Définition mathématiques de l'information.

par AYOUBA Anrezki *15/01/2025*

1 Les algorithmes de dissimulation et d'extraction.

1.1 LSB

1.2 DTC

1.3 MPA

2 Base de données

2.1 Prototypage :

```
#V2
import sqlite3

def create_database():
    # Connexion à la base de données SQLite (création si elle n'existe pas)
    conn = sqlite3.connect("local_db.db")
    cursor = conn.cursor()

    # Création de la table hote_data
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS hote_data (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            path TEXT NOT NULL,
            type TEXT NOT NULL,
            methode TEXT NOT NULL,
            taille REAL NOT NULL,
            note TEXT
        )
    ''')

    # Création de la table hide_data
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS hide_data (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            content TEXT NOT NULL,
            type TEXT NOT NULL CHECK(type = 'texte'),
            id_hoteData INTEGER NOT NULL,
            taille REAL NOT NULL,
            note TEXT,
            FOREIGN KEY (id_hoteData) REFERENCES hote_data (id)
        )
    ''')

    # Création de la table cover_data
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS cover_data (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            path TEXT NOT NULL,
```

```
            type TEXT NOT NULL,
            id_hideDat INTEGER NOT NULL,
            FOREIGN KEY (id_hideDat) REFERENCES hide_data (id)
        )
    ''')

    # Création de la table data
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS data (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            id_coverData INTEGER NOT NULL,
            entropie_des_donnees_cachees REAL,
            variance_des_donnees_porteuses REAL,
            tests_de_normalite INTEGER CHECK(tests_de_normalite IN (0, 1)),
            resistance_a_la_compression REAL,
            resistance_au_bruit REAL,
            test_du_khi_carree INTEGER CHECK(test_du_khi_carree IN (0, 1)),
            spectre BLOB,
            transformation_en_ondelette BLOB,
            la_transformee_de_laplace BLOB,
            la_dct_pour_la_compression BLOB,
            l_analyse_de_la_densite_spectrale_de_puissance BLOB,
            FOREIGN KEY (id_coverData) REFERENCES cover_data (id)
        )
    ''')

    cursor.execute('''
        CREATE TABLE IF NOT EXISTS data_without_transformation (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            id_hote INTEGER NOT NULL,
            entropie_des_donnees_cachees REAL,
            variance_des_donnees_porteuses REAL,
            tests_de_normalite INTEGER CHECK(tests_de_normalite IN (0, 1)),
            resistance_a_la_compression REAL,
            resistance_au_bruit REAL,
            test_du_khi_carree INTEGER CHECK(test_du_khi_carree IN (0, 1)),
            spectre BLOB,
            transformation_en_ondelette BLOB,
            la_transformee_de_laplace BLOB,
            la_dct_pour_la_compression BLOB,
            l_analyse_de_la_densite_spectrale_de_puissance BLOB,
            FOREIGN KEY (id_hote) REFERENCES hote_data (id)
        )
                ''')

    # Validation des modifications
    conn.commit()

    # Fermeture de la connexion
    conn.close()
    print("Base de données 'local_db' créée avec succès.")
    return "local_db.db"


# Exécution de la fonction pour créer la base de données
create_database()
```

```python
#V3
import sqlite3

def create_database():
    # Connexion à la base de données SQLite (création si elle n'existe pas)
    conn = sqlite3.connect("local_db.db")
    cursor = conn.cursor()

    # Création de la table hote_data
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS hote_data (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            path TEXT NOT NULL,
            type TEXT NOT NULL,
            methode TEXT NOT NULL,
            taille REAL NOT NULL,
            note TEXT
        )
    ''')

    # Création de la table hide_data
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS hide_data (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            content TEXT NOT NULL,
            type TEXT NOT NULL CHECK(type = 'texte'),
            id_hoteData INTEGER NOT NULL,
            taille REAL NOT NULL,
            note TEXT,
            FOREIGN KEY (id_hoteData) REFERENCES hote_data (id)
        )
    ''')

    # Création de la table cover_data
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS cover_data (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            path TEXT NOT NULL,
            type TEXT NOT NULL,
            id_hideDat INTEGER NOT NULL,
            FOREIGN KEY (id_hideDat) REFERENCES hide_data (id)
        )
    ''')

    # Création de la table data
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS data (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            id_coverData INTEGER NOT NULL,
            entropie_des_donnees_cachees REAL,
            variance_des_donnees_porteuses REAL,
            resistance_a_la_compression REAL,
            resistance_au_bruit REAL,
            FOREIGN KEY (id_coverData) REFERENCES cover_data (id)
        )
    ''')
```

```
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS data_without_transformation (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            id_hote INTEGER NOT NULL,
            entropie_des_donnees_cachees REAL,
            variance_des_donnees_porteuses REAL,
            resistance_a_la_compression REAL,
            resistance_au_bruit REAL,
            FOREIGN KEY (id_hote) REFERENCES hote_data (id)
        )
                    ''')

    # Validation des modifications
    conn.commit()

    # Fermeture de la connexion
    conn.close()
    print("Base de données 'local_db' créée avec succès.")
    return "local_db.db"


# Exécution de la fonction pour créer la base de données
create_database()
```

Note a moi même : récuper juste des donnée sur le net et impose leur le traitement et bim

3 Source

3.1 Image :

- https://www.kaggle.com/datasets/arnaud58

3.2 Son

- https://www.kaggle.com/datasets/vjcalling/speaker-recognition-audio-dataset