مقارنة أنظمة قواعد البيانات وربطها مع Django

دراسة تحليلية حول أنظمة قواعد البيانات المختلفة وتوافقها مع إطار العمل Django

إعداد الطالب طارق العمري

إشراف م. مالك المصنف

طارق العمري

١٥ أغسطس ٢٠٢٥

مقدمة عن قواعد البيانات

قواعد البيانات هي مجموعة منظمة من البيانات المهيكلة، المخزنة والمتاحة إلكترونياً في نظام حاسوبي. وهي تمثل العمود الفقري لمعظم التطبيقات الحديثة، حيث تسمح بتخزين كميات كبيرة من المعلومات وتنظيمها واسترجاعها بكفاءة عالية. اختيار نظام قاعدة البيانات المناسب يعد من القرارات الحاسمة في هندسة البرمجيات، حيث يؤثر مباشرة على أداء النظام وقابليته للتوسع والتطوير مستقبلاً.

لا تكليف ٤ - هندسة البرمجيات

تختلف أنظمة قواعد البيانات في كيفية تنظيمها للبيانات وتخزينها ومعالجتها. ويعتمد اختيار النظام المناسب على طبيعة التطبيق واحتياجاته من حيث حجم البيانات، عدد المستخدمين، متطلبات الأمان، والأداء المطلوب.

فهرس المحتويات

عرض تقديمي شامل حول أنظمة قواعد البيانات المختلفة وطرق ربطها مع إطار عمل Django، مع التركيز على التوافق والإعدادات

طارق العمري

١٥ أغسطس ٢٠٢٥

أنواع أنظمة قواعد البيانات

الفروقات الأساسية بين قواعد البيانات العلائقية والغير علائقية ومعايير اختيار النوع المناسب للمشروع

قواعد البيانات العلائقية (SQL)

- تخزن البيانات في جداول وأعمدة وصفوف مع علاقات محددة
- تدعم مبادئ ACID للمعاملات (الذرية، الاتساق، العزل، المتانة)
- مثالية للتطبيقات المالية والأنظمة التي تتطلب تكاملاً في البيانات
- أمثلة: PostgreSQL، MySQL، SQLite، Oracle

قواعد البيانات غير العلائقية (NoSQL)

- تخزن البيانات بأشكال مختلفة (وثائق، أزواج مفتاح-قيمة، أعمدة)
- مرونة أعلى في الهيكل وقابلية للتوسع الأفقي
- مناسبة للبيانات الضخمة والتطبيقات ذات المتطلبات المتغيرة
- أمثلة: MongoDB، Redis، Cassandra

طارق العمري

١٥ أغسطس ٢٠٢٥

أشهر أنظمة قواعد البيانات المدعومة

يدعم إطار عمل Django رسمياً العديد من أنظمة إدارة قواعد البيانات العلائقية مما يوفر مرونة عالية للمطورين في اختيار النظام المناسب لمتطلبات مشاريعهم المختلفة.

PostgreSQL

نظام قوي مفتوح المصدر يدعم ACID بالكامل مع مميزات متقدمة مثل البيانات المكانية والـJSON. يعتبر الخيار المفضل للمشاريع المعقدة والبيانات الضخمة، ويدعمه Django بشكل ممتاز من خلال مكتبة psycopg2.

MySQL

أحد أشهر أنظمة قواعد البيانات العلائقية مفتوحة المصدر. يتميز بسهولة الاستخدام والأداء العالي مع البيانات البسيطة. يدعمه Django بشكل كامل من خلال مكتبة mysqlclient، ويستخدم غالباً في المشاريع متوسطة الحجم.

SQLite

قاعدة بيانات خفيفة مدمجة في ملف واحد. تأتي مع Django بشكل افتراضي، وهي مثالية للتطوير والاختبار والمشاريع الصغيرة، ولكنها محدودة من حيث التزامن والأداء مع البيانات الكبيرة.

أنظمة أخرى

يدعم Django أيضاً Oracle Database وMariaDB. بينما تتوفر حلول طرف ثالث لدعم أنظمة NoSQL مثل MongoDB، لكنها غير مدعومة رسمياً من Django.

طارق العمري

١٥ أغسطس ٢٠٢٥

مقارنة بين PostgreSQL و MySQL و SQLite

مقارنة تفصيلية بين أنظمة قواعد البيانات الشائعة ومدى توافقها مع إطار عمل Django، مع التركيز على المميزات والقدرات الأساسية لكل نظام.

تختلف أنظمة إدارة قواعد البيانات في مميزاتها وأدائها، وفيما يلي مقارنة بين الأنظمة الثلاثة الأكثر استخداماً مع إطار عمل Django.

المعيار	PostgreSQL	MySQL	SQLite
نوع النظام	RDBMS	RDBMS	RDBMS
دعم المعاملات (ACID)	ممتاز	ممتاز (InnoDB)	أساسي
دعم JSON	قوي	جيد	محدود
الأداء مع البيانات الكبيرة	قوي	جيد جداً	محدود
المرونة في الأنواع	عالية	متوسطة	محدودة
التوافق مع Django	مدعوم رسمياً	مدعوم رسمياً	مدعوم افتراضياً

طارق العمري

١٥ أغسطس ٢٠٢٥

دعم Django لمحركات قواعد البيانات

يقدم إطار عمل Django دعماً رسمياً لعدة أنظمة قواعد بيانات، مما يتيح للمطورين المرونة في اختيار نظام قاعدة البيانات المناسب لمشاريعهم. يتم ذلك من خلال طبقة ORM التي توفر تجريداً موحداً فوق قواعد البيانات المختلفة، مما يسهل على المطورين الانتقال بين الأنظمة المختلفة دون تغيير كبير في الكود.

PostgreSQL

مدعوم بشكل كامل من Django، ويُنصح به للمشاريع المتقدمة. يتطلب تثبيت حزمة psycopg2 أو psycopg3، ويتميز بدعم ممتاز للميزات المتقدمة مثل الأنواع الجغرافية والحقول JSON.

MySQL

مدعوم بشكل كامل، ويتطلب تثبيت حزمة mysqlclient. يُعد خياراً شائعاً للمشاريع متوسطة الحجم ويتكامل بسهولة مع Django عبر محرك InnoDB.

SQLite

مدمج افتراضياً مع Django، مما يجعله مثالياً للتطوير والاختبار. لا يحتاج إلى تكوين إضافي ويأتي مثبتاً مسبقاً مع Python، لكنه غير مناسب للإنتاج مع التطبيقات متعددة المستخدمين.

Oracle

مدعوم رسمياً ويتطلب تثبيت حزمة cx_Oracle أو oracledb. يستخدم غالباً في بيئات المؤسسات الكبيرة التي تعتمد على أنظمة Oracle.

معايير اختيار نظام قاعدة البيانات

اختيار نظام قاعدة البيانات المناسب يعتبر قرارًا استراتيجيًا يؤثر على أداء التطبيق ومرونته وقابليته للتطوير مستقبلاً

المعايير المتعلقة بالمشروع

- **التوافق مع Django:** سهولة الدمج وكفاءة التشغيل
- **تعقيد المشروع:** هل المشروع بسيط أم متقدم؟
- **الميزانية:** تكاليف الاستضافة والتراخيص والصيانة
- **خبرة الفريق:** المعرفة السابقة بنظام معين
- **متطلبات المستقبل:** القدرة على التطور مع احتياجات المشروع

المعايير التقنية

- **حجم البيانات:** القدرة على التعامل مع كميات كبيرة من البيانات
- **الأداء:** سرعة الاستجابة وقدرة معالجة الاستعلامات
- **قابلية التوسع:** القدرة على التحجيم عند نمو التطبيق
- **الموثوقية:** ضمان سلامة البيانات وتوافرها
- **الأمان:** حماية البيانات والتحكم بالوصول

طارق العمري

١٥ أغسطس ٢٠٢٥

خطوات ربط Django مع PostgreSQL

يعتبر PostgreSQL من أقوى أنظمة قواعد البيانات المدعومة في Django، وسنتعرف على خطوات الربط العملية وكيفية إعداد المشروع للتعامل معه بكفاءة

تثبيت المكتبة اللازمة

أولاً، يجب تثبيت مكتبة psycopg2 التي تمثل وسيط الاتصال بين Python/Django وقاعدة بيانات PostgreSQL:

```
pip install psycopg2
```

يمكن أيضًا استخدام الإصدار البديل الأحدث مع نفس الواجهة البرمجية:

```
pip install psycopg2-binary
```

إعدادات ملف settings.py

بعد تثبيت المكتبة، يتم تعديل إعدادات قاعدة البيانات في ملف الإعدادات الرئيسي للمشروع:

```
DATABASES = {
'default': {
'ENGINE': 'django.db.backends.postgresql',
'NAME': 'اسم_قاعدة_البيانات',
'USER': 'اسم_المستخدم',
'PASSWORD': 'كلمة_المرور',
'HOST': 'localhost',
'PORT': '5432',
}
}
```

طارق العمري

١٥ أغسطس ٢٠٢٥

خطوات ربط Django مع MySQL

يدعم Django العديد من أنظمة قواعد البيانات، ومن بينها MySQL الذي يعتبر من الأنظمة الشائعة والمستخدمة على نطاق واسع. سنتعرف على كيفية ربط مشروع Django مع قاعدة بيانات MySQL بخطوات بسيطة.

+ متطلبات وتثبيت المكتبات

لربط Django بقواعد بيانات MySQL، تحتاج أولاً إلى تثبيت المكتبة اللازمة باستخدام الأمر:

```
pip install mysqlclient
```

تأكد من تثبيت MySQL على جهازك أو الخادم الذي ستعمل عليه، وقم بإنشاء قاعدة بيانات جديدة باستخدام الأمر:

```
CREATE DATABASE اسم_قاعدة_البيانات CHARACTER SET utf
```

+ تكوين ملف الإعدادات settings.py

قم بتعديل ملف settings.py في مشروع Django لإضافة إعدادات الاتصال بقاعدة البيانات:

```
# settings.py
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'اسم_قاعدة_البيانات',
        'USER': 'اسم_المستخدم',
        'PASSWORD': 'كلمة_المرور',
        'HOST': 'localhost',
        'PORT': '3306',
        'OPTIONS': {
            'charset': 'utf8mb4'
        }
    }
}
```

طارق العمري

١٥ أغسطس ٢٠٢٥

اختبار الاتصال بقاعدة البيانات مع Django

بعد تكوين إعدادات الاتصال بقاعدة البيانات في ملف settings.py، تحتاج للتأكد من نجاح الاتصال وحل المشاكل المحتملة. يوفر Django أوامر وأدوات تساعد المطور في تشخيص وإصلاح مشاكل الاتصال بقواعد البيانات.

```
# تنفيذ أمر الترحيل للتأكد من الاتصال
python manage.py migrate
```

رسائل النجاح

عند نجاح الاتصال، سترى رسائل تؤكد تنفيذ الترحيلات مثل "Operations to perform" و "Apply all migrations:". هذا يعني أن Django استطاع الاتصال بقاعدة البيانات وإجراء التغييرات اللازمة على الهيكل.

مشاكل الاتصال الشائعة

خطأ "OperationalError": يحدث عند عدم القدرة على الاتصال بالخادم، تأكد من تشغيل خدمة قاعدة البيانات وصحة بيانات الاتصال (اسم المستخدم، كلمة المرور، المنفذ، إلخ).

حل المشاكل

تأكد من تثبيت محرك قاعدة البيانات (مثل PostgreSQL أو MySQL) وتشغيله على نظامك. تحقق من تثبيت مكتبة الاتصال المناسبة (psycopg2 لـ PostgreSQL أو mysqlclient لـ MySQL). استخدم أمر "dbshell" للاتصال المباشر: python manage.py dbshell

التحقق من الإعدادات

استخدم أمر python manage.py check للتحقق من صحة الإعدادات. يمكن تجربة الاتصال من خلال Python shell باستخدام أمر python manage.py shell والتحقق من إمكانية الوصول للنماذج واستعلامها.

طارق العمري

١٥ أغسطس ٢٠٢٥

الخاتمة والتوصيات

+ اختيار النظام المناسب

PostgreSQL هو الخيار الأفضل للمشاريع الكبيرة والمتقدمة التي تتطلب مميزات معقدة وأداء عالي للبيانات الضخمة، بينما MySQL مناسب لأغلب التطبيقات الصغيرة والمتوسطة لسهولة استخدامه وانتشاره الواسع، SQLite يعتبر خيار سريع ومثالي للتجارب والتطوير.

+ تكامل Django

يوفر Django دعمًا ممتازًا لجميع أنظمة قواعد البيانات الرئيسية، مع إعدادات بسيطة للربط. يكفي تثبيت مكتبات الاتصال المناسبة وتعديل ملف الإعدادات settings.py للتكامل مع أي نظام. وهذا ما يجعل Django إطار عمل مرن ومثالي للتطبيقات التي قد تحتاج للتنقل بين أنظمة قواعد بيانات مختلفة.