



الاسم/ ضرار علي محمد عقلان م-٤-١ مجموعة (B2)

تكليف رقم (4) مقرر: هندسة برمجيات م/مالك المصنف

تكليف: مقارنة بين أنظمة إدارة قواعد البيانات:

أولاً: ما هي أنظمة إدارة قواعد البيانات؟

أنظمة إدارة قواعد البيانات (Database Management Systems - DBMS) هي برامج تُستخدم لتخزين وإدارة واسترجاع البيانات من قواعد البيانات. تُقسم بشكل عام إلى فئتين رئيسيتين:

١. قواعد البيانات العلائقية (Relational Databases): تُخزن البيانات في جداول منظمة مع علاقات محددة بينها. تستخدم لغة SQL (Structured Query Language) لإدارة البيانات.

٢. قواعد البيانات غير العلائقية (Non-Relational Databases - NoSQL): تُخزن البيانات بتنسيقات أكثر مرونة، مثل المستندات أو أزواج المفاتيح-القيمة، ولا تتطلب مخططاً ثابتاً. تُستخدم غالباً في التطبيقات التي تحتاج إلى قابلية توسع عالية أو التعامل مع البيانات غير المهيكلة.

ثانياً: مقارنة بين أنظمة إدارة قواعد البيانات

سنقارن بين بعض أشهر الأنظمة العلائقية وغير العلائقية:

### SQLite.1

• النوع: علائقية.

• المميزات:

- خفيفة الوزن ومدمجة (تُخزن في ملف واحد).
- لا تتطلب خادمًا منفصلاً.
- سهولة الإعداد والاستخدام، ومثالية للتطبيقات الصغيرة أو لأغراض التطوير والاختبار.

• العيوب:

- لا تناسب التطبيقات الكبيرة التي تتطلب قابلية توسع أو تعدد المستخدمين.
- أداء محدود في بيئات الإنتاج المعقدة.

### PostgreSQL.2

• النوع: علائقية.

#### • المميزات:

- نظام قوي ومتقدم يدعم ميزات معقدة مثل أنواع البيانات المتقدمة، والفهرسة المتطورة، والبحث النصي الكامل.
- يتميز بالموثوقية وسلامة البيانات.(ACID compliance)
- مناسب للتطبيقات الكبيرة والمعقدة التي تتطلب موثوقية عالية.

#### • العيوب:

- قد يكون إعداداته وإدارته أكثر تعقيداً من الأنظمة الأخرى.

### MySQL.3

#### • النوع: علائقية.

#### • المميزات:

- أكثر الأنظمة العلائقية شهرة واستخداماً في تطبيقات الويب.
- أداؤه سريع في عمليات القراءة المتكررة.
- سهل الاستخدام وله مجتمع كبير يدعمه.

#### • العيوب:

- قد لا يكون بنفس قوة PostgreSQL في التعامل مع بعض الميزات المتقدمة.
- لا يتوافق دائماً بشكل كامل مع معايير SQL.

### MongoDB.4

#### • النوع: غير علائقية.(NoSQL)

#### • المميزات:

- يخزن البيانات في مستندات شبيهة بـJSON.
- مرناً جداً ولا يتطلب مخططاً ثابتاً، مما يسهل التعامل مع البيانات غير المهيكلة.
- قابل للتوسع الأفقي(horizontally scalable) ، مما يجعله مثالياً للبيانات الضخمة والتطبيقات الموزعة.

#### • العيوب:

- لا يدعم المعاملات المعقدة بنفس طريقة قواعد البيانات العلائقية.
- قد يكون أداؤه أبطأ في عمليات البحث التي تتطلب علاقات معقدة.
- غير مناسب للتطبيقات التي تتطلب مخططاً صارماً.

### Microsoft SQL Server.5

• النوع :علائقي.

• المميزات:

- قوة وموثوقية:مصمم للتعامل مع كميات هائلة من البيانات والمعاملات المعقدة في بيئات الإنتاج الكبرى.
- الأداء: يتميز بأداء عالٍ في معالجة الاستعلامات، خاصةً على أنظمة ويندوز.
- التكامل مع منتجات مايكروسوفت: يتكامل بشكل ممتاز مع أدوات أخرى مثل Power BI و Azure، مما يجعله خيارًا مفضلًا للشركات التي تستخدم نظام Microsoft.
- ميزات أمنية متقدمة: يوفر SQL Server ميزات أمان قوية لحماية البيانات.

• العيوب:

- التكلفة:النسخ المدفوعة قد تكون باهظة الثمن.
- التعقيد:إعداده وإدارته قد يكونان أكثر تعقيدًا مقارنةً بـ PostgreSQL أو MySQL.
- التوافق: قد لا يكون سهل الاستخدام على أنظمة التشغيل غير ويندوز مقارنةً بالأنظمة الأخرى.

ثالثًا: طريقة استخدامها في (Django)

لتغيير قاعدة البيانات في Django ، يجب تعديل إعدادات المشروع في ملف settings.py.

١. تثبيت الموصل: (Connector) يجب تثبيت مكتبة Python التي تربط Django بقاعدة البيانات.

○ لـ PostgreSQL: pip install psycopg2

○ لـ MySQL: pip install mysqlclient

٢. تعديل ملف settings.py: داخل قاموس DATABASES ، قم بتحديد المحرك (ENGINE) ، واسم قاعدة البيانات (NAME) ، وبيانات الاعتماد (USER) ، (PASSWORD ، HOST ، PORT).

مثال لربط: PostgreSQL

```
DATABASES = {  
'default' : {  
'ENGINE': 'django.db.backends.postgresql',  
'NAME': 'your_db_name',  
'USER': 'your_db_user',  
'PASSWORD': 'your_db_password',  
'HOST': 'localhost' # أو عنوان IP لخادم قاعدة البيانات  
'PORT': '5432',
```

```
}  
}
```

## مثال لربط: MySQL

```
DATABASES = {  
    'default' : {  
        'ENGINE': 'django.db.backends.mysql',  
        'NAME': 'your_db_name',  
        'USER': 'your_db_user' ,  
        'PASSWORD': 'your_db_password',  
        'HOST': 'localhost' ,  
        'PORT': '3306',  
    }  
}
```

### تشغيل الهجرات (Migrations)

بعد تعديل الإعدادات، قم بتشغيل الأوامر التالية لإنشاء الجداول في قاعدة البيانات الجديدة:

```
python manage.py makemigrations
```

```
python manage.py migrate
```

### استخدام قواعد البيانات غير العلائقية (NoSQL)

لا يدعم Django قواعد بيانات NoSQL بشكل رسمي، ولكن يمكن استخدامها عبر حزم إضافية (third-party packages) مثل Djongo الذي يسمح بربط Django بـ MongoDB. هذه الحزم تعمل على تكييف ORM الخاص بـ Django للعمل مع NoSQL.

مثال لربط MongoDB باستخدام Djongo

١. تثبيت Djongo: `pip install djongo`

٢. تعديل settings.py:

```
DATABASES = {  
    'default': {
```

```
'ENGINE': 'djongo',
'NAME': 'your_mongo_db_name',
}
{
```

## استخدام SQL Server مع Django

### طريقة الاستخدام

١. تثبيت الموصل: (Connector) أشهر حزمة لربط Django بـ SQL Server هي **django-mssql-backend**.

○ تثبيت الحزمة:

```
pip install django-mssql-backend
```

تثبيت مكتبة الاتصال: ستحتاج أيضاً إلى تثبيت مكتبة للاتصال بقاعدة البيانات، مثل **pyodbc**، التي تُستخدم للاتصال بقواعد البيانات عبر ODBC.

```
pip install pyodbc
```

## 2. تعديل ملف settings.py

بعد تثبيت الحزم، قم بتعديل قاموس DATABASES في ملف الإعدادات.

```
DATABASES = {
'default': {
'ENGINE': 'sql_server.pyodbc',
'NAME': 'your_db_name',
'USER': 'your_db_user',
'PASSWORD': 'your_db_password',
'HOST': 'localhost', # أو عنوان IP لخادم SQL Server
'PORT': '1433', # المنفذ الافتراضي لـ SQL Server
'OPTIONS': {
'driver': 'ODBC Driver 17 for SQL Server' # قد تحتاج إلى تغيير اسم الـ
driver
}
}
```

```
}  
  
}
```

### 3. تشغيل الهجرات: (Migrations)

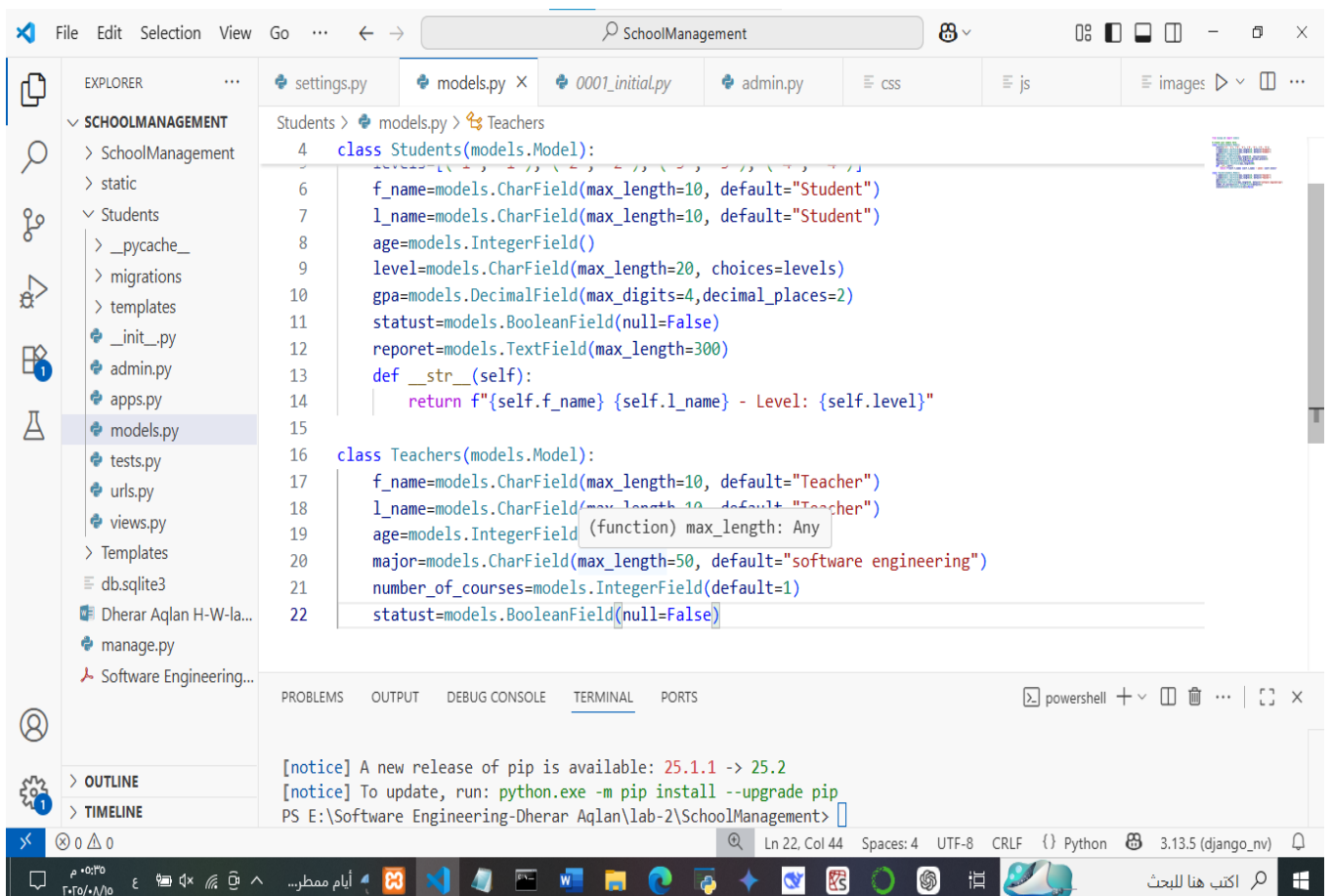
بعد تعديل الإعدادات، يمكنك تشغيل أوامر الهجرة لإنشاء الجداول:

```
python manage.py makemigrations
```

```
python manage.py migrate
```

تكليف ٢: إضافة جدول خاص بالمعلمين.

١. شكل MODEL جدول المعلمين.



## ٢. ملف migrations

```
Students > migrations > 0002_teachers.py > ...
5
6 class Migration(migrations.Migration):
7
8     dependencies = [
9         ('Students', '0001_initial'),
10    ]
11
12    operations = [
13        migrations.CreateModel(
14            name='Teachers',
15            fields=[
16                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False,
17                ('f_name', models.CharField(default='Teacher', max_length=10)),
18                ('l_name', models.CharField(default='Teacher', max_length=10)),
19                ('age', models.IntegerField()),
20                ('major', models.CharField(default='software engineering', max_length=50)),
21                ('number_of_courses', models.IntegerField(default=1)),
22                ('statust', models.BooleanField()),
23            ],
24        ),
25    ]
```

## ٣. الجدول في admin

### Site administration

AUTHENTICATION AND AUTHORIZATION

Groups

+ Add   ✎ Change

Users

+ Add   ✎ Change

STUDENTS

Studentss

+ Add   ✎ Change

Teacherss

+ Add   ✎ Change

Recent actions

My actions

+ Teachers object (1)  
Teachers

+ Dherar Ali - Level: 4  
Students



## ٤. الجدول في قاعدة البيانات.

The screenshot shows the phpMyAdmin interface for a database named 'lab4'. The left sidebar displays the database structure, including tables like 'auth\_group', 'auth\_group\_permissions', 'auth\_permission', 'auth\_user', 'auth\_user\_groups', 'auth\_user\_user\_permissions', 'django\_admin\_log', 'django\_content\_type', 'django\_migrations', 'django\_session', 'students\_students', and 'students\_teachers'. The main area shows a list of tables with columns: 'الجدول' (Table), 'الحجم' (Size), 'التجميع' (Collation), 'النوع' (Type), 'صفوف' (Rows), 'العملية' (Operation), and 'المجموع' (Summary). The table 'students\_teachers' is highlighted, showing 66 rows and a size of 416.0 KB.

الجدول	الحجم	التجميع	النوع	صفوف	العملية	المجموع
auth_group	32.0 كيلوبايت	utf8mb4_general_ci	InnoDB	0	إدخال	0
auth_group_permissions	48.0 كيلوبايت	utf8mb4_general_ci	InnoDB	0	إدخال	0
auth_permission	32.0 كيلوبايت	utf8mb4_general_ci	InnoDB	32	إدخال	0
auth_user	32.0 كيلوبايت	utf8mb4_general_ci	InnoDB	1	إدخال	0
auth_user_groups	48.0 كيلوبايت	utf8mb4_general_ci	InnoDB	0	إدخال	0
auth_user_user_permissions	48.0 كيلوبايت	utf8mb4_general_ci	InnoDB	0	إدخال	0
django_admin_log	48.0 كيلوبايت	utf8mb4_general_ci	InnoDB	2	إدخال	0
django_content_type	48.0 كيلوبايت	utf8mb4_general_ci	InnoDB	8	إدخال	0
django_migrations	16.0 كيلوبايت	utf8mb4_general_ci	InnoDB	20	إدخال	0
django_session	32.0 كيلوبايت	utf8mb4_general_ci	InnoDB	1	إدخال	0
students_students	16.0 كيلوبايت	utf8mb4_general_ci	InnoDB	1	إدخال	0
students_teachers	16.0 كيلوبايت	utf8mb4_general_ci	InnoDB	1	إدخال	0
المجموع	416.0 كيلوبايت	utf8mb4_general_ci	InnoDB	66		0

## ٥. عرض بيانات المعلم المضاف في الجدول.

The screenshot shows the phpMyAdmin interface for the 'students\_teachers' table. The main area displays the table data with columns: 'statust', 'number\_of\_courses', 'major', 'age', 'l\_name', 'f\_name', and 'id'. The table contains one row of data for a teacher named Ali Dherar, who is 35 years old and teaches software engineering.

statust	number_of_courses	major	age	l_name	f_name	id
1	3	software engineering	35	Ali	Dherar	1