60 Best Interview Questions - Data Analyst

UniBank Haiti - Commercial Banking Context

Niveau: Intermédiaire à Senior
Format: Question + Réponse complète avec introduction

GLOSSAIRE DES DÉFINITIONS CLÉS

SQL et Bases de Données

Terme	Définition
SQL	Structured Query Language - langage standardisé pour interroger et manipuler des bases de données relationnelles
JOIN	Opération qui combine des lignes de deux ou plusieurs tables basées sur une colonne liée
CTE (Common Table Expression)	Requête nommée temporaire définie dans une clause WITH, utilisable dans la requête principale
Window Function	Fonction qui effectue un calcul sur un ensemble de lignes liées à la ligne courante
Index	Structure de données qui améliore la vitesse des opérations de récupération des données
N+1 Problem	Anti-pattern où une requête initiale génère N requêtes supplémentaires, une par résultat
Agrégation	Opération qui combine plusieurs valeurs en une seule (SUM, AVG, COUNT, etc.)

Power BI et DAX

Terme	Définition
DAX	Data Analysis Expressions - langage de formules pour Power BI, Analysis Services et Power Pivot
Mesure	Calcul dynamique évalué au moment de la requête selon le contexte de filtre
Colonne calculée	Colonne dont les valeurs sont calculées à l'importation et stockées dans le modèle
Filter Context	Ensemble des filtres actifs qui déterminent quelles données sont incluses dans un calcul
Row Context	Contexte qui évalue une expression ligne par ligne dans une table
CALCULATE	Fonction DAX qui évalue une expression dans un contexte de filtre modifié
Time Intelligence	Ensemble de fonctions DAX pour les calculs temporels (YTD, YoY, etc.)

Statistiques

Terme	Définition
Moyenne (Mean)	Somme des valeurs divisée par le nombre d'observations
Médiane	Valeur centrale qui sépare un ensemble de données en deux moitiés égales
Mode	Valeur la plus fréquente dans un ensemble de données
Écart-type	Mesure de la dispersion des données autour de la moyenne
Variance	Carré de l'écart-type, mesure la dispersion des données
Skewness (Asymétrie)	Mesure de l'asymétrie d'une distribution par rapport à sa moyenne
Kurtosis	Mesure de l'épaisseur des queues d'une distribution
p-value	Probabilité d'obtenir un résultat au moins aussi extrême que celui observé, si H_0 est vraie
Corrélation	Mesure statistique de la force et de la direction de la relation entre deux variables
Intervalle de confiance	Plage de valeurs qui contient probablement le paramètre de population réel

KPIs Bancaires

Terme	Définition
NPL (Non-Performing Loans)	Prêts en défaut ou proches du défaut (généralement > 90 jours de retard)
ROE (Return on Equity)	Rendement des capitaux propres - mesure la rentabilité pour les actionnaires
ROA (Return on Assets)	Rendement des actifs - mesure l'efficacité d'utilisation des actifs
NIM (Net Interest Margin)	Marge nette d'intérêt - différence entre revenus et charges d'intérêts
CAR (Capital Adequacy Ratio)	Ratio de solvabilité - fonds propres / actifs pondérés par les risques
LDR (Loan-to-Deposit Ratio)	Ratio prêts/dépôts - mesure la liquidité et l'utilisation des dépôts
CIR (Cost-to-Income Ratio)	Ratio d'efficacité opérationnelle - charges / revenus
AML (Anti-Money Laundering)	Ensemble des procédures pour prévenir le blanchiment d'argent

Data Analysis

Terme	Définition
EDA (Exploratory Data Analysis)	Analyse exploratoire des données - processus d'investigation initiale des données
Outlier	Valeur anormalement éloignée des autres observations
Data Wrangling	Processus de nettoyage, transformation et préparation des données
ETL	Extract, Transform, Load - processus d'intégration de données
DataFrame	Structure de données tabulaire en Pandas avec lignes et colonnes indexées

---

SECTION 1: SQL (15 Questions)

Q1. Quelle est la différence entre WHERE et HAVING?

Réponse:

Cette question teste votre compréhension de l'ordre d'exécution des requêtes SQL. WHERE et HAVING sont tous deux des clauses de filtrage, mais ils s'appliquent à des moments différents dans l'exécution de la requête.

- **WHERE** filtre les lignes **AVANT** l'agrégation (GROUP BY)
- **HAVING** filtre les groupes **APRÈS** l'agrégation

```sql
-- WHERE: filtre les transactions > 1000 avant de grouper
SELECT client_id, SUM(montant)
FROM transactions
WHERE montant > 1000
GROUP BY client_id;

-- HAVING: filtre les clients dont le total > 50000
SELECT client_id, SUM(montant) as total
FROM transactions
GROUP BY client_id
HAVING SUM(montant) > 50000;
```

Q2. Expliquez la différence entre ROW_NUMBER, RANK et DENSE_RANK.

Réponse:

Ces trois fonctions sont des Window Functions qui attribuent un rang aux lignes, mais elles diffèrent dans leur traitement des valeurs égales. Comprendre cette différence est essentiel pour les classements et la pagination.

Fonction	Comportement avec égalité	Exemple
ROW_NUMBER	Toujours unique, arbitraire pour égalités	1, 2, 3, 4, 5
RANK	Saute les rangs après égalité	1, 2, 2, 4, 5
DENSE_RANK	Ne saute pas les rangs	1, 2, 2, 3, 4

```sql
SELECT
    client_id,
    montant,
    ROW_NUMBER() OVER (ORDER BY montant DESC) as row_num,
    RANK() OVER (ORDER BY montant DESC) as rank,
    DENSE_RANK() OVER (ORDER BY montant DESC) as dense_rank
FROM transactions;
```

Q3. Qu'est-ce qu'une CTE et quand l'utiliser?

Réponse:

Une CTE (Common Table Expression) est une requête nommée temporaire qui améliore la lisibilité et la maintenabilité du code SQL. Elle est définie dans une clause WITH et peut être référencée comme une table dans la requête principale.

```sql
WITH clients_actifs AS (
    SELECT client_id, COUNT(*) as nb_tx
    FROM transactions
    WHERE date >= CURRENT_DATE - INTERVAL '30 days'
    GROUP BY client_id
),
clients_vip AS (
    SELECT client_id FROM clients WHERE segment = 'VIP'
)
SELECT c.client_id, ca.nb_tx
FROM clients_vip c
JOIN clients_actifs ca ON c.client_id = ca.client_id;
```

Utiliser quand: requêtes complexes avec sous-requêtes multiples, besoin de réutiliser un résultat, ou pour améliorer la lisibilité.

Q4. Comment calculer un running total (cumul) en SQL?

Réponse:

Le calcul d'un total cumulatif est une opération courante en analyse financière, notamment pour suivre l'évolution des soldes ou des ventes. En SQL moderne, on utilise les Window Functions avec une clause de frame.

```sql
SELECT
    date,
    montant,
    SUM(montant) OVER (ORDER BY date ROWS UNBOUNDED PRECEDING) as cumul
FROM transactions;
```

Explication: `ROWS UNBOUNDED PRECEDING` indique que la somme inclut toutes les lignes depuis le début du résultat jusqu'à la ligne courante.

Q5. Quelle est la différence entre INNER JOIN, LEFT JOIN et FULL OUTER JOIN?

Réponse:

Les différents types de JOIN déterminent comment les lignes des deux tables sont combinées et quelles lignes sont incluses dans le résultat final. Le choix du type de JOIN dépend des besoins métier.

JOIN Type	Résultat
INNER	Seulement les lignes qui ont une correspondance dans les deux tables
LEFT	Toutes les lignes de la table gauche + correspondances de droite (NULL si absent)
RIGHT	Toutes les lignes de la table droite + correspondances de gauche

JOIN Type	Résultat
FULL OUTER	Toutes les lignes des deux tables, avec NULL où il n'y a pas de correspondance

```sql
-- Tous les clients avec leurs transactions (même sans transactions)
SELECT c.nom, t.montant
FROM clients c
LEFT JOIN transactions t ON c.id = t.client_id;
```

Q6. Comment trouver les doublons dans une table?

Réponse:

Identifier les doublons est une étape cruciale du nettoyage des données. Il existe plusieurs méthodes, chacune adaptée à des situations spécifiques.

```sql
-- Méthode 1: GROUP BY + HAVING - identifie les valeurs dupliquées
SELECT email, COUNT(*) as occurrences
FROM clients
GROUP BY email
HAVING COUNT(*) > 1;

-- Méthode 2: Window function - retourne les lignes dupliquées elles-mêmes
SELECT * FROM (
    SELECT *, ROW_NUMBER() OVER (PARTITION BY email ORDER BY id) as rn
    FROM clients
) t WHERE rn > 1;
```

Q7. Comment calculer la variation mois sur mois (MoM)?

Réponse:

Le calcul de la variation période sur période est fondamental en analyse financière pour mesurer la croissance ou la décroissance. La fonction LAG permet d'accéder aux valeurs des lignes précédentes.

```sql
WITH monthly AS (
    SELECT
        DATE_TRUNC('month', date) as mois,
        SUM(montant) as total
    FROM transactions
    GROUP BY DATE_TRUNC('month', date)
)
SELECT
    mois,
    total,
    LAG(total) OVER (ORDER BY mois) as mois_precedent,
    ROUND((total - LAG(total) OVER (ORDER BY mois)) /
        NULLIF(LAG(total) OVER (ORDER BY mois), 0) * 100, 2) as variation_pct
FROM monthly;
```

Q8. Qu'est-ce que le problème N+1 et comment le résoudre?

Réponse:

Le problème N+1 est un anti-pattern de performance critique qui se produit principalement avec les ORM. Il survient quand une requête initiale est suivie de N requêtes supplémentaires pour charger les données liées.

Définition: 1 requête pour la liste principale + N requêtes pour les détails = N+1 total.

```
# MAUVAIS - N+1 (génère 1001 requêtes pour 1000 clients)
clients = session.query(Client).all()
for client in clients:
    print(client.transactions)  # 1 requête par client!

# BON - Eager loading (génère seulement 2 requêtes)
clients = session.query(Client).options(selectinload(Client.transactions)).all()
```

Solutions: JOIN (joinedload), batch loading avec IN (selectinload), ou requête SQL directe avec agrégation.

Q9. Comment optimiser une requête lente?

Réponse:

L'optimisation des requêtes est une compétence essentielle pour tout analyste de données travaillant avec des volumes importants. Voici une approche méthodique en 6 étapes.

1. **EXPLAIN ANALYZE** - Comprendre le plan d'exécution et identifier les goulots d'étranglement
2. **Ajouter des index** - Sur les colonnes utilisées dans WHERE, JOIN et ORDER BY
3. **Sélectionner les colonnes nécessaires** - Éviter SELECT * qui charge toutes les colonnes
4. **Éviter les fonctions dans WHERE** - Elles empêchent l'utilisation des index
5. **Partitionner les grandes tables** - Généralement par date pour les données temporelles
6. **Utiliser EXISTS plutôt que IN** - Plus efficace pour les sous-requêtes volumineuses

---

Q10. Expliquez COALESCE et son utilisation.

Réponse:

COALESCE est une fonction SQL standard qui retourne la première valeur non-NULL parmi ses arguments. Elle est essentielle pour gérer les valeurs manquantes et fournir des valeurs par défaut.

```
-- Retourne le premier numéro de téléphone disponible
SELECT
    client_id,
    COALESCE(telephone_mobile, telephone_fixe, 'N/A') as telephone
FROM clients;

-- Évite les NULL dans les agrégations
SELECT COALESCE(SUM(montant), 0) as total FROM transactions;
```

Q11. Comment faire un pivot en SQL?

Réponse:

Le pivotement transforme les valeurs d'une colonne en colonnes distinctes, permettant une vue matricielle des données. Cette technique est courante pour les rapports financiers et les tableaux croisés.

```sql
-- Transactions par client et type (pivot manuel avec CASE)
SELECT
    client_id,
    SUM(CASE WHEN type = 'DEPOT' THEN montant ELSE 0 END) as depots,
    SUM(CASE WHEN type = 'RETRAIT' THEN montant ELSE 0 END) as retraits,
    SUM(CASE WHEN type = 'TRANSFERT' THEN montant ELSE 0 END) as transferts
FROM transactions
GROUP BY client_id;
```

Q12. Quelle est la différence entre DELETE, TRUNCATE et DROP?

Réponse:

Ces trois commandes suppriment des données, mais avec des implications très différentes en termes de performance, récupération et structure. Le choix dépend du cas d'usage.

Commande	Action	Rollback possible	Clause WHERE	Performance
DELETE	Supprime des lignes	Oui	Oui	Lent (log chaque ligne)
TRUNCATE	Vide la table entière	Non*	Non	Rapide (pas de log par ligne)
DROP	Supprime la table et sa structure	Non	N/A	Instantané

*Note: TRUNCATE est plus rapide car il ne journalise pas chaque suppression individuellement.

Q13. Comment calculer le percentile en SQL?

Réponse:

Les percentiles sont essentiels pour comprendre la distribution des données et identifier les seuils. SQL offre des fonctions dédiées pour ce calcul.

```sql
-- Médiane (50e percentile) et P95 des montants
SELECT
    PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY montant) as mediane,
    PERCENTILE_CONT(0.95) WITHIN GROUP (ORDER BY montant) as p95
FROM transactions;

-- Créer des quartiles avec NTILE
SELECT
```

```
    *,
    NTILE(4) OVER (ORDER BY montant) as quartile
FROM transactions;
```

Q14. Comment gérer les NULL dans les comparaisons?

Réponse:

NULL représente une valeur inconnue en SQL et ne peut pas être comparé avec les opérateurs standards. Cette particularité est source de nombreuses erreurs pour les débutants.

```
-- NE FONCTIONNE PAS - retourne toujours faux
SELECT * FROM clients WHERE telephone = NULL;

-- CORRECT - utiliser IS NULL / IS NOT NULL
SELECT * FROM clients WHERE telephone IS NULL;
SELECT * FROM clients WHERE telephone IS NOT NULL;

-- Comparaison sécurisée avec COALESCE
SELECT * FROM clients
WHERE COALESCE(telephone, '') = COALESCE(@param, '');
```

Q15. Écrivez une requête pour trouver les clients inactifs depuis 90 jours.

Réponse:

Identifier les clients inactifs est crucial pour les stratégies de rétention. Cette requête illustre l'utilisation de LEFT JOIN pour trouver les absences de correspondance.

```
-- Méthode 1: LEFT JOIN avec condition de date
SELECT c.*
FROM clients c
LEFT JOIN transactions t ON c.id = t.client_id
    AND t.date >= CURRENT_DATE - INTERVAL '90 days'
WHERE t.id IS NULL;

-- Méthode 2: NOT EXISTS (souvent plus performant)
SELECT c.*
FROM clients c
WHERE NOT EXISTS (
    SELECT 1 FROM transactions t
    WHERE t.client_id = c.id
    AND t.date >= CURRENT_DATE - INTERVAL '90 days'
);
```

SECTION 2: Power BI / DAX (12 Questions)

Q16. Quelle est la différence entre une mesure et une colonne calculée?

Réponse:

Cette distinction fondamentale en DAX détermine quand et comment un calcul est effectué. Le mauvais choix peut impacter significativement les performances et les résultats.

Aspect	Colonne Calculée	Mesure
Moment du calcul	À l'importation des données	À chaque requête/interaction
Stockage	Oui, dans le modèle (augmente la taille)	Non, calculée à la volée
Contexte	Row context (accès aux valeurs de la ligne)	Filter context (agrégation dynamique)
Usage typique	Catégorisation, calculs fixes par ligne	KPIs, totaux, pourcentages dynamiques

```
// Colonne calculée - valeur fixe par ligne
Profit = Ventes[Revenus] - Ventes[Couts]

// Mesure - s'adapte aux filtres visuels
Total Ventes = SUM(Ventes[Montant])
```

Q17. Expliquez CALCULATE et son importance.

Réponse:

CALCULATE est la fonction la plus importante et la plus puissante de DAX. Elle permet de modifier le contexte de filtre dans lequel une expression est évaluée, ce qui est essentiel pour créer des calculs conditionnels.

Définition: CALCULATE évalue une expression dans un contexte de filtre modifié par les arguments de filtre spécifiés.

```
// Ventes totales (sans modification de contexte)
Total = SUM(Ventes[Montant])

// Ventes VIP seulement - ajoute un filtre
Ventes VIP = CALCULATE(SUM(Ventes[Montant]), Clients[Segment] = "VIP")

// Ventes année précédente - modifie le contexte de date
Ventes LY = CALCULATE(SUM(Ventes[Montant]), SAMEPERIODLASTYEAR(Calendrier[Date]))
```

Q18. Comment calculer le pourcentage du total en DAX?

Réponse:

Le calcul du pourcentage du total est un pattern DAX fondamental qui nécessite de comprendre comment supprimer les filtres avec ALL pour obtenir le total global.

```
// Pourcentage du total global
% du Total =
DIVIDE(
    SUM(Ventes[Montant]),
```

```
    CALCULATE(SUM(Ventes[Montant]), ALL(Ventes)),
    0
)

// Pourcentage au sein d'une catégorie (garde le filtre catégorie)
% Catégorie =
DIVIDE(
    SUM(Ventes[Montant]),
    CALCULATE(SUM(Ventes[Montant]), ALLEXCEPT(Ventes, Ventes[Categorie]))
)
```

Q19. Quelle est la différence entre ALL, ALLEXCEPT et ALLSELECTED?

Réponse:

Ces trois fonctions modifient le contexte de filtre de manières différentes. Comprendre leurs nuances est crucial pour créer des calculs de pourcentage et de comparaison corrects.

Fonction	Effet sur les filtres
ALL	Supprime TOUS les filtres de la table ou colonne spécifiée
ALLEXCEPT	Supprime tous les filtres SAUF ceux des colonnes spécifiées
ALLSELECTED	Supprime les filtres visuels locaux, mais garde les filtres de slicers et de page

```
// Ignore tous les filtres - total absolu
Total Global = CALCULATE(SUM(Montant), ALL(Ventes))

// Garde le filtre sur Région - total par région
Total Région = CALCULATE(SUM(Montant), ALLEXCEPT(Ventes, Ventes[Region]))
```

Q20. Comment créer une mesure Year-to-Date (YTD)?

Réponse:

Les calculs Year-to-Date sont essentiels pour le reporting financier. DAX offre des fonctions dédiées de Time Intelligence qui simplifient ces calculs.

```
// Méthode 1: TOTALYTD - la plus simple
Ventes YTD = TOTALYTD(SUM(Ventes[Montant]), Calendrier[Date])

// Méthode 2: CALCULATE + DATESYTD - plus flexible
Ventes YTD v2 =
CALCULATE(
    SUM(Ventes[Montant]),
    DATESYTD(Calendrier[Date])
)

// YTD avec année fiscale personnalisée (fin en juin)
Ventes YTD Fiscal = TOTALYTD(SUM(Ventes[Montant]), Calendrier[Date], "06-30")
```

Q21. Comment comparer avec l'année précédente (YoY)?

Réponse:

La comparaison Year-over-Year est un indicateur clé de croissance. SAMEPERIODLASTYEAR est la fonction de base pour ce type de calcul en DAX.

```
// Valeur de l'année précédente
Ventes LY =
CALCULATE(
    SUM(Ventes[Montant]),
    SAMEPERIODLASTYEAR(Calendrier[Date])
)

// Variation en pourcentage
Variation YoY =
VAR CurrentYear = SUM(Ventes[Montant])
VAR LastYear = [Ventes LY]
RETURN
DIVIDE(CurrentYear - LastYear, LastYear, 0)

// Formatage pour affichage
Variation YoY % = FORMAT([Variation YoY], "0.0%")
```

Q22. Qu'est-ce que le row context vs filter context?

Réponse:

La compréhension des contextes est fondamentale en DAX. Ces deux types de contextes déterminent comment les expressions sont évaluées et quelles données sont accessibles.

Context	Où il existe	Comportement
Row Context	Colonnes calculées, itérateurs (SUMX, AVERAGEX)	Évalue l'expression pour chaque ligne individuellement
Filter Context	Mesures, CALCULATE	Définit quelles lignes sont incluses avant le calcul

```
// Row context - calculé pour chaque ligne de la table
Marge = Ventes[Prix] - Ventes[Cout]

// Filter context - agrège toutes les lignes visibles selon les filtres
Total Marge = SUM(Ventes[Marge])
```

Q23. Comment créer un classement (ranking) en DAX?

Réponse:

RANKX est la fonction DAX pour créer des classements dynamiques. Elle est particulièrement utile pour identifier les top/bottom performers.

```
// Rang des clients par ventes (1 = meilleur)
Rang Ventes =
```

```
RANKX(
    ALL(Clients),      // Table sur laquelle ranger
    [Total Ventes],    // Expression de classement
    ,                  // Valeur à comparer (optionnel)
    DESC,              // Ordre décroissant
    DENSE              // Type de rang (pas de trous)
)

// Flag pour identifier le Top 10
Top 10 Flag = IF([Rang Ventes] <= 10, 1, 0)
```

Q24. Qu'est-ce que FILTER et quand l'utiliser?

Réponse:

FILTER est une fonction de table qui retourne un sous-ensemble filtré d'une table. Elle est nécessaire quand les conditions de filtrage simples ne suffisent pas.

```
// Filtre simple (préféré pour les cas simples - plus performant)
Ventes VIP = CALCULATE(SUM(Montant), Clients[Segment] = "VIP")

// FILTER pour conditions complexes basées sur des mesures
Gros Clients =
CALCULATE(
    SUM(Montant),
    FILTER(
        Clients,
        CALCULATE(SUM(Ventes[Montant])) > 100000
    )
)
```

Règle: Utiliser FILTER uniquement quand la condition dépend d'un calcul (mesure), pas pour les filtres simples sur des colonnes.

Q25. Comment utiliser les variables (VAR/RETURN)?

Réponse:

Les variables DAX améliorent la lisibilité et la performance en stockant des valeurs intermédiaires qui ne sont calculées qu'une seule fois.

```
// Sans variables - calculs répétés, difficile à lire
Marge % = DIVIDE(SUM(Ventes[Montant]) - SUM(Ventes[Cout]), SUM(Ventes[Montant]), 0)

// Avec variables - plus clair et plus performant
Marge Nette % =
VAR Revenus = SUM(Ventes[Montant])
VAR Couts = SUM(Ventes[Cout])
VAR Marge = Revenus - Couts
RETURN
DIVIDE(Marge, Revenus, 0)
```

Q26. Comment gérer les relations many-to-many?

Réponse:

Les relations many-to-many (plusieurs-à-plusieurs) sont complexes en Power BI car une valeur peut correspondre à plusieurs enregistrements des deux côtés. Il existe plusieurs stratégies.

1. **Table de pont (bridge table)** - Créer une table intermédiaire avec les paires de correspondances
2. **CROSSFILTER** - Activer temporairement le filtrage bidirectionnel
3. **TREATAS** - Créer une relation virtuelle sans modifier le modèle

```
// Utilisation de TREATAS pour simuler une relation
Ventes par Tag =
CALCULATE(
    SUM(Ventes[Montant]),
    TREATAS(VALUES(Tags[TagID]), Produits[TagID])
)
```

Q27. Quelle est la différence entre Import, DirectQuery et Live Connection?

Réponse:

Le mode de connexion aux données détermine où les données sont stockées et quand elles sont rafraîchies. Le choix impacte les performances et les fonctionnalités disponibles.

Mode	Où sont les données	Rafraîchissement	Performance	Fonctionnalités DAX
Import	Copiées dans le modèle Power BI	Planifié (manuel ou automatique)	Excellente	Complètes
DirectQuery	Restent à la source	Temps réel à chaque interaction	Dépend de la source	Limitées
Live Connection	SSAS ou datasets Power BI	Temps réel	Dépend de la source	Définies à la source

Recommandation: Import par défaut pour les meilleures performances. DirectQuery pour les données volumineuses (>1GB) ou le temps réel requis.

SECTION 3: Statistiques (10 Questions)

Q28. Quelle est la différence entre moyenne et médiane? Quand utiliser chacune?

Réponse:

La moyenne et la médiane sont deux mesures de tendance centrale qui résument différemment la position centrale des données. Le choix entre les deux dépend de la distribution des données.

Mesure	Calcul	Sensibilité aux outliers	Quand utiliser
Moyenne	Somme / N	Très sensible	Distributions symétriques
Médiane	Valeur centrale	Robuste	Distributions asymétriques, présence d'outliers

Exemple concret - Revenus: [30K, 35K, 40K, 45K, 500K] - Moyenne = 130K (fortement influencée par le 500K) - Médiane = 40K (représente mieux la situation typique)

Règle pratique: Utiliser la médiane pour les revenus, prix immobiliers, et toute variable avec des valeurs extrêmes.

Q29. Qu'est-ce que le skewness et comment l'interpréter?

Réponse:

Le skewness (asymétrie) mesure la déformation d'une distribution par rapport à une distribution symétrique. Il indique si les données sont concentrées d'un côté avec une queue étendue de l'autre.

Valeur	Interprétation	Relation entre mesures centrales
= 0	Distribution symétrique	Moyenne = Médiane = Mode
> 0	Asymétrie positive (queue à droite)	Moyenne > Médiane > Mode
< 0	Asymétrie négative (queue à gauche)	Moyenne < Médiane < Mode

Exemple bancaire: La distribution des soldes de compte a généralement un skewness > 0, car il y a beaucoup de petits soldes et quelques très gros comptes.

Q30. Expliquez la p-value et comment l'interpréter.

Réponse:

La p-value est un concept fondamental en statistique inférentielle. Elle quantifie la force de l'évidence contre l'hypothèse nulle dans un test d'hypothèse.

Définition: La p-value est la probabilité d'observer un résultat au moins aussi extrême que celui obtenu, en supposant que l'hypothèse nulle (H_0) est vraie.

p-value	Interprétation	Décision
< 0.01	Très forte évidence contre H_0	Rejeter H_0
< 0.05	Forte évidence contre H_0	Rejeter H_0 (seuil standard)
< 0.10	Évidence modérée	Marginalement significatif
≥ 0.10	Évidence faible	Ne pas rejeter H_0

Attention importante: Significatif statistiquement ne signifie pas important pratiquement. Une différence de 0.1% peut être significative avec un grand échantillon mais sans importance business.

Q31. Quelle est la différence entre corrélation et causalité?

Réponse:

Cette distinction est cruciale pour éviter de tirer des conclusions erronées à partir des données. La corrélation est une observation, la causalité est un mécanisme.

- **Corrélation:** Deux variables varient ensemble de manière prévisible
- **Causalité:** Une variable provoque directement le changement de l'autre

Exemple classique: - Observation: Les ventes de glaces et les noyades sont corrélées positivement - Fausse conclusion: Les glaces causent les noyades - Réalité: La chaleur (variable confondante) augmente les deux

Pour établir la causalité: Expérience contrôlée randomisée, analyse temporelle (la cause précède l'effet), élimination des variables confondantes.

Q32. Comment détecter les outliers?

Réponse:

Les outliers sont des observations qui s'écartent significativement du reste des données. Les détecter est essentiel avant toute analyse car ils peuvent fausser les résultats.

Méthode IQR (Interquartile Range):

```
Q1 = 25e percentile
Q3 = 75e percentile
IQR = Q3 - Q1

Outlier si: x < Q1 - 1.5×IQR  ou  x > Q3 + 1.5×IQR
```

Méthode Z-score:

```
z = (x - moyenne) / écart-type
Outlier si: |z| > 3 (plus de 3 écarts-types de la moyenne)
```

```python
# Implémentation Python
Q1, Q3 = df['montant'].quantile([0.25, 0.75])
IQR = Q3 - Q1
outliers = df[(df['montant'] < Q1 - 1.5*IQR) | (df['montant'] > Q3 + 1.5*IQR)]
```

Q33. Qu'est-ce qu'un intervalle de confiance à 95%?

Réponse:

L'intervalle de confiance quantifie l'incertitude autour d'une estimation statistique. Il fournit une plage de valeurs plausibles pour le paramètre réel de la population.

Interprétation: Si on répétait l'échantillonnage 100 fois, 95 des intervalles calculés contiendraient la vraie valeur de la population.

Formule pour la moyenne:

```
IC = x̄ ± z × (s/√n)
IC 95% = x̄ ± 1.96 × (s/√n)
```

Exemple bancaire: Montant moyen des transactions = 500 HTG avec IC 95% = [480, 520]. On est 95% confiant que le vrai montant moyen de la population se situe entre 480 et 520 HTG.

Q34. Quel test statistique utiliser pour comparer deux moyennes?

Réponse:

Le choix du test dépend des caractéristiques des données (normalité, indépendance) et du type de comparaison souhaitée. Voici un guide décisionnel.

Situation	Test recommandé
2 groupes indépendants, données normales	t-test pour échantillons indépendants
2 groupes indépendants, données non-normales	Mann-Whitney U (test non-paramétrique)
2 mesures sur les mêmes sujets (avant/après)	t-test apparié
3+ groupes indépendants, données normales	ANOVA
3+ groupes indépendants, données non-normales	Kruskal-Wallis

Exemple bancaire: Comparer le montant moyen des prêts entre l'agence A et l'agence B → t-test pour échantillons indépendants (si normalité vérifiée).

Q35. Qu'est-ce que le coefficient de variation (CV)?

Réponse:

Le coefficient de variation est une mesure de dispersion relative qui permet de comparer la variabilité de variables ayant des unités ou des échelles différentes.

Formule: CV = (Écart-type / Moyenne) × 100

Avantage: Exprimé en pourcentage, il est indépendant de l'unité de mesure.

Exemple: - Revenus: moyenne = 50,000 HTG, écart-type = 10,000 HTG → CV = 20% - Âge: moyenne = 40 ans, écart-type = 8 ans → CV = 20%

Les deux variables ont la même dispersion relative (20%) malgré des unités et des échelles complètement différentes.

Q36. Comment interpréter un coefficient de corrélation (r)?

Réponse:

Le coefficient de corrélation de Pearson (r) mesure la force et la direction de la relation linéaire entre deux variables quantitatives. Sa valeur est toujours comprise entre -1 et +1.

Valeur de	r
0.9 à 1.0	Très forte
0.7 à 0.9	Forte
0.5 à 0.7	Modérée
0.3 à 0.5	Faible
0 à 0.3	Très faible ou inexistante

Le signe indique la direction: Positif (+) = les deux variables augmentent ensemble. Négatif (-) = quand une augmente, l'autre diminue.

R^2 (R-squared): Le carré de r représente le pourcentage de variance expliquée. Si r = 0.7, alors $r^2 = 0.49$, signifiant que 49% de la variance de Y est expliquée par X.

Q37. Qu'est-ce que l'erreur de Type I et Type II?

Réponse:

Ces deux types d'erreurs représentent les deux façons dont un test d'hypothèse peut se tromper. Comprendre ce compromis est essentiel pour interpréter les résultats de tests.

Erreur	Description	Analogie	Conséquence
Type I (α)	Rejeter H_0 alors qu'elle est vraie	Faux positif	Déclarer une fraude inexistante
Type II (β)	Ne pas rejeter H_0 alors qu'elle est fausse	Faux négatif	Manquer une vraie fraude

Relation importante: Il existe un compromis entre les deux types d'erreurs. Réduire le risque de Type I augmente le risque de Type II, et vice versa. Le choix du seuil α (souvent 0.05) reflète ce compromis.

SECTION 4: KPIs Bancaires (8 Questions)

Q38. Qu'est-ce que le NPL Ratio et pourquoi est-il important?

Réponse:

Le NPL Ratio (Non-Performing Loans Ratio) est l'indicateur clé de la qualité du portefeuille de crédit d'une banque. Il mesure la proportion des prêts en difficulté par rapport au total des prêts accordés.

Définition: NPL Ratio = Prêts non-performants / Total des prêts × 100

Qu'est-ce qu'un prêt non-performant? Un prêt est classé NPL quand le paiement est en retard de plus de 90 jours ou quand il est peu probable que l'emprunteur rembourse intégralement.

Valeur	Interprétation
< 3%	Excellent - Portefeuille de très bonne qualité
3-5%	Acceptable - Niveau de risque gérable

Valeur	Interprétation
5-10%	Préoccupant - Nécessite une attention particulière
> 10%	Critique - Risque significatif pour la banque

Importance: Un NPL élevé signale des problèmes de qualité de crédit, peut nécessiter des provisions importantes, et impacte directement la rentabilité.

Q39. Expliquez ROE, ROA et leur différence.

Réponse:

Le ROE et le ROA sont deux indicateurs fondamentaux de rentabilité bancaire qui mesurent différents aspects de la performance financière.

Indicateur	Formule	Ce qu'il mesure	Valeur typique
ROE	Résultat Net / Capitaux Propres	Rentabilité pour les actionnaires	12-18%
ROA	Résultat Net / Total Actifs	Efficacité d'utilisation des actifs	1-2%

Relation clé: ROE = ROA × Levier financier (Actifs / Capitaux Propres)

Interprétation: Un ROE élevé peut résulter soit d'un bon ROA (bonne gestion opérationnelle), soit d'un fort levier financier (beaucoup de dettes). Il est important d'analyser les deux pour comprendre la source de la performance.

Q40. Qu'est-ce que le NIM (Net Interest Margin)?

Réponse:

Le NIM (Net Interest Margin) est l'indicateur principal de la rentabilité des activités d'intermédiation bancaire. Il mesure l'écart entre ce que la banque gagne sur ses prêts et ce qu'elle paie sur ses dépôts.

Définition: NIM = (Revenus d'intérêts - Charges d'intérêts) / Actifs productifs moyens × 100

Valeur typique: 3-5% pour les banques commerciales

Interprétation: - NIM élevé = Bonne marge sur les activités de prêt - NIM en baisse = Compression des marges (concurrence accrue, environnement de taux bas) - NIM stable = Bonne gestion de la tarification des produits

---

Q41. Qu'est-ce que le ratio LDR (Loan-to-Deposit)?

Réponse:

Le LDR (Loan-to-Deposit Ratio) mesure la proportion des dépôts clients utilisée pour financer les prêts. C'est un indicateur clé de liquidité et d'efficacité de l'utilisation des ressources.

Définition: LDR = Total des prêts / Total des dépôts × 100

Valeur	Interprétation
< 70%	Sous-utilisation des dépôts - La banque pourrait prêter plus
70-90%	Optimal - Bon équilibre entre liquidité et rentabilité
90-100%	Agressif - Utilisation maximale, moins de marge de sécurité
> 100%	Risqué - La banque dépend de financements externes

Q42. Qu'est-ce que le CAR (Capital Adequacy Ratio)?

Réponse:

Le CAR (Capital Adequacy Ratio) est le ratio réglementaire principal qui mesure la capacité d'une banque à absorber des pertes et à protéger les déposants. Il est étroitement surveillé par les régulateurs.

Définition: CAR = Fonds propres réglementaires / Actifs pondérés par les risques (RWA) × 100

Seuils réglementaires: - Minimum BRH (Haïti): 12% - Minimum Bâle III international: 8% - Coussin de conservation: +2.5%

Importance: Un CAR insuffisant peut entraîner des restrictions sur les activités de la banque, voire une intervention du régulateur. C'est un indicateur de solvabilité et de stabilité financière.

Q43. Comment calculer l'Expected Loss (EL)?

Réponse:

L'Expected Loss (Perte Attendue) est la perte moyenne anticipée sur un portefeuille de crédit. C'est un concept fondamental en gestion des risques de crédit.

Formule: EL = PD × LGD × EAD

Où: - **PD (Probability of Default):** Probabilité que l'emprunteur fasse défaut sur une période donnée - **LGD (Loss Given Default):** Pourcentage du montant exposé perdu en cas de défaut (1 - taux de récupération) - **EAD (Exposure at Default):** Montant total exposé au moment du défaut

Exemple de calcul: - PD = 5% (probabilité de défaut) - LGD = 40% (la banque récupère 60% en cas de défaut) - EAD = 100,000 HTG - EL = 0.05 × 0.40 × 100,000 = **2,000 HTG**

Q44. Qu'est-ce que le CIR (Cost-to-Income Ratio)?

Réponse:

Le CIR (Cost-to-Income Ratio) mesure l'efficacité opérationnelle d'une banque en comparant ses charges d'exploitation à ses revenus. C'est un indicateur clé de productivité.

Définition: CIR = Charges d'exploitation / Produit Net Bancaire (PNB) × 100

Valeur	Interprétation
< 45%	Excellent - Banque très efficace
45-55%	Bon - Efficacité dans la norme
55-65%	Moyen - Potentiel d'amélioration
> 65%	Inefficace - Charges trop élevées par rapport aux revenus

Levier d'amélioration: Digitalisation des processus, optimisation des agences, automatisation.

Q45. Quels sont les red flags AML (Anti-Money Laundering)?

Réponse:

L'AML (Anti-Money Laundering) vise à détecter et prévenir le blanchiment d'argent. En tant que Data Analyst, vous devez connaître les signaux d'alerte qui peuvent indiquer des activités suspectes.

Principaux red flags à surveiller:

1. **Structuration (Smurfing):** Transactions délibérément fractionnées juste sous le seuil de déclaration (ex: plusieurs dépôts de 9,999 USD au lieu d'un seul de 30,000 USD)
2. **Activité anormale:** Volume de transactions » moyenne historique du client sans justification économique
3. **Pays à risque:** Transactions avec des juridictions à haut risque (liste GAFI)
4. **Cash intensif:** Dépôts ou retraits cash fréquents et importants sans activité économique justifiant ce besoin
5. **Shell companies:** Entreprises sans activité économique visible mais avec des flux financiers importants
6. **Montants ronds répétitifs:** Transferts de montants exactement identiques de manière répétitive (ex: exactement 10,000 USD chaque semaine)

SECTION 5: Python / Pandas (8 Questions)

Q46. Comment gérer les valeurs manquantes en Pandas?

Réponse:

La gestion des valeurs manquantes est une étape cruciale du nettoyage des données. Pandas offre plusieurs méthodes pour identifier et traiter ces valeurs.

```
# ÉTAPE 1: Identifier les valeurs manquantes
df.isnull().sum()          # Compte par colonne
df.info()                  # Vue d'ensemble avec types et non-null count
df.isnull().mean() * 100   # Pourcentage de valeurs manquantes

# ÉTAPE 2: Traiter - Option A: Supprimer
df.dropna()                          # Supprime toutes les lignes avec au moins un NA
df.dropna(subset=['col1', 'col2'])   # Supprime si NA dans ces colonnes spécifiques
df.dropna(thresh=3)                  # Garde les lignes avec au moins 3 valeurs non-NA
```

```
# ÉTAPE 2: Traiter - Option B: Imputer
df['col'].fillna(df['col'].median())    # Médiane (recommandé pour données numériques asymétriques)
df['col'].fillna(df['col'].mean())      # Moyenne (pour données symétriques)
df['col'].fillna(df['col'].mode()[0])   # Mode (pour données catégorielles)
df['col'].fillna(method='ffill')        # Forward fill (pour séries temporelles)
```

Q47. Comment faire une agrégation groupée en Pandas?

Réponse:

L'agrégation groupée (groupby) est l'équivalent Pandas du GROUP BY en SQL. Elle permet de calculer des statistiques par catégorie.

```
# Agrégation simple - une mesure
df.groupby('region')['montant'].sum()

# Agrégations multiples sur une colonne
df.groupby('region')['montant'].agg(['sum', 'mean', 'count', 'std'])

# Agrégations différentes par colonne
df.groupby('region').agg({
    'montant': ['sum', 'mean'],
    'client_id': 'nunique',      # Nombre de clients uniques
    'date': ['min', 'max']       # Période d'activité
})

# Avec reset_index pour obtenir un DataFrame propre
result = df.groupby('region')['montant'].sum().reset_index()
result.columns = ['region', 'total_montant']
```

Q48. Comment créer une colonne conditionnelle?

Réponse:

Créer des colonnes conditionnelles permet de catégoriser ou transformer les données selon des règles métier. Pandas et NumPy offrent plusieurs approches.

```
import numpy as np

# Méthode 1: np.where - pour conditions binaires (2 valeurs possibles)
df['flag'] = np.where(df['montant'] > 1000, 'High', 'Low')

# Méthode 2: np.select - pour conditions multiples (3+ valeurs possibles)
conditions = [
    df['montant'] < 100,
    df['montant'] < 1000,
    df['montant'] >= 1000
]
choices = ['Low', 'Medium', 'High']
df['category'] = np.select(conditions, choices, default='Unknown')

# Méthode 3: apply avec fonction lambda (plus flexible, mais plus lent)
```

```python
df['category'] = df['montant'].apply(lambda x: 'High' if x > 1000 else 'Low')

# Méthode 4: pd.cut pour créer des bins
df['bin'] = pd.cut(df['montant'], bins=[0, 100, 1000, np.inf], labels=['Low', 'Medium', 'High'])
```

Q49. Comment fusionner deux DataFrames?

Réponse:

La fusion de DataFrames est l'équivalent Pandas des JOIN SQL. C'est une opération fondamentale pour combiner des données de différentes sources.

```python
# Merge (équivalent JOIN SQL) - basé sur des colonnes communes
df = pd.merge(clients, transactions, on='client_id', how='left')

# Merge avec colonnes de noms différents
df = pd.merge(df1, df2, left_on='id', right_on='client_id', how='inner')

# Types de merge (how):
# 'inner' = seulement les correspondances
# 'left' = tout de gauche + correspondances de droite
# 'right' = tout de droite + correspondances de gauche
# 'outer' = tout des deux côtés

# Concat - empiler des DataFrames
df = pd.concat([df1, df2], axis=0)   # Verticalement (ajouter des lignes)
df = pd.concat([df1, df2], axis=1)   # Horizontalement (ajouter des colonnes)
```

Q50. Comment créer un pivot table?

Réponse:

Les pivot tables restructurent les données pour faciliter l'analyse. Elles transforment les valeurs uniques d'une colonne en nouvelles colonnes.

```python
# Pivot table avec agrégation
pivot = df.pivot_table(
    values='montant',           # Colonne à agréger
    index='region',             # Lignes
    columns='type',             # Colonnes
    aggfunc='sum',              # Fonction d'agrégation
    fill_value=0,               # Remplacer NA par 0
    margins=True                # Ajouter les totaux
)

# Crosstab - pour les comptages et pourcentages
pd.crosstab(df['region'], df['type'])                    # Comptages
pd.crosstab(df['region'], df['type'], normalize='index') # % par ligne
pd.crosstab(df['region'], df['type'], normalize='columns') # % par colonne
pd.crosstab(df['region'], df['type'], normalize='all')   # % du total
```

Q51. Comment optimiser la mémoire d'un DataFrame?

Réponse:

L'optimisation mémoire est cruciale pour travailler avec de grands datasets. Pandas utilise par défaut des types de données généreux qui peuvent être réduits.

```
# Vérifier l'usage mémoire actuel
df.info(memory_usage='deep')
print(f"Mémoire: {df.memory_usage(deep=True).sum() / 1024**2:.2f} MB")

# Optimiser les types numériques
df['col_int'] = df['col_int'].astype('int32')      # int64 → int32 (divise par 2)
df['col_int'] = df['col_int'].astype('int16')      # Encore plus petit si valeurs < 32767
df['col_float'] = pd.to_numeric(df['col_float'], downcast='float')

# Convertir les colonnes texte à faible cardinalité en catégories
df['col_cat'] = df['col_cat'].astype('category')   # Très efficace si peu de valeurs uniques

# Lire seulement les colonnes nécessaires
df = pd.read_csv('file.csv', usecols=['col1', 'col2', 'col3'])

# Lire en chunks pour fichiers très volumineux
for chunk in pd.read_csv('big_file.csv', chunksize=10000):
    process(chunk)
```

Q52. Comment calculer une moyenne mobile?

Réponse:

Les moyennes mobiles lissent les données temporelles pour révéler les tendances sous-jacentes. Elles sont essentielles en analyse financière et de séries temporelles.

```
# Moyenne mobile simple (SMA - Simple Moving Average)
df['MA_7'] = df['montant'].rolling(window=7).mean()      # Fenêtre de 7 périodes
df['MA_30'] = df['montant'].rolling(window=30).mean()    # Fenêtre de 30 périodes

# Moyenne mobile exponentielle (EMA - donne plus de poids aux valeurs récentes)
df['EMA_7'] = df['montant'].ewm(span=7).mean()

# Moyenne mobile par groupe (ex: par client)
df['MA_7_client'] = df.groupby('client_id')['montant'].transform(
    lambda x: x.rolling(7, min_periods=1).mean()
)

# Somme mobile (pour volumes cumulés sur une période)
df['sum_30d'] = df['montant'].rolling(window=30).sum()
```

Q53. Comment travailler avec les dates?

Réponse:

La manipulation des dates est omniprésente en analyse de données, surtout pour les données bancaires. Pandas offre des outils puissants via l'accessor `.dt`.

```python
# Conversion en datetime
df['date'] = pd.to_datetime(df['date'])
df['date'] = pd.to_datetime(df['date'], format='%Y-%m-%d')  # Format spécifique

# Extractions de composantes
df['year'] = df['date'].dt.year
df['month'] = df['date'].dt.month
df['day'] = df['date'].dt.day
df['day_of_week'] = df['date'].dt.dayofweek      # 0=Lundi, 6=Dimanche
df['week'] = df['date'].dt.isocalendar().week
df['quarter'] = df['date'].dt.quarter
df['is_weekend'] = df['date'].dt.dayofweek >= 5

# Filtrage par date
df[df['date'] >= '2024-01-01']
df[df['date'].between('2024-01-01', '2024-03-31')]

# Calcul de différences
df['days_since'] = (pd.Timestamp.now() - df['date']).dt.days
df['days_between'] = (df['date2'] - df['date1']).dt.days

# Resampling (agrégation temporelle)
df.set_index('date').resample('M')['montant'].sum()   # Par mois
df.set_index('date').resample('W')['montant'].mean()  # Par semaine
```

SECTION 6: Data Visualization & EDA (7 Questions)

Q54. Quel graphique utiliser pour quelle situation?

Réponse:

Le choix du bon graphique est essentiel pour communiquer efficacement vos insights. Chaque type de graphique est optimisé pour un type de message spécifique.

Objectif	Graphique recommandé	Quand éviter
Distribution d'une variable continue	Histogramme, KDE	Si peu de données (<30)
Comparer des catégories	Bar chart (horizontal si noms longs)	Si trop de catégories (>10)
Évolution temporelle	Line chart	Si données non ordonnées
Relation entre 2 variables	Scatter plot	Si variables catégorielles
Composition d'un tout	Pie chart (< 5 parts), Stacked bar	Si > 5-6 catégories
Comparer distributions, détecter outliers	Box plot	Si audience non-technique
Visualiser corrélations multiples	Heatmap	Si pas de pattern clair

Règle d'or: Un graphique = Un message clair. Ne pas surcharger d'informations.

Q55. Quelles sont les étapes d'une EDA?

Réponse:

L'EDA (Exploratory Data Analysis) est un processus systématique d'investigation des données avant toute analyse approfondie. Voici un framework en 7 étapes.

```
1. COMPRENDRE LE CONTEXTE BUSINESS
   → Quel problème essayons-nous de résoudre?
   → Quelles décisions seront prises avec ces données?

2. EXPLORER LA STRUCTURE
   → df.shape, df.columns, df.dtypes
   → df.head(), df.tail(), df.sample(10)

3. STATISTIQUES DESCRIPTIVES
   → df.describe() pour variables numériques
   → df.describe(include='object') pour catégorielles
   → Distribution de chaque variable clé

4. QUALITÉ DES DONNÉES
   → Valeurs manquantes: df.isnull().sum()
   → Doublons: df.duplicated().sum()
   → Outliers: IQR ou Z-score

5. RELATIONS ENTRE VARIABLES
   → Corrélations: df.corr()
   → Scatter plots pour paires intéressantes
   → Analyses bivariées

6. SEGMENTATION ET PATTERNS
   → Patterns par groupe (région, segment client)
   → Tendances temporelles
   → Anomalies

7. DOCUMENTER
   → Insights clés découverts
   → Questions pour la suite
   → Prochaines étapes recommandées
```

Q56. Comment présenter des données à des non-techniciens?

Réponse:

Communiquer des insights à un public non-technique est une compétence essentielle. Le but est de transmettre le message sans perdre l'audience dans les détails techniques.

Principes clés:

1. **Commencer par le "So What?"** - Présentez l'insight principal et son impact business d'abord, pas la méthodologie

2. **Simplifier** - Évitez le jargon statistique. Dites “la moitié des clients” plutôt que “la médiane”
3. **Utiliser des comparaisons concrètes** - “20% de plus que l’an dernier” est plus parlant que “un ratio de 1.2”
4. **Visualiser** - Un bon graphique remplace souvent des pages d’explication
5. **Contextualiser** - Expliquez si c’est bon ou mauvais et pourquoi
6. **Recommander des actions** - Terminez toujours par des recommandations concrètes et actionnables

---

Q57. Qu’est-ce qu’un bon dashboard?

Réponse:

Un bon dashboard permet à l’utilisateur de comprendre rapidement la situation et de prendre des décisions éclairées. Il doit être à la fois informatif et facile à utiliser.

Principes de conception:

1. **Hiérarchie visuelle** - KPIs les plus importants en haut et en grand
2. **Cohérence** - Même palette de couleurs, même style de graphiques
3. **Interactivité appropriée** - Filtres utiles, mais pas trop
4. **Performance** - Chargement rapide (<3 secondes)
5. **Responsive** - Adapté au mobile si nécessaire

Structure typique recommandée:

```
KPIs principaux (cartes avec indicateurs)

Tendance          Répartition
(line chart)      (bar chart / pie)

Tableau détaillé avec tri et filtres
```

Q58. Comment gérer les outliers?

Réponse:

Les outliers sont des observations anormalement éloignées du reste des données. Leur traitement dépend du contexte et de l’objectif de l’analyse.

Options de traitement:

1. **Investiguer d’abord** - Sont-ils des erreurs de saisie ou des valeurs légitimes?
2. **Supprimer** - Si ce sont des erreurs confirmées
3. **Cap/Floor (Winsorizing)** - Remplacer les valeurs extrêmes par des percentiles (ex: 1% et 99%)
4. **Transformation** - Appliquer log ou racine carrée pour réduire l’asymétrie
5. **Analyse séparée** - Traiter les outliers comme un segment distinct

Important: En détection de fraude bancaire, les outliers SONT souvent les cas intéressants! Ne les supprimez pas aveuglément.

Q59. Comment valider la qualité des données?

Réponse:

La qualité des données est fondamentale pour des analyses fiables. Un framework de validation couvre plusieurs dimensions complémentaires.

Les 5 dimensions de la qualité:

Dimension	Question	Vérification
Complétude	Toutes les données sont-elles présentes?	Taux de valeurs manquantes
Unicité	Y a-t-il des doublons?	Comptage de duplicates
Exactitude	Les valeurs sont-elles correctes?	Plages attendues, règles métier
Cohérence	Les données sont-elles logiques?	Cross-checks entre colonnes
Fraîcheur	Les données sont-elles à jour?	Date de dernière mise à jour

```
def data_quality_report(df):
    """Génère un rapport de qualité des données"""
    return {
        'rows': len(df),
        'columns': len(df.columns),
        'duplicates': df.duplicated().sum(),
        'missing_pct': df.isnull().mean().to_dict(),
        'dtypes': df.dtypes.astype(str).to_dict()
    }
```

Q60. Comment aborder un nouveau dataset inconnu?

Réponse:

Face à un nouveau dataset, un analyste expérimenté suit un processus systématique pour comprendre rapidement les données. Voici un framework en 10 minutes.

```
# 1. STRUCTURE (30 secondes)
print(f"Dimensions: {df.shape}")
print(f"Colonnes: {df.columns.tolist()}")

# 2. TYPES DE DONNÉES (30 secondes)
print(df.dtypes)
print(df.info())

# 3. APERÇU (1 minute)
display(df.head(10))
display(df.sample(5))
```

```python
# 4. STATISTIQUES DESCRIPTIVES (2 minutes)
display(df.describe())
display(df.describe(include='object'))

# 5. QUALITÉ DES DONNÉES (2 minutes)
print("Valeurs manquantes:")
print(df.isnull().sum())
print(f"\nDoublons: {df.duplicated().sum()}")

# 6. DISTRIBUTIONS (2 minutes)
df.hist(figsize=(15, 10))
plt.show()

# 7. CORRÉLATIONS (2 minutes)
plt.figure(figsize=(10, 8))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.show()
```

À la fin de ces 10 minutes, vous devez savoir: - Taille et structure du dataset - Types de variables (numériques, catégorielles, dates) - Qualité des données (missing, doublons) - Distributions générales - Premières corrélations intéressantes

---

Résumé Final

Top 10 des Questions les Plus Probables pour UniBank Haiti

1. **Q2 (SQL):** ROW_NUMBER vs RANK vs DENSE_RANK - Classement des clients
2. **Q8 (SQL):** Problème N+1 - Optimisation des requêtes
3. **Q17 (DAX):** CALCULATE - Fonction centrale de Power BI
4. **Q18 (DAX):** % du total avec ALL - Reporting
5. **Q28 (Stats):** Moyenne vs Médiane - Analyse des montants
6. **Q30 (Stats):** p-value - Tests d'hypothèse
7. **Q38 (KPI):** NPL Ratio - Qualité du portefeuille
8. **Q39 (KPI):** ROE vs ROA - Rentabilité bancaire
9. **Q47 (Python):** groupby - Agrégations par catégorie
10. **Q55 (EDA):** Étapes d'une analyse exploratoire

BONNE CHANCE POUR VOTRE ENTRETIEN CHEZ UNIBANK HAITI!