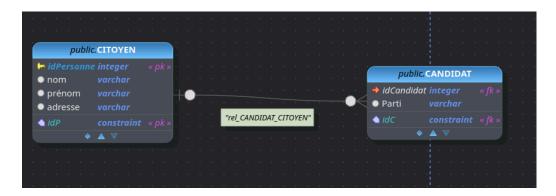


Dans ce projet, j'ai rédigé un script SQL complet permettant la création de cinq tables : region, sub\_region, country, disaster et climate\_disaster. L'objectif était de modéliser des données géographiques et climatiques en respectant les bonnes pratiques relationnelles. Pour cela, j'ai commencé par la table region, qui constitue la base du modèle, puis j'ai progressivement ajouté les autres tables en respectant les dépendances entre les clés étrangères. Par exemple, la table sub\_region ne peut être créée qu'après region, car elle en dépend via une clé étrangère. J'ai utilisé différents types de données comme INTEGER, VARCHAR et CHAR, ainsi que des contraintes telles que PRIMARY KEY, REFERENCES et NOT NULL pour garantir l'intégrité des données. La table climate\_disaster comprend une clé primaire composée de trois attributs, ce qui m'a permis de gérer des données multifactorielles. J'ai également veillé à structurer mon script de manière lisible, avec des commentaires explicatifs et une indentation claire. Ce travail m'a permis de renforcer mes compétences en création de tables, gestion des relations, clés primaires et étrangères, et organisation logique d'un script SQL.



Dans pgModeler, chaque colonne doit être **définie avec un type de données précis**, comme INTEGER, VARCHAR, ou CHAR, tout comme dans un script SQL classique. J'ai également ajouté les **contraintes nécessaires**, telles que les **clés primaires**, les **clés étrangères**, et les **contraintes NOT NULL**, directement dans l'interface graphique.

Une fois le modèle terminé, pgModeler permet de **générer automatiquement le script SQL complet**, que j'ai pu relire, commenter et exécuter pour créer la base de données.