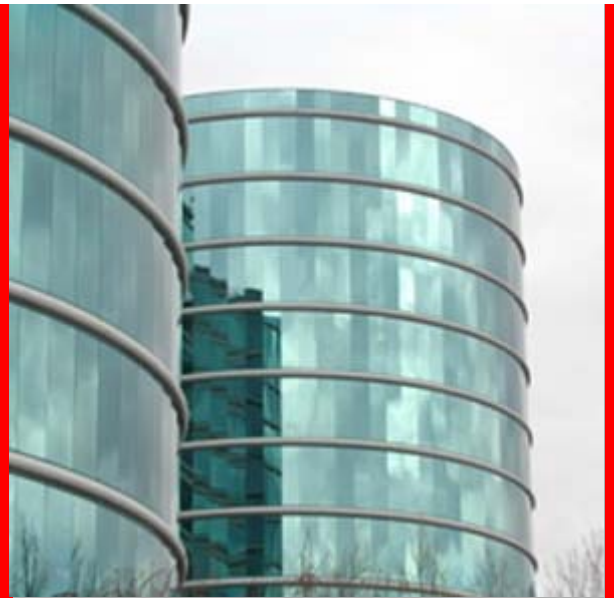


ORACLE®

Breaking limits with the database JVM

Carsten Czarski
Business Unit Database
Oracle Deutschland B.V. & Co KG

How to do ... this one ...?

- Get a directory listing as a virtual table

```
select file_name, file_size, last_modified from table(
 func-read-dir('/oracle/u01/app/oracle/product/11.1.0')
)

 FILE_NAME              FILE_SIZE LAST_MODIFIED
-------------------- ---------- ----------------
racg                         4096 14.10.2007 11:43
owm                          4096 14.10.2007 11:32
samples                      4096 31.05.2006 01:42
network                      4096 14.10.2007 11:46
bin                         12288 01.10.2008 01:53
install.platform               37 01.10.2008 01:32
oc4j                         4096 01.10.2008 01:48
hs                           4096 14.10.2007 11:46
emcli                        4096 01.10.2008 01:37
```

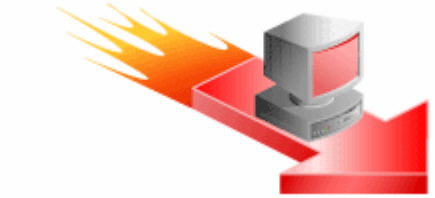
This is also possible ...

- Download a BLOB from an FTP Server

```
select ftp_client.ftp_download (
 'ftp://ftpserv.mydomain.com/path/to/file.jpg')
) content from dual

CONTENT
--------------------------------------------------------
FFD8FFE000104A4649460001000100600060000FFFE001F4C4541
4420546563686E6F6C6F676965732049E632E2056312E303100FF
DB00840008060607060508070707 0A09080A0D160E0D0C0C0D1B

1 row selected.
```



And this ...

- Pack or unpack ZIP archives ...

```
select file_path, compressed_size, file_size
from table(
  zip.list(file_type.get_file(
    '/mnt/hgfs/D/XML-Dateien.zip').get_content_as_blob()
)) where rownum<15;

FILE_PATH                                COMPRESSED_SIZE  FILE_SIZE
---------------------------------------- --------------- ----------
purchaseOrder.xsd                                    811       4960
xml/ADAMS-20011127121040988PST.xml                   514       1172
xml/ADAMS-2001127121104128PST.xml                    579       1599
xml/ADAMS-20011127121041969PST.xml                  1177       6626
xml/ADAMS-20011127121043261PST.xml                  1292       7207
xml/ADAMS-2001127121104402PST.xml                   1337       7441
xml/ADAMS-20011127121044232PST.xml                  1326       7546
xml/ADAMS-20011127121044463PST.xml                  1044       5414
xml/ADAMS-2001127121104492PST.xml                   1032       5093
:                                                      :          :
```

Oracle Database JVM

- Database-embedded JVM
  - Part of the RDBMS kernel – no separate processes
  - Fully J2SE compliant: 100% Java
- Introduced with Oracle8*i*
  - Oracle 9.2: Java 1.3
  - Oracle 10.2: Java 1.4
  - Oracle 11.1: Java 1.5
  - Oracle 11.2: Java 1.5

Oracle JVM: Characteristics

- Threading
  - Java Thread API is supported ... BUT ...
  - ... all threads are being serialized!
  - Do "real" parallel processing with **DBMS_SCHEDULER**
- Graphics: AWT
  - Oracle10*g* Release 2 and higher: *Headless AWT*
  - Provides AWT java methods for image processing

Administering the Oracle JVM
DBMS_JAVA

- Grant or revoke privileges
  - GRANT_PERMISSION, REVOKE_PERMISSION
- Redirect "console" output
  - Oracle8*i*:
    STDOUT redirection to SQL*Plus with SET_OUTPUT
    STDERR is always written to the tracefile
  - Oracle11*g*
    STDOUT and STDERR can be redirected to custom targets
- Misc. Settings
  - Export / Import of Java Bytecode
  - Compiler options
  - System properties

Java Stored Procedures

- Stored procedures or functions: in Java!
- Java Sources or Bytecode
  - Compile outside the database and load bytecode
  - Load Java Source and compile in the database
- Howto
  - SQL: CREATE JAVA SOURCE
  - Command line utilities: loadjava, dropjava

Java Stored Procedures

A simple example

- Load java source: "HelloWorld"

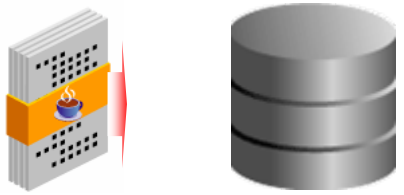
```
create or replace java source named "HelloWorldTest" as

public class HelloWorld {
  public static void sayHello() {
    System.out.println("Hello World");
  }

  public static String getHello(String sName) {
    return "Hello World " + sName;
  }
}
/

alter java source "HelloWorldTest" compile
/
```

Loading Java: loadjava



- What can be loaded ...?
  - Sourcecode (.java)
  - Bytecode (.class)
  - Archives (.jar, .zip)
  - Oracle11*g*: JAR-Archives can be kept as single unit

- Resolving dependencies:
  - The *Resolver* is the databases' CLASSPATH
  - Attention!
    By default Oracle tries to resolve **all** dependencies while loading the java code – Note the "genmissing" option

Loading Java: loadjava

- Load Java classes (bytecode / archives)

```
$ loadjava -u scott/tiger -o -r -v MyJavaProgramm.class

$ loadjava
    -jarsasdbobjects
    -prependjarnames
    -u scott/tiger -o -r -v MyJavaProgramm.jar
```

- Remove java classes from the database

```
$ dropjava -u scott/tiger -o -v MyJavaProgramm.class

$ dropjava -u scott/tiger -o -v MyJavaProgramm.jar
```

The Java code has been loaded ...

- A java object in a database schema ...

```
SQL> select name, source from user_java_classes

NAME                                SOURCE
----------------------------------- -------------------------
HelloWorld                          HelloWorldTest
:                                   :

5 rows selected.
```

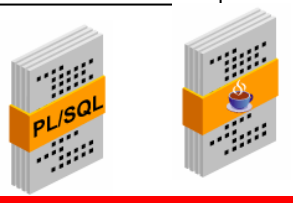
- What now ...?
- How do we call this thing ...?

Java and PL/SQL
Bringing it together

- Generate "PL/SQL Call Specifications" ...
  - Java methods must be "static" – wrapper class might be needed
  - Execution like any other PL/SQL unit

```
create or replace procedure say_hello
is language java name 'HelloWorld.sayHello()';
/

create or replace function get_hello(
  p_name in varchar2
) return varchar2 is language java name
'HelloWorld.getHello(java.lang.String) return java.lang.String';
/
```



Java Stored Procedures

How to access tables, views and other objects

- Usual Java approach: JDBC
- Special JDBC driver
  - *"Server Side Internal Driver"*
  - Connects to the current session
  - Username and password (if given) are ignored

```
Connection con = DriverManager.getConnection(
  "jdbc:default:connection:"
);
```

- Standard JDBC programming afterwards

Type mappings

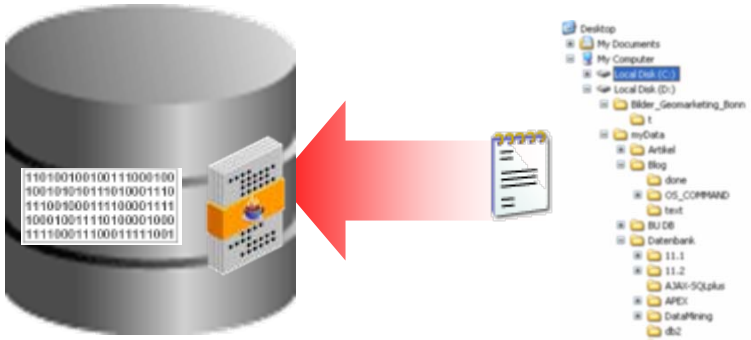
- Mapping of Java types to their SQL pendants
  - Database JVM uses JDBC mappings
  - Complex data types (Arrays, Object Types) are supported
  - "Pure PL/SQL" types (**boolean**, **record**) are *not* supported

Table 6-1 Legal Data Type Mappings

SQL Type	Java Class
NUMBER	oracle.sql.NUMBER
	java.lang.Byte
	java.lang.Short
	java.lang.Integer
	java.lang.Long

Complexe data type mappings
We need those for complex requirements

- An Example: BLOB or CLOB
- Get a file as a BLOB:
  1. **Java**: Create a *temporary LOB* with JDBC
  2. **Java**: Open the file and get the *InputStream*
  3. **Java**: Copy the bytes from InputStream to temporary LOB
  4. **PL/SQL**: Create the *call specification*



Get file contents

Java-Code ...

```
public BLOB getContent(String sFilePath) thr
// Create temporary LOB to return
Connection con = DriverManager.getConnection(
BLOB fileContent = BLOB.createTemporary(con, true, BLOB.DURATION_CALL);

// Buffer for copying the file contents to the temporary LOB
byte[] bBuffer = new byte[fileContent.getChunkSize()];

// Open the Stream objects for copying
OutputStream blobWriter = fileContent.getBinaryOutputStream(0L);
InputStream fileReader = new FileInputStream(sFilePath);

// Do the copying (PSEUDOCODE)
copy(fileReader, blobWriter, bBuffer);

// Close all the handles
fileReader.close();
blobWriter.flush();
blobWriter.close();

// Return the temporary BLOB
return fileContent;
}
```

Returns:
oracle.sql.BLOB

Get file contents

Create and test PL/SQL call specification

- PL/SQL wrapper funktion

```
create or replace function get_file_content(
  p_file_path in varchar2
) return blob
as language java name
'FileContent.getContent(java.lang.String) return oracle.sql.BLOB';
/
```

- The first test ...

```
SQL> select get_file_content('/mnt/hgfs/D/HSE_Produktivitaet.pdf') from dual;

ERROR:
ORA-29532: Java-Aufruf durch nicht abgefangene Java-Exception beendet:
java.security.AccessControlException: the Permission (java.io.FilePermission
/mnt/hgfs/D/HSE_Produktivitaet.pdf read) has not been granted to DOAG. The
PL/SQL to grant this is dbms_java.grant_permission( 'DOAG',
'SYS:java.io.FilePermission', '/mnt/hgfs/D/HSE_Produktivitaet.pdf',
'read' )
```

Security Model

- Java-Engine has its own security model
  - Implementation of "Java2 Security"
  - OS ressources are protected
  - Fine-grained privileges

- Java-Privileges are being checked as *Invokers' Rights*

- JVM superuser role for testing
  - JAVASYSPRIV: "JVM DBA" – everything is allowed

Grant read privilege for a file

- Grant privilege (as DBA)

```
begin
 dbms_java.grant_permission(
   grantee           => 'DOAG',
   permission_type   => 'SYS:java.io.FilePermission',
   permission_name   => '/mnt/hgfs/D/HSE_Produktivitaet.pdf',
   permission_action => 'read'
 );
end;
```

- Next attempt ...

```
select get_file_content('/mnt/hgfs/D/HSE_Produktivitaet.pdf') from dual

GET_FILE_CONTENT('/MNT/HGFS/D/HSE_PRODUKTIVITAET.PDF')
------------------------------------------------------------
255044462D312E340D25E2E3CFD30D0A31362030206F626A203C3C2F4C69
6E656172697A656420312F4C203132333431322F4F2031392F4520313030
3630332F4E20322F542031323330343 52F48205B
```

Privileges: Data Dictionary
USER_JAVA_POLICY

- Privileges granted to the schema user
  - Positive (GRANT) and negative (RESTRICT) Privileges
  - Fine-grained privileges (individual files)
  - Compliant to *Java 2 Security*

KIND	TYPE_NAME	NAME	ACTION
GRANT	java.io.FilePermission	/mnt/hgfs/D/HSE_Produktivitaet.pdf	read
GRANT	java.lang.RuntimePermission	createSecurityManager	
GRANT	java.lang.RuntimePermission	exitVM	
RESTRICT	java.lang.RuntimePermission	loadLibrary.*	
GRANT	java.lang.RuntimePermission	modifyThread	
GRANT	java.lang.RuntimePermission	modifyThreadGroup	
GRANT	java.lang.RuntimePermission	preferences	
GRANT	java.net.SocketPermission	*	connect,resolve
GRANT	java.util.PropertyPermission	*	read
GRANT	java.util.PropertyPermission	user.language	write

Even more complex type mappings

Mapping of arrays and object types

- Note the JDBC mapping for object types
  - oracle.sql.ARRAY
  - oracle.sql.STRUCT
- Create corresponding SQL object types first
  - CREATE TYPE ... AS OBJECT
  - CREATE TYPE ... AS TABLE | VARRAY () OF ...

Even more complex type mappings
How to

- Step 1: Create the SQL object type

```
create type FILE_TYPE as object (
  file_name      varchar2(4000),
  file_size      number,
  last_modified  date,
  is_dir         char(1),
  is_writeable   char(1),
  is_readable    char(1)
);
/
```

Even more complex type mappings
How to

- Step 2: Create the java class

```
public static STRUCT getFileDetails(String filePath) throws Exception{
 Connection con = DriverManager.getConnection(
   "jdbc:default:connection:");
 StructDescriptor sDescr = StructDescriptor.createDescriptor(
   "FILE_TYPE", con);
 File f = new File(filePath);

 Object[] oFileType = new Object[6];
 oFileType[0] = f.getName();
 oFileType[1] = new BigDecimal(f.length());
 oFileType[2] = new java.sql.Timestamp(f.lastModified());
 oFileType[3] = (f.isDirectory()?"Y":"N");
 oFileType[4] = (f.canWrite()?"Y":"N");
 oFileType[5] = (f.canRead()?"Y":"N");

STRUCT   oraFileType = new STRUCT(sDescr, con, oFileType);
 return oraFileType;
}
```

Even more complex type mappings
How to

- Step 3: PL/SQL call specification

```
create or replace function get_file_details(
  p_file_path in varchar2
) return file_type
is language java name
'FileType.getFileDetails(java.lang.String) return oracle.sql.STRUCT';
/
```

```
select get_file_details('/mnt/hgfs/D/HSE_Produktivitaet.pdf')
from dual

FILE_NAME                 FILE_SIZE LAST_MOD IS_DIR IS_READABLE
------------------------- ---------- -------- ------ -----------
HSE_Produktivitaet.pdf        123412 02.10.08 N      Y

1 Zeile wurde ausgewählt.
```

Java Performance

- New in Oracle11*g*: Just-In-Time Compiler
  - Automatic "Native-Compile" *on demand*
  - Forget the NCOMP utility
  - Enabled by default (parameter **`java_jit_enabled`**)

- Compile individual classes with **DBMS_JAVA**
  - DBMS_JAVA.COMPILE_CLASS
  - DBMS_JAVA.COMPILE_METHOD
  - DBMS_JAVA.UNCOMPILE_CLASS
  - DBMS_JAVA.UNCOMPILE_METHOD

What to use: Java or PL/SQL

- Both are reasonable alternatives
  - Today (Oracle11*g*) this also applies to performance:
    - Java: Just-In-Time-Compiler
    - PL/SQL: Native Compilation
- Best approach depends on requirements
  - Java solves problems PL/SQL can't
  - PL/SQL suits, due to simpler type mapping, better for data-centric logic
- Developers' knowhow is also an important factor

More examples

- Load all files in a folder into a table
  http://sql-plsql-de.blogspot.com/2008/09/ein-einfacher-ansatz-dateien-eines.html
- Email Client for the Oracle database
  http://plsqlmailclient.sourceforge.net
- Execute "tkprof" from the SQL layer
  http://sql-plsql-de.blogspot.com/2008/10/aktuelle-session-tracedatei-ansehen-mit.html
- Handle ZIP archives
  http://www.oracle.com/global/de/community/tipps/zip/index.html
- Call web services from the database
  http://www.oracle.com/global/de/community/tipps/webservices-3/index.html

Operating system interaction

File system interaction with SQL and PL/SQL

Packages
FILE_PKG
FILE_TYPE
LOB_WRITER_PLSQL
OS_COMMAND

Overview Deprecated Index Help

File system interaction with SQL and PL/SQL

FRAMES NO FRAMES

SUMMARY: FIELD | METHOD DETAIL: FIELD | METHOD

File system interaction with SQL and PL/SQL

This project is about file system interaction from the Oracle database using SQL and PL_SQL. Four PL/SQL objects are provided for this purpose:

- **FILE_TYPE** represents an operating system file
- **FILE_PKG** is a helper package to obtain one or multiple file handles
- **OS_COMMAND** is a package to execute shell commands
- **LOB_WRITER_PLSQL** is a helper package to write LOBs (CLOB, BLOB) to operating system files with pure PL/SQL and "traditional" directory objects

Usage examples:

1. executing a shell command returning text output (ls -la)

```
SQL> select os_command.exec_clob('/bin/ls -la /home/oracle') COMMAND
  2* from dual

COMMAND
--------------------------------------------------------------------------------

insgesamt 121920
drwx------  20 oracle oracle        4096 18. Jan 09:16 .
drwxr-xr-x   3 root   root          4096 24. Apr 2007  ..
-rw-------   1 oracle oracle           0 24. Apr 2007  .autorun.lck
-rw-------   1 oracle oracle       11067 17. Jan 12:17 .bash_history
-rw-r--r--   1 oracle oracle          24 24. Apr 2007  .bash_logout
-rw-r--r--   1 oracle oracle        1342 13. Nov 11:47 .bash_profile
-rw-r--r--   1 oracle oracle         124 24. Apr 2007  .bashrc
-rwx------   1 oracle oinstall      1583 14. Jan 09:33 calc.sql
drwx------   3 oracle oracle        4096 24. Apr 2007  .config
-rwx------   1 oracle oinstall       161  6. Nov 14:49 csv.txt
drwxr-xr-x   4 oracle oinstall      4096  6. Dez 10:49 dbws
:
```

2. creating a directory

Call external web services

```
SQL> desc stockquote
:
FUNCTION GETQUOTE RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 ARG0                           VARCHAR2                IN
:

SQL> select stockquote.getquote('ORCL') from dual;

STOCKQUOTE.GETQUOTE('ORCL')
--------------------------------------------------------------------------------
<StockQuotes><Stock><Symbol>ORCL</Symbol><Last>21.26</Last><Dat
e>1/16/2008</Date><Time>10:21am</Time><Change>-0.05</Change><Op
en>21.11</Open><High>21.34</High><Low>20.85</Low><Volume>154212
32</Volume><MktCap>109.2B</MktCap><PreviousClose>21.31</Previou
sClose><PercentageChange>-0.23%</PercentageChange><AnnRange>15.
97 - 23.31</AnnRange><Earns>0.914</Earns><P-E>23.32</P-E><Name>
ORACLE CORP</Name></Stock></StockQuotes>
```

More information

- Documentation: Java Developers' Guide
  `http://download.oracle.com/docs/cd/B28359_01/java.111/b31225/toc.htm`
- The presenters' blog
  `http://sql-plsql-de.blogspot.com`

QUESTIONS
ANSWERS