

Whitepaper

amapolisTechnik

Im Dialog mit unterschiedlichen Datenbank-Typen



Version: 1

Stand: 22.08.2008

Die sanfte IT (R)Evolution

m:\amapolis\marketing\whitepapers\german\leplStandaloneUse.doc

I. Zusammenfassung

In diesem Dokument wird aufgezeigt, wie Sie mit der Programmiersprache amapolisPL die Verbindung mit Datenbanken unterschiedlichen Typs herstellen und dann in einheitlicher Syntax in der Datenbank arbeiten.

amapolis IT Services GmbH
Gewerbestr. 42
70565 Stuttgart
Tel. +49-711 / 655500-40
Fax. +49-711 / 655500-49
www.amapolis.com

II. Copyright

Copyright © amapolis IT Services GmbH. Alle Rechte vorbehalten.

amapolis IT Services GmbH haftet nicht für etwaige Fehler in dieser Dokumentation. Die Haftung für mittelbare und unmittelbare Schäden, die in Zusammenhang mit der Lieferung oder dem Gebrauch dieser Dokumentation entstehen, ist ausgeschlossen, soweit dies Gesetzlich zulässig ist.

Alle Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung, sowie der Übersetzung, bleiben vorbehalten. Kein Teil der Dokumentation darf in irgendeiner Form (durch Fotokopie, Mikrofilm oder anderes Verfahren) ohne vorherige Zustimmung von amapolis IT Services GmbH reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

Alle aufgeführten Marken sind Warenzeichen der jeweiligen Hersteller.
Diese Dokumentation ersetzt nicht die übergebenen Produkthandbücher, sondern ergänzt diese.

III. Inhaltsverzeichnis

Kapitel	Seite
1 Einleitung.....	4
1.1 Zielgruppe.....	4
1.2 Ziel des Dokuments.....	4
1.3 Begriffe.....	4
2 Voraussetzungen.....	4
3 ODBC-Verbindungen einrichten und testen	5
3.1 Oracle.....	5
3.2 SQL Server	6
3.3 MySQL.....	7
4 Mit unterschiedlichen Datenbanken arbeiten	8
4.1 Connect via amapolisPL standalone	8
4.1.1 Datenbank Oracle.....	8
4.1.2 Datenbank SQL Server.....	9
4.1.3 Datenbank MySQL	9
4.2 Anweisungen und Programme ausführen	10
5 Beispiele downloaden	15
6 Weitere Informationen	15
7 amapolisPL und Unix.....	15
8 Feedback	15

1 Einleitung

1.1 Zielgruppe

Dieses Dokument wendet sich in erster Linie an amapolis-Anwender und an der Entwicklungsumgebung amapolis *Technik* Interessierte.

1.2 Ziel des Dokuments

Sie sehen Beispiele, wie Sie mit der Programmiersprache amapolisPL in einheitlicher Notation auf Datenbanken unterschiedlichen Typs (hier Oracle, MySQL und SQLServer) zugreifen.

1.3 Begriffe

Zur Entwicklungsumgebung amapolisPL gehört die Programmiersprache lepl.

amapolisPL standalone: amapolis-Datenbankzugriff über Command-Line mit lepl

Das Connect zur Datenbank geschieht am einfachsten mit amapolisPL standalone.

amapolisPL: amapolis-Systemzugriff über Command-Line mit lepl

Die erweiterten Möglichkeiten des Zugriffs zur Datenbank via amapolis *Applikationsserver* und amapolis *Datenbankserver* werden hier nicht erläutert.

2 Voraussetzungen

Für Oracle, MySQL und SQLServer-DB sind entsprechende Datenbanken eingerichtet. Das Programm lepl steht zur Verfügung.

Betriebssystem Windows: auf dem lokalen Rechner sind ODBC-Verbindungen eingerichtet, über die die Datenbanken mittels folgender Parameter erreichbar sind.

Datenbanktyp	Datenquellenname	DB-User	DB-Password
Oracle	leORACLE	root	root
SQLServer	leSQLSERVER	root	root
MySQL	leMYSQL	root	root

3 ODBC-Verbindungen einrichten und testen

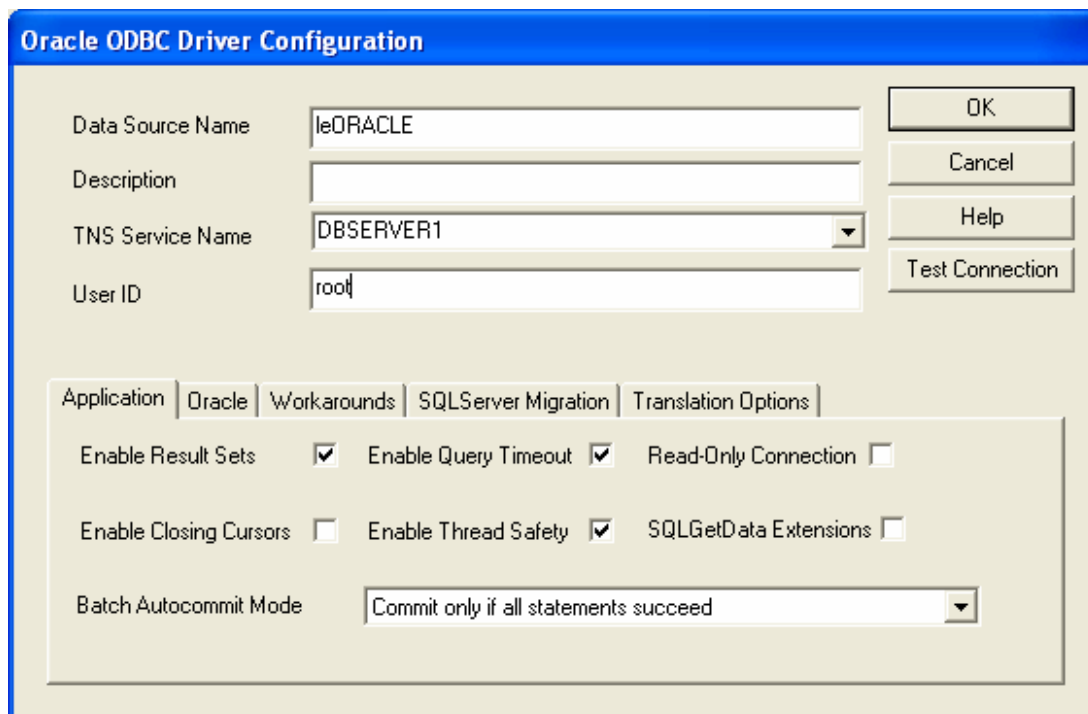
Wählen Sie bitte unter Systemsteuerung -> Verwaltung -> Datenquellen (ODBC) aus. Legen Sie falls notwendig die ODBC-Verbindungen unter Benutzer-DSN oder System-DSN an und testen Sie das Connect zur Datenbank. Dazu benötigen Sie Administrator-Rechte.

3.1 Oracle

Der Datenquellenname DSN ist generell frei wählbar, TNS ist ein auf dem Rechner in tnsnames.ora eingetragener Connect-String zum Remote-Zugriff auf die Datenbank. Der Datenbank-Teil der Installation ist dann erfolgreich, wenn die folgenden Kommandos fehlerfrei arbeiten.

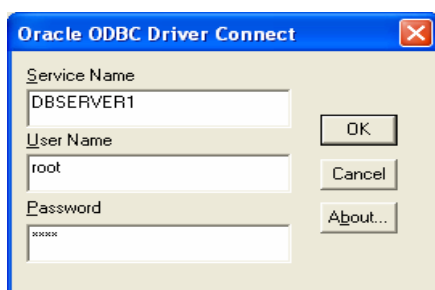
```
...>tnsping DBSERVER1
```

```
...>sqlplus root/root@DBSERVER1
```



The screenshot shows the 'Oracle ODBC Driver Configuration' dialog box. The 'Data Source Name' field contains 'leORACLE'. The 'TNS Service Name' dropdown is set to 'DBSERVER1'. The 'User ID' field contains 'root'. On the right side, there are buttons for 'OK', 'Cancel', 'Help', and 'Test Connection'. Below the main fields, there are tabs for 'Application', 'Oracle', 'Workarounds', 'SQLServer Migration', and 'Translation Options'. Under the 'Oracle' tab, there are several checkboxes: 'Enable Result Sets' (checked), 'Enable Query Timeout' (checked), 'Read-Only Connection' (unchecked), 'Enable Closing Cursors' (unchecked), 'Enable Thread Safety' (checked), and 'SQLGetData Extensions' (unchecked). At the bottom, there is a 'Batch Autocommit Mode' dropdown menu set to 'Commit only if all statements succeed'.

Mit diesen Angaben ist dann auch ein **Test Connection** erfolgreich.



The screenshot shows the 'Oracle ODBC Driver Connect' dialog box. It has three input fields: 'Service Name' with 'DBSERVER1', 'User Name' with 'root', and 'Password' with 'xxxx'. There are buttons for 'OK', 'Cancel', and 'About...' on the right side.



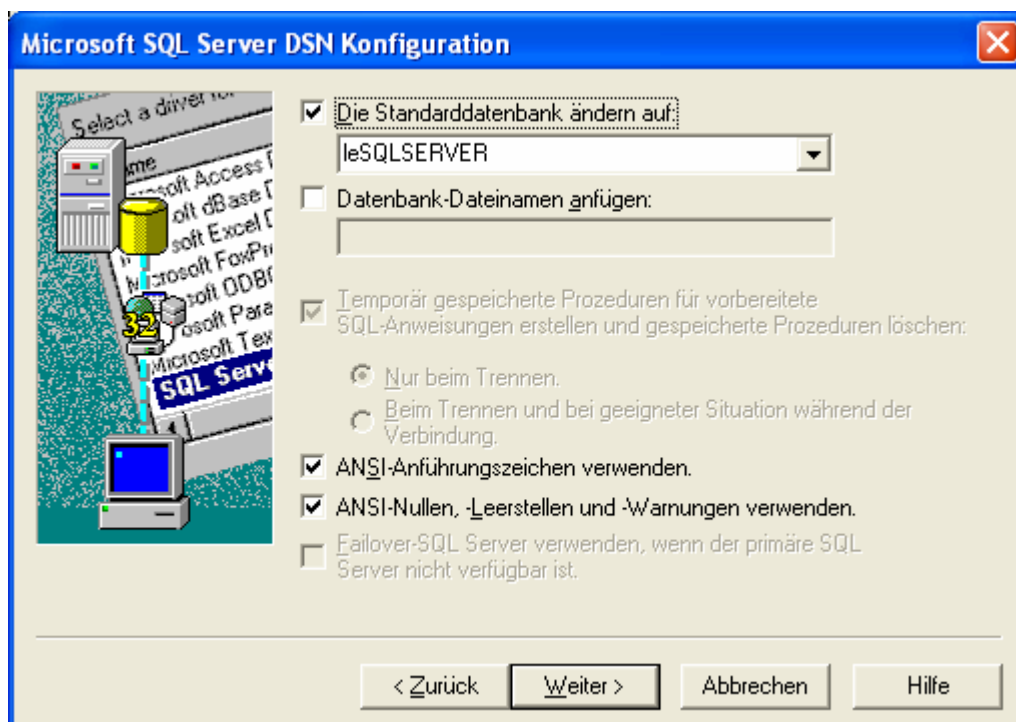
The screenshot shows a small dialog box titled 'Testing Connection'. It contains the text 'Connection successful' and an 'OK' button at the bottom.

3.2 SQL Server

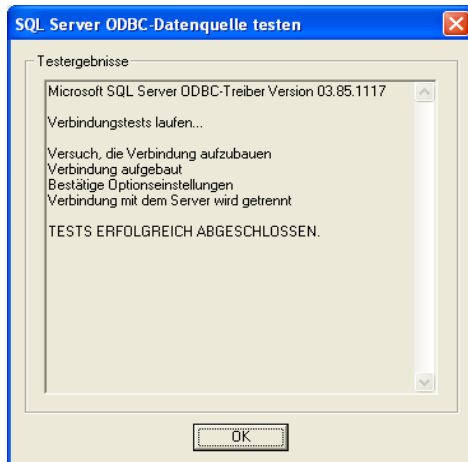
Der Zugriff zu Remote Datenbanken erfolgt durch Angabe des SQL Servers.



Auf dem Rechner DBSERVER3 existieren mehrere SQL Server-Datenbanken, hier kann der Name der Datenbank unabhängig vom DSN bei Bedarf ausgewählt werden.

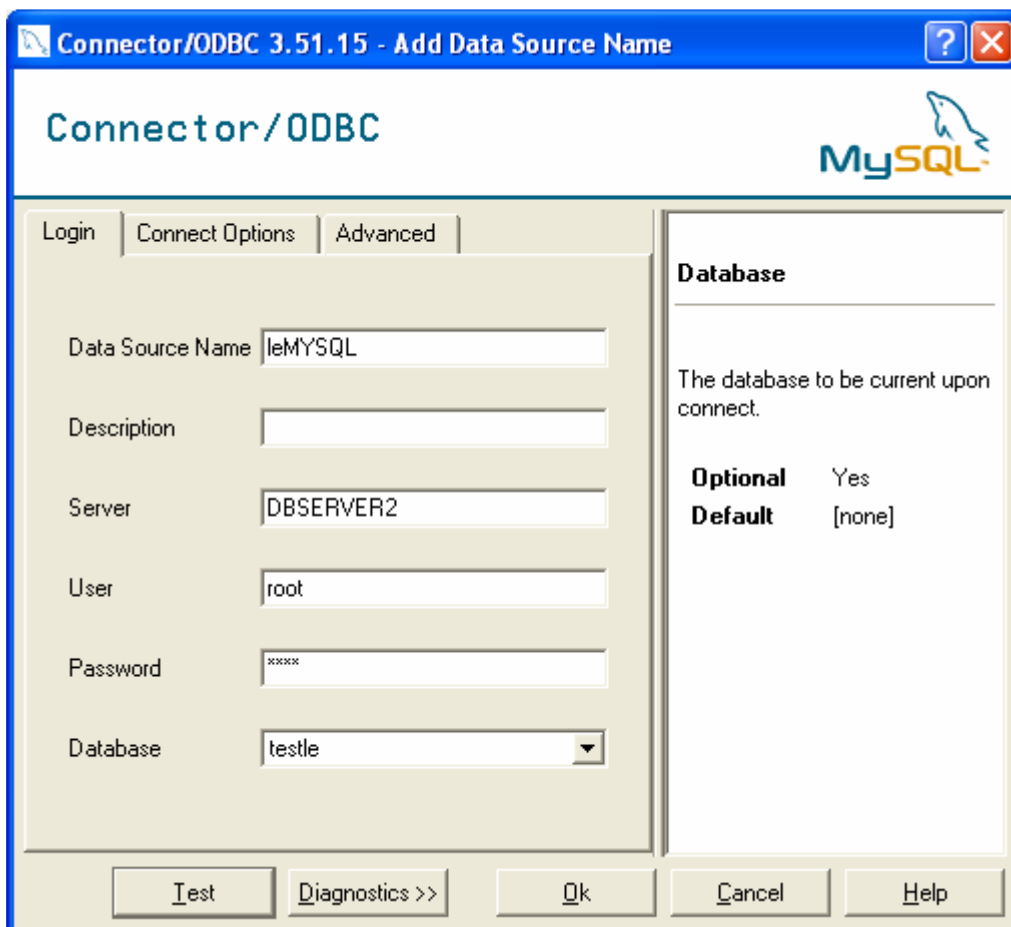


Sie überprüfen den Zugriff unter **Weiter** mit Button **Datenquelle testen**.



3.3 MySQL

Der MySQL-Server befindet sich auf dem Rechner DBSERVER2, dort ist eine Datenbank testle angelegt.



Der Button **Test** erlaubt den Aufbau einer Verbindung.



4 Mit unterschiedlichen Datenbanken arbeiten

amapolisPL standalone benötigt als Parameter für ein Connect den ODBC-Datenquellennamen, den Datenbanktyp (oracle, sql_server, mysql, ...) und für die Datenbank selbst einen DB-User und dessen DB-Passwort.

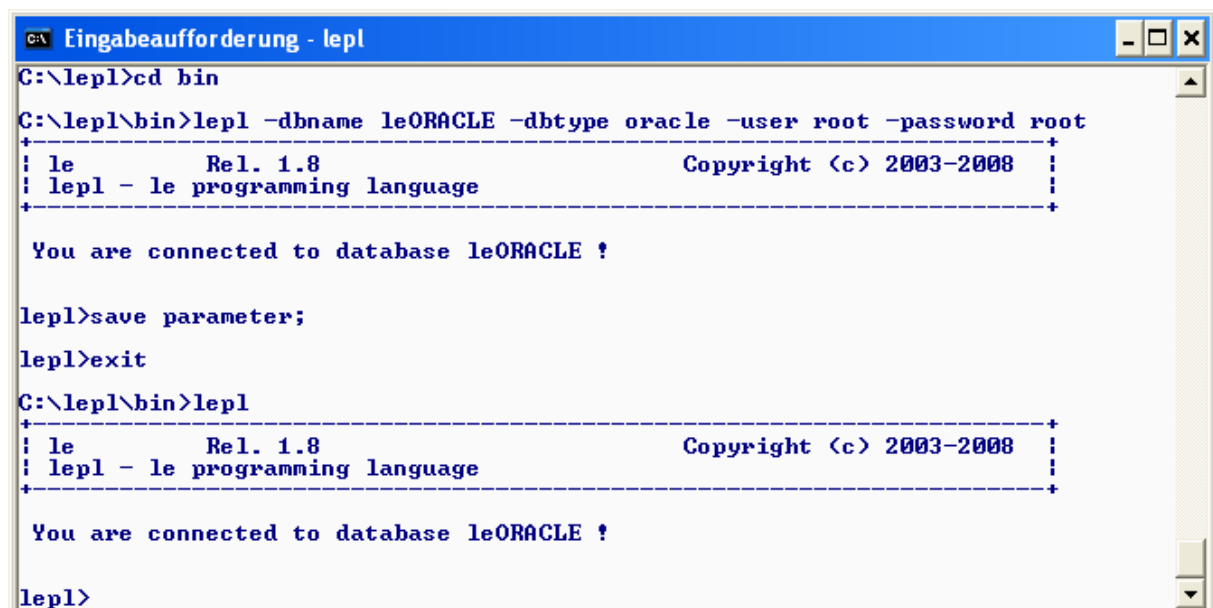
4.1 Connect via amapolisPL standalone

Die benötigten Parameter werden günstig in der Initialisierungsdatei lepl.ini bereitgestellt. Da das DB-Passwort in der ini-Datei verschlüsselt gespeichert sein muss, lässt es sich für einen ersten Zugriff nur unverschlüsselt extern in der Command-Line angeben. Am einfachsten werden alle 4 Parameter beim Aufruf angegeben, damit erübrigt es sich, vorab eine Datei lepl.ini anzulegen.

Starten Sie das Programm lepl immer in einem bin-Pfad!

4.1.1 Datenbank Oracle

Mit **save parameter** speichern Sie die Eingabewerte in der Datei ..lepl.ini. Beim nächsten Aufruf von lepl sind dann keine Parameterangaben mehr notwendig.

A screenshot of a Windows command prompt window titled "Eingabeaufforderung - lepl". The window shows the following sequence of commands and output:

```
C:\>lepl>cd bin
C:\lepl\bin>lepl -dbname leORACLE -dbtype oracle -user root -password root
+-----+
| le      Rel. 1.8                               Copyright (c) 2003-2008 |
| lepl - le programming language                 |
+-----+

You are connected to database leORACLE !

lepl>save parameter;
lepl>exit
C:\lepl\bin>lepl
+-----+
| le      Rel. 1.8                               Copyright (c) 2003-2008 |
| lepl - le programming language                 |
+-----+

You are connected to database leORACLE !

lepl>
```

4.1.2 Datenbank SQL Server

Sie können alternativ im Home-Verzeichnis der Installation (welches das bin-Verzeichnis mit lepl.exe enthalten muss) eine Datei lepl.ini erstellen. Für SQL Server sind dann diese Angaben erforderlich:

```
[LEPL]
LEPL_DBNAME=leSQLSERVER
LEPL_DBTYPE=sql_server
LEPL_USER=root
```

Sie rufen lepl jetzt nur mit DB-Passwort auf uns sichern danach diese Konfiguration.



```
C:\lepl>cd bin
C:\lepl\bin>lepl -password root
+-----+
| le      Rel. 1.8                               Copyright (c) 2003-2008 |
| lepl - le programming language                 |
+-----+

You are connected to database leSQLSERVER !

lepl>save param;
lepl>exit
C:\lepl\bin>lepl
+-----+
| le      Rel. 1.8                               Copyright (c) 2003-2008 |
| lepl - le programming language                 |
+-----+

You are connected to database leSQLSERVER !

lepl>
```

4.1.3 Datenbank MySQL

In gleicher Weise verbinden Sie sich mit einer MySQL-Datenbank. Bei entsprechend geänderten Angaben für den Namen und den Typ der Datenbank wird

```
[LEPL]
LEPL_DBNAME=leMySQL
LEPL_DBTYPE=mysql
LEPL_USER=root
```

```

C:\lepl\bin>lepl -password root
+-----+
| le          Rel. 1.8                Copyright (c) 2003-2008 |
| lepl - le programming language     |
+-----+

You are connected to database leMYSQL !

lepl>save param;
lepl>exit
    
```

Die abgespeicherte vollständige Datei lepl.ini enthält einige weitere Angaben. Neben den Variablen für den hier nicht behandelten Zugriff zur Datenbank über ein amapolis-System (also *amapolisApplikationsServer* und *amapolisDatenbankServer*) finden Sie Angaben zur Format-Steuerung der Ausgaben, zum Message-Niveau und zum Commit.

```

[LEPL]
...
LEPL_DATEFORMAT=dd.mm.yyyy
LEPL_LINESIZE=79
LEPL_PAGESIZE=0
LEPL_TERMOUT=true
LEPL_HEADING=true
LEPL_AUTOCOMMIT=false
LEPL_MESSAGES=true
    
```

4.2 Anweisungen und Programme ausführen

Als einheitliches Werkzeug setzt amapolisPL Datentypen, Funktionen und Anweisungen entsprechend der Notation der Zieldatenbank um. Beispiele zur Syntax:

amapolisPL	Oracle	SQL Server	MySQL
date	date	datetime	datetime
number	number	decimal	decimal
varchar2	varchar2	varchar	varchar
long	long	mediumtext	text
length	length	length	len
substr	substr	substr	substring
nvl	nvl	ifnull	isnull
...where rownum < <i>n</i>	...where rownum< <i>n</i> ...	limit <i>n-1</i>	select top <i>n-1</i> ...

Step 1: Anmeldung an der Datenbank

Starten Sie amapolisPL standalone und melden Sie sich an einer Ihrer Datenbanken an:

```
...>lepl -dbtype sql_server -dbname leSQLSERVER -user root -password root
```

```
+-----+
| le                                           |
| lepl - le programming language              |
+-----+
```

```
You are connected to database leSQLSERVER
```

```
lepl>
```

Step 2: Tabelle anlegen

Legen Sie zum Test eine Tabelle an. Entsprechend dem unten angezeigten Download können Sie das Skript **pa_100.lep** verwenden.

Vorsicht! Im Skript wird eine Tabelle amapolis_test gelöscht, bevor sie neu angelegt wird. Löschen einer Tabelle führt ev. zum Datenverlust!

```
/*=====*/
/* pa_100.lep :                               amapolis IT Services      */
/* -----                               Copyright (c) 2008          */
/*                                           All rights reserved.        */
/*                                           */
Subject: Create Demo Table amapolis_test      */
/*                                           */
/*-----*/
/* Changes                                     */
/*-----*/
/* Rel.   | Date      | Name       | Contents      */
/*-----+-----+-----+-----*/
/*       | 03.03.08 | amapolis  | Origin        */
/*       |          |          |               */
/*=====*/

drop table amapolis_test;

create table amapolis_test
(
  TestId      integer      NOT NULL
  ,TestName   varchar2(20)
  ,TestDate   date
);

/*=====*/
```

Starten Sie pa100.lep.

```
lepl>start "pa_100.lep";

      Table amapolis_test was dropped
      Table amapolis_test was created
```

Step 3: Sätze einfügen

Füllen Sie die Tabelle. Sie können dazu das Skript pa_200.lep verwenden.

```
/*=====*/
/* pa_200.lep :                               amapolis IT Services          */
/* -----                                     Copyright (c) 2008           */
/*                                             All rights reserved.             */
/*                                             */
/* Subject: Insert into Demo Table amapolis_test                          */
/*                                             */
/*-----*/
/* Changes                                     */
/*-----*/
/* Rel.   | Date      | Name      | Contents          */
/*-----+-----+-----+-----*/
/*       | 03.03.08 | amapolis  | Origin           */
/*       |          |          |                  */
/*=====*/

insert into amapolis_test ( TestId, TestName, TestDate )
      values ( 100 , 'Peter' , sysdate );
insert into amapolis_test ( TestId, TestName, TestDate )
      values ( 101 , 'Roger' , sysdate );
insert into amapolis_test ( TestId, TestName, TestDate )
      values ( 102 , 'Jan'   , sysdate );
insert into amapolis_test ( TestId, TestName, TestDate )
      values ( 103 , 'Jane'  , sysdate );
insert into amapolis_test ( TestId, TestName, TestDate )
      values ( 104 , 'Sarah' , sysdate );
insert into amapolis_test ( TestId, TestName, TestDate )
      values ( 105 , 'Jennifer', sysdate );
insert into amapolis_test ( TestId, TestName, TestDate )
      values ( 106 , null    , sysdate );
insert into amapolis_test ( TestId, TestName, TestDate )
      values ( 107 , 'Dana'  , sysdate );
insert into amapolis_test ( TestId, TestName, TestDate )
      values ( 108 , 'Andy'  , sysdate );
insert into amapolis_test ( TestId, TestName, TestDate )
      values ( 109 , 'Sandy' , sysdate );
commit;
```

Starten Sie pa_200.lep.

```
lepl>start "pa_200.lep";

      1 row(s) inserted
      ...
      Commit was executed
```

Step 4: Daten anzeigen

Zeigen Sie die Daten an. Sie können dazu das Skript **pa_300.lep** verwenden. In der select-Anweisung werden dabei einige Datenbankfunktionen angewendet. Der Cursor zeigt die prozeduralen Möglichkeiten von amapolisPL.

```

/*=====*/
/* pa_300.lep :                               amapolis IT Services      */
/* -----                               Copyright (c) 2008          */
/*                               All rights reserved.                  */
/*                               */
/* Subject: Select Data from Demo Table amapolis_test                 */
/*                               */
/*-----*/
/* Changes                               */
/*-----*/
/* Rel.   | Date      | Name       | Contents                               */
/*-----+-----+-----+-----*/
/*       | 03.03.08 | amapolis  | Origin                                */
/*       |         |         |                                         */
/*-----*/

cursor cAtest is
    select testid                               testid
           ,nvl(testname,'XXX')                 n_testname
           ,nvl(length(testname),0)             L_testname
           ,substr(nvl(testname,'XXX'),1,2)     s_testname
           ,testdate                             testdate
    from amapolis_test
    order by testname;

leprint('Id          Name          Length      Date');
leprint('-----');

for ( rAtest in cAtest)
{
    leprint( lerpadd(to_char(rAtest.testid),10) ++ ' ' ++
             lerpadd(rAtest.n_testname, 20) ++ ' ' ++
             lerpadd(rAtest.l_testname, 1) ++ ' ' ++
             lerpadd(rAtest.s_testname, 2) ++ ' ' ++
             to_char(rAtest.testdate,"dd.mm.yy")
    );
}
/* end */

/*=====*/

```

Starten Sie bitte pa_300.lep.

```

lepl>start "pa_300.lep";

```

Id	Name	Length	Date
106	XXX	0 XX	06.03.09
108	Andy	4 An	06.03.09
107	Dana	4 Da	06.03.09
102	Jan	3 Ja	06.03.09
103	Jane	4 Ja	06.03.09
105	Jennifer	8 Je	06.03.09

100	Peter	5	Pe	06.03.09
101	Roger	5	Ro	06.03.09
109	Sandy	5	Sa	06.03.09
104	Sarah	5	Sa	06.03.09

Das Programm **pa_400.lep** verwendet im select keinerlei Funktionen, sondern realisiert die gleiche Funktionalität in der Druck-Aufbereitung mit *amapolisPL*-Funktionen, die einen großen Teil der Leistungsfähigkeit der Programmiersprache ausmachen.

```

/*=====*/
/* pa_400.lep :                               amapolis IT Services      */
/* -----                               Copyright (c) 2008          */
/*                               All rights reserved.                  */
/*                               */
/* Subject: Select Data from Demo Table amapolis_test                  */
/*                               */
/*-----*/
/* Changes                               */
/*-----*/
/* Rel.   | Date      | Name      | Contents                               */
/*-----+-----+-----+-----*/
/*       | 03.03.08 | amapolis  | Origin                               */
/*       |          |          |          */
/*-----*/

integer    iCount;

cursor cAtest is
    select testid
           ,testname
           ,testdate
    from amapolis_test
    order by testname;

leprint('Id          Name          Date');
leprint('-----');

for ( rAtest in cAtest)
{
    leprint(lerpad(to_char(rAtest.testid),10)
           ++      lerpadd(rAtest.testname,20)
           ++      to_char(rAtest.testdate,'dd.mm.yy')
           );
}

leprint('');

select count(*)
    into iCount
    from amapolis_test;

leprint('Number of Records = '++ iCount);

/*=====*/

```

Starten Sie nun pa_400.lep.

```
lepl>start "pa_400.lep";
```

Id	Name	Date
106		06.03.09
108	Andy	06.03.09
107	Dana	06.03.09
102	Jan	06.03.09
103	Jane	06.03.09
105	Jennifer	06.03.09
100	Peter	06.03.09
101	Roger	06.03.09
109	Sandy	06.03.09
104	Sarah	06.03.09

Number of Records = 10

Step 5: Tabelle löschen

```
lepl> drop table amapolis_test;  
exit;
```

5 Beispiele downloaden

Die hier gezeigten Beispiele und lepl.exe halten wir für Sie im Download-Bereich unseres Web-Auftritts **amapolisPL standalone** bereit. Quittieren Sie dort bitte vor dem Download unsere Lizenzbedingungen.

6 Weitere Informationen

Wir haben für Sie hier lediglich ein Einführungsbeispiel zum eigenen Test beschrieben. Zusätzliche Angaben zu amapolisPL finden Sie in weiteren Whitepapers. Sie können auch gern die komplette Sprachbeschreibung amapolisPL anfordern.

7 amapolisPL und Unix

Wenn Sie lepl auf anderen Betriebssystemen einsetzen wollen, wenden Sie sich bitte an info@amapolis.com. Wir schicken Ihnen das Gewünschte gern zu.

8 Feedback

Bitte nehmen Sie sich einen Moment Zeit, um uns Ihre Meinung, Kritiken und gewünschte Verbesserungen oder Erweiterungen mitzuteilen: info@amapolis.com. Dafür bedanken wir uns bei Ihnen im voraus!