2025/11/17 20:16 1/5 MultiDB - PDO

MultiDB - PDO

In questa sezione sono descritte le configurazioni necessarie per interfacciarsi con i diversi driver per la connessione con i DBMS.

MySQL

Attualmente non prevista la gestione.

PostgreSQL

La dll del driver è già presente in php. Nel file 'php.ini' occorre abilitare la seguente istruzione:

```
extension=php_pdo_pgsql.dll
```

TODO: completare caso per caso

MS SQL Server (da 2008 in avanti)

Installazione driver

Scaricare il driver direttamente dal sito di Microsoft utilizzando questo link. Prendere SQLSRV32.EXE eseguirlo e copiare su '...\xampp\php\ext' solo i file php_pdo_sqlsrv_56_ts.dll e php_sqlsrv_56_ts.dll generati dall'exe.

ATTENZIONE! Dalla versione 3.1 in avanti, è necessario scaricare anche il driver ODBC per MS Sql Server, sempre dal sito di Microsoft, utilizzando questo link.

Configurazione php.ini

Dopo aver installato i driver, nel file 'php.ini' occorre aggiungere:

```
extension=php_sqlsrv_56_ts.dll
extension=php pdo sqlsrv 56 ts.dll
```

Oracle

Installazione driver

Utilizzare Instant Client (a seconda della versione di Oracle)

Documentazione di riferimento

php.net

Guida all'installazione su CentOS

Oracle PDO e CentOS

Configurazione php.ini

Dopo aver installato i driver, nel file 'php.ini' occorre abilitare la seguente istruzione:

extension=php pdo oci.dll

Struttura delle classi in ita Engine

config.ini

Per abilitare PDO, modificare la seguente impostazione nel file config.ini:

[dbms]
dbengine=DBPD0
attivaTransazioni=1

lib/DBPDO

- ItaDB: Classe Proxy
- ItaDBFacade: Classe Facade per interfaccia "legacy" con i driver
- PDPFacade: Classe Facade per interfaccia PDO con i driver
- ItaDBError: Errori specifici DB
- PDODriver: Superclasse Driver DB
- PDOHelper: Helper funzioni PDO
- PDOTableDef: Classe che contiene la definizione della tabella
- PDOTableDefFactory: Factory per creazione TableDef
- PDOTableDefValidator: Classe per validazione tabella
- PDO<Driver specifico>: Classi specifiche per i driver (MS SQL Server, Oracle, Postgres...)

https://wiki.nuvolaitalsoft.it/ Printed on 2025/11/17 20:16

Cache TableDef

Le definizioni delle tabelle vengono messe in cache. Per attivare la cache, occorre modificare il file config.ini aggiungendo le seguenti righe: (Oltre alle definizioni delle colonne, la tableDef contiene anche le eventuali relazioni)

```
[cache]
type=FILE
root=C:/Works/PhpDev/dati/itaTest/cache
```

Validatori

Per la validazione, la logica è la seguente:

- Se esiste il validatore specifico per un model, questo viene utilizzato
- Se non esiste il validatore specifico per un model, viene effettuata la fallback sulla classe 'PDOTableDefValidator.class'

Per stabilire il validatore specifico, la regola è la seguente:

- Nome model = cwbBtaNazion.php
- Classe validator specifica = validators/cwbBtaNazionValidator.php

Può essere necessario escludere alcuni campi dalla validazione, implementando il metodo **initExcludeFields**, es:

```
public function initExcludeFields() {
    parent::initExcludeField("IS03166_A2");
    $this->addExcludeField("IS03166_A3");
    $this->addExcludeField("IS03166_N3");
    $this->addExcludeField("CODNAZICO");
    $this->addExcludeField("CODNAZIMC");
    $this->addExcludeField("CODGOVE");
    $this->addExcludeField("LLPIVANAZ");
    $this->addExcludeField("CODAGEO");
    $this->addExcludeField("CODEST_770");
}
```

Gestione relazioni

E' possibile effettuare il salvataggio e cancellazione (in transazione) dei dati aggiuntivi rispetto al record principale. I tipi di relazione gestiti sono:

- One-To-One
- · One-To-Many

Last update: 2018/03/19 10:45

• Many-To-One

Occorre effettuare l'override del metodo 'caricaDatiAggiuntivi'. Come esempio, vedere 'cwbBtaGrunaz Rel'.

Transazioni

Per ogni connessione, esiste una sola transazione (non vengono annidate).

Abilitare sul Hook.ini il listner:

[connectionPerRequestPDOHook.php]
active=1

Gestione delle connessione e della transazione

La connessione viene salvata come resources all'interno del App→\$utente in modo da avere una connessione per request Punti fermi:

- 1. Elenco numeratoAd ogni request la connessione al db muore
- Il nome della connessione deve essere univico (ci saranno la connessione 'cityware', 'numeratori',ecc)
- 3. le query e il service utilizzano sempre la stessa connessione per request presa dal App::\$utente

Transazioni: Punti fermi:

- 1. Esiste un unica transazione aperta per ogni connessione (logica multidb)
- 2. Service: di default effettua begin_Transaction (se non la trova aperta) e commit_transaction
- 3. Service: è possibile non fare la commit\rollback se viene impostato l'attributo '\$startedTransaction'(o in fase di builder o impostando un "set")
- 4. E' possibile gestire manualmente la transazione usando 'manual' in fase di addTransaction.
- 5. Nel aprire manualmente una nuova sessione c'è il controllo delle transazioni aperte. In caso sia aperta lancia eccezione.
- 6. Microsoft Mssql con driver installato su macchine linux non è permesso gestire le transazioni.

La chiusura (commit\rollback) deve essere fatta esplicitamente con l'attributo 'manual'. Qualsiasi altro salvataggio\cancellazione\inserimento lanciato con attivo lo stato manual non interferisce con la transazione.

Lettura e scrittura dei binari con il PDO

Per leggere e scrivere correttamente un blob effettuare le seguenti operazioni:

- 1. Nella function che ritorna la stringa sql (esempio cwbLibDB_BGE function getSqlLeggiBgeAgidConfEfil) è obbligatorio specificare tutti i campi e non il il classico tabella.*
- 2. Nella function che ritorna la stringa sql (esempio cwbLibDB_BGE function getSqlLeggiBgeAgidConfEfil) è obbligatorio specificare tutti i campi e non il il classico tabella.*
- 3. Sempre nella stessa function per i campi binari usare questa function per formattare la stringa

https://wiki.nuvolaitalsoft.it/

- \$this→getCitywareDB()→adapterBlob("NOMECAMPOBINARIO").Questo risolve il problema dei fetch null con oracle su piattaforma linux
- 4. Sulla leggi effettiva (esempio cwbLibDB_BGE function leggiBgeAgidConfEfil..) passare un array con oggetto e metodo. E' la callback chiamata per il database mssql per caricare il binario per caricare sull'array principale i dati del blob per ogni record
- 5. Implementare questa callback (esempio cwbLibDB_BGE leggiBgeAgidConfEfilBinary) specificando il tipo blob

From:

https://wiki.nuvolaitalsoft.it/ - wiki

Permanent link:

https://wiki.nuvolaitalsoft.it/doku.php?id=sviluppo:multidb&rev=1512384123

Last update: 2018/03/19 10:45

