* 🧠 **Brainstorming**
  + **Features we are going to predict** taking into account new data previous day (**input**).
  + Types de données **khubeo**

Structure de la table **T\_COMPTE\_RENDU** (PostgreSQL):

|  |  |
| --- | --- |
| Action | Code couleur |
| CREATION |  |

|  |  |
| --- | --- |
| Descriptif de la table à indiquer | Table contenant les rapports de compte pour entrainer le modèle AI |
| Champ clé unique | ID\_CLIENT |
| Index clustered *(si pas de clé unique)* | Nom applicable |
| Index non-clustered | ID\_DOSSIER, CR\_DATE |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Nom champ* | *Description* | *Type* | *NULL autorise* | *Increment auto* |
| ID\_CLIENT | Identifiant client unique | INT | NON | NON |
| ID\_DOSSIER | Identifiant du dossier | INT | NON | NON |
| ID\_ENSEIGNE | Identifiant enseigne | INT | NON | NON |
| NOM\_ENSEIGNE | Nom de l'enseigne | VARCHAR | NON | NON |
| CR\_DATA\_GENERATION | Date de génération des données | DATE | NON | NON |
| CR\_TYPE\_DATA | Type des données | VARCHAR | NON | NON |
| CR\_DATE | Date du rapport | DATE | NON | NON |
| CR\_MONTANT\_HT | Montant hors taxes | NUMERIC | NON | NON |
| CR\_MONTANT\_TOTAL | Montant avec taxes | NUMERIC | NON | NON |
| CR\_QUANTITE | Quantité | INT | NON | NON |
| ID\_CLIENT | Identifiant client unique | INT | NON | NON |
| ID\_DOSSIER | Identifiant du dossier | INT | NON | NON |

* + **Methods for transferring** data from SQL to PostgreSQL.

For large datasets:

* + - **Best for simplicity and performance**: Use bcp + PostgreSQL COPY.

**bcp (Bulk Copy Program)** is a command-line tool provided by Microsoft for **bulk exporting and importing data** between a SQL Server database and files. It is part of the SQL Server installation and is widely used for transferring large datasets efficiently.

**What Does bcp Do?**

* + - * Exports data from SQL Server tables or queries into files (e.g., CSV, text).
      * Imports data from files into SQL Server tables.
      * Operates in a high-performance, bulk-loading mode for large datasets.

**Why Use bcp?**

* + - * **High Performance**: Optimized for bulk operations, making it suitable for transferring millions of rows.
      * **Lightweight**: A simple command-line utility that doesn’t require additional setup.
      * **Flexible**: Supports various file formats and delimiter options.
      * **Interoperability**: Works well for exporting data to files that can then be used by other tools (e.g., PostgreSQL COPY).

**How to Use bcp?**

**1. Export Data from SQL Server**

To export data into a CSV file:

bcp "SELECT \* FROM database.schema.table" queryout "output.csv" -c -t, -S SERVER\_NAME -d DATABASE\_NAME -U USERNAME -P PASSWORD

* + - * **queryout**: Specifies that the output should go to a file.
      * **c**: Indicates character mode (for human-readable text files).
      * **t,**: Specifies the delimiter (, for CSV).
      * **S**: Specifies the SQL Server instance name.
      * **d**: Specifies the database name.
      * **U / P**: Specifies the username and password for authentication.

**2. Import Data into SQL Server**

To import data from a CSV file into a SQL Server table:

bcp database.schema.table in "input.csv" -c -t, -S SERVER\_NAME -U USERNAME -P PASSWORD

* + - * **in**: Specifies that the file should be loaded into the table.

**Use Case for PostgreSQL**

If you're transferring data from SQL Server to PostgreSQL:

* + - * Use bcp to export data from SQL Server into a **CSV file**.
      * Use PostgreSQL’s COPY command to load the CSV file into the PostgreSQL table:
      * COPY table\_name FROM '/path/to/output.csv' DELIMITER ',' CSV HEADER;

**Advantages of bcp for Large Data Sets**

* + - * **Handles Large Volumes**: Capable of processing millions of rows efficiently.
      * **Supports Queries**: Allows exporting data using custom SQL queries.
      * **Fast and Lightweight**: Faster than many GUI-based tools or manual methods.
      * **Standard Tool**: Comes with SQL Server, so no extra installation is needed.

**Example Workflow for SQL Server to PostgreSQL**

* + - * **Export from SQL Server**:
      * bcp "SELECT \* FROM BDERP.DBO.T\_COMPTE\_RENDU" queryout "compte\_rendu.csv" -c -t, -S SQL\_SERVER -d BDERP -U sa -P password
      * **Transfer File to PostgreSQL Server**: Use a file transfer method like scp to move the file:
      * scp compte\_rendu.csv postgres\_user@postgres\_server:/path/to/data/
      * **Import into PostgreSQL**: Use the COPY command:
      * COPY compte\_rendu FROM '/path/to/data/compte\_rendu.csv' DELIMITER ',' CSV HEADER;

**Conclusion**

bcp is an ideal tool for **large dataset transfers** when working with SQL Server. Combined with PostgreSQL’s COPY, it provides a fast and reliable way to move data between the two systems.

* + ¿Por qué utilizar el método CSV?
    - **High Performance**: Optimized for bulk operations, making it suitable for transferring millions of rows.
    - **Lightweight**: A simple command-line utility that doesn’t require additional setup.
    - **Flexible**: Supports various file formats and delimiter options.
    - **Interoperability**: Works well for exporting data to files that can then be used by other tools (e.g., PostgreSQL COPY).