le niveau de détail de la feuille.

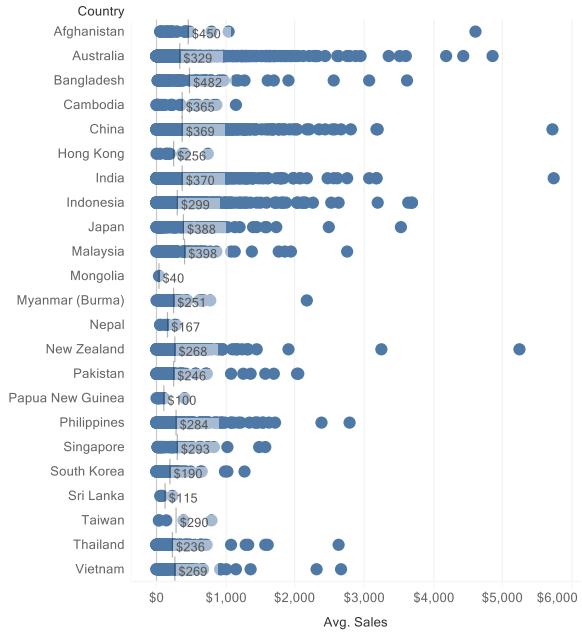
N'affichez que les repères nécessaires pour répondre à la question posée

Dans Tableau, il existe souvent plusieurs manières de calculer une même valeur. Ainsi, dans le tableau de bord suivant, les deux feuilles permettent de déterminer le montant moyen d'une commande dans chaque pays.

Avg Order Size 1



Avg Order Size 2

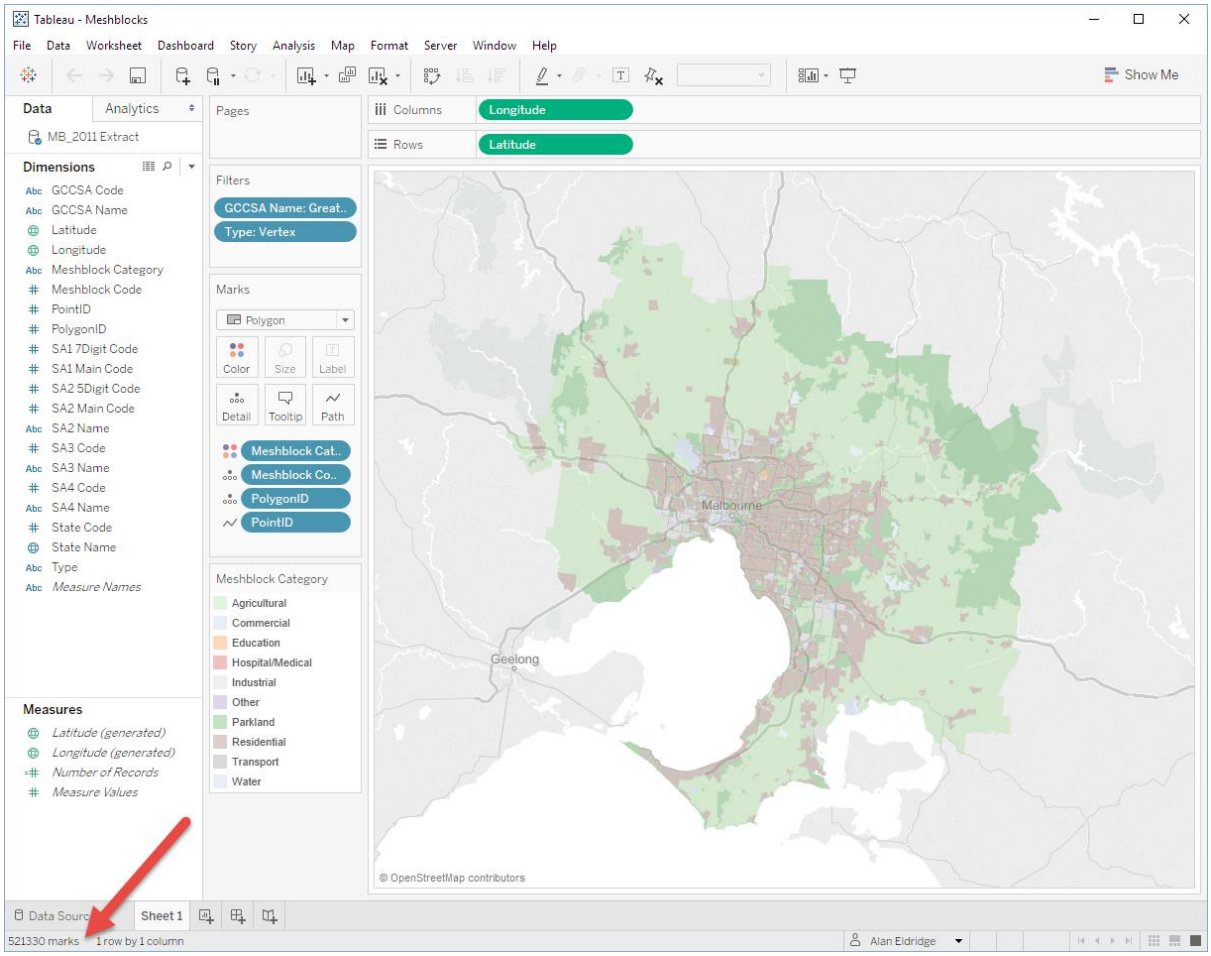


La feuille 1 affiche un seul repère, qui représente le montant moyen des commandes pour chaque pays, et 23 enregistrements seulement sont renvoyés depuis la source de données. Néanmoins, la feuille 2 affiche un repère pour chaque commande dans chaque pays, puis calcule la moyenne sous forme de ligne de référence. Il faut pour cela récupérer 5 436 enregistrements dans la source de données.

La feuille 1 est la meilleure solution si vous souhaitez simplement connaître le montant moyen des commandes par pays. La feuille 2 vous donne aussi cette réponse, mais avec plus d'informations sur la plage des montants, ce qui rend les valeurs atypiques plus faciles à repérer.

Évitez les visualisations trop complexes

Il est important de surveiller le nombre de points de données à représenter dans chaque visualisation. Cette valeur est indiquée dans la barre d'état de la fenêtre de Tableau Desktop.

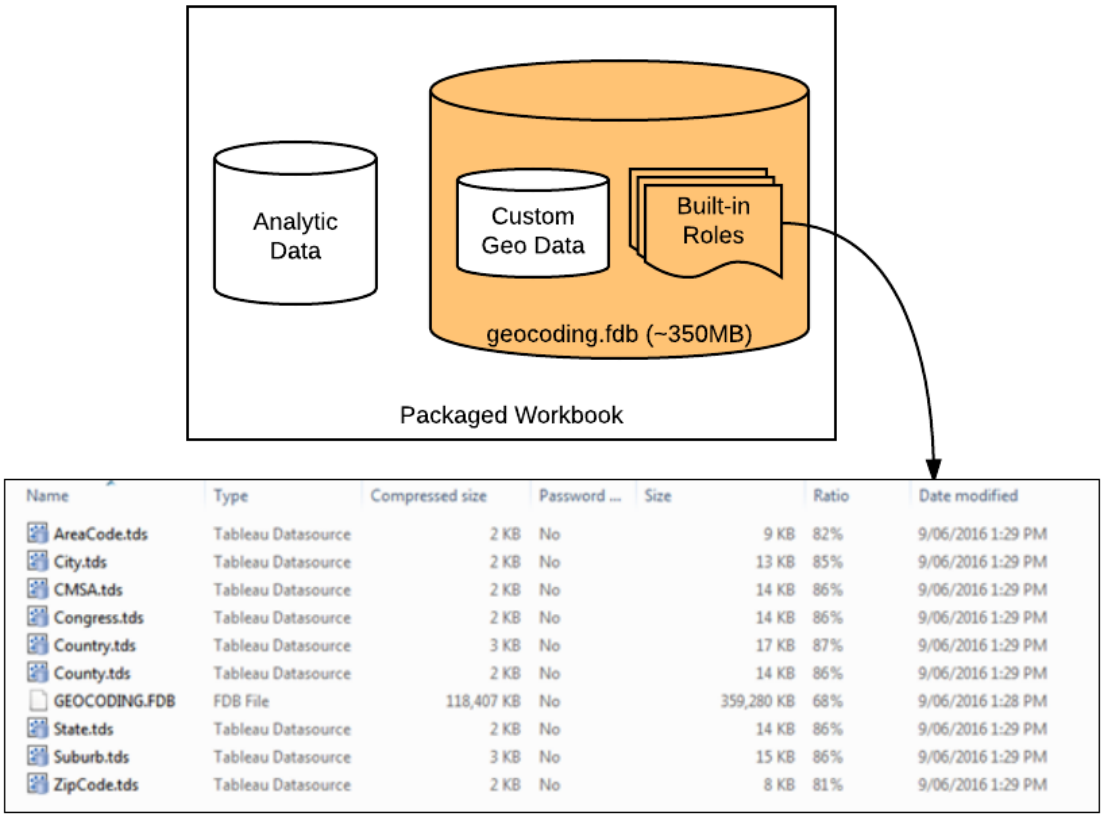


Bien qu'il n'existe aucune règle absolue pour savoir quand les repères sont trop nombreux, notez que plus le nombre de repères est élevé, plus vous aurez besoin de ressources (processeur et mémoire). Soyez particulièrement attentif aux tableaux croisés, aux nuages de points ou aux cartes de grande taille comportant des polygones personnalisés complexes.

Les cartes

Le géocodage personnalisé

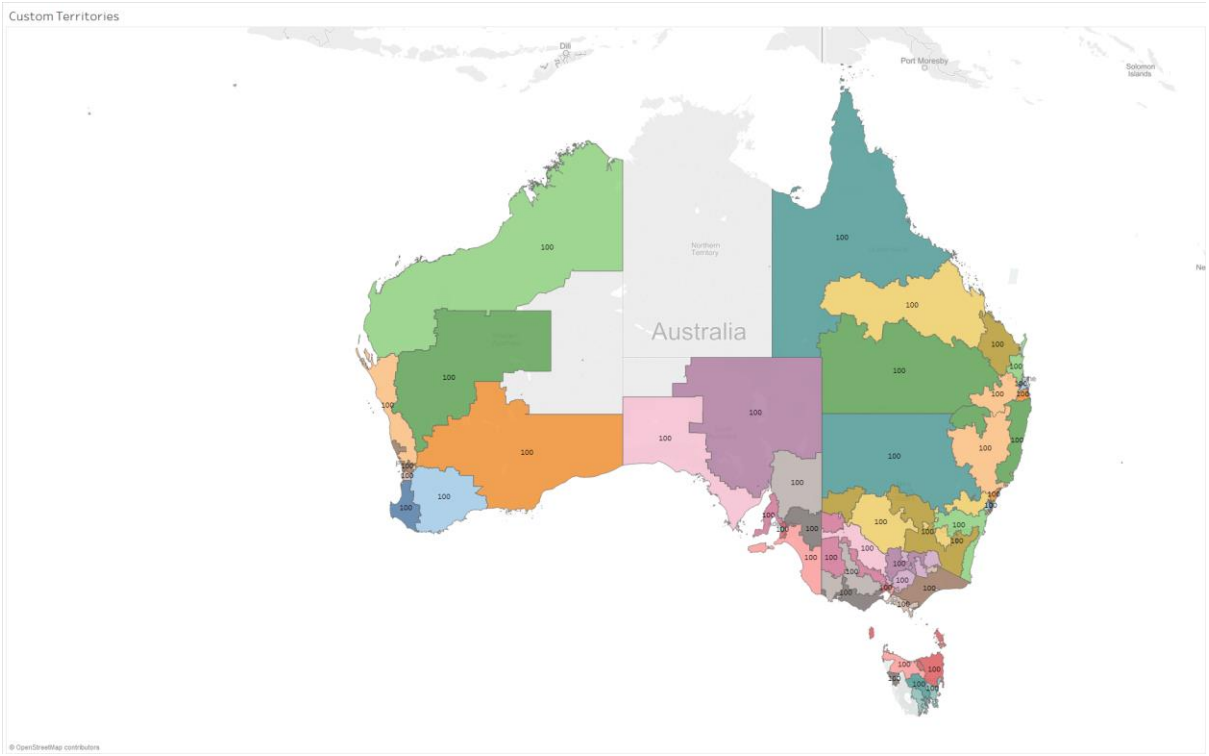
Lorsque vous importez un rôle de géocodage personnalisé, celui-ci est écrit dans la base de données de géocodage, à savoir un fichier Firebird stocké par défaut dans le dossier C:\Program Files\Tableau\Tableau 10.0\Local\data\geocoding.fdb. Si vous utilisez ce rôle dans un classeur que vous enregistrez en tant que classeur complet, tout le fichier de base de données sera compressé dans TWBX, soit environ 350 Mo.



Ce fichier sera par conséquent très volumineux, car une fois compressée, la base de données de géocodage occupe tout de même 110 Mo et mettra du temps à s'ouvrir, du fait que la décompression initiale doit s'effectuer sur un ensemble de données plus grand. Il est plus efficace de ne pas importer les données sous forme de rôle de géocodage personnalisé et d'utiliser la fusion dans le classeur pour combiner les données analytiques aux données géospatiales. Ainsi, le fichier geocoding.fdb ne sera pas inclus et le fichier TWBX contiendra uniquement les données analytiques et les données de géocodage personnalisées.

Les territoires personnalisés

Les territoires personnalisés, une nouvelle fonctionnalité de Tableau 10, permettent aux utilisateurs de combiner des zones des bases de données de géocodage internes pour créer des régions agrégées.



Le rendu initial des territoires personnalisés basés sur un grand nombre de régions de niveau inférieur peut être très lent. Aussi, n'utilisez pas inconsidérément cette fonctionnalité. Toutefois, une fois le rendu effectué, les territoires personnalisés sont mis en cache, de sorte que les performances s'amélioreront par la suite.

Les cartes pleines et les cartes de points

Les repères des cartes pleines, qu'ils soient utilisés sur une carte ou en tant que repères dans un autre type de graphique, consomment beaucoup de ressources pour le rendu côté client, car il faut envoyer les données des polygones pour définir la forme, ce qui constitue une opération complexe. Envisagez d'utiliser une carte de symboles si le rendu est lent.

Les repères de polygone

Les visualisations qui utilisent des repères de polygone obligent Tableau Server à effectuer un rendu côté serveur, ce qui peut affecter l'expérience utilisateur. Utilisez-les avec parcimonie.

Autres facteurs

Grands tableaux croisés

Dans les versions précédentes de ce document, nous vous recommandions d'éviter les grands tableaux croisés, parce que le rendu est très lent. Les mécanismes sous-jacents de ce type de visualisation ont été améliorés dans les versions les plus récentes, de sorte que ces tableaux s'affichent désormais aussi rapidement qu'une série de petits graphiques. Utilisez-les néanmoins avec discernement, car ils impliquent la lecture d'une grande quantité de données dans la source sous-jacente et ne sont pas utiles sur le plan analytique.

Infobulles

Par défaut, le fait de placer une dimension sur l'étagère Infobulle entraîne son agrégation avec la fonction Attribut ATTR(). Il faut alors effectuer deux agrégations, MIN() et MAX(), dans la source de données, et les deux résultats sont renvoyés vers l'ensemble de résultats. Reportez-vous à la section *Utilisation de la fonction ATTR()* dans ce document pour en savoir plus.

Si vous n'avez pas besoin d'avoir plusieurs valeurs de dimensions, il est plus efficace d'utiliser une seule agrégation au lieu de la fonction ATTR() par défaut. Utilisez soit MIN(), soit MAX(), mais tenez-vous à votre choix pour maximiser les chances de bénéficier des données du cache.

Autre possibilité, si vous savez que cela n'affectera pas le niveau de détail de votre visualisation : vous pouvez placer la dimension sur l'étagère du niveau de détail plutôt que sur l'étagère Infobulle. Le champ de dimension sera ainsi utilisé directement dans les clauses SELECT et GROUP BY de la requête. Nous vous recommandons de tester cette option pour vérifier qu'elle est plus efficace qu'une agrégation unique. Cela dépend en effet des performances de votre plate-forme de données.

Légendes

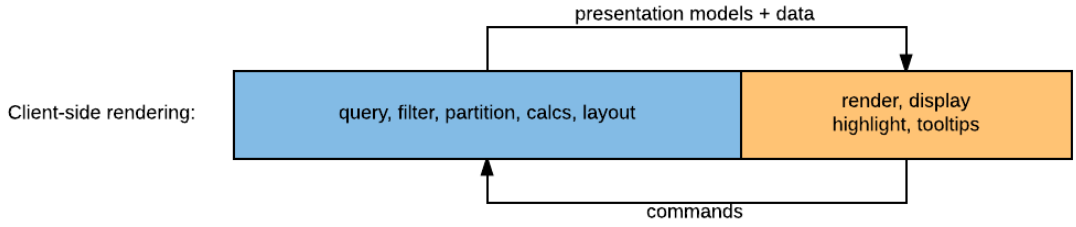
En général, les légendes n'ont pas d'impact sur les performances, car leurs domaines sont renseignés à l'aide du cache des résultats de requête. Néanmoins, elles peuvent peser sur le rendu si le domaine énuméré est grand, car les données doivent être transmises au navigateur client. Si tel est le cas, la légende n'est généralement pas utile et peut être supprimée.

Étagère des pages

Certains utilisateurs s'imaginent que l'étagère Pages fonctionne comme l'étagère Filtres et réduit le nombre d'enregistrements renvoyés depuis la source de données. C'est inexact dans la mesure où la requête destinée à la feuille de calcul renvoie les enregistrements pour tous les repères de toutes les pages. Si vous utilisez une dimension de page ayant un degré de cardinalité élevé (par exemple avec de nombreuses valeurs uniques), cela peut augmenter considérablement la taille de cette requête et donc affecter les performances. Utilisez cette fonctionnalité avec parcimonie.

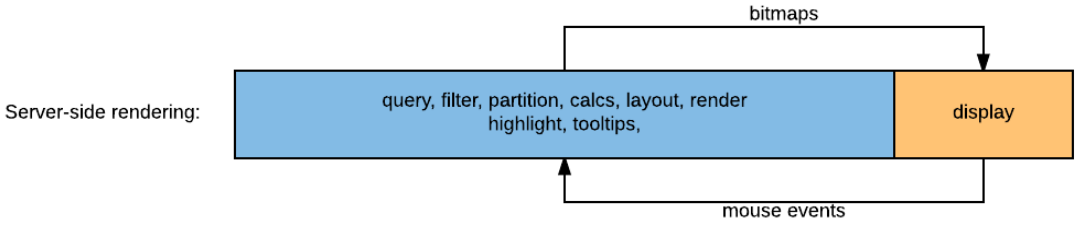
Les différences entre le rendu côté client et le rendu côté serveur

Avant que les repères et les données d'une vue s'affichent dans un navigateur Web client, ils sont récupérés, interprétés et restitués. Tableau Server peut exécuter ce processus dans le navigateur Web client ou sur le serveur. Par défaut, le rendu s'effectue côté client, car son traitement et toutes les interactions ayant lieu sur le serveur peuvent se traduire par un transfert accru de données sur le réseau et des retards dus aux allers-retours. Avec le rendu côté client, la plupart des interactions avec les vues sont plus rapides, car elles sont interprétées et restituées directement dans le client.



Bleu = action sur le serveur, Orange = action dans le navigateur client

Cependant, pour certaines vues, le rendu sur le serveur est plus efficace, car la puissance de calcul est plus importante. Le rendu côté serveur est utile pour une vue complexe, dans la mesure où les fichiers image utilisent bien moins de bande passante que les données utilisées pour créer ces images. Par ailleurs, les tablettes ayant une puissance de calcul inférieure à celle des ordinateurs, elles ne peuvent pas gérer le même niveau de complexité. En contrepartie, les interactions simples comme les infobulles et les sélections peuvent être lentes avec un rendu côté serveur, car elles nécessitent un aller-retour sur le serveur.



Bleu = action sur le serveur, Orange = action dans le navigateur client

Tableau Server est configuré pour gérer automatiquement toutes ces situations : un seuil de complexité déclenche le rendu d'une vue sur le serveur plutôt que dans le navigateur. Ce seuil n'est pas le même pour les ordinateurs et les appareils mobiles. Il se peut donc qu'une vue ouverte dans le navigateur d'un ordinateur résulte d'un rendu côté client, alors que le rendu sera effectué côté serveur si cette même vue est ouverte dans le navigateur d'une tablette. Le filtrage peut également influencer le comportement du rendu. Un classeur peut d'abord s'ouvrir avec un rendu côté serveur, puis passer à un rendu côté client lorsqu'un filtre est appliqué. Par ailleurs, si une visualisation utilise des repères de polygones ou l'étagère Pages, le rendu sera uniquement effectué côté serveur, même si elle remplit tous les critères pour un rendu côté client. Utilisez donc ces fonctionnalités avec précaution.

En tant qu'administrateur, vous pouvez tester ou ajuster ce paramètre pour les ordinateurs et les tablettes. Consultez cet article de l'aide en ligne pour en savoir plus :

https://onlinehelp.tableau.com/current/server/fr-fr/browser_rendering.htm

Utilisez des filtres efficaces

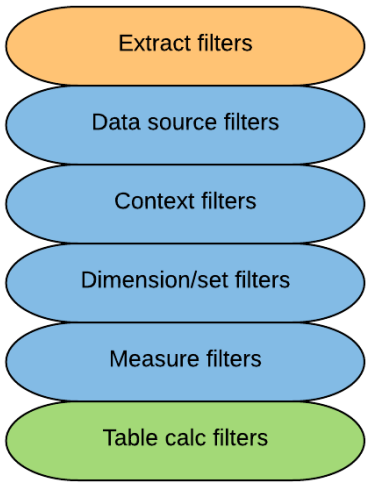
Dans Tableau, le filtrage est extrêmement puissant et expressif. Les filtres inefficaces sont pourtant l'une des principales causes du manque de performance des classeurs et des tableaux de bord. Découvrez dans les sections suivantes plusieurs bonnes pratiques en matière d'utilisation des filtres.

Remarque : l'efficacité des filtres est fortement influencée par la présence et la maintenance d'index dans la source de données. Reportez-vous à la section sur les index pour en savoir plus.

Les types de filtre

La priorité des filtres

Dans Tableau, les filtres sont appliqués dans l'ordre suivant :



Filtres d'extrait

Ces filtres sont applicables uniquement lorsque vous utilisez des extraits de données. Le cas échéant, ils sont appliqués de manière logique avant les autres filtres. Ils limitent les données récupérées dans la source de données d'origine, et peuvent agir sur les dimensions ou les mesures. Par ailleurs, ils peuvent exécuter une opération TOP ou SAMPLE pour réduire le nombre d'enregistrements renvoyés, en fonction de la plate-forme des données source.

Filtres de source de données

Ces filtres constituent le niveau de filtrage le plus élevé sur les connexions en direct. L'une des différences majeures avec les filtres contextuels, c'est qu'ils s'appliquent sur l'ensemble de la source de données, alors que les filtres contextuels sont définis pour chaque feuille de calcul. Par conséquent, lorsqu'ils sont utilisés dans une source de données publiée, les filtres de source de données peuvent être imposés, tandis que les filtres contextuels sont appliqués uniquement au niveau de la feuille.

Les filtres de source de données peuvent être efficaces pour empêcher les utilisateurs finaux d'exécuter par inadvertance une très grande requête sur une source de données. Vous pouvez par exemple définir ce filtre pour limiter les requêtes sur une table transactionnelle aux données des 6 mois précédents.

Filtres contextuels

Par défaut, tous les filtres que vous définissez dans Tableau sont calculés séparément. Autrement dit, chacun d'eux accède à toutes les lignes de votre source de données sans tenir compte des autres filtres. Néanmoins, en spécifiant un filtre contextuel, vous pouvez faire en sorte que les autres filtres que vous définissez en dépendent et traitent uniquement les données qu'il a préalablement traitées.

Utilisez des filtres contextuels lorsqu'ils sont nécessaires pour obtenir la réponse correcte, par exemple un classement Premiers N filtré. Supposons par exemple qu'une vue présente les 10 principaux produits en fonction de la somme des ventes, avec un filtrage par région. Sans filtre contextuel, la requête suivante est exécutée :

```
SELECT [Superstore APAC].[Product Name] AS [Product Name],
  SUM([Superstore APAC].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[Superstore APAC] [Superstore APAC]
  INNER JOIN (
  SELECT TOP 10 [Superstore APAC].[Product Name] AS [Product Name],
    SUM([Superstore APAC].[Sales]) AS [$__alias__0]
  FROM [dbo].[Superstore APAC] [Superstore APAC]
  GROUP BY [Superstore APAC].[Product Name]
  ORDER BY 2 DESC
) [t0] ON ([Superstore APAC].[Product Name] = [t0].[Product Name])
WHERE ([Superstore APAC].[Region] = 'Oceania')
GROUP BY [Superstore APAC].[Product Name]
```

Les résultats renseignent sur la part de l'Océanie dans les 10 premiers produits à l'échelle mondiale. Si vous vouliez en fait avoir la liste des 10 premiers produits dans la région Océanie, ajoutez le filtre Région au contexte. Cette requête serait alors exécutée :

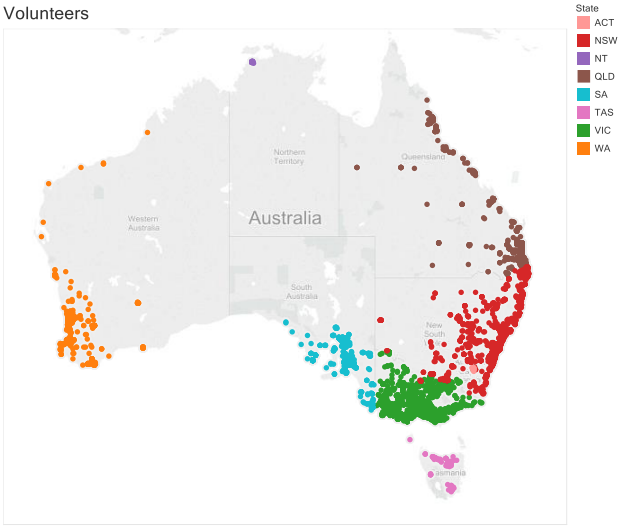
```
SELECT TOP 10 [Superstore APAC].[Product Name] AS [Product Name],
   SUM([Superstore APAC].[Sales]) AS [sum:Sales:ok],
   SUM([Superstore APAC].[Sales]) AS [$__alias__0]
FROM [dbo].[Superstore APAC] [Superstore APAC]
WHERE ([Superstore APAC].[Region] = 'Oceania')
GROUP BY [Superstore APAC].[Product Name]
ORDER BY 3 DESC
```

Au départ, les filtres contextuels ont été ajoutés pour jouer le rôle de table temporaire dans la source de données. Ce n'est plus le cas aujourd'hui. Le plus souvent, les filtres contextuels sont mis en œuvre dans le cadre d'une requête sur la source de données (comme dans l'exemple précédent) ou traités localement dans le moteur de données.

Vous ne devez plus vous servir des filtres contextuels pour améliorer les performances des requêtes.

Filtrage des dimensions de catégorie

Observez la visualisation suivante, qui présente une carte de l'Australie avec des repères pour chaque code postal :



Il existe différents moyens de filtrer la carte pour afficher uniquement les codes postaux de l'Australie-Occidentale (les points orange).

- Vous pouvez sélectionner tous les repères WA et conserver uniquement cette sélection.
- Vous pouvez sélectionner tous les repères sauf ceux de la zone WA, puis exclure cette sélection.
- Vous pouvez utiliser l'option Conserver uniquement sur un autre attribut, par exemple la dimension État.
- Vous pouvez filtrer par plage, soit sur les valeurs des codes postaux, soit sur les valeurs de latitude/longitude.

Discret

Avec les deux premières options, nous constatons que les options Conserver uniquement et Exclure sont peu efficaces. Elles peuvent même être plus lentes que l'ensemble de données non filtré. Cela est dû au fait qu'elles sont exprimées en tant que liste discrète de valeurs de codes postaux, incluses ou exclues par le système de base de données, que ce soit via la clause WHERE IN, ou, si les valeurs sont nombreuses, en créant une table temporaire avec les valeurs sélectionnées et en utilisant une jointure interne avec les tables principales. Pour un ensemble de repères volumineux, cela peut aboutir à une requête très difficile à évaluer.

La troisième option est rapide dans cet exemple, car le filtre résultant, (`WHERE STATE="Western Australia"`), est très simple et peut être traité efficacement par la base de données. Cette approche perd cependant en efficacité si le nombre de membres de dimension nécessaires pour exprimer le filtre augmente. On se rapproche alors des performances obtenues avec une sélection au lasso et l'option Conserver uniquement.

Plage de valeurs

L'utilisation d'une plage de valeurs permet également à la base de données d'évaluer une simple clause de filtre (`WHERE POSTCODE >= 6000 AND POSTCODE <= 7000` ou `WHERE LONGITUDE < 129`), accélérant ainsi l'exécution. Cependant, cette approche, à l'inverse d'un filtre sur une dimension liée, ne rend pas l'opération plus complexe à mesure que la cardinalité des dimensions augmente.

Les filtres avec plage de valeurs sont souvent plus rapides à évaluer que les longues listes détaillées de valeurs discrètes et sont préférables à l'option Conserver uniquement ou Exclure pour les ensembles contenant de nombreux repères.

Les filtres de séparation

Ces filtres s'exécutent sur des dimensions qui ne sont pas utilisées dans la visualisation (elles ne font pas partie du niveau de détail de la visualisation). Une visualisation peut par exemple porter sur les ventes totales par pays, mais être filtrée par région. Dans cet exemple, la requête s'exécute comme suit :

```
SELECT [Superstore APAC].[Country] AS [Country],
   SUM([Superstore APAC].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[Superstore APAC] [Superstore APAC]
WHERE ([Superstore APAC].[Region] = 'Oceania')
GROUP BY [Superstore APAC].[Country]
```

Ces filtres deviennent plus complexes si vous décomposez le résultat d'une agrégation. Par exemple, si vous filtrez la visualisation ci-dessus non pas par région mais de manière à afficher les ventes pour les 10 produits les plus rentables, Tableau doit exécuter deux requêtes : une au niveau des produits pour isoler les 10 plus rentables, et une autre au niveau des pays, alors limitée par les résultats de la première.

```
SELECT [Superstore APAC].[Country] AS [Country],
   SUM([Superstore APAC].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[Superstore APAC] [Superstore APAC]
   INNER JOIN (
   SELECT TOP 10 [Superstore APAC].[Product Name] AS [Product Name],
      SUM([Superstore APAC].[Profit]) AS [$__alias__0]
   FROM [dbo].[Superstore APAC] [Superstore APAC]
   GROUP BY [Superstore APAC].[Product Name]
   ORDER BY 2 DESC
) [t0] ON ([Superstore APAC].[Product Name] = [t0].[Product Name])
GROUP BY [Superstore APAC].[Country]
```

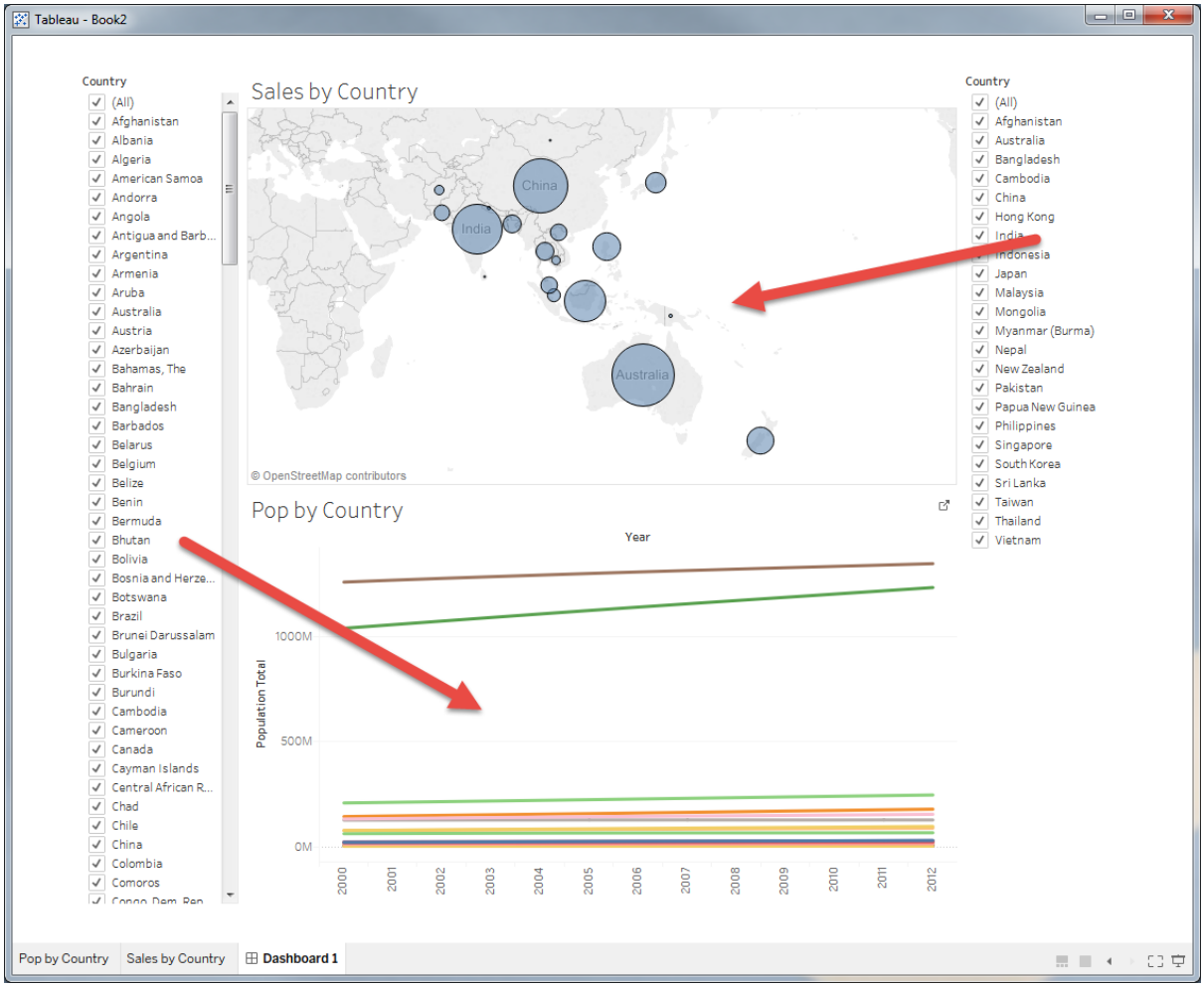
Utilisez les filtres de séparation avec prudence, car leur évaluation peut ne pas être pratique. Par ailleurs, comme la dimension n'est pas incluse dans le cache des résultats des requêtes, vous ne pouvez pas exécuter rapidement des filtres de séparation dans les navigateurs. Pour en savoir plus, reportez-vous à la section précédente sur le rendu côté client et côté serveur.

Les filtres sur plusieurs sources de données

Ces filtres sont une nouvelle fonctionnalité de Tableau 10. Il est désormais possible d'appliquer un filtre sur plusieurs sources de données ayant un ou plusieurs champs en commun. Les relations sont définies de la même manière que pour la fusion : automatiquement d'après les correspondances nom/type ou manuellement en fonction de relations personnalisées via le menu Données.

Ces filtres ont la même incidence sur les performances que les filtres rapides dans un tableau de bord. Le fait de les modifier peut entraîner la mise à jour de plusieurs zones, ce qui peut nécessiter l'exécution de plusieurs requêtes. Utilisez-les de manière judicieuse, et si vous prévoyez que les utilisateurs les modifieront plusieurs fois, vous pouvez proposer un bouton « Appliquer ». Ainsi, ils pourront déclencher l'actualisation une fois toutes les sélections effectuées.

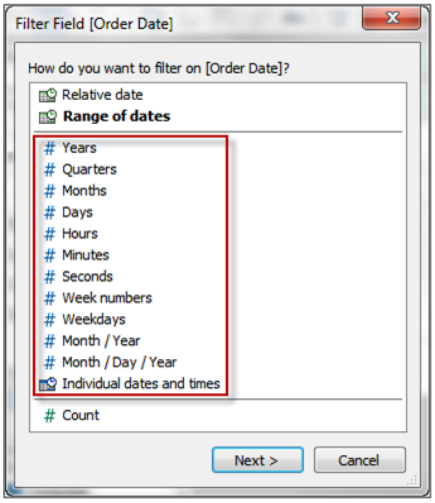
Notez également que le domaine du filtre est récupéré depuis la source de données principale, c'est-à-dire la première source de données utilisée pour la feuille sur laquelle le filtre a été créé. Si un champ connexe comporte différents domaines dans différentes sources de données, faites attention à celui que vous utilisez, car le même filtre peut produire des valeurs différentes, comme vous le voyez ci-dessous.

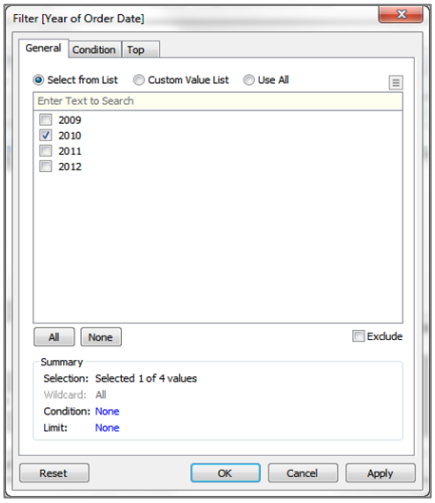


Filtrage de dates : discrètes et relatives, plages

Les champs de date sont un type de dimension spécial que Tableau gère souvent différemment par rapport aux données de catégorie standard. C'est particulièrement vrai lorsque vous créez des filtres de date. Les filtres de date sont extrêmement communs et sont répartis en trois catégories : les filtres Date relative, qui affichent une plage de dates par rapport à un jour spécifique, les filtres Plage de dates, qui affichent une plage définie de dates discrètes, et les filtres Date discrète, qui affichent les dates individuelles que vous avez sélectionnées dans une liste. Comme indiqué plus haut, la méthode utilisée peut avoir un impact sur l'efficacité de la requête résultante.

Date discrète





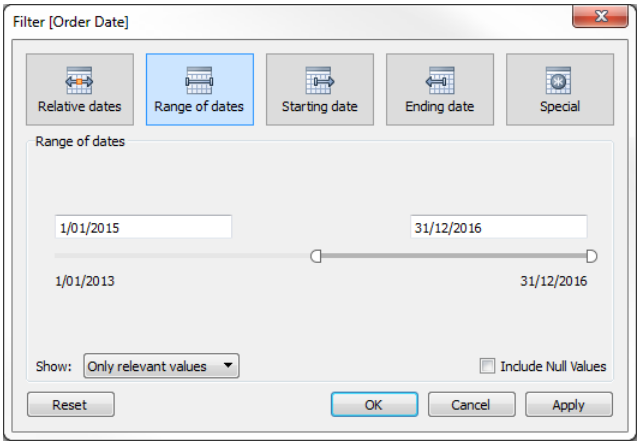
Vous pouvez parfois appliquer un filtre pour inclure des dates individuelles spécifiques ou des niveaux de dates entiers. Ce type de filtre est appelé Date discrète, car des valeurs discrètes sont définies à la place d'une plage. L'expression de date est alors transmise à la base de données sous la forme d'un calcul dynamique :

```
SELECT [FactSales].[Order Date], SUM([FactSales].[SalesAmount])
FROM [dbo].[FactSales] [FactSales]
WHERE (DATEPART(year,[FactSales].[Order Date]) = 2010)
GROUP BY [FactSales].[Order Date]
```

Dans la plupart des cas, les outils d'optimisation des requêtes sont capables d'évaluer le calcul DATEPART, mais parfois l'utilisation des filtres Date discrète entraîne une exécution médiocre de la requête. C'est le cas par exemple des requêtes sur une table partitionnée avec un filtre Date discrète sur la clé de partition de la date. La table n'étant pas partitionnée sur la valeur DATEPART, certaines bases de données évaluent le calcul sur toutes les partitions pour trouver des enregistrements correspondant aux critères, bien que cela ne soit pas nécessaire. Dans ce cas, vous pouvez obtenir de bien meilleures performances en utilisant un filtre Plage de dates ou Date relative avec une date spécifique en guise de point de repère.

Vous pouvez optimiser les performances de ce type de filtre en matérialisant le calcul avec un extrait de données. Créez d'abord un champ calculé qui met en œuvre la fonction DATEPART de manière explicite. Si vous créez ensuite un extrait de données Tableau, ce champ calculé sera matérialisé sous forme de valeurs stockées dans l'extrait (car le résultat de l'expression est déterministe). Le filtrage sur le champ calculé au lieu de l'expression dynamique sera plus rapide, car la valeur peut être simplement recherchée, au lieu d'être calculée au moment de la requête.

Plage de dates

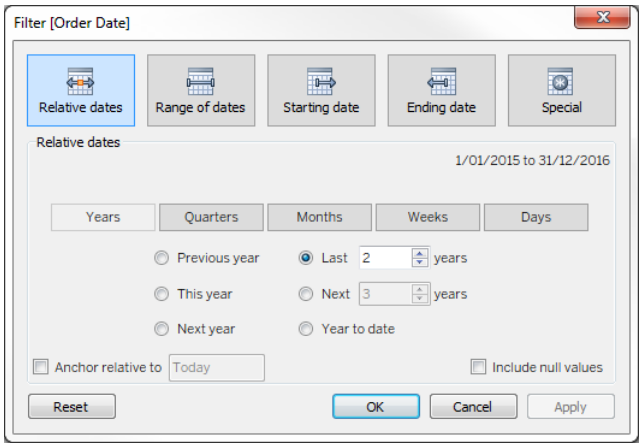


Ce type de filtre s'utilise lorsque vous spécifiez une plage de dates contiguës. La structure de requête suivante est transmise à la base de données :

```
SELECT SUM([factOrders].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[factOrders] [factOrders]
WHERE (([factOrders].[Order Date] >= {d '2015-01-01'}) AND
([factOrders].[Order Date] <= {d '2016-12-31'}))
GROUP BY ()
```

Ce type de clause WHERE est très efficace pour les outils d'optimisation des requêtes, ce qui permet aux plans d'exécution de tirer pleinement parti des index et des partitions. Si les requêtes sont lentes lorsque vous ajoutez des filtres Date discrète, essayez plutôt d'utiliser des filtres Plage de dates et voyez si cela résout les problèmes.

Date relative



Ce filtre permet de définir une plage de dates mise à jour en fonction de la date et de l'heure d'ouverture de la vue. Par exemple, vous pouvez voir les ventes annuelles cumulées, tous les enregistrements des 30 derniers jours ou les bugs résolus la semaine précédente. Les filtres Date relative peuvent également être associés à une date repère plutôt qu'à la date du jour.

```
SELECT SUM([factOrders].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[factOrders] [factOrders]
WHERE (([factOrders].[Order Date] >= {ts '2015-01-01 00:00:00'}) AND
([factOrders].[Order Date] < {ts '2017-01-01 00:00:00'}))
GROUP BY ()
```

Comme vous pouvez le constater, la clause WHERE résultante utilise la syntaxe d'une plage de dates, ce qui constitue également un filtre de dates efficace.

Remarque : étant donné la variabilité des filtres de date qui sont relatifs à la date/heure actuelle, via l'option de filtre Date relative ci-dessus ou par le biais de l'utilisation explicite de NOW() ou TODAY() dans une formule, le cache des requêtes n'est pas aussi efficace pour les requêtes qui utilisent ces filtres. Tableau les considère comme des requêtes transitoires dans le serveur de cache, et leurs résultats ne sont pas conservés aussi longtemps que ceux des autres requêtes.

Les filtres affichés

Afficher trop de filtres ralentira les performances, en particulier si vous les configurez pour qu'ils utilisent les valeurs pertinentes uniquement et si les listes discrètes sont nombreuses. Optez pour une approche analytique plus ciblée et utilisez plutôt des filtres d'action dans un tableau de bord. Si vous créez une vue en ajoutant beaucoup de filtres pour la rendre très interactive, demandez-vous s'il ne serait pas plus utile de créer plusieurs tableaux de bord avec différents niveaux et thèmes. C'est très certainement le cas.

Filtres énumérés et non énumérés

Si des filtres énumérés sont proposés dans la vue, Tableau doit chercher les valeurs des champs dans la source de données avant de pouvoir afficher les résultats du filtre. Cela inclut les éléments suivants :

- Liste à valeurs multiples : *tous les membres des dimensions*
- Liste à valeur unique : *tous les membres des dimensions*
- Liste compacte : *tous les membres des dimensions*
- Curseur : *tous les membres des dimensions*
- Filtres de mesure : *valeurs MIN et MAX*
- Filtres Plage de dates : *valeurs MIN et MAX*

D'un autre côté, les filtres non énumérés affichés dans la vue ne nécessitent pas de connaître les valeurs potentielles des champs. Cela inclut :

- Liste de valeurs personnalisées
- Correspondance avec des caractères génériques
- Filtres Date relative
- Filtres de date avec choix des périodes

Par conséquent, afficher des filtres non énumérés dans la vue réduit le nombre de requêtes liées aux filtres qui doivent être exécutées par la source de données. Par ailleurs, le rendu est plus rapide pour les filtres non énumérés affichés dans la vue lorsqu'il y a de nombreux membres de dimension à montrer.

Utiliser ces filtres dans la vue peut améliorer les performances, mais au détriment du confort visuel pour l'utilisateur final.

Valeurs pertinentes

Les filtres énumérés affichés dans la vue peuvent être configurés pour afficher les valeurs potentielles des champs de trois manières différentes.

- Toutes les valeurs de la base de données : lorsque vous sélectionnez cette option, toutes les valeurs de la base de données sont affichées, quels que soient les autres filtres de la vue. Pour ce filtre, il n'est pas nécessaire de renvoyer une requête sur la base de données lorsque les autres filtres sont modifiés.
- Toutes les valeurs du contexte : cette option est disponible uniquement lorsque vous avez des filtres contextuels actifs. Le filtre affiché dans la vue montre toutes les valeurs du contexte,

sans tenir compte des autres filtres présents. Il ne nécessite pas le renvoi d'une requête sur la base de données lorsque des filtres de mesure ou de dimension sont modifiés. Cela est toutefois nécessaire si le contexte change.

- Les valeurs pertinentes uniquement : lorsque vous sélectionnez cette option, les autres filtres sont pris en compte et seules les valeurs qu'ils conservent sont affichées. Par exemple, un filtre sur l'État n'affichera que les États de la région Est lorsqu'un filtre est défini sur la région. Par conséquent, le filtre doit renvoyer une requête sur la source de données lorsque les autres filtres sont modifiés.

Vous voyez que l'option « Les valeurs pertinentes uniquement » peut être très utile pour aider l'utilisateur à effectuer les sélections appropriées, mais elle peut augmenter considérablement le nombre de requêtes à exécuter lors de ses interactions avec le tableau de bord. Ne l'utilisez que si c'est nécessaire.

Alternatives à l'affichage des filtres dans la vue

D'autres méthodes produisent un résultat similaire du point de vue analytique en évitant la surcharge liée aux requêtes. Vous pouvez par exemple créer un paramètre et effectuer un filtrage en fonction des sélections de l'utilisateur.

- AVANTAGES :
  - Les paramètres ne nécessitent pas de requête sur la source de données avant le rendu.
  - Les paramètres associés aux champs calculés peuvent mettre en œuvre une logique plus complexe que celle d'un simple filtre de champ.
  - Les paramètres permettent de filtrer sur plusieurs sources de données. Dans les versions antérieures à Tableau 10, les filtres n'agissent que dans une seule source de données. Depuis Tableau 10, les filtres peuvent être configurés pour s'exécuter sur plusieurs sources de données connexes.
- INCONVÉNIENTS :
  - Les paramètres utilisent seulement des valeurs uniques. Vous ne pouvez pas les utiliser pour sélectionner plusieurs valeurs.
  - Les paramètres ne sont pas dynamiques : la liste des valeurs est définie lors de leur création et elle n'est pas actualisée en fonction des valeurs du système de base de données.

Vous pouvez également utiliser des actions de filtre entre les vues.

- AVANTAGES :
  - Les actions prennent en charge les sélections multiples, que ce soit visuellement avec le lasso ou avec la combinaison de touches CTRL+Maj.
  - Les actions présentent une liste dynamique de valeurs, laquelle est évaluée au moment de l'exécution.
  - Les actions permettent de filtrer plusieurs sources de données. Dans les versions antérieures à Tableau 10, les filtres n'agissent que dans une seule source de données. Depuis Tableau 10, les filtres peuvent être configurés pour s'exécuter sur plusieurs sources de données connexes.
- INCONVÉNIENTS :
  - Les actions de filtre sont plus complexes à configurer que les filtres affichés dans la vue.

- Les actions ne proposent pas la même interface utilisateur que les paramètres ou les filtres. En général, elles nécessitent davantage d'espace sur l'écran.
- La feuille source de l'action doit toujours interroger la source de données, mais elle bénéficie de la mise en cache dans le système de traitement de Tableau.

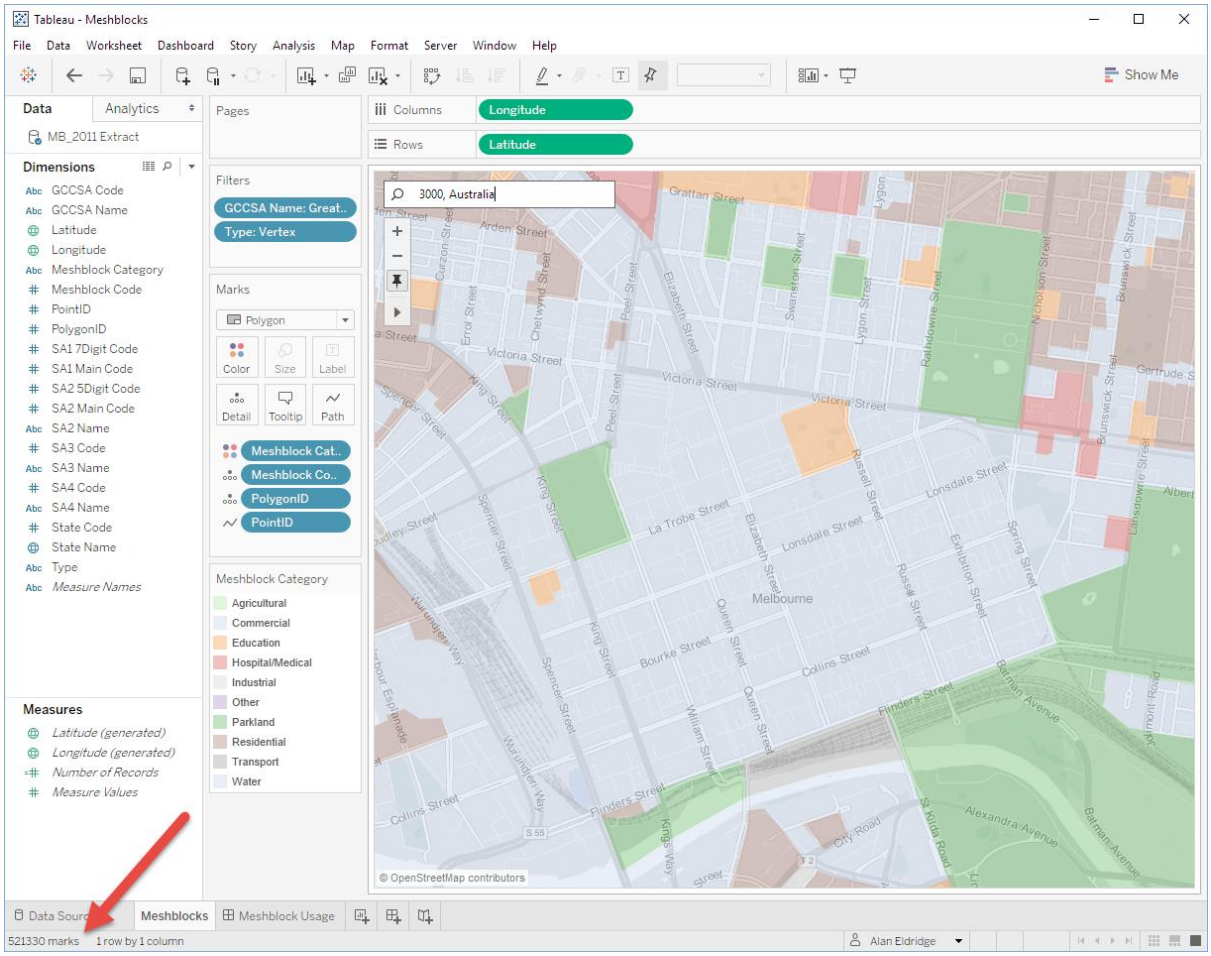
Pour en savoir plus sur les autres techniques de conception qui ne s'appuient pas sur les filtres des vues, reportez-vous à la section précédente qui explique la bonne méthode pour concevoir un tableau de bord.

Les filtres utilisateur

Lorsqu'un classeur contient des filtres utilisateur, créés via la boîte de dialogue Créer un filtre utilisateur ou via des champs calculés utilisant l'une des fonctions utilisateur intégrées comme ISMEMBEROF(), le cache de modélisation n'est pas commun à toutes les sessions utilisateur. Cela peut fortement réduire le taux de réutilisation du cache, ce qui peut créer une charge plus importante pour Tableau Server. N'utilisez pas ces types de filtre inconsidérément.

Les différences entre le zoom et le filtrage

Lorsque vous zoomez sur une visualisation comportant de nombreux repères, sachez que ceux que vous ne pouvez pas voir ne sont pas exclus. Seule la zone visible change. Le nombre total de repères manipulés ne varie pas, comme vous pouvez le voir sur l'image suivante.



Si vous avez uniquement besoin d'un sous-ensemble des données, excluez celles qui sont inutiles et laissez la fonction de zoom automatique définir la zone visible.

Les calculs sont-ils en cause ?

Dans de nombreux cas, la source de données ne propose pas tous les champs nécessaires pour répondre à toutes vos questions. Les champs calculés vous permettent de créer toutes les dimensions et mesures dont vous avez besoin pour votre analyse.

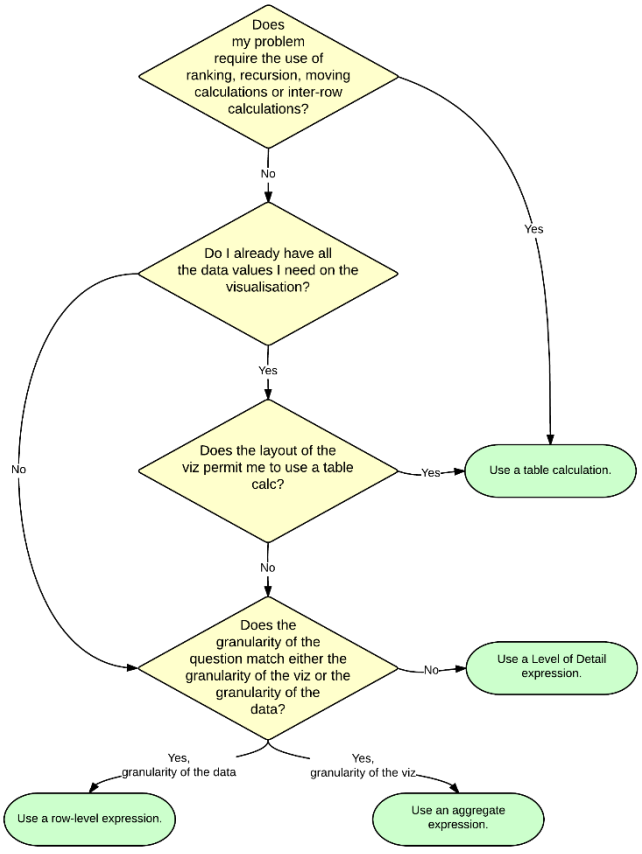
Dans un champ calculé, vous pouvez définir une constante non modifiable (comme un taux d'imposition, par exemple), effectuer des opérations mathématiques très simples comme les soustractions ou les multiplications (par exemple soustraire les charges du chiffre d'affaires), utiliser des formules mathématiques plus complexes, effectuer des tests logiques (IF/THEN, CASE), effectuer des conversions de type, etc.

Une fois défini, le champ calculé est disponible dans tout le classeur, à condition que les feuilles utilisent toutes la même source de données. Vous pouvez utiliser des champs calculés dans votre classeur de la même manière que vous utilisez les dimensions et les mesures de votre source de données.

Il existe quatre types de calcul dans Tableau :

- Les calculs au niveau des lignes
- Les calculs agrégés
- Les calculs de table
- Les expressions LOD (Level Of Detail, ou niveau de détail)

Utilisez le schéma suivant pour sélectionner l'approche la mieux adaptée :



La bibliothèque de référence Tableau sur les calculs constitue une excellente source d'informations en matière de calculs complexes. De plus, les utilisateurs peuvent partager des solutions aux problèmes courants sur le forum. Visitez la page suivante pour en savoir plus :

> https://community.tableau.com/community/viz-talk/tableau-community-library/calculation-reference-library

Les types de calcul

Calculs au niveau des lignes et calculs agrégés

Les calculs au niveau des lignes et les calculs agrégés sont exprimés dans le cadre de la requête envoyée à la source de données, et sont donc calculés par la base de données. Par exemple, une visualisation affichant la somme des ventes totales pour chaque année enverrait la requête suivante à la source de données :

```
SELECT DATEPART(year,[OrdersFact].[Order Date]) AS [yr:Order Date:ok],
   SUM([OrdersFact].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[OrdersFact] [OrdersFact]
GROUP BY DATEPART(year,[OrdersFact].[Order Date])
```

YEAR est un calcul au niveau des lignes et SUM(SALES) est un calcul agrégé.

En général, les calculs au niveau des lignes et les calculs agrégés sont efficaces, et plusieurs techniques d'optimisation de base de données permettent d'améliorer leurs performances.

Notez que dans Tableau, la requête vers la base de données n'est pas forcément une traduction directe des calculs de base utilisés dans la visualisation. Par exemple, la requête suivante est exécutée lorsque la visualisation contient un champ calculé sur le taux de rentabilité, défini comme suit `SUM([Profit])/SUM([Sales])` `SUM([Profit])/SUM([Sales])` :

```
SELECT DATEPART(year,[OrdersFact].[Order Date]) AS [yr:Order Date:ok],
   SUM([OrdersFact].[Profit]) AS
[TEMP(Calculation_0260604221950559)(1796823176)(0)],
   SUM([OrdersFact].[Sales]) AS
[TEMP(Calculation_0260604221950559)(3018240649)(0)]
FROM [dbo].[OrdersFact] [OrdersFact]
GROUP BY DATEPART(year,[OrdersFact].[Order Date])
```

Tableau récupère les éléments du calcul et exécute la fonction de division dans la couche client. Cela garantit que `SUM([Profit])` `SUM([Profit])` et `SUM([Sales])` au niveau de l'année sont mis en cache et peuvent être utilisés partout ailleurs dans le classeur, sans avoir à revenir à la source de données.

Enfin, la fusion de requêtes, qui consiste à combiner plusieurs requêtes logiques en une seule, peut modifier la requête exécutée sur la source de données. Cela peut se traduire par la combinaison de plusieurs mesures issues de plusieurs feuilles en une seule requête, si ces autres feuilles ont la même granularité.

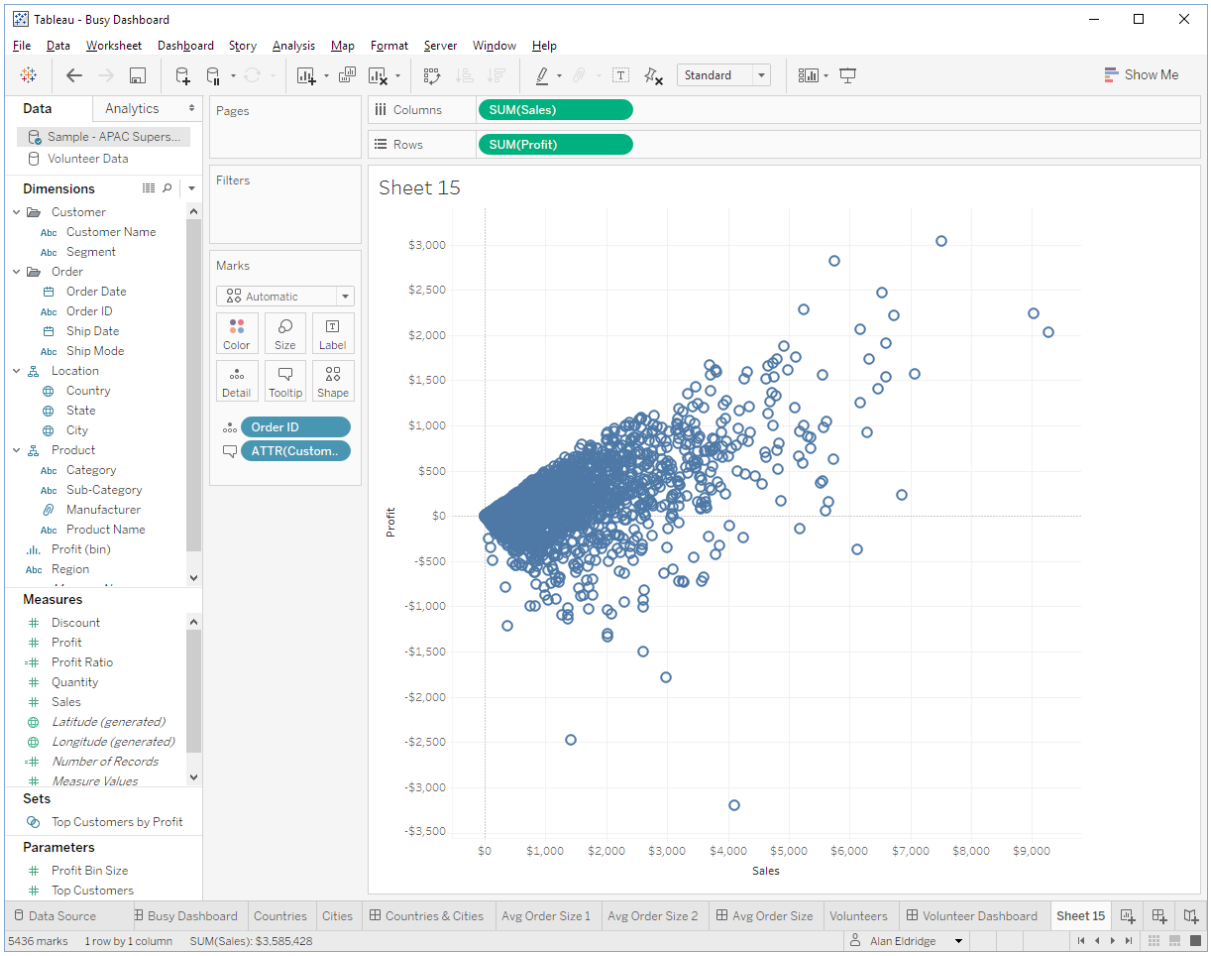
Utilisation de la fonction ATTR()

ATTR() est une fonction puissante, souvent utilisée comme agrégation pour les dimensions de catégorie. Elle renvoie une valeur si elle est unique. Sinon, elle renvoie *. Techniquement, cette fonction exécute le test logique suivant :

```
IF MIN([dimension]) = MAX([dimension])
 THEN [dimension]
 ELSE "*"
END
```

Il faut alors effectuer deux agrégations, MIN() et MAX(), dans la source de données, et les deux résultats sont renvoyés vers l'ensemble de résultats. Si vous n'avez pas besoin d'avoir plusieurs valeurs de dimensions, il est plus efficace d'utiliser une seule agrégation, MIN() ou MAX(). Quelle que soit l'agrégation choisie, utilisez ensuite toujours la même pour maximiser les chances de bénéficier des données du cache.

Pour illustrer l'impact d'ATTR() sur la source de données, nous avons créé la visualisation suivante.



Par défaut, lorsque vous placez une dimension sur l'étagère Infobulle, ATTR() est utilisée comme agrégation. Cela entraîne l'exécution de la requête suivante :

```
SELECT [Superstore APAC].[Order ID] AS [Order ID],
   MAX([Superstore APAC].[Customer Name]) AS [TEMP(attr:Customer
Name:nk)(2542222306)(0)],
   MIN([Superstore APAC].[Customer Name]) AS [TEMP(attr:Customer
Name:nk)(3251312272)(0)],
   SUM([Superstore APAC].[Profit]) AS [sum:Profit:ok],
   SUM([Superstore APAC].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[Superstore APAC] [Superstore APAC]
GROUP BY [Superstore APAC].[Order ID]
```

Si vous savez qu'il n'existe pas plusieurs clients pour chaque ID de commande, vous pouvez utiliser l'agrégation MIN() à la place et simplifier la requête :

```
SELECT [Superstore APAC].[Order ID] AS [Order ID],
   MIN([Superstore APAC].[Customer Name]) AS [min:Customer Name:nk],
   SUM([Superstore APAC].[Profit]) AS [sum:Profit:ok],
   SUM([Superstore APAC].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[Superstore APAC] [Superstore APAC]
GROUP BY [Superstore APAC].[Order ID]
```

Le tableau suivant montre les différences entre les deux options en termes de performances. Les résultats sont exprimés en secondes nécessaires à l'exécution de la requête, comme dans le fichier journal.

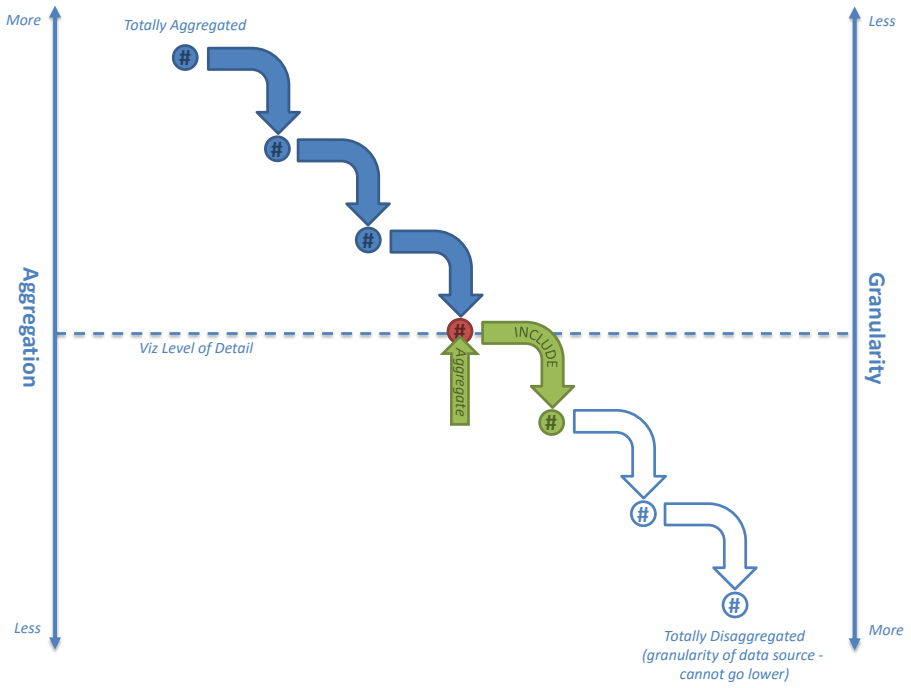
Test #	ATTR	MIN
1	2.824	1.857
2	2.62	2.09
3	2.737	2.878
4	2.804	1.977
5	2.994	1.882
Average	2.7958	2.1368
% improvement		**24%**

Pour en savoir plus sur la fonction ATTR(), reportez-vous à l'article suivant sur le blog d'InterWorks :

http://bit.ly/1YEuZhX

Les calculs LOD

Les expressions LOD permettent de définir le niveau de détail auquel le calcul sera exécuté, indépendamment du niveau de détail de votre visualisation.



Les expressions LOD peuvent dans certains cas remplacer les calculs plus laborieux qu'il était possible de créer dans les versions précédentes de Tableau :

- Vous avez peut-être utilisé un calcul de table pour déterminer la première et la dernière période dans une partition, par exemple pour calculer les effectifs d'une entreprise le premier jour et le dernier jour de chaque mois.
- Vous avez peut-être utilisé des calculs de table, des champs calculés et des lignes de référence pour classer des champs agrégés, par exemple pour déterminer la moyenne d'un total distinct de clients.

- Vous avez peut-être utilisé la fusion de données pour obtenir un filtrage de date par rapport à la date maximale de vos données, par exemple pour calculer des totaux de l'année jusqu'au jour actuel en fonction de la date maximale, si vos données sont actualisées chaque semaine.

Les expressions LOD sont générées dans le cadre d'une requête sur la source de données sous-jacente. Elles sont exprimées en tant que sélection imbriquée et dépendent des performances du système de gestion de la base de données :

```
SELECT T1.[State],SUM(T2.[Sales per County, State])
FROM [DB Table] T1 INNER JOIN
   (SELECT [State], [County], AVG([Sales]) AS [Sales per County, State]
   FROM [DB Table] GROUP BY [State],[County]) T2
ON T1.[State] = T2.[State]
GROUP BY T1.[State]
```

Cela signifie que dans certains cas, il est plus efficace de résoudre une problématique à l'aide d'un calcul de table ou d'une fusion de données qu'avec une expression LOD, et inversement. Si vous suspectez que le ralentissement des performances est dû à une expression LOD, essayez de la remplacer par un calcul de table ou une fusion de données si possible, pour observer si les performances s'améliorent. Par ailleurs, l'élimination des jointures peut avoir un impact important sur les expressions LOD. Nous vous recommandons de vous reporter à cette section particulière si vos requêtes s'exécutent lentement lorsque vous utilisez des expressions LOD.

Pour mieux comprendre les mécanismes des expressions LOD, lisez le livre blanc Mieux comprendre les expressions LOD (Level of Detail) :

http://www.tableau.com/fr-fr/learn/whitepapers/understanding-lod-expressions

Vous pouvez également lire l'article de blog « Top 15 des expressions LOD » de Bethany Lyons, qui propose des exemples pour des problèmes courants :

http://www.tableau.com/fr-fr/about/blog/LOD-expressions

Enfin, les blogs de la communauté proposent de nombreux articles utiles sur le sujet, dont cette sélection très intéressante sur la page *data + science* :

http://bit.ly/1MpkFV5

Les calculs de table

À l'inverse des calculs au niveau des lignes et des calculs agrégés, les calculs de table ne sont pas exécutés par la base de données, mais effectués par Tableau sur l'ensemble des résultats de la requête. Cela sollicite davantage Tableau, mais ce calcul est généralement effectué sur un ensemble d'enregistrements plus restreint que la source de données d'origine.

Si les performances des calculs de table constituent un problème (par exemple si l'ensemble de résultats renvoyé à Tableau est très volumineux), vous pouvez envisager de déplacer certains aspects du calcul vers la couche de la source de données. Pour ce faire, vous pouvez utiliser une expression LOD, ou tirer parti d'un extrait de données agrégé. Supposons par exemple que vous souhaitiez déterminer la moyenne hebdomadaire des ventes totales réalisées chaque jour par plusieurs magasins. Vous pouvez utiliser ce calcul de table de la manière suivante :

```
WINDOW_AVG(SUM([Sales])
```

Cependant, si le nombre de jours/magasins est très élevé, ce calcul peut devenir lent. Pour renvoyer SUM([Sales]) sur la couche de données, créez un extrait agrégé qui agrège la dimension DATE au niveau

des jours. Le calcul peut être effectué simplement avec AVG([Sales]), car l'extrait a déjà matérialisé les totaux quotidiens.

Dans certains cas, vous savez que la valeur du facteur d'agrégation ne change pas dans la partition. Le cas échéant, utilisez les fonctions d'agrégation MIN() ou MAX() au lieu d'AVG() ou d'ATTR(), car elles sont plus rapides à évaluer. Tenez-vous ensuite à votre choix pour maximiser les chances de bénéficier des données du cache.

Les calculs externes

Tableau permet de transmettre la logique complexe à un moteur externe pour les calculs. Cela inclut notamment les appels à R (par une connexion Rserve) ou Python (par une connexion serveur Dato).

Dans Tableau, ces opérations sont similaires aux calculs de table, et sont donc effectuées sur les partitions. Autrement dit, elles peuvent être appelées plusieurs fois pour la même visualisation, ce qui peut avoir un impact sur les performances. Elles ne sont en aucun cas optimisées ou regroupées, et vous devez vous demander si vous pouvez obtenir plusieurs valeurs renvoyées dans un appel de fonction unique (comme une chaîne concaténée) au lieu d'effectuer plusieurs appels de fonction. Enfin, le temps de transfert des données échangées avec le moteur de calcul externe peut également ralentir les performances.

Les analyses

Le volet Analyse permet d'accéder rapidement à plusieurs analyses avancées, comme :

- Totaux
- Lignes de référence
- Courbes de tendance
- Prévisions
- Clusters (nouvelle fonctionnalité de Tableau 10)

Ces analyses ne nécessitent généralement pas l'exécution de requêtes supplémentaires et fonctionnent plutôt comme des calculs de table sur les données du cache des résultats de requête. Comme les calculs de table, si les résultats de la requête comportent un très grand ensemble de données, le calcul peut prendre du temps, mais cela ne contribue généralement pas énormément aux mauvaises performances du classeur.

Les agrégations de mesures additives et non additives affectent également les performances pour les totaux et les lignes de référence. Pour les agrégations additives (SUM, MIN ou MAX), le calcul des totaux et des lignes de référence peut être effectué localement dans Tableau. Pour les agrégations non additives (COUNTD, TOTAL, MEDIAN ou PERCENTILE), vous devez revenir à la source de données pour calculer les valeurs des totaux et des lignes de référence.

Les calculs et les fonctionnalités natives

Les utilisateurs créent parfois des champs calculés pour exécuter des fonctions qui pourraient facilement être assurées avec des fonctionnalités natives de Tableau. Par exemple :

- Pour regrouper des membres d'une dimension, vous pouvez utiliser des groupes ou des ensembles.
- Pour regrouper des valeurs de mesure en plages de bandes, vous pouvez utiliser des classes.
- Pour tronquer des dates afin d'obtenir une granularité plus ordinaire, comme les mois ou les semaines, vous pouvez utiliser des champs de dates personnalisées.

- Pour créer un ensemble de valeurs concaténant deux dimensions différentes, vous pouvez utiliser des champs combinés.
- Pour afficher des dates avec un format spécifique ou convertir des valeurs numériques en KPI, vous pouvez utiliser les fonctionnalités de mise en forme intégrées.
- Pour modifier les valeurs affichées pour les membres de dimension, vous pouvez utiliser des alias.

Cela n'est pas toujours possible. Vous pouvez par exemple avoir besoin de classes de tailles variables, ce que ne permettent pas les classes de base. Gardez cependant les fonctionnalités de base à l'esprit dans la mesure du possible. Elles sont souvent plus efficaces qu'un calcul manuel, et vous pourrez profiter des améliorations continuellement apportées par nos développeurs.

Impact sur les types de données

Lorsque vous créez des champs calculés, il est important de comprendre que le type de données utilisé a un impact significatif sur la rapidité du calcul. En règle générale

- Les entiers et les booléens sont plus rapides que les chaînes et les dates.

Les calculs des chaînes et des dates sont très lents. Il y a souvent entre 10 et 100 instructions de base à exécuter pour chaque calcul. En comparaison, les calculs numériques et booléens sont très efficaces.

C'est vrai pour la plupart des bases de données, et pas uniquement pour le moteur de calcul de Tableau. Les calculs de base et agrégés étant déplacés vers la base de données, ils s'exécutent plus rapidement s'ils sont exprimés avec une logique numérique plutôt qu'avec des chaînes.

Les techniques d'optimisation des performances

Prenez en compte les techniques suivantes pour garantir que vos calculs seront aussi efficaces que possible.

Utilisez des booléens pour les calculs de logique simple

Si vous utilisez un calcul qui produit un résultat binaire (oui/non, échec/réussite, supérieur/inférieur), assurez-vous de renvoyer un résultat booléen au lieu d'une chaîne. Exemple :

```
IF [Date] = TODAY() THEN "Today"
   ELSE "Not Today"
END
```

Ce calcul sera lent car il utilise des chaînes. Vous pouvez l'accélérer en renvoyant un booléen :

```
[Date] = TODAY()
```

Utilisez ensuite des alias pour renommer les résultats TRUE et FALSE en « Today » (Aujourd'hui) et « Not Today » (Pas aujourd'hui).

Recherches de chaînes

Supposons que vous recherchiez tous les enregistrements dans lesquels le nom de produit contient une certaine chaîne. Vous pouvez utiliser un paramètre pour obtenir la chaîne recherchée par l'utilisateur, puis créer le champ calculé suivant.

```
IF FIND([Product Name],[Product Lookup]) > 0
   THEN [Product Name]
   ELSE NULL
END
```

Ce calcul est lent, car ce n'est pas une manière efficace de tester si des éléments sont contenus dans un ensemble. Vous devriez plutôt utiliser la fonction CONTAINS, car elle est convertie en SQL optimisé lorsqu'elle est transmise à la base de données :

```
CONTAINS([Product Name],[Product Lookup])
```

Dans ce cas cependant, la meilleure solution serait de ne pas utiliser de champ calculé, mais de proposer un filtre permettant d'effectuer une recherche avec des caractères génériques.

Paramètres pour les calculs conditionnels

Une technique courante dans Tableau consiste à mettre un paramètre à la disposition de l'utilisateur final pour lui permettre de sélectionner une valeur qui déterminera comment un calcul est effectué. Imaginons par exemple que vous laissiez l'utilisateur final contrôler le niveau d'agrégation des dates dans une vue en effectuant une sélection dans une liste de valeurs possibles. Beaucoup créeraient un paramètre de chaîne :

```
Value
Year
Quarter
Month
Week
Day
```

Puis l'utiliseraient dans un calcul comme ceci :

```
CASE [Parameters].[Date Part Picker]
   WHEN "Year" THEN DATEPART('year',[Order Date])
   WHEN "Quarter THEN DATEPART('quarter',[Order Date])
..
END
```

Il serait plus pertinent d'utiliser la fonction intégrée DATEPART() et de créer le calcul comme suit :

```
DATEPART(LOWER([Parameters].[Date Part Picker]), [Order Date]))
```

Dans les versions précédentes de Tableau, cette seconde approche était bien plus rapide. Néanmoins, ces deux solutions procurent les mêmes performances, car la logique conditionnelle de l'instruction CASE a été réduite en fonction de la valeur du paramètre, et seul l'élément DATEPART() approprié est transmis à la source de données.

N'oubliez pas que la pérennité de votre solution compte également. Dans cette optique, le second calcul serait certainement le plus approprié.

Conversion de date

Les utilisateurs disposent souvent de données de date qui ne sont pas stockées dans des formats de date en natif, mais par exemple comme chaînes ou horodatages numériques. La fonction DATEPARSE() permet d'effectuer ces conversions facilement. Vous pouvez simplement fournir une chaîne de mise en forme :

```
DATEPARSE("yyyyMMdd", [YYYYMMDD])
```

Notez que la fonction DATEPARSE() est prise en charge uniquement sur un sous-ensemble des sources de données :

- Connexions à des fichiers Excel et texte récents
- MySQL
- Oracle

- PostgreSQL
- Extrait de données Tableau

Si DATEPARSE() n'est pas prise en charge pour votre source de données, vous pouvez analyser le champ et le transformer en chaîne exprimant une date, par exemple « 2012-01-01 ». Notez que les chaînes ISO sont préférables, car elles sont compatibles avec l'internationalisation. Ensuite, fournissez le résultat dans la fonction DATE().

Si les données d'origine sont un champ numérique, la conversion en chaîne, puis en date, n'est vraiment pas efficace. Il vaut mieux garder les données sous leur forme numérique, et utiliser DATEADD() et des valeurs littérales pour effectuer le calcul.

Voici un exemple de calcul lent (conversion d'un champ numérique au format ISO) :

```
DATE(LEFT(STR([YYYYMMDD]),4)
   + "-" + MID(STR([YYYYMMDD]),4,2)
   + "-" + RIGHT(STR([YYYYMMDD]),2))
```

Il serait plus efficace d'exécuter le calcul ainsi :

```
DATEADD('day', INT([yyyymmdd])% 100 - 1,
   DATEADD('month', INT([yyyymmdd]) % 10000 / 100 - 1,
       DATEADD('year', INT([yyyymmdd]) / 10000 - 1900,
       #1900-01-01#)))
```

Ou mieux encore, utilisez la fonction MAKEDATE() si elle est prise en charge dans votre source de données :

```
MAKEDATE(2004, 4, 15)
```

Notez que les gains de performances sont très perceptibles avec les ensembles de données volumineux. Avec un échantillon de plus d'un milliard d'enregistrements, le premier calcul a pris 4 h, alors que le second a pris environ une minute.

Déclarations logiques

Testez le résultat le plus fréquent en premier

Lorsque Tableau évalue un test logique, il arrête de chercher les différents résultats possibles dès qu'il trouve une correspondance. Cela signifie que vous devez tester le résultat le plus probable en premier, pour que la majorité des cas évalués arrêtent la recherche après le premier test.

Observez l'exemple suivant. La logique suivante :

```
IF <unlikely_test>
   THEN <unlikely_outcome>
   ELSEIF <likely_test>
    THEN <likely_outcome>
END
```

Sera plus lente à évaluer que celle-ci :

```
IF <likely_test>
   THEN <likely_outcome>
   ELSEIF <unlikely_test>
    THEN <unlikely_outcome>
END
```

ELSEIF > ELSE IF

Lorsque vous utilisez des déclarations logiques complexes, gardez à l'esprit que ELSEIF > ELSE IF. Cela est dû au fait qu'un élément IF imbriqué calcule une déclaration CASE imbriquée dans la requête résultante, et non dans le cadre de la première. Le champ calculé suivant :

```
IF [Region] = "East" AND [Segment] = "Consumer"
   THEN "East-Consumer"
   ELSE IF [Region] = "East" and [Segment] <> "Consumer"
    THEN "East-All Others"
   END
END
```

Produit le code SQL suivant avec deux déclarations CASE imbriquées et 4 tests conditionnels :

```
(CASE
   WHEN (([Global Superstore].[Region] = 'East')
   AND ([Global Superstore].[Segment] = 'Consumer'))
      THEN 'East-Consumer'
      ELSE (CASE
       WHEN (([Global Superstore].[Region] = 'East')
       AND ([Global Superstore].[Segment] <> 'Consumer'))
        THEN 'East-All Others'
        ELSE CAST(NULL AS NVARCHAR)
      END)
END)
```

Ce calcul serait plus rapide s'il était réécrit avec ELSEIF au lieu de l'élément IF imbriqué, et il utiliserait les tests conditionnels plus efficacement :

```
IF [Region] = "East" AND [Segment] = "Consumer"
   THEN "East-Consumer"
   ELSEIF [Region] = "East”
    THEN "East-All Others"
END
```

Vous obtenez ainsi une seule déclaration CASE dans le code SQL :

```
(CASE
   WHEN (([Global Superstore].[Region] = 'East')
   AND ([Global Superstore].[Segment] = 'Consumer'))
    THEN 'East-Consumer'
   WHEN ([Global Superstore].[Region] = 'East')
    THEN 'East-All Others'
   ELSE CAST(NULL AS NVARCHAR)
END)
```

Néanmoins, le calcul suivant est plus rapide. Bien qu'il utilise une déclaration IF imbriquée, il optimise l'utilisation des tests conditionnels :

```
IF [Region] = "East" THEN
   IF [Segment] = "Consumer"
    THEN "East-Consumer"
    ELSE "East-All Others"
   END
END
```

Voici le code SQL qui en résulte :

```
(CASE
   WHEN ([Global Superstore].[Region] = 'East')
   THEN (CASE
    WHEN ([Global Superstore].[Segment] = 'Consumer')
       THEN 'East-Consumer'
       ELSE 'East-All Others'
   END)
```

```
   ELSE CAST(NULL AS NVARCHAR)
END)
```

Évitez les vérifications de logique redondantes

Évitez aussi les vérifications de logique redondantes. Le calcul suivant :

```
IF [Sales] < 10 THEN "Bad"
   ELSEIF[Sales]>= 10 AND [Sales] < 30 THEN "OK"
   ELSEIF[Sales] >= 30 THEN "Great"
END
```

Génère le code SQL suivant :

```
(CASE
   WHEN ([Global Superstore].[Sales] < 10)
    THEN 'Bad'
   WHEN (([Global Superstore].[Sales] >= 10)
   AND ([Global Superstore].[Sales] < 30))
    THEN 'OK'
   WHEN ([Global Superstore].[Sales] >= 30)
    THEN 'Great'
   ELSE CAST(NULL AS NVARCHAR)
END)
```

Il serait plus efficace ainsi :

```
IF [Sales] < 10 THEN "Bad"
   ELSEIF[Sales] >= 30 THEN "Great"
   ELSE "OK"
END
```

Et générerait le code SQL suivant :

```
(CASE
   WHEN ([Global Superstore].[Sales] < 10)
    THEN 'Bad'
   WHEN ([Global Superstore].[Sales] >= 30)
    THEN 'Great'
   ELSE 'OK'
END)
```

Quelques conseils supplémentaires

Beaucoup d'autres petits détails peuvent avoir un impact sur les performances. Jugez plutôt :

- Il peut être compliqué de gérer la logique des dates. Au lieu d'écrire des tests élaborés en utilisant plusieurs fonctions de date, comme MONTH() et YEAR(), pensez aux autres fonctions intégrées, comme DATETRUNC(), DATEADD() et DATEDIFF(). Elles permettent de réduire de manière significative la complexité des requêtes générées vers la source de données.
- Les valeurs des totaux distincts constituent l'un des types d'agrégation les plus lents pour presque toutes les sources de données. Utilisez l'agrégation COUNTD avec parcimonie.
- L'utilisation de paramètres avec un impact très large, par exemple dans une déclaration SQL personnalisée, peut affecter les performances du cache.
- Le filtrage de calculs complexes peut avoir pour effet que plusieurs index seront ignorés dans la source de données sous-jacente.
- Utilisez NOW() uniquement si vous avez besoin du niveau de détail d'horodatage. Utilisez TODAY() pour les calculs au niveau des dates.
- Gardez à l'esprit que tous les calculs de base sont répercutés sur la source de données sous-jacente, même les calculs littéraux comme les chaînes d'étiquette. Si vous avez besoin de créer des étiquettes, par exemple pour les en-têtes de colonnes, et que votre source

de données est très volumineuse, créez une simple source de données Excel ou texte ne comportant qu'un seul enregistrement destiné à stocker ces étiquettes, pour éviter de surcharger la grande source de données. C'est particulièrement important si votre source de données utilise des procédures stockées. Reportez-vous à cette section pour en savoir plus.

Les requêtes sont-elles en cause ?

Ce qui fait la force de Tableau, c'est qu'il peut utiliser des données en mémoire (les connexions de données extraites), ainsi que des données en situation (des connexions de données en direct). Les connexions en direct sont une fonctionnalité très puissante et permettent de tirer parti de la puissance de calcul existant déjà dans la source de données. Néanmoins, étant donné que vous dépendez des performances de la plate-forme des données source, il est indispensable d'optimiser l'efficacité des requêtes exécutées sur cette source.

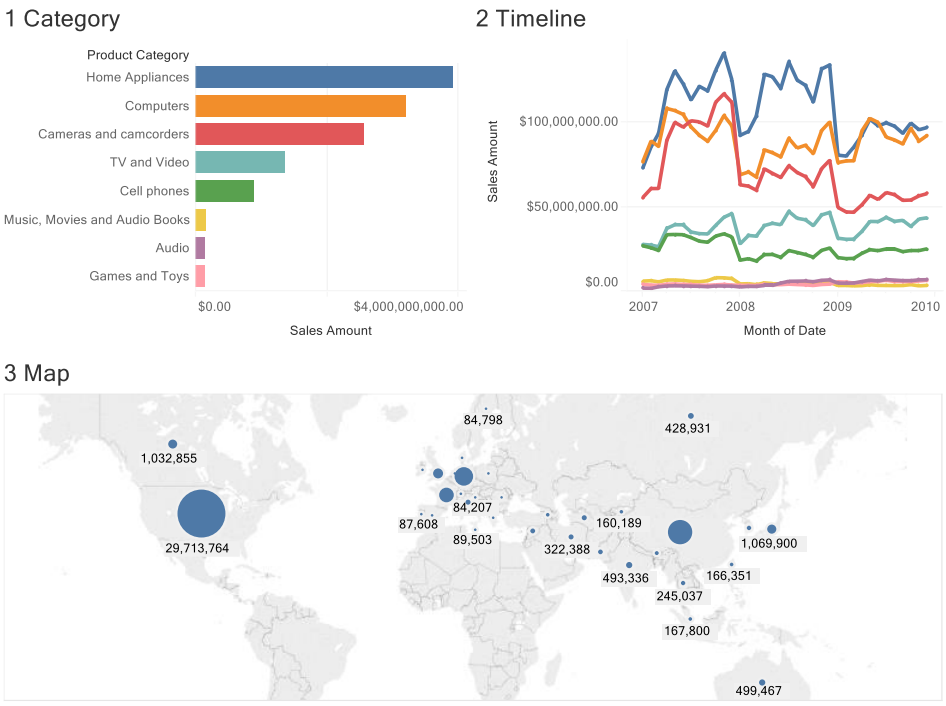
Comme nous l'avons vu, plus la question est complexe, plus les données à afficher sont nombreuses, et plus vous incluez d'éléments dans vos tableaux de bord. Tout cela se traduit par une charge accrue sur la source de données. Pour rendre vos classeurs aussi efficaces que possible, vous devez réduire le nombre de requêtes, faire en sorte qu'elles soient évaluées aussi efficacement que possible, réduire la quantité de données qu'elles renvoient et réutiliser autant que possible les données d'une requête à l'autre.

Les optimisations automatiques

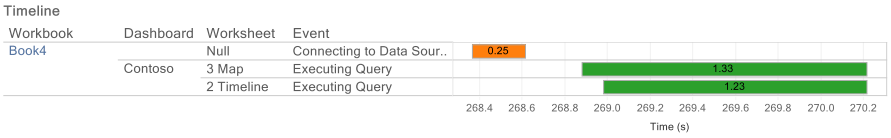
Il existe de nombreuses manières d'améliorer soi-même les performances des requêtes, cependant Tableau effectue de nombreuses optimisations automatiquement pour assurer l'efficacité des classeurs. Il s'agit généralement de ce que vous ne pouvez pas contrôler, et dans la plupart des cas vous ne devriez pas avoir à y penser, mais le fait de comprendre ces aspects vous permettra de mieux en tirer parti pour améliorer les performances.

L'exécution de requêtes en parallèle

Tableau exploite la capacité des bases de données source à exécuter plusieurs requêtes à la fois. Observez le tableau de bord suivant.



L'ouverture de ce classeur dans Tableau implique les requêtes suivantes :



Comme vous pouvez le voir, l'exécution successive de ces requêtes dure 2,66 secondes. Avec une exécution en parallèle, la durée totale correspond au temps que prend la requête la plus longue, à savoir 1,33 seconde.

Le degré de parallélisme varie selon les systèmes source, car certaines plates-formes savent mieux gérer les requêtes simultanées que d'autres. En général, 16 requêtes parallèles au maximum sont exécutées pour toutes les sources de données, à part les fichiers texte et Excel et les fichiers statistiques (limités à une requête à la fois). Pour certaines sources de données, la limite est inférieure à 16. Reportez-vous à la page suivante pour en savoir plus :

> http://kb.tableau.com/articles/HowTo/Configuring-Parallel-Queries-in-Tableau-Desktop?lang=fr-fr

La plupart du temps, vous n'avez pas besoin de modifier ces paramètres et vous devriez conserver leurs valeurs par défaut. Néanmoins, si vous devez contrôler précisément le degré de parallélisme, vous pouvez définir :

- Une limite globale du nombre de requêtes parallèles pour Tableau Server
- Des limites pour un type de source de données particulier, comme SQL Server
- Des limites pour un type de source de données particulier sur un serveur spécifique
- Des limites pour un type particulier de source de données, sur un serveur spécifique, lors de la connexion à une base de données spécifique

Ces paramètres sont gérés par le fichier connection-configs.xml, que vous créez et enregistrez dans le dossier de l'application sous Windows (`C:\Program Files\Tableau\Tableau 10.0`) et sous Mac (cliquez avec le bouton droit sur l'application, cliquez sur Afficher le contenu du paquet, puis placez le fichier à cet emplacement) pour Tableau Desktop, ou dans le répertoire de configuration du dossier vizqlserver (par exemple : `C:\ProgramData\Tableau\TableauServer\data\tabsvc\config\vizqlserver`) pour Tableau Server. Vous devez copier ce fichier de configuration dans tous les répertoires de configuration vizqlserver sur tous les ordinateurs des utilisateurs.

Pour en savoir plus sur la configuration des requêtes parallèles, y compris sur la syntaxe du fichier connection-configs.xml, reportez-vous à la page suivante :

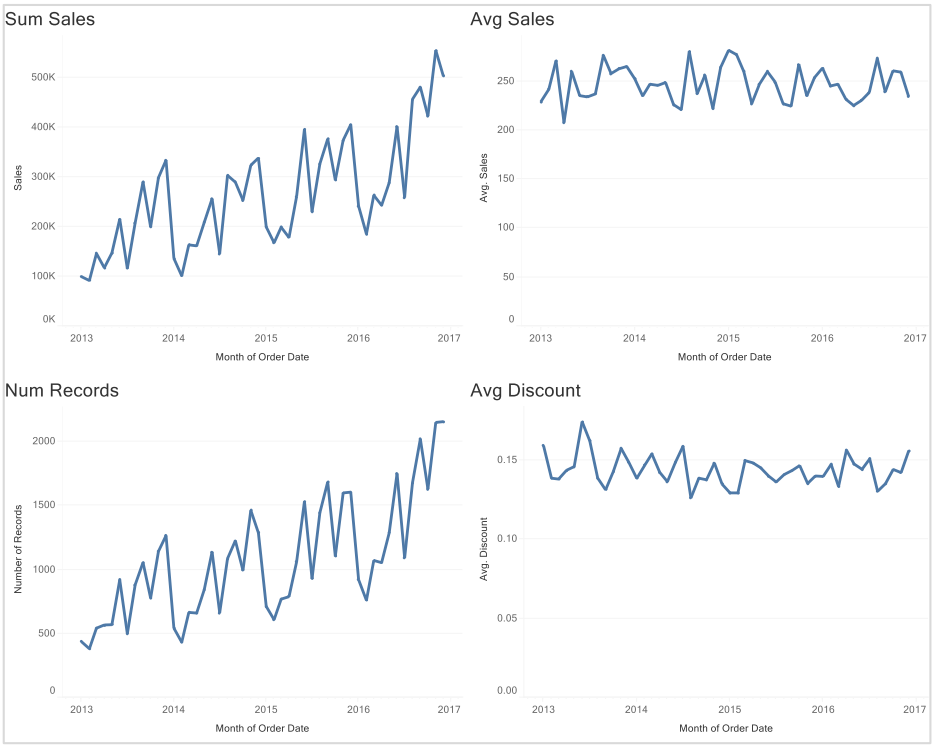
> http://kb.tableau.com/articles/knowledgebase/parallel-queries-tableau-server?lang=fr-fr

Réduction du nombre de requêtes exécutées

Vous pouvez voir dans l'exemple ci-dessus que vous avez exécuté deux requêtes seulement, au lieu de trois. En combinant les requêtes, Tableau peut éliminer celles qui sont redondantes. L'outil d'optimisation de Tableau trie les requêtes de sorte à exécuter les plus complexes en premier. Normalement, les suivantes peuvent être traitées à partir du cache des résultats. Dans cet exemple, étant donné que la chronologie inclut la catégorie de produit et que l'agrégation SUM du montant des ventes est entièrement additive, les données du graphique 1 (avec les catégories) peuvent être

obtenues à partir du cache des requêtes de la feuille Timeline. Il n'est donc pas nécessaire d'interroger la source de données.

L'outil d'optimisation recherche également les requêtes dont le niveau de détail est le même (elles sont spécifiées par le même ensemble de dimensions) et les combine ensuite en une requête unique qui renverra toutes les mesures demandées. Observez le tableau de bord suivant.



Ce tableau de bord contient quatre feuilles, affichant chacune une mesure différente dans le temps. Elles utilisent le même niveau de détail, car elles présentent les données sous forme de mois continus. Au lieu d'exécuter quatre requêtes différentes sur la source de données, Tableau les combine en une requête unique :

La requête est la suivante :

```
SELECT AVG(cast([Global Superstore].[Discount] as float)) AS
[avg:Discount:ok],
   AVG(cast([Global Superstore].[Sales] as float)) AS [avg:Sales:ok],
   SUM(CAST(1 as BIGINT)) AS [sum:Number of Records:ok],
   SUM([Global Superstore].[Sales]) AS [sum:Sales:ok],
   DATEADD(month, DATEDIFF(month, CAST('0001-01-01 00:00:00' AS datetime2),
[Global Superstore].[Order Date]), CAST('0001-01-01 00:00:00' AS datetime2))
AS [tmn:Order Date:ok]
FROM [dbo].[Global Superstore] [Global Superstore]
GROUP BY DATEADD(month, DATEDIFF(month, CAST('0001-01-01 00:00:00' AS
datetime2), [Global Superstore].[Order Date]), CAST('0001-01-01 00:00:00' AS
datetime2))
```

Comme vous pouvez le constater, cette optimisation, que l'on appelle « fusion de requêtes », permet d'améliorer notablement les performances globales. Lorsque c'est possible, vous devez envisager de définir le même niveau de détail pour plusieurs feuilles sur un tableau de bord.

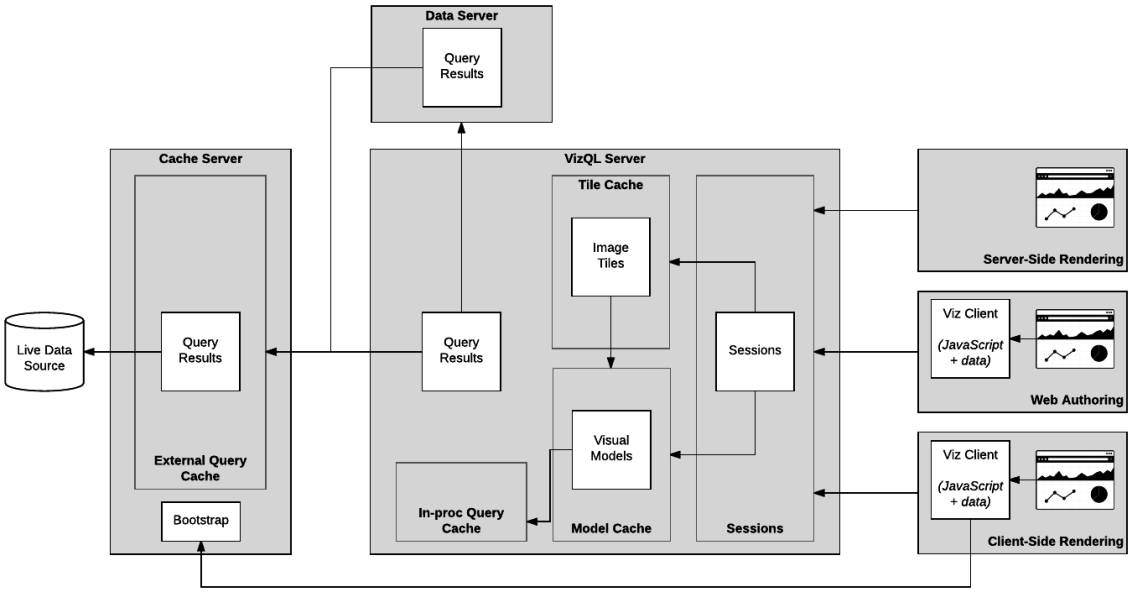
Notez que les requêtes sur les sources de données qui sont des extraits de données Tableau ne sont pas fusionnées.

Mise en cache : aucune requête n'est exécutée

Quoi de mieux que d'exécuter moins de requêtes ? N'en exécuter aucune bien sûr ! Tableau Server et Tableau Desktop proposent une mise en cache extensive, ce qui permet de réduire notablement le nombre de requêtes à exécuter sur la source de données sous-jacente.

Tableau Server

Dans Tableau Server, la mise en cache comporte plusieurs couches :



La première couche de mise en cache dépend du type de rendu utilisé par la session (côté client ou côté serveur). Reportez-vous à la section correspondante pour en savoir plus sur ces deux modèles de rendu et sur la manière dont Tableau détermine lequel utiliser.

Si la session utilise le rendu côté client, le navigateur doit tout d'abord charger le client du visualiseur. Il faut pour cela charger un ensemble de bibliothèques JavaScript et les données, pour effectuer le rendu de la vue initiale, procédure appelée « bootstrapping », ce qui peut prendre plusieurs secondes. En sachant que les tableaux de bord seront certainement consultés par plusieurs ordinateurs, les réponses liées à cette opération sont mises en cache afin de réduire le délai d'attente pour les demandes de consultation suivantes. Chaque session consulte d'abord le cache pour vérifier si une réponse a déjà été créée et peut être réutilisée. Le cas échéant, le navigateur charge simplement cette réponse depuis le cache, ce qui accélère grandement le rendu de la vue initiale. Une fois que le client du visualiseur est chargé dans le navigateur client, certaines interactions, telles que les sélections ou les infobulles, peuvent être effectuées intégralement avec les données locales, ce qui accélère et améliore l'expérience utilisateur.

Si le tableau de bord est affiché via un rendu côté serveur, le serveur restitue les éléments sous forme de séries de fichiers image (appelés « tuiles »). Ces tuiles sont transmises au navigateur, qui les assemble pour afficher la visualisation. Comme précédemment, nous estimons que ces tableaux de bord seront consultés par de nombreux utilisateurs, de sorte que le serveur met les images en cache sur le disque. Chaque requête consulte le cache des tuiles pour vérifier si le rendu de l'image a déjà été effectué et, le cas échéant, charge simplement le fichier depuis le cache. Si la

consultation du cache donne un résultat, le délai de réponse sera accéléré, ce qui réduira la charge sur le serveur. Il est possible d'améliorer l'efficacité du cache des tuiles en définissant une taille fixe pour vos tableaux de bord.

Dans l'ensemble, le rendu côté client produit des interactions plus fluides et réactives, et allège la charge de Tableau Server. Lorsque c'est possible, faites en sorte que vos classeurs utilisent le rendu côté client.

La couche suivante est le modèle visuel. Un modèle visuel décrit la procédure du rendu d'une feuille individuelle. Ainsi, la consultation d'un tableau de bord peut faire référence à plusieurs modèles visuels, un pour chaque feuille utilisée. Il inclut les résultats des calculs locaux (par exemple les calculs de table, les lignes de référence, les prévisions ou les clusters), la disposition visuelle (le nombre de lignes/colonnes à afficher pour les séries de petits graphiques et les tableaux croisés, le nombre de graduations et lignes de grille à tracer et leur intervalle, ou encore le nombre d'étiquettes de repères à afficher et leur emplacement).

Les modèles visuels sont créés et mis en cache dans la mémoire par VizQL Server, qui partage les résultats avec toutes les sessions utilisateur dans la mesure du possible. Voici les principaux facteurs qui déterminent si un modèle visuel peut être partagé :

- Taille de la zone d'affichage de la visualisation : les modèles peuvent uniquement être partagés dans des sessions utilisant la même taille de visualisation. Le fait de définir une taille fixe pour vos tableaux de bord avantage aussi bien le cache des modèles que le cache des tuiles, dans la mesure où cela permet de réutiliser les données et de réduire la charge sur le serveur.
- Concordance des sélections/filtres : si l'utilisateur modifie les filtres, les paramètres ou explore les données, le modèle est partagé uniquement avec les sessions qui ont la même configuration de vue. Évitez de publier des classeurs avec l'option « Afficher les sélections » cochée, pour ne pas réduire les chances de concordance entre différentes sessions.
- Informations de connexion à la source de données : si les utilisateurs sont invités à fournir des informations d'identification pour se connecter à la source de données, le modèle peut uniquement être partagé entre des sessions utilisateur utilisant les mêmes informations d'identification. Utilisez cette fonctionnalité avec prudence, car elle peut réduire l'efficacité du cache des modèles.
- Utilisation d'un filtre utilisateur : si le classeur contient des filtres utilisateur ou des calculs incluant des fonctions comme USERNAME() ou ISMEMBEROF(), le modèle n'est pas partagé avec les autres sessions utilisateur. Utilisez ces fonctionnalités avec prudence, car elles peuvent réduire considérablement l'efficacité du cache des modèles.

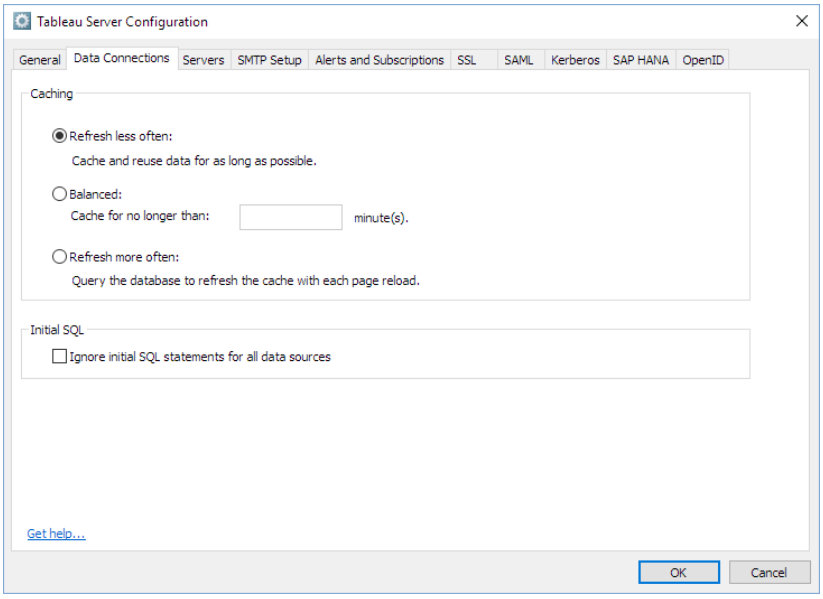
La dernière couche de mise en cache est le cache des requêtes. Il stocke les résultats des requêtes exécutées en prévision de leur réutilisation pour traiter les requêtes suivantes. L'utilisation de ce cache est très efficace et permet d'éviter la répétition des requêtes dans la source de données. Il suffit d'utiliser les données du cache. Dans certains cas, ce cache permet également de traiter des requêtes en utilisant les résultats d'autres requêtes. Nous avons abordé dans une section précédente les avantages du cache dans le cadre de la réduction du nombre de requêtes et de la fusion des requêtes.

Ce cache se compose de deux parties : l'une est intégrée au processus VizQL (cache interne) et l'autre est partagée entre plusieurs processus via le serveur du cache (cache externe). Si une requête est envoyée à une instance VizQL qui l'a déjà exécutée précédemment, elle est traitée à partir du cache interne. Notez que ce cache est local pour chaque processus VizQL et conservé en mémoire.

En plus de ces caches internes aux processus, il y a un cache externe commun, utilisé non seulement dans toutes les instances VizQL, mais aussi dans TOUS les processus qui ont accès à la source de données sous-jacente, par exemple les serveurs de tâches d'arrière-plan, les serveurs de données, etc.
Le serveur de cache est un service qui gère le cache externe pour l'ensemble du cluster. Notez que toutes les requêtes ne sont pas écrites dans le cache externe. Si une requête s'exécute très rapidement, sa réexécution peut être plus rapide que la consultation du cache. Il existe donc un seuil minimal pour le délai d'exécution. Par ailleurs, si une requête produit un résultat très volumineux, il ne sera pas forcément efficace de l'écrire dans le serveur du cache. Il existe donc également un seuil pour la taille maximale.

Le cache externe améliore nettement l'efficacité de la mise en cache pour les déploiements comprenant plusieurs instances VizQL. À l'inverse des caches internes qui sont volatiles, le cache externe est persistant (le serveur du cache écrit les données sur le disque) et ne disparaît pas entre les instanciations des services.

Pour maximiser l'efficacité des caches, vous pouvez ajuster les paramètres de configuration de Tableau Server pour conserver ces caches le plus longtemps possible.



Vous pouvez augmenter la taille des caches si les capacités de votre serveur le permettent. Vous n'êtes pas pénalisé si vous indiquez des valeurs supérieures. Aussi, si vous disposez de suffisamment de RAM, vous pouvez augmenter ces valeurs de manière significative pour éviter que le contenu soit retiré du cache. Vous pouvez surveiller l'efficacité du cache grâce aux fichiers journaux ou à l'aide de TabMon.

- **Cache des modèles** : par défaut, 60 modèles sont mis en cache par instance VizQL. Vous pouvez régler ce paramètre en utilisant la commande *tabadmin*. Si vous disposez de suffisamment de RAM, vous pouvez augmenter cette valeur pour l'aligner sur le nombre de vues publiées sur le serveur :
  - tabadmin set vizqlserver.modelcachesize 200
- **Cache interne** : par défaut, 512 Mo de résultats de requêtes sont mis en cache par instance VizQL. Cela peut paraître peu, mais gardez à l'esprit que vous mettez en cache les résultats de requêtes agrégées. Vous pouvez ajuster ce paramètre en utilisant la commande *tabadmin* :
  - tabadmin set vizqlserver.querycachesize 1024

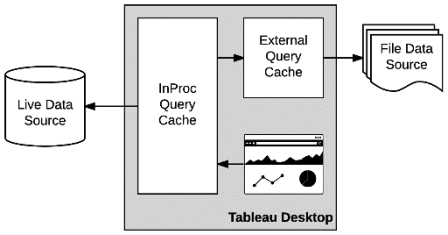
Vous pouvez également augmenter la capacité et le débit du cache de requêtes externe en ajoutant des instances du serveur de cache. Il est recommandé d'exécuter une instance de serveur de cache pour chaque instance VizQL Server.

- **Cache externe** : par défaut, 512 Mo de résultats de requêtes sont mis en cache pour chaque instance du serveur de cache.

Enfin, l'un des avantages du partage du cache, c'est que vous pouvez le remplir à l'avance pour les classeurs qui sont lents lors de leur première consultation, grâce à un abonnement. Si cette opération est réalisée tôt le matin avant que vos employés commencent à travailler, cela vous donne l'assurance que le classeur est prêt et que les résultats des requêtes sont dans le cache. En supposant que les résultats ne sont pas supprimés du cache ou n'expirent pas entre-temps, les utilisateurs consultant le classeur tirent profit du cache, ce qui accélère la consultation initiale.

Tableau Desktop

La mise en cache dans Tableau Desktop est beaucoup plus simple que dans Tableau Server, dans la mesure où il s'agit d'une application mono-utilisateur. Vous n'avez donc pas à gérer plusieurs sessions. Seul le cache des requêtes est présent dans Tableau Desktop.



À l'instar de Tableau Server, Tableau Desktop utilise un cache interne et un cache externe. Le cache interne, situé dans la mémoire et volatile, est utilisé pour les connexions à toutes les sources de données, alors que le cache externe est utilisé uniquement pour les sources de données basées sur des fichiers tels que les extraits de données ou les fichiers Excel, Access, texte, etc. Comme dans Tableau Server, le cache externe est persistant. Autrement dit, les résultats qu'il stocke sont conservés d'une session Tableau Desktop à l'autre. Si vous utilisez une source de données de type fichier, redémarrez Tableau Desktop et réutilisez cette source, et vous bénéficierez toujours du cache.

Le cache externe est stocké dans le dossier `%LOCALAPPDATA%\Tableau\Caching` sous Windows et `~/Library/Caches/com.tableau/` sous Mac. Il est limité à environ 750 Mo par défaut, et devient inutile si l'utilisateur actualise manuellement la source de données (par exemple en appuyant sur F5, ⌘R).

Enfin, dans Tableau Desktop, les données du cache de requêtes externe sont incluses au fichier lorsqu'il est enregistré en tant que classeur complet. De cette façon, Tableau Desktop et Tableau Reader peuvent effectuer rapidement le rendu de la vue initiale du classeur, tout en poursuivant la décompression du fichier de la source de données, plus volumineux, en arrière-plan. Cela permet d'améliorer fortement la réactivité de l'ouverture du classeur. Notez que les données du cache sont incluses uniquement si elles ne représentent pas plus de 10 % de la taille du fichier de la source de données, et les requêtes utilisant des filtres Date relative sont exclues.

Optimisation des connexions aux sources de données

Avant Tableau 10, lorsqu'un utilisateur ouvrait un classeur comportant plusieurs sources de données, la connexion s'établissait initialement avec toutes celles qui ne nécessitaient pas

l'identification avec un nom d'utilisateur et un mot de passe. Il fallait donc parfois du temps pour se connecter à des sources de données qui n'étaient pas utilisées par la feuille, le tableau de bord ou l'histoire consultés.

Tableau 10 se connecte désormais à une source de données uniquement si l'utilisateur en a besoin pour consulter ces documents. Ainsi, un classeur avec une vue à onglets (dans Tableau Desktop) s'ouvre plus vite, et les utilisateurs peuvent commencer à explorer leurs données plus rapidement.

Les jointures

Généralement, si vous utilisez plusieurs tables dans Tableau, il est conseillé de définir les jointures dans la fenêtre de connexion de données. Dans ce cas, vous ne définissez pas une requête spécifique, mais plutôt la manière dont les tables sont liées les unes aux autres. Lorsque vous faites glisser des champs dans une visualisation, Tableau utilise ces informations pour générer la requête nécessaire et récupérer uniquement les données requises.

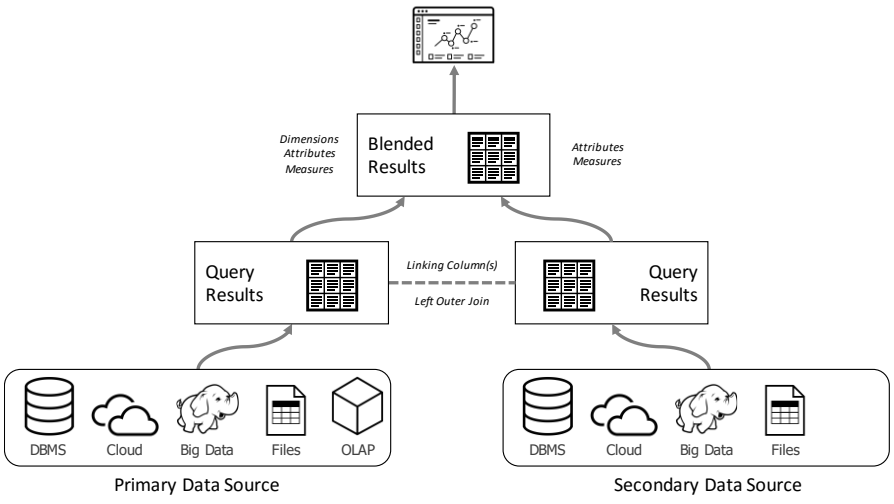
La règle d'or en matière de performances consiste à réduire autant que possible le nombre de tables jointes et d'inclure uniquement les tables nécessaires à une feuille de calcul ou une visualisation spécifiques. En fonction de la complexité de la connexion de données et des feuilles de calcul, il peut également être utile de séparer les connexions de données par feuille et de créer des jointures spécifiques pour ces feuilles.

Les jointures sont plus efficaces lorsque les limites entre les tables sont claires. Avec ces limites, Tableau Desktop peut simplifier les requêtes et les appliquer uniquement aux tables nécessaires pour répondre à certains éléments d'une question (légendes, filtres).

La fusion

Au moment de choisir entre la jointure et la fusion des tables de données dans Tableau, tenez compte de la provenance des données, du nombre de connexions de données et du nombre d'enregistrements.

Lorsque vous utilisez la fusion, la cardinalité des champs fusionnés liant les deux ensembles influence davantage les performances que le nombre d'enregistrements de chaque source de données. La fusion envoie une requête sur les données des deux sources au niveau des champs de liaison, puis fusionne les résultats des deux requêtes dans la mémoire locale de Tableau.



Si les valeurs uniques sont nombreuses, cela peut nécessiter une grande quantité de mémoire. Il est recommandé d'utiliser la version 64 bits de Tableau Desktop lorsque vous effectuez une fusion de données, pour éviter de manquer de mémoire. Néanmoins, le temps de calcul de ce type de fusion risque tout de même d'être long.

En matière de fusion, il est recommandé d'éviter de fusionner des dimensions comportant de nombreuses valeurs uniques (ID de commande, ID de client, date/heure précises, etc.)

Groupes et alias principaux

Si vous devez fusionner deux sources de données car l'une d'elles contient les enregistrements factuels et l'autre les attributs dimensionnels, vous pouvez peut-être améliorer les performances en créant un groupe ou alias principal.

Des groupes ou des alias sont créés dans la source de données principale pour refléter des attributs listés dans la source de données secondaire. Les groupes principaux sont utilisés pour des relations de type 1:plusieurs, alors que les alias principaux sont utilisés pour des relations 1:1. Vous pouvez ensuite utiliser l'objet approprié dans la source de données principale, de sorte que la fusion n'est plus nécessaire au moment de la consultation.

Pour les exemples suivants, nous allons utiliser les trois tables suivantes.

ID	Value
A	89
B	94
C	74
D	88
E	58
F	89
G	95

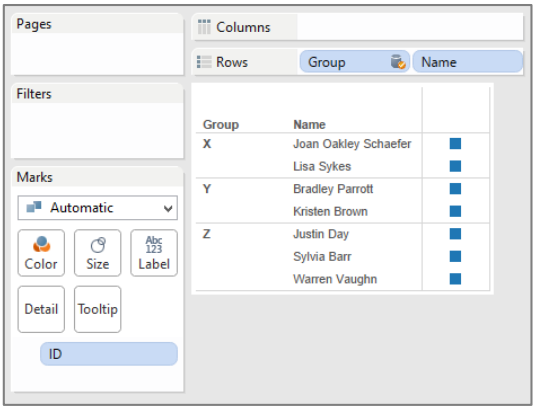
ID	Name
A	Lisa Sykes
B	Joan Oakley Schaefer
C	Bradley Parrott
D	Kristen Brown
E	Sylvia Barr
F	Warren Vaughn
G	Justin Day

ID	Group
A	X
B	X
C	Y
D	Y
E	Z
F	Z
G	Z

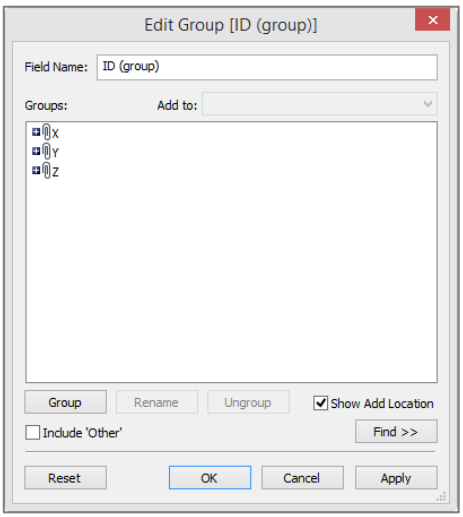
Les groupes principaux sont utiles lorsque la source de données secondaire contient un attribut du type 1:plusieurs associé à des membres de dimension de la source de données principale. Imaginons que vous souhaitiez produire le résultat suivant à partir des données ci-dessus.

Group	Name
X	Lisa Sykes
	Joan Oakley Schaefer
Y	Bradley Parrott
	Kristen Brown
Z	Sylvia Barr
	Warren Vaughn
	Justin Day

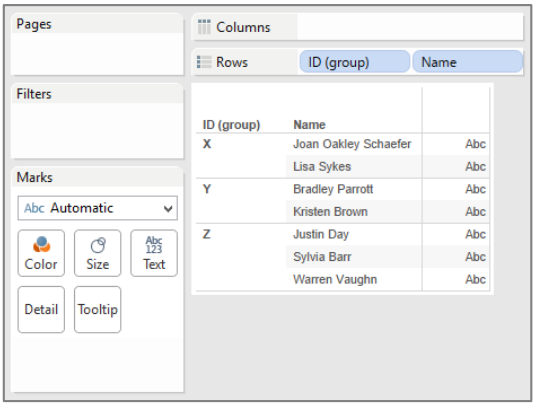
Vous pouvez recourir à une fusion, mais comme nous l'avons déjà vu, cela ralentirait beaucoup les performances si le nombre d'ID est important.



Si vous cliquez avec le bouton droit sur le champ Groupe et sélectionnez « Créer un groupe principal », Tableau crée un objet groupe dans la source de données principale. Cet objet associe le champ de liaison (l'ID, dans ce cas) à la dimension de la source de données secondaire sélectionnée (Groupe).



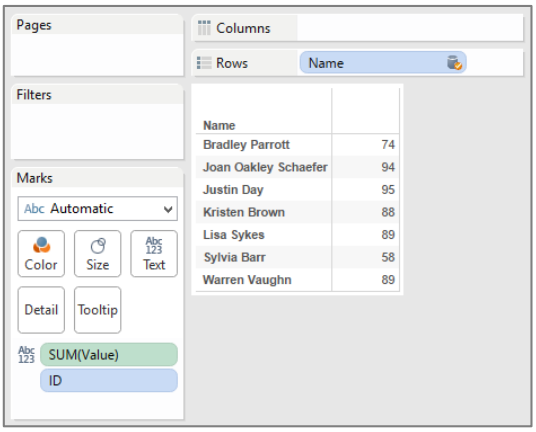
Vous pouvez maintenant recréer cette table sans avoir besoin d'une fusion.



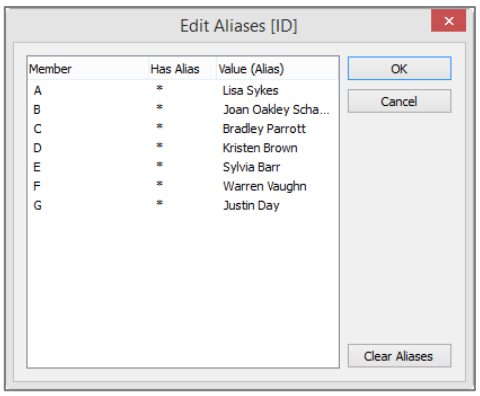
Les alias principaux sont utiles lorsque la source de données secondaire contient un attribut du type 1:plusieurs associé à des membres de dimension de la source de données principale. Imaginons que vous souhaitiez produire le résultat suivant à partir des données ci-dessus.

Name	Value
Lisa Sykes	89
Joan Oakley Schaefer	94
Bradley Parrott	74
Kristen Brown	88
Sylvia Barr	58
Warren Vaughn	89
Justin Day	95

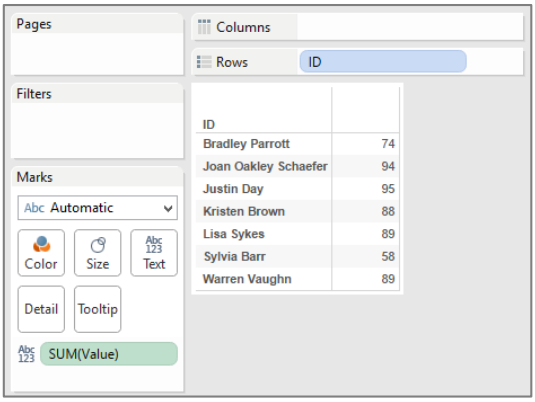
Vous pouvez recourir à une fusion entre les deux sources, mais comme nous l'avons déjà vu, cela ralentirait beaucoup les performances si le nombre d'ID est important.



Si vous cliquez avec le bouton droit sur le champ Nom et sélectionnez « Modifier les alias principaux », vous pouvez relier ponctuellement le nom à l'ID, avec des valeurs d'alias.



Vous pouvez maintenant créer la visualisation requise sans fusion, ce qui est bien plus rapide.



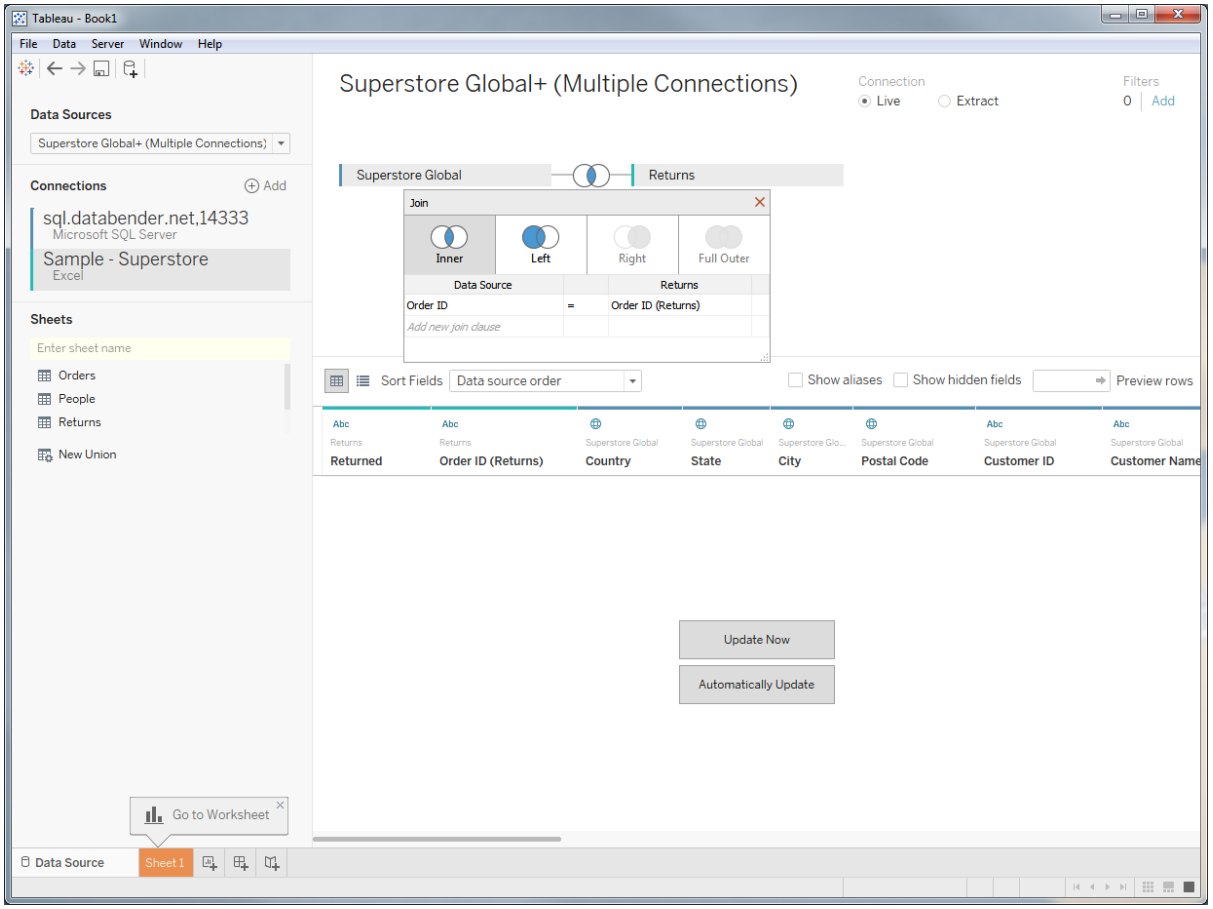
Notez que les groupes et les alias principaux ne sont pas dynamiques et doivent être mis à jour si les données changent. Par conséquent, ils ne sont pas utiles pour les données fréquemment mises à jour, mais si vous avez besoin d'un mappage rapide, ils peuvent vous éviter d'utiliser des fusions peu pratiques.

Pour en savoir plus, reportez-vous à cette page :

> http://onlinehelp.tableau.com/current/pro/desktop/fr-fr/help.htm#multipleconnections_create_primary_group.html

L'intégration des données

L'intégration des données est une nouvelle fonctionnalité de Tableau 10. Les sources de données peuvent maintenant combiner des données issues de plusieurs connexions potentiellement hétérogènes.



L'une des principales différences entre l'intégration de données et la fusion de données est que l'intégration est une jointure au niveau des lignes, alors que la fusion est exécutée sur l'ensemble

de résultats agrégé de chaque source de données. Cela signifie que la taille des données originales compte pour l'intégration des données. Ne perdez pas de vue les 4 faits essentiels suivants :

- Plus le volume de données à déplacer est important, plus cela prend de temps : les données sont extraites au niveau des lignes pour chaque connexion de données, le volume est donc un facteur clé.
- Plus les données parcourent une longue distance, plus cela prend de temps : le fait d'effectuer une jointure avec une source de données dont la connexion a une latence élevée aura un impact sur les performances.
- Plus le flux de données est lent, plus cela prend de temps : le fait d'effectuer une jointure avec une source de données dont la connexion a une faible bande passante aura un impact sur les performances.
- Plus il y a d'éléments à faire correspondre, plus cela prend de temps : comme pour le premier point, le nombre d'enregistrements à relier peut avoir un impact sur les performances.

Dans Tableau 10, l'intégration des données peut uniquement être utilisée avec les sources de données extraites, c'est-à-dire que les résultats de chaque connexion de données doivent être extraits dans un fichier TDE. Cette source de données extraite peut ensuite être publiée sur Tableau Server où d'autres utilisateurs pourront l'exploiter.

Tableau 10 ne permet pas d'effectuer une intégration de données sur les sources de données en direct ni sur les sources de données publiées. Nous envisageons d'intégrer ces fonctionnalités dans les futures versions du produit.

Le SQL personnalisé

Les utilisateurs qui découvrent Tableau appliquent parfois des techniques désuètes à leurs classeurs, par exemple en créant des sources de données à l'aide d'instructions SQL qu'ils élaborent eux-mêmes. Cette pratique est souvent contre-productive, car Tableau peut générer des requêtes bien plus efficaces lorsque vous définissez simplement les relations de jointure entre les tables et laissez le moteur de requêtes écrire le code SQL pour la vue créée. Néanmoins, la définition des jointures dans la fenêtre de connexion de données n'offre parfois pas toute la flexibilité dont vous avez besoin pour définir les relations dans vos données.

La création d'une connexion de données à l'aide d'instructions SQL créées manuellement peut parfois se révéler très puissante, mais un grand pouvoir implique de grandes responsabilités, et ce n'est pas Peter Parker qui vous dira le contraire. Dans certains cas, le SQL personnalisé peut en réalité réduire les performances. En effet, le SQL personnalisé n'est jamais décomposé comme les jointures, et il est toujours exécuté selon le principe d'atomicité, c'est-à-dire entièrement ou pas du tout. Cela signifie que rien n'est éliminé et que vous pouvez vous retrouver dans une situation où la base de données doit traiter la requête entière pour une seule colonne, comme ici :

```
SELECT SUM([TableauSQL].[Sales])
FROM (
   SELECT [OrdersFact].[Order ID] AS [Order ID],
       [OrdersFact].[Date ID] AS [Date ID],
       [OrdersFact].[Customer ID] AS [Customer ID],
       [OrdersFact].[Place ID] AS [Place ID],
       [OrdersFact].[Product ID] AS [Product ID],
       [OrdersFact].[Delivery ID] AS [Delivery ID],
       [OrdersFact].[Discount] AS [Discount],
       [OrdersFact].[Cost] AS [Cost],
       [OrdersFact].[Sales] AS [Sales],
       [OrdersFact].[Qty] AS [Qty],
       [OrdersFact].[Profit] AS [Profit]
```

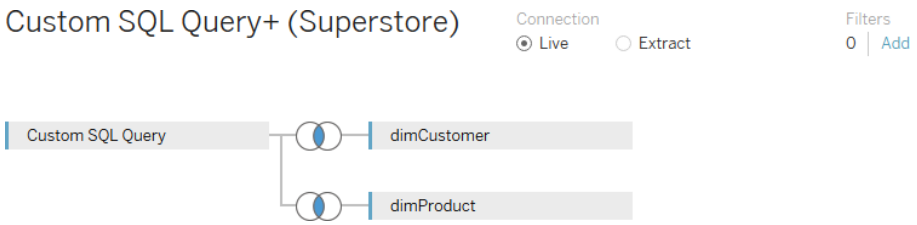
```
   FROM [dbo].[OrdersFact] [OrdersFact]
) [TableauSQL]
HAVING (COUNT_BIG(1) > 0)
```

Vérifiez bien que votre instruction SQL personnalisée ne contient pas de clauses superflues. Par exemple, si votre requête contient des clauses GROUP BY ou ORDER BY, celles-ci auront généralement un impact sur les performances, car Tableau va créer ses propres clauses en fonction de la structure de la visualisation. Si possible, éliminez ce type de clauses dans vos requêtes.

Il est recommandé d'utiliser le SQL personnalisé conjointement avec des extraits de données. De cette manière, la requête indivisible est exécutée une fois seulement, pour charger les données dans l'extrait de données. Le reste de l'analyse dans Tableau s'effectue avec des requêtes dynamiques et optimisées sur cet extrait de données. Bien entendu, cette règle a aussi ses exceptions : créez un extrait de données lorsque vous utilisez le SQL personnalisé, SAUF si votre code SQL personnalisé contient des paramètres.

Utiliser des paramètres dans une déclaration SQL personnalisée peut parfois rendre les connexions en direct plus performantes, car la requête de base peut être plus dynamique (les clauses de filtrage utilisant des paramètres seront évaluées correctement). Vous pouvez aussi l'utiliser pour transmettre des valeurs de limitation des performances, comme TOP ou SAMPLE, afin de restreindre la quantité de données renvoyées par la base de données. Néanmoins, si vous utilisez un extrait de données, celui-ci sera éliminé et renouvelé chaque fois que vous modifierez le paramètre, ce qui peut augmenter le temps de traitement. Notez également que les paramètres peuvent uniquement être utilisés pour transmettre des valeurs littérales, et non pour changer de manière dynamique les clauses SELECT ou FROM.

Enfin, il est également possible de lier des tables à du SQL personnalisé.



Cela vous permet d'écrire un code SQL personnalisé plus spécifique qui fait référence à un sous-ensemble du schéma global. Les jointures vers les autres tables peuvent ainsi potentiellement être éliminées, comme des jointures de table normales, ce qui améliore l'efficacité des requêtes.

Les alternatives au SQL personnalisé

Au lieu d'utiliser le SQL personnalisé directement dans Tableau, il est parfois préférable de déplacer la requête dans la source de données sous-jacente. Bien souvent, cela permet d'améliorer les performances, car la source de données peut décomposer efficacement la requête, ou bien une requête complexe n'a besoin d'être exécutée qu'une seule fois. Cela constitue également une bonne pratique en matière de gestion dans la mesure où la même définition peut être utilisée par plusieurs classeurs et sources de données.

Il existe plusieurs méthodes.

Les vues

Presque tous les systèmes de gestion de bases de données prennent en charge le concept de vues, à savoir une table virtuelle représentant le résultat d'une requête de base de données. Pour certains systèmes, le fait d'extraire une requête d'une instruction SQL personnalisée et de l'instancier dans

la base de données sous forme de vue améliore nettement les performances. En effet, l'outil d'optimisation de la requête est capable de générer un meilleur plan d'exécution que lorsque la requête est incluse à l'intérieur d'une déclaration SELECT englobante. Pour en savoir plus, reportez-vous à la discussion suivante sur les forums de la communauté :

https://community.tableau.com/thread/208019

Le fait de définir la logique d'une requête personnalisée en utilisant une vue plutôt qu'un classeur Tableau permet également de la réutiliser dans plusieurs classeurs et sources de données. Par ailleurs, de nombreux systèmes de gestion de bases de données prennent en charge le concept de vues matérialisées ou d'instantanés. Les résultats d'une requête de vue sont alors préalablement calculés et mis en cache, ce qui accélère le temps de réponse au moment de la requête. Ces éléments sont similaires aux tables récapitulatives que vous voyez ci-dessous, mais sont gérés automatiquement par le système de bases de données.

Les procédures stockées

Les procédures stockées sont similaires aux vues, mais peuvent contenir une logique bien plus complexe et potentiellement effectuer plusieurs étapes de la préparation de données. Elles peuvent également comporter des paramètres, ce qui leur permet de renvoyer un ensemble de données ciblé en fonction des choix de l'utilisateur.

Les procédures stockées sont prises en charge dans Sybase ASE, SQL Server et Teradata. Pour éviter d'exécuter la procédure stockée plusieurs fois pour une seule visualisation, Tableau l'exécute et stocke l'ensemble de résultats dans une table temporaire, dans la base de données. Les requêtes de la visualisation sont ensuite exécutées sur cette table temporaire. L'exécution de la procédure stockée et le remplissage de la table temporaire ont lieu à l'ouverture initiale du classeur, et chaque fois que les paramètres des processus stockés sont modifiés. Cela peut prendre du temps, et les interactions peuvent être lentes.

Si vous extrayez les résultats d'une procédure stockée et dotée de paramètres sous forme d'extrait de données, cet extrait sera éliminé et renouvelé chaque fois que vous modifierez les valeurs des paramètres.

Pour en savoir plus sur l'utilisation des procédures stockées, reportez-vous à l'aide en ligne de Tableau :

http://onlinehelp.tableau.com/current/pro/desktop/fr-fr/help.htm#connect_basic_stored_procedures.html

Les tables récapitulatives

Si vous disposez d'un très grand ensemble de données détaillé, que vous synthétisez généralement pour vos requêtes (vous stockez par exemple des transactions individuelles mais utilisez un récapitulatif des données par jour, région, client, produit, etc.), vous pouvez envisager de créer une table récapitulative et de l'utiliser avec Tableau pour accélérer les requêtes.

Remarque : vous pouvez utiliser des extraits de données Tableau pour obtenir un résultat similaire en créant un extrait de données agrégé. Reportez-vous à la section sur les extraits pour en savoir plus.

La déclaration SQL initiale

Au lieu du SQL personnalisé, vous pouvez également utiliser la déclaration SQL personnalisée dans un bloc SQL initial si la source de données le permet. Vous pouvez ainsi créer une table temporaire, qui sera ensuite la table sélectionnée dans votre requête. Étant donné que le bloc SQL initial est exécuté une fois seulement à l'ouverture du classeur, et non à chaque modification de la visualisation

pour le SQL personnalisé, cela peut notablement améliorer les performances dans certains cas.

Notez que dans Tableau Server, l'administrateur peut appliquer une restriction à la déclaration SQL initiale pour empêcher son exécution. Vérifiez que vous pouvez l'utiliser dans votre environnement si vous prévoyez de publier un classeur pour le mettre à la disposition d'autres utilisateurs.

Pour en savoir plus sur le bloc SQL initial, reportez-vous à la documentation en ligne :

http://onlinehelp.tableau.com/current/pro/desktop/fr-fr/help.htm#connect_basic_initialsql.html

Les données sont-elles en cause ?

La capacité de Tableau à se connecter à des données stockées sur de nombreuses plates-formes différentes constitue l'une des fonctionnalités les plus puissantes. D'une manière générale, ces plates-formes se classent parmi les catégories suivantes :

- Sources de données basées sur des fichiers : comme Excel ou format CSV
- Sources de données de bases de données relationnelles : comme Oracle, Teradata et SQL Server, ainsi que les dispositifs analytiques spécialisés comme HP Vertica, IBM Netezza, etc.
- Sources de données OLAP : comme Microsoft Analysis Services ou Oracle Essbase
- Sources de données Big Data : comme Hadoop
- Sources de données dans le cloud : comme Salesforce, Google, etc.

Chaque type de source de données possède ses avantages et ses inconvénients, et est traité différemment.

Notez que Tableau Desktop est compatible à la fois avec Windows et Mac OS X, et que l'ensemble des sources de données prises en charge sous Windows n'est pas le même que sous Mac. Tableau s'efforce de réduire ces différences entre les plates-formes, mais actuellement certaines sources de données ne sont prises en charge que sur l'une d'elles.

Conseils d'ordre général

Utilisez des pilotes natifs

Tableau 10 prend en charge la connexion native à plus de 40 sources de données différentes, pour lesquelles Tableau a mis en œuvre des techniques, capacités et optimisations spécifiques. Les activités d'ingénierie et de test pour ces connexions permettent d'optimiser leur robustesse.

Tableau prend également en charge les pilotes ODBC généraux pour accéder aux sources de données autrement qu'avec la liste de connecteurs natifs. Cette norme étant publique, de nombreux prestataires proposent des pilotes ODBC pour leurs bases de données. Tableau utilise ces pilotes pour se connecter aux données. La manière d'interpréter ou de mettre en œuvre les fonctionnalités de la norme ODBC peut varier d'un prestataire à l'autre. Dans certains cas, Tableau vous conseillera ou vous demandera de créer un extrait de données pour continuer d'utiliser un pilote particulier. Il existe également certains pilotes ODBC et certaines bases de données auxquels Tableau n'est pas en mesure de se connecter.

S'il existe un pilote natif pour la source de données que vous exploitez, utilisez-le au lieu de recourir à une connexion ODBC, car il offrira généralement de meilleures performances. Notez également que les connexions ODBC sont uniquement disponibles sous Windows.

Restez aussi proche que possible des données

Comme nous l'avons vu plus haut, si une source de données exécute des requêtes lentement, le logiciel Tableau sera lent également. Dans la mesure du possible, vous pouvez tester les performances brutes de la source de données en installant Tableau Desktop sur la même machine que celle-ci, puis en exécutant quelques requêtes. Cela permettra d'éliminer des facteurs comme la bande passante ou la latence, et vous aidera à mieux comprendre les performances brutes de la requête dans la source de données. Par ailleurs, le fait d'utiliser *localhost* pour le nom de la source de données au lieu de son nom DNS permet de déterminer si des facteurs comme une résolution de noms ou des serveurs proxy lents contribuent aux faibles performances.

Les sources de données

Fichiers

Cette catégorie inclut tous les formats de données basés sur des fichiers, notamment les fichiers CSV, les feuilles de calcul Excel et les fichiers Microsoft Access, mais également les fichiers des plates-formes d'analyse statistique SPSS, SAS et R. Les utilisateurs métier utilisent souvent des données stockées dans ces formats, parce que c'est un moyen classique d'extraire les données de leur environnement contrôlé, en exécutant un rapport, ou une requête pour obtenir un extrait.

En général, il est recommandé d'importer des sources de données basées sur des fichiers dans le moteur de données rapide de Tableau. Cela permet d'accélérer les requêtes et de réduire la taille du fichier stockant les valeurs de données. Toutefois, s'il s'agit d'un petit fichier ou si vous souhaitez que les changements des données soient reflétés, vous pouvez opter pour une connexion en direct.

Extraits temporaires

Lorsque vous vous connectez à des fichiers Excel/texte récents ou à des fichiers statistiques, Tableau crée, de manière invisible, un fichier d'extrait dans le cadre du processus de connexion. Grâce à ces extraits temporaires, l'utilisation des données est bien plus rapide que si vous deviez exécuter une requête directement sur le fichier.

La première fois que vous utilisez un fichier volumineux, vous pouvez constater que le volet d'aperçu des données met plusieurs secondes à s'afficher. C'est parce que Tableau est en train d'extraire les données du fichier et de les écrire dans un fichier d'extrait temporaire. Par défaut, ces fichiers sont créés dans le dossier `C:\Users\<username>\AppData\Local\Tableau\Caching\TemporaryExtracts`, avec un nom produit par l'algorithme de hachage en fonction du chemin d'accès et de la date de dernière modification du fichier de données. Tableau conserve dans ce dossier les extraits temporaires des cinq derniers fichiers de source de données utilisés, et supprime le plus ancien lorsqu'il en crée un nouveau. Si vous réutilisez par la suite un fichier pour lequel il existe un extrait temporaire, Tableau ouvre simplement le fichier de cet extrait, et l'aperçu des données s'affiche presque instantanément.

Bien que les extraits temporaires contiennent des données sous-jacentes et d'autres informations comme un extrait Tableau standard, ils sont enregistrés dans un format différent, avec l'extension .ttde. Autrement dit, ils ne peuvent pas être utilisés de la même manière que les extraits Tableau.

Anciens connecteurs pour les fichiers Excel et texte

Dans les versions précédentes de Tableau, les connexions aux fichiers Excel et texte utilisaient le pilote du moteur de données Microsoft JET. Depuis la version 8.2, Tableau utilise par défaut un pilote natif qui offre de meilleures performances et peut traiter des fichiers plus volumineux et plus complexes. Néanmoins, dans certains cas, vous pouvez préférer utiliser les anciens pilotes, par exemple pour travailler avec le SQL personnalisé. Sachez qu'il est possible de revenir à l'ancien pilote JET. Notez que la lecture des fichiers MS Access nécessite toujours le pilote JET.

Pour connaître en détail les différences entre les deux pilotes, reportez-vous à la page suivante :

> http://onlinehelp.tableau.com/current/pro/desktop/fr-fr/help.htm#upgrading_connection.html

Notez que les pilotes JET ne sont pas disponibles pour Mac OS. Tableau Desktop pour Mac ne permet donc pas de lire des fichiers MS Access et ne propose pas l'ancien connecteur pour les fichiers Excel et texte.

Les bases de données relationnelles

Les sources de données relationnelles constituent le type de source le plus courant pour les utilisateurs Tableau, et le logiciel fournit des pilotes natifs pour de nombreuses plates-formes. Ces données, personnelles ou professionnelles, peuvent être basées sur des lignes ou des colonnes et être accessibles via des pilotes natifs ou la norme ODBC générique. Techniquement, cette catégorie inclut également les sources de données MapReduce, car elles sont accessibles par l'intermédiaire de couches d'accès SQL comme Hive ou Impala. Nous allons toutefois les présenter plus loin, dans la section sur le Big Data.

De nombreux facteurs internes ont un impact sur la vitesse des requêtes dans un système de gestion de bases de données relationnelles. Il faut généralement l'intervention de l'administrateur de la base de données pour modifier ou régler ces paramètres, mais cela permet d'améliorer significativement les performances.

Disposition en lignes ou en colonnes

Il existe deux types de systèmes de gestion de bases de données : ceux qui reposent sur les lignes et ceux qui reposent sur les colonnes. La disposition en lignes est bien adaptée aux charges de travail de type OLTP (traitement transactionnel en ligne), qui sont plus importantes en raison des transactions interactives. La disposition en colonnes est bien adaptée aux charges de travail analytiques (citons par exemple les entrepôts de données), qui impliquent généralement l'exécution de requêtes très complexes sur de grands ensembles de données.

Aujourd'hui, de nombreuses solutions analytiques très performantes s'appuient sur des systèmes basés sur les colonnes et permettent une exécution plus rapide des requêtes. Tableau prend en charge notamment les bases de données en colonnes Actian Vector, Amazon Redshift, HP Vertica, IBM Netezza, MonetDB, Pivotal Greenplum, SAP HANA et SAP Sybase IQ.

SQL en tant qu'interface

De nombreux systèmes ne reposent pas sur la technologie classique d'un système de gestion de bases de données relationnelles, mais sont tout de même présentés comme des sources relationnelles car ils proposent une interface basée sur le SQL. Dans le cas de Tableau, cela inclut plusieurs plates-formes NoSQL, comme MarkLogic, DataStax ou MongoDB, ainsi que des plates-formes d'accélération des requêtes comme Kognitio, AtScale et JethroData.

Indexation

Une bonne indexation de la base de données est essentielle pour optimiser les performances des requêtes.

- Assurez-vous d'avoir des index pour les colonnes utilisées dans les jointures de tables.
- Assurez-vous d'avoir des index pour les colonnes utilisées dans les filtres.
- Sachez que dans certaines bases de données, il se peut que les requêtes ignorent les index des colonnes de date et de date/heure si vous utilisez des filtres Date discrète. Nous y reviendrons dans la section sur les filtres, mais avec un filtre Plage de dates, vous avez la garantie que l'index des dates est utilisé. Par exemple, au lieu d'utiliser `YEAR([DateDim])=2010`, exprimez le filtre avec la syntaxe `[DateDim] >= #2010-01-01# and [DateDim] <= #2010-12-31#)`.
- Assurez-vous que les statistiques sont activées sur les données pour que l'outil d'optimisation des requêtes puisse créer des plans de grande qualité.

De nombreux environnements de gestion de bases de données disposent d'outils de gestion qui examinent les requêtes et recommandent des index utiles.

Valeurs NULL

La présence de valeurs NULL dans les colonnes des dimensions peut réduire l'efficacité des requêtes. Supposons que nous voulions présenter le total des ventes par pays pour les 10 principaux produits dans notre visualisation. Tableau génère tout d'abord une sous-requête pour déterminer les 10 principaux produits, puis il relie ces données à la requête « par pays » pour produire le résultat final.

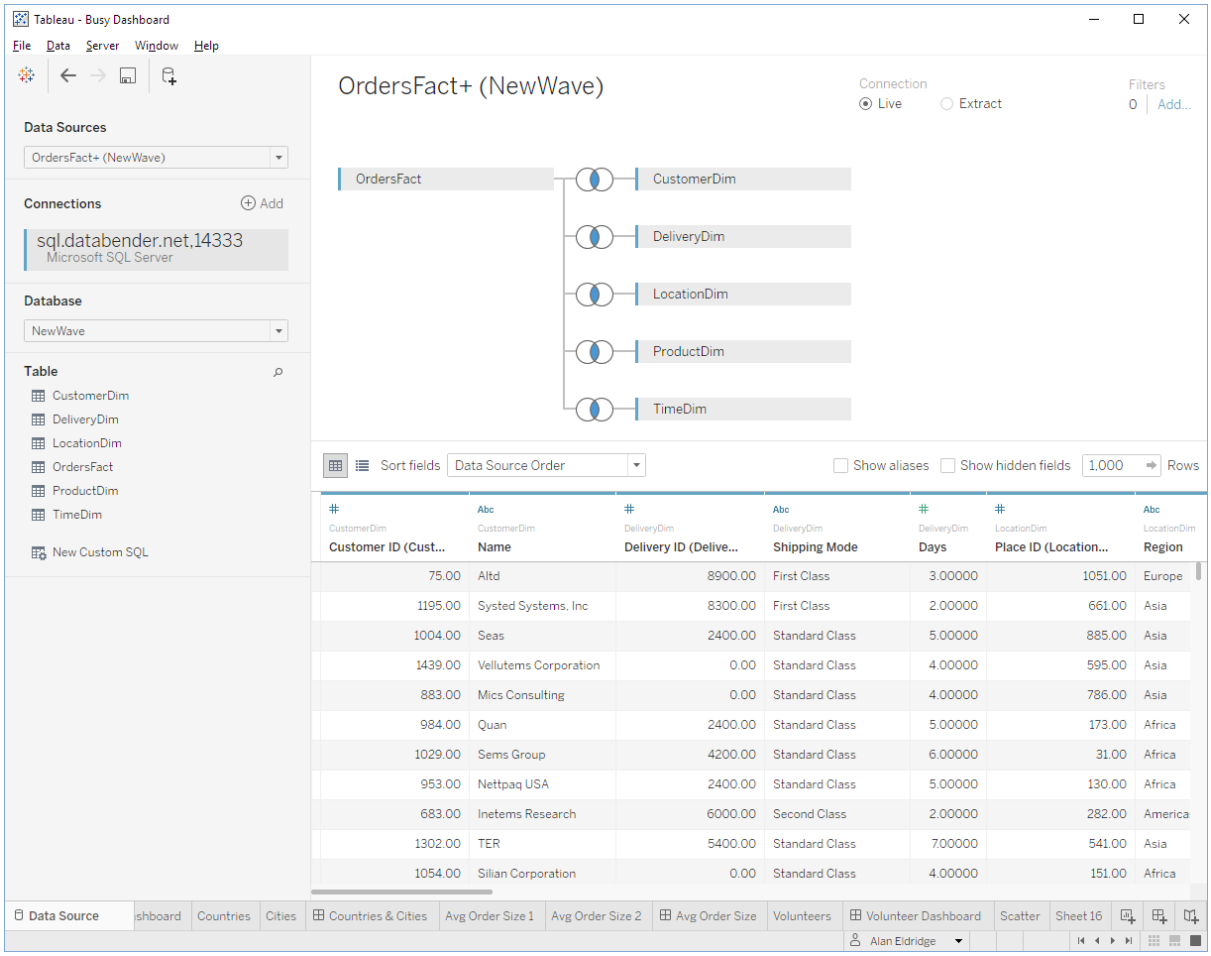
Si la colonne des produits accepte les valeurs NULL, vous devez exécuter une requête pour vérifier si des valeurs NULL figurent parmi les 10 produits renvoyés par la sous-requête. Si tel est le cas, vous devez effectuer une jointure préservant les valeurs NULL, ce qui requiert plus de capacités de calcul qu'une jointure classique. Si la colonne des produits est configurée pour ne pas accepter les valeurs NULL, vous savez qu'il n'y en aura pas dans le résultat de la sous-requête, et vous pouvez effectuer une jointure normale, ce qui vous dispense de la requête vérifiant la présence de valeurs NULL.

C'est pourquoi il est recommandé, dans la mesure du possible, de définir les colonnes des dimensions pour qu'elles n'acceptent pas les valeurs NULL.

Intégrité référentielle

Lorsque vous reliez plusieurs tables dans une source de données, Tableau propose une fonctionnalité utile (et généralement invisible), à savoir l'élimination des jointures. Étant donné que le traitement des jointures nécessite du temps et utilise des ressources sur le serveur de bases de données, nous n'allons pas énumérer à chaque fois toutes les jointures déclarées dans la source de données. L'élimination des jointures permet d'envoyer des requêtes uniquement sur les tables pertinentes, plutôt que sur toutes les tables définies dans la jointure.

Imaginez le scénario suivant, dans lequel nous avons joint plusieurs tables dans un petit schéma en étoile.



Avec l'élimination des jointures, un double-clic sur la mesure Sales génère la requête suivante :

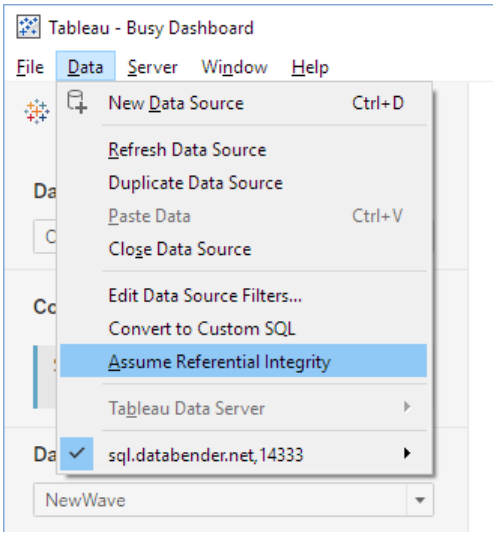
```
SELECT SUM([OrdersFact].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[OrdersFact] [OrdersFact]
GROUP BY ()
```

Sans cette fonction, une requête bien moins efficace est générée :

```
SELECT SUM([OrdersFact].[Sales]) AS [sum:Sales:ok]
FROM [dbo].[OrdersFact] [OrdersFact]
   INNER JOIN [dbo].[CustomerDim] [CustomerDim]
    ON ([OrdersFact].[Customer ID] = [CustomerDim].[Customer ID])
   INNER JOIN [dbo].[DeliveryDim] [DeliveryDim]
    ON ([OrdersFact].[Delivery ID] = [DeliveryDim].[Delivery ID])
   INNER JOIN [dbo].[LocationDim] [LocationDim]
    ON ([OrdersFact].[Place ID] = [LocationDim].[Place ID])
   INNER JOIN [dbo].[ProductDim] [ProductDim]
    ON ([OrdersFact].[Product ID] = [ProductDim].[Product ID])
   INNER JOIN [dbo].[TimeDim] [TimeDim]
    ON ([OrdersFact].[Date ID] = [TimeDim].[Date ID])
GROUP BY ()
```

Toutes les tables doivent être jointes pour garantir que les sommes adéquates seront calculées dès le début pour les mesures. Par exemple, si la table des faits contenait des données pour 2008-2012, mais que la table de la dimension temporelle contenait des valeurs pour 2010-2012 uniquement, le calcul SUM([Sales]) résultant pourrait changer selon que la table temporelle est incluse ou non.

Par le passé, une intégrité référentielle stricte était requise lorsque la règle était appliquée par le système de gestion de bases de données. Néanmoins, de nombreux clients disposent de sources de données dans lesquelles l'intégrité référentielle est appliquée soit sur la couche applicative soit par un processus ETL. C'est ce que l'on appelle l'intégrité référentielle « douce ». En utilisant l'option Estimer l'intégrité référentielle, les utilisateurs peuvent indiquer à Tableau qu'une telle intégrité référentielle est en place et que l'élimination des jointures peut être utilisée sans risques.



Bien que Tableau puisse utiliser les deux types d'intégrité référentielle, il est souvent préférable d'utiliser la version stricte, car la base de données peut également effectuer l'élimination de jointures. Pour en savoir plus, lisez les articles de Russel Christopher sur son blog Tableau Love :

http://bit.ly/1HACmPV
http://bit.ly/1HACqPn

Partitionnement

Le partitionnement d'une table améliore les performances en divisant une table volumineuse en plusieurs tables plus petites, appelées partitions. Les requêtes sont alors exécutées plus rapidement, car il y a moins de données à analyser, ou que davantage de lecteurs peuvent gérer les entrées/sorties. Le partitionnement est recommandé pour les grandes quantités de données, et cela ne fait aucune différence pour Tableau.

Le partitionnement est efficace dans Tableau s'il est effectué dans une dimension, comme l'heure, la région ou la catégorie, qui est généralement filtrée. Ainsi, chaque requête n'a à lire que les enregistrements d'une seule partition.

Notez que pour certaines bases de données, les filtres Plage de dates (non discrets) sont nécessaires pour garantir une utilisation correcte des index de partition. Dans le cas contraire, une analyse complète de la table se traduira par un très net ralentissement des performances.

Tables temporaires

Dans Tableau, de nombreuses opérations peuvent impliquer l'utilisation de tables temporaires, comme la création de groupes et d'ensembles ad hoc ou la fusion de données. Il est recommandé d'autoriser les utilisateurs à créer et supprimer les tables temporaires et de vérifier que l'espace est suffisant dans votre environnement pour le traitement des requêtes exécutées.

OLAP

Tableau prend en charge plusieurs sources de données OLAP :

- Microsoft Analysis Services
- Microsoft PowerPivot (PowerPivot pour Excel et PowerPivot pour SharePoint)
- Oracle Essbase
- SAP Business Warehouse
- Teradata OLAP

Il existe des différences fonctionnelles entre les connexions à des bases OLAP et les connexions à des bases relationnelles, du fait que les langages sous-jacents ne sont pas les mêmes : MDX/DAX dans un cas, et SQL dans l'autre. Notez bien que les deux proposent la même interface dans Tableau, les mêmes visualisations et le même langage d'expression pour les mesures calculées. Les différences reposent principalement sur les métadonnées (comment elles sont définies et où), le filtrage, le fonctionnement des totaux et des agrégations, et la façon dont la source de données est utilisée dans la fusion de données.

Pour en savoir plus sur les différences entre les sources de données relationnelles et les sources de données OLAP dans Tableau, reportez-vous à l'article suivant de la base de connaissances :

> http://kb.tableau.com/articles/knowledgebase/functional-differences-olap-relational?lang=fr-fr

Extraits de données SAP BW

SAP BW est un cas particulier parmi nos sources de données OLAP, car vous pouvez extraire les données depuis des cubes SAP BW dans le moteur de données Tableau. Notez que cela nécessite un code spécial que vous pouvez demander à Tableau. Tableau récupère les nœuds du niveau feuille, et non pas les données du niveau d'exploration, et les transforme en source de données relationnelle. Étant donné que la transformation d'une source multidimensionnelle en source relationnelle ne préserve pas toutes les structures du cube, le fait d'alterner entre les extraits et les connexions en direct librement sans changer l'état de la visualisation n'est pas pris en charge pour les extraits

de cube. Vous devrez faire votre choix avant de commencer à créer votre visualisation, même si vous n'avez pas besoin de prendre toutes les décisions au préalable. Vous pouvez changer les options d'alias (clé, nom long, etc.) après l'extraction.

Pour en savoir plus sur la création des extraits SAP BW, reportez-vous à la page suivante :

https://community.tableau.com/docs/DOC-9914

Le Big Data

Le terme « Big Data » peut prendre de nombreuses significations dans le domaine de l'analyse de données, mais dans ce document il fait plus particulièrement référence aux plates-formes basées sur Hadoop. Tableau 10 prend en charge quatre distributions Hadoop qui utilisent des connexions Hive ou Impala :

- Amazon EMR
  - HiveServer
  - HiveServer2
  - Impala
- Cloudera Hadoop
  - HiveServer
  - HiveServer2
  - Impala
- HortonWorks Hadoop Hive
  - HiveServer
  - HiveServer2
  - HortonWorks Hadoop Hive
- MapR Hadoop Hive
  - HiveServer
  - HiveServer2
- Spark SQL
  - SharkServer *
  - SharkServer2 *
  - SparkThriftServer

** Notez que les connexions SharkServer et SharkServer2 sont proposées à l'utilisation, mais ne sont pas prises en charge par Tableau.*

Hive joue le rôle de couche de traduction SQL-Hadoop et traduit la requête en MapReduce. Celle-ci est ensuite exécutée sur des données HDFS. Impala exécute la déclaration SQL directement sur les données HDFS (sans nécessiter MapReduce). Tableau prend également en charge Spark SQL, un moteur de traitement open source pour le Big Data qui peut être 100 fois plus rapide que MapReduce en travaillant dans la mémoire plutôt que sur le disque.

Impala est généralement bien plus rapide que Hive, et Spark est encore plus rapide.

Même avec ces composants additionnels, Hadoop n'est parfois pas suffisamment réactif pour les requêtes analytiques comme celles que crée Tableau. Les extraits de données Tableau sont généralement utilisés pour améliorer les délais de réponse. Nous aborderons les extraits et la manière de les utiliser avec le Big Data de manière plus détaillée plus loin dans ce document.

Pour en savoir plus sur l'amélioration des performances avec les sources de données Hadoop, reportez-vous à l'article suivant :

http://kb.tableau.com/articles/knowledgebase/hadoop-hive-performance?lang=fr-fr

Le cloud

Tableau prend actuellement en charge les sources de données cloud suivantes :

- Salesforce.com
- Google Analytics
- oData (y compris Windows Azure Marketplace DataMarket)

Les sources de ce premier groupe lisent un ensemble d'enregistrements depuis un service Web et les chargent dans un fichier d'extrait de données Tableau. La connexion en direct n'est pas disponible pour ces sources, mais le fichier de l'extrait peut être actualisé pour mettre à jour les données qu'il contient. Avec Tableau Server, cette mise à jour peut être automatisée et planifiée.

Tableau prend également en charge les plates-formes de données cloud suivantes :

- Amazon Aurora
- Amazon Redshift
- Amazon RDS
- Google BigQuery
- Google Cloud SQL
- Google Sheets
- Microsoft Azure SQL Data Warehouse
- Snowflake

À la différence du groupe de sources de données précédent, celles-ci fonctionnent comme des sources de données relationnelles et acceptent à la fois les connexions en direct et les extraits. Nous n'allons pas entrer dans les détails (vous pouvez vous reporter à la section ci-dessus sur les sources de données relationnelles), mais vous souhaiterez généralement les conserver en tant que connexions en direct pour éviter de transférer d'importants volumes de données depuis le cloud.

Enfin, Tableau propose également un connecteur de données générique, le connecteur de données Web, pour les importations depuis les services Web publiant des données au format JSON, XML ou HTML.

Salesforce

Lorsque vous vous connectez à Salesforce, ne perdez pas de vue les limitations suivantes s'appliquant au connecteur.

Pas de possibilité de connexion en direct

Plusieurs raisons expliquent notre choix de proposer uniquement des extraits au lieu d'une connexion en direct :

- Performances : les requêtes analytiques en direct sur Salesforce sont (généralement) lentes.
- Quotas d'API : une fréquence trop élevée de requêtes en direct sur Salesforce peut entraîner une suspension de compte si le quota quotidien est atteint.En créant un extrait, nous utilisons les API de manière efficace afin de réduire le nombre d'appels, et d'éviter ainsi de dépasser la limite.Pour garantir des performances optimales et faire en sorte que l'API Force.com soit disponible pour tous ses clients, Salesforce.com équilibre les charges des transactions en imposant deux types de limitation,d'une part pour le nombre de requêtes d'API simultanées (http://sforce.co/1f19cQa) et d'autre part pour le nombre total de requêtes d'API (http://sforce.co/1f19kiH).

L'extraction initiale peut être très lente

La première extraction de données depuis Salesforce peut prendre du temps, notamment en fonction de la taille de la table ou de la charge de Force.com, parce que les objets sont téléchargés dans leur intégralité.

Les options de jointure sont limitées

Si vous décidez d'utiliser plusieurs tables, souvenez-vous que vous pouvez uniquement joindre des objets en fonction de leur clé primaire et de leur clé étrangère (jointures gauche et interne seulement).

Vous ne pouvez pas préalablement filtrer les données

Le connecteur Salesforce ne permet pas de filtrer préalablement les données. Si vous avez absolument besoin de ce préfiltrage, vous pouvez utiliser un pilote ODBC Salesforce tiers, par exemple de Simba ou DataDirect, qui prend en charge les connexions en direct. Vous pouvez ensuite créer un extrait avec cette connexion.

DBAmp propose également une solution permettant d'envoyer des données Salesforce dans une base de données SQL Server. Vous pouvez ensuite la connecter à Tableau grâce au connecteur SQL Server.

Les colonnes avec des formules ne peuvent pas être transférées

Si vous disposez de champs calculés, vous devrez les recréer dans Tableau une fois les données extraites.

Les requêtes sont soumises à une limite de 10 000 caractères

L'API Force.com limite les requêtes à un total de 10 000 caractères. Si vous vous connectez à une ou plusieurs tables très volumineuses, contenant de nombreuses colonnes dont les noms sont potentiellement longs, vous risquez d'atteindre cette limite en créant un extrait. Dans une telle situation, vous devez sélectionner moins de colonnes pour réduire la taille de la requête. Dans certains cas, Salesforce.com peut augmenter cette limite pour votre entreprise. Prenez contact avec votre conseiller Salesforce pour en savoir plus.

Google Analytics

Google Analytics (GA) échantillonne vos données lorsqu'un rapport inclut de nombreuses dimensions ou une grande quantité de données. Si vos données pour une propriété Web particulière dans une plage de dates donnée dépassent (pour un compte GA normal) 50 000 visites, GA agrège les résultats et renvoie un ensemble échantillonné de ces données. Lorsque GA renvoie un échantillon de ces données vers Tableau, Tableau affiche le message suivant dans le coin inférieur droit d'une vue :

« Google Analytics a renvoyé des données échantillonnées. L'échantillonnage a lieu lorsque la connexion inclut un grand nombre de dimensions ou de données. Reportez-vous à la documentation Google Analytics pour savoir en quoi l'échantillonnage peut affecter les résultats de votre rapport. »

Il est important de savoir quand vos données sont échantillonnées, car l'agrégation de certains échantillons d'ensembles de données peut entraîner des inférences biaisées ou imprécises. Supposons par exemple que vous agrégiez un ensemble échantillonné décrivant une catégorie inhabituelle de vos données. Les inférences sur l'ensemble agrégé risquent d'être biaisées, car il n'y a pas assez d'échantillons dans cette catégorie. Pour créer des vues GA vous permettant de créer des inférences précises sur vos données, vérifiez que vous disposez d'un échantillon suffisamment important pour la catégorie concernée. La taille d'échantillon recommandée est 30.

Pour en savoir plus sur le réglage de la taille des échantillons GA et sur leur création, reportez-vous à la documentation relative à ce service :

https://support.google.com/analytics/answer/1042498?hl=fr

Pour éviter l'échantillonnage, vous avez le choix entre deux approches :

- Exécutez plusieurs rapports GA au niveau de la session ou de l'appel pour décomposer les données en morceaux non échantillonnés. Vous pouvez ensuite télécharger les données dans un fichier Excel et utiliser le moteur d'extrait Tableau pour ajouter des données depuis une source de données, et les réassembler dans un ensemble de données unique.
- Optez pour un compte GA Premium pour augmenter le nombre d'enregistrements que vous pouvez inclure dans un rapport. Cela permet de segmenter les données plus facilement afin de les analyser. Google a annoncé que les titulaires d'un compte Premium pourront exporter leurs données de session ou d'appel vers Google BigQuery pour les analyser. Cette approche sera simplifiée, car Tableau permet de se connecter directement à BigQuery.

Enfin, notez que l'API que Tableau utilise pour effectuer des requêtes GA limite les requêtes à un maximum de 7 dimensions et 10 mesures.

Connecteur de données Web

Un connecteur de données Web Tableau vous permet d'accéder aux données qui n'ont pas encore de connecteur. Utilisez-le pour créer des connexions avec toutes les données accessibles via HTTP (ou presque), y compris les services Web, les données JSON et XML, les API REST et bien d'autres sources. Comme vous contrôlez la manière dont les données sont collectées, vous pouvez même combiner des données provenant de diverses sources.

Vous pouvez créer un connecteur de données Web en composant une page Web qui contient du code JavaScript et HTML. Une fois le connecteur créé, vous pouvez le mettre à la disposition des autres utilisateurs Tableau en le publiant sur Tableau Server.

Pour vous aider à créer des connecteurs de données Web, nous proposons un kit de développement (SDK) qui inclut des modèles, des exemples de code et un simulateur pour tester vos connecteurs Web. Cette documentation inclut également un didacticiel qui vous explique comment créer un connecteur de données Web de toutes pièces. Ce SDK pour connecteur de données Web est open source. Pour en savoir plus, reportez-vous à la page webdataconnector Tableau sur GitHub.

Reportez-vous à la page suivante pour en savoir plus sur les connecteurs Web :

http://tableau.github.io/webdataconnector/

La préparation des données

Les données dont vous avez besoin dans vos classeurs ne sont pas toujours optimisées. Elles ne sont pas forcément bien « propres » ou structurées, et elles peuvent être réparties entre plusieurs fichiers ou bases de données. Par le passé, il était nécessaire de fournir à Tableau des données propres et normalisées pour un fonctionnement optimal, ce qui impliquait parfois de les préparer d'abord à l'aide d'autres outils.

Ce n'est plus le cas. Tableau propose désormais de nombreuses fonctionnalités qui permettent de charger des données mal organisées, notamment :

- La permutation
- L'union
- L'interpréteur de données
- La fusion de colonnes

La préparation des données à la volée vous aide à rester concentré sur votre analyse et vous permet de continuer à analyser vos données dans Tableau si vous ne disposez pas d'outils de préparation externes. Néanmoins, ces opérations nécessitent généralement une certaine puissance de calcul, en particulier pour les grands volumes de données, et peuvent avoir des répercussions sur les performances pour les personnes qui utilisent des rapports. Dans la mesure du possible, il est recommandé d'utiliser un extrait de données pour matérialiser les données à la suite de ces opérations.

Peut-être devriez-vous également utiliser ces fonctionnalités en complément de processus ETL et déterminer s'il est préférable et plus efficace de recourir au traitement préalable des données en amont si vous comptez les utiliser dans plusieurs rapports ou les partager avec d'autres outils d'aide à la décision.

Les extraits de données

Nous avons jusqu'ici abordé les techniques permettant d'améliorer les performances des connexions de données, lorsque ces données conservent leur format d'origine. Il s'agit de connexions en direct, et les performances et le bon fonctionnement dépendent alors de la plate-forme de la source de données. Pour améliorer les performances des connexions en direct, il est souvent nécessaire d'apporter des modifications dans la source de données, ce qui n'est souvent pas possible pour de nombreux clients.

Il existe une alternative accessible à tous : s'appuyer sur le moteur de données rapide de Tableau et extraire les données du système source dans un extrait de données Tableau. Pour la plupart des utilisateurs, il s'agit là de la solution la plus rapide et la plus simple pour améliorer de manière significative les performances d'un classeur avec tous les types de source de données.

Qu'est-ce qu'un extrait de données ?

- Un extrait est un cache persistant de données, écrit sur un disque et reproductible.
- Un extrait est un magasin de données en colonnes, dont les données ont été optimisées pour les requêtes analytiques.
- Un extrait est totalement coupé de la base de données pendant les requêtes. Un extrait se substitue à la connexion de données en direct.
- Un extrait peut être actualisé, que ce soit par une régénération complète ou de manière incrémentielle en y ajoutant des lignes de données.
- Un extrait prend en compte l'architecture, à l'inverse de la plupart des technologies en mémoire, et n'est pas limité par la quantité de RAM disponible.
- Un extrait peut être déplacé, car il est stocké sous forme de fichier pouvant être copié sur un disque dur local et utilisé sans que l'utilisateur soit connecté au réseau de l'entreprise. Il peut également être utilisé pour incorporer des données sous la forme de classeurs complets, à utiliser avec Tableau Reader.
- Un extrait est souvent bien plus rapide qu'une connexion en direct aux données sous-jacentes.

Tom Brown, de The Information Lab, a écrit un excellent article décrivant plusieurs cas d'utilisation avantageuse des extraits. Lisez les commentaires pour découvrir des exemples soumis par d'autres utilisateurs :

http://bit.ly/1F2iDnT

Notez bien que les extraits de données sont à utiliser en complément d'un entrepôt de données, et ne les remplacent en aucun cas. Bien qu'ils puissent être utilisés pour collecter et agréger des données dans le temps, en ajoutant des données de manière incrémentielle en fonction d'un cycle

périodique, vous devez considérer cela comme une solution tactique, et non une solution à long terme. Les mises à jour incrémentielles ne prennent pas en charge les actions de mise à jour ou de suppression pour les enregistrements déjà traités. Vous devez régénérer intégralement l'extrait pour modifier ces enregistrements.

Enfin, il n'est pas possible de créer des extraits pour les sources de données OLAP, comme SQL Server Analysis Services ou Oracle Essbase. Vous pouvez toutefois créer des extraits depuis SAP BW (reportez-vous à la section correspondante ci-dessus).

Quand utiliser des extraits ? Quand utiliser des connexions en direct ?

De même qu'il y a un temps pour tout, il y a un moment opportun pour utiliser les extraits de données. Voici quelques situations dans lesquelles utiliser des extraits peut être utile.

- Exécution lente des requêtes : si votre système de source de données met du temps à traiter les requêtes générées par Tableau Desktop, vous pouvez créer un extrait pour facilement améliorer les performances. Le format de l'extrait de données est conçu à la base pour répondre rapidement aux requêtes analytiques. Dans ce cas, l'extrait peut servir de cache d'accélération des requêtes. C'est recommandé pour certains types de connexion de données, comme les gros fichiers texte ou les connexions SQL personnalisées, et certaines sources fonctionnent uniquement dans ce modèle. Reportez-vous à la section sur les sources de données dans le cloud.
- Analyse hors connexion : vous avez besoin de travailler avec des données lorsque la source d'origine n'est pas disponible, par exemple lorsque vous êtes en déplacement ou travaillez de chez vous, et n'êtes donc pas connecté au réseau. Les extraits de données sont des fichiers persistants que vous pouvez facilement copier sur un appareil portable. Il s'agit simplement d'alterner entre extraits et connexions en direct si vous n'êtes pas connecté en permanence au réseau.
- Classeurs complets pour Tableau Reader/Online/Public : si vous prévoyez de partager vos classeurs avec d'autres utilisateurs qui les ouvriront dans Tableau Online, ou si vous souhaitez les publier sur Tableau Online ou Tableau Public, vous devez intégrer les données dans un classeur complet. Même si le classeur utilise des sources de données pouvant également être intégrées, par exemple les sources sous forme de fichiers, les extraits de données offrent un niveau de compression supérieur, et le classeur complet résultant sera moins volumineux.
- Fonctionnalité supplémentaire : pour certaines sources de données, par exemple les sources sous forme de fichiers via l'ancien pilote MS JET, certaines fonctions de Tableau Desktop ne sont pas prises en charge. C'est le cas notamment des agrégations Médiane, Total (distinct), Rang, Centile, des opérations Inclus/Exclus, etc.) L'extraction des données donne accès à ces fonctions.
- Sécurité des données : si vous souhaitez partager un sous-ensemble de données à partir du système de la source de données, vous pouvez créer un extrait et le mettre à la disposition des autres utilisateurs. Vous pouvez limiter le nombre de champs ou de colonnes à inclure, et partager des données agrégées si vous souhaitez proposer des valeurs récapitulatives au lieu des données au niveau des enregistrements individuels.

Les extraits sont très puissants, mais ne sont pas une panacée. Dans certains cas, l'utilisation d'extraits n'est pas appropriée.

- Données en temps réel : les extraits étant un instantané de données à un moment précis, ils ne sont pas appropriés si vous souhaitez utiliser des données en temps réel. Il est possible d'automatiser l'actualisation des extraits à l'aide de Tableau Server, et de nombreux clients

procèdent de cette manière quotidiennement, mais un accès à des données réellement à jour nécessite une connexion en direct.

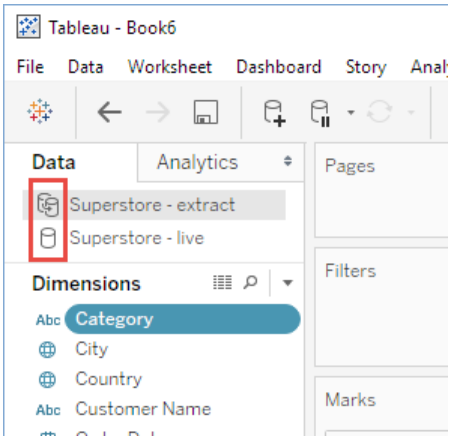
- Données volumineuses : si vous avez besoin d'une énorme quantité de données, c'est-à-dire des millions ou des milliards d'enregistrements, travailler avec un extrait risque de ne pas être pratique. Le fichier résultant serait trop volumineux, ou le processus d'extraction serait trop chronophage. Notez qu'il existe des exceptions à cette recommandation. Si vous disposez d'un ensemble de données volumineux mais souhaitez travailler avec un sous-ensemble filtré, échantillonné ou agrégé de ces données, l'utilisation d'extraits peut être utile. Le moteur d'extraction de Tableau a été conçu pour fonctionner correctement avec quelques centaines de millions d'enregistrements, mais la forme et la cardinalité des données ont un effet sur les performances.
- Fonctions SQL RAW directes : si votre classeur utilise des fonctions directes, sachez que ces dernières ne sont pas compatibles avec les extraits de données.
- Sécurité robuste pour les utilisateurs : si vous avez besoin d'appliquer une sécurité stricte pour les utilisateurs, vous devez la mettre en œuvre dans la source de données. Si des filtres de niveau d'utilisateur sont appliqués au niveau du classeur, un utilisateur peut toujours les retirer pour accéder à l'intégralité des données de l'extrait. Exception à cette recommandation : si l'extrait de données a été publié sur Tableau Server avec des filtres de source de données définis et si d'autres utilisateurs accèdent à cet extrait par le serveur de données. Remarque : n'oubliez pas de révoquer les autorisations de téléchargement des utilisateurs, afin qu'ils ne puissent pas contourner les filtres en place.

Création d'extraits dans Tableau Desktop

Dans la plupart des cas, la création initiale d'un extrait s'effectue dans Tableau Desktop, et c'est une opération très simple. Une fois connecté à vos données, accédez au menu Données et sélectionnez Extraire des données, puis acceptez les valeurs par défaut dans la boîte de dialogue (nous y reviendrons plus tard). Tableau vous invite à choisir un emplacement pour enregistrer l'extrait. Sélectionnez l'emplacement de votre choix, bien que Tableau propose par défaut Mon référentiel Tableau/Sources de données.

Patientez pendant la création de l'extrait. Le temps nécessaire dépend de la technologie de base de données utilisée, de la vitesse du réseau, du volume de données, mais aussi de vitesse et des capacités de votre ordinateur, car la création d'un extrait demande de la puissance de calcul et de la mémoire.

L'icône de la source de données change lorsque l'opération est terminée. Une autre icône de base de données s'affiche derrière, pour signaler qu'il s'agit d'une copie de la première, et c'est exactement ce qu'est un extrait.



Lorsque vous créez un extrait via Tableau Desktop, l'opération se déroule sur votre ordinateur. Vous devez donc vous assurer que celui-ci dispose de suffisamment de ressources pour cette tâche. La création d'extraits utilise tous les types de ressource, processeur, mémoire, espace de stockage, E/S réseau, et le traitement de volumes importants de données sur un ordinateur peu performant peut aboutir à des erreurs si les ressources sont insuffisantes. Pour les extraits volumineux, il est conseillé d'utiliser un ordinateur dont les caractéristiques sont suffisantes : processeur à plusieurs cœurs, beaucoup de RAM, E/S rapides, etc.

La création d'un extrait nécessite un espace disque temporaire pouvant atteindre deux fois la taille du fichier final. Cet espace de travail est attribué dans le répertoire spécifié par la variable d'environnement TEMP (généralement `C:\WINDOWS\TEMP` ou `C:\Users\USERNAME\AppData\Local\Temp`). S'il n'y a pas assez de place sur le lecteur, modifiez la variable pour désigner un emplacement disposant de davantage d'espace.

S'il n'est pas possible de créer le premier extrait sur un poste de travail, vous pouvez recourir à la solution suivante pour créer un extrait vide et le publier ensuite sur Tableau Server. Créez un champ calculé contenant `DateTrunc("minute", now())`. Ajoutez-le ensuite aux filtres de l'extrait et excluez la valeur unique qu'il affiche (soyez rapide, car le filtre reste valide pendant une minute seulement). Si vous avez besoin de plus de temps, augmentez l'intervalle de publication (en arrondissant à 5 minutes, 10 minutes ou 1 heure par exemple). Cela permet de créer un extrait vide sur votre bureau. Lorsque vous le publiez sur Tableau Server et déclenchez la programmation d'actualisation, les données viennent remplir l'extrait, car l'horodatage exclu précédemment sera différent.

Création d'extraits à l'aide de l'API d'extraction de données

Tableau propose également une API pour permettre aux développeurs de créer directement un fichier TDE. Avec cette API, ils peuvent générer des extraits à partir d'un logiciel sur site ou d'un logiciel en tant que service (SaaS, Software-as-a-Service). Cette API permet de récupérer systématiquement les données dans Tableau lorsqu'il n'existe pas de connecteur natif pour la source de données utilisée.

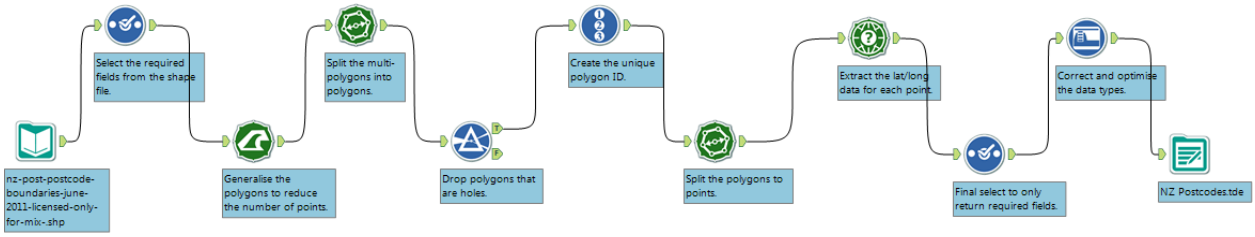
Elle est disponible pour les développeurs en Python et C/C++/Java sous Windows et Linux. Pour en savoir plus sur cette API, reportez-vous à la page suivante :

http://onlinehelp.tableau.com/current/pro/desktop/fr-fr/extracting_TDE_API.html

Création d'extraits à l'aide d'outils tiers

De nombreux développeurs d'outils tiers utilisent l'API d'extraction de données pour ajouter le format de sortie TDE en natif à leurs applications. Ces applications incluent notamment des plates-formes analytiques comme Adobe Marketing Cloud et des outils ETL comme Alteryx et Informatica.

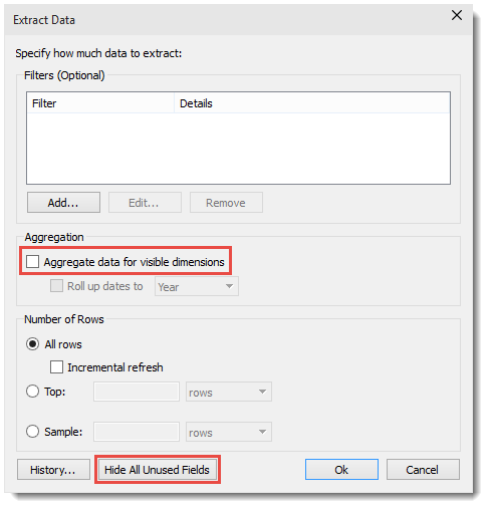
Pour une préparation des données plus exigeante, Alteryx et Informatica permettent d'effectuer les étapes ETL efficacement et de produire les données préparées sous forme de fichier TDE à utiliser dans Tableau Desktop.



Extraits agrégés

L'utilisation d'un extrait agrégé permet toujours d'améliorer les performances. Même si vous utilisez Teradata ou Vertica, avec de très grandes quantités de données, l'extraction permet d'améliorer les performances, du moment que les données sont correctement filtrées et agrégées. Vous pouvez par exemple les filtrer si seules les plus récentes vous intéressent.

Vous pouvez définir l'extrait à l'avance en choisissant les champs souhaités et en sélectionnant l'option « Agréger les données pour les dimensions visibles » dans la boîte de dialogue Extraire les données de Tableau Desktop. Vous pouvez également revenir à la boîte de dialogue Extraire les données et cliquer sur le bouton « Masquer tous les champs inutilisés » après avoir effectué votre analyse et créé votre tableau de bord lorsque vous êtes prêt à publier le tout. Les données extraites représenteront le volume minimal requis pour créer la vue.



La création d'extraits est une technique très puissante lorsque vous disposez d'un important volume de données de base, mais que vous devez créer des vues récapitulatives qui nécessitent des requêtes sur l'ensemble de données complet. Prenons par exemple un milliard d'enregistrements de données transactionnelles détaillées représentant des ventes sur 10 ans, et supposons que vous souhaitiez commencer par afficher la tendance globale des ventes sur cette période. La requête pour la vue initiale sera potentiellement lente, car elle doit envoyer des requêtes sur toutes les lignes. En créant un extrait agrégé au niveau des années, vous pouvez réduire les efforts requis pour les requêtes

au moment de la consultation, car l'extrait comportera uniquement 10 valeurs. Il s'agit évidemment d'un exemple très simpliste. En réalité, vous pouvez disposer d'autres dimensions en plus du temps, mais il est extrêmement utile de pouvoir réduire le nombre d'enregistrements à interroger au moment de la consultation.

Vous pouvez obtenir des classeurs très sophistiqués contenant plusieurs niveaux de détail en créant un extrait agrégé pour chacun de ces niveaux, ou en combinant les extraits agrégés aux connexions en direct. Vous pouvez avoir par exemple un ensemble de vues récapitulatives initiales basées sur un extrait très agrégé, mais en accédant aux détails, vous utilisez des filtres d'action vers une autre feuille qui utilise une connexion en direct. Cela signifie que les vues récapitulatives seront rapides, car elles n'exploitent pas l'intégralité de l'ensemble de données de base, et que vous n'avez pas besoin d'extraire toutes les données de base pour vos actions d'exploration. Par ailleurs, la connexion en direct sera rapide, car vous accédez uniquement à un petit ensemble de données lors de votre exploration.

De cette manière, vous pouvez combiner les solutions et agréger à différents niveaux pour résoudre pratiquement tous les problèmes de performances et obtenir les résultats suffisamment rapidement. Étant donné que Tableau gère efficacement la mémoire, l'amélioration des performances avec cette méthode est généralement simple, et vous pouvez exécuter plusieurs extraits en même temps.

Optimisation des extraits

Tableau Server optimise les colonnes physiques dans la base de données, mais aussi les colonnes additionnelles créées dans Tableau. Ces colonnes incluent les résultats des calculs déterministes, comme les manipulations et concaténations de chaînes, dont le résultat ne change pas, ainsi que les groupes et les ensembles. Les résultats des calculs non déterministes, comme ceux impliquant un paramètre ou des agrégations telles que la somme ou la moyenne qui sont calculés au moment de l'exécution, ne peuvent être stockés.

L'utilisateur peut actualiser un extrait après avoir ajouté seulement deux lignes de données et constater que la taille de l'extrait est passée de 100 Mo à 120 Mo. Cette augmentation est due à l'optimisation, qui produit des colonnes additionnelles contenant les valeurs des champs calculés, car il est plus rentable de stocker des données sur le disque que de les recalculer chaque fois qu'elles sont nécessaires.

Si vous créez des copies d'une connexion vers un extrait de données, vous devez vérifier que tous les champs calculés existent dans la connexion sélectionnée pour les options d'optimisation ou d'actualisation. Si tel n'est pas le cas, Tableau ne matérialisera pas les champs qui sont considérés comme inutilisés. Il est recommandé de définir tous les champs calculés dans la source de données principale, et de les copier selon les besoins dans les autres connexions, puis d'optimiser ou d'actualiser l'extrait depuis la source de données principale.

> *Remarque : comme un collègue nous l'a fait remarquer, bien qu'il soit possible de créer plusieurs connexions dans un fichier TDE unique, Tableau n'a pas été développé pour prendre cela en charge. Vous pouvez rencontrer de nombreux problèmes si les connexions ne sont plus synchronisées avec la structure du fichier TDE. Il est recommandé de ne pas procéder ainsi.*

Actualisation des extraits

Pour actualiser un extrait dans Tableau Desktop, sélectionnez Données > [Votre source de données] > Extrait > Actualiser. Les données sont mises à jour et les éventuelles nouvelles lignes sont ajoutées. Dans Tableau Server, pendant ou après le processus de publication, vous pouvez ajouter une programmation définie par un administrateur pour actualiser l'extrait automatiquement. La plus petite

valeur incrémentielle autorisée est 15 minutes. Vous pouvez également effectuer des actualisations tous les jours à la même heure, toutes les semaines, etc. Vous pouvez établir une « fenêtre flottante » pour actualiser les données en continu et récupérer les données les plus récentes.

Remarque : si vous souhaitez actualiser les données plus souvent que toutes les 15 minutes, il est préférable de vous connecter aux données en direct ou de définir une base de données de rapports synchronisée.

Deux types de programmations d'actualisation sont possibles pour un extrait :

- L'actualisation incrémentielle ajoute uniquement des lignes et n'inclut pas les modifications apportées aux lignes existantes.
- L'actualisation complète remplace l'extrait actuel et en génère un nouveau à partir de la source de données.

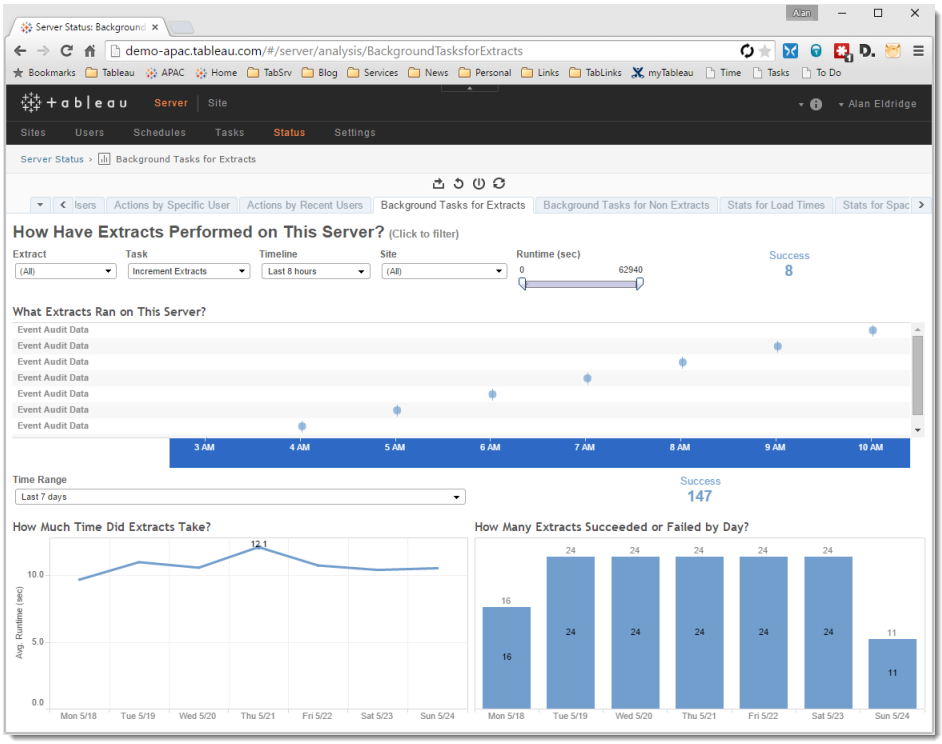
Que se passe-t-il si l'actualisation prend plus de temps que l'incrémentation ?

Si l'actualisation prend plus de temps que l'incrémentation, les actualisations devant intervenir entre-temps sont ignorées. La programmation est par exemple définie pour actualiser les données toutes les heures, mais le volume de données est si important que l'actualisation prend 1 h 30. Par conséquent :

- La première actualisation commence à 1 h et se termine à 2 h 30.
- L'actualisation suivante doit commencer à 2 h, mais comme la première n'est pas encore terminée, elle est ignorée.
- L'actualisation suivante commence à 3 h et se termine à 4 h 30.

Maintenance des extraits

Les écrans de maintenance présentent les tâches d'arrière-plan en cours d'exécution, ainsi que celles qui ont eu lieu au cours des 12 heures précédentes. Des couleurs sont utilisées pour indiquer l'état de ces tâches. Les écrans de maintenance sont disponibles pour les administrateurs et pour certains utilisateurs disposant des autorisations appropriées, qui peuvent lancer une mise à jour ad hoc d'un extrait. Par exemple, si une base de données va être chargée, vous pouvez définir un déclencheur d'extrait à la fin de ce chargement.



Vous pouvez également actualiser un classeur de manière incrémentielle ou par l'outil de ligne de commande `tabcmd` si vous utilisez Tableau Server, ou l'outil de ligne de commande de Tableau.exe si vous utilisez Tableau Desktop. Pour une programmation plus complexe, vous pouvez recourir à un outil de programmation externe, comme le planificateur de tâches Windows. Cette approche est nécessaire si vous avez besoin d'un cycle d'actualisation inférieur au minimum de 15 minutes que permet l'interface de Tableau Server.

Enfin, vous pouvez utiliser le client de synchronisation de Tableau Online pour garder les sources de données sur site à jour grâce à des programmations définies dans le service en ligne. Pour en savoir plus sur cet utilitaire, reportez-vous à la page suivante :

http://onlinehelp.tableau.com/current/online/fr-fr/qs_refresh_local_data.htm

Gouvernance des données

Bien qu'il ne s'agisse pas d'une source de données en soi, vous pouvez vous connecter à des sources de données par le biais du serveur de données de Tableau Server. Le serveur de données prend en charge les connexions en direct et les extraits de données, et offre plusieurs avantages par rapport aux connexions de données autonomes.

- Comme les métadonnées sont stockées de manière centralisée sur Tableau Server, elles peuvent être partagées entre plusieurs classeurs et plusieurs auteurs ou analystes. Les classeurs conservent un pointeur vers la définition centralisée des métadonnées, et à chaque ouverture ils vérifient si des modifications ont été apportées. Le cas échéant, l'utilisateur est invité à mettre à jour la copie intégrée dans son classeur. Cela signifie que les modifications à la logique métier se font à un seul endroit et peuvent ensuite être propagées dans tous les classeurs associés.
- Si la source de données est un extrait, elle peut être utilisée dans plusieurs classeurs. Sans le serveur de données, chaque classeur contient sa propre copie locale de l'extrait. Cela réduit

le nombre de copies redondantes, et de fait l'espace de stockage requis sur le serveur, ainsi que la duplication des processus d'actualisation.

- Si la source de données est une connexion en direct, il n'est pas nécessaire d'installer les pilotes pour cette source de données sur l'ordinateur de chaque analyste. Il suffit de les installer sur Tableau Server. Le serveur de données fait office de proxy pour les requêtes provenant de Tableau Desktop.

L'environnement est-il en cause ?

Parfois, un classeur est efficace lorsqu'il est testé par un seul utilisateur, mais il devient médiocre une fois déployé sur Tableau Server et utilisé par plusieurs personnes. Voyons dans quels domaines les différences entre ces deux scénarios jouent un rôle important.

Mise à niveau

Nos développeurs cherchent constamment à améliorer les performances et l'ergonomie de notre produit. La mise à niveau vers la version la plus récente de Tableau Server permet parfois d'améliorer nettement les performances et la stabilité, sans que vous ayez à modifier le classeur.

Reportez-vous à la page des notes de version et effectuez la mise à niveau vers la dernière version possible :

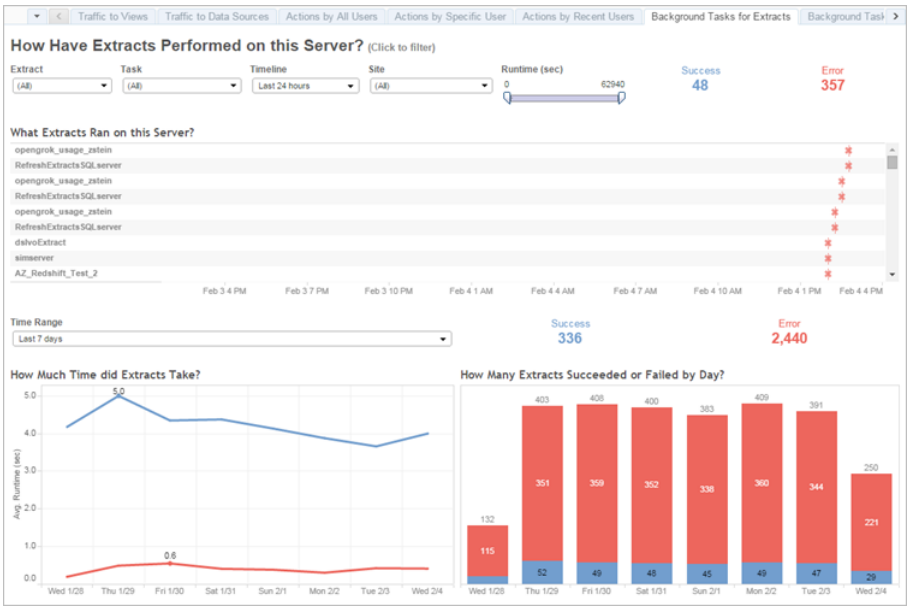
http://www.tableau.com/fr-fr/support/releases

Testez Tableau Desktop sur la machine de Tableau Server

Parfois, votre classeur fonctionne très bien dans Tableau Desktop sur votre ordinateur, mais moins efficacement lorsque vous le consultez par le biais de Tableau Server. Le fait d'ouvrir un classeur dans une copie de Tableau Desktop installée sur le même ordinateur que Tableau Server permet de déterminer si le problème provient du classeur ou d'un problème de configuration du serveur. Cette méthode aide à identifier les problèmes d'incompatibilité des pilotes ou les problèmes de réseau, notamment avec le routage, le DNS ou les configurations du proxy.

Séparez les actualisations et les charges de travail liées à l'interactivité

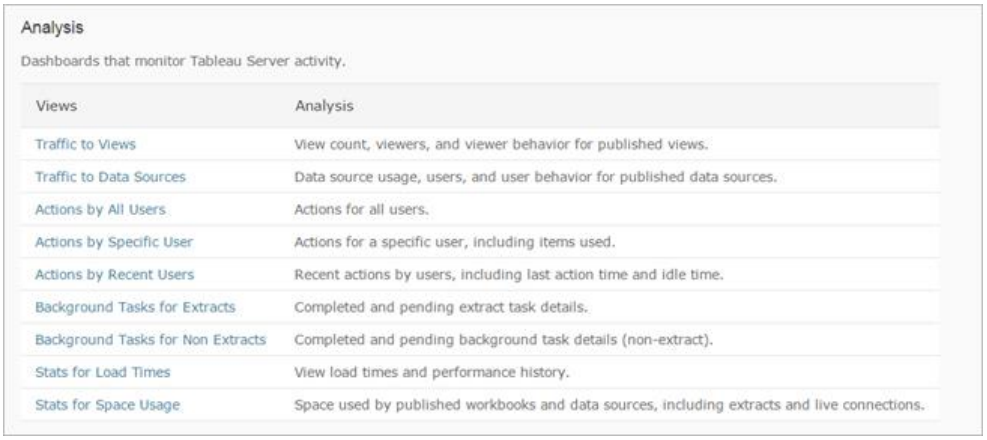
Si les performances du serveur sont faibles, utilisez la vue administrative des tâches d'arrière-plan pour afficher vos programmations d'actualisation actuelles.



Dans la mesure du possible, programmez les actualisations pendant les heures creuses. Si votre configuration matérielle le permet, vous pouvez également déplacer des processus en arrière-plan vers un nœud worker dédié.

Surveillez et optimisez Tableau Server

Dans Tableau Server, plusieurs vues permettent aux administrateurs de surveiller l'activité sur le serveur. Ces vues se trouvent dans la table Analyses de la section Maintenance du serveur.



Pour en savoir plus sur ces vues, rendez-vous sur cette page :

http://onlinehelp.tableau.com/current/server/fr-fr/adminview.htm

Par ailleurs, vous pouvez également créer des vues administratives personnalisées en vous connectant à la base de données PostgreSQL, stockée dans le référentiel Tableau. Rendez-vous sur cette page pour en savoir plus :

http://onlinehelp.tableau.com/current/server/fr-fr/adminview_postgres.htm

Vérifiez le délai d'expiration de la session VizQL

Par défaut, la durée de la session VizQL est limitée à 30 minutes. La session VizQL consomme des cycles de mémoire et de processeur même si elle est inactive. Si une limite inférieure suffit, utilisez tabadmin pour modifier le paramètre vizqlserver.session.expiry.timeout :

http://onlinehelp.tableau.com/current/server/fr-fr/reconfig_tabadmin.htm#ida6864815-db67-4a51-b8b6-c93617582091

Évaluez la configuration des processus

Tableau Server est divisé en plusieurs composants différents appelés services. Leur configuration par défaut a été conçue pour convenir à de nombreux scénarios, mais vous pouvez également les reconfigurer. Vous pouvez notamment savoir sur quels ordinateurs les processus s'exécutent, et combien sont exécutés. Reportez-vous à cette page pour en savoir plus sur l'amélioration des performances de Tableau Server dans les déploiements sur un, deux ou trois ordinateurs :

http://onlinehelp.tableau.com/current/server/fr-fr/perf_extracts_view.htm#idd21e8541-07c4-420f-913e-92dcaf5f0c34

Infrastructure

64 bits

Les versions de Tableau Server antérieures à la version 10 peuvent s'exécuter sur les systèmes d'exploitation Microsoft 32 bits, mais il est fortement recommandé d'utiliser la version 64 bits pour votre environnement de production. La version 32 bits est uniquement recommandée pour le développement et le test.

À partir de Tableau 10, Tableau Server est disponible uniquement en tant qu'application 64 bits.

Utilisez un processeur plus performant et ajoutez de la RAM

En règle générale, vous obtiendrez de meilleures performances avec davantage de cœurs de processeur et de RAM, que vous exécutiez Tableau Server sur un ou plusieurs ordinateurs. Assurez-vous que

votre ordinateur correspond à la configuration matérielle et logicielle requise pour Tableau Server (http://onlinehelp.tableau.com/current/server/fr-fr/requ.htm#ida4d2bd02-00a8-49fc-9570-bd41140d7b74) et lisez cet article de l'aide en ligne sur l'ajout des workers et la reconfiguration (http://onlinehelp.tableau.com/current/server/fr-fr/distrib_when.htm#idfdd60003-298e-47e6-8133-3d2490e21e07) pour savoir si vous devez ajouter des ordinateurs supplémentaires.

TabMon, que nous avons présenté plus tôt dans ce document, est un outil excellent pour collecter des informations sur l'utilisation qui vous aideront à mieux planifier la capacité de votre installation.

Ne sous-estimez pas l'importance des entrées/sorties

Certaines actions dans Tableau sont très gourmandes en E/S, notamment le chargement, la création ou l'actualisation d'un extrait de données, et l'utilisation de SSD au lieu de disques à plateaux peut être bénéfique. Russel Christopher a publié plusieurs articles intéressants sur son blog Tableau Love. Il analyse notamment l'impact du nombre de cœurs, de la cadence du processeur et du nombre d'entrées/sorties par seconde (IOPS) sur les performances globales. Bien qu'il ait réalisé ses tests sur AWS, ces informations sont valables pour tous les environnements :

http://bit.ly/1f17oa3
http://bit.ly/1f17n5O

Infrastructure physique ou virtualisée ?

De nombreux clients déploient Tableau Server sur une infrastructure virtualisée. La virtualisation a toujours eu un impact sur les performances, aussi elle n'offrira pas la même rapidité qu'une installation sur une infrastructure physique. Néanmoins, la technologie des hyperviseurs modernes permet de réduire cet impact de manière significative.

Lorsque vous effectuez une installation sur des machines virtuelles, n'oubliez pas de vérifier que Tableau Server dispose de ressources dédiées pour la RAM et le processeur. S'il partage ces ressources avec d'autres machines virtuelles sur l'hôte physique, cela aura un impact important sur les performances.

Vous pouvez vous reporter au livre blanc « Deploying Extremely Latency-Sensitive Applications in vSphere 5.5 » de VMware pour en savoir plus sur le sujet. La page 15 propose notamment une liste des meilleures pratiques pour les applications sensibles à la latence :

http://www.vmware.com/fr.html

Navigateur

Tableau utilise très largement JavaScript. Par conséquent, la vitesse de l'outil d'interprétation du navigateur a un impact sur la vitesse du rendu. Le marché des navigateurs évolue rapidement et les plus récents sont assez proches en termes de performances, mais il est important de se demander si la version et les performances du vôtre ont un impact sur votre expérience.

Conclusion

Un jour, quelqu'un m'a donné un conseil très sage : « Dites-leur ce que vous allez leur dire, dites-le, puis dites-leur ce que vous leur avez dit ».

Voici donc un récapitulatif des points les plus importants présentés dans ce document :

- Il n'existe pas de solution miracle. De nombreux facteurs peuvent expliquer le ralentissement des performances. Collectez des données pour identifier la source du problème. Attachez-vous ensuite à améliorer ce qui est le plus important, puis passez au niveau suivant, et ainsi de suite. Vous pourrez vous arrêter lorsque vous estimerez que les gains de performance ne valent pas les efforts que vous devez fournir.
- De nombreux classeurs lents sont le fruit d'une mauvaise conception. Privilégiez la simplicité. Ne cherchez pas à montrer trop de choses et concevez vos visualisations de manière à guider l'analyse, dans la mesure du possible.
- N'utilisez pas les données dont vous n'avez pas besoin. Utilisez des filtres, masquez les champs inutilisés et agrégez vos données.
- Ne cherchez pas à lutter contre le moteur de données. Il est là pour vous aider et travaille de manière intelligente. Vous avez donc la certitude qu'il générera généralement des requêtes efficaces.
- Plus vos données sont épurées et mieux elles reflètent la structure de vos questions, plus vos classeurs seront rapides et votre expérience positive.
- Les extraits sont un moyen simple et rapide d'accélérer la plupart des classeurs. Si vous n'avez pas besoin de données en temps réel et n'utilisez pas des milliards de lignes de données, vous pouvez opter pour les extraits.
- Les chaînes et les dates sont plus lentes que les données numériques et booléennes.
- Bien que ce document soit basé sur les meilleurs conseils de professionnels et connaisseurs, il ne s'agit que de recommandations. Vous devrez les tester pour savoir lesquelles permettront d'améliorer les performances dans une situation spécifique.
- La plupart ont un effet limité lorsque vous utilisez de petits volumes de données, et vous pouvez vous contenter de techniques qui ne sont pas optimisées. Il est néanmoins toujours utile de suivre ces recommandations pour tous vos classeurs, au cas où le volume de données augmenterait.
- C'est en forgeant que l'on devient forgeron.

À vous de jouer maintenant !