

BURKINA FASO

-------------

Unité – Progrès -Justice

**Institut Supérieur des Sciences de la Population**

Licence Professionnelle en Analyse Statistique 2è année



**Programmation web pyt****hon**

\*\*\*\*\*\*\*\*\*\*\* **PARTIE THEORIQUE**\*\*\*\*\*\*\*\*

**Groupe 3 Professeur**

**GAMENE Yann Arnaud Dr. Ousmane BARA**

**SAWADOGO Orianne-Aurele**

**YAMEOGO Saidou**

**Table des matières**

**Introduction**

[**I)** **Définition et description générale d’un framework web** 4](#_Toc197623251)

[**II)** **Présentation et comparaison des framework web en python** 5](#_Toc197623252)

[**II.I)** **Les frameworks complets** 5](#_Toc197623253)

**1) Django**……………………………………………………………………….6

**2) TurboGears**………………………………………………………………..7

**3) Pyramid**…………………………………………………………………...7

[**II.II)** **Les microframeworks web** 8](#_Toc197623254)

[**1)** **Flask** 8](#_Toc197623255)

[**2)** **Bottle** 9](#_Toc197623256)

[**3)** **Dash** 9](#_Toc197623257)

[**4)** **Les avantages et les limites des microframework** 10](#_Toc197623258)

[**II.III)** **Les frameworks Python asynchrones** 11](#_Toc197623259)

[**1)** **Tornado** 11](#_Toc197623260)

[**2)** **Sanic** 12](#_Toc197623261)

[**3)** **CherryPy** 12](#_Toc197623262)

[**4)** **Les avantages et les limites des framework Python asynchrones** 13](#_Toc197623263)

[**I)** **Base de données relationnelles** 14](#_Toc197623264)

[**II)** **Fonctionnement des SGBDR (Systèmes de Gestion de Bases de Données Relationnelles)** 15](#_Toc197623265)

[**III)** **Etude de quelques SGBD compatible avec python** 18](#_Toc197623266)

[**III.I)** **Le module sqlite3 en Python** 18](#_Toc197623267)

[**III.II)** **MySQL** 19](#_Toc197623268)

[**III.III)** **PostgreSQL** 20](#_Toc197623269)

[**III.IV)** **MongoDB** 20](#_Toc197623270)

[**IV)** **Comparaison des SBGD** 21](#_Toc197623271)

**Conclusion**

**Introduction**

Python est l'un des langages informatiques les plus utilisés ces dernières années. C'est un outil très apprécié par les développeurs web ainsi que les grandes entreprises comme Instagram ou Google qui optent pour des frameworks Python. La principale raison de cet engouement est que Python propose de nombreuses fonctionnalités à travers ces divers frameworks fullstack ou non fullstack. Ses solutions permettent aux équipes web de finaliser plus rapidement leur produit tout en économisant du temps et de l'argent. Ainsi, ce rapport : Projet Python a pour but d’explorer les fonctionnalités de ce logiciel.

Le projet Python est une forme d’examen soumis aux étudiants de la deuxième année d’analyse statistique de l’ISSP. Ce projet est donné pour un délai de 14 jours et est réalisé par des groupes composés de trois étudiants chacun. Il a pour objectif d’évaluer les compétences de chaque étudiant en ce qui concerne l’assimilation et la compréhension du logiciel Python ainsi que la maîtrise de l’utilisation des composantes qui y sont intégrées. Chaque groupe devra faire une partie théorique qui sera constituée de l’étude des framework web python ainsi que les bases de données compatibles avec ce logiciel. En ce qui concerne la phase pratique, il s’agira de concevoir une application web de gestion de livres à travers l’utilisation du framework web Django.

**PREMIERE PARTIE : ETUDE COMPARATIVE DES FRAMEWORK WEB EN PYTHON**

# **Définition et description générale d’un framework web**

En informatique, le framework web désigne une infrastructure de développement web**.** Il s’agitd’unensemble d'outils associés aux langages de programmation informatique qui permettent la plupart du temps de simplifier le travail des développeurs web. Les frameworks web sont ainsi caractérisés par des codes réutilisables, une programmation standardisée et une architecture formalisée (gestion des requêtes http, génération de pages, sécurité web…) et adaptée aux besoins de chaque entreprise. Ce sont des outils qui sont souvent améliorés en tenant compte de l'expérience de chaque version développée avant.

Un framework web Python est donc un environnement de travail qui regroupe l’ensemble des outils associés à ce langage de programmation. Les frameworks web Python possèdent des caractéristiques qui leur sont propres. Ils sont classés en trois grandes catégories :

* **Les frameworks complets**
* **Les micro frameworks**
* **Les frameworks Python asynchrones**

# **Présentation et comparaison des framework web en python**

## **Les frameworks complets**

Ce sont des solutions tout-en-un avec des bibliothèques configurées pour travailler ensemble de manière pertinente. Ils fournissent une solution complète pour le développement web incluant gestion de bases de données (ORM), validation des formulaires, système d’authentification, interface d’administration, etc.

* + - 1. **Django**

Django figure parmi les meilleurs et populaires frameworks web de Python. Il offre de nombreuses fonctionnalités hors de la boîte comme le panneau d'administration ou les vues et formes génériques.

Sa force réside dans sa polyvalence et sa flexibilité. Il peut être utilisé aussi bien pour des petits projets que pour des travaux complexes et chronophages tels que la création de MVP (Minimum Viable Product), permettant aux start-ups d’optimiser leur temps et leur budget.

**** Ses caractéristiques en font un framework Python polyvalent, rapide et très évolutif :

* **Le système d'authentification**
* **Un routage d'URL puissant**
* **L'ORM (Object Relationnal Mapper)**
* **Le moteur de modèle**
* **La migration de schémas de bases de données**
* **Etc.**

Contrairement aux autres, ce framework web Python réunit toutes les fonctionnalités nécessaires par défaut au même endroit, au lieu de les proposer en bibliothèques séparées. Son ORM aide notamment à mapper des objets sur les tables de base de données. Plusieurs bases de données peuvent fonctionner avec ce même code, qui se transfère facilement d'une base à l'autre.

* + - 1. **TurboGears**

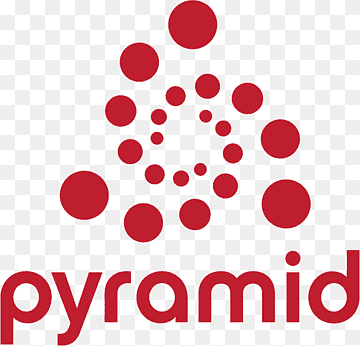
****TurboGears est un framework web d’application**,** construit sur de nombreux intergiciels et de nombreuses bibliothèques et a été initialement conçu pour combiner les meilleurs composants des autres frameworks Python.  
Concrètement, TurboGears permet aux développeurs de concevoir rapidement des applications web extensibles basées sur des données. Celles-ci sont fournies depuis un moteur de modélisation intuitif et ergonomique, renforcé par l’ORM souple et performant du framework.

Ses caractéristiques sont :

* **Une architecture de type MVC**
* **La validation avec FormEncode**
* **La prise en charge des bases de données multiples**
* **L’intégration des Genshi et Kid dans les langages de référence préférés**
* **Le soutien à SQLAlchemy et SQLObject**
* **Les templates PasteScript**
* **La bibliothèque d’applications Tosca Widgets, qui simplifie la coordination de la conception du front-end et du développement des serveurs**
* **Etc.**
  + - 1. **Pyramid**

Pyramid est une application web basée sur Python, en open-source. Son but est de faire le plus possible avec un minimum de complexité. Fonctionnant sous Python 3, Pyramid suit les progrès technologiques.

La caractéristique la plus frappante de Pyramid est sa capacité à bien fonctionner avec de petites et grandes applications.

****Ses autres caractéristiques sont :

* **Applications à fichier unique**
* **Génération d’URL**
* **Configuration extensible**
* **Modèles globaux et spécifications des actifs**
* **Authentification et autorisation souples**
* **Test, soutien et documentation complète sur les données**
* **Prédictions de vue et nombreuses vues par itinéraire**
* **Prédicats**
* **Renderers**
  + - 1. **Les avantages et les limites des framework complets**
* Avantages

De façon générale, les framework complets possèdent les avantages suivants :

* Productivité accrue grâce à des fonctionnalités prêtes à l’emploi
* Documentation complète et écosystème mature (plugins, communauté)
* Adaptés aux projets complexes (sites e-commerce, CMS)
* Limites
* Moins flexibles (structure imposée)
* Surcharge inutile pour les petits projets ou API simples

## **Les microframeworks web**

Un microframework web sert à fournir uniquement l’ensemble des composants nécessaires à la construction d’une application web. Il peut également se concentrer sur la fourniture des fonctionnalités nécessaires pour une sphère particulière.

### **Flask**

Disponible sous la licence BSD, Flask est un framemork web Python qui convient aux petits projets et qui s'appuie sur la boite à outils WSGI (Web Server Gateway Interface) de Werkzeug et sur le modèle de template Jinja2. L'objectif de Flask est de participer à la construction d'une base solide d'applications web. Flask permet de travailler sur tous les projets lorsque les extensions adaptées sont installées. C'est un peu la solution par défaut pour un projet qui ne correspond pas à Django.

* **Ce framework Python est très apprécié en raison de ses caractéristiques intéressantes :**
* **La conformité à la norme WSGI 1.0 ;**
* **La base Unicode ;**
* **L’envoi de la demande RESTful ;**
* **Le traitement efficace des requêtes HTTP ;**
* **La possibilité de brancher n’importe quel ORM ;**
* **La prise en charge des cookies sécurisés (sessions côté client) ;**

### **Bottle**

Initialement destiné à la construction d’API, Bottle implémente tout dans un seul fichier source. Il n’a pas de dépendances, à part la Python Standard Library. Il ne convient que si vous créez une très petite application avec au maximum 500 lignes de code et sans exigences particulières.

Ses caractéristiques par défaut comprennent :

* **Routage : prend en charge les demandes de mappage d’appel de fonction, permettant d’obtenir des URL propres et dynamiques**
* **Templates : un moteur de templates rapide et prêt à l’emploi, avec prise en charge complète de mako, jinja2 et cheetah.**
* **Utilitaires : accès confortable aux données des formulaires, aux téléchargements de fichiers, aux cookies, aux en-têtes et autres métadonnées liées à HTTP**
* **Serveur : un serveur de développement HTTP intégré qui prend en charge fapws3, bjoern, GAE, СherryPy, et tout autre serveur HTTP compatible WSGI**

### **Dash**

Dash est un framework web basé sur Python particulièrementadapté pour la conception d’applications web analytiques, c’est-à-dire qui se basent sur les données. Il possède à cet effet une bibliothèque bien remplie. Dash s’utilise avec le framework Python Flask, la bibliothèque d’interface utilisateur web React JS et la bibliothèque de traçage JavaScript Plotly.

* **La gestion des erreurs (Dash Deployment Server)**
* **Le haut degré de personnalisation**
* **L’intégration LDAP**
* **La prise en charge des plugins**
* **Le routage d’URL**
* **La facilité de démarrage des applications Dash ; en effet, elles réclament très peu de code passe-partout.**

### **Les avantages et les limites des microframework**

* Avantages
* Flexibilité totale (choix des composants)
* Courbe d’apprentissage douce, idéaux pour les débutants ou prototypes rapides
* Performants pour les petites applications ou API REST simples
* Limites
* Temps de développement accru pour les projets complexes (intégration manuelle des modules)
* Ne possèdent pas la plupart des fonctionnalités d’un framework complet, comme un moteur de templates web, la fonctionnalité d’authentification, des comptes, une validation des entrées et un assainissement des entrées.

## **Les frameworks Python asynchrones**

Il s’agit d’un ensemble d'outils et de bibliothèques permettant de développer des applications web capables de gérer de nombreuses connexions simultanées sans bloquer l'exécution du programme. Ces frameworks utilisent la programmation asynchrone avec la syntaxe async/await pour optimiser la gestion des entrées/sorties (I/O), ce qui améliore la performance et la scalabilité, notamment pour les applications en temps réel ou à forte charge.

### **Tornado**

Tornado est un framework Python asynchrone et non bloquant conçu pourgérer des connexions longues et un grand nombre de requêtes simultanées. Initialement développé par FriendFeed (racheté par Facebook), il est particulièrement adapté aux applications web en temps réel et aux services nécessitant une haute performance.

Tornado se distingue par sa capacité à traiter des milliers de connexions en parallèle grâce à sa boucle d’événements optimisée. Il ne repose pas sur les serveurs WSGI traditionnels, ce qui le rend idéal pour les applications où la latence doit être minimisée.

Ses caractéristiques distinctives sont :

* **Serveur HTTP natif ultra-performant**
* **Programmation asynchrone avec boucle d’événements intégrée**
* **Support natif des WebSockets**
* **Compatible avec les futures Python (async / await)**
* **Parfait pour les applications en temps réel (chat, dashboard, etc.)**
* **Outils intégrés pour la gestion des utilisateurs, cookies, sécurité**
* **Faible latence et grande scalabilité sur les architectures modernes**

### **Sanic**

Sanic est un micro-framework asynchrone conçu pour offrir des performances ultra-rapides, en tirant parti du modèle async / await de Python 3. Il permet de gérer un grand volume de requêtes simultanées tout en conservant une faible latence, ce qui le rend idéal pour les APIs modernes et les applications temps réel.

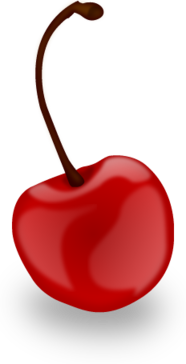
****Inspiré de Flask dans sa syntaxe, Sanic ajoute un support natif pour la programmation non bloquante, ce qui en fait un excellent choix pour des services réactifs à forte charge.

Quelques-unes de ces caractéristiques sont :

* **Framework asynchrone ultra-rapide**
* **Syntaxe proche de Flask avec support async / await**
* **Gestion native des WebSockets**
* **Support des middlewares, blueprints, routing dynamique**
* **Intégration facile avec les outils modernes (ORMs, moteurs de templates)**
* **Documentation claire et active communauté**

### **CherryPy**

CherryPy est un framework disponible en open source. Il s’agit d’un programme orienté objet, mais largement extensible, qui permet à ses utilisateurs de construire des applications web dans le langage Python. CherryPy s’appuie sur un serveur multithread qui augmente le taux de flexibilité du langage en question. Ainsi, une application web qui fonctionne avec ce framework est une application autonome qui intègre son propre serveur multithread.

Ses caractéristiques sont :

* **Un serveur compatible HTTP/1.1 et mutualisé par le fil de discussion WSGI ;**
* **Un système de configuration puissant ;**
* **Un système de plug-in flexible ;**
* **La simplicité avec laquelle on peut exécuter plusieurs serveurs HTTP à la fois ;**
* **La possibilité de fonctionner sur Python 2.7+, Python 3.1+, PyPy, Jython et Android ;**
* **Des outils prêts à l’emploi pour le codage, l’authentification, les sessions, le contenu statique, etc.**

### **Les avantages et les limites des framework Python asynchrones**

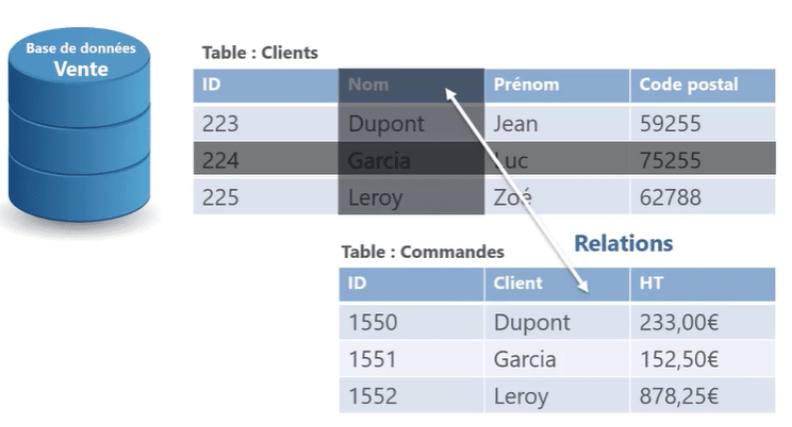
* Avantages
* Scalabilité élevée grâce au traitement concurrentiel
* Latence réduite pour les requêtes simultanées
* Adaptés aux architectures modernes (microservices, WebSockets)
* Limites
* Courbe d’apprentissage plus raide (gestion explicite des coroutines)
* Moins adaptés aux applications nécessitant des fonctionnalités complexes (ORM, authentification) sans extensions

**DEUXIEME PARTIE : ETUDE COMPARATIVE DES BASE DE DONNEES RELATIONNELLE EN PYTHON**

1. **Base de données relationnelles**

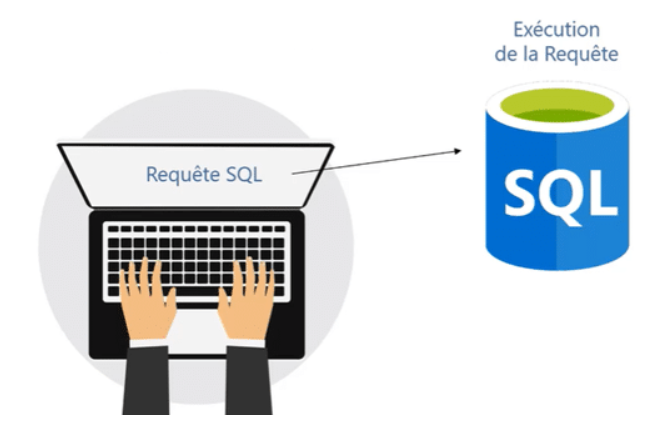
**Qu'est-ce qu'une base de données relationnelle ?**

Une base de données relationnelle est un type de base de données qui stocke les informations sous forme de tables composées de lignes et de colonnes. Elle facilite les relations entre ces tables, par exemple, on peut avoir une table « clients » avec des champs comme « id », « nom », « prénom », et « code postal », et une autre table « commandes » contenant « id », « commande » et « HT ». Pour relier ces deux tables, notamment pour associer les commandes aux clients, on utilise des bases de données relationnelles, et plus précisément ce qu’on appelle les jointures.



1. **Fonctionnement des SGBDR (Systèmes de Gestion de Bases de Données Relationnelles)**

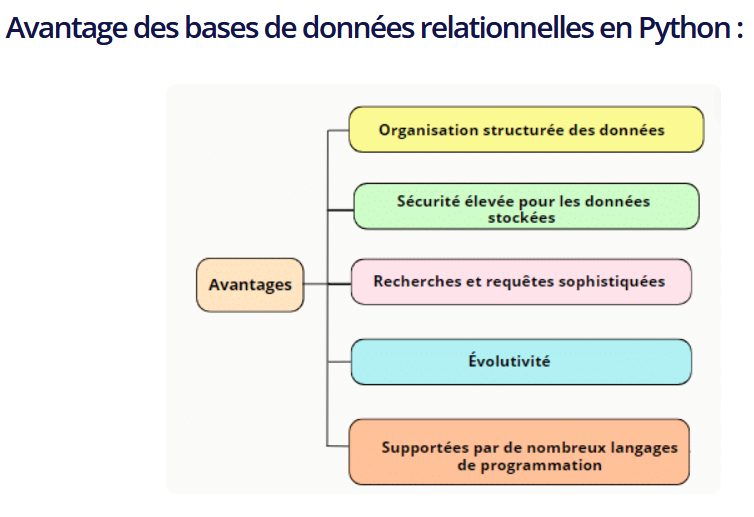
Les **SGBDR** utilisent le langage SQL, qui est un langage de programmation déclaratif. Cela signifie que l’utilisateur décrit ce qu’il souhaite accomplir sans avoir à spécifier comment l’opération doit être réalisée. Par exemple, pour effectuer une requête, l’utilisateur tape la commande SQL sur le clavier, en indiquant l’opération souhaitée (comme sélectionner, insérer ou mettre à jour des données). Cette requête est ensuite envoyée au SGBDR, qui se charge de l’exécuter et de renvoyer les résultats appropriés.



Dans un SGBDR, le fonctionnement est divisé en plusieurs étapes clés :

* **Écriture de la Requête :** L’utilisateur écrit une requête SQL pour définir l’opération à effectuer sur la base de données, comme récupérer des données, ajouter des enregistrements, ou modifier des informations existantes.
* **Envoi de la Requête :** La requête est envoyée au moteur du SGBDR, qui interprète le code SQL.
* **Exécution par le SGBDR :** Le moteur du SGBDR analyse et optimise la requête, puis l’exécute en suivant le meilleur chemin d’accès aux données. Le SGBDR s’occupe de tous les aspects complexes de l’exécution, tels que la gestion des index, l’accès aux données, et la maintenance de l’intégrité référentielle.
* **Résultat :** Le résultat de la requête est retourné à l’utilisateur, affichant les données demandées ou confirmant la modification ou insertion effectuée.

Ce fonctionnement en mode déclaratif simplifie la gestion des bases de données, car l’utilisateur n’a pas besoin de connaître les détails techniques sur la façon dont les données sont manipulées en interne. Le SGBDR prend en charge ces détails, optimisant l’exécution des requêtes pour assurer des performances efficaces et une utilisation cohérente des ressources système.



**En général les SGBD sont compatibles avec de nombreux langages de programmation** :

Les bases de données relationnelles sont compatibles avec une large gamme de langages de programmation (comme [Python](https://blog.alphorm.com/apprendre-python-les-bases-essentielles), Java, C#, etc.), permettant aux développeurs d’intégrer facilement des fonctionnalités de gestion de données dans leurs applications.

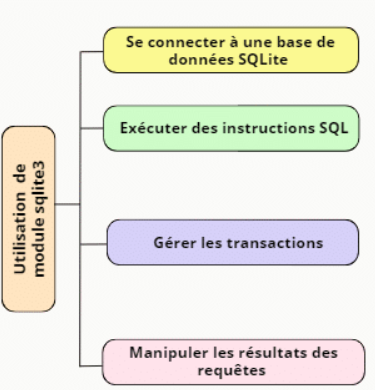
Dans la suite nous présenterons quelques SGBD compatibles avec le langage de programmation python

1. **Etude de quelques SGBD compatible avec python**
2. **Le module sqlite3 en Python**

* Qu'est-ce que le module sqlite3 en Python ?

Le module sqlite3 est une bibliothèque intégrée dans Python qui permet de manipuler des bases de données **SQLite** directement à partir de votre code Python. Il fournit une interface simple et cohérente pour interagir avec des bases de données **SQLite**, offrant des fonctionnalités telles que la création, la modification, et la gestion de bases de données, ainsi que l’exécution de requêtes SQL.

Le module sqlite3 en Python permet d’interagir facilement avec les bases de données SQLite. Voici ce qu’il vous permet de faire :

****

* **Se connecter à une base de données SQLite** : Créer une nouvelle base de données ou se connecter à une base existante.
* **Exécuter des instructions SQL** : Inclure des commandes pour créer des tables, insérer des données, interroger et mettre à jour les données.
* **Gérer les transactions** : Commencer, valider, et annuler des transactions.
* **Manipuler les résultats des requêtes** : Accéder aux résultats des requêtes SQL de manière structurée.
* Avantages

Il est particulièrement utile pour les applications locales et les projets qui nécessitent une gestion de base de données simple sans avoir besoin d’un serveur de base de données complexe. A cela s’ajoute la facilité d’utilisation simple et la configuration préétablie.

* Aucune installation requise (intégré dans Python via sqlite3).
* Fichier unique .db, très léger.
* Parfait pour les tests unitaires ou les prototypes.
* Limites
* Ne supporte pas bien le multi-utilisateur ou la forte concurrence.
* Fonctionnalités avancées (transactions complexes, sécurité fine) limitées

1. **MySQL**

**MySQL est un Système de Gestion de Base de Données relationnel (SGBDR), open source, très populaire pour stocker et gérer des données sous forme de tables relationnelles. Son utilisation en python nécessite l’installation du module mysql-connector-python. Il présente quelques avantages :**

* Avantages
  + **Très bonne performance en lecture/écriture.**
  + **Large écosystème et documentation.**
  + **Compatible avec de nombreux outils (PHP, Django, Flask).**
  + **Idéal pour les application de gestion avec un grand nombre d’utilisateurs.**

**Cependant il présente des limites à l’image de :**

* Limites
  + **Fonctionnalités avancées parfois limitées (ex : gestion des contraintes).**
  + **Plusieurs moteurs de stockage (InnoDB, MyISAM) avec des comportements différents.**

1. **PostgreSQL**

**PostgreSQL (souvent abrégé Postgres) est un Système de Gestion de Base de Données relationnel (SGBDR) open source avancé, reconnu pour sa robustesse, sa conformité aux standards SQL et sa richesse fonctionnelle. Son utilisation en python nécessite l’installation du module psycopg2-binary**

* **Utilisation typique : systèmes d’information complexes, analytics, data science.**
* Avantages :
  + **Très robuste, hautement conforme aux standards SQL.**
  + **Prise en charge avancée des types (JSONB, ARRAY, géospatial).**
  + **Transactions ACID très fiables.**
  + **Excellente scalabilité verticale et bonne horizontalement.**
* Limites :
  + **Plus complexe à configurer.**
  + **Moins rapide que MySQL pour de simples lectures.**

1. **MongoDB**

**MongoDB est un Système de Gestion de Base de Données NoSQL orienté documents, conçu pour gérer des données semi-structurées ou non structurées, avec une forte capacité de scalabilité horizontale. Son utilisation en python nécessite l’installation du module mysql-connector-python**

* Avantages :
  + **Stockage flexible (documents JSON-like, BSON).**
  + **Très performant pour les grandes volumétries de données.**
  + **Scalabilité horizontale native.**
  + **Intégration simple avec Python via pymongo.**
  + **Sécurité : gestion fine des rôles, authentification, chiffrement possible.**
  + **Multiplateforme : fonctionne sur Windows, macOS, Linux, et dans le cloud (MongoDB Atlas)**
* Limites :
  + **Pas de jointures complexes nativement (mais possibilités avec lookup).**
  + **Moins rigide sur la structure**
  + **peut générer des incohérences si mal géré.**

1. **Comparaison des SBGD**

**Tableau de comparaison**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Critère | SQLite | MySQL | PostgreSQL | MongoDB |
| Compatibilité avec Python | Excellente (sqlite3, SQLAlchemy) | Excellente (mysql-connector, ORM) | Excellente (psycopg2, SQLAlchemy) | Excellente (pymongo) |
| Type de SGBD | Relationnel léger | Relationnel | Relationnel avancé | Non relationnel (NoSQL, documents) |
| Performance | Très bon pour petits projets | Bon (lecture/écriture rapide) | Très bon (requêtes complexes) | Très bon (flexible, scalable) |
| Facilité d’installation | Très facile (embarqué, sans serveur) | Moyenne (nécessite un serveur) | Moyenne (plus complexe que MySQL) | Moyenne (serveur, configuration JSON) |
| Communauté et support | Bonne | Très bonne | Excellente | Très bonne |
| Sécurité | Basique | Bonne | Excellente (contrôle fin des accès) | Bonne (contrôle par rôles) |
| Support ACID | Oui (limité) | Oui (mais limité sur certaines engines) | Oui (complet) | Partiel (ACID depuis la version 4.0) |
| Gros volumes de données | Non recommandé | Bon | Excellente (meilleure gestion) | Excellente (scalabilité horizontale) |
| Fonctionnalités avancées | Très limité | Moyennes (réplication, partitionnement) | Très riche (index, JSON, géospatial, etc.) | Riche (sharding, réplication, index, etc.) |
| Licence / Coût | Gratuit (Domaine public) | Gratuit (GPL) ou commercial | Gratuit (Licence PostgreSQL) | Gratuit (SSPL), offre cloud payante |

**Conclusion**

L’étude comparative des frameworks web et des systèmes de gestion de bases de données compatibles avec Python met en évidence la richesse de l’écosystème Python pour le développement d’applications web. Chaque outil présente des avantages spécifiques selon le contexte du projet, les besoins techniques et les ressources disponibles.

Du côté des frameworks, **Django** se distingue par sa complétude, son ORM intégré et son architecture « tout-en-un » adaptée aux applications web structurées. **Flask** séduit par sa simplicité et sa souplesse, idéal pour des projets légers ou à forte personnalisation. **Pyramid**, bien que moins populaire, reste un bon choix pour les projets modulaires complexes nécessitant un haut niveau de contrôle.

Concernant les bases de données, **PostgreSQL** ressort comme la solution la plus complète, combinant performance, sécurité, conformité ACID et richesse fonctionnelle. **MySQL** reste une option solide pour des projets web classiques, tandis que **SQLite** convient parfaitement aux projets de petite envergure, grâce à sa facilité d’utilisation. Enfin, **MongoDB**, base NoSQL, s’impose dans des contextes nécessitant flexibilité, scalabilité horizontale et traitement de données semi-structurées.

En définitive, le choix d’un framework web et d’un SGBD doit se faire en fonction des spécificités du projet, du niveau de compétence de l’équipe et des objectifs de performance. La complémentarité des outils Python permet d’assembler des solutions performantes, durables et évolutives.