Construisez vos pipelines de données avec Databricks

Restituez vos analyses avec les dashboards AI/BI

Correction

Retrouvez aussi les requêtes de correction sur GitHub. Vous pouvez également importer le dashboard de référence.

La reformulation. Durée totale écoutée, divisée par le nombre d'abonnés distincts ayant écouté, par offre active à la date de chaque écoute, sur la période filtrée. Chaque mot de cette phrase écarte une requête fausse.

La requête de l'indicateur.

```
SELECT a.offre,
       SUM(e.duree_ecoutee) / 60.0 / COUNT(DISTINCT e.id_utilisateur) AS minutes_par_abonne,
       COUNT(DISTINCT e.id_utilisateur) AS abonnes_actifs
FROM frequence.socle.ecoutes e
JOIN frequence.socle.abonnements a
  ON e.id_utilisateur = a.id_utilisateur
 AND e.horodatage >= a.date_debut
 AND e.horodatage < COALESCE(a.date_fin, CURRENT_TIMESTAMP())
GROUP BY a.offre;
```

La jointure datée est celle du chapitre 4 : un abonné qui a changé d'offre ne compte qu'une fois par écoute. La colonne `abonnes_actifs` n'apparaît pas forcément sur le graphique, mais elle sert au contrôle du point 4. Le filtre de période du dashboard s'applique sur `e.horodatage`.

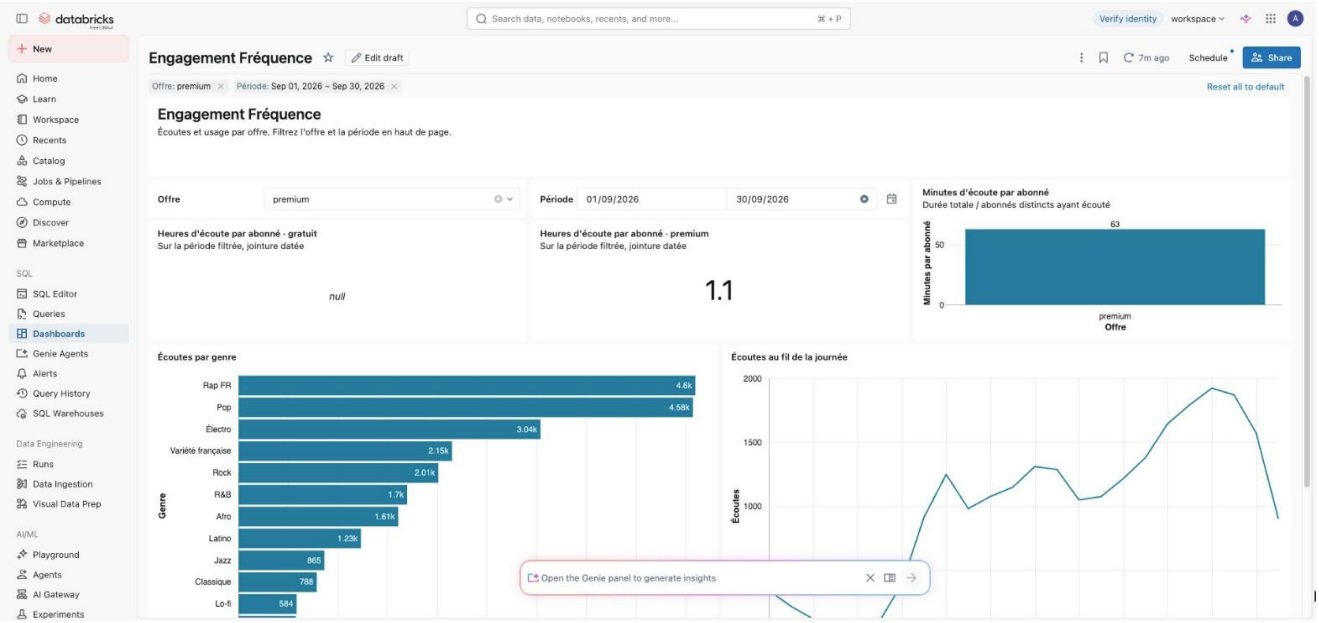
Telle quelle, cette requête agrège déjà toute la période : un filtre de date posé dans le dashboard n'aurait plus de colonne `horodatage` à filtrer. Pour que l'indicateur réagisse à la période, deux options : utiliser un **paramètre** de dates dans le `WHERE` de la requête (`WHERE e.horodatage BETWEEN :date_debut AND :date_fin`), ou exposer un jeu de données au grain jour × offre (durée totale, abonnés) et calculer le ratio dans la visualisation. Attention dans ce second cas : un nombre d'abonnés distincts ne s'additionne pas d'un jour à l'autre. Une troisième option, celle du dashboard de référence, évite ce piège : garder un jeu de données au grain jour × offre × abonné, et définir l'indicateur comme une mesure (**Add custom calculation**) : `SUM(duree_ecoutee) / 60.0 / COUNT(DISTINCT id_utilisateur)`. Le dénombrement des abonnés distincts est alors refait sur la période filtrée, à chaque changement de filtre.

Résultat attendu, sur toute la période : **265 minutes** par abonné pour l'offre gratuite (1 975 abonnés actifs) et **598 minutes** pour l'offre premium (1 387 abonnés actifs), soit environ 2,3 fois plus. Filtré sur premium et septembre 2026 : 63 minutes.

L'erreur la plus probable : la moyenne de moyennes. Deux variantes plausibles et fausses. La première, `AVG(duree_ecoutee)`, donne la durée moyenne d'une écoute, pas d'un abonné : elle est bien plus petite, et répond à une autre question. La seconde calcule d'abord la moyenne par abonné dans une sous-requête, puis fait `AVG` de ces moyennes. Elle donne le même poids à un abonné qui a écouté une fois et à un abonné qui a écouté mille fois. Ce n'est pas absurde (c'est l'« abonné moyen » plutôt que la « minute moyenne »), mais ce n'est pas ce que dit la maquette, et le chiffre diffère. Si

vous l'avez choisie en le disant explicitement dans votre reformulation, le choix est défendable ; si vous l'avez choisie sans le voir, c'est l'erreur à retenir. Les ordres de grandeur trahissent ces deux erreurs : `AVG(duree_ecoutee)` donne 2,46 minutes (gratuit) et 2,90 (premium), la moyenne des moyennes par abonné 2,47 et 2,88, bien loin des 265 et 598 minutes attendues.

Le dashboard de référence.



Le mode de publication. Deux prestataires externes assistent au comité, et aucun jeu de données actuel ne lit `utilisateurs`. Publier en partageant vos permissions sur les données est ici acceptable : les lecteurs voient des agrégats sur des tables non restreintes, et personne n'a besoin de droits sur le catalogue pour ouvrir la page (il faut seulement que chaque lecteur, prestataires compris, dispose d'un accès au compte Databricks de l'entreprise avec lequel vous partagez le dashboard). La justification attendue tient en deux phrases : « Les quatre jeux de données ne lisent que `ecoutes`, `titres` et `abonnements`, aucune table restreinte. Je publie en partageant mes permissions pour que le comité, prestataires compris, ouvre le dashboard sans configuration. »

Dans quel autre contexte l'autre mode s'imposerait-il ? Dès qu'un graphique lirait `utilisateurs`, par exemple une répartition par région ou par tranche d'âge. Publier avec des permissions individuelles garantirait alors que les prestataires voient une erreur à la place du graphique, et que les salariés autorisés le voient. Le bon réflexe n'est pas « toujours tel mode » : c'est de relire la liste des tables lues avant chaque publication, et de refaire ce contrôle à chaque indicateur ajouté.