Modules et tables spécifiques

01 Contexte

Certains modules Drupal sont de véritables **applications Web** qui nécessitent des **tables spécifiques** pour fonctionner. Ces tables doivent être créées lors de l'installation du module.

C'est pourquoi Drupal propose un mécanisme permettant à un module de rajouter **ses propres tables** dans la base de données Drupal lors de son installation.

Pour cela le module **MODULE** doit implanter le HOOK_schema et le HOOK_install dans un **fichier spécifique** nommé **MODULE.install**.

Le **HOOK_schema** ne fait que renvoyer un ensemble d'informations qu'on appelle le schéma du module.
Le **HOOK_install** peut exécuter des fonctions, initialiser des variables, initialiser des tables, ...

02 Etats d'un module

Une fois que les fichiers du module ont été rajoutés à Drupal, le module peut être dans **3 états** différents :
1. ni installé ni activé, 2. installé et activé, 3. installé mais désactivé.

Dans votre installation Drupal affichez la page admin/modules pour visualiser de quoi nous parlons. Pour chaque module, il y a une ligne avec une case à cocher, le nom et la version du module, sa description. Tout à droite il y a une colonne OPERATIONS qui peut contenir jusqu'à trois liens : Help, Permissions et Configure.

Lors de la première activation, en cochant la case correspondante dans le formulaire admin/modules, le module est à la fois installé et activé.
En décochant la case correspondante dans le formulaire admin/modules, le module est désactivé mais reste installé.
Pour désinstaller un module qui a été désactivé précédemment, il faut utiliser l'onglet Désinstaller.

03 Etude du code générant le formulaire admin/modules

Pour cette étude vous devez utiliser notepad++. Cet éditeur permet de faire des recherches dans un fichier (Ctrl+F) ou dans tous les fichiers d'un répertoire.

a. le chemin admin/modules

Dans le fichier modules/system/system.module, recherchez l'item de menu correspondant au chemin admin/modules. Dîtes quel est le numéro de ligne où se trouve cet item et recopiez le dans votre rapport.
Cet item permet-il d'afficher un formulaire ? Expliquez pourquoi. Quelle est le nom de la fonction qui va construire ce formulaire ? Dans quel fichier se trouve-t-elle ? Quel droit l'utilisateur doit-il avoir pour afficher ce formulaire ?

b. Le formulaire system_modules

1.Recherchez le code de la fonction system_modules.
2. Quel est le nom de la fonction qui va récupérer les informations de tous les modules pour les mettre dans le tableau $files ?
3. A quelle ligne commence la boucle qui itère sur la liste des modules ?
4. A quelle ligne commence le code qui crée le lien Help d'un module. Recopiez ces lignes de code dans votre rapport.
5. A quelle ligne commence le code qui crée le lien Permissions d'un module ? Recopiez ces lignes de code dans votre rapport.
6. A quelle ligne commence le code qui crée le lien Configure d'un module ? Recopiez ces lignes de code dans votre rapport.

7. A quelle ligne la fonction _system_modules_build_row renvoie-t-elle la ligne correspondant à un module pour qu'elle soit rajoutée au formulaire ?

c. Test de debuggage dans Eclipse
Dans Eclipse, mettez un breakpoint sur la dernière ligne de la fonction system_modules.
Lancez un debuggage puis cliquez sur le menu modules dans la barre d'administration.
Lorsque l'exécution du programme s'arrête à ce breakpoint, examinez le contenu de la variable $form['modules']['Core']['block']. Recopiez ce contenu dans votre rapport. Expliquez.

04 Activation d'un module

Le texte suivant explique comment, lors de l'activation d'un module, Drupal teste s'il faut créer des tables spécifiques pour ce module et procède à cette création le cas échéant.
Vous devez compléter ce texte en donnant les noms des fichiers et les numéros de lignes aux emplacements notés (?fichierX) et (?ligneX).

Fonctionnement de l'activation :
- Pour activer un module, il faut cocher la case correspondante dans le formulaire affichant la liste des modules et contenu dans la page admin/modules.

- En cliquant sur le bouton *Save configuration*, nous exécutons la fonction nommée **system_modules_submit** (?fichier1 ?ligne1a) qui appelle (?ligne1b) la fonction module_enable.

- Lors de la première activation d'un module (ou lorsqu'un module est réactivé après une désinstallation), la fonction **module_enable** (?fichier2 ?ligne2a), déclenche l'exécution du hook_schema puis exécute le hook_install du module. Plus précisément, module_enable, appelle (?ligne2b) la fonction drupal_install_schema.

- La fonction **drupal_install_schema** (?fichier3 ?ligne3a) appelle d'abord la fonction drupal_get_schema_unprocessed (?ligne3b).

- La fonction **drupal_get_schema_unprocessed** (?fichier4 ?ligne4a) récupère le schéma d'un module en exécutant : $schema = module_invoke($module, 'schema'); (?ligne4b)

- Le **hook_schema** du module renvoie le schéma du module.

- La fonction **drupal_install_schema** appelle (?ligne3c) la fonction _drupal_schema_initialize pour compléter le schéma.

- La fonction **drupal_install_schema** commande la création des tables du schéma en exécutant (?ligne3d) : foreach ($schema as $name => $table) { db_create_table($name, $table); }

- La fonction **db_create_table** (?fichier5 ?ligne5) exécute l'appel de méthode :
Database::getConnection()->schema()->createTable($name, $table);

- La méthode **createTable** de la classe DatabaseSchema (?fichier6 ?ligne6a) appelle la méthode createTableSql.

- La méthode **createTableSql** de la classe DatabaseSchema_mysql, qui se trouve dans le fichier (?fichier7) ligne (?ligne7a), génère la requête SQL de création d'une table à partir de son schéma.

- La méthode **createTable** lance ensuite (?ligne6b) l'exécution de la requête.

Le **hook_install** est, quant à lui, appelé ensuite par module_enable à la ligne (?ligne2c) par l'expression : module_invoke($module, 'install');

Dans votre rapport, pour faciliter la correction, pour chaque fonction indiquez :
nom de la fonction, fichierX = xxx, ligneXa = xxx, ligneXb = xxx etc.

05 La notion de schéma

Drupal utilise des schémas pour décrire les tables de base de données dont il demande la création. Un schéma est un tableau associatif écrit en PHP.

Au premier niveau, les clés correspondent aux noms des tables à créer et les valeurs à des sous-tableaux donnant les informations pour créer une table.
Au second niveau on peut avoir les clés suivantes : 'description', 'fields', 'indexes', 'unique keys', 'foreign keys', 'primary key'.
La valeur correspondant à la clé 'fields' est un tableau associatif décrivant les champs de la table.
Pour chaque champ la clé est le nom du champ et la valeur est un tableau associatif décrivant le champ en utilisant les étiquettes 'type', 'unsigned', length', 'size', 'not null', 'serialize', 'description' …

Exemple de schéma avec une seule table :

```
function MODULE_schema(){
  $schema['table1'] = array(
    'description' => 'Phrase de description table1.',
    'fields' => array(
      'id' => array('type' => 'serial', 'unsigned' => TRUE, 'not null' => TRUE),
      'title'=> array('type'=>'varchar','length'=>255,'not null'=>TRUE, 'default' => ''),
      'uid' => array('type' => 'int', 'not null' => TRUE, 'default' => 0),
      'data' => array('type'=>'blob', 'not null'=>FALSE, 'size'=>'big',
        'serialize'=> TRUE, 'description' => 'Les données contenues dans l\'item.',
      ),
      'status' => array('type' => 'int', 'not null' => TRUE, 'default' => 1),
    ),
    'indexes' => array(
      'uid'        => array('uid'),
    ),
    'unique keys' => array(
      'title' => array('title'),
    ),
    'foreign keys' => array(
       'author' => array('table' => 'users', 'columns' => array('uid' => 'uid'),),
    ),
    'primary key' => array('id'),
  );
  return $schema;
}
```

06 Exercice :

1. Désinstaller votre module puis dans le fichier .install ajouter le HOOK_schema donné dans l'exemple.
2. Réactivez votre module et vérifier la création de la table table1 dans la BD de Drupal.
3. Désactivez votre module. Que ce passe-t-il dans la BD de Drupal ?
4. Désinstallez complètement votre module. Que ce passe-t-il dans la BD de Drupal ?

Quand votre exercice fonctionnera, faîtes une démonstration à votre enseignant.

07 Comment créer les tables dans une autre base de données

HOOK_schema est utilisé (indirectement) par la fonction drupal_install_schema qui crée forcément les tables dans la base de Drupal.

Pour créer les tables dans une autre base, il faut donc renvoyer le schema avec une autre fonction que nous pouvons appeler : MODULE_schema_AUTREBASE. AUTREBASE est le nom de la base cible. Le code de cette fonction est le même que celui que nous mettrions dans HOOK_schema si nous voulions créer les tables dans la base de Drupal.
Cette fonction doit être appelée dans HOOK_install de la manière suivante :

```
function MODULE_install() {
  ...
  //On imite ce qui se passe dans la fonction drupal_install_schema
  $schema = MODULE_schema_AUTREBASE();
  _drupal_schema_initialize($schema, MODULE, FALSE);
  db_set_active(AUTREBASE); //changement de base active
  foreach ($schema as $name => $table) {
    db_create_table($name, $table);
  }
  db_set_active(); //retour à la bd Drupal
  ...
}
```

Attention, AUTREBASE est une chaîne de caractères.

08 Exercice :

1. Créez une nouvelle BD de nom bd_externe avec les mêmes droits que la BD de Drupal.
2. Modifiez le fichier settings.php pour que Drupal puisse accéder à cette base, de la manière suivante :
- Ajouter une clé 'bd_externe' dans le tableau $databases.
- Donner lui la même valeur que celle de la clé 'default' en remplaçant simplement la valeur de la clé 'database' par 'bd_externe'.
3. Modifier votre fichier MODULE.install pour que table1 soit créée dans bd_externe
4. Désinstaller votre module.
5. Réactivez votre module et vérifier la création de la table table1 dans bd_externe.

Quand votre exercice fonctionnera, faîtes une démonstration à votre enseignant.