Rapport sur les traitements des données

P4: Collectez des données en respectant les normes RGPD

Giulia Governatori - 10/01/2025

Description: traitement des dossiers de demande d'assurance automobile reçus, en respectant le Règlement Général de Protection des Données.

1. Contexte :
Dev'Immediat a été sanctionnée par la CNIL pour non-conformité aux normes du RGPD.
En conséquence, le CRM actuel est inutilisable, et vous ne pouvez pas exploiter les données personnelles de vos clients pendant au moins 6 mois.

2. Objectifs :
Pour permettre à votre service commercial de continuer à fonctionner, il est nécessaire d'extraire les données tarifaires de manière totalement anonymisée.

Ce rapport vise également à vous instruire sur les bonnes pratiques à adopter pour garantir la conformité aux normes RGPD et éviter de nouvelles sanctions à l'avenir. Il propose des recommandations concrètes pour une meilleure gestion et protection des données personnelles.

3. Préconisations :
L'utilisation des données personnelles doit être autorisée. Dans votre cas, cela nécessite d'obtenir le consentement des personnes concernées pour chaque traitement des données, à condition qu'il soit explicite, légitime et exprimé par un acte positif et non passif.

Pour assurer votre conformité, il est recommandé de :

- Tenir un registre des consentements collectés.
- Tenir un registre de tous les traitements effectués.
- Informer de manière transparente sur les finalités des traitements des données.
- Communiquer clairement sur les droits des personnes concernées, notamment :
    - droit d'accès (obtenir une copie des données personnelles traitées)
    - droit de rectification (corriger des données inexactes ou incomplètes)
    - droit à l'effacement (« droit à l'oubli »)
    - droit à la limitation du traitement
    - droit à la portabilité des données (récupérer les données dans un format structuré)
    - droit d'opposition (refuser l'utilisation des données à des fins spécifiques).

- Définir un cycle de vie des données personnelles (collecte, conservation, suppression) et s'assurer qu'il soit respecté. Une fois qu'ils ont atteint leur finalité, elles doivent être effacées ou anonymisées.

4. Anonymisation :
Dans ce cas, les données doivent être anonymisées pour pouvoir être utilisées. Après cette opération, il devient impossible d'identifier les clients. Cette opération est permanente et irréversible.

1. Toutes les données superflues doivent être supprimées.
2. Tous les noms, adresses e-mail et identifiants doivent être supprimés et remplacés par un identifiant aléatoire unique.
3. Les revenus, les devis et les valeurs de leurs biens doivent être regroupés par tranches.
4. Les numéros des enfants bénéficiant d'une "conduite accompagnée" doivent être convertis en valeur binaire (oui / non).
5. Les adresses doivent être plus génériques, par exemple sous forme de code postal ou départemental. La longitude et la latitude permettent d'obtenir une localisation précise du domicile de la personne, pourtant elles doivent être supprimées.
6. Les dates de naissance doivent être transformées en âge au moment de la demande.
7. Concernant les dates de la demande, nous pouvons garder le mois et l'année.

5. Documentation :

1. Le processus d'extraction de données.
2. L'élimination des données non nécessaires.
3. Les étapes pour retravailler le jeu de données et les explications associées.
4. La documentation des étapes de traitement (code M, Power Query).
5. Les requêtes SQL pour vérifier le succès de l'anonymisation
   5.1. Vérification de l'unicité des identifiants anonymes
   5.2. Vérification des revenus pour détecter d'éventuelles valeurs aberrantes
   5.3. Vérification des dates
   5.4. Vérification des codes postaux et départementaux
   5.5 Contrôler la répartition des données anonymes
6. Conclusion

---

1. EXTRACTION DE DONNÉES DU 2022 AVEC DOSSIER "COMPLET"

```sql
SELECT etat_dossier,
count(etat_dossier) AS count
FROM base_client
GROUP BY etat_dossier;
```

	etat_dossier	count
1	complet	8941
2	incomplet	1361

Sélection uniquement pour les données "complètes" de 2022, c'est-à-dire 1158.

```sql
SELECT etat_dossier,
count(etat_dossier) AS count
FROM base_client
WHERE date_demande LIKE "2022%"
AND etat_dossier = "complet"
GROUP BY etat_dossier;
```

	etat_dossier	count
1	complet	1158

Extraction de la sélection depuis SQLite et exportation au format CSV. Puis l'importer dans Excel.
SQLite > Tools > Export > Query results:

```sql
SELECT * FROM base_client
WHERE date_demande LIKE "2022%"
AND etat_dossier = "complet";
```

Vérifier que l'extraction a eu lieu pour toutes les données (1158), une ligne est la ligne d'en-têtes de colonnes.

1	metier	employeur	num_ss	groupe_sanguin
1154	chercheur en	Renaud	2,541E+14	B-
1155	v©lectromv(	Auger	1,8611E+14	O-
1156	aide-soignan	Benoit S.A.S.	2,4111E+14	AB-
1157	conseiller en	Briand Leclerc	2,9506E+14	B-
1158	web-ergonor	Guv©rin	2,7511E+14	A-
1159	agenceur de	Bernier	1,0207E+14	B+

2. MINIMISATION : élimination des données non nécessaires

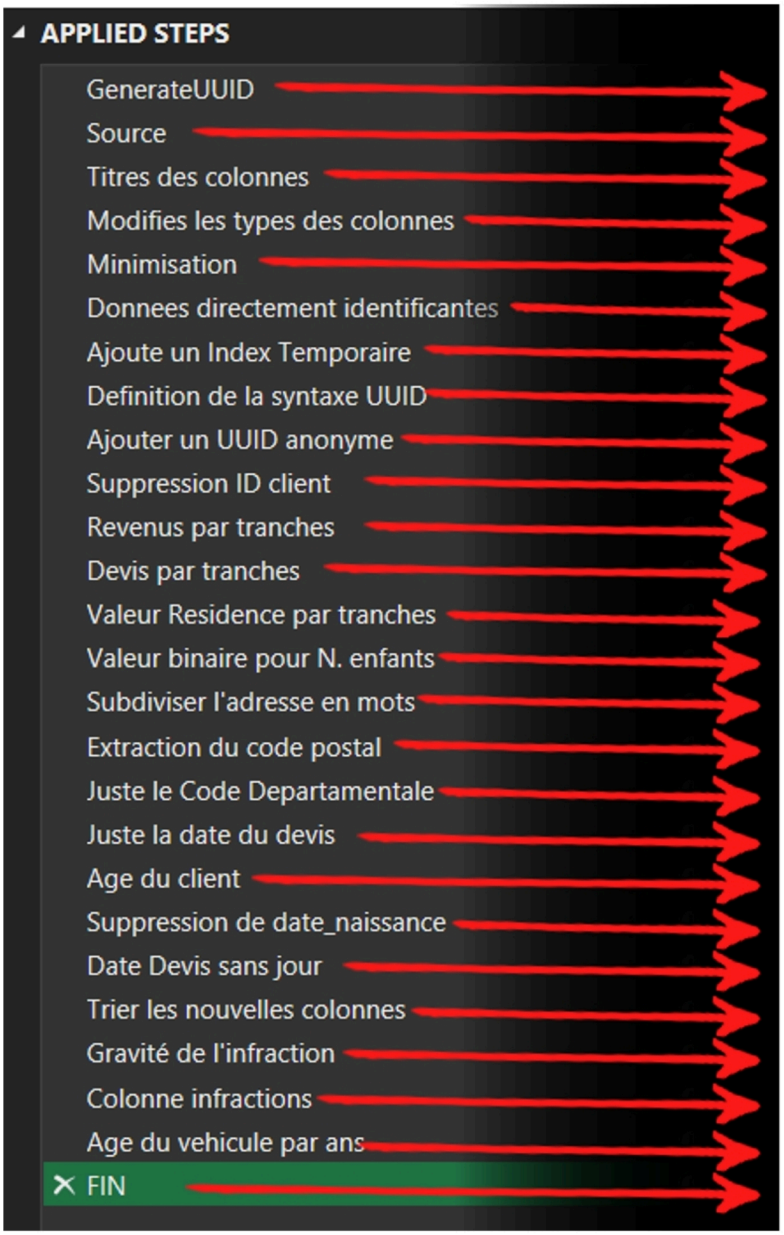
Dictionnaire ORIGINALE:	
champs non essentiels	en rouge
données identifiants :	en jaune
données à modifier :	en bleu
données OK :	en blanc

Nom Champ	Type	Description
metier	string	métier du client
employeur	string	employeur du client
num_ss	string	numéro de sécurité sociale du conducteur principal
groupe_sanguin	string	groupe sanguin du client
id_site_web	string	identifiant utilisé pour l'inscription sur notre site internet
nom	string	nom et prénom du client
sexe	string	sexe
email	string	email de contact du client
date_naissance	string	date de naissance
id_client	string	identifiant unique du client dans notre CRM
enfant_conduite_accompagne	string	nombre d'enfants en conduite accompagnée
nombre_enfants	string	nombre d'enfants du client
revenus	string	revenus de référence du client
valeur_residence_prin	string	valeur de la résidence principale du client (estimation du client)
formation	string	type de formation du client
usage_vehicule	string	usage principal du véhicule
type_vehicule	string	type de véhicule
est_rouge	string	indique si le véhicule du client est rouge
points_perdus	string	points retirés sur le permis de conduire à la date de la demande d`assurance
age_vehicule	string	âge du véhicule à assurer
type_conduite	string	type de conduite envisagée
date_demande	string	date de la demande de devis faite sur internet
etat_dossier	string	état du dossier : complet / incomplet
formule	string	formule choisie par le client lors de sa demande internet
tarif_devis	string	tarif accordé au client en fonction de son profil
adresse	string	adresse de la résidence principale du client
lat	string	latitude de la résidence principale du client
lon	string	longitude de la résidence principale du client

DICTIONNAIRE RÉSULTANT

Nom Champ	PK	Type	Description
UUID	PK	string	identifiant unique aléatoire dans le CRM
code_dep		string	code départemental de la résidence principale du client
date_demande		date	date de la demande de devis faite sans le jour (01/mm/yyyy)
formule		string	formule choisie par le client lors de sa demande internet
usage_vehicule		string	usage principal du véhicule
type_vehicule		string	type de véhicule
ans_vehicule		int64	âge du véhicule à assurer en années
type_conduite		string	type de conduite envisagée
infractions		string	gravité des infractions sur base de points retirés sur le permis de conduire à la date de la demande d'assurance
conduite_accompagne		string	conduite accompagnée en valeur binaire (oui/non)
age_client		int64	l'âge du client
sexe		string	le sexe du client
tranches_revenus		string	revenus de référence du client trié par tranches de 10k
valeur_residence_prin		string	valeur de la résidence principale du client (estimation client), trié par tranches de 50k
devis_tranches		string	tarif accordé au client en fonction de son profil, arrondi et trié par tranches de 10k

3. Résumé des mesures appliquées avec M sur Power Query



Fonction pour UUID
La source du fichier (BDD CRM)
Définition des titres des colonnes
Automatisation des types de données
Suppression des colonnes superflues
Autre suppression
Nouveau ID unique par ligne
Imposition de "0000-0000-0000..."
Création de UUID anonymes.
Suppression des client ID
Regroupement des montants des revenus
Regroupement des montants des devis
Regroupement de la valeur de leur biens
Conduite accompagnée: Oui/Non
sur de nouvelles colonnes
Uniquement le code postal
Uniquement le departement
Changement de format DateTime > Date
Calcul de l'âge des clients
Suppression de l'ancienne colonne
Juste le mois et l'année
Un petit rangement
Transformation de "points perdus"
Renommer la colonne "infractions"
De mois en années
Renommer la colonne "âge véhicule" - FIN

4. Code M de Power Query pour les modifications :

```
let

    // FONCTIONS M: Les fonctions sur M doivent etre declarees avant pour ne pas generer erreur de syntax

    // Fonction necessaire pour UUID: permettant de générer une valeur aléatoire pour chaque '0' rencontré dans la
"valeur de base" (chiffres et lettres majuscules/minuscules)

GenerateUUID = (baseText as text) as text =>
    Text.Combine(
        List.Transform(
            Text.ToList(baseText),
            each if _ = "0" then
                // Génère un nombre aléatoire ou une lettre aléatoire (majuscule ou minuscule)
                let
                    randomChoice = Number.RoundDown(Number.RandomBetween(1, 36)) // 36 pour inclure lettres et
chiffres
                in
                    if randomChoice <= 10 then
                        Text.From(Number.RoundDown(Number.RandomBetween(1, 9)))  // Génère un nombre aléatoire de
1 a 9
                    else if randomChoice <= 20 then
                        Character.FromNumber(Number.RoundDown(Number.RandomBetween(65, 90))) // majuscules (A-Z)
                    else
                        Character.FromNumber(Number.RoundDown(Number.RandomBetween(97, 122))) // minuscoles (a-z)
            else
                _
        ),
        ""
    ),

    // DEBUT DES OPERATIONS

    // Automatiosation de Power Query pour les types de colonnes
    Source = Csv.Document(File.Contents("C:\Users\Administrator\Desktop\P4\CRM_2022_complet.csv"),[Delimiter=";",
Columns=28, Encoding=65001, QuoteStyle=QuoteStyle.None]),

  #"Titres des colonnes" = Table.PromoteHeaders(Source, [PromoteAllScalars = true]),
  #"Modifies les types des colonnes" = Table.TransformColumnTypes(#"Titres des colonnes", {{"metier", type text},
{"employeur", type text}, {"num_ss", Int64.Type}, {"groupe_sanguin", type text}, {"id_site_web", type text},
{"nom", type text}, {"sexe", type text}, {"email", type text}, {"date_naissance", type date}, {"id_client",
Int64.Type}, {"enfant_conduite_accompagne", Int64.Type}, {"nombre_enfants", Int64.Type}, {"revenus", Int64.Type},
{"valeur_residence_prin", Int64.Type}, {"formation", type text}, {"usage_vehicule", type text}, {"type_vehicule",
type text}, {"est_rouge", type text}, {"points_perdus", Int64.Type}, {"age_vehicule", Int64.Type},
{"type_conduite", type text}, {"date_demande", type datetime}, {"etat_dossier", type text}, {"formule", type
text}, {"tarif_devis", Int64.Type}, {"adresse", type text}, {"lat", Int64.Type}, {"lon", Int64.Type}}, "it"),

    // Supprimer les colonnes pour MINIMISATION des données
    #"Minimisation" = Table.RemoveColumns(#"Modifies les types des colonnes", {"metier","employeur","num_ss",
"groupe_sanguin", "nombre_enfants", "formation", "est_rouge","etat_dossier"}),

    //ANONIMISATION DES DONNEES: LES DONNEES IDENTIFIANTS DIRECTEMENT

    // Supprimer les colonnes identifiantes directement
    #"Donnees directement identificantes" = Table.RemoveColumns(#"Minimisation", {"id_site_web","nom","email",
"lat", "lon"}),

    // Ajoute d'un ID temporaire
    #"Ajoute un Index Temporaire" = Table.AddIndexColumn(#"Donnees directement identificantes", "Index", 1, 1,
Int64.Type),

    // Ajoute d`une colonne temporaire avec "valeur_base" avec la valeur fixe "0000-0000-0000-0000-0000-..."
    #"Definition de la syntaxe UUID" = Table.AddColumn(
        #"Ajoute un Index Temporaire",
        "valeur_base",
        each "0000-0000-0000-0000-0000-0000-0000-0000-0000-0000"
    ),

    // Creation de la colonne "UUID" : applique la fonction pour chaque ligne
    #"Ajouter un UUID anonyme" = Table.AddColumn(
        #"Definition de la syntaxe UUID",
        "UUID",
        each GenerateUUID([valeur_base])
    ),

    // Supprimer les colonnes inutiles
    #"Suppression ID client" = Table.RemoveColumns(#"Ajouter un UUID anonyme", {"id_client", "Index",
"valeur_base"}),

    //ANONIMISATION DES DONNEES: LES DONNEES IDENTIFIANTES INDIRECTEMENT

    // Trier les revenues par tranches
    #"Revenus par tranches" = Table.RemoveColumns(
        Table.AddColumn(
            #"Suppression ID client",
            "tranches_revenus",
            each
```

```
                let
                    variabileA = Number.Round([revenus] / 10000) * 10,
                    variabileB = 0
                in
                    if [revenus] = 0 then null
                    else if [revenus] = null then null
                    else if variabileA = 0 then "0 - 10K"
                    else if variabileA > 200 then "200K +"
                    else Text.From((variabileB + variabileA - 10)) & " - " & Text.From(variabileA) & "K",
            type text
        ),
        {"revenus"}
),

    // Tarif du devis divisés par tranches de la meme façon que pour les revenus
    #"Devis par tranches" = Table.RemoveColumns(
    Table.AddColumn(
        #"Revenus par tranches",
        "devis_tranches",
        each
            let
                variabileC = Number.Round([tarif_devis] / 10000) * 10,
                variabileD = 0
            in
                if [tarif_devis] = 0 then null
                else if [tarif_devis] = null then null
                else if variabileC = 0 then "0 - 10K"
                else if variabileC > 60 then "60K +"
                else Text.From((variabileD + variabileC - 10)) & " - " & Text.From(variabileC) & "K",
        type text
    ),
    {"tarif_devis"}
),

// Valeur de la résidence principale par tranches sur la même colonne
#"Valeur Residence par tranches" = Table.TransformColumns(
    #"Devis par tranches",
    {
        "valeur_residence_prin",
        each
            let
                valeur_residence_prin = _, // L'underscore "_" représente la valeur de la colonne
                variabileE = Number.Round(valeur_residence_prin / 50000) * 50,
                variabileF = 0
            in
                if valeur_residence_prin = 0 then null
                else if valeur_residence_prin = null then null
                else if variabileE = 0 then "0 - 50K"
                else if variabileE > 450 then "450K +"
                else Text.From((variabileF + variabileE - 50)) & " - " & Text.From(variabileE) & "K",
        type text
    }
),

    // Remplacement le nombre d'enfants par une valeur binaire
    #"Valeur binaire pour N. enfants" = Table.RemoveColumns(Table.AddColumn(#"Valeur Residence par tranches",
"conduite_accompagne", each if [enfant_conduite_accompagne] = 0 then "non" else "oui"),
{"enfant_conduite_accompagne"}),

    // Subdiviser l'adresse en mots, uniquement si l'adresse n'est pas NULL
    #"Subdiviser l'adresse en mots" = Table.AddColumn(#"Valeur binaire pour N. enfants", "MotAdresse", each
        if [adresse] <> null then Text.Split([adresse], " ") else {}, type list),

    // Fonction permettant de trouver le code postal dans l'adresse
    #"Extraction du code postal" = Table.RemoveColumns(Table.AddColumn(#"Subdiviser l'adresse en mots",
"code_dep", each
        let
            // Extraction de mots d'adresse dans une liste provisoire
            mots = [MotAdresse],

            // Filtre pour obtenir uniquement les mots qui sont des nombres et qui contiennent au moins 4 chiffres
            chiffres = List.Select(mots, each
                try
                    Number.IsNaN(Number.FromText(_)) = false and Text.Length(_) >= 4
                otherwise
                    false
            ),

            // Il renvoie le dernier numéro valide
            codePostal = if List.Count(chiffres) > 0 then List.Last(chiffres) else null
        in
            // Il renvoie le code postal trouvé, sinon NULL
            codePostal
    ), {"adresse", "MotAdresse"}),

    // Estraction du code departemental
    #"Juste le Code Departamentale" = Table.TransformColumns(
    #"Extraction du code postal",
    {{"code_dep", each Text.Start(Text.From(_), 2), type text}}
),
```

```
    // Ne prendre que la date du devis, pas l'horaire: changer le format de la colonne
    #"Juste la date du devis" = Table.TransformColumns(
    #"Juste le Code Departamentale",
    {{"date_demande", each DateTime.Date(_), type date}}
),

    // Calculer l'âge sur la base de date_demande
    #"Age du client" = Table.AddColumn(
        #"Juste la date du devis",
        "age_client",
        each
            let
                age = Number.RoundDown(
                    Duration.Days(Duration.From(Date.From([date_demande]) - Date.From([date_naissance]))) / 365.25
                )
            in
                if age < 18 then null else age,
        Int64.Type
),

// Supprimer la vieille colonne date_naissance
    #"Suppression de date_naissance" = Table.RemoveColumns(#"Age du client", {"date_naissance"}),

    // Maintenant on elimine le jour: il mettra 01 comme jour à chaque mois
    #"Date Devis sans jour" = Table.TransformColumns(
    #"Suppression de date_naissance",
    {{"date_demande", each Date.FromText(Text.End(Text.From(_), 7)), type date}}
),

    // Réorganisation des colonnes du nouveau fichier .CSV
    #"Trier les nouvelles colonnes" = Table.ReorderColumns(
    #"Date Devis sans jour",
    {"UUID", "code_dep", "date_demande", "formule", "usage_vehicule", "type_vehicule", "age_vehicule",
"type_conduite", "points_perdus", "conduite_accompagne", "age_client", "sexe", "tranches_revenus",
"valeur_residence_prin", "devis_tranches"}
),

    // Remplacement des "points perdus" avec la "gravité de l'infraction" parce que les points perdus pourraient
rester identifiants
    #"Gravité de l'infraction" = Table.TransformColumns(
    #"Trier les nouvelles colonnes",
    {{"points_perdus",
        each
            if _ = null or _ = 0 then "aucune"
            else if _ = 1 then "légère"
            else if _ = 2 or _ = 3 then "modérée"
            else if _ >= 4 and _ <= 6 then "grave"
            else if _ > 6 then "très grave"
            else null,
        type text}}
),

    // Renommer la colonne "points perdus" en "infractions"
    #"Colonne infractions" = Table.RenameColumns(
        #"Gravité de l'infraction",
        {{"points_perdus", "infractions"}}
),

    // Remplacement de l'age du vehicule par mois, en ans
        #"Age du vehicule par ans" = Table.TransformColumns(
        #"Colonne infractions",
        {{"age_vehicule",
            each
                if _ = null then null
                else
                    let
                        age_en_ans = Number.RoundDown(_ / 12)
                    in
                        if age_en_ans >= 15 then "15+"
                        else Text.From(age_en_ans),
            type text}}
),

    // Renommer la colonne “age_vehicule” en “ans_vehicule
    #"FIN" = Table.RenameColumns(
        #"Age du vehicule par ans",
        {{"age_vehicule", "ans_vehicule"}}
)

in
    FIN
```

Pour vérifier que les données ont été **correctement anonymisées,** il est utile d'utiliser des requêtes SQL ciblées afin d'identifier des modèles ou des valeurs potentiellement sensibles.

5.1. Vérification de l'unicité des identifiants anonymes

Vérifier que l'identifiant aléatoire (ID_Random) est unique et ne suit pas de modèle reconnaissable.

```sql
SELECT UUID, COUNT(*)
FROM CRM
GROUP BY UUID
HAVING COUNT(*) > 1;
```

UUID	COUNT(*)

Le résultat est vide. S'il n'était pas vide, cela signifierait qu'il y aurait des doublons, ce qui pourrait compromettre l'unicité de la clé étrangère.

5.2. Vérification des revenus pour détecter d'éventuelles valeurs aberrantes

Vérifier que les revenus ont été correctement regroupés en tranches et ne révèlent pas de valeurs précises :

```sql
SELECT tranches_revenus, COUNT(*)
FROM CRM
GROUP BY tranches_revenus
ORDER BY COUNT(*) ASC;
```

tranches_reven...	COUNT(*)
180 - 190K	5
150 - 160K	7
190 - 200K	7
160 - 170K	9
200K +	10
170 - 180K	13
140 - 150K	15
130 - 140K	18

Si des valeurs aberrantes ou des schémas apparaissent, comme un revenu précis non arrondi, l'anonymisation peut ne pas être complète.

5.3. Vérification des dates

Nous pouvons conserver le mois et l'année de la demande de devis, mais pas le jour.

```sql
SELECT date_demande, COUNT(*)
FROM CRM
GROUP BY date_demande
ORDER BY date_demande;
```

date_demande	COUNT(*)
01/01/2022	236
01/02/2022	197
01/03/2022	254
01/04/2022	230
01/05/2022	241

5.4. Vérification des codes postaux et départementaux

Dans un premier temps, j'ai extrait les codes postaux des adresses.
Toutefois, comme le montre la requête SQL suivante, 625 personnes ont pu être identifiées parce qu'elles étaient les seules de leur commune à avoir demandé un devis.

```sql
CREATE VIEW code_postale_unique_view AS
SELECT code_postale, COUNT(*)
FROM CRM
GROUP BY code_postale
HAVING COUNT(*) =1;

SELECT count(*) AS "communes uniques"
FROM code_postale_unique_view;
```

communes uniques
625

J'ai ensuite extrait les deux premiers chiffres du code postal pour obtenir les codes départementaux.

```sql
SELECT code_dep, COUNT(*)
FROM CRM
GROUP BY code_dep
HAVING COUNT(*) = 1;
```

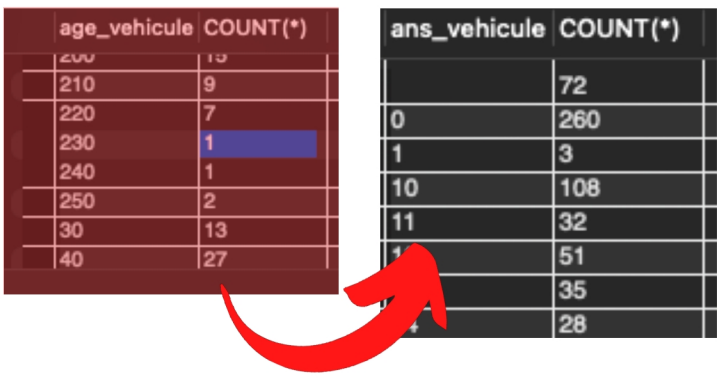
code_dep	COUNT(*)
97	1

La seule valeur unique applicable aux départements est celle des départements d'outre-mer. Cette valeur peut être négligée, car le département 97 comprend la Guadeloupe, la Martinique, la Guyane française, La Réunion et Mayotte, des régions très éloignées les unes des autres, et il est impossible de savoir à quel département cette donnée spécifique se rapporte.

5.5. Contrôler la répartition des données anonymes

Si les données sont regroupées, par exemple en fonction du revenu ou des codes postaux, vérifiez que les groupes soient homogènes. Exemple de requête SQL:

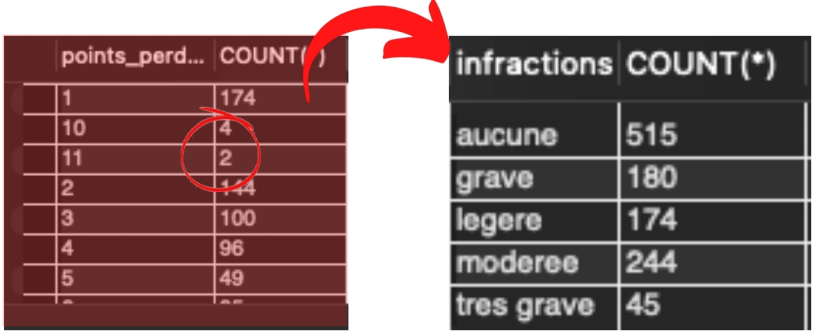
```sql
SELECT ans_vehicule, COUNT(*)
FROM CRM
GROUP BY ans_vehicule
ORDER BY ans_vehicule;
```



age_vehicule	COUNT(*)
210	9
220	7
230	1
240	1
250	2
30	13
40	27

ans_vehicule	COUNT(*)
	72
0	260
1	3
10	108
11	32
	51
	35
	28

L'âge du véhicule semble devoir être revu, ainsi que les « points perdus », les « tranches de revenus », et les « devis ».

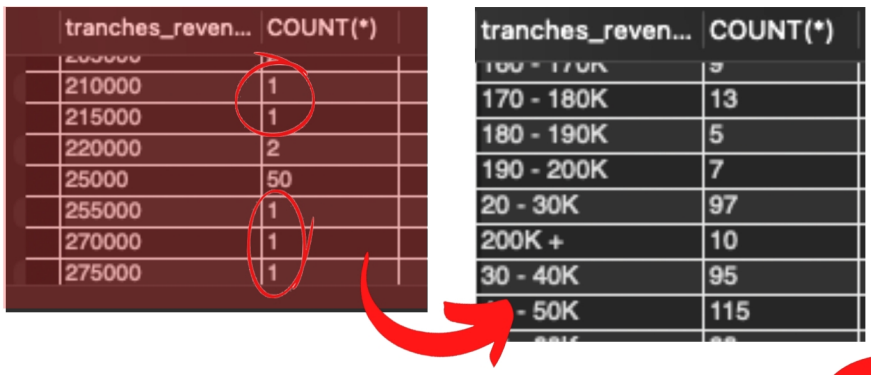


points_perd...	COUNT(*)
1	174
10	4
11	2
2	144
3	100
4	96
5	49

infractions	COUNT(*)
aucune	515
grave	180
legere	174
moderee	244
tres grave	45

Les “points perdus” ainsi acquièrent de la valeur et de l’anonymat.

Gravité	Points perdus	Exemples principaux
Infraction légère	1	Légère excès de vitesse, absence de clignotants.
Infraction modérée	2 o 3	Non-respect d'un stop, utilisation du téléphone en conduisant.
Infraction grave	4 o 6	Dépassements dangereux, conduite à contre-sens.
Infraction très grave	plus de 6	Conduite sous l'influence de l'alcool ou de drogues.

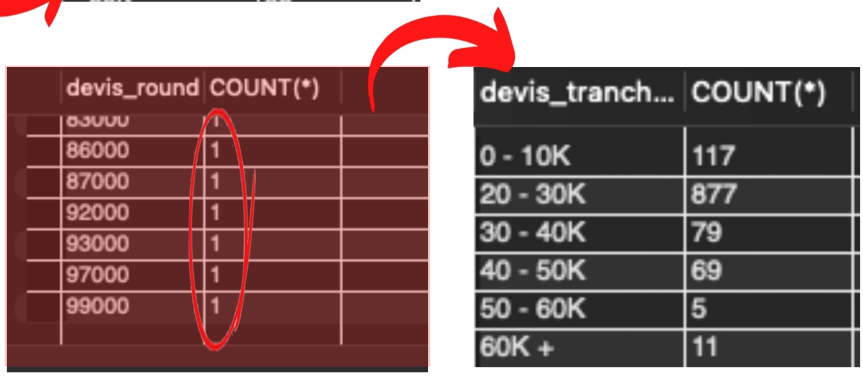


tranches_reven...	COUNT(*)
210000	1
215000	1
220000	2
25000	50
255000	1
270000	1
275000	1

tranches_reven...	COUNT(*)
170 - 180K	13
180 - 190K	5
190 - 200K	7
20 - 30K	97
200K +	10
30 - 40K	95
- 50K	115

Les tranches de revenus doivent être améliorées pour les riches: arrondir à 5000 n'est pas suffisante:

Les devis doivent également être révisés. L'arrondi par milliers n'est pas bon, il doit être fait par dizaines de milliers:



devis_round	COUNT(*)
86000	1
87000	1
92000	1
93000	1
97000	1
99000	1

devis_tranch...	COUNT(*)
0 - 10K	117
20 - 30K	877
30 - 40K	79
40 - 50K	69
50 - 60K	5
60K +	11

Des distributions fortement déséquilibrées peuvent indiquer des problèmes d'agrégation ou d’anonymisation.

Conclusion:

Ces requêtes vous aident à vérifier que :

1. Les données anonymes ne présentent pas de schémas évidents.
2. Les données sont cohérentes et agrégées correctement.
3. Il n'est pas possible d'identifier une personne de manière unique sur la base de combinaisons de champs.

Une fois que les données ont été complètement et irréversiblement anonymisées, elles peuvent être utilisées librement, sans aucune contrainte de temps ou d'utilisation. Toutefois, un usage éthique reste fortement recommandé.