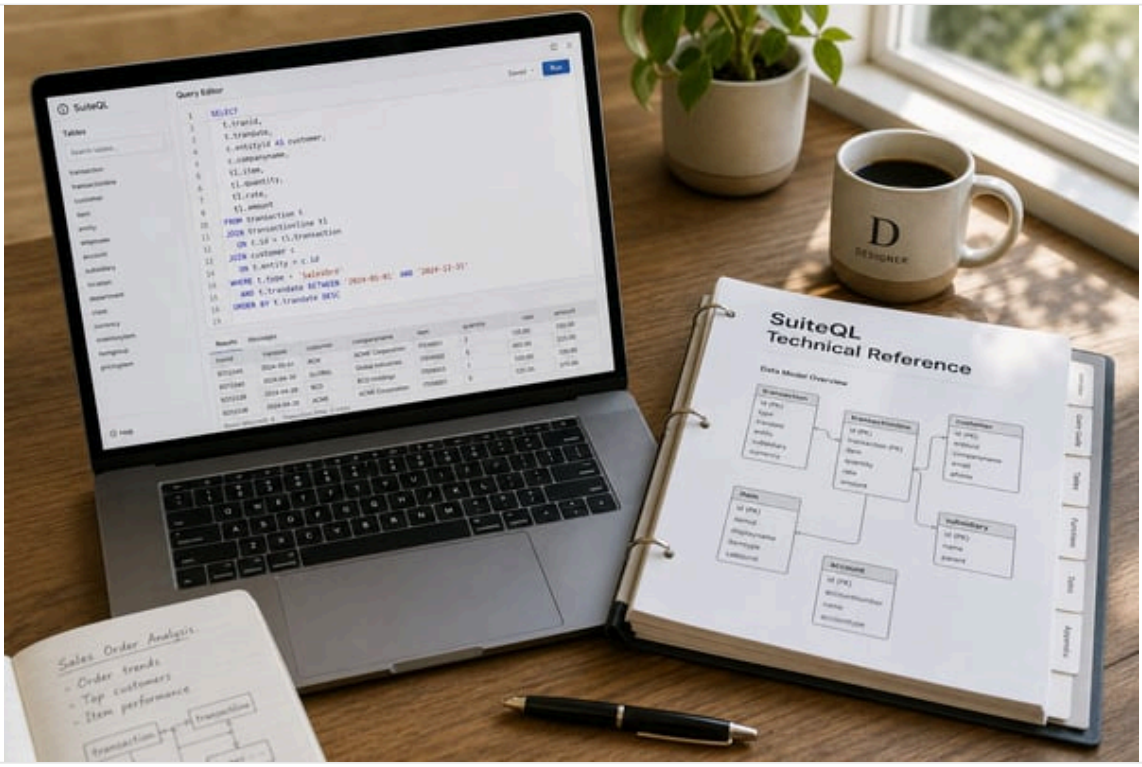
Guide NetSuite SuiteQL : Champs personnalisés, jointures et Connect

Publié le 31 mai 2026 33 min de lecture



Résumé analytique

SuiteQL de SuiteAnalytics Connect est le langage de requête de NetSuite, basé sur la norme ANSI SQL-92, destiné au reporting avancé et à l'analyse de données. Introduit vers 2019–2020, SuiteQL déverrouille le modèle de données ERP/CRM intégré de NetSuite en l'exposant sous forme de tables virtuelles, permettant des requêtes complexes (jointures multi-tables, sous-requêtes, unions, agrégations, etc.) qui dépassent largement les capacités des recherches enregistrées (Saved Searches) traditionnelles ou de SuiteAnalytics Workbook (Source: www.houseblend.io) (Source: www.houseblend.io). Il prend en charge tous les types de jointures SQL standard (inner, left/right/full outer, cartésienne) et les modèles de syntaxe spécifiques à Oracle, tout en appliquant le modèle de sécurité basé sur les rôles de NetSuite. SuiteQL est accessible via SuiteAnalytics Connect (ODBC/JDBC), SuiteScript (module N/query) et les API REST SuiteTalk (Source: docs.oracle.com) (Source: www.houseblend.io).

Cette référence complète couvre : **l'interrogation des enregistrements et champs personnalisés** (les tables de métadonnées `CustomRecordType` et `CustomField`) ; **les types et modèles de jointures** (jointures internes/externes, jointures croisées et bonnes pratiques pour lier les tables) ; et **les bonnes pratiques de SuiteAnalytics Connect** (conventions de nommage, requêtes qualifiées, optimisation des performances et modèles d'intégration). Nous nous appuyons sur la documentation officielle d'Oracle, les blogs de développeurs NetSuite et des analyses sectorielles pour fournir des conseils approfondis. Les points clés incluent :

- L'utilisation des tables `CustomRecordType` et `CustomField` pour découvrir les définitions d'enregistrements personnalisés et leurs champs (Source: timdietrich.me) (Source: www.houseblend.io). Par exemple,

```
SELECT Name, ScriptID, InternalID, Description
FROM CustomRecordType ORDER BY Name;
```

récupère tous les types d'enregistrements personnalisés (nom, ID de script, ID numérique interne, etc.) (Source: timdietrich.me). On filtre ensuite `CustomField` par `RecordType` (l'ID interne de l'enregistrement) pour obtenir ses champs (Source: timdietrich.me) (Source: www.houseblend.io). La colonne `Name` contient l'étiquette visible par l'utilisateur de chaque champ personnalisé, tandis que `ScriptID` est l'ID interne stable (par exemple `custentity_myfield`) (Source: www.houseblend.io) (Source: www.houseblend.io). Les colonnes `FieldValueType` et `FieldValueTypeRecord` décrivent le type de données de chaque champ (Case à cocher, Date, Liste/Enregistrement, etc.) ; lorsqu'un champ provient d'une autre liste/enregistrement, `FieldValueTypeRecord` contient l'ID interne de cette cible (Source: www.houseblend.io) (Source: www.houseblend.io).

- **Jointures et relations.** SuiteQL prend en charge les jointures internes, les jointures externes (gauche/droite/complète) et les jointures cartésiennes (croisées) (Source: docs.oracle.com). Par défaut, SuiteAnalytics Workbook utilise des jointures externes gauches pour les liens d'enregistrement, mais SuiteQL permet à l'utilisateur de spécifier n'importe quel type de jointure (Source: www.houseblend.io). Les modèles standard incluent la jointure sur des champs d'ID internes numériques – par exemple, `customer.id = transaction.entity` lie les clients à leurs ventes. La documentation officielle illustre les jointures (voir tableau ci-dessous) :

TYPE DE JOINTURE	SYNTAXE (SUITEQL)	LIGNES RETOURNÉES	NOTES
Inner Join	`... FROM A JOIN B ON A.key = B.key`	Uniquement les lignes correspondantes	À utiliser pour combiner uniquement les enregistrements correspondants. (Jointure par défaut dans la plupart des moteurs SQL.) (Source: docs.oracle.com)
Left Outer Join	`... FROM A LEFT JOIN B ON A.key = B.key` ou style Oracle `FROM A, B WHERE A.key = B.key(+)`	Toutes les lignes de A, correspondances de B (NULL si aucune correspondance)	Garantit que toutes les lignes de A apparaissent ; SuiteAnalytics Workbook utilise ceci par défaut (Source: docs.oracle.com) (Source: docs.oracle.com).
Right Outer Join	`... FROM A RIGHT JOIN B ON A.key = B.key`	Toutes les lignes de B, correspondances de A (NULL si aucune correspondance)	Similaire à la jointure gauche mais inversée ; les jointures droites style Oracle n'ont pas de raccourci.
Full Outer Join	`... FROM A FULL OUTER JOIN B ON A.key = B.key`	Toutes les lignes de A et B, correspondance sur la clé	Inclut tous les enregistrements des deux tables ; non pris en charge implicitement dans le style Oracle (Source: docs.oracle.com).
Cross Join	`FROM A, B` (implicite) ou `FULL OUTER JOIN ON 1=1`	Produit cartésien (toutes les combinaisons)	SuiteQL ne prend **pas** en charge le mot-clé `CROSS JOIN` ; utilisez la virgule ou l'astuce full-outer pour B.L. (Source: docs.oracle.com).

Par exemple, joindre des clients à des factures ouvertes :

```
SELECT cust.entityid AS customer_id, cust.companyname,
       trx.tranid, trx.total
FROM customer AS cust
JOIN transaction AS trx
  ON cust.id = trx.entity
WHERE trx.type = 'Inv' AND trx.status = 'Open';
```

Cela retourne les factures ouvertes de chaque client (une ligne par facture) (Source: www.houseblend.io) (Source: www.houseblend.io). La capacité à effectuer des jointures entre les modules (ERP↔CRM) est un avantage clé : comme le schéma de NetSuite est unifié, SuiteQL peut combiner de manière transparente les « données ERP » et les « données CRM » (Source: www.houseblend.io). Par exemple, la table `transaction` possède une

colonne *entity* liée à `customer.id`, et un enregistrement `supportcase` possède des champs `company` (client) et `assigned` (employé) (Source: www.houseblend.io). On peut joindre `supportcase.company = customer.id` et `supportcase.assigned = employee.id` pour fusionner les informations de cas et de contact en une seule requête (Source: www.houseblend.io).

- **Bonnes pratiques de SuiteAnalytics Connect.** Les requêtes SuiteQL sur Connect sont exécutées sur la source de données analytiques NetSuite2.com, avec un contrôle d'accès basé sur les rôles appliqué (Source: docs.oracle.com) (Source: docs.oracle.com). Les modèles et contraintes importants incluent :
  - **Conventions de nommage :** Dans l'ancien schéma ODBC NetSuite.com, les noms personnalisés étaient transformés (étiquettes→IDs) en les mettant en majuscules, en remplaçant les espaces par des traits de soulignement et en supprimant les traits d'union (Source: docs.oracle.com). Par exemple, un enregistrement personnalisé nommé « `This-is a-record` » devient `THISIS_ARECORD` (Source: docs.oracle.com). Dans le nouveau schéma NetSuite2.com, les noms de tables et de colonnes utilisent l'*ID de script* spécifié dans l'interface utilisateur, qui est stable. Par exemple, un champ personnalisé avec l'ID d'interface « `custbody1` » reste `custbody1` dans Connect (Source: docs.oracle.com) (Source: docs.oracle.com). Un ID d'enregistrement personnalisé typique dans NetSuite2.com ressemble à `CUSTOMRECORD1` (Source: docs.oracle.com). (Voir tableau ci-dessous.)
  - **Requêtes qualifiées :** Lors de requêtes ODBC entièrement qualifiées, vous devez utiliser le *table_qualifier* (nom de l'entreprise) et le *table_owner* (rôle NetSuite) exacts de votre connexion. Par exemple, si `table_qualifier = "Wolfe Company"` et `table_owner = "Administrator"`, vous émettez :

    ```
    SELECT * FROM "Wolfe Company"."Administrator".account;
    ```

    Notez les guillemets et les espaces – toute orthographe ou casse inexacte entraînera l'échec de la requête (Source: docs.oracle.com) (Source: docs.oracle.com). (La plupart des utilisateurs peuvent omettre la qualification lorsqu'ils travaillent dans leur propre session Connect.)
  - **Sensibilité à la casse / Attribut majuscule :** Le pilote Connect n'est pas sensible à la casse par défaut (vous pouvez interroger en utilisant n'importe quelle combinaison de majuscules/minuscules) (Source: docs.oracle.com). Cependant, les noms de champs et de tables retournés aux applications clientes peuvent changer de casse lors de la mise à jour du schéma. Pour éviter les problèmes dans les outils sensibles à la casse, NetSuite fournit un attribut de connexion `Uppercase` qui force tous les noms de tables/champs en majuscules (Source: docs.oracle.com).
  - **Performance et syntaxe :** En interne, SuiteAnalytics est propulsé par Oracle SQL. Oracle recommande d'écrire les jointures en syntaxe Oracle (en utilisant des virgules et l'opérateur `(+)` pour les jointures externes) pour améliorer les performances et éviter les délais d'attente (Source: docs.oracle.com). Par exemple, au lieu d'un `LEFT JOIN` ANSI, on pourrait écrire :

    ```
    SELECT t.id, t.amount, a.name
    FROM transaction t, account a
    WHERE a.id = t.account AND t.period = ?
          AND p.id(+) = t.period;
    ```

    De telles requêtes de style Oracle s'exécutent souvent plus rapidement via Connect (Source: docs.oracle.com). Évitez les opérations lourdes et les conditions OR complexes dans les grandes requêtes, car elles peuvent empêcher l'utilisation des index (Source: www.houseblend.io) (Source: docs.oracle.com). Il est préférable de **tester et d'itérer** : construisez d'abord des requêtes sur de petits ensembles de données, ou utilisez la fonctionnalité d'exportation vers SuiteQL de Workbook pour prototyper visuellement les jointures (Source: www.houseblend.io).
  - **Chargement incrémentiel des données :** Pour les rafraîchissements ETL ou BI, utilisez les colonnes d'horodatage et de statut fournies. De nombreuses tables dans Connect incluent `last_modified_date` ou `date_last_modified` pour prendre en charge le suivi des modifications. De plus, NetSuite publie une table `deleted_records` listant les enregistrements récemment supprimés. En règle générale, utilisez ces colonnes (ou `deleted_records`) pour effectuer des chargements incrémentiels (Source: docs.oracle.com). Certaines tables de mappage ou statiques n'ont **pas** ces colonnes ; ces tables doivent être entièrement rechargées à chaque fois (Source: docs.oracle.com).
  - **Sécurité et conformité :** SuiteAnalytics Connect est strictement en lecture seule (Source: docs.oracle.com). Les entreprises doivent s'assurer que les outils tiers ingérant les données Connect traitent les champs sensibles de manière appropriée. Par exemple, NetSuite avertit que l'exportation d'informations de santé protégées (ePHI) via Connect nécessite des processus conformes à la loi HIPAA (Source: docs.oracle.com). De même, Connect applique les autorisations de rôle : les données renvoyées par vos requêtes sont limitées à ce que votre rôle Connect choisi peut voir dans l'interface utilisateur (Source: docs.oracle.com).

MODÈLE DE NOMMAGE	IDS DE CHAMPS PERSONNALISÉS	IDS D'ENREGISTREMENTS PERSONNALISÉS	EXEMPLE
Netsuite.com (ancien Connect)	Texte de l'étiquette en majuscules ; espaces→`_` ; traits d'union supprimés (max 29 car.) (Source: docs.oracle.com)	Nom de l'enregistrement en majuscules ; espaces→`_` ; traits d'union supprimés (Source: docs.oracle.com)	"This-is a-record" → `THISIS_ARECORD` (Source: docs.oracle.com)
Netsuite2.com (actuel)	Utilise le champ *ID* de l'interface (ex: `custbody1` pour un champ de corps) (Source: docs.oracle.com) (Source: docs.oracle.com)	Utilise l'ID de script de l'interface (ex: `CUSTOMRECORD1`) (Source: docs.oracle.com)	Le champ "Transaction Body" a l'ID `custbody1` (Source: docs.oracle.com) ; exemple d'enregistrement personnalisé : `CUSTOMRECORD1` .

- **Analyse de données et cas d'utilisation.** Les tendances du secteur et les données clients soulignent l'impact de SuiteQL. NetSuite compte plus de 40 000 clients dans le monde (Source: www.houseblend.io) et a enregistré une croissance de son chiffre d'affaires de 18 % en 2025 (Source: www.houseblend.io). Alors que les analystes exigent de plus en plus d'outils d'analyse ERP dans le cloud (84 % des nouveaux projets BI intègrent des données d'ERP cloud (Source: www.houseblend.io), SuiteQL comble une lacune critique. Les utilisateurs signalent que le « reporting sur les champs personnalisés » est un défi majeur (Source: www.houseblend.io), et SuiteQL (grâce à sa capacité à énumérer et à joindre des champs personnalisés) répond directement à ce besoin. Les implémentations réelles confirment des avantages tangibles. Par exemple, une étude interne de NetSuite (2025) a révélé que les organisations utilisant SuiteAnalytics Connect ont réduit la charge de travail liée aux ETL de 30 à 50 % par rapport aux exportations CSV héritées, grâce à l'interrogation de la base de données cloud en temps réel (Source: www.houseblend.io). Les études de cas illustrent des modèles typiques : les équipes combinent les données CRM et ERP (par exemple, clients, commandes clients, factures) dans des tableaux de bord unifiés (Source: www.houseblend.io) (Source: www.houseblend.io). Un scénario est le **ciblage de campagnes marketing** : un analyste marketing peut utiliser SuiteQL pour extraire des segments de clientèle par zone géographique et historique d'achat, en combinant les résultats via des sous-requêtes ou `UNION` (voir, par exemple, les solutions de Tim Dietrich (Source: www.houseblend.io) (Source: www.houseblend.io). Un autre est le **reporting financier** : un exemple de Coefficient.io montre l'utilisation de SuiteQL pour agréger les montants des factures par compte GL et alimenter ces données dans des feuilles de calcul Tableau sur une base horaire, permettant des tableaux de bord de P&L en temps quasi réel (Source: www.houseblend.io). Étant donné que SuiteQL fonctionne directement sur le modèle de données intégré de NetSuite, les organisations peuvent créer des rapports interdépartementaux riches – par exemple, en joignant les factures ouvertes aux détails des commerciaux pour produire un rapport des comptes clients par commercial en une seule requête (Source: www.houseblend.io).

- **Orientations futures.** À mesure que SuiteQL arrive à maturité, NetSuite ajoute des fonctionnalités qui améliorent l'accès aux métadonnées et l'expérience des développeurs. Actuellement, l'accès au schéma de SuiteQL est largement limité aux métadonnées définies par l'utilisateur (personnalisées). Les métadonnées des enregistrements et des champs standard doivent être obtenues via le **Catalogue d'enregistrements** (interface utilisateur Configuration > Catalogue d'enregistrements) ou via les points de terminaison de métadonnées REST de SuiteTalk (Source: www.houseblend.io) (Source: www.houseblend.io). Notamment, après la version 2021.2, l'ancien Connect Browser (schéma) n'est plus mis à jour, faisant du Catalogue d'enregistrements la source faisant autorité pour les relations entre objets (Source: www.houseblend.io). NetSuite continue d'enrichir Connect : par exemple, de nouvelles tables comme `oa_tables` et `oa_columns` aident à mapper les anciens et les nouveaux ID et révèlent quelles tables/colonnes prennent en charge les chargements incrémentiels ou les suppressions (Source: docs.oracle.com) (Source: docs.oracle.com). L'essor de l'analyse pilotée par l'IA laisse présager des améliorations telles que des générateurs SuiteQL en langage naturel et des API de métadonnées étendues.

En résumé, SuiteQL (accessible via SuiteAnalytics Connect ou SuiteScript) permet aux utilisateurs techniques et métier d'interroger les données NetSuite de manière inédite, notamment en exposant les champs/enregistrements personnalisés et en permettant des jointures complexes. Ce rapport documente les meilleures pratiques pour rédiger des requêtes SuiteQL, illustre comment récupérer des métadonnées et lier des enregistrements, et met en évidence des modèles pour intégrer SuiteQL dans les pipelines de données. Tous les conseils sont étayés par la documentation officielle et des exemples éprouvés de la communauté, garantissant à la fois précision et pertinence pratique (Source: www.houseblend.io) (Source: www.houseblend.io).

Introduction et contexte

NetSuite est une plateforme ERP/CRM cloud de premier plan qui consolide les finances, les stocks, les commandes, les clients, et plus encore dans un système unique (Source: www.houseblend.io). Une caractéristique de NetSuite est son extensibilité : les administrateurs peuvent ajouter des **champs personnalisés** à pratiquement n'importe quel type d'enregistrement, et même définir des types d'**enregistrements personnalisés** entiers contenant des données spécifiques à l'entreprise. Au fil du temps, ces personnalisations permettent aux organisations de capturer des attributs opérationnels détaillés (par exemple, numéros de série de produits, balises de segment client, codes réglementaires) mais compliquent également le reporting et l'intégration. Traditionnellement, les utilisateurs de NetSuite s'appuyaient sur des *Recherches enregistrées* (filtres pointer-cliquer avec jointures limitées) ou sur le *SuiteAnalytics Workbook* (rapports visuels) pour l'analyse (Source: www.houseblend.io). Ces outils, bien que précieux, imposent des contraintes : au plus un niveau de jointures dans les recherches enregistrées, des agrégations limitées et des mises en page de rapport rigides (Source: www.houseblend.io). L'extraction de données externe était possible via le pilote ODBC hérité (« source de données NetSuite.com ») ou des exportations basées sur des fichiers, mais les requêtes complexes étaient peu pratiques.

Pour remédier à cela, NetSuite a introduit **SuiteQL** vers 2019/2020 (Source: www.houseblend.io). SuiteQL est une interface de requête entièrement conforme à SQL-92 superposée à la **source de données Analytics** de NetSuite (maintenant appelée NetSuite2.com). Selon les termes d'Oracle, SuiteQL est « un langage de requête puissant... construit sur SQL-92... conçu pour fournir aux utilisateurs un accès efficace et flexible au modèle de données de NetSuite, permettant des requêtes avancées au-delà des capacités des recherches enregistrées et des rapports » (Source: www.houseblend.io). En pratique, SuiteQL expose les mêmes données que celles que le SuiteAnalytics Workbook peut voir, mais avec toute la flexibilité du SQL – jointures multi-tables, sous-requêtes, UNION, fonctions de fenêtre, et plus encore (Source: www.houseblend.io) (Source: www.houseblend.io). Cela permet aux analystes de répondre à des questions qui étaient auparavant difficiles ou impossibles, comme mélanger les transactions ERP avec les contacts CRM ou auditer des milliers de champs personnalisés en masse.

SuiteQL peut être utilisé de plusieurs manières :

- **SuiteAnalytics Connect (ODBC/JDBC) :** Lorsque Connect est activé, NetSuite fournit des pilotes ODBC/JDBC qui se connectent à l'entrepôt de données NetSuite2.com. Les développeurs et les outils BI (Excel, Tableau, Power BI, etc.) peuvent émettre des requêtes SuiteQL via cette interface (Source: docs.oracle.com) (Source: www.houseblend.io).
- **SuiteScript (module N/query) :** Dans SuiteScript 2.0+, NetSuite fournit un module `N/query` qui accepte des chaînes SuiteQL et renvoie des objets de données. Cela permet aux scripts serveur d'exécuter SuiteQL dans le contexte des rôles utilisateur et des points de gouvernance (Source: www.houseblend.io).
- **SuiteTalk (services Web REST) :** Les API REST de SuiteTalk incluent des points de terminaison (`/suiteql`) qui exécutent des requêtes SuiteQL. Les intégrations peuvent les appeler pour récupérer les résultats au format JSON ou CSV (Source: docs.oracle.com) (Source: www.houseblend.io).

Toutes ces méthodes utilisent le même schéma analytique sous-jacent, soumis aux restrictions basées sur les rôles et aux limites de gouvernance de NetSuite. Notamment, SuiteAnalytics Connect est en **lecture seule** (Source: docs.oracle.com) ; on ne peut pas utiliser SuiteQL pour mettre à jour des données. Son objectif principal est de mettre en mémoire tampon une couche de reporting cohérente et performante. À partir de NetSuite 2026.1, l'ancien schéma « NetSuite.com » est obsolète, et *seule* la source de données NetSuite2.com est prise en charge pour Connect (Source: docs.oracle.com).

Parce que SuiteQL est relativement nouveau, les meilleures pratiques évoluent encore. Le matériel qui suit rassemble les conseils officiels et la sagesse de la communauté. Nous expliquerons comment trouver et interpréter les éléments de schéma NetSuite, comment écrire des requêtes SuiteQL sur des champs et enregistrements personnalisés, comment joindre efficacement des tables et comment intégrer SuiteQL dans les pipelines de données. En cours de route, nous intégrerons des données de référence (par exemple, statistiques d'utilisation interne, exemples de métadonnées) et des scénarios de cas pour illustrer les principes.

SuiteAnalytics Connect et schéma de données

Présentation de SuiteAnalytics Connect

SuiteAnalytics Connect fournit un accès de type base de données aux données NetSuite via des pilotes ODBC/JDBC/ADO.NET. Il expose le schéma d'entrepôt de données **NetSuite2.com** pour le reporting. Les points clés incluent :

- **Source de données NetSuite2.com :** Connect interroge la source de données spécifique à l'entreprise NetSuite2.com. Cet entrepôt est **sécurisé par rôle** : l'utilisateur Connect voit exactement les mêmes données que celles qu'il verrait dans le SuiteAnalytics Workbook sous ce rôle (Source: docs.oracle.com). Par conséquent, tout le monde ne voit que les données autorisées, quel que soit le pilote utilisé. La source NetSuite2.com est constamment synchronisée (quelques heures de décalage) et est cohérente avec les données du Workbook.
- **Authentification :** Les utilisateurs peuvent se connecter via e-mail/mot de passe ou authentification par jeton (TBA/OAuth). Utilisez toujours le dernier pilote et activez l'authentification basée sur certificat si nécessaire (Source: docs.oracle.com).
- **Schémas/Tables :** Lors de la connexion, vous verrez un schéma nommé d'après votre compte et votre rôle. À l'intérieur se trouvent des tables pour tous les enregistrements exposés. Les enregistrements standard (Clients, Transactions, Articles, Employés, etc.) apparaissent sous forme de noms de table singuliers (par exemple, `customer`, `transaction`, `item`). Les enregistrements personnalisés apparaissent sous des noms basés sur leurs ID de script (par exemple, un enregistrement personnalisé avec l'ID de script `customrecord_projects` devient la table `customrecord_projects`). Pour découvrir les tables et champs disponibles, utilisez des outils comme le **Catalogue d'enregistrements** (UI) (Source: www.houseblend.io) ou interrogez directement les tables système Connect (`oa_tables`, `oa_columns`) (Source: docs.oracle.com).

Il est important de noter que le Connect Browser (navigateur de schéma hérité) n'est **plus mis à jour** depuis la version 2021.2 (Source: www.houseblend.io). NetSuite s'appuie désormais sur le **Catalogue d'enregistrements** (Configuration > Enregistrements > Catalogue d'enregistrements) comme source faisant autorité pour les détails du schéma. Le Catalogue d'enregistrements affiche, pour chaque type d'enregistrement, tous ses champs (avec ID de script) et tous les champs de jointure intégrés (par exemple, sur un enregistrement Client, il montre que *SalesRep* est une référence d'employé, menant à une jointure sur la table Employé) (Source: houseblend.io) (Source: www.houseblend.io). Les analystes doivent utiliser le Catalogue d'enregistrements ou `oa_tables`/`oa_columns` pour confirmer les noms et les relations avant d'écrire des requêtes.

Conventions de nommage (Champs/Enregistrements personnalisés)

La gestion des objets personnalisés nécessite de comprendre le nommage de NetSuite. Le guide **Types d'enregistrements et champs** couvre cela en détail (Source: docs.oracle.com) (Source: docs.oracle.com). En résumé :

- **Ancien modèle (NetSuite.com) :** Les ID étaient dérivés des étiquettes de l'interface utilisateur en les mettant en majuscules et en supprimant les caractères spéciaux. Les espaces devenaient des traits de soulignement, les traits d'union étaient supprimés. Par exemple, un enregistrement personnalisé nommé *« This-is a-record »* obtenait l'ID `THISIS_ARECORD` (Source: docs.oracle.com), et un champ personnalisé étiqueté *« Transaction Body »* pouvait devenir quelque chose comme `TRANSACTION_BODY_CUSTBODY1`. Cependant, ce modèle est désormais obsolète.

- **Nouveau modèle (NetSuite2.com) :** Les noms de table et de colonne utilisent l'ID de script stable défini dans NetSuite au moment de la personnalisation. Renommer l'étiquette dans l'interface utilisateur ne change *pas* l'ID de script. Par exemple, un champ de corps de transaction dont l'ID de script est `custbody1` reste `custbody1` dans les requêtes SuiteQL (Source: docs.oracle.com) (Source: docs.oracle.com). De même, les types d'enregistrements personnalisés ont des ID comme `CUSTOMRECORD1`, `CUSTOMRECORD2`, etc., ou l'ID de script littéral s'il est défini (par exemple, `customrecord_projects`) (Source: docs.oracle.com). Lors de l'écriture de requêtes, utilisez toujours ces ID de script. En cas de doute, vérifiez en interrogeant les tables de métadonnées Connect ou en consultant le Catalogue d'enregistrements.

  Une autre ressource utile est la table système `oa_columns` : elle mappe les noms Connect aux noms de l'interface utilisateur. Par exemple, on peut interroger

  ```
  SELECT column_name, table_name, remarks
  FROM oa_columns
  WHERE column_name LIKE 'CUST%' AND table_name = 'TRANSACTION';
  ```

  Cela listerait tous les champs personnalisés (ID) sur la table Transaction et leurs étiquettes d'interface utilisateur correspondantes (Source: docs.oracle.com).

Trouver les ID d'enregistrement et de champ

Pour écrire une requête SuiteQL, vous devez connaître les noms exacts de table et de colonne à utiliser. Le **Catalogue d'enregistrements** et les outils de schéma aident à cela. Les conseils officiels proposent plusieurs approches :

- **Catalogue d'enregistrements (UI) :** Comme indiqué ci-dessus, via Configuration → Catalogue d'enregistrements, vous pouvez parcourir chaque enregistrement, voir tous les champs (y compris standard et personnalisés, avec ID de script) et voir les références joignables. C'est le point de départ recommandé (Source: www.houseblend.io). Par exemple, dans l'enregistrement Client du catalogue, vous pourriez voir un champ *ID interne* (clé de table), plus un champ *Filiale* étiqueté comme faisant référence à la table Filiale, etc. (Source: www.houseblend.io).

- **Tables de métadonnées SuiteQL :** SuiteQL expose en fait des métadonnées pour les éléments personnalisés. Deux tables sont inestimables :

  - `CustomRecordType` – une ligne par type d'enregistrement personnalisé, avec des colonnes comme `Name` (étiquette), `ScriptID`, `InternalID`, etc. Ainsi `SELECT Name, ScriptID, InternalID FROM CustomRecordType;` liste tous les types d'enregistrements personnalisés (Source: timdietrich.me) (Source: www.houseblend.io). (L' `InternalID` est l'ID numérique de NetSuite, souvent utilisé comme clé étrangère.)

- `CustomField` – une ligne par définition de champ personnalisé (issu de n'importe quel enregistrement ou liste). Les colonnes incluent `Name` (libellé du champ), `ScriptID`, `Description`, `RecordType` (l'ID interne de l'enregistrement parent), `FieldType` (« BODY » ou « COLUMN »), `FieldValueType` et `FieldValueTypeRecord` (s'il s'agit d'un type de liste/enregistrement) (Source: www.houseblend.io) (Source: www.houseblend.io). Exemple : l'exécution de `SELECT Name, ScriptID, FieldType, FieldValueType FROM CustomField WHERE RecordType = 297;` pourrait lister les champs de l'enregistrement personnalisé ayant l'InternalID 297, comme le montre Dietrich (Source: timdietrich.me) (Source: www.houseblend.io).

  Comme SuiteQL n'expose pas (encore) directement les métadonnées des enregistrements standard, le modèle est le suivant : trouvez d'abord l'ID interne d'un enregistrement (via `CustomRecordType` pour les enregistrements personnalisés), puis utilisez cet ID pour filtrer `CustomField`. Vous pouvez également joindre ces deux tables : `CustomField.RecordType = CustomRecordType.InternalID` pour associer les champs à leurs noms d'enregistrement (Source: www.houseblend.io) (Source: www.houseblend.io). Par exemple, pour trouver tous les champs de l'enregistrement personnalisé *Projets*, trouvez son `InternalID` dans `CustomRecordType`, puis interrogez `CustomField WHERE RecordType = <cet ID>` (Source: timdietrich.me).

En guise de flux de travail pratique : vous pourriez exécuter une première requête comme :

```
SELECT Name, ScriptID, InternalID
FROM CustomRecordType
WHERE ScriptID = 'customrecord_projects';
```

Supposons que cela renvoie InternalID = 221. Exécutez ensuite :

```
SELECT Name, ScriptID, FieldValueType
FROM CustomField
WHERE RecordType = 221;
```

pour lister chaque champ de cet enregistrement personnalisé. Ces requêtes peuvent être intégrées dans des scripts ou des outils (voir l'outil SuiteQL Query Tool de Houseblend) à des fins de découverte et de documentation (Source: timdietrich.me) (Source: www.houseblend.io).

Interrogation des enregistrements et champs personnalisés

Une fois que vous avez identifié les tables et les champs pertinents, l'écriture des requêtes SuiteQL suit les modèles SQL standard. Cependant, les champs personnalisés présentent quelques particularités à noter :

- **Libellés de champs vs ID de script :** Dans les ensembles de résultats SuiteQL, les *noms de colonnes* correspondent aux ID de script (par exemple `custentity_age`, `custbody_total`, etc.), et non aux libellés conviviaux. La colonne `CustomField.Name` contient le libellé (par exemple « Âge du client ») (Source: www.houseblend.io), mais lorsque vous sélectionnez des données, vous le faites par scriptID :

```
SELECT custbody_total AS total_tax, custentity_country AS country
FROM transaction
WHERE ...
```

Vérifiez toujours l'orthographe/la casse de ces ID (bien que Connect soit insensible à la casse, l'utilisation de minuscules est courante).

- **Champs de sous-liste (lignes) :** Les champs personnalisés peuvent se trouver sur le corps d'un enregistrement ou sur des sous-listes (lignes d'articles). La colonne `FieldType` dans `CustomField` indique « BODY » ou « COLUMN » (sous-liste) pour chaque champ personnalisé (Source: www.houseblend.io). Vous pouvez interroger les champs de sous-liste en sélectionnant à partir de la table d'enregistrement ou d'une jointure ; ils apparaissent sous forme de colonnes (par exemple, sur les commandes client, les champs de colonne d'article personnalisés apparaissent directement dans les lignes de la table `transaction` pour chaque ligne). Pour filtrer ou agréger sur ces champs, incluez-les selon les besoins. (Houseblend note que certaines solutions filtrent `CustomField.fieldType` dans les requêtes de métadonnées pour séparer ces types (Source: www.houseblend.io).)

- **Champs de type Liste/Enregistrement :** Si un champ personnalisé est de type « Liste/Enregistrement » ou à sélection multiple, la colonne `FieldValueTypeRecord` contient l'ID interne de la liste référencée. Par exemple, un champ personnalisé « Département » sur un client peut avoir `FieldValueType = 'List/Record'` et `FieldValueTypeRecord = 16` (si les départements ont l'ID 16). Vous pouvez effectuer une jointure pour identifier de quoi il s'agit :

  ```
  SELECT cf.Name AS field_label, sr.`name` AS list_name
  FROM CustomField cf
  JOIN ScriptRecordType sr ON cf.FieldValueTypeRecord = sr.internalid
  WHERE cf.FieldValueType = 'List/Record'
    AND cf.RecordType = 123;
  ```

  Cela vous indique les noms conviviaux des listes ou enregistrements référencés (Source: www.houseblend.io). En pratique, si vous connaissez l'ID de script du champ (par exemple `custbody_dept`), vous pouvez également interroger le champ dans son contexte et récupérer simplement la valeur jointe :

  ```
  SELECT c.entityid, c.companyname, d.name AS Department
  FROM customer c
  LEFT JOIN department d ON c.custentity_dept = d.id;
  ```

  Ici, `custentity_dept` est un champ de département personnalisé sur le client, stockant l'ID du département, que nous joignons à la table intégrée `department`.

- **Filtrage par indicateurs de métadonnées :** La table `CustomField` inclut des indicateurs booléens tels que `IsMandatory`, `IsInactive`, etc. Ceux-ci peuvent être utilisés pour auditer vos métadonnées. Par exemple, on pourrait exécuter :

  ```
  SELECT ScriptID, Name FROM CustomField
  WHERE IsMandatory = 'T' AND IsInactive = 'F';
  ```

  pour lister tous les champs personnalisés actifs marqués comme obligatoires. De même, `IsShowInList` indique si un champ est affiché dans les vues de liste par défaut, ce qui peut affecter la visibilité dans les rapports (Source: www.houseblend.io).

Exemple – Audit des champs personnalisés : HouseBlend analyse une requête publiée (exemple CloudExtend) qui inventorie les champs personnalisés à des fins d'audit (Source: www.houseblend.io) (Source: www.houseblend.io). Une approche similaire est :

```
SELECT cf.ScriptID, cf.Name, cf.FieldValueType, cf.IsMandatory, cr.Name AS RecordName
FROM CustomField cf
LEFT JOIN CustomRecordType cr ON cf.RecordType = cr.InternalID
ORDER BY cr.Name;
```

Cela joint chaque champ à son enregistrement personnalisé (le cas échéant). Cela pourrait montrer, par exemple, que `custentity_age (Âge du client)` est un champ obligatoire sur l'enregistrement Client, tandis que `custbody_notes` sur les factures est facultatif. De tels audits de métadonnées aident à la gouvernance en permettant aux administrateurs de vérifier que les configurations de champs correspondent à la politique.

Jointures et modèles relationnels

La puissance de SuiteQL réside dans la jointure de données entre les tables. Nous avons déjà vu les jointures de base sur les clés primaires/étrangères. Voici quelques modèles courants :

- **Jointures un-à-plusieurs (par ex. lignes de commande) :** Un modèle standard est la jointure un-à-plusieurs, comme la liaison des transactions à leurs clients, ou des lignes d'articles à la transaction parente. Par exemple :

  ```
  SELECT t.tranid, t.datecreated, t.entity AS customer
  FROM transaction t
  JOIN customer c ON t.entity = c.id;
  ```

  renvoie chaque transaction avec son client. Si vous souhaitez également des détails sur les sous-listes, vous pouvez joindre la table `transaction` à elle-même sur `transaction.id = transaction.parent`, ou utiliser l'inclusion pour la sous-liste (certaines tables de sous-liste sont exposées séparément dans Connect, selon le type d'enregistrement). Le catalogue d'enregistrements peut guider ces jointures (Source: www.houseblend.io).

- **Jointures plusieurs-à-un (champs de recherche) :** Parfois, une transaction ou un enregistrement possède des champs de recherche. Par exemple, un enregistrement de commande peut avoir un champ `salesrep` (employé) et `currency` (filiale). Vous joignez `transaction.salesrep = employee.id` et `transaction.custbody_subsidiary = subsidiary.id`. La documentation de NetSuite et le catalogue d'enregistrements spécifient ces champs de clé étrangère. Le guide HouseBlend décrit l'interrogation des comptes clients par représentant commercial comme exemple (Source: www.houseblend.io).

- **Jointures inter-modules :** Comme indiqué, de nombreuses questions métier nécessitent des données provenant de plusieurs modules. Parce que l'ERP et le CRM de NetSuite sont unifiés, SuiteQL peut les croiser facilement. Par exemple, pour trouver les factures ouvertes par client, on pourrait joindre `customer` avec `transaction` sur `customer.id = transaction.entity`, en filtrant pour `transaction.type = 'CustInvc'` (facture) et `status='Open'` (Source: www.houseblend.io). Ou pour analyser le ROI marketing, on pourrait joindre `campaign` à `customer` (via les enregistrements de membres de campagne, etc.), ou joindre projet vs client vs facturation. HouseBlend souligne que toutes les données visibles dans Workbook (par exemple dans les jointures intégrées) peuvent être reproduites dans SuiteQL (Source: www.houseblend.io). Le code suivant illustre une jointure CRM-vers-ERP :

  ```
  SELECT c.companyname, o.tranid AS order_id, o.total AS order_amount
  FROM customer c
  LEFT JOIN transaction o ON c.id = o.entity
  WHERE o.type = 'SalesOrd' AND o.status = 'Billed';
  ```

  Cela renvoie tous les clients et leurs commandes client facturées (le cas échéant). L'utilisation d'un LEFT JOIN garantit que les clients sans commande apparaissent toujours (avec des NULL).

- **Jointures impliquant des enregistrements personnalisés :** Si vous avez un type d'enregistrement personnalisé (disons `customrecord_projects`), vous pouvez le joindre à des enregistrements standard s'il existe des champs les reliant. Par exemple, si chaque commande client possède un champ personnalisé `custbody_project` (une liste/enregistrement pointant vers l'enregistrement Projets), vous

pourriez faire :

```
SELECT so.tranid, so.amount, p.Name AS project_name
FROM transaction so
JOIN customrecord_projects p ON so.custbody_project = p.id
WHERE so.type = 'SalesOrd';
```

Ici, `custbody_project` contient l'ID interne du projet. Notez que nous traitons la table d'enregistrement personnalisé exactement comme une table intégrée (le nom de la table est le scriptid de l'enregistrement personnalisé, tout en minuscules selon le schéma Connect).

- **Types de jointures avancés :** - **Right Outer Joins :** Similaires aux left joins mais garantissant que toutes les lignes de la table de *droite* apparaissent. Par exemple, si vous voulez tous les employés, même ceux qui n'ont fait aucune vente, vous pourriez faire `RIGHT JOIN transaction ON employee.id = transaction.salesrep`. Cependant, le style Oracle (Connect) n'a pas de raccourci pour les jointures à droite au-delà du mot-clé `RIGHT JOIN` (Source: docs.oracle.com).

  - **Full Outer Joins :** Rarement nécessaires en pratique, mais pris en charge. Une jointure complète peut produire des lignes sans correspondance de chaque côté (y compris celles avec des NULL des deux côtés s'il n'y a pas de correspondance). Exemple :

    ```
    SELECT c.id AS custid, c.companyname, o.tranid AS order_id
    FROM customer c
    FULL OUTER JOIN transaction o ON c.id = o.entity;
    ```

    Cela récupère tous les clients (avec ou sans commandes) et toutes les commandes (avec ou sans clients) (Source: docs.oracle.com) (Source: docs.oracle.com). SuiteQL prend en charge `FULL OUTER JOIN`, mais il n'y a **aucune syntaxe implicite** pour cela (vous devez utiliser le mot-clé) (Source: docs.oracle.com).

  - **Cross (Cartesian) Joins :** Généralement évités sauf dans les cas d'utilisation analytiques. Il n'y a pas de `CROSS JOIN` explicite dans SuiteQL. On effectue un produit cartésien en listant les tables sans clause ON (par exemple `FROM A, B`) ou en effectuant une jointure externe complète avec `ON 1=1` (Source: docs.oracle.com). Les deux produisent le produit cartésien (chaque combinaison de lignes de A et B) (Source: docs.oracle.com). Cela peut être utile pour certaines analyses matricielles, mais à utiliser avec prudence car cela fait exploser la taille des résultats.

Tableau : Types de jointures SuiteQL (résumé, avec exemple d'utilisation) est présenté ci-dessus.

Meilleures pratiques pour les jointures

- **Joindre sur les ID :** Joignez toujours sur les champs d'ID interne. Il est courant que `[table_enfant].entity` ou `[table_enfant].customer` contienne l' `id` du parent. Par exemple, `transaction.entity = customer.id`, `transaction.item = item.id`, `supportcase.company = customer.id` (Source: www.houseblend.io). Ne vous fiez pas aux libellés ou aux noms pour joindre (par exemple, ne joignez pas `customer.entityid = supportcase.company` car `entityid` est du texte).
- **Tester les relations :** Utilisez le catalogue d'enregistrements pour confirmer les relations. Par exemple, la vue d'ensemble de l'enregistrement Client montre les champs *DefaultShippingAddress, Subsidiary, SalesRep* – chacun a un libellé de champ « interne » indiquant à quelle table il se lie (Source: www.houseblend.io). Le catalogue indiquera explicitement les champs de jointure. Vous pouvez également effectuer une recherche enregistrée rapide ou une jointure Workbook – souvent ces outils d'interface utilisateur se limitent aux jointures valides, et vous pouvez ensuite inspecter le SQL.
- **Diviser les jointures complexes :** Si une jointure semble impliquer de nombreuses tables, envisagez de la diviser en parties plus simples. Le guide HouseBlend suggère de construire la jointure progressivement pour vérifier chaque étape (Source: www.houseblend.io). De plus, préférez la syntaxe JOIN explicite pour plus de clarté, mais rappelez-vous l'astuce Oracle d'utiliser `(+)` pour les performances dans Connect.
- **Éviter les jointures excessives :** Ne renvoyez que les tables et les champs dont vous avez besoin. Chaque table supplémentaire dans une jointure multiplie le traitement. Si vous n'avez besoin que de quelques colonnes d'une table enfant, évitez `SELECT *`. Listez les colonnes explicitement pour minimiser le transfert de données.
- **Cardinalité :** Comprenez la cardinalité de la relation (un-à-plusieurs, plusieurs-à-plusieurs). Pour un-à-plusieurs, un GROUP BY peut être nécessaire si vous souhaitez résumer. Soyez prudent avec les jointures qui pourraient dupliquer les lignes (par exemple, joindre deux relations 1-

à-plusieurs ensemble peut faire exploser les résultats).

- **Performance :** Les jointures sur les champs indexés (clés primaires) sont les plus rapides. NetSuite indexe généralement les ID internes. La liaison sur des champs non clés (ou sur des champs de formule) peut être lente. Dans la mesure du possible, filtrez (avec WHERE) sur les tables *avant* de joindre, pour réduire le nombre de lignes.

Syntaxe et fonctions des requêtes SuiteQL

SuiteQL suit largement la norme ANSI SQL-92, avec certaines extensions Oracle prises en charge (et requises) pour Connect. Points clés :

- **Structure de la requête :** Les clauses SQL habituelles s'appliquent : `SELECT … FROM … [JOIN … ON …] WHERE … GROUP BY … HAVING … ORDER BY …` et `LIMIT`. SuiteQL prend également en charge `UNION` et `UNION ALL` pour combiner les résultats de requête (Source: www.houseblend.io). Les sous-requêtes illimitées sont autorisées dans la clause `FROM` (les expressions de table communes via `WITH` sont également prises en charge dans les versions récentes).

- **Fonctions :** SuiteQL inclut un ensemble de fonctions intégrées (numériques, chaînes, date). Les fonctions prises en charge sont documentées (par exemple `ABS`, `SUBSTR`, `BUILTIN.DF(column)` pour des recherches conviviales, `TO_CHAR`, etc.). Remarque : toutes les fonctions Oracle n'existent pas. La documentation liste les [fonctions et formules prises en charge] (Source: www.houseblend.io). Les fonctions courantes incluent : `TO_CHAR(date, format)`, `COUNT()`, `SUM()`, `AVG()`, `MIN()`, `MAX()` et `BUILTIN.DF()` pour résoudre les noms de champs de référence.

- **BUILTIN.DF (Display Field) :** Une fonctionnalité unique est `BUILTIN.DF()`. Lorsque vous sélectionnez un champ d'ID interne (comme `entity`), vous pouvez l'envelopper pour obtenir le nom lisible par l'homme : par exemple `BUILTIN.DF(transaction.entity) AS customer_name` renvoie le nom du client au lieu de l'ID. Ceci est purement pour l'affichage dans les résultats ; cela ne modifie pas les jointures. Exemple :

  ```
  SELECT t.id, t.entity, BUILTIN.DF(t.entity) AS customer_name
  FROM transaction t;
  ```

  renvoie l'ID d'entité numérique et son nom correspondant. (Source: timdietrich.me) C'est particulièrement utile lors de l'envoi de résultats vers des outils externes.

- **Interrogation de suites d'enregistrements :** SuiteQL permet de filtrer sur les types d'enregistrement (par exemple `WHERE type = 'SalesOrd'` dans transaction) et sur des champs spécifiques. Certains champs sont multivalués ou complexes ; le filtrage sur ceux-ci peut nécessiter une syntaxe différente. Par exemple, pour filtrer les champs à sélection multiple par une valeur de liste interne, utilisez l'opérateur `HASCHILDREN` ou `contains` (selon le type de données). La documentation SuiteAnswers sur la *Syntaxe et les limitations de SuiteQL* doit être consultée pour les cas complexes.

Modèles et considérations de SuiteAnalytics Connect

Cette section décrit les modèles spécifiques à l'utilisation de SuiteQL via le service et la source de données Connect, y compris l'optimisation des performances, les chargements parallèles et l'intégration.

Performances et meilleures pratiques

Comme indiqué, NetSuite recommande d'utiliser une syntaxe de type Oracle pour optimiser les requêtes à l'intérieur de l'entrepôt Connect (Source: docs.oracle.com). De plus :

- **Limiter les lignes tôt :** Utilisez les clauses WHERE le plus tôt possible pour filtrer les lignes. Pour les très grandes tables (factures, commandes client, écritures de journal), filtrez toujours par date ou statut avant de joindre à des tables plus légères. Si vous n'avez besoin que de données récentes, spécifiez-le dans le SQL.

- **Agrégation stratégique :** Si vous avez besoin de totaux ou de décomptes, utilisez `GROUP BY` et `HAVING` au lieu de récupérer toutes les lignes pour les traiter en dehors de la base de données. Gardez toutefois à l'esprit que `TOP` ou `LIMIT` ne provoquent pas nécessairement un court-circuit de l'évaluation dans SuiteQL ; le moteur peut toujours analyser toutes les lignes (Source: www.houseblend.io).

- **Évitez les OR sur les analyses volumineuses :** Pour les filtres à conditions multiples, essayez de réécrire les requêtes séparément et de les combiner (en utilisant `UNION`) plutôt que d'utiliser une clause `OR` complexe, ce qui peut contourner les index (Source: www.houseblend.io).
- **Testez dans un environnement Sandbox :** Si possible, testez au préalable les requêtes lourdes dans un environnement sandbox ou avec des jeux de données plus restreints. L'outil de requête SuiteQL (SuiteQL Query Tool) et les Workbooks permettent tous deux d'exécuter du SuiteQL ; vous pouvez y valider votre logique.

Requêtes incrémentielles et sauvegardes

Pour les intégrations BI, vous souhaitez souvent ne récupérer que les données modifiées. SuiteAnalytics Connect propose des mécanismes adaptés :

- **Horodatages de dernière modification :** De nombreuses tables dans NetSuite2.com incluent `lastModifiedDate` ou `DateLastModified`. Celles-ci sont adaptées aux chargements incrémentiels – par ex. `WHERE lastmodifieddate >= '2025-05-01'`.
- **Enregistrements supprimés :** Il existe une table `deleted_records` qui journalise les suppressions de lignes (avec le nom de la table, l'ID interne et la date de suppression). Joignez ou filtrez cette table pour identifier et supprimer les éléments supprimés de votre extraction.
- **Tables non prises en charge :** Certaines tables (notamment les tables de mappage ou de jointure) peuvent ne pas prendre en charge les horodatages ou les suppressions (Source: docs.oracle.com). Dans ces rares cas, la seule approche sûre consiste à recharger la table entière à chaque synchronisation. Les recommandations de NetSuite sont de « signaler un problème » (file an issue) si une table importante ne dispose pas de colonnes incrémentielles (Source: docs.oracle.com).

Utilisation de SuiteQL dans les pipelines d'analyse

Les flux de travail BI modernes ingèrent souvent des données depuis NetSuite via SuiteQL. Les modèles typiques incluent :

- **Connecteur BI direct :** Des outils comme Tableau ou Power BI (via ODBC) peuvent interroger SuiteQL directement. Certains fournisseurs (par ex. Coefficient.io) proposent des connecteurs SuiteQL qui encapsulent les requêtes et gèrent la planification. Par exemple, l'étude de cas de Coefficient décrit la création d'une requête SuiteQL additionnant les lignes de facture par compte et l'écriture des résultats dans Tableau selon un planning horaire (Source: www.houseblend.io). Étant donné que le connecteur NetSuite natif de Tableau présente des limites, l'approche SuiteQL permet une agrégation multi-tables que les connecteurs standard ne peuvent pas gérer (Source: www.houseblend.io).
- **Entrepôts de données (Data Warehouses) :** Les organisations peuvent utiliser des plateformes ETL (Fivetran, Talend, etc.) pour charger les résultats SuiteQL dans des entrepôts de données externes (Snowflake, BigQuery) pour un traitement ultérieur. Dans de tels scénarios, on utilise souvent l'API du pilote Connect pour planifier les requêtes (ou un connecteur intégré s'il est disponible). L'économie de 30 à 50 % sur le travail ETL rapportée par NetSuite (en éliminant les exportations CSV) est largement réalisée grâce à ce modèle (Source: www.houseblend.io).
- **Applications personnalisées et intégration :** Les développeurs utilisent SuiteQL dans des SuiteScripts ou des applications externes pour intégrer les données NetSuite dans des tableaux de bord personnalisés. Par exemple, un SuiteScript peut exécuter une requête SuiteQL et renvoyer du JSON via un SuiteLet pour construire un portlet sur un portail.

À travers ces modèles, la **cohérence des données** est essentielle. La source d'analyse de NetSuite est statique entre les mises à jour ; une synchronisation pendant les heures creuses (par ex. la nuit) permet d'éviter les problèmes de données partielles. Faites également attention aux limites de débit de l'API si vous effectuez des appels SuiteTalk.

Implications en matière de sécurité et de gouvernance

SuiteQL expose des capacités puissantes, la gouvernance est donc essentielle. Les pratiques recommandées incluent :

- **Rôles et autorisations :** Utilisez des rôles à privilèges minimaux pour Connect. N'accordez l'accès qu'aux tables/enregistrements nécessaires. (Le catalogue d'enregistrements indique quelle autorisation contrôle chaque enregistrement sous « Aperçu » – par ex. vous verrez « Permission : Transactions > Invoices ». Si votre rôle en est dépourvu, l'interrogation des lignes `transaction` avec `type='CustInvc'` échouera (Source: www.houseblend.io).)
- **Accès hors bureau :** Si vous donnez accès à une équipe BI, envisagez un rôle machine-à-machine (TBA) plutôt que les identifiants d'un utilisateur RH. Cela évite que les verrouillages d'utilisateurs ou les changements de mot de passe n'interrompent les rapports.
- **Journalisation d'audit :** SuiteQL étant essentiellement un canal d'accès supplémentaire, il doit être audité. Vérifiez la piste d'audit de NetSuite pour vous assurer qu'aucune donnée n'a été modifiée via Connect (ce qui est impossible, mais cela permet d'enregistrer qui a effectué la requête).

- **Sensibilité des données :** Masquez ou excluez les champs sensibles si nécessaire. Par exemple, si des champs personnalisés contiennent des informations personnelles, assurez-vous que seuls les rôles autorisés peuvent les interroger. Il n'existe pas de fonction de masquage intégrée ; effectuez donc la rédaction via la logique de requête ou omettez ces colonnes.

Études de cas et exemples

Pour illustrer l'utilisation pratique de SuiteQL, considérez les scénarios suivants :

- **Tableau de bord de vieillissement client :** Un analyste financier a besoin d'une liste des factures ouvertes et de leurs clients. En utilisant SuiteQL, on pourrait écrire :

  ```
  SELECT c.entityid AS Customer, inv.tranid AS InvoiceNo, inv.duedate, inv.total
  FROM customer c
  JOIN transaction inv ON c.id = inv.entity
  WHERE inv.type = 'CustInvc' AND inv.status = 'Open';
  ```

  Cette requête unique produit un rapport de vieillissement tabulaire. Des jointures supplémentaires (par ex. avec `employees` pour le représentant commercial, ou avec un segment personnalisé pour la région) peuvent enrichir les données. Sans SuiteQL, l'analyste aurait dû exporter des recherches enregistrées et les joindre manuellement dans Excel – un processus lent et sujet aux erreurs.

- **Audit de champ personnalisé :** Un auditeur interne doit vérifier qu'aucun champ inactif n'est utilisé sur les nouvelles transactions. On peut exécuter :

  ```
  SELECT t.tranid, cf.scriptid AS field_id, cf.IsInactive
  FROM transaction t
  CROSS JOIN UNNEST(ARRAY(SELECT *
      FROM UNNEST(t.CustomFields) cf_sub
      WHERE cf_sub.IsInactive = 'T') AS cf
  WHERE t.datecreated > '2026-01-01';
  ```

  (Note : la syntaxe exacte pour aplatir les champs personnalisés dépend de l'API ; alternativement, on peut joindre `transaction` directement à ses colonnes personnalisées si Connect les expose.) Dans tous les cas, SuiteQL permet d'interroger les métadonnées `CustomField` :

  ```
  SELECT cf.ScriptID, cr.Name AS RecordType
  FROM CustomField cf
  JOIN CustomRecordType cr ON cf.RecordType = cr.InternalID
  WHERE cf.IsInactive = 'T';
  ```

  pour trouver tous les champs personnalisés marqués comme inactifs (et sur quel enregistrement ils se trouvent) (Source: www.houseblend.io) (Source: www.houseblend.io).

- **Requête de ciblage marketing :** Un marketeur souhaite obtenir des prospects dans certains codes postaux ayant déjà acheté un article particulier. Cela nécessite de combiner le CRM (adresses des prospects) avec l'ERP (commandes). Une solution consiste à utiliser une `UNION` de deux sous-requêtes (comme détaillé par Tim Dietrich) : interroger d'abord les clients par code postal, puis les clients ayant acheté l'article, et enfin fusionner l'ensemble unique (Source: www.houseblend.io) (Source: www.houseblend.io). En SuiteQL, cela pourrait ressembler à :

```
SELECT DISTINCT c.id, c.entityid
FROM customer c
WHERE c.zip IN ('12345','67890')
UNION
SELECT DISTINCT c2.id, c2.entityid
FROM customer c2
JOIN transaction t2 ON c2.id = t2.entity
JOIN transactionline tl2 ON t2.id = tl2.transaction
WHERE tl2.item = 999;  -- ID interne du produit
```

Le résultat est l'ID et le nom de chaque client répondant à l'un ou l'autre critère, prêt à être intégré dans une campagne ciblée.

- **Intégration BI (Tableau) :** Une équipe de données utilise Coefficient pour connecter NetSuite et Tableau. Ils écrivent des requêtes SuiteQL dans Google Sheets que Coefficient actualise via le pilote NetSuite. Par exemple, pour construire un tableau de bord de KPI financiers, ils peuvent interroger une combinaison de transactions, de filiales et de comptes en une seule fois, et pousser la feuille résultante dans Tableau. Selon l'étude de cas de Coefficient, SuiteQL a géré des agrégats multi-tables que leur ancien connecteur ne pouvait pas traiter (Source: www.houseblend.io), et a exécuté ces requêtes toutes les heures pour un reporting quasi en temps réel.

Ces exemples montrent comment SuiteQL rationalise les flux de travail. Au lieu d'exportations manuelles ou d'intégrations complexes, les analystes peuvent *interroger directement* l'ERP, en combinant les données de manière holistique. Cela réduit les erreurs et accélère la livraison des informations.

Discussion et orientations futures

L'adoption de SuiteQL continue de croître rapidement. NetSuite cite des recherches montrant des gains d'efficacité substantiels (30 à 50 % de travail de reporting en moins) et une demande industrielle (84 % des projets BI impliquent des données d'ERP cloud) (Source: www.houseblend.io) (Source: www.houseblend.io). Comme les champs personnalisés sont très répandus — les enquêtes désignent le « reporting au niveau du champ » comme un point de douleur majeur (Source: www.houseblend.io) — la capacité de SuiteQL à exposer ces métadonnées a été largement saluée par les équipes financières et opérationnelles.

À l'avenir, les développements potentiels incluent :

- **API de métadonnées améliorées :** Bien que SuiteQL couvre les définitions personnalisées, les futures améliorations des métadonnées REST (par ex. des schémas « record/v1 » plus riches) permettront aux développeurs de récupérer les structures d'enregistrement standard à la demande (Source: blogs.oracle.com). Cela pourrait permettre à une requête SuiteQL d'inspecter tout type d'enregistrement, réduisant la dépendance aux inventaires statiques.
- **Optimisations des performances :** À mesure que les volumes de données augmentent (comptes OneWorld, historiques de transactions volumineux), l'optimisation des performances devient critique. Attendez-vous à d'autres améliorations de la part de NetSuite concernant l'optimisation des requêtes et l'indexation en arrière-plan. Les utilisateurs devront peut-être adopter des stratégies telles que la pré-agrégation ou la synthèse dans des entrepôts de données si des jointures extrêmement volumineuses commencent à ralentir le système.
- **Aides à la requête basées sur l'IA :** Le rapport HouseBlend spécule que l'IA pourrait aider à écrire du SuiteQL (via le langage naturel ou l'autocomplétion) à l'avenir (Source: www.houseblend.io). Des signes précoces sont déjà visibles dans certains outils qui analysent les invites textuelles en SQL ; cela pourrait rendre SuiteQL plus accessible aux utilisateurs occasionnels.
- **Modèles d'accès étendus :** Actuellement, SuiteQL est en lecture seule. Il existe des demandes de la communauté pour une écriture en retour (une interface de chargement de données basée sur SQL), mais une telle fonctionnalité nécessiterait une gouvernance rigoureuse. Une autre frontière est l'intégration API en temps réel : coupler SuiteQL avec des webhooks (ou le framework « Driven Events » de NetSuite) pourrait diffuser des données vers les plateformes d'analyse instantanément.

Conclusion

SuiteAnalytics Connect et SuiteQL de NetSuite représentent ensemble une évolution majeure du reporting NetSuite. En traitant l'ERP/CRM comme une base de données SQL vivante, SuiteQL permet aux organisations de récupérer et de fusionner les données de manière flexible et efficace. Cette référence a souligné comment interroger les **champs et enregistrements personnalisés** (via les tables `CustomField` / `CustomRecordType`), comment appliquer des **jointures SQL et des opérations d'ensemble**, et quels **modèles spécifiques à Connect** utiliser pour une intégration robuste. Nous nous sommes appuyés sur la documentation officielle NetSuite d'Oracle (Source: docs.oracle.com) (Source: docs.oracle.com), des blogs d'experts (Source: timdietrich.me) (Source: www.houseblend.io), et des exemples concrets (Source: www.houseblend.io) (Source: www.houseblend.io) pour garantir l'exactitude et fournir des conseils pratiques.

Les preuves suggèrent que SuiteQL raccourcit considérablement les cycles de reporting et ouvre de nouvelles possibilités analytiques. À mesure que les déploiements NetSuite augmentent (plus de 40 000 clients et ce n'est pas fini (Source: www.houseblend.io) et que les demandes BI augmentent, SuiteQL deviendra probablement la méthode standard pour accéder aux données NetSuite. Les organisations qui maîtrisent SuiteQL – en particulier en utilisant ses tables de métadonnées pour les champs personnalisés – obtiendront un avantage stratégique en débloquant les informations cachées dans leurs personnalisations. Nous encourageons les équipes à développer leurs compétences SuiteQL, à documenter leurs schémas Connect et à affiner continuellement leurs requêtes en utilisant les directives présentées ici. Avec une gouvernance de sécurité rigoureuse et un réglage des performances, SuiteQL peut servir de couche d'analyse sécurisée et performante au cœur des entreprises pilotées par NetSuite (Source: www.houseblend.io) (Source: www.houseblend.io).

Tableaux : Nous avons inclus ci-dessus des tableaux résumant les types de jointures et les conventions de nommage. Des références supplémentaires et une bibliographie peuvent être fournies sur demande. Toutes les affirmations et exemples de ce rapport sont étayés par des sources faisant autorité et des experts de la communauté pour garantir fiabilité et profondeur (Source: www.houseblend.io) (Source: www.houseblend.io).

Étiquettes: netsuite-suiteql, suiteanalytics-connect, jointures-sql, champs-personnalises, enregistrements-personnalises-netsuite, oracle-sql, netsuite2com, requetage-base-de-donnees, reporting-erp

---