

Q-Checker Datenbankanbindung (jdbc-booking)

Folgende Dateien müssen angepaßt werden

1.) **qcheckerV5.vbs / qcheckerV5 / qcheckerV4**

Im Q-Checker Skript sind die Pfade der Java-Dateien anzupassen. Der Name der JAVA_JDBC-Datei ist abhängig von der verwendeten Datenbank (DB2, Oracle, MySQL, postgresql, ...).

```
QCHECKER_DB_METHOD = "JDBC"
...
QCHECKER_JAVA_CLASSES = "C:\IFOR\JRE1.3.0\JRE\LIB\rt.jar"
QCHECKER_JAVA_BOOK = QCHECKER_LOAD_JAVA & "\Q-CheckerDB.jar"
QCHECKER_JAVA_JDBC = QCHECKER_LOAD_JAVA & "\postgresql.jar "
QCHECKER_JAVA_CALL = "C:\IFOR\JRE1.3.0\JRE\BIN\javaw.exe"
```

2.) **CATIA-Q-Checker-Environment.txt / QCHECKER.dcls**

Q-Checker V5

In der CATIA-Q-Checker-Environment.txt muß die Variable QCLICDB so gesetzt werden, dass die Datenbanklizenz (TC-qcheckerV5-DB) angezogen wird.

```
QCLICDB=YES
```

Q-Checker V4

In der QCHECKER.dcls muß die Variable QCLICDB so gesetzt werden, dass die Datenbanklizenz (TC-qchecker-all-DB) angezogen wird.

```
alias QCLICDB = catia.QCHECKER_LICENSE_DATABASE='YES';
```

3.) **QCHECKER.par**

In der Q-Checker-Konfigurations-Datei ist einzutragen, dass die xml-Reports zum Einbuchen der Daten in die Datenbank erzeugt werden

```
qchecker.DB_CONNECT_BATCH          YES
...
qchecker.DB_CONNECT_INTERACTIVE    YES
```

4.) **QCHECKER.db**

Die Datenbank-Konfigurations-Datei ist auf die verwendete Datenbank anzupassen.

```
# Define database type
qchecker.DB_TYPE          POSTGRESQL

# Define user with write access to database
qchecker.DB_USER          qchecker

# Define password of user with write access to database
qchecker.DB_PASSWORD      qchecker

# Define address of database server
qchecker.DB_SERVER        pcprog3

# Define database name
qchecker.DB_NAME          qc_v5

# Define port for remote database access
qchecker.DB_PORT          5432
```

Möglichkeiten der Kontrolle:

- Lizenz:

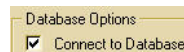
Kann Q-Checker interaktiv gestartet werden?

Info: Die Lizenz wird beim Start des Q-Checker abgefragt. Ist die Lizenz nicht vorhanden wird der Start abgebrochen und eine entsprechende Fehlermeldung ausgegeben. Im Umkehrschluß: kann der Q-Checker gestartet werden, ist auch die Lizenz vorhanden und verfügbar.

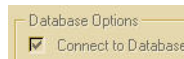
Können bei den Batch-Einstellungen die Datenbank Optionen aktiviert werden?

Q-Checker V5

Info: Das Anziehen der Datenbanklizenz wird in der CATIA-Q-Checker-Environment.txt gesteuert. Ist die Variable „QCLICDB=YES“ gesetzt, können in den Batch-Einstellungen die Datenbank Optionen aktiviert werden. Die Datenbanklizenz ist verfügbar.



Ist die Variable „QCLICDB=NO“ gesetzt können in den Batch-Einstellungen die Datenbank Optionen nicht aktiviert werden. Die Datenbanklizenz ist nicht verfügbar.



Q-Checker V4

Analog Q-CheckerV5. Das Anziehen der Datenbanklizenz wird in der QCHECKER.dcls gesteuert.

- QCHECKER-par

Werden die xml-Reports geschrieben?

Info: Ist die QCHECKER.par entsprechend konfiguriert - und kann die Datenbanklizenz gezogen werden, s.o. - wird nach jeder interaktiven und / oder Batch -Prüfung eines Modells, zusätzlich zum qcreport-Report, auch das xml-Report geschrieben.


Das xml-Report enthält alle, zum Einbuchen in die Datenbank, relevanten Informationen. Dies sind neben den Daten selbst auch die Informationen aus der QCHECKER.db.

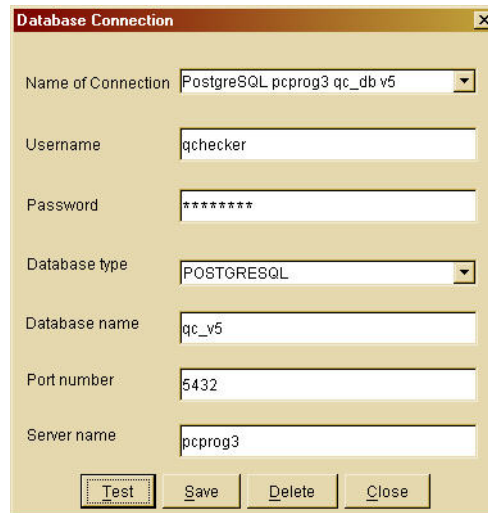
Speicherort ist das Report-Verzeichnis, dies wird in der CATIA-Q-Checker-Environment.txt durch die Variable „QCREPORT=Pfad“ definiert, bzw. in the QCHECKER.dcls durch die Variable „alias QCREPORT = catia.QCHECKER_REPORT='Pfad““.

Bem.: Im Gegensatz *.xml ist das *.qcanalysis.xml nicht zum Einbuchen in die Datenbank vorgesehen, sondern zur Ansicht im Q-Checker Analysis Viewer.

- QCHECKER.db

Sind die Verbindungsinformationen korrekt?

Info: Im Q-Monitor können über den Schalter „connect“  die Datenbankinformationen angezeigt werden.



Die Informationen im DB_INFO-Block des xml-Reports müssen mit denen der Database Connection des Q-Monitors übereinstimmen.

```
<DB_INFO>
  <DB_TYPE> 'POSTGRESQL' </DB_TYPE>
  <DB_USER> 'qchecker' </DB_USER>
  <DB_PASSWORD> 'qchecker' </DB_PASSWORD>
  <DB_SERVER> 'pcprog3' </DB_SERVER>
  <DB_NAME> 'qc_v5' </DB_NAME>
  <DB_PORT> '5432' </DB_PORT>
</DB_INFO>
```

Ist der DB_INFO-Block leer, so kann dies zwei Gründe haben.

- Die QCHECKER.db ist nicht vorhanden.
- Beim Speichern der angepassten Datei wurde von Editor ein .txt angehängt. Einige Editoren machen dies automatisch. Die Datei QCHECKER.db.txt wird von Q-Checker nicht erkannt. In beiden Fällen erzeugt der Q-Checker eine neue (leere) QCHECKER.db im db-Verzeichnis.

- qcheckerV5.vbs

Wird die richtige JAVA_JDBC-Datei verwendet?

Info: Die JAVA_JDBC-Datei ist abhängig von der verwendeten Datenbank. Daher eignet sich an dieser Stelle der Q-Monitor zum Testen der Datei. Im Fenster „Database Connection“ (siehe vorheriger Test) den Schalter „Test“ klicken. Erscheint die Meldung mit dem Text „Connection OK“ ist die in der qmonitor.bat eingetragene Datei korrekt.



qmonitor.bat

```
start C:\Programme\JavaSoft\JRE\1.3.1\bin\java.exe -classpath
.\qmonitor.2.5.8.jar;postgresql.jar qmon.QMonitor .\qmonitor.ini
```

Eine Kopie dieser Datei ist im Q-Checker-load-Verzeichnis zu speichern. In der qcheckerV5.vbs (bzw. qcheckerV5, qcheckerV4) ist der Eintrag QCHECKER_JAVA_JDBC anzupassen.

QCHECKER_JAVA_JDBC = QCHECKER_LOAD_JAVA & "\postgresql.jar "