

**Technique de compilation**

**Mini Projet Flex et Bison :**

***Mise en œuvre d'un***

***Interpréteur SQL***

***Pour l'exécution de***

***Requêtes***

### *Etudiants :* Najjar Mariem Badraoui Roua Yahya Adem Yacoub Rami

*Dirigé par :*

*Mme.Hayet MOGAADI*

Année Universitaire : 2021-2022

**Table des matières**

1. **Introduction**
2. **Etude de cahier de charge**
3. **Développement d’un analyseur lexical des requêtes SQL**
4. **L’analyseur syntaxique (Bison)**
5. **Développement d’un analyseur syntaxique des requêtes SQL**
6. **L’outil Bison**
7. **Analyseur sémantique**
8. **[Tests](#_bookmark10) et validation**
9. **Conclusion**

**Table des figures**

Figure 1.1 – Les phases de compilation

Figure 1.2 – Fichier.l

Figure 1.3 – Generation fichier flex

Figure 1.4 – Liste de token

Figure 1.6 – grammaire de sql

Figure 1.7 – grammaire de sql

Figure 1.8 – génération fichier bison

Figure 1.9 – Fichiers Génères

Figure 1.10 – Test

Figure 1.11 – Resultas

## Introduction :

C’est quoi un Compilateur ? Un compilateur est un programme qui traite les instructions écrites dans un langage de programmation donné pour les traduire en langage machine, ou code, utilisé par le processeur ordinateur. Il contient différentes phases, chaque phase transforme le programme source d’une représentation à une autre. Chaque phase prend les entrées de son étape précédente et transmet sa sortie a la phase suivante du compilateur.

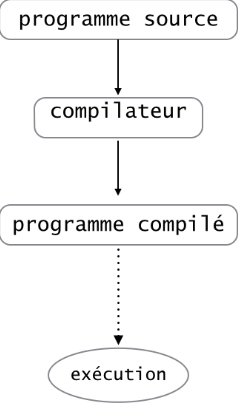


Figure 1.1 – Les phases de compilation

## Cahier des charges : Interpréteur Flex/Bison pour requêtes de

## Manipulation et d'accès à une base de données

* 1. **Introduction**

Le présent cahier des charges décrit les exigences d'un interpréteur Flex/Bison pour accepter des requêtes de manipulation et d'accès à une base de données.L’ Interpreter doit être capable de traiter les requêtes suivantes :

* Requêtes de type CREATE/DELETE/UPDATE
* Requêtes de type SELECT

1. **Specifications fonctionnelles**

### **Fonctionnalités principales**

* L'interpréteur doit accepter des requêtes utilisateur saisies manuellement.
* L'interpréteur doit analyser les requêtes utilisateur en utilisant les outils Flex et Bison.
* L'interpréteur doit identifier le type de requête (CREATE, DELETE, UPDATE ou SELECT).
* L'interpréteur doit extraire les informations pertinentes de la requête, telles que le nom de la table, les noms des champs et les valeurs des données.
* L'interpréteur doit exécuter la requête sur la base de données en utilisant une interface appropriée (par exemple, JDBC, ODBC).
* L'interpréteur doit afficher les résultats de la requête à l'utilisateur.

### **Gestion des erreurs**

* L'interpréteur doit détecter les erreurs syntaxiques dans les requêtes utilisateur et afficher des messages d'erreur appropriés.
* L'interpréteur doit gérer les erreurs d'exécution, telles que les erreurs de base de données ou les erreurs de logique métier.

1. **Specifications techniques**

### **Outils de développement**

* Flex et Bison doivent être utilisés pour développer l'analyseur lexical et l'analyseur syntaxique de l'interpréteur.
* Un langage de programmation approprié (par exemple, C, C++) doit être utilisé pour développer le reste du code de l'interpréteur.
* Une interface de base de données appropriée (par exemple, JDBC, ODBC) doit être utilisée pour interagir avec la base de données.

### **Environnement d'exécution**

* L'interpréteur doit fonctionner sur un système d'exploitation standard (par exemple, Windows, Linux, MacOs).
* L'interpréteur doit pouvoir se connecter à une base de données standard (par exemple, MySQL, PostgreSQL, Oracle).

1. **Tests et validation**

* L'interpréteur doit être testé avec un ensemble de requêtes représentatives couvrant toutes les fonctionnalités requises.
* Les tests doivent inclure des cas de test positifs et négatifs pour vérifier le comportement correct de l'interpréteur.
* La documentation de l'interpréteur doit inclure des instructions d'installation, d'utilisation et de dépannage.

## Développement d’un analyseur lexical des requêtes SQL

## ****Introduction****

Ce rapport présente la conception et le développement d'un analyseur lexical pour les requêtes SQL. L'analyseur lexical est un composant essentiel d'un compilateur ou d'un interpréteur SQL, responsable de l'identification et de la classification des jetons individuels dans une requête SQL.

**Conception de l'analyseur lexical**

L'analyseur lexical est implémenté à l'aide de l'outil Flex (Fast Lexical Analyzer Generator). Flex est un outil puissant et flexible pour la création d'analyseurs lexicaux. Il utilise des expressions régulières pour définir les jetons et génère du code C pour l'analyse lexicale.

**Définition des jetons**

Les jetons identifiés par l'analyseur lexical incluent :

* **Mots-clés SQL:** SELECT, FROM, WHERE, JOIN, INSERT, UPDATE, DELETE, CREATE, DROP, TABLE, etc.
* **Identificateurs:** Noms de tables, noms de colonnes, noms d'alias.
* **Littéraux:** Chaînes de caractères, nombres entiers, nombres réels, valeurs booléennes.
* **Opérateurs:** Opérateurs arithmétiques (+, -, \*, /), opérateurs de comparaison (=, <, >, <=, >=, !=), opérateurs logiques (AND, OR, NOT).
* **Symboles de ponctuation:** Parenthèses (), accolades {}, virgules (,), points-virgules (;).

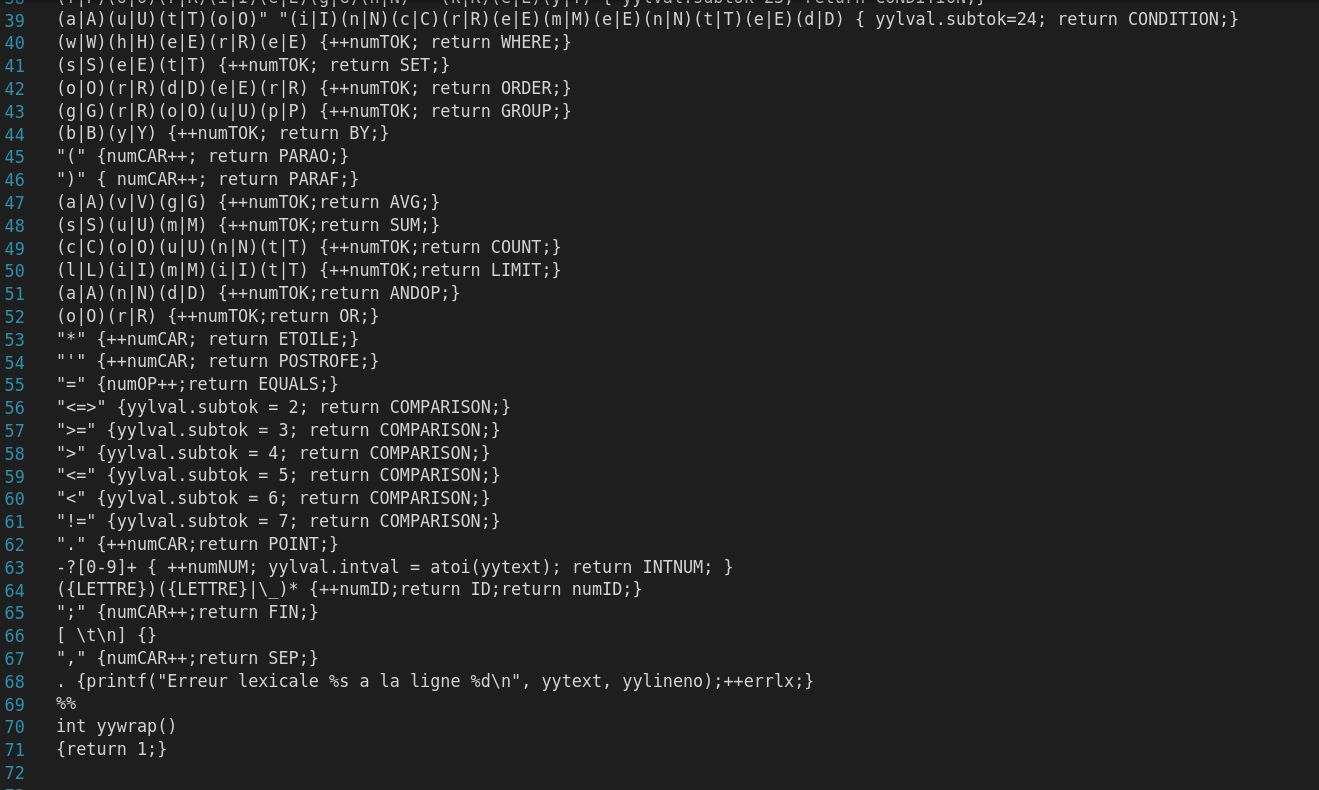
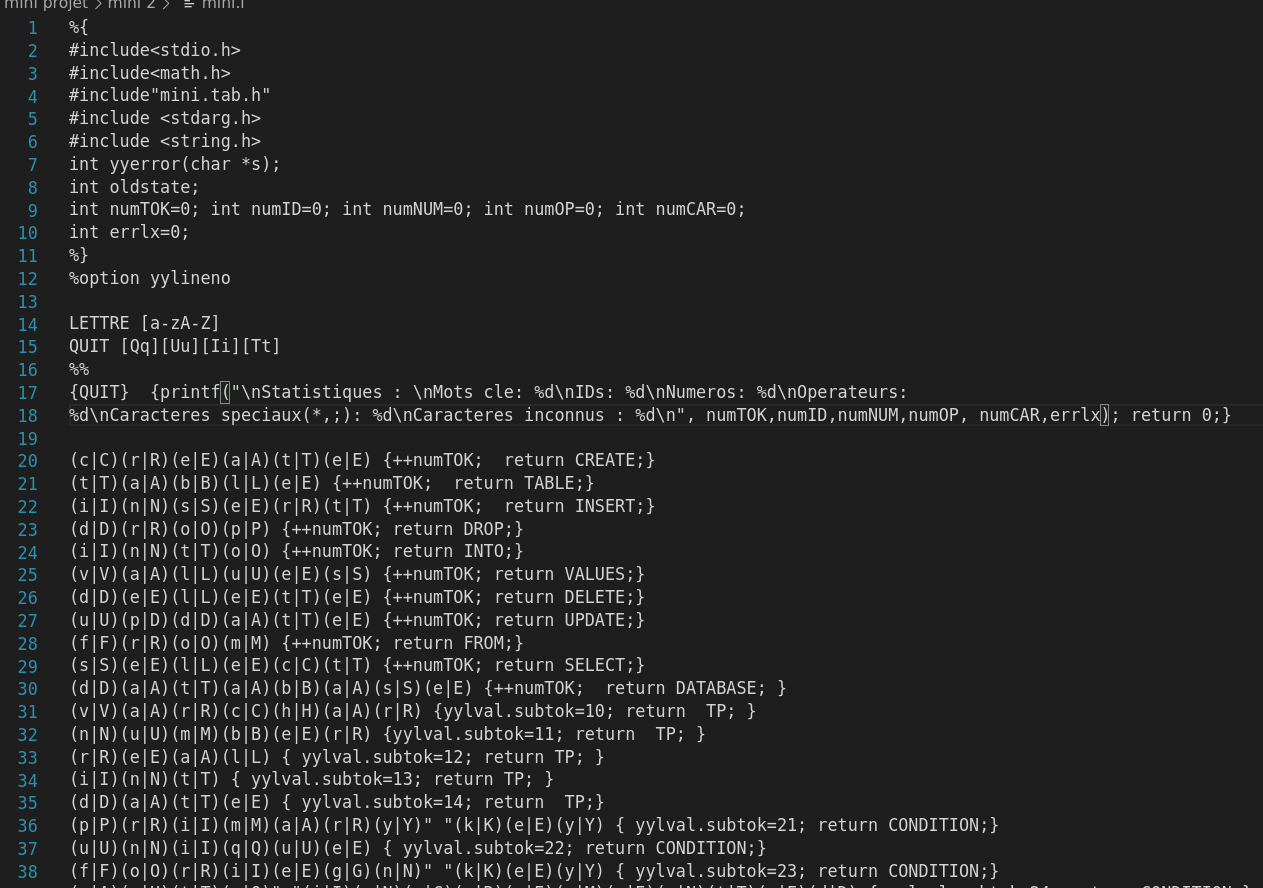


Figure 1.2 – Fichier.l

**4. Implémentation de l'analyseur lexical**

L'analyseur lexical est implémenté à l'aide d'un fichier Flex nommé **mini.l .** Ce fichier contient des expressions régulières pour définir chaque type de jeton. Le code C généré par Flex est ensuite inclus dans le programme principal de l'analyseur lexical.



## 438326892_1893582401087858_1264886506518162043_n

Figure 1.3 – Generation fichier flex

## Analyseur santaxique:

L'analyseur syntaxique, également connu sous le nom d'analyseur syntaxique ou parseur, est un composant crucial d'un compilateur ou d'un interpréteur SQL. Il joue un rôle essentiel dans la transformation d'une requête SQL en une structure arborescente représentant sa syntaxe et sa sémantique.

**Fonctionnement de l'analyseur syntaxique**

L'analyseur syntaxique reçoit un flux de jetons provenant de l'analyseur lexical et les analyse en suivant les règles grammaticales définies pour le langage SQL. Il utilise ces règles pour construire une arborescence d'analyse syntaxique, également appelée arbre de syntaxe abstrait (AST), qui représente la structure hiérarchique de la requête.

L'AST capture les relations entre les différents composants de la requête, tels que les mots-clés, les noms de table, les noms de colonne, les expressions et les opérateurs. Il permet à l'analyseur de vérifier la validité syntaxique de la requête et d'identifier les éventuelles erreurs grammaticales ou structurales.

**Rôle de l'analyseur syntaxique**

En plus de la vérification syntaxique, l'analyseur syntaxique joue un rôle important dans la préparation de la requête pour les étapes ultérieures de la compilation ou de l'interprétation. Il peut effectuer des tâches telles que :

* **Résolution de portée:** Déterminer la portée des identificateurs (noms de table, noms de colonne) et vérifier leur existence.
* **Analyse des types:** Vérifier la compatibilité des types de données dans les expressions et les opérations.
* **Vérification des contraintes:** Vérifier si les valeurs des données respectent les contraintes définies sur les tables et les colonnes.
* **Préparation des requêtes:** Préparer la requête pour l'exécution en générant un code intermédiaire ou en construisant un plan d'exécution.

**Outils pour la construction de l'analyseur syntaxique**

Plusieurs outils peuvent être utilisés pour construire l'analyseur syntaxique d'un compilateur SQL. Parmi les plus populaires, on trouve :

* **Bison (Berkeley Syntax Analyzer Generator):** Un outil puissant et flexible pour la génération d'analyseurs syntaxiques à partir de descriptions formelles de grammaire.
* **ANTLR (Another Tool for Language Recognition):** Un outil populaire pour la construction de langages de programmation et d'analyseurs syntaxiques, offrant une grande flexibilité et des fonctionnalités avancées.
* **LL Parsers:** Des analyseurs syntaxiques basés sur l'analyse LL (Left-to-Left) sont utilisés pour des langages dont la grammaire est sans ambiguïté.

## Développement d'un analyseur syntaxique des requêtes SQL

Un analyseur syntaxique SQL joue un rôle crucial dans un système de gestion de base de données (SGBD) en vérifiant la validité syntaxique des requêtes SQL soumises par les utilisateurs. Il garantit que les requêtes respectent les règles grammaticales du langage SQL et qu'elles sont structurées de manière cohérente.

**Étapes clés du développement d'un analyseur syntaxique SQL :**

**Compréhension de la grammaire SQL:** Avant de commencer à construire l'analyseur syntaxique, il est essentiel de comprendre en profondeur la grammaire formelle du langage SQL. Cela inclut la connaissance des structures de requêtes, des mots-clés, des opérateurs, des fonctions et des règles de composition des expressions.

**Définition de la grammaire YACC:** La grammaire SQL est décrite à l'aide du langage YACC (Yet Another Compiler Compiler), qui est un outil formel pour la spécification de grammaires contextuelles sans ambiguïté. Le fichier YACC définit les règles de production qui décrivent comment les différents composants d'une requête SQL peuvent être combinés.

**Génération de l'analyseur syntaxique avec Bison:** Le fichier YACC est ensuite traité par l'outil Bison (Berkeley Syntax Analyzer Generator) pour générer un analyseur syntaxique spécifique au langage SQL. Bison produit du code C qui implémente les règles de grammaire définies dans le fichier YACC.

**Intégration avec l'analyseur lexical:** L'analyseur syntaxique fonctionne en étroite collaboration avec l'analyseur lexical, qui est responsable de la reconnaissance des jetons individuels dans une requête SQL. L'analyseur lexical fournit un flux de jetons à l'analyseur syntaxique, qui les analyse et les structure selon les règles de grammaire.



**Vérification de la syntaxe et détection des erreurs:** L'analyseur syntaxique analyse le flux de jetons, vérifiant si la requête respecte les règles grammaticales du langage SQL. En cas d'erreurs syntaxiques, l'analyseur syntaxique génère des messages d'erreur pertinents pour aider l'utilisateur à identifier et corriger les problèmes.

## Outil Bison : Bison, ou **Berkeley Syntax Analyzer Generator**, est un outil puissant et flexible pour la génération d'analyseurs syntaxiques à partir de descriptions formelles de grammaire. Il est largement utilisé dans la construction de compilateurs, d'interprètes et d'outils de traitement du langage.

**Fonctionnalités principales de Bison :**

* **Génération d'analyseurs syntaxiques à partir de descriptions formelles de grammaire en langage YACC (Yet Another Compiler Compiler).**
* **Prise en charge des grammaires LR(1), SLR(1) et LALR(1).**
* **Gestion des erreurs syntaxiques et génération de messages d'erreur pertinents.**
* **Production de code C ou C++ optimisé pour l'analyse syntaxique.**
* **Intégration avec les analyseurs lexicaux générés par Flex (Fast Lexical Analyzer Generator).**

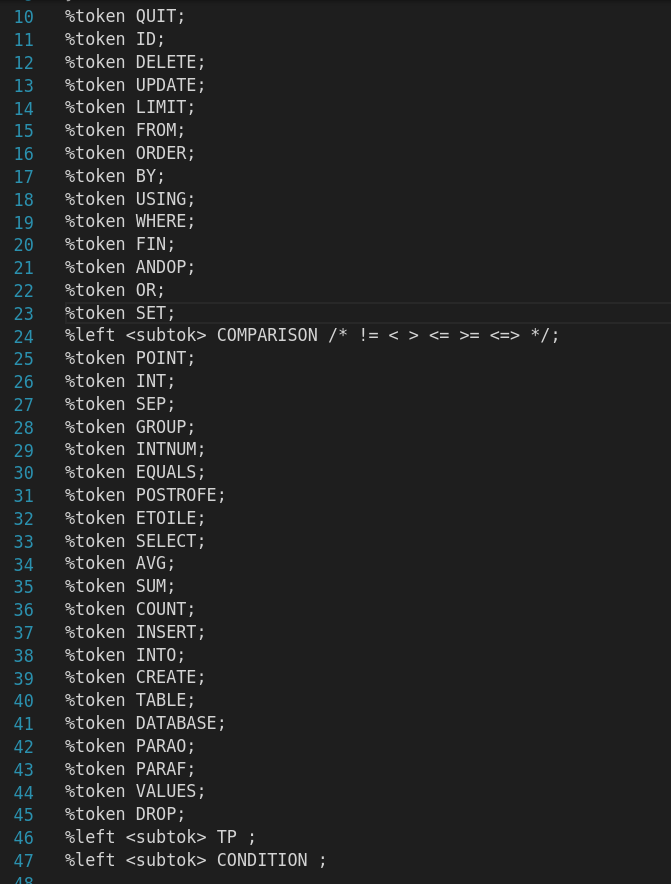


Figure 1.4 – Liste de token

**Les différents grammaire :**

****

Figure 1.5 – grammaire de sql

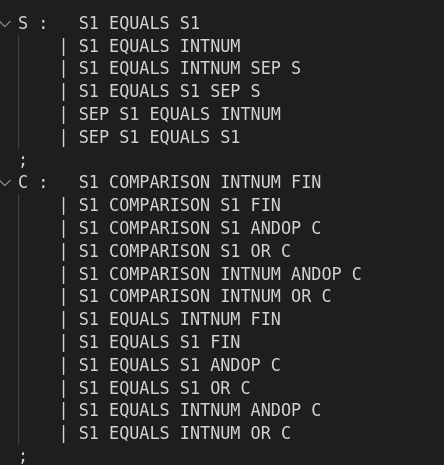
****

Figure 1.6 – grammaire de where

**Avantages de l'utilisation de Bison :**

* **Simplicité et expressivité:** Bison permet de décrire des grammaires complexes de manière claire et concise en utilisant le langage YACC.
* **Puissance et flexibilité:** Bison peut gérer des grammaires contextuelles sensibles et générer des analyseurs syntaxiques efficaces pour une large gamme de langages.
* **Portabilité:** Le code C ou C++ généré par Bison peut être facilement compilé et exécuté sur différentes plateformes.
* **Outils et ressources disponibles:** De nombreux outils et ressources sont disponibles pour faciliter l'apprentissage et l'utilisation de Bison, tels que des tutoriels, des manuels de référence et des exemples de code.



Ou



435131533_324190277362064_6383849180203724356_n

Figure 1.7 – génération de bison

## ****Analyseur sémantique :****

**Mise en œuvre de l'analyse sémantique pour un analyseur syntaxique SQL:**

**Construction d'une table des symboles:** Au cours de l'analyse syntaxique, l'analyseur sémantique construit une table des symboles qui stocke des informations sur les entités nommées dans la requête, telles que les noms de table, de colonne et d'alias.

**Vérification des types de données:** L'analyseur sémantique parcourt l'arbre d'analyse syntaxique généré par l'analyseur syntaxique et vérifie les types de données des expressions et des opérations. Il utilise la table des symboles pour résoudre les références aux colonnes et aux tables.

**Application des contraintes d'intégrité:** L'analyseur sémantique vérifie que les requêtes respectent les contraintes d'intégrité définies sur les tables de la base de données. Cela implique de consulter les métadonnées de la base de données pour connaître les contraintes en vigueur.

**Résolution des ambiguïtés:** L'analyseur sémantique utilise des techniques d'analyse sémantique pour identifier les requêtes ambiguës et lever les ambiguïtés si possible. Cela peut impliquer de demander des informations supplémentaires à l'utilisateur ou de fournir des avertissements sur les requêtes potentiellement problématiques.

**Génération d'informations d'optimisation:** L'analyseur sémantique peut générer des informations sur la structure et le sens des requêtes SQL qui peuvent être utilisées par l'optimiseur de requêtes pour améliorer les performances d'exécution.

**L'analyse sémantique joue un rôle crucial dans la création d'un analyseur syntaxique SQL robuste et fiable.** En garantissant la correction sémantique des requêtes SQL, l'analyseur sémantique contribue à la qualité et à l'intégrité des données dans une base de données.

## Test et Validation

Maintenant, nous nous intéressons à une phase de test. En fait, nous essayons plusieurs cas de test afin de s’assurer que le programme fonctionne correctement et obéit aux spécifications fonctionnelles et aux actions sémantiques qu’on a ajoutées.

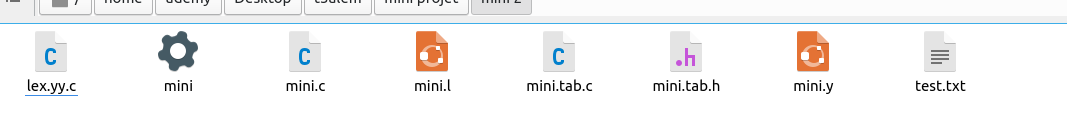
436839502_1139367877067296_3516773993715945110_n

Figure 1.8 – Fichiers Génères

## Test

## 437027591_1132956617743699_310415646614512800_n

Figure 1.9– Test

## Optimisation :

## Fichier txt de test

## 437027599_1120706609166869_5875427718141090685_n

Figure 1.10 – Données d’entrées

## Resultat final

## 437022729_2464368967284506_5014272550785207909_n

Figure 1.11 – Resultat

## Conclusion :

Ce projet présente la conception et le développement d'un analyseur lexical et syntaxique pour les requêtes SQL, des composants cruciaux dans la manipulation et l'interprétation de ces requêtes. L'analyseur lexical, créé avec Flex, identifie et classe les jetons tels que les mots-clés, les identificateurs et les opérateurs, tandis que l'analyseur syntaxique, développé avec Bison, vérifie la validité structurelle des requêtes selon les règles grammaticales du langage SQL.

L'approche méthodique du rapport comprend la définition des jetons, l'implémentation des analyseurs, et l'explication du fonctionnement et du rôle de chaque composant dans le processus d'analyse des requêtes SQL. Les outils tels que Flex et Bison offrent puissance, flexibilité et facilité d'utilisation pour la création d'analyseurs efficaces.

La phase de test et de validation garantit la fiabilité et la conformité du programme aux spécifications fonctionnelles et sémantiques. En incluant des cas de test positifs et négatifs, l'interpréteur est rigoureusement évalué pour assurer son bon fonctionnement dans divers scénarios d'utilisation.