Avancement SAE

S2.01 Conception et implémentation d'une base de données

Ibrahim BENKHERFELLAH Axel COULET

Université Sorbonne Paris Nord

May 18, 2025

Table des matières

Objectif du projet

- Comprendre et modéliser le commerce des technologies à faible émission de carbone.
- Concevoir une base de données normalisée à partir de deux jeux de données CSV.
- Interroger et visualiser les données pour produire des analyses pertinentes.

Objectif du projet

- Comprendre et modéliser le commerce des technologies à faible émission de carbone.
- Concevoir une base de données normalisée à partir de deux jeux de données CSV.
- Interroger et visualiser les données pour produire des analyses pertinentes.

Objectif du projet

- Comprendre et modéliser le commerce des technologies à faible émission de carbone.
- Concevoir une base de données normalisée à partir de deux jeux de données CSV.
- Interroger et visualiser les données pour produire des analyses pertinentes.

Étude des sources de données

- Fichiers CSV étudiés :
  - `Trade_in_Low_Carbon_Technology_Products.csv`
  - `Bilateral_Trade_in_Low_Carbon_Technology_Products.csv`
- Exploration initiale avec Python (pandas) : `.head()`, `.info()`, etc.
- Identification de colonnes clés : `CTS_Code`, `Indicator`, `Trade_Flow`, etc.
- Difficultés rencontrées : [à compléter]

Étude des sources de données

- Fichiers CSV étudiés :
  - `Trade_in_Low_Carbon_Technology_Products.csv`
  - `Bilateral_Trade_in_Low_Carbon_Technology_Products.csv`
- Exploration initiale avec Python (pandas) : `.head()`, `.info()`, etc.
- Identification de colonnes clés : `CTS_Code`, `Indicator`, `Trade_Flow`, etc.
- Difficultés rencontrées : [à compléter]

Étude des sources de données

- Fichiers CSV étudiés :
  - `Trade_in_Low_Carbon_Technology_Products.csv`
  - `Bilateral_Trade_in_Low_Carbon_Technology_Products.csv`
- Exploration initiale avec Python (pandas) : `.head()`, `.info()`, etc.
- Identification de colonnes clés : `CTS_Code`, `Indicator`, `Trade_Flow`, etc.
- Difficultés rencontrées : [à compléter]

Étude des sources de données

- Fichiers CSV étudiés :
  - `Trade_in_Low_Carbon_Technology_Products.csv`
  - `Bilateral_Trade_in_Low_Carbon_Technology_Products.csv`
- Exploration initiale avec Python (pandas) : `.head()`, `.info()`, etc.
- Identification de colonnes clés : `CTS_Code`, `Indicator`, `Trade_Flow`, etc.
- Difficultés rencontrées : [à compléter]

Sources et structuration des fichiers

- Données structurées en colonnes d'années : F1994 à F2023.
- Présence de colonnes de description redondantes (`CTS_Code`, `CTS_Name`, `CTS_Full_Descriptor`).
- Première étape : lecture manuelle pour repérage des redondances et structures tabulaires.

Sources et structuration des fichiers

- Données structurées en colonnes d'années : F1994 à F2023.
- Présence de colonnes de description redondantes (`CTS_Code`, `CTS_Name`, `CTS_Full_Descriptor`).
- Première étape : lecture manuelle pour repérage des redondances et structures tabulaires.

Sources et structuration des fichiers

- Données structurées en colonnes d'années : F1994 à F2023.
- Présence de colonnes de description redondantes (`CTS_Code`, `CTS_Name`, `CTS_Full_Descriptor`).
- Première étape : lecture manuelle pour repérage des redondances et structures tabulaires.

Préparation et normalisation des données

- Conversion des colonnes FXXXX en format court (année, valeur).
- Normalisation des années (F1994 à F2023) :
  - Chaque colonne FXXXX devient une ligne avec une valeur d'année.
  - Transformation effectuée via `UNION ALL` dans SQL pour correspondre à la logique de `stack()` en Python.
- Problèmes rencontrés :
  - Ambiguïté sur le pays origine.
  - Duplication de lignes si filtrage incorrect.

Préparation et normalisation des données

- Conversion des colonnes FXXXX en format court (année, valeur).
- Normalisation des années (F1994 à F2023) :
  - Chaque colonne FXXXX devient une ligne avec une valeur d'année.
  - Transformation effectuée via `UNION ALL` dans SQL pour correspondre à la logique de `stack()` en Python.
- Problèmes rencontrés :
  - Ambiguïté sur le pays origine.
  - Duplication de lignes si filtrage incorrect.

Préparation et normalisation des données

- Conversion des colonnes FXXXX en format court (année, valeur).
- Normalisation des années (F1994 à F2023) :
  - Chaque colonne FXXXX devient une ligne avec une valeur d'année.
  - Transformation effectuée via `UNION ALL` dans SQL pour correspondre à la logique de `stack()` en Python.
- Problèmes rencontrés :
  - Ambiguïté sur le pays origine.
  - Duplication de lignes si filtrage incorrect.

Données structurées et décisions clés

- Création d'une table `echanger_avec` pour les données bilatérales :
  - Relation réflexive entre deux pays (deux clés étrangères vers `country`).
  - Attribution de `pays_origine` selon le sens du flux.
- Table `donnee_pays` pour les données nationales.
- Lien systématique aux dimensions : pays, indicateur, catégorie, année.

Données structurées et décisions clés

- Création d'une table `echanger_avec` pour les données bilatérales :
  - Relation réflexive entre deux pays (deux clés étrangères vers `country`).
  - Attribution de `pays_origine` selon le sens du flux.
- Table `donnee_pays` pour les données nationales.
- Lien systématique aux dimensions : pays, indicateur, catégorie, année.

Données structurées et décisions clés

- Création d'une table `echanger_avec` pour les données bilatérales :
  - Relation réflexive entre deux pays (deux clés étrangères vers `country`).
  - Attribution de `pays_origine` selon le sens du flux.
- Table `donnee_pays` pour les données nationales.
- Lien systématique aux dimensions : pays, indicateur, catégorie, année.

Modèle conceptuel (EA)

- Type Entités : `Country`, `Indicator`, `CTS`, `Trade_Flow`, `Year`
- Type Associations :
  - `Échanger_Avec` (réflexive sur `Country`)
  - `Donnée_Pays` (relation simple avec agrégation)
- Justification des cardinalités et des relations
- Illustration du schéma :

Modèle conceptuel (EA)

- Type Entités : `Country`, `Indicator`, `CTS`, `Trade_Flow`, `Year`
- Type Associations :
  - `Échanger_Avec` (réflexive sur `Country`)
  - `Donnée_Pays` (relation simple avec agrégation)
- Justification des cardinalités et des relations
- Illustration du schéma :

Modèle conceptuel (EA)

- Type Entités : `Country`, `Indicator`, `CTS`, `Trade_Flow`, `Year`
- Type Associations :
  - `Échanger_Avec` (réflexive sur `Country`)
  - `Donnée_Pays` (relation simple avec agrégation)
- Justification des cardinalités et des relations
- Illustration du schéma :

Modèle conceptuel (EA)

- Type Entités : `Country`, `Indicator`, `CTS`, `Trade_Flow`, `Year`
- Type Associations :
  - `Échanger_Avec` (réflexive sur `Country`)
  - `Donnée_Pays` (relation simple avec agrégation)
- Justification des cardinalités et des relations
- Illustration du schéma :

Structure relationnelle finale

- Schéma relationnel résultant de la modélisation EA :
  - `country(idCountry, Country, ISO2, ISO3)`
  - `cts(idCTS, CTS_Code, CTS_Name, CTS_Full_Descriptor)`
  - `trade_flow(idTrade, Trade_Flow, Scale)`
  - `indicator(idIndicator, Indicator_, Source, Units)`
  - `year(id_Year, Year)`
  - `echanger_avec(id_country_1, id_country_2, pays_origine, id_indicator, id_cts, id_trade, id_year, trade_value)`
  - `donnee_pays(id_country, id_indicator, id_cts, id_trade, id_year, trade_value)`

Forme normale : 1NF

- Toutes les valeurs sont atomiques : aucun champ multivalué ou composé.
- Transformation des colonnes F1994 à F2023 en une colonne `id_year` via pivot SQL.
- Tables relationnelles sans redondance horizontale.
- **Conclusion** : le schéma respecte la **première forme normale (1NF)**.

Forme normale : 2NF

- Le schéma est en 1NF.
- Aucune dépendance fonctionnelle partielle dans les tables à clés composites.
- Exemple : les descripteurs CTS ont été extraits dans une table distincte, reliée par `CTS_Code`.
- **Conclusion** : toutes les dépendances fonctionnelles concernent la clé entière $\rightarrow$ **2NF validée**.

Forme normale : 3NF

- Le schéma est en 2NF.
- Suppression des dépendances transitives.
- Exemple : l'unité d'un indicateur dépend directement de `indicator`, pas d'un pays ou d'un code CTS.
- **Conclusion** : toutes les colonnes non-clés dépendent uniquement de la clé primaire $\rightarrow$ **schéma en 3NF**.

Forme normale : BCNF

- Toutes les dépendances fonctionnelles ont un antécédent qui est une super-clé.
- Exemple : dans `echanger_avec`, seule la combinaison complète des identifiants détermine `trade_value`.
- Il n'existe pas de dépendance fonctionnelle violant cette condition.
- **Conclusion** : le schéma respecte la forme normale de **Boyce-Codd (BCNF)**.

Création de la table Country

```
CREATE TABLE Country(
    id_Country INTEGER PRIMARY KEY,
    Country VARCHAR(50),
    ISO2 CHAR(2),
    ISO3 CHAR(3)
);
```

Création de la table Indicator

```
CREATE TABLE Indicator(
    id_Indicator INTEGER PRIMARY KEY,
    Indicator_ VARCHAR(80),
    Source VARCHAR,
    Units VARCHAR(50)
);
```

Création de la table CTS

```
CREATE TABLE CTS(
    id_CTS INTEGER PRIMARY KEY,
    CTS_Code VARCHAR(6),
    CTS_Name VARCHAR(100),
    CTS_Full_Descriptor VARCHAR(150)
);
```

Création de la table Trade_Flow

```
CREATE TABLE Trade_Flow(
    id_Trade INTEGER PRIMARY KEY,
    Trade_Flow VARCHAR(20),
    Scale VARCHAR(5)
);
```

Création de la table Year

```
CREATE TABLE Year(
    id_Year INTEGER PRIMARY KEY,
    Year DATE
);
```

Création de la table Echanger_Avec

```
CREATE TABLE Echanger_Avec (
    id_Country_1 INTEGER REFERENCES Country(id_Country),
    id_Country_2 INTEGER REFERENCES Country(id_Country),
    Pays_Origine INTEGER REFERENCES Country(id_Country),
    id_Indicator INTEGER REFERENCES Indicator(id_Indicator),
    id_CTS INTEGER REFERENCES CTS(id_CTS),
    id_Trade INTEGER REFERENCES Trade_Flow(id_Trade),
    id_Year INTEGER REFERENCES Year(id_Year),
    trade_value DOUBLE PRECISION,
    PRIMARY KEY(id_Country_1, id_Country_2, Pays_Origine,
                id_Indicator, id_CTS, id_Trade, id_Year),
    CHECK (id_Country_1 < id_Country_2)
);
```

Création de la table Donnee_Pays

```
CREATE TABLE Donnee_Pays (
    id_Country INTEGER REFERENCES Country(id_Country),
    id_Indicator INTEGER REFERENCES Indicator(id_Indicator),
    id_CTS INTEGER REFERENCES CTS(id_CTS),
    id_Trade INTEGER REFERENCES Trade_Flow(id_Trade),
    id_Year INTEGER REFERENCES Year(id_Year),
    trade_value DOUBLE PRECISION,
    PRIMARY KEY(id_Country, id_Indicator, id_CTS,
    id_Trade, id_Year)
);
```

Peuplement de la base

- Utilisation de `COPY`, `INSERT INTO` avec jointures.
- Alignement avec les traitements Python (`melt + filtering`).
- Cas particuliers traités :
  - Échanges d'un pays avec lui-même
  - Lignes `NULL` / Trade Flow = Not Applicable
- Difficultés rencontrées : [à compléter]

Requêtes d'analyse

- Nombre d'échanges par année
- Top pays exportateurs/importateurs
- Évolution des indicateurs par catégorie CTS
- Comparaison SQL x CSV avec Python

Visualisation

- Utilisation de Metabase pour interroger la BD
- Utilisation de Tableau pour valider les valeurs issues du CSV
- Comparaison SQL / Python $\rightarrow$ vérification de l'intégrité
- Exemples visuels : [insérer graphiques ou screenshots]

Bilan du projet

- Base de données fonctionnelle et fidèle au CSV (dans nos rêves)
- Respect des étapes du processus de modélisation
- Difficultés sur l'import (pivot, valeurs nulles, sens du flux)
- Vérification croisée avec Python
- Améliorations possibles : [à compléter]

Merci pour votre attention

Questions ?