Node.JS DB Tool使用說明

1.

kplug-mssql: 為MSSQL函式庫

kplug-mysql: 為MySQL函式庫

kplug-oracle: 為ORACLE函式庫

2.

test目錄: 為測試範例

testMysql.js: 為MySQL測試範例

testMSSQL.js: 為MSSQL測試範例

testMSSQL2.js: 為MSSQL測試範例

testOracle1.js: 為ORACLE測試範例

testOracle2.js: 為ORACLE測試範例

3.資料庫設定檔設置:

config.js為設定檔案

* **config**.**mssql**: 為MSSQL連線資訊
* **config**.**oracle**: 為ORACLE連線資訊
* **config**.**mysql**: 為MySQL連線資訊

4.使用方法

第一支啟動的js必須執行DB Tool的初始化

如下

* require(**'../kplug-mssql/DB'**).create(**config**.**mssql**);
* require(**'../kplug-oracle/DB'**).create(**config**.**oracle**);
* *require*(**'../kplug-mysql/DB'**).create(**config**.**mysql**);

於須查詢DB的JS import下列各種DB的Query Agent

* + - **var** queryAgent = *require*(**'../kplug-oracle/QueryAgent'**);
    - **var** queryAgent = require(**'../kplug-mssql/QueryAgent'**);
    - **var** queryAgent = *require*(**'../kplug-oracle/QueryAgent'**);

功能:

|  |  |  |
| --- | --- | --- |
| method | 功能介紹 |  |
| query | 單筆資料查詢 | 第一個參數:使用的xml dao name  第二個參數:為條件  第三個參數:為資料回傳Callback   1. 第一個回傳參數為錯誤 2. 第二個回傳參數為查詢結果資料 |
| QueryAgent.query(**'test\_mysql'**, {**seq**: 2}, **function** (err, row) {  **if** (err) {  ***console***.error(err);  }  **if** (row != **null**)  ***console***.log(row); }); | | |
| queryList | 多筆資料查詢 | 第1個參數:使用的xml dao name  第2個參數:為條件  第3個參數:查詢起始位置  第4個參數:查詢筆數，0為全部資料  第5個參數:為資料回傳Callback   1. 第一個回傳參數為錯誤 2. 第二個回傳參數為查詢結果資料(為Array) |
| queryAgent.queryList(**'test\_test2'**, {**hotel\_cod**: **'0001'**, **keep\_dat**: **new moment**(**'2016/05/15 22:03:11'**, **'YYYY/MM/DD HH:mm:ss'**).toDate()}, 0, 0, **function** (err, row) {  **if** (err) {  ***console***.error(err);  }  ***console***.log(**'row:'** + row.**length**);  **if** (row.**length** > 0) {  ***console***.log(**'row:'** + row[0].rvreserve\_nos);  ***console***.log(**'row:'** + row[0].**keep\_dat**);  } }); | | |

SQL XML介紹

位置:dao目錄

service\_dao.xml為xml清單，描述使用哪些sql xml檔案

範例:

<**dao name="test\_error"**>  
 <**statement**><![CDATA[SELECT \* FROM hotel\_rf WHERE hotel\_cod = ? and hotel\_nam like ? and aa =1]]></**statement**>  
 <**parameter type="likestring" kind="2" condition="atten1\_add like ?"**>atten1\_add</**parameter**>  
 <**parameter type="int"**>hotel\_cod</**parameter**>  
 <**parameter type="likestring"**>hotel\_nam</**parameter**>  
</**dao**>

<**dao name="test\_test"**>  
 <**statement**><![CDATA[SELECT hotel\_nam FROM hotel\_rf WHERE hotel\_cod = ?]]></**statement**>  
 <**statement test="\_d(param.aa) == 2"**><![CDATA[and hotel\_nam like :hotel\_nam]]></**statement**>  
 <**parameter type="likestring" kind="2" condition="atten1\_add like ?"**>atten1\_add</**parameter**>  
 <**parameter type="int"**>hotel\_cod</**parameter**>  
 <**parameter type="likestring" kind="3"**>hotel\_nam</**parameter**>  
 <**groupby test="\_d(param.aa) == 2"**>group by hotel\_nam</**groupby**>  
 <**orderby test="\_d(param.aa) == 2"**>order by hotel\_nam</**orderby**>  
</**dao**>

1. **statement儲存sql指定，可以為多筆，多筆指令會自動相接成一句**
2. **statement中有test屬性，屬性中的表達式為true時，statement中的SQL才會加入相接的SQL(\_v為是否存在;\_d為是否有符合此條件)**
3. **parameter描述帶入值參數，type參數表達參數的性質是int、string、 date、likestring，parameter欄位value描述所使用的帶入查詢的參數key值**
4. **parameter的kind，有1、2、3這三種，沒kind為1**

**kind=1為帶入SQL中有?部分，kind=1的parameter有順序關係，按照順序對應?**

**kind=2根據這parameter的value值所對應的參數值key，檢查傳入的條件，此key值條件，若不是空值，將condition中的帶入SQL相接起來**

**kind=3根據這parameter的value值所對應的參數值key，帶入:key**

1. groupby將group by 指令加入SQL
2. orderby將order by 指令加入SQL