Analyse des opportunités d'emploi

Introduction:

Ce rapport documente le processus de nettoyage des données,la visualization et la création d'une base de données SQL pour stocker les informations relatives aux offres d'emploi. L'objectif de ce projet était de transformer des données brutes en un ensemble structuré et cohérent, prêt à être utilisé pour des analyses approfondies dans le domaine du marché du travail. Le nettoyage des données est une étape essentielle pour garantir la qualité et la fiabilité des informations sur lesquelles nous allons baser nos futures décisions et analyses.

Nettoyage:

1. Types de Données et Conversion

La première étape du processus de nettoyage des données consistait à examiner les types de données de chaque colonne à l'aide de la commande df.dtypes. Afin d'assurer une analyse précise et un stockage efficace, certaines colonnes ont nécessité une conversion.

2. Gestion des Doublons

Avant de procéder au nettoyage des données, nous avons vérifié la présence de doublons dans l'ensemble de données à l'aide de la commande df.duplicated().sum(). Le nombre de doublons trouvés était de 202. Pour éliminer ces doublons, la fonction df.drop_duplicates() a été utilisée pour ne conserver que les enregistrements uniques dans l'ensemble de données.

3. Modification des Noms de Colonnes

Les noms des colonnes de l'ensemble de données ont été examinés, et il a été remarqué que certains noms de colonnes comportaient des espaces vides à la fin. Pour résoudre ce problème, la commande df.columns.str.rstrip() a été utilisée pour supprimer les espaces vides supplémentaires à la fin des noms de colonnes.

4. Gestion des Valeurs Manquantes

L'ensemble de données a ensuite été analysé pour détecter les valeurs manquantes à l'aide de la commande df.isnull().sum(). Les colonnes 'Experience level' et 'Salary' contenaient des valeurs manquantes. Les valeurs manquantes dans la colonne 'Experience level' ont été remplacées par 'Not-Specified', tandis que les valeurs manquantes dans la colonne 'Salary' ont été remplacées par 0.

5. Conversion du Salaire en USD

Pour faciliter une analyse et une comparaison significatives, la colonne 'Salary' devait être convertie en USD. La colonne 'Salary' contenait des valeurs de salaire dans divers formats, y compris des valeurs numériques avec des suffixes 'K', ainsi que des codes de devise tels que 'EUR' et 'GBP'. Une fonction convert_salary_to_usd a été créée pour extraire les valeurs numériques et les convertir en USD en fonction des taux de change pour les codes de devise 'EUR' et 'GBP'. Après la conversion, la colonne 'Salary' a été remplacée par la nouvelle colonne 'MNM_Salary_USD', et le nom de colonne d'origine a été mis à jour.

6. Extraction des Informations de Lieu

Pour mieux comprendre les lieux d'emploi, une fonction extract_country a été développée en utilisant des expressions régulières pour extraire les informations de pays de la colonne 'Location'. Par exemple, 'Richardson, TX, United States' a été transformé en 'United States'. Cette information de pays extraite a été ajoutée en tant que nouvelle colonne 'Country' dans l'ensemble de données.

7. Normalisation des Titres de Poste

La colonne 'Job Title' contenait une variété de titres qui devaient être normalisés pour l'analyse. Une liste de titres de poste standard a été créée, et la colonne 'Job Title' a été normalisée en utilisant la bibliothèque fuzzywuzzy. Les titres de poste avec un score minimal de 90 ont été associés aux titres de poste standard, tandis que les autres sont restés inchangés.

8. Traitement de la Colonne des Installations

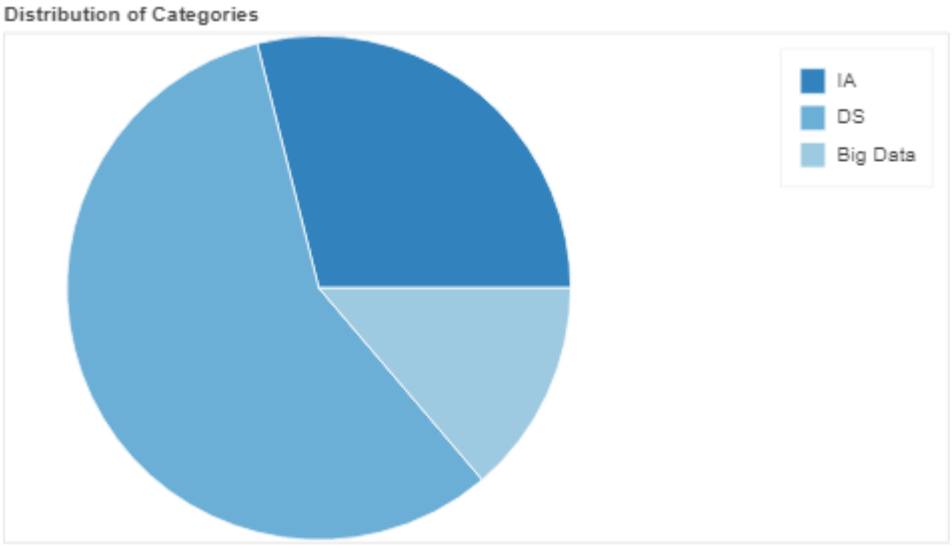
La colonne 'Facilities' contenait des valeurs séparées par des virgules. Pour nettoyer cette colonne, plusieurs remplacements ont été effectués pour supprimer les virgules supplémentaires et les virgules finales. Une fonction fixe_Facilities a été utilisée pour diviser les valeurs séparées par des virgules en listes. De plus, une fonction split_comma_column_values a été créée pour convertir les listes séparées par des virgules dans la colonne 'Requirment of the company' en lignes distinctes, réduisant ainsi la redondance.

9. Normalisation des Exigences de l'Entreprise

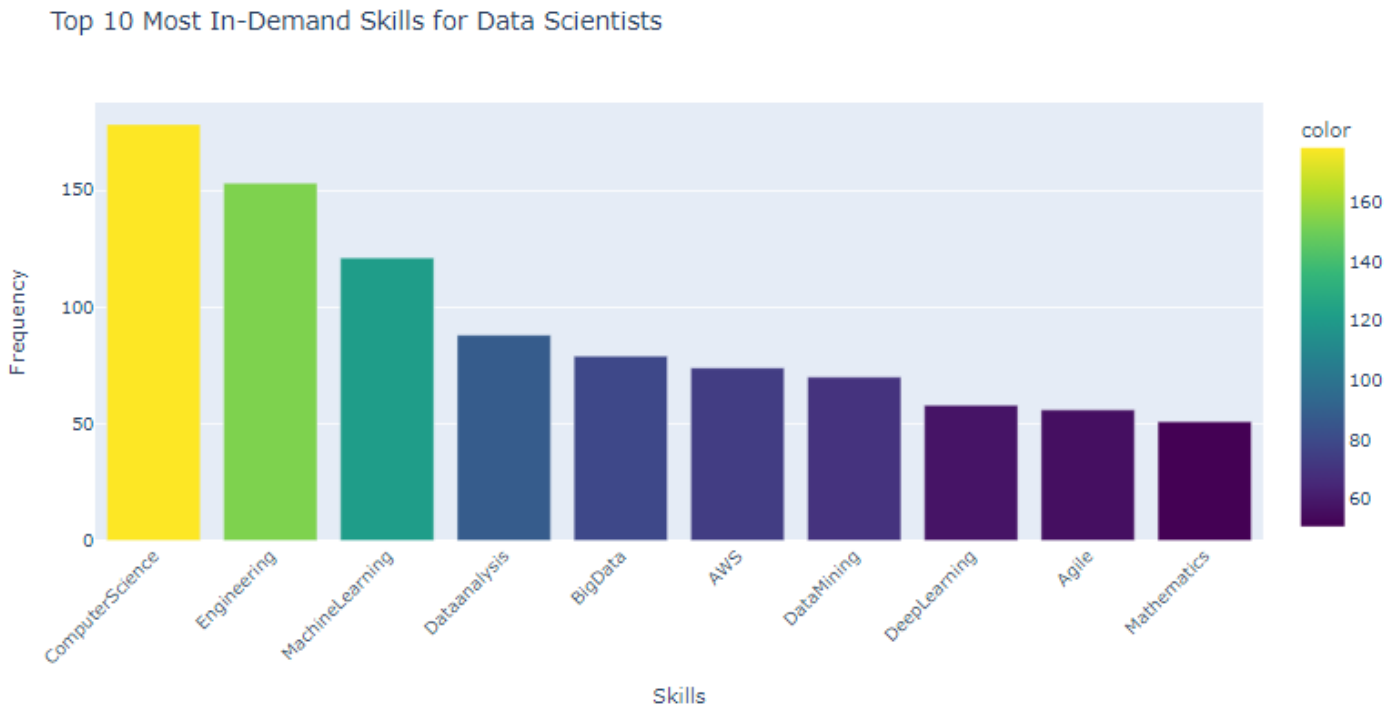
La colonne 'Requirment of the company' contenait une liste d'exigences séparées par des virgules, nécessitant une séparation et une normalisation appropriée pour faciliter l'analyse. Pour ce faire, nous avons utilisé la fonction personnalisée split_comma_column_values() qui a divisé les valeurs de cette colonne en plusieurs lignes distinctes. Chaque exigence de l'entreprise a été isolée et les doublons ont été éliminés à l'aide de drop_duplicates().

Visualization et Analyze:

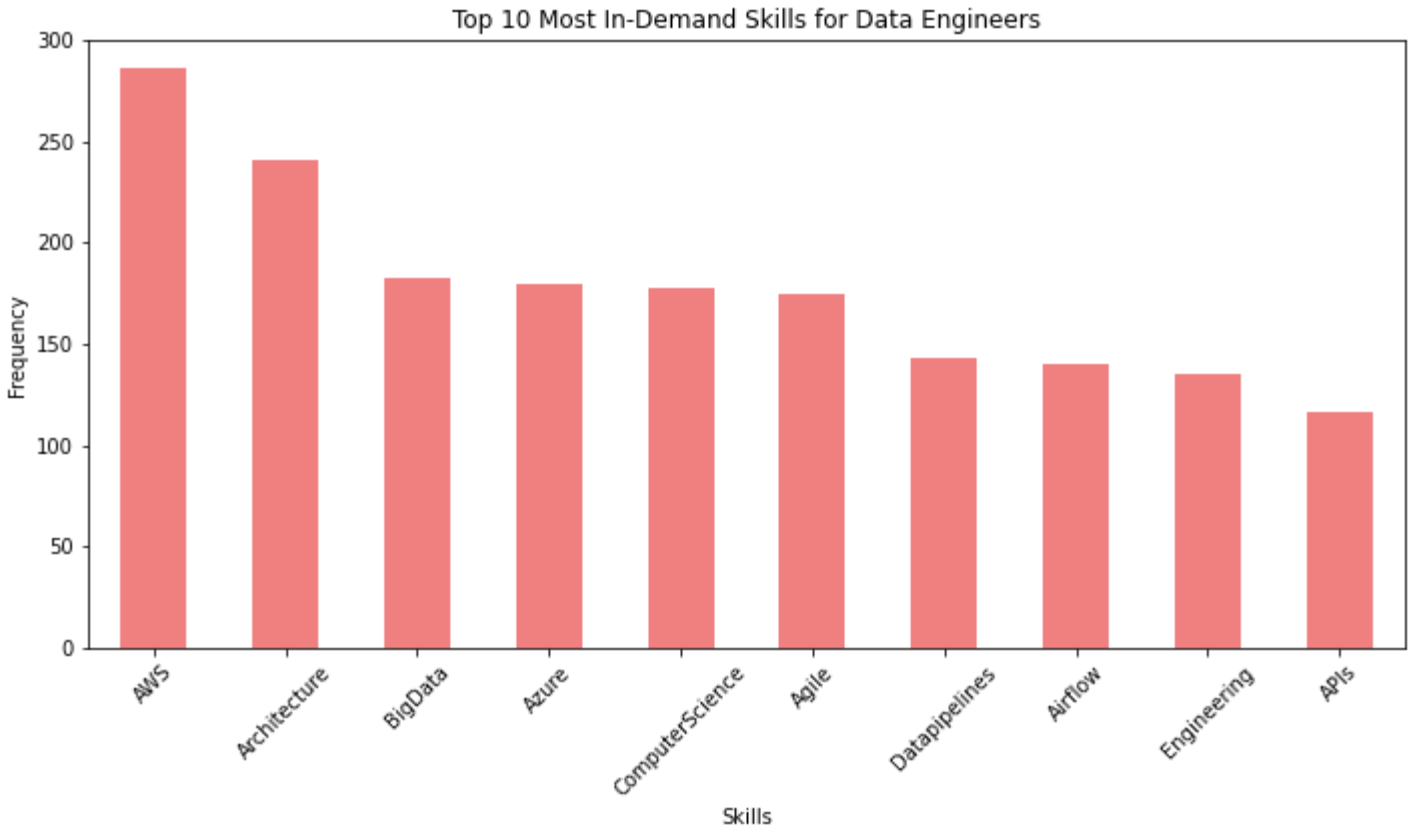
Les tendances clés du marché de l'emploi en IA, DS et Big Data



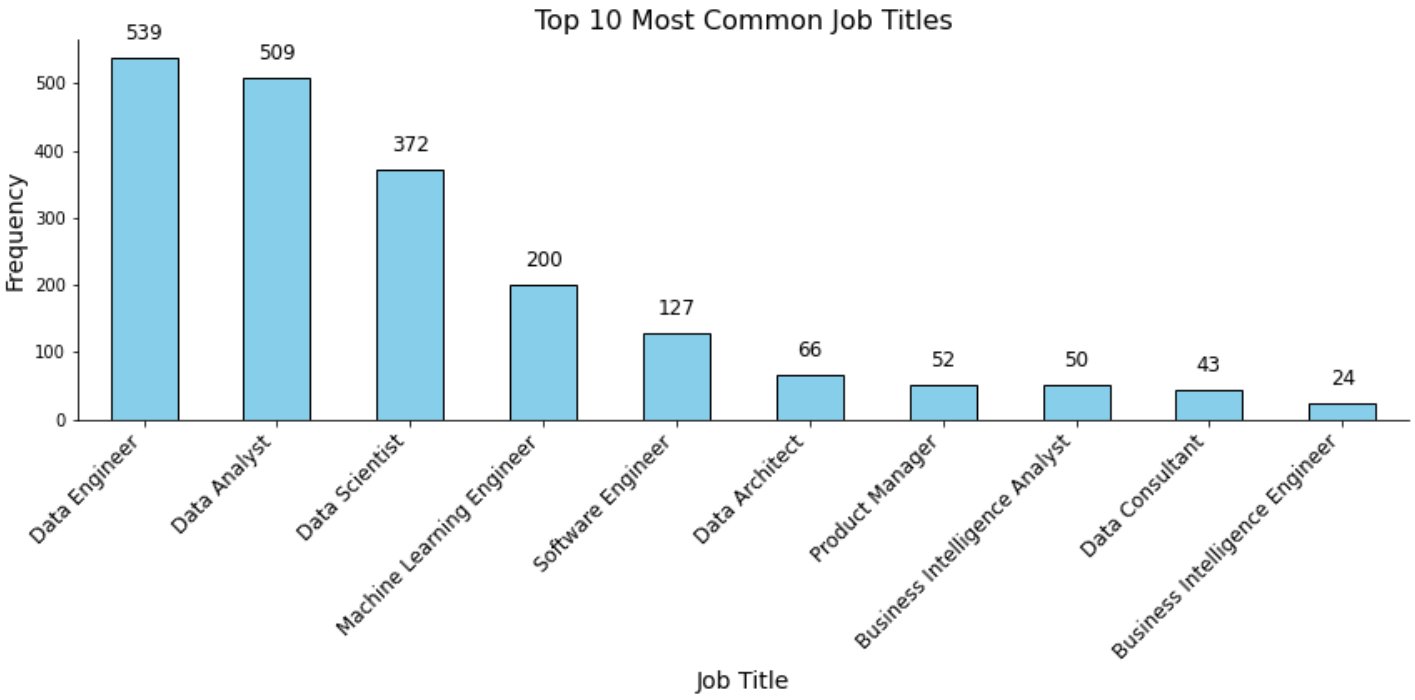
Les compétences les plus demandées pour un Data Scientist



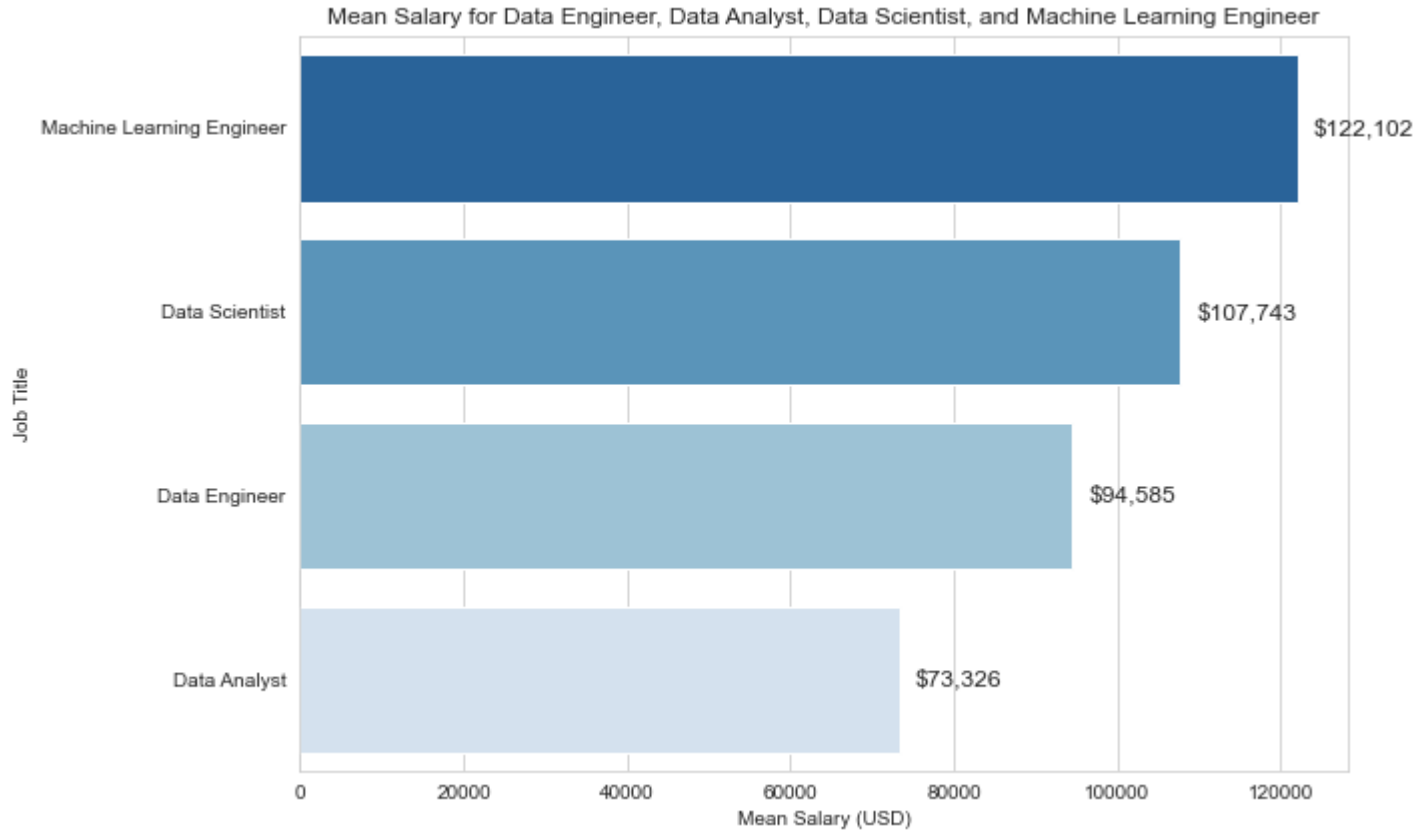
Les compétences les plus demandées pour un Data Engineer



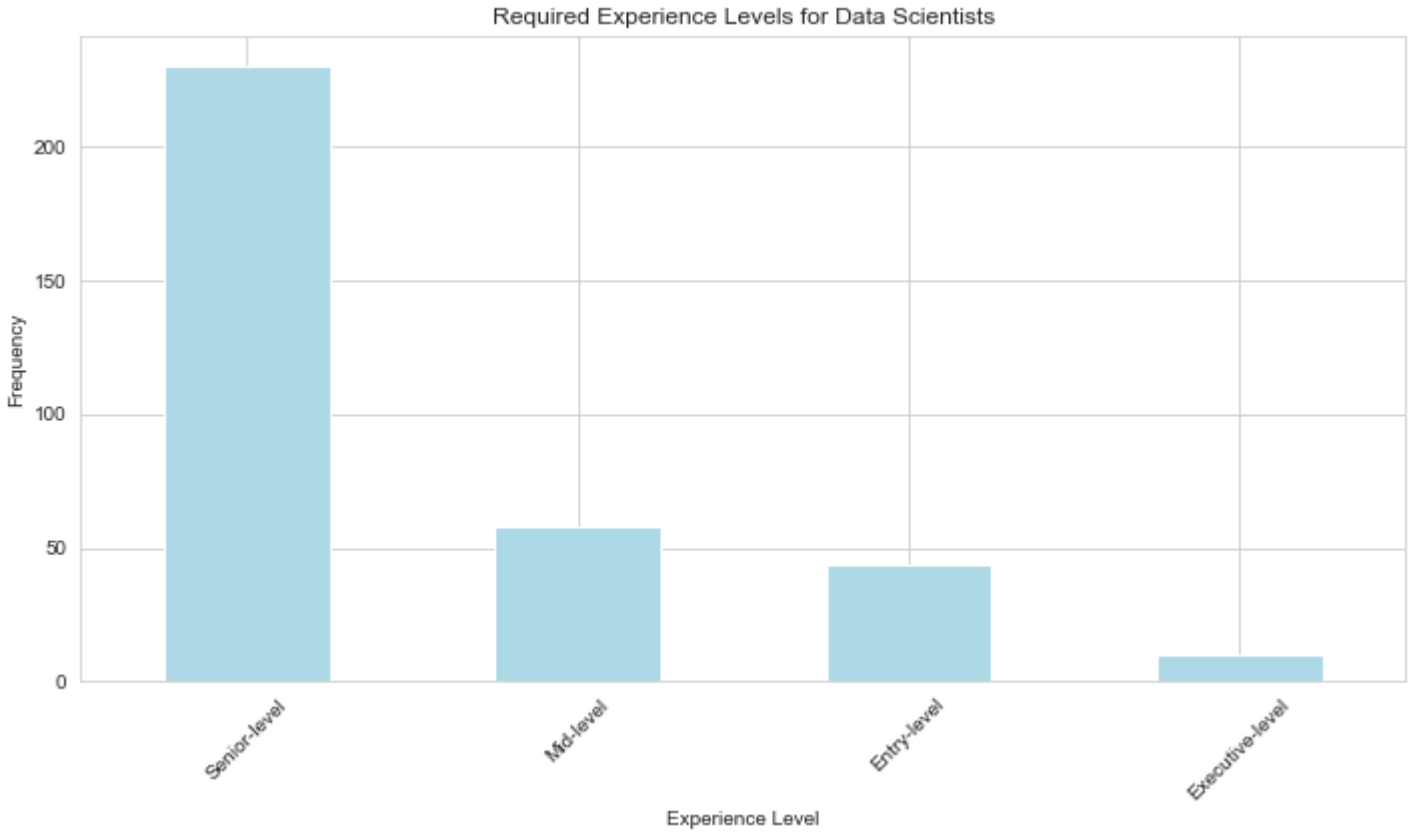
Top 10 des titres d'emploi les plus courants sur le marché du travail



Comparaison des salaires moyens pour Data Engineer, Data Analyst, Data Scientist, and Machine Learning Engineer

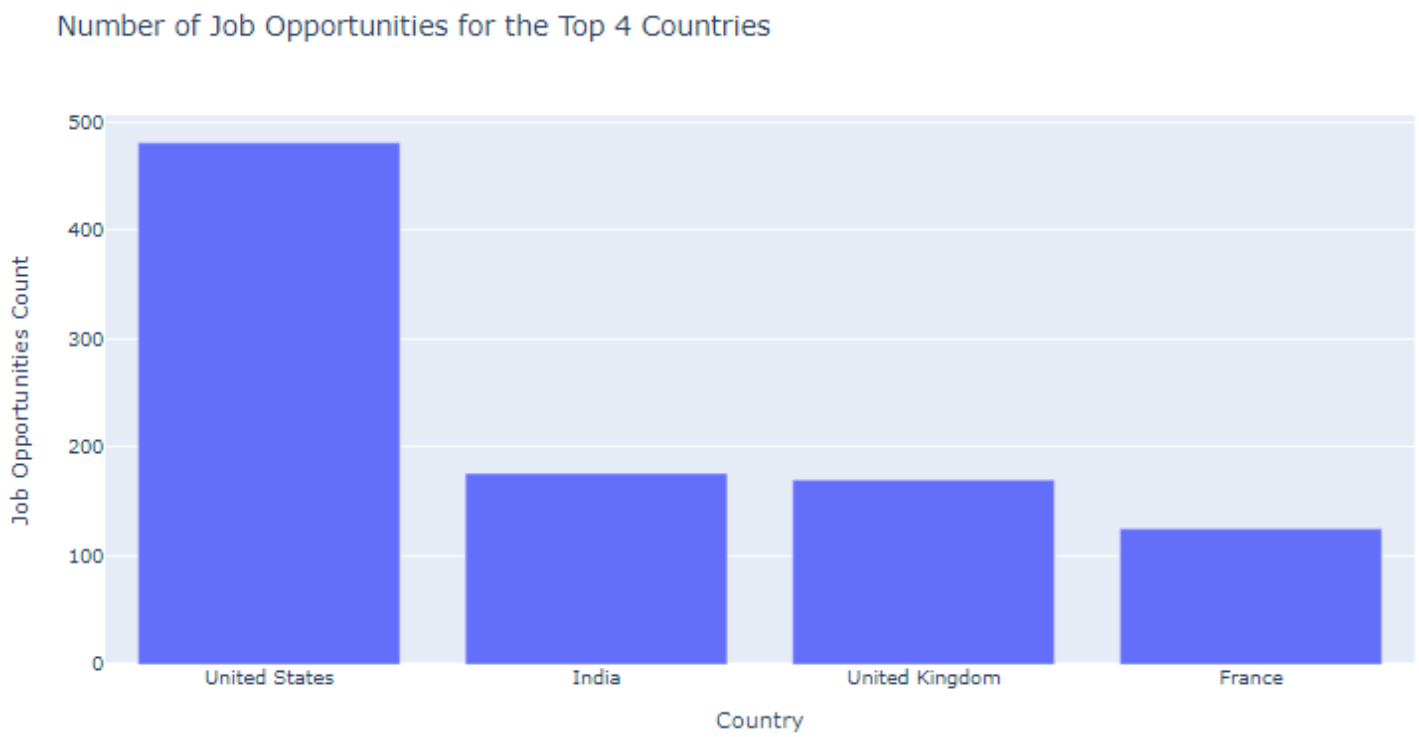


Niveaux d'expérience requis pour les Data Scientists



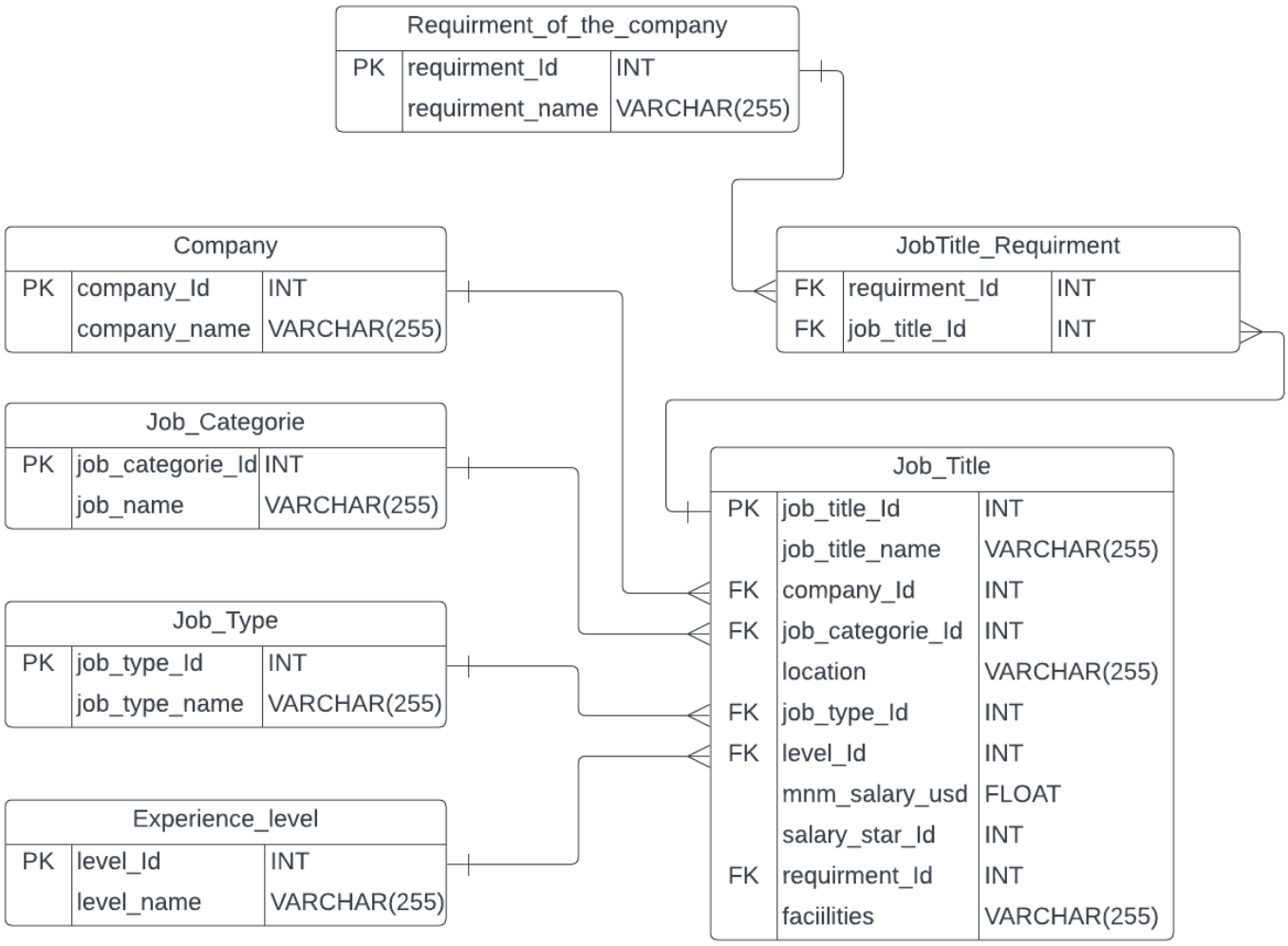
Types d'emplois pour Data Scientists, Data Analysts, and Data Engineers

<u>Top 4 des pays offrant le plus d'opportunités d'emploi</u>



Conception:

Avant d'importer les données dans la base de données SQL Server, la première étape consiste à concevoir la modélisation de la base de données. Cela comprend la création de classes, leurs attributs et les relations entre elles. Voici la modélisation conceptuelle de la base de données :



SQL:

La partie SQL du projet consiste à créer une base de données pour stocker les données nettoyées relatives aux offres d'emploi. Cette base de données sera utilisée pour stocker les informations sur les entreprises, les catégories d'emplois, les types d'emplois, les niveaux d'expérience, les exigences de l'entreprise et les titres de postes. Pour ce faire, nous utilisons Python et la bibliothèque pandas pour prétraiter les données, et nous utilisons le module pyodbc pour établir une connexion à une base de données SQL Server.

1. <u>Prétraitement des données</u>

Avant de créer les tables de la base de données, nous effectuons quelques étapes de prétraitement des données à l'aide de pandas. Tout d'abord, nous chargeons les données à partir du fichier 'CleanedReq.csv' dans un DataFrame appelé 'sql'. Ensuite, nous renommons certaines colonnes du DataFrame pour les rendre compatibles avec le schéma de la base de données que nous allons créer. Nous sélectionnons également certaines colonnes spécifiques pour des tables spécifiques.

En particulier, nous avons les tables suivantes :

- Table 'Company' : Cette table stocke les informations sur les entreprises. Elle comprend les colonnes 'company_Id' et 'company_name', où 'company_Id' est une clé primaire unique pour chaque entreprise.

- Table 'Job_Categorie' : Cette table contient les informations sur les catégories d'emplois. Elle comprend les colonnes 'job_categorie_Id' et 'job_name', où 'job_categorie_Id' est une clé primaire unique pour chaque catégorie d'emploi.

- Table 'Job_Type' : Cette table stocke les informations sur les types d'emplois. Elle comprend les colonnes 'job_type_Id' et 'job_type_name', où 'job_type_Id' est une clé primaire unique pour chaque type d'emploi.

- Table 'Experience_level' : Cette table contient les informations sur les niveaux d'expérience requis pour les emplois. Elle comprend les colonnes 'level_Id' et 'level_name', où 'level_Id' est une clé primaire unique pour chaque niveau d'expérience.

- Table 'Requirment_of_the_company' : Cette table stocke les exigences spécifiques de l'entreprise pour chaque offre d'emploi. Elle comprend les colonnes 'requirment_Id' et 'requirment_name', où 'requirment_Id' est une clé primaire unique pour chaque exigence de l'entreprise.

- Table 'Job_Title' : Cette table contient les informations détaillées sur les titres de postes, y compris les informations sur l'entreprise, la catégorie d'emploi, le type d'emploi, le niveau d'expérience, les salaires, les exigences de l'entreprise, etc. Elle comprend les colonnes 'job_title_Id', 'job_title_name', 'company_Id', 'job_categorie_Id', 'location', 'job_type_Id', 'level_Id', 'mnm_salary_usd', 'salary_star_Id', 'requirment_Id' et 'facilities'. 'job_title_Id' est une clé primaire unique pour chaque titre de poste.

- Table 'JobTitle_Requirment' : Cette table établit une relation entre les titres de postes et les exigences de l'entreprise. Elle comprend les colonnes 'requirment_Id' et 'job_title_Id', qui sont des clés étrangères faisant référence aux tables 'Requirment_of_the_company' et 'Job_Title', respectivement.

2. Création des tables et insertion des données

Une fois que nous avons défini la structure des tables, nous utilisons la connexion pyodbc pour créer une base de données SQL Server (dans notre cas, la base de données s'appelle 'Jobs') et créer les tables avec les contraintes appropriées, telles que les clés primaires et les clés étrangères.

Après avoir créé les tables, nous insérons les données prétraitées dans chacune des tables en utilisant des boucles pour parcourir le DataFrame correspondant. Avant d'insérer les données, nous vérifions si les enregistrements existent déjà dans la table en effectuant une requête de comptage pour éviter les doublons.

Conclusion:

En nettoyant les données et en créant une base de données SQL, nous avons réussi à transformer des données brutes en un ensemble bien structuré, prêt à être utilisé pour des analyses dans le domaine du marché du travail. Le nettoyage des données nous a permis d'améliorer la qualité et la cohérence des informations, tandis que la base de données SQL offre une solution robuste pour stocker et organiser les données de manière efficace. Ce projet a été une étape importante dans le processus d'exploration des données liées aux offres d'emploi, ouvrant la voie à des analyses plus avancées et à une meilleure compréhension du marché du travail.