

# 4D Server Handbuch

-  Einleitung
-  4D Server in 10 Minuten
-  4D Server verwenden
-  4D Server Verwaltungsfenster
-  Datenbankmethoden
-  Remote 4D verwenden
-  4D Server und Programmiersprache
-  Alphabetische Liste der Befehle

# **Einleitung**

 Überblick

 4D Server Architektur

## Überblick

---

4D Server ist der multi-user, cross-plattform Daten- und Anwendungsserver für 4D. Mit 4D Server können Sie multi-user Datenbanken und eigene Anwendungen in einer Client/Server-Architektur erstellen und einsetzen. Die plattformunabhängige Client/Server Architektur verwaltet nahtlos Anwendungen für remote 4D unter Windows und auf Macintosh. 4D Server bietet professionell ausgerichtete Entwicklungswerkzeuge, volle Skalierbarkeit, Datensicherheit und Connectivity Optionen für Unternehmenssysteme.

4D Server liefert eine vollständig integrierte Architektur, wobei Client und Server auf dieselbe 4D Applikation zugreifen. 4D Server befreit Entwickler von der Notwendigkeit, separate front-end und back-end Applikationen einzurichten. Darüberhinaus ist 4D Server ein "zero admin" Server. Er lässt sich leicht installieren, einsetzen und verwalten und ist äußerst kosteneffektiv.

4D Server füllt die Lücke zwischen low-end File-sharing basierten Systemen und komplexen SQL-basierten RDBMS. Jede 4D Server Applikation interagiert mühelos mit vorhandenen Unternehmensdatenbanken wie Oracle, Sybase, oder jedem ODBC kompatiblen Server. 4D Server erfüllt die Ansprüche von Workgroups in Unternehmen jeglicher Größe.

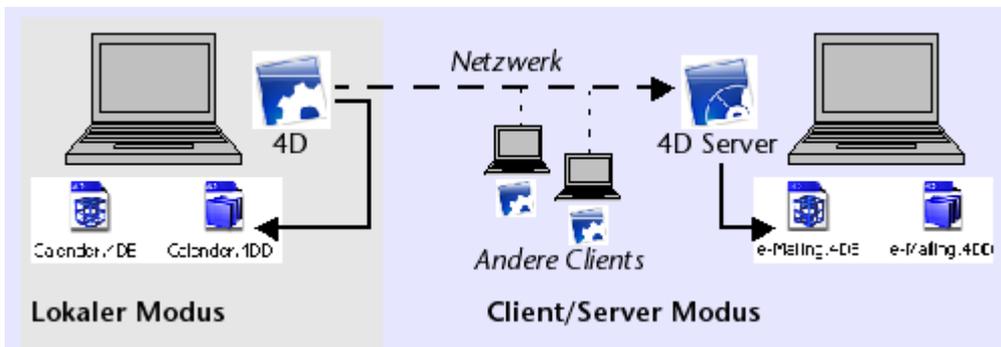
### **Integrierte Back-end und Front-end Architektur**

---

Mit 4D Server gibt es keinen Unterschied zwischen front-end und back-end Anwendungen. Client Software und Server Anwendung sind zwei Seiten desselben Produkts, nämlich 4D. Die 4D Server Anwendung selbst ist in zwei Teile gegliedert – 4D Server und 4D im remote Modus – diese entsprechen den Elementen der Client/Server Architektur.

4D Server, der Motor der Datenbank, liegt auf dem Server-Rechner. Er speichert und verwaltet die Datenbank auf dem Server und erlaubt Benutzern, von ihren eigenen Rechnern (Clients) aus in der Datenbank zu arbeiten.

Die 4D Anwendung liegt auf jedem Client-Rechner. Sie kann im lokalen oder im remote Modus verwendet werden. Im lokalen Modus arbeiten Benutzer mit einer Datenbank oder einer 4D Anwendung, die lokal auf Ihrem Rechner gespeichert ist. Im remote Modus können Benutzer auf die Datenbank auf dem Server zugreifen und verschiedene Operationen ausführen. Sie können Daten hinzufügen, Berichte erstellen und das Design der Datenbank verändern. Alles, was mit 4D im lokalen Modus möglich ist, kann genauso mit 4D Server und 4D im remote Modus ausgeführt werden.

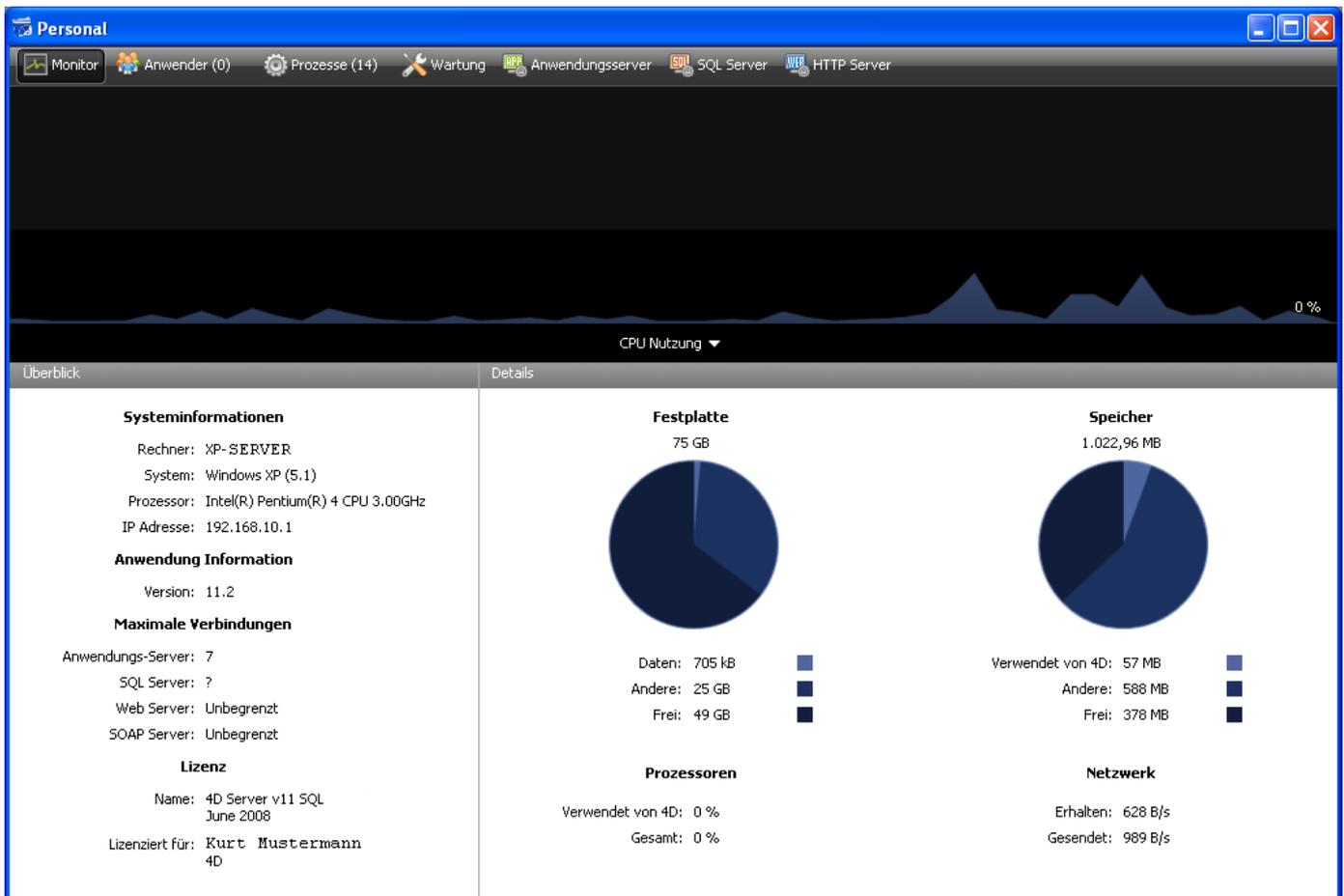


Weder zusätzliche Middleware noch weitere Entwicklungsphasen sind für den Einsatz in einer Client/Server-Umgebung erforderlich. 4D und 4D Server nutzen dieselben Tools für die Benutzeroberfläche, dieselbe Sprache und dasselbe Datenmanagement-System.

Jede lokale Anwendung, die für eine Plattform (Windows oder Macintosh) erstellt wurde, kann mühelos auf eine Client/Server-Lösung für Arbeitsgruppen skaliert werden. Genauso ist umgekehrt die Abwärtsskalierung einer mit 4D Server generierten Anwendung auf eine eigenständige Anwendung im lokalen Modus möglich.

## Wartungsfreier Daten- und Anwendungs-Server

4D Server profitiert von dem hohen Maß an Benutzerfreundlichkeit in 4D. 4D Server ist ein echtes Plug&Play-System (PNP).



### 4D Server Verwaltungsfenster

- **Grafisch übersichtlich zusammengefasste Informationen:** Das Verwaltungsfenster von 4D Server zeigt automatisch alle wichtigen Informationen an: der für 4D Server reservierte Speicher, der Daten-Cache,

Anzahl und Name der angemeldeten Benutzer, Anzahl und Status der laufenden Prozesse, Überwachung von Backups und Anfragen, sowie der gesamten Serveraktivität.

- **Automatische Konfigurierung und Skalierbarkeit:** 4D Server ist so konzipiert, dass das Hinzufügen neuer Clients und neuer Plug-Ins sowie für die Anbindung an andere Systeme ohne Neukonfigurierung oder Neuprogrammierung möglich ist.
- **Automatische und dynamische Aktualisierung sowie Versionskontrolle von Client-Workstations:** Alle angemeldeten 4D Rechner werden automatisch und dynamisch aktualisiert, wenn die Datenbank modifiziert oder ein Plug-In bzw. eine Komponente hinzugefügt, entfernt oder geändert wird. Haben Sie eine eigene Client/Server Anwendung erstellt, können Sie auch automatisch neue Versionen der ausführbaren Client-Anwendungen laden, wenn die 4D Server Anwendung aktualisiert wird.
- **Automatische asynchrone Verbindungen über das Standardprotokoll TCP/IP:** 4D Server und 4D kommunizieren über das TCP/IP Protokoll transparent miteinander, unabhängig auf welcher Plattform Client und Server laufen. Das TCP/IP Protokoll ist in alle Betriebssysteme integriert, so dass keine zusätzliche Installation notwendig ist.
- **Simultane Verwaltung von Sitzung und Status der SQL und HTTP Verbindungen:** 4D Server erstellt und verwaltet automatisch eine aktuelle Arbeitsumgebung für jede Tabelle-/Prozess-/Benutzer-Kombination. Diese sitzungsorientierte Architektur macht es möglich, dass jeder Prozess Daten für jeden einzelnen Benutzer unabhängig und simultan verwalten kann. Der SQL Server von 4D Server verwaltet automatisch interne oder externe SQL Anfragen. Der HTTP Server in 4D Server antwortet auf HTTP Anfragen und auf SOAP Anfragen, wenn er aktiviert ist.
- **Automatische Datensatzsperrung:** 4D Server ermöglicht das automatische Sperren und Freigeben von Datensätzen, wodurch Probleme vermieden werden, die häufig durch das Ändern gerade verwendeter Objekte entstehen. Das Sperren von Datensätzen verhindert ebenfalls Probleme, die beim Sperren von Seiten oder Dateien auftreten können.
- **In der Benutzeroberfläche integriertes Meldungssystem:** Die ursprünglich desktop-orientierte 4D Server Software umfasst sämtliche Aspekte einer Bedienerführung, vereint mit modernen, integrierten Entwicklungsumgebungen. So kann 4D Server Clients über Wartungsarbeiten am Server informieren, etwa über geplante Unterbrechungen, Backups und vieles mehr.
- **Automatisierte Start- und Abmeldemethoden:** 4D Server löst automatisch fünf Datenbankmethoden als Antwort auf spezifische Ereignisse aus: **On Server Startup, On Server Shutdown, On Server Open Connection, On Server Close Connection** und **On Web Connection**. Die **Datenbankmethode On Server Startup** kann automatisch alle Objekte initialisieren und laden, die während der Sitzung benötigt werden.

## **Ein unschlagbares Leistungsspektrum**

---

Zusätzlich zu den Leistungsmerkmalen von 4D bietet 4D Server folgende Funktionalitäten:

- **Datenverwaltung im Mehrplatzbetrieb:** Mehrere Benutzer können gleichzeitig Operationen in der Datenbank ausführen. Sie können z.B. in Datensätzen, die in der gleichen oder in verschiedenen Dateien liegen, zur gleichen Zeit Datensätze hinzufügen, ändern, löschen, sortieren oder ausdrucken. Das integrierte System zum Sperren gerade verwendeter Datensätze sorgt dafür, dass die Datenintegrität gewahrt bleibt.
- **Entwicklung im Team:** Mehrere Entwickler können gleichzeitig eine Datenbank einrichten und strukturieren. So können z.B. mehrere Benutzer zur selben Zeit Dateidefinitionen bearbeiten, Layouts, Skripte und Methoden anlegen und verändern. Das integrierte System zum Sperren gerade verwendeter Objekte und die Versionskontrolle über ein Check-in/Check-out-System sorgen für die Integrität Ihrer Datenbank.
- **Plattformunabhängige Client /Server Architektur:** 4D Server steuert die Datenbank für Clients unter Windows und auf Mac OS gleichermaßen. Dazu gehört die simultane und plattformübergreifende Entwicklung im Team ebenso wie eine transparente Schnittstelle zu allen eingegebenen bzw. geänderten Daten auf den Arbeitsstationen in einer heterogenen Hardware Umgebung.
- **64-bit Versionen:** 4D Server ist unter Windows seit Version 12.1, auf Mac OS seit Version 15.1 als 64-bit Version verfügbar. Die 64-bit Architektur ermöglicht, Ihren Anwendungen mehr RAM zuzuweisen.
- **4D Plug-In Architektur für Windows und Mac OS:** In der Windows und Mac OS Version von 4D Server können Sie sowohl Plug-Ins für Windows als auch für Mac OS installieren. So lassen sich plattformunabhängige 4D Plug-Ins schnell und problemlos zwischen 4D Server und 4D aufteilen, unabhängig von der Plattform der Arbeitsstation.
- **Integrierter HTTP Server:** Wie 4D im lokalen Modus enthalten auch 4D Server und 4D im remote Modus eine HTTP Engine, mit der Sie 4D Datenbanken im Web publizieren können. Sie müssen dafür weder ein Datenbanksystem, noch eine Web Site entwickeln, noch eine CGI Schnittstelle zwischen beiden einrichten. Ihre Datenbank ist Ihre Web Site. Sie können auch jede Arbeitsstation in einen Web Server umwandeln. Weitere Informationen dazu finden Sie im Handbuch *4D Programmiersprache* im Abschnitt **[SORT ARRAY](#)**.
- **Sicherheit der Verbindung:** Sie können 4D Server Verbindungen verschlüsseln. Die "klassische" Client/Server Architektur kann die TLS/SSL Verschlüsselungssysteme verwenden. Weitere Informationen dazu finden Sie im Abschnitt **[Client/Server Verbindungen verschlüsseln](#)**.
- **Trigger:** Ein Trigger ist eine Methode, die einer Tabelle zugewiesen ist. Demzufolge ist es eine Tabelleneigenschaft. Trigger rufen Sie nicht selbst auf, sie werden automatisch von der 4D Datenbank Engine ausgelöst, wenn Sie Datensätze in der Tabelle bearbeiten, d.h. Datensätze hinzufügen, ändern und löschen. Mit 4D Server werden Trigger auf dem Server-Rechner ausgeführt. Die über Trigger festgelegten Regeln für die Datenbank gelten für alle Arbeitsstationen, egal, ob es sich um 4D oder um eine über ODBC angemeldete Anwendung handelt. Weitere Informationen dazu finden Sie im Handbuch *4D Programmiersprache* im Abschnitt **[ARRAY REAL](#)**.
- **Serverprozeduren:** Sie können 4D Methoden erstellen, die in einem eigenen Prozess lokal auf dem Server-Rechner laufen. In der Client/Server Terminologie wurde dafür der Begriff "Serverprozedur" eingeführt. 4D Server dagegen bietet

eine Architektur, die weit über das gängige Konzept von Serverprozeduren hinausgeht. Mit 4D Server ist eine Serverprozedur ein eigener Server Prozess, mit dem Ihr Code asynchron und unabhängig von allen anderen Prozessen auf dem Server- oder Client-Rechner läuft. In der herkömmlichen Client/Server Architektur führt eine Serverprozedur eine Anweisung aus und gibt ein Ergebnis synchron oder asynchron zurück. Mit 4D Server können Sie eine Serverprozedur starten, die während einer ganzen Client/Server Sitzung läuft und Anfragen von Clients beantwortet. Sie können auch genauso eine Serverprozedur einrichten, die nicht mit Clients interagiert, sondern Daten mit einem SQL-Server oder einem anderen 4D Server abgleicht. Die Anzahl gleichzeitig ablaufender Serverprozeduren wird nur durch Hardware und Speicher begrenzt. Eine 4D Serverprozedur läuft in ihrem eigenen Prozess und kann so wie jeder andere Benutzerprozess ein individuelles Datenbankkontext haben, z.B. aktuelle Auswahlen. Die 4D Programmiersprache bietet außerdem Befehle, mit denen Client-Prozesse die Prozessvariablen jeder beliebigen Serverprozedur (einschließlich BLOB Variablen) lesen und schreiben können. Dadurch ist eine ausgeklügelte und flexible Kommunikation zwischen Clients und Serverprozeduren möglich. Schließlich können Sie mit Serverprozeduren auch neue und eigene Dienste in 4D Server hinzufügen. Weitere Informationen dazu finden Sie im Abschnitt **Serverprozeduren**.

- **Auf Client ausgeführte Serverprozeduren:** Mit 4D Server können Sie von einem Server- oder Client-Rechner aus Serverprozeduren auf einem oder mehreren anderen Client-Rechnern ausführen. So können Sie die Verteilung des Arbeitsaufkommens optimieren oder Anwendungen erstellen, welche die Kommunikation zwischen den Clients ermöglicht. Weitere Informationen dazu finden Sie im Abschnitt **Serverprozeduren**.
- **Server Pfad:** Sie können den Pfad auf eine Serverdatenbank mit einem Benutzerkennwort sichern. Der Benutzer kann sich durch Doppelklick auf ein 4DLink Dokument an die Serverdatenbank anmelden. Weitere Informationen dazu finden Sie im Abschnitt **Anmeldung an eine 4D Server Datenbank**.
- **Registrieren als Service:** 4D Server lässt sich unter Windows und auf Mac OS als Service starten.
- **Integriertes Backup System:** 4D Server enthält ein komplettes Backup-Modul mit automatischer Restore-Funktion. Damit können Sie eine Datensicherung im laufenden Betrieb ausführen, ohne die Anwendung zu beenden. Backups lassen sich in regulären Zeitabständen und ohne Einwirken des Benutzers automatisch oder manuell starten. Sie können auch festlegen, dass die Datenbank im Störfall automatisch wiederhergestellt bzw. neu gestartet wird.
- **Backup durch logisches Spiegeln:** Bei kritischen Anwendungen, die z.B. 24 Stunden laufen müssen, können Sie ein Backup durch logisches Spiegeln einrichten. Tritt ein Zwischenfall in der laufenden Datenbank ein, können Sie sofort mit dem Duplikat der Datenbank weiterarbeiten.
- **Connectivity Plug-Ins:** Mit den 4D Connectivity Plug-In (4D ODBC Pro, etc.) können 4D Server und 4D direkt auf andere Systeme zugreifen, z.B. ORACLE oder andere Datenbankserver, die auf ODBC basieren. Zwischen diesen Datenbanken lässt sich Information interaktiv gemeinsam nutzen. Darüberhinaus bietet 4D einen ODBC Treiber für 4D Server, über den ein beliebiger ODBC Client sich an 4D Server anmelden und damit arbeiten kann.

## 4D Server Architektur

---

Die 4D Server-Architektur ist leistungsstark, weil sich Arbeitsstationen und Server die Arbeit teilen.

Der Server speichert und verwaltet die Daten. Die Arbeitsstationen schicken dem Server Anfragen und erhalten von diesem die Ergebnisse.

Um z.B. nach einer Reihe von Datensätzen zu suchen, sendet die Arbeitsstation eine Suchanfrage an den Server. Der Server führt bei Empfangen der Anfrage die Suchoperation lokal auf dem Server-Rechner aus und sendet nach Beenden der Suche das Ergebnis, d.h. die gefundenen Datensätze an die Arbeitsstation zurück.

Die Architektur von 4D Server basiert auf dem Client/Server Modell. Schon seit vielen Jahren hat sich diese Architektur gegenüber dem alten Gegenspieler, der File Sharing Architektur, durchgesetzt und ist zum effizientesten Modell für Anwendungen im Mehrplatzbetrieb geworden.

4D Server gleicht den Client/Server-Architekturen, die in der Welt der Minicomputer eingesetzt werden. Es gibt jedoch zwei signifikante Unterschiede:

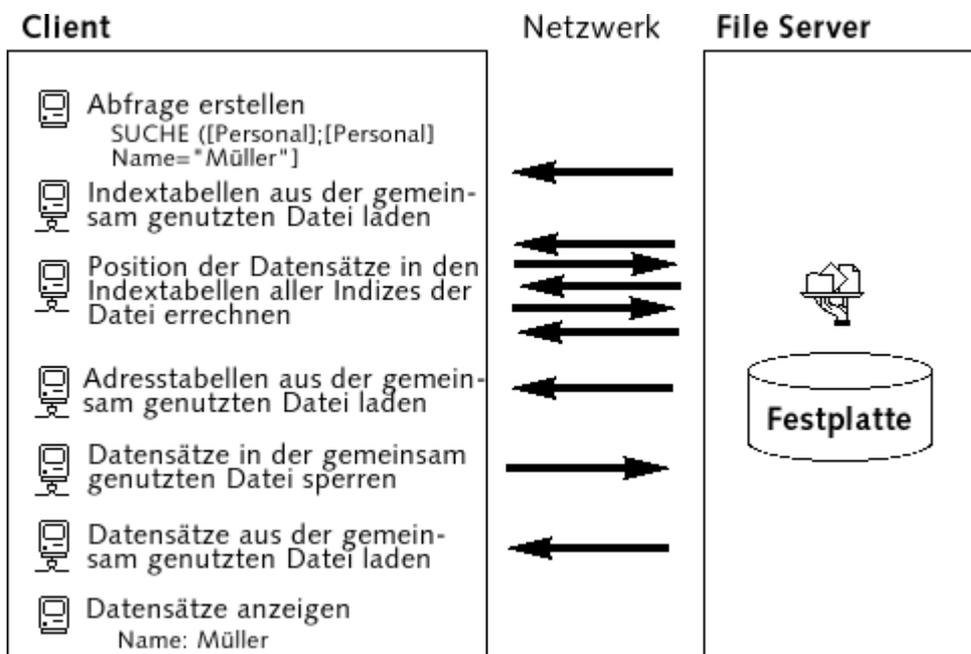
- Die benutzerfreundliche, grafische Oberfläche auf allen Ebenen der Datenbank
- Eine integrierte Architektur, die für gesteigerte Leistung und Geschwindigkeit sorgt.

### **File Sharing Architektur (nicht bei 4D Server)**

---

Vor Einführung der Client-/Server-Architektur verwendeten Systeme im Mehrplatzbetrieb das File Sharing Modell der Netzwerk-Architektur. In diesem Modell nutzen alle Benutzer denselben Datenbestand, die Daten werden jedoch nicht über eine zentrale Datenbank-Engine verwaltet. Jede Arbeitsstation muss eine Kopie der Datenbankstruktur und -Engine speichern, während der Server nur die Software steuert, die zum gemeinsamen Nutzen der Dateien auf dem Netzwerk erforderlich ist.

Im File Sharing Modell führt jede Arbeitsstation alle Änderungen an den Daten lokal aus. Das führt zu massivem Datenverkehr im Netz, da jede Anfrage aus unzähligen Netzwerkübertragungen besteht. Folgendes Beispiel zeigt den Datenverkehr im Netz, wenn ein Benutzer in der Datenbank nach allen Personen mit dem Nachnamen Müller sucht:



Ein weiterer Nachteil ist, dass die Daten nicht in einem gemeinsamen Cache-Speicher abgelegt werden können, denn dann könnten vom gleichen Datensatz verschiedene Versionen existieren, was zu Dateninkonsistenz führen würde. Da jede Station einen eigenen Cache-Speicher verwendet, wissen die Stationen nicht, welcher Teil des Datensatzes sich gerade bei einer Station befindet. Der Server bleibt passiv, d.h., er speichert die Daten, anstatt sie aktiv zu verwalten. Er unterstützt die Benutzer nicht. Sie müssen selbst die Daten heranziehen und auf der Station bearbeiten. Von daher herrscht im Netz ein ständiger Datenverkehr.

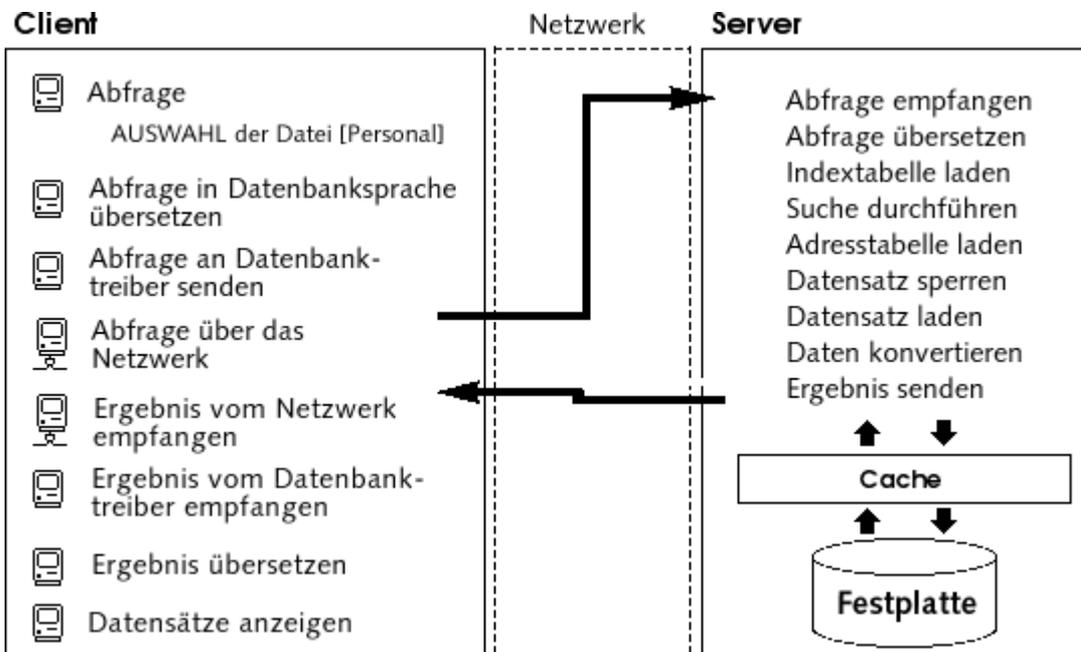
## Heterogene Client/Server Architektur

In der Welt der Minicomputer ist die Client/Server Architektur wegen ihrer Leistungsstärke und Geschwindigkeit für umfangreiche Datenbanksysteme weit verbreitet. In dieser Architektur ist die Arbeit zwischen dem Server-Rechner und den Clients aufgeteilt, was die Performance entscheidend verbessert.

Der Server enthält die zentrale Datenbank-Engine, welche die Daten speichert und verwaltet. Die Engine ist die einzige Software, die auf die Daten zugreift, die auf der Festplatte gespeichert sind. Sendet ein Client eine Anfrage an den Server, sendet der Server das Ergebnis zurück. Das kann ein spezifischer Datensatz sein, den der Client ändert oder eine sortierte Liste von Datensätzen.

Im allgemeinen werden die meisten Client/Server Architekturen auch heterogene Architekturen genannt, da die Front-end Anwendungen, die auf den Client-Rechnern laufen und die Datenbank-Engine, die auf dem Server-Rechner läuft, zwei vollkommen unabhängige Einheiten sind. Sie brauchen daher einen Datenbank-Treiber, der als Übersetzer zwischen Arbeitsstation und Server dient.

Um beispielsweise nach einem Datensatz zu suchen, sendet der Client eine Anfrage an den Server. Da die Datenbank auf dem Server liegt, führt der Server den Befehl lokal auf dem Server-Rechner aus und sendet das Ergebnis an den Client-Rechner. Folgende Abbildung zeigt den Datenverkehr im Netz, wenn ein Benutzer den Server beauftragt, nach allen Personen mit dem Nachnamen Müller zu suchen und anschließend die Datensätze anzuzeigen:



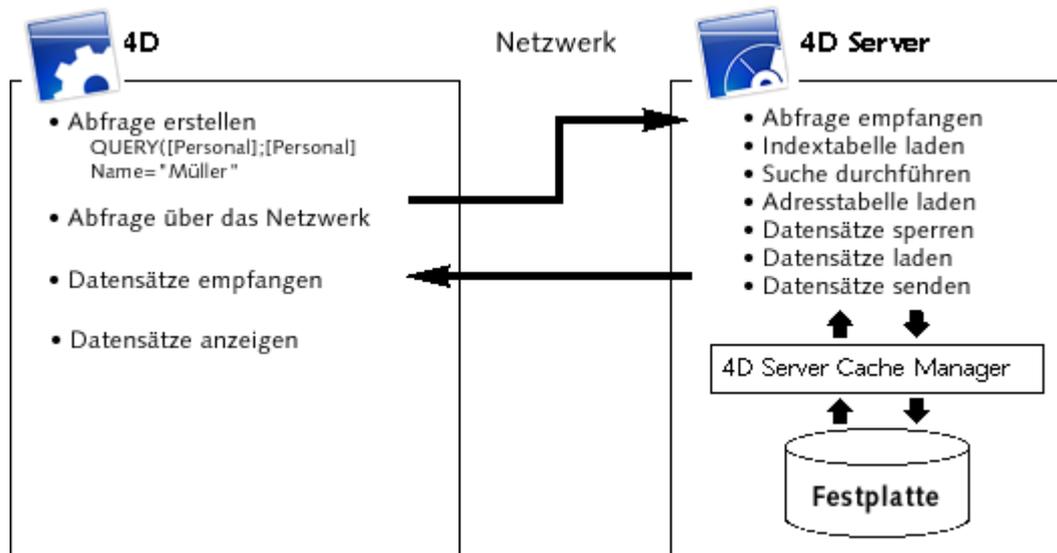
Dieses Beispiel zeigt zwei entscheidende Unterschiede zwischen File Sharing und dem Client/Server Betrieb:

- **Die Client/Server Architektur arbeitet mit Cache-Speicher:** Da nur die Engine real auf die Daten zugreift, kann der Server einen Cache-Speicher betreiben, der geänderte Datensätze speichert, bis sie auf die Festplatte geschrieben werden. Die Daten werden von einer zentralen Stelle gesendet, so dass die Arbeitsstationen immer die aktuellste Version des Datensatzes empfangen. So ist die Datenintegrität gewährleistet. Darüberhinaus werden Operationen in der Datenbank durch Verwendung des Cache-Speichers beschleunigt, da der Zugriff nicht über die Festplatte, sondern über den Cache läuft. In File Sharing Architekturen wird immer auf die Festplatte zugegriffen.
- **Datenbankoperationen im Hintergrund laufen auf dem Server ab:** Die Client/Server Architektur bietet deutlich höhere Geschwindigkeit, da Datenbankoperationen, wie Erstellen eines Index oder Sortieren von Tabellen, auf dem Server Rechner mit dessen Rechengeschwindigkeit ausgeführt werden. In File Sharing Architekturen laufen die gleichen Operationen durch Netzwerkübertragungen und Grenzen der Arbeitsstation viel langsamer ab.

## Zusammenarbeit von Arbeitsstation und Server

Bei herkömmlichen Client/Server-Architekturen besitzt der Server keine Informationen über die Umgebung der Arbeitsstation. Die Daten vom Server sind Rohdaten, die die Arbeitsstation interpretiert, z.B. mit übergebenen Arrays. Bei 4D Server arbeiten Server und Arbeitsstation mit derselben Datenstruktur. Die Daten, die beide ans Netz senden, sind bereits strukturiert. Das heißt, Arbeitsstation

und Server können sie direkt lesen.



Die Arbeitsstation ist zuständig für:

- **Anfragen:** Der Client sendet Anfragen an den 4D Server. Sie können Anfragen über die integrierten Such- und Sortiereditoren erstellen, per Programmierung oder via SQL. 4D bietet Editoren zum Erstellen und Ändern von Methoden und verwaltet Variablen und Arrays.
- **Antworten:** Der Client empfängt Antworten vom 4D Server und aktualisiert die Daten auf der Arbeitsstation. Fragt der Client z.B. nach allen Datensätzen mit dem Nachnamen "Müller", erhält er diese Datensätze vom Server und zeigt sie in einem Formular an.

Der Server ist zuständig für:

- **Zugriffe:** 4D Server verwaltet über Multitasking konkurrierende Zugriffe und alle Prozesse, die von Clients erzeugt werden.
- **Struktur und Datenobjekte:** 4D Server speichert und verwaltet sämtliche Objekte der Datenbank, d.h. Daten (Datensätze, aktuelle Auswahl) bzw. Strukturobjekte (Tabellen, Datenfelder, Formulare, Methoden und Menüs).
- **Cache:** 4D Server unterhält einen Cache, der Datensätze sowie spezifische Datenobjekte enthält, wie z.B. Auswahlen und Mengen einzelner Clients.
- **Low-level Datenbankoperationen:** 4D Server führt Datenbankoperationen, wie Such- und Sortierläufe nach indizierten Datensätzen aus.

### Beispiel:

Die Arbeitsstation fragt beim Server ein Formular ab. Ein umfangreiches Formular braucht eine bestimmte Zeit im Netz. Der Server schickt das Formular an die Arbeitsstation. Dort wird es ohne weitere Arbeit für den Server selbständig bearbeitet. Dadurch reduziert sich die Serveraktivität erheblich. So ist die Auslastungsrate des Servers für 10 Stationen, die gleichzeitig im Benutzer- und Änderungsmodus in 10 verschiedenen Formularen mit jeder Menge Bilder arbeiten, relativ gering.

Die gegenseitige Verständigung zwischen Server und Arbeitsstation ist besonders vorteilhaft beim Anzeigen von Listen und Verwalten der aktuellen Auswahl. Der Server schickt mehrere Datensätze als Paket an die Arbeitsstation. Dieses enthält im Gegensatz zu einer herkömmlichen, heterogenen Client/Server-Architektur keinen Code. Die Arbeitsstation kann selbst die Daten zerlegen. Die

Anzeigeschnittstelle wird automatisch verwaltet. Da die Engine der Arbeitsstation mit dem Server kommunizieren kann, überträgt die Arbeitsstation nur die tatsächlich benutzten Datenfelder.

Die Arbeitsteilung zwischen Arbeitsstation und Server ist eindeutig festgelegt. Sie erfolgt automatisch durch 4D. Sie benötigen dafür keine besonderen Kenntnisse. Sie kann vielmehr zu neuen Strategien für Ihre Anwendungen beitragen.

# **4D Server in 10 Minuten**

-  Die Installation prüfen
-  Eine Server Datenbank erstellen
-  Ein remote 4D an Server Datenbank anmelden
-  Die Datenbankstruktur festlegen
-  Daten mit 4D Server bearbeiten
-  Eine eigene Menüleiste hinzufügen
-  Gleichzeitig mit 4D Server arbeiten

## Die Installation prüfen

---

Dieses Kapitel gibt eine kurze Einführung in folgende Bereiche:

- Eine Server Datenbank erstellen
- Clients an die erstellte Server Datenbank anmelden
- Eine Datenbankstruktur einrichten, inkl. Tabellen, Datenfeldern, Formularen und Methoden
- Einen zweiten Benutzer anmelden und gleichzeitig arbeiten

Zur Durchführung dieser Schritte benötigen Sie mindestens zwei Rechner:

- Einen Rechner, auf dem 4D Server, 4D und ein Web Browser installiert sind
- Einen anderen Rechner, auf dem 4D installiert ist

Bevor Sie zum ersten Mal mit 4D Server und 4D im lokalen Modus arbeiten, sollten Sie Ihre Installation prüfen. Lesen Sie dazu die nachfolgenden Abschnitte.

### **Installierte Elemente**

---

Dieser Abschnitt beschreibt den Speicherort der Elemente, die bei der Standardinstallation von 4D und 4D Server auf Ihrer Festplatte installiert werden:

### **Windows**

---

Die Elemente werden im Ordner **Programme/4D/4D vXX** installiert und erscheinen im Menü **Start**:

- **4D Server**: Dieser Ordner enthält die 4D Server Anwendung und die dazugehörigen Dateien und Ordner. Um 4D Server zu starten, doppelklicken Sie auf das Icon **4D Server.exe**.
- **4D**: Dieser Ordner enthält die 4D Anwendung und die dazugehörigen Dateien und Ordner. Um 4D zu starten, doppelklicken Sie auf das Icon **4D.exe**.

### **Mac OS**

---

Die Elemente werden im Ordner **Applications:4D:4D vXX** installiert und erscheinen in den Anwendungen:

- **4D Server**: 4D Server Software Paket. Um 4D Server zu starten, doppelklicken Sie auf dieses Paket.
- **4D**: 4D Software Paket. Um 4D zu starten, doppelklicken Sie auf dieses Paket.

Für diese spezifische Übung müssen Sie 4D noch auf einem anderen Rechner installieren.

## Wie geht es weiter?

---

Ihre Rechner können nur über das Netzwerk miteinander kommunizieren, wenn das Netzwerkprotokoll TCP/IP konfiguriert ist.

Sind 4D Server und 4D korrekt installiert, gehen Sie zum nächsten Abschnitt **Eine Server Datenbank erstellen**. Fehlen einige der oben aufgelisteten Ordner, schlagen Sie dazu im *4D Installationshandbuch* nach.

## Eine Server Datenbank erstellen

Dieser Abschnitt beschreibt, wie Sie eine Server Datenbank erstellen, auf die 4D im remote Modus über das Netzwerk zugreifen kann. Bevor Sie zum ersten Mal mit 4D Server und 4D arbeiten, sollten Sie Ihre Installation prüfen. Lesen Sie dazu im vorigen Abschnitt **Die Installation prüfen** nach.

**Hinweis:** Dieses Beispiel setzt voraus, dass Sie Ihre 4D Server Lizenz bereits aktiviert haben. Für die Verwendung von 4D im remote Modus ist keine Lizenz auf dem Client-Rechner erforderlich. Die Lizenzen werden auf dem 4D Server Rechner verwaltet. Weitere Informationen dazu finden Sie im *4D Installationshandbuch*.

Um eine 4D Server Datenbank zu öffnen, bzw. eine vorhandene zu erstellen, starten Sie 4D Server.

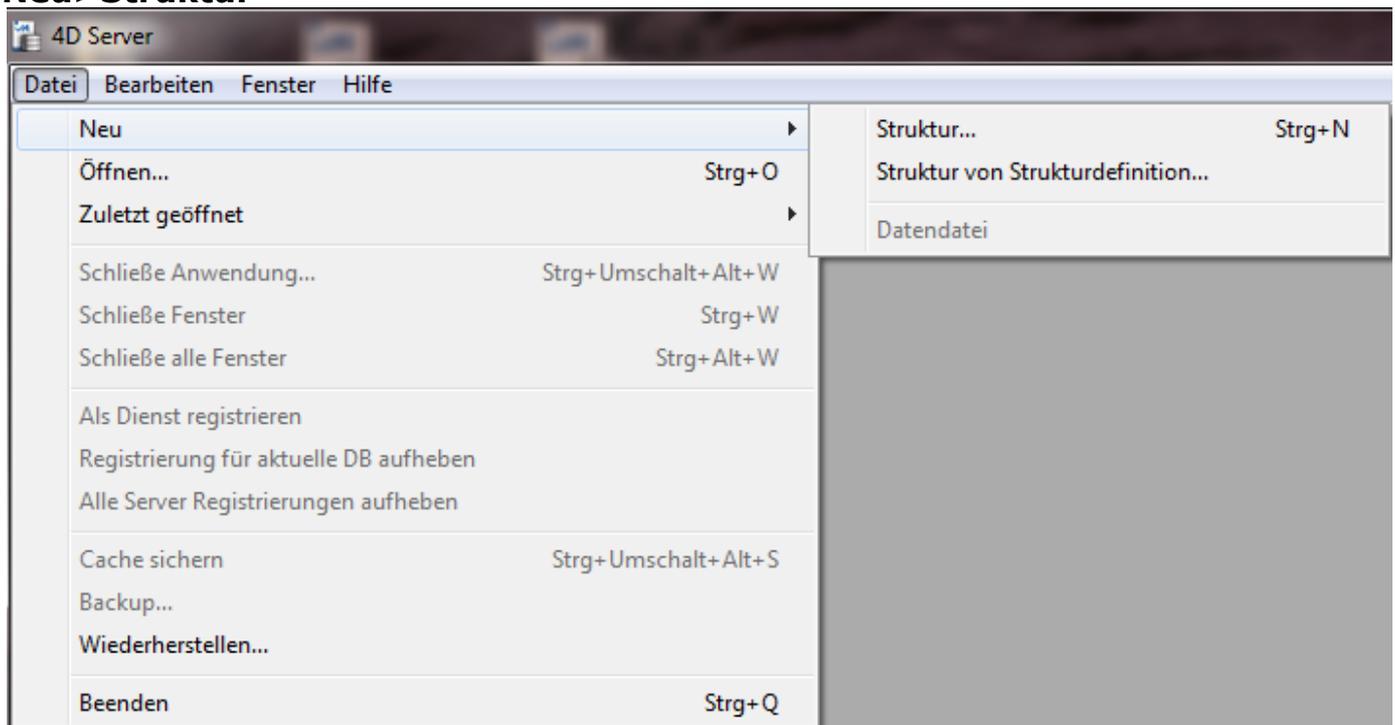
### 1. Doppelklicken Sie auf das Icon für 4D Server.



Starten Sie die Anwendung zum ersten Mal, erscheint auf dem Bildschirm das Dialogfenster zum Aktivieren der Anwendung. Danach erscheint bei jedem Start ein leeres Fenster. Diese Funktionsweise können Sie in den Einstellungen der Anwendung konfigurieren.

In diesem Beispiel erstellen wir eine leere Datenbank.

### 2. Wählen Sie im Menü Datei/Ablage von 4D Server den Befehl Neu>Struktur

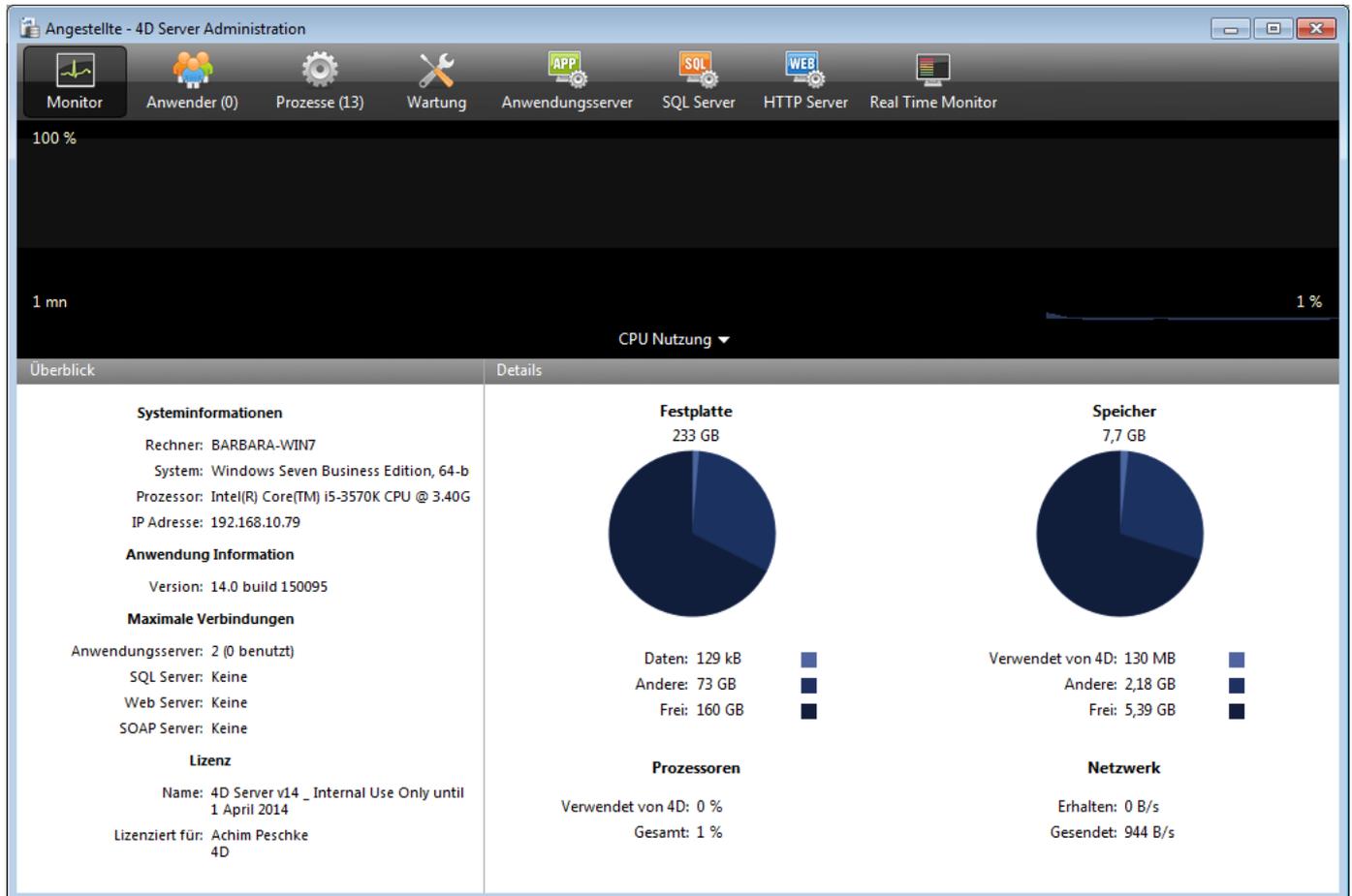


Auf dem Bildschirm erscheint ein Standard-Sichern Dialog, in dem Sie Name und Speicherort der neuen Datenbank angeben.

### 3. Legen Sie den Speicherort fest und geben dann den Namen der Datenbank ein. Geben Sie "Angestellte" ein und klicken auf die Schaltfläche

## Speichern.

4D Server erstellt automatisch die erforderlichen Dateien und Ordner für das Operieren der Datenbank. Dann erscheint das Server Verwaltungsfenster:



Das Verwaltungsfenster hat mehrere Seiten, die Sie über die Registerkarten am oberen Rand öffnen können. Die Seite Monitor zeigt dynamische Angaben zur verwendeten Datenbank, sowie Informationen zum System und der 4D Server Anwendung.

Die Seiten **Anwender** und **Prozesse** zeigen jeweils die Anzahl der an die Datenbank angemeldeten Benutzer und die aktuell laufenden Prozesse an. In diesem Stadium sind keine Clients an die Datenbank angemeldet, die Anzahl der Anwender zeigt Null an. Standardmäßig werden zwölf Prozesse angezeigt. Diese Prozesse werden von der Datenbank-Engine (Kernel) und den in 4D Server integrierten Servern angelegt (Anwendungsserver, HTTP Server, SQL Server).

## Wie geht es weiter?

Die Datenbank steht nun über das Netzwerk für die Anbindung von remote Anbindungen unter Windows und/oder auf Macintosh zur Verfügung. Sie ist jedoch noch nicht für HTTP Anbindungen verfügbar, da diese nicht standardmäßig zugelassen sind.

Weitere Informationen dazu finden Sie im nächsten Abschnitt **Ein remote 4D an Server Datenbank anmelden**. Sie lernen zuerst, wie Sie 4D im remote Modus anmelden, wie Sie die Struktur der Datenbank definieren und einige Datensätze in der Datenbank anlegen.

## Ein remote 4D an Server Datenbank anmelden

Dieser Abschnitt beschreibt folgendes:

- Ein remote 4D an die erstellte Server Datenbank anmelden.
- Eine Datenbankstruktur anlegen. Sie lernen auch, Tabellen und Datenfelder in der Datenbank zu erstellen, neue Datensätze einzugeben und bestehende Datensätze zu ändern.
- Einen zweiten Anwender anmelden.
- Gleichzeitig mit zwei remote Clients arbeiten.

### Anmelden an die Datenbank

Obwohl Sie die Datenbank mit 4D Server erstellt haben (siehe Abschnitt **Eine Server Datenbank erstellen**), werden alle Änderungen in der Datenbankstruktur sowie der aktuelle Datenbestand auf der Arbeitsstation angezeigt. Dieser Abschnitt beschreibt, wie Sie eine Arbeitsstation an den Server anmelden und von dort aus die Server Datenbank öffnen.

#### 1. Doppelklicken Sie auf das Icon remote 4D.



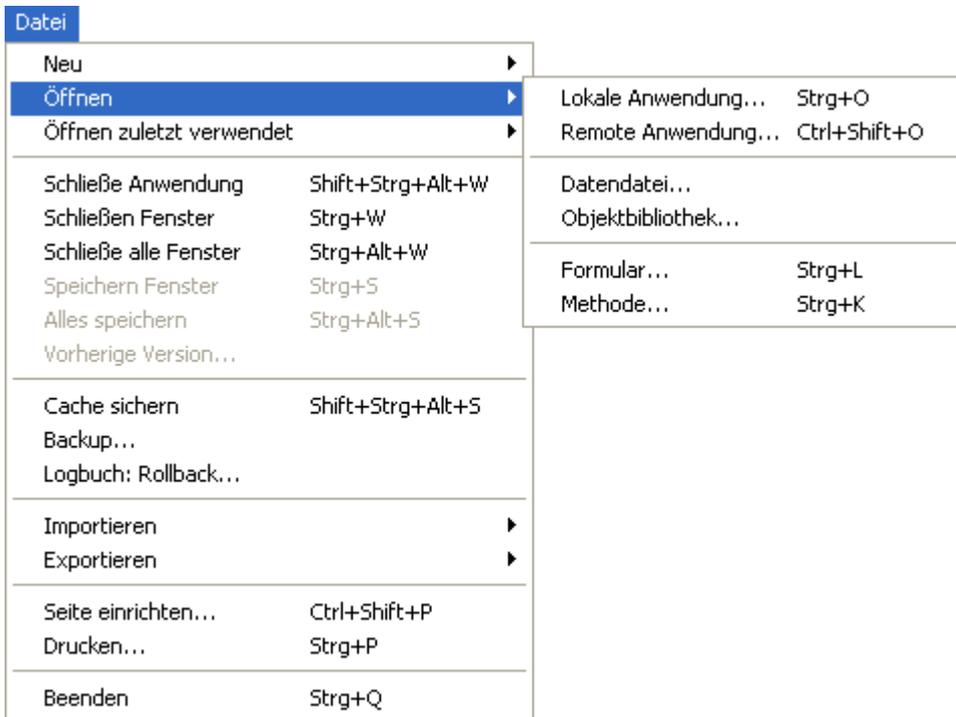
**Hinweis:** Für diesen Lernschritt können Sie 4D auf demselben Rechner wie 4D Server installieren.

Starten Sie das 4D Programm zum ersten Mal oder haben Sie die Startup-Parameter nicht geändert, erscheint auf dem Bildschirm der Willkommen-Assistent:

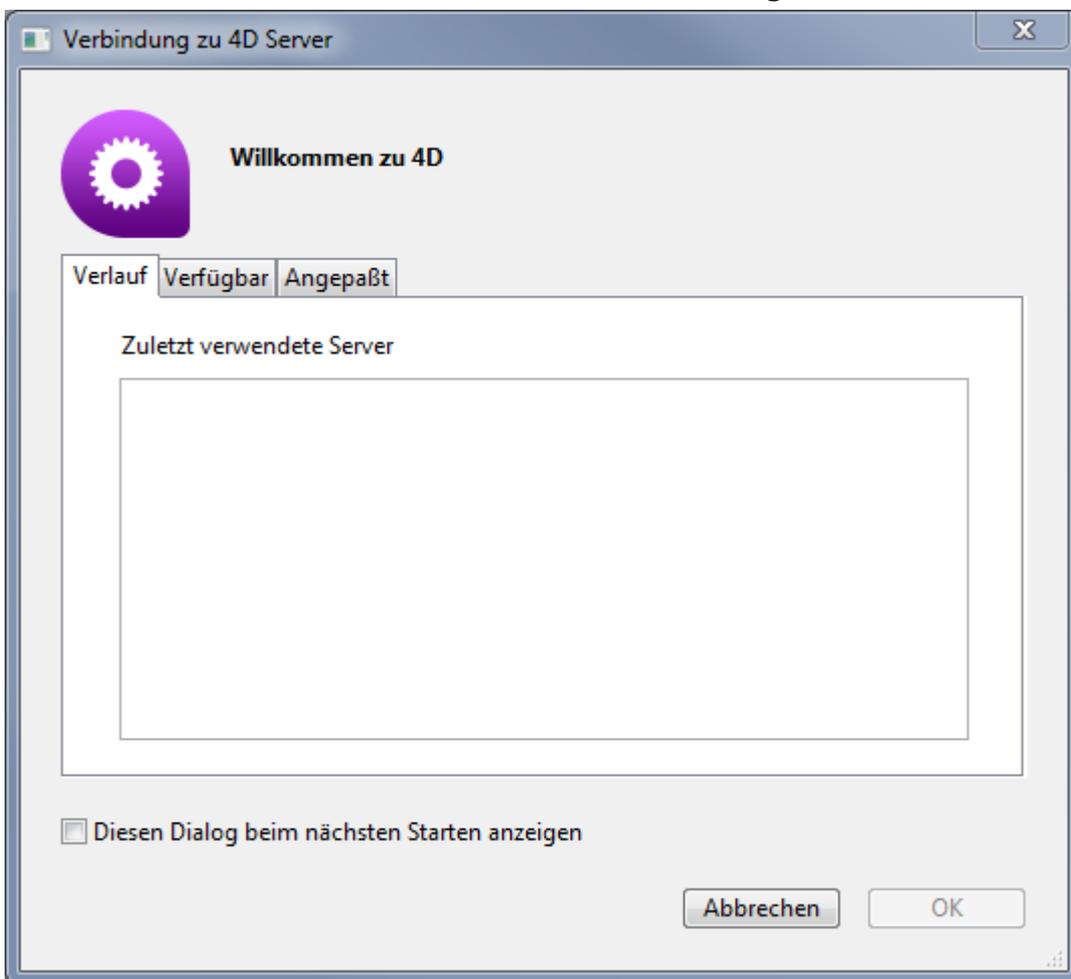


#### 2. Klicken Sie auf die Option "An 4D Server anmelden" ODER

wenn dieser Dialog nicht erscheint, wählen Sie im Menü Datei/Ablage von 4D den Befehl Öffnen>Remote Anwendung

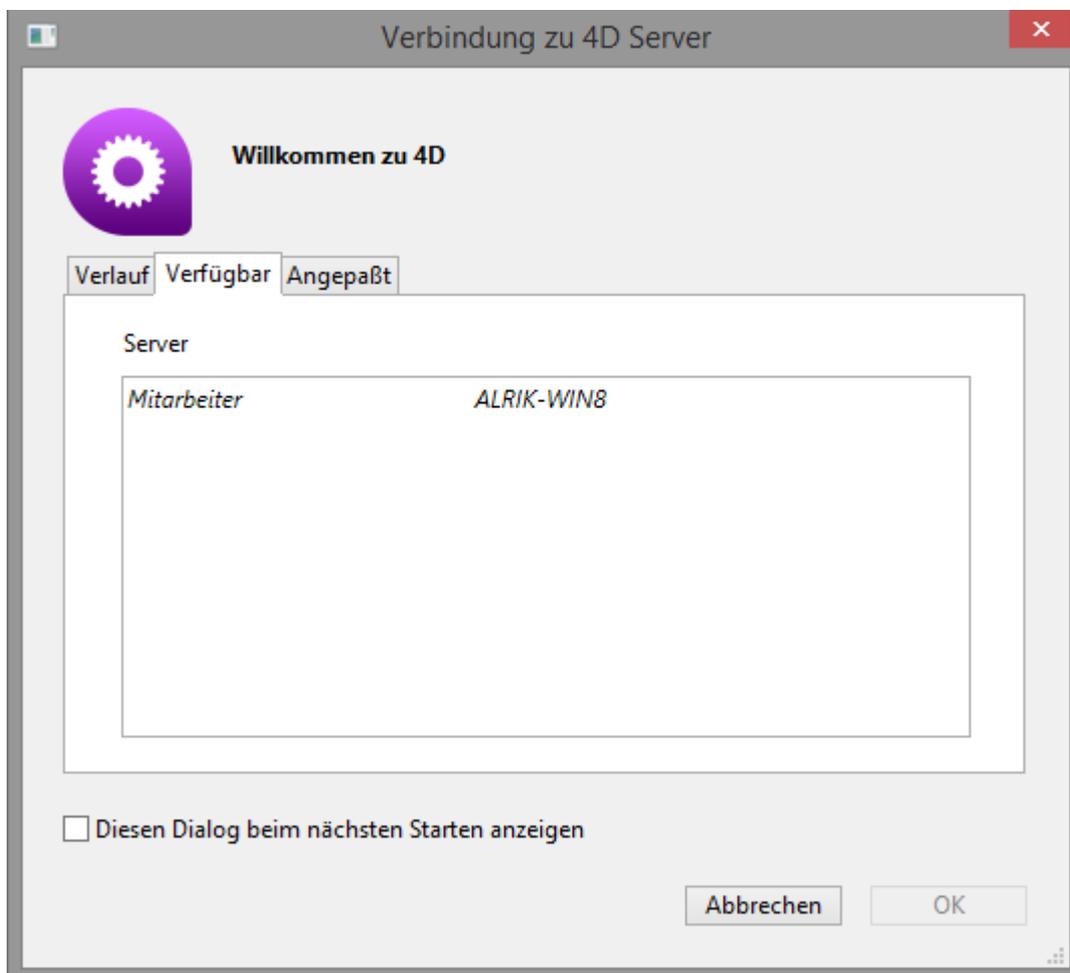


Auf dem Bildschirm erscheint der Anmeldedialog:



**3. Klicken Sie auf die Registerkarte Verfügbar, um die Liste der im Netzwerk veröffentlichten 4D Datenbanken anzuzeigen.**

In der Liste könnte z.B. die Datenbank Mitarbeiter erscheinen:



#### 4. Klicken Sie auf den Eintrag in der Liste.

Die Datenbank wird auf der Arbeitsstation in der Designumgebung geöffnet. Sie können sofort mit der Struktur Ihrer Datenbank beginnen.

### Tipps zur Fehlerbehebung

---

Erscheint nicht der Name der Datenbank, die Sie soeben mit 4D Server erstellt haben, überprüfen Sie folgende Punkte:

- Läuft 4D Server noch auf dem anderen Rechner?
- Wenn Sie einen zweiten Rechner einsetzen, sind beide Rechner an das Netzwerk angeschlossen?
- Sind die TCP/IP Einstellungen auf beiden Rechnern korrekt?
- Sind Sie noch nicht vertraut mit diesen Dialogfenstern, lesen Sie den Abschnitt **Eine Server Datenbank erstellen**.

### Server Aktivität

---

Im Verwaltungsfenster von 4D Server zeigt die Registerkarte in der Leiste am oberen Rand **Anwender (1)** an. Öffnen Sie die Seite, erscheinen der Anwendername und weitere Angaben.



Die ersten Prozesse betreffen den Server, sie wurden beim Starten von 4D Server eingerichtet. Die neuen Prozesse betreffen den ersten Benutzer, der sich am Server angemeldet hat:

- Der Anwendungsprozess verwaltet das Fenster zur Anzeige der Datensätze und die Anwendungsumgebung
- Der Design-Prozess verwaltet die Designumgebung

Jeder weitere Benutzer fügt in der Prozessliste mehrere weitere Prozesse hinzu.

Über die Schaltflächen **Anwender Prozesse**, **4D Prozesse** oder **Reserve Prozesse** und den Suchbereich in der rechten oberen Ecke der Seite können Sie die Liste der angezeigten Prozesse filtern.

## Wie geht es weiter?

---

Sie sind jetzt angemeldet und können mit der Datenbank arbeiten. Ihnen stehen alle Funktionalitäten von 4D im lokalen Modus zur Verfügung. Als nächstes definieren Sie die Struktur der Datenbank. Gehen Sie dazu zum Abschnitt **Die Datenbankstruktur festlegen**.

## Die Datenbankstruktur festlegen

---

Nachdem Sie an die Server Datenbank angemeldet sind (siehe Abschnitt **Ein remote 4D an Server Datenbank anmelden**), wählen Sie im Menü **Design** den Befehl **Datenbankstruktur**.

Auf dem Bildschirm erscheint das Strukturfenster, es ist standardmäßig leer. Wir erstellen eine einfache Tabelle.

### Die Tabelle [Angestellte] erstellen - ein Beispiel

---

1. Wählen Sie im Menü **Datei/Ablage** den Befehl **Neu>Tabelle** oder klicken Sie in der Werkzeugleiste auf das Icon für Tabellen

ODER

klicken Sie mit der rechten Maustaste in das Strukturfenster und wählen im Kontextmenü den Eintrag **Neue Tabelle**

ODER

Klicken Sie im Strukturfenster am oberen Rand auf die Schaltfläche **Hinzufügen** (mit Pluszeichen) und wählen den Eintrag **Tabelle**.

Die Tabelle wird angelegt.

2. Klicken Sie in den Titelbereich und geben den Namen "Angestellte" ein.



3. Doppelklicken Sie in den Bereich für Felder, um ein neues Feld anzulegen.



4. Nennen Sie das Feld *Vorname* und behalten den Typ *Alpha (255)* bei. Mit Doppelklick in das Feld können Sie das Inspektorfenster anzeigen.

5. Legen Sie in der Tabelle [Angestellte] auf dieselbe Weise folgende Datenfelder an:

#### **Feldname**   **Typ**

Vorname     Alphanumerisch (255 Zeichen)

Name         Alphanumerisch (255 Zeichen)

Gehalt        Zahl

Abteilung    Alphanumerisch (255 Zeichen)

Angestellte	
ID	2 <sup>32</sup>
Vorname	A
Name	A
Gehalt	0.5
Abteilung	A

**Hinweis:** Arbeiten andere remote 4D Anwendungen zur gleichen Zeit in der Server-Datenbank, erscheinen die Datenfelder, die Sie gerade angelegt haben, nach ein paar Sekunden auf diesen Client-Rechnern. Die Änderungen werden in Echtzeit auf den Server übertragen, erscheinen aber nicht sofort auf den anderen Bildschirmen, um zu häufiges Aktualisieren zu vermeiden.

## Fomulare für die Tabelle [Angestellte] erstellen

---

Nachdem Sie die Tabelle [Angestellte] erstellt haben, benötigen Sie Formulare, um mit den Datensätzen arbeiten zu können. Sie können dafür den Formularassistenten verwenden und Formulare für Ihren Bedarf erstellen. 4D bietet jedoch praktische Tastaturkürzel für die rasche Erstellung von Ein- und Ausgabeformularen.

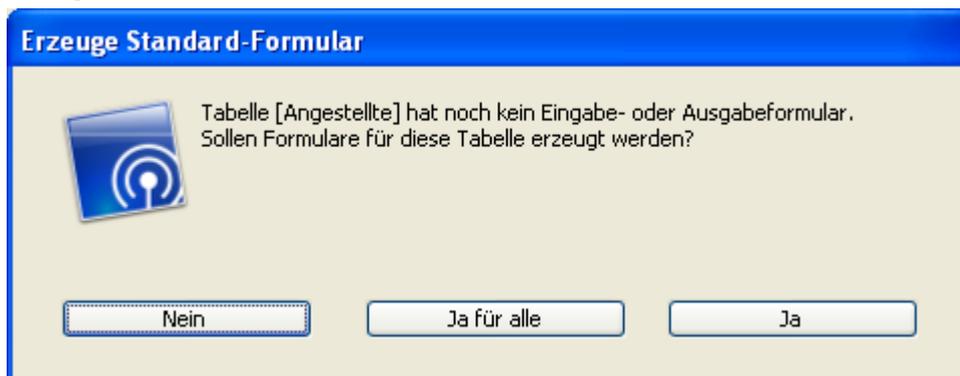
### 1. Klicken Sie in der Werkzeugleiste von 4D auf die Schaltfläche Tabelle:



### ODER

### wählen Sie im Menü Datensätze den Befehl Aktuelle Tabelle anzeigen.

Auf dem Bildschirm erscheint das Fenster für Datensätze. 4D stellt fest, dass der Tabelle noch keine Formulare zugeordnet sind und fragt Sie, ob das Programm diese anlegen soll.



### 2. Klicken Sie auf die Schaltfläche Ja.

4D erstellt automatisch ein Eingabeformular und ein Ausgabeformular. Im Eingabeformular können Sie Datensätze einzeln hinzufügen bzw. anzeigen. Im Ausgabeformular können Sie mehrere Datensätze in einer Liste hinzufügen bzw. anzeigen.

## Wie geht es weiter?

---

Sie können nun Daten in der Server-Datenbank eingeben und bearbeiten. Gehen Sie zum nächsten Abschnitt **Daten mit 4D Server bearbeiten.**

## Daten mit 4D Server bearbeiten

Im Abschnitt **Die Datenbankstruktur festlegen** haben Sie die Tabelle *[Angestellte]* und Standardformulare dafür erstellt. Sie können nun Datensätze eingeben.

### Datensätze eingeben

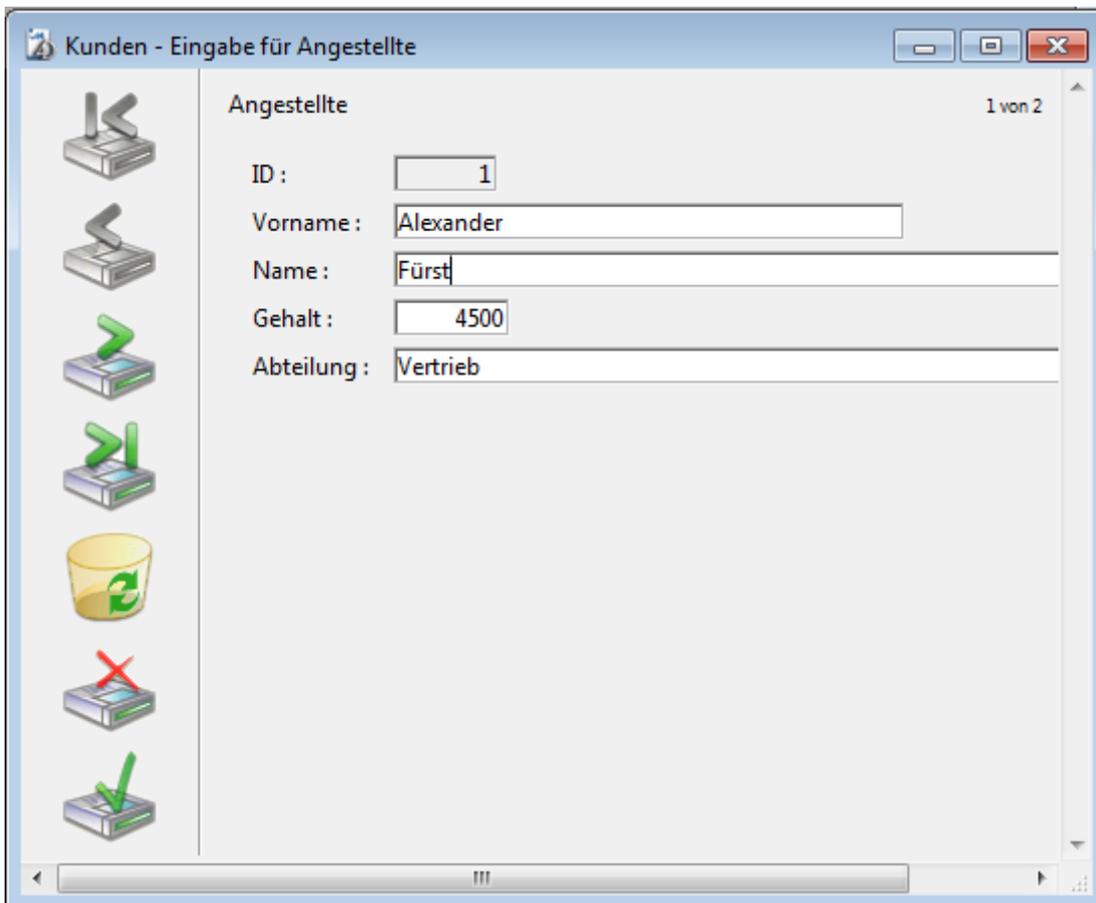
4D bietet im Designmodus eine Reihe von Werkzeugen und Editoren, um Datensätze einzugeben, zu suchen, zu ändern und zu drucken. Auf diese Weise können Sie nach und nach Ihre eigenen Werkzeuge für die Anwendungsumgebung einrichten.

#### 1. Wählen Sie im Menü **Datensätze** den Befehl **Neuer Datensatz**.

Es erscheint ein leeres Eingabeformular.

#### 2. Geben Sie Ihren ersten Datensatz ein.

Wechseln Sie per Mausklick oder mit der Tabulatortaste von einem Feld in das nächste.



Angestellte		1 von 2
ID :	<input type="text" value="1"/>	
Vorname :	<input type="text" value="Alexander"/>	
Name :	<input type="text" value="Fürst"/>	
Gehalt :	<input type="text" value="4500"/>	
Abteilung :	<input type="text" value="Vertrieb"/>	

#### 3. Klicken Sie am linken Rand auf die Schaltfläche **Bestätigen**. Der Datensatz wird gesichert.

Auf dem Bildschirm erscheint ein neues Leerformular zur weiteren Eingabe von Datensätzen.

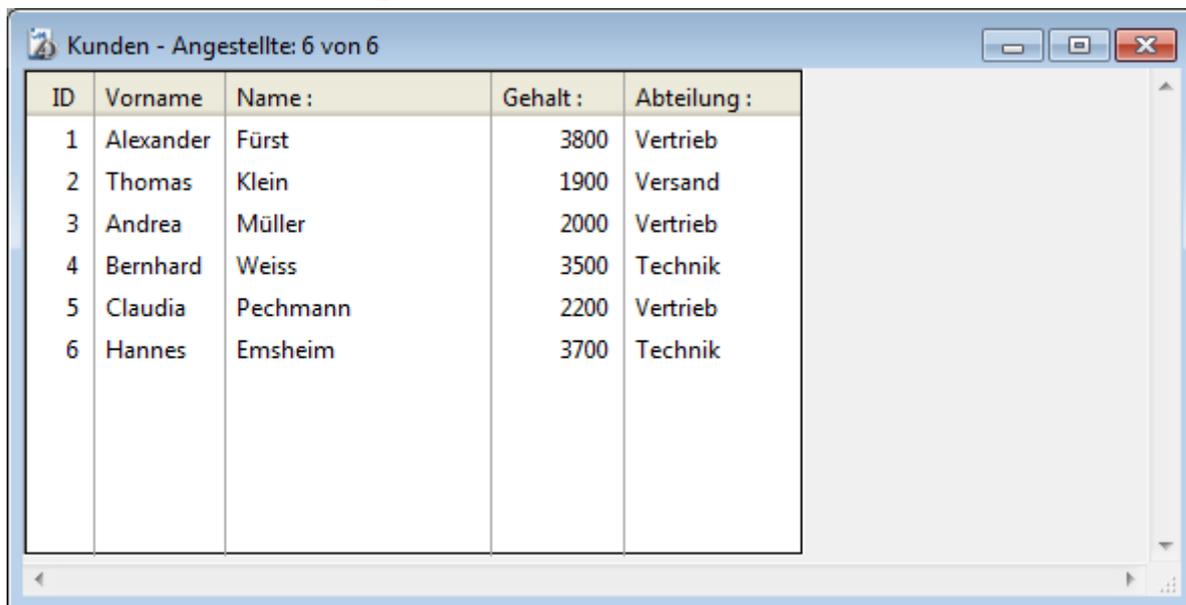
#### 4. Geben Sie fünf weitere Datensätze mit folgenden Werten ein:

Vorname	Name	Gehalt	Abteilung
Alexander	Fürst	3800	Vertrieb
Thomas	Klein	1900	Versand
Andrea	Müller	2000	Vertrieb
Bernhard	Weiß	3500	Technik
Claudia	Pechmann	2200	Versand

Haben Sie den letzten Datensatz eingegeben, klicken Sie auf Abbrechen (Icon mit Kreuz am linken Rand), um das neue Leerformular zu annullieren. 4D Server kehrt zum Ausgabeformular zurück.

**5. Werden nicht alle sechs Datensätze angezeigt, wählen Sie im Menü Datensätze den Befehl Alle anzeigen und passen das Fenster bzw. die Spalten bei Bedarf an.**

Ihr Bildschirm sollte folgendermaßen aussehen:



Die Datensätze sind nun in der Datenbank auf dem Server-Rechner gespeichert. Meldet sich ein zweites remote 4D auf dem Server-Rechner an, kann es die soeben eingegebenen Datensätze anzeigen. Ebenso können Sie von Ihrer Arbeitsstation aus alle Datensätze anzeigen, die in der Datenbank gespeichert sind, inkl. der Datensätze, die von anderen Arbeitsstationen eingegeben wurden. Wählen Sie dazu im Menü **Datensätze** den Befehl **Alle anzeigen**. Jeder Benutzer hat Zugang zu den Datensätzen, die auf dem Server gespeichert sind.

## Datensätze suchen

Sobald Sie Datensätze in der Tabelle *[Angestellte]* eingegeben haben, können Sie mit den Datensätzen arbeiten, z.B. Such- oder Sortierläufe durchführen oder drucken.

Sie können beispielsweise nach den Angestellten einer Abteilung suchen.

**1. Klicken Sie in der Menüleiste auf das Icon für Suchen.**



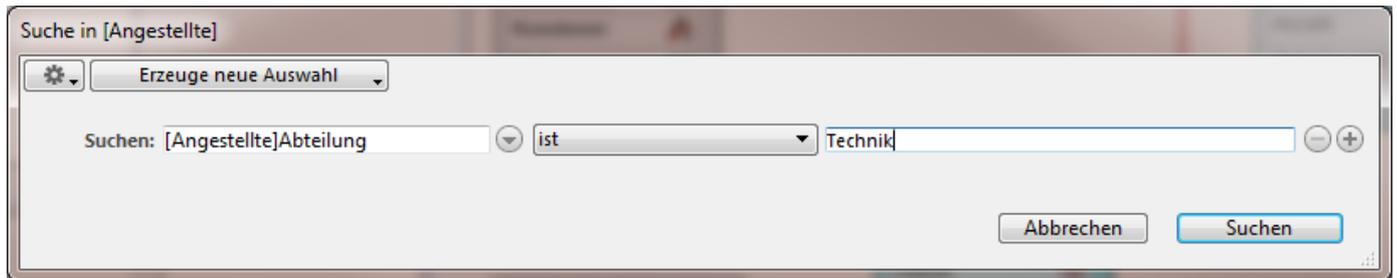
Auf dem Bildschirm erscheint der Sucheditor.

**2. Lassen Sie [Angestellte]Abteilung" als Suchkriterium, "ist" in der Liste für Operatoren und geben im 3. Bereich "Technik" ein.**

Der Sucheditor sieht jetzt folgendermaßen aus:

### 3. Klicken Sie am unteren Rand auf die Schaltfläche Suchen.

Die Anfrage wird an den 4D Server gesendet, 4D Server sendet das Ergebnis an den Client zurück. Im Ausgabeformular erscheinen nur die Angestellten der Abteilung Technik.



### 4. Wollen Sie wieder alle Datensätze anzeigen, wählen Sie im Menü Datensätze den Befehl Alle anzeigen.

## Wie geht es weiter?

---

Sie haben in wenigen Minuten eine Server Datenbank erstellt, eine Tabelle angelegt, Datensätze eingegeben und dann mit diesen Daten eine Suche ausgeführt.

Sie lernen nun, eine eigene Menüleiste hinzuzufügen. Gehen Sie zum nächsten Abschnitt **Eine eigene Menüleiste hinzufügen.**

## Eine eigene Menüleiste hinzufügen

---

In diesem Abschnitt erstellen Sie zwei Methoden und eine eigene Menüleiste, d.h. Sie erstellen Ihre eigene 4D Anwendung.

### Zwei Methoden hinzufügen

---

#### 1. Wählen Sie im Menü Datei/Ablage den Befehl Neu>Methode.

Es erscheint das Dialogfenster Neue Methode.

#### 2. Geben Sie im Bereich Methodename "M\_Hinzufügen" ein und klicken auf die Schaltfläche OK.

Auf dem Bildschirm erscheint das Methodenfenster mit dem Titel "M\_Hinzufügen".

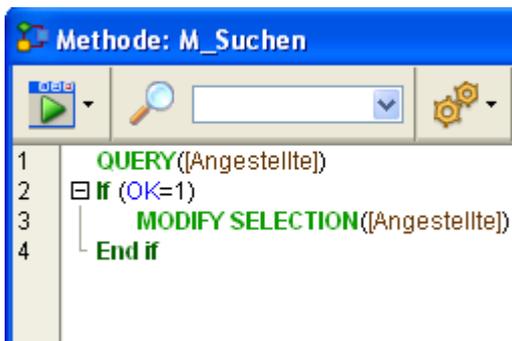
#### 3. Geben Sie folgenden Code ein:



The screenshot shows the 4D Method Editor window titled 'Methode: M\_Hinzufügen'. The code editor contains the following code:

```
1 Repeat
2   Hinzufügen([Angestellte])
3 Until (OK=0)
```

#### 4. Erstellen Sie eine 2. Methode mit Namen "M\_Suchen" und geben folgenden Code ein:



The screenshot shows the 4D Method Editor window titled 'Methode: M\_Suchen'. The code editor contains the following code:

```
1 QUERY([Angestellte])
2 If (OK=1)
3   MODIFY SELECTION([Angestellte])
4 End if
```

Haben Sie beide Methoden erstellt, können Sie eine eigene Menüleiste einrichten und diese Methoden Ihren eigenen Menübefehlen hinzufügen.

### Eine eigene Menüleiste einrichten

---

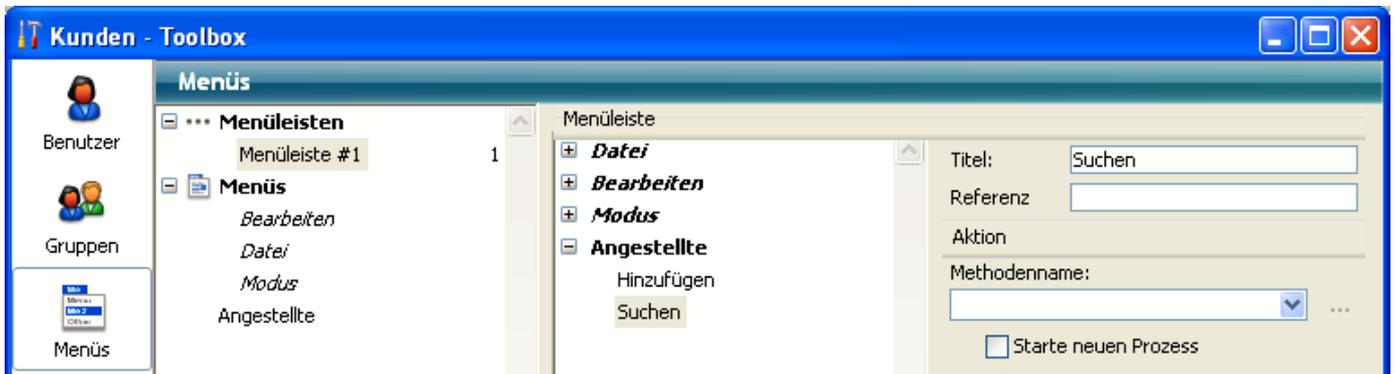
#### 1. Wählen Sie im Menü Design den Befehl Toolbox>Menüs.

Auf dem Bildschirm erscheint der Menüeditor mit der Standard Menüleiste.

#### 2. Wählen Sie im mittleren Fenster den Eintrag Menüleiste # 1 und klicken auf das Pluszeichen am unteren Rand.

#### 3. Geben Sie als Menütitel "Angestellte" ein und drücken die Eingabetaste.

**4. Klicken Sie mit der rechten Maustaste auf diesen Menütitel, um eine neue Zeile hinzuzufügen.**



**5. Geben Sie im Textbereich "Hinzufügen" ein und klicken auf die Eingabetaste.**

**6. Klicken Sie erneut mit der rechten Maustaste auf den Menütitel, um einen zweiten Befehl hinzuzufügen.**

**7. Geben Sie "Suchen" ein und klicken auf die Eingabetaste.**

Ihre Menüleiste sieht folgendermaßen aus:



**8. Klicken Sie auf den Menübefehl "Hinzufügen" und wählen im Bereich Methodenname "M\_Hinzufügen".**

**9. Klicken Sie auf den Menübefehl "Suchen" und wählen im Bereich Methodenname "M\_Suchen".**

Ihre Menüleiste sieht nun folgendermaßen aus:

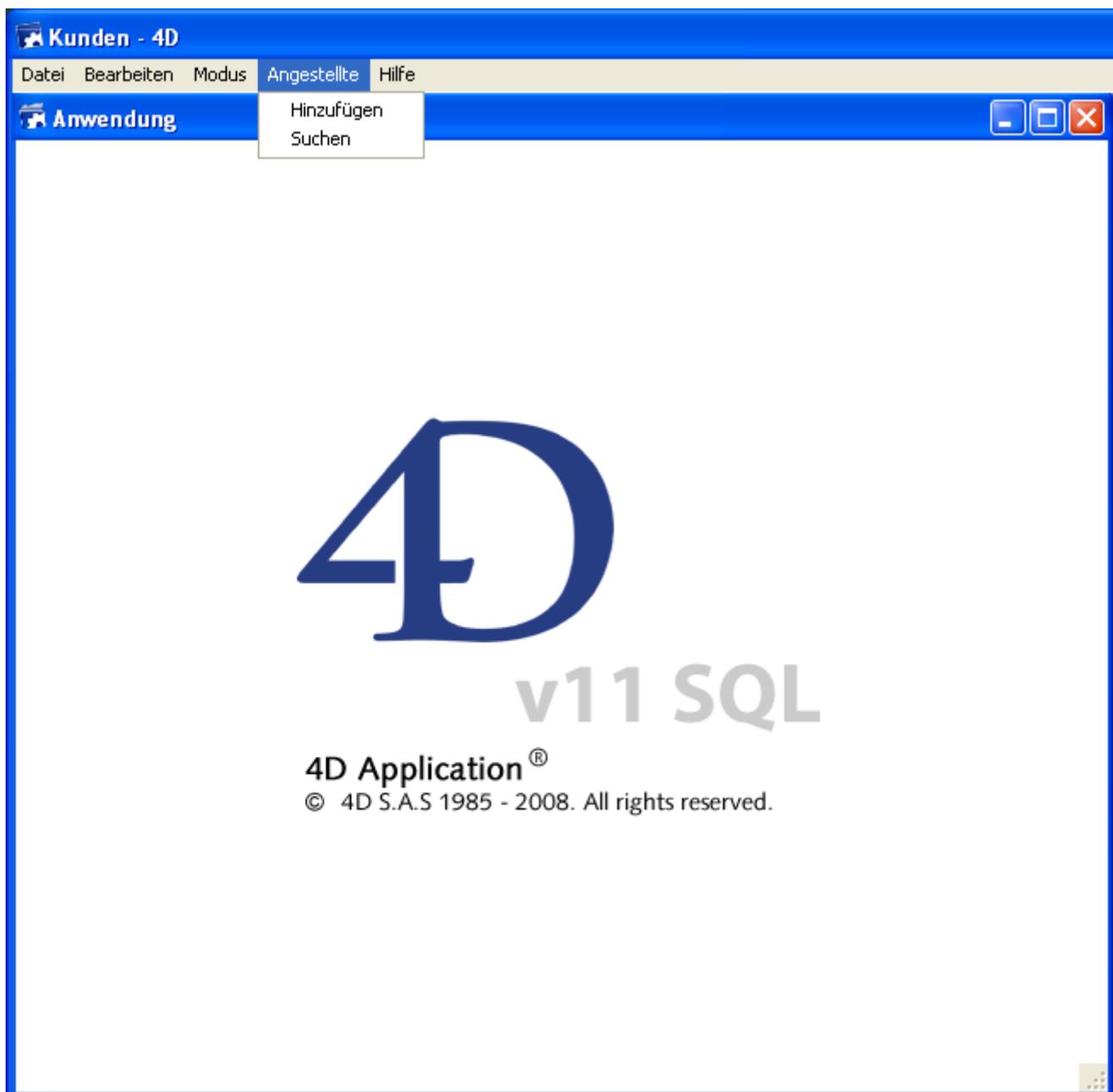


**10. Schließen Sie das Fenster Toolbox.**

Ihr eigenes Menü ist fertig!

**11. Wählen Sie im Menü Start den Befehl Anwendung testen.**

Sie verwenden jetzt Ihre eigene Anwendung mit den selbsterstellten Menüs.

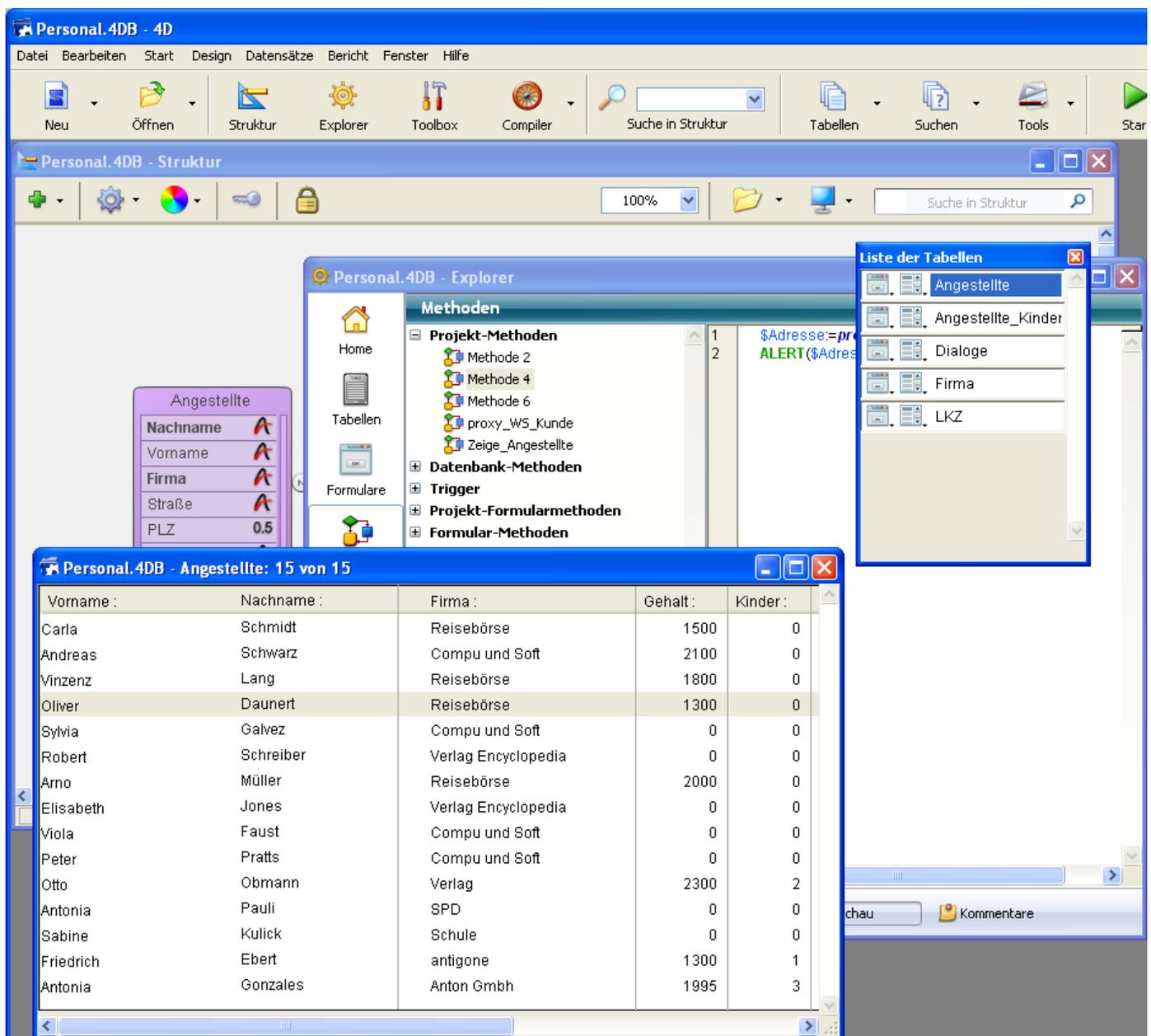


Wählen Sie jetzt im Menü **Angestellte** den Befehl **Suchen**, erscheint der standardmäßige 4D Sucheditor. Sie können Ihre Suche definieren, die gefundenen Datensätze anzeigen und ändern.

Sie haben, ohne zu wissen, soeben eine Mehrplatz-Anwendung entwickelt!

Die Erklärung dafür finden Sie im nächsten Abschnitt **Gleichzeitig mit 4D Server arbeiten**.



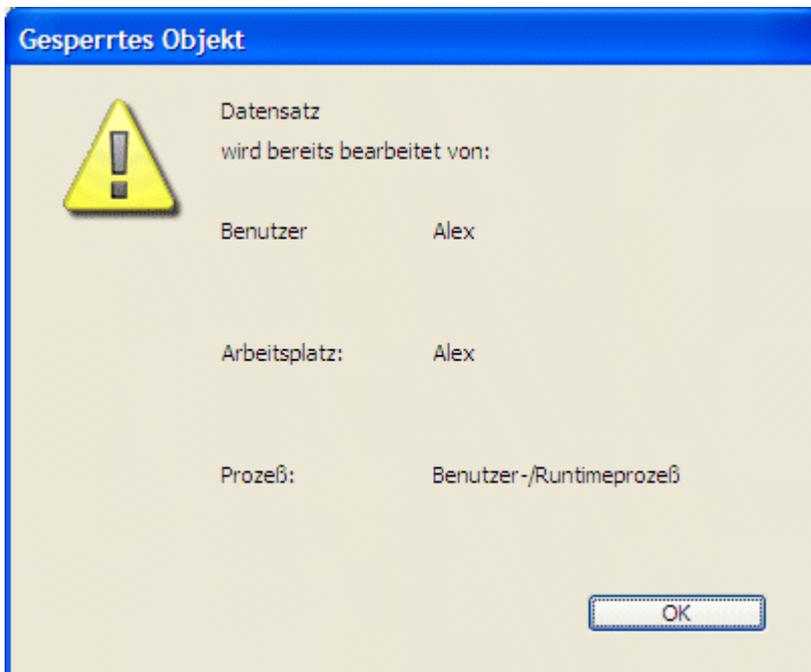


Alle Datensätze und Methoden stehen zur Verfügung.

## Gleichzeitig mit Datensätzen arbeiten

- 1. Wechseln Sie auf dem ersten remote Rechner in den Modus Anwendung testen und wählen im Menü Angestellte den Befehl Suchen. Suchen Sie nach den Datensätzen mit dem Kriterium "Firma ist gleich Compu und Soft".**
- 2. Wiederholen Sie denselben Vorgang für den zweiten remote Rechner.** Sie erhalten auf beiden Rechnern dieselbe Liste.
- 3. Doppelklicken Sie auf dem ersten Rechner auf den Datensatz "Andreas Schwarz".**
- 4. Wiederholen Sie denselben Vorgang für den zweiten Rechner.**

4D Server aktiviert den eingebauten Mechanismus zum Sperren von Datensätzen. Sie erhalten die Meldung, dass der Datensatz bereits benutzt wird:



Sie können diesen Datensatz aber im Lesemodus öffnen, d.h. Sie können ihn anzeigen, jedoch nicht ändern.

**5. Ändern Sie auf dem ersten Rechner den Vornamen um in "Michael" und bestätigen Sie Ihre Änderungen.**

Auf dem Bildschirm erscheint die aktualisierte Liste:



**6. Annullieren Sie auf dem zweiten Rechner die Anzeige des Datensatzes im Eingabeformular. Auf dem Bildschirm erscheint ebenfalls die aktualisierte Liste:**

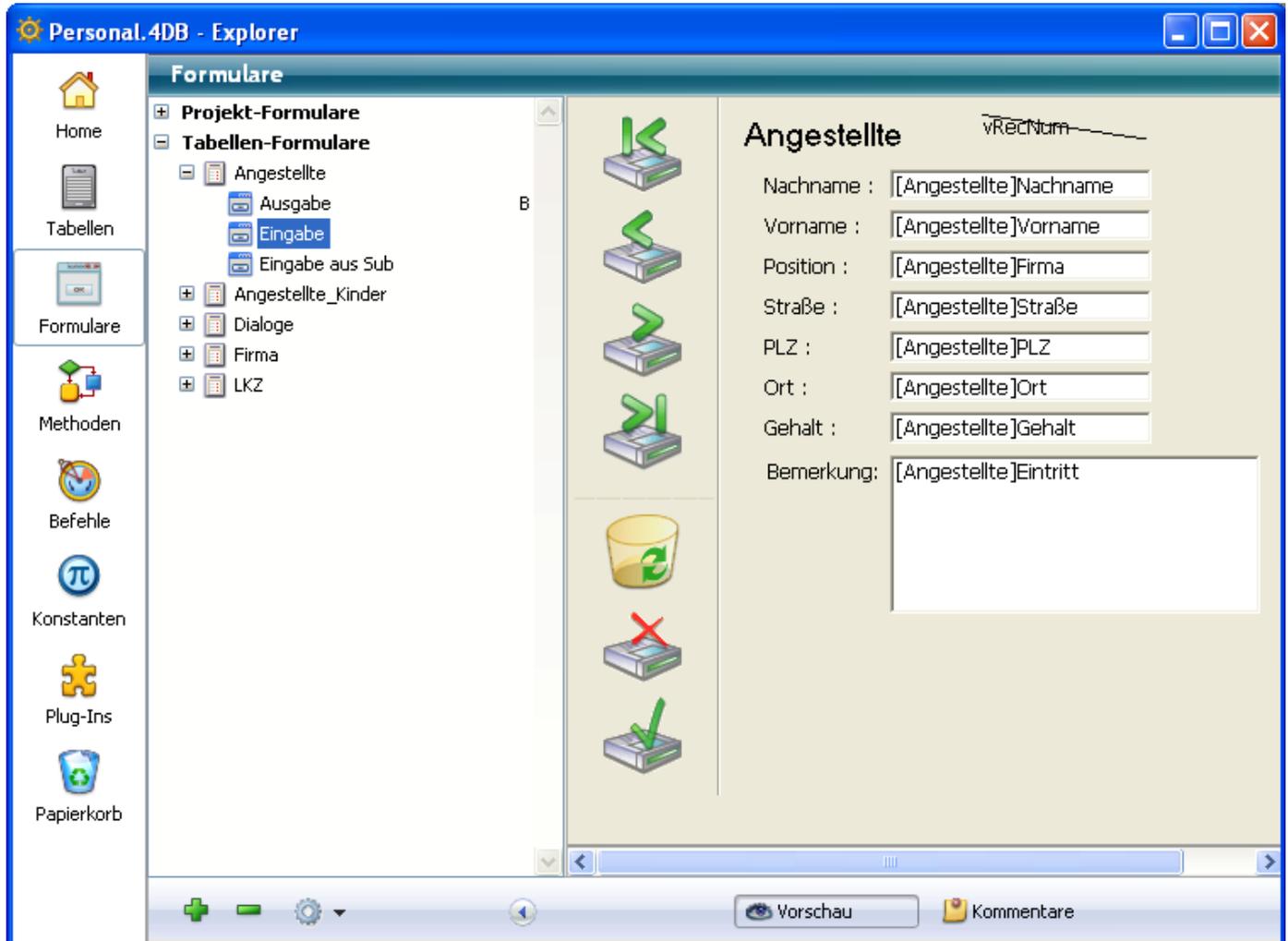


## Gleichzeitig mit Design Objekten arbeiten

4D Server ist Daten- und Anwendungsserver zugleich. Das bedeutet folgendes:

- 1. Drücken Sie auf der zweiten Arbeitsstation die Escape-Taste und wählen Sie im Menü Modus den Befehl Zurück zur Designumgebung.**
- 2. Wiederholen Sie diesen Vorgang auf dem ersten Rechner.**
- 3. Wählen Sie auf der ersten Arbeitsstation im Menü Design den Befehl Explorer>Formular.**

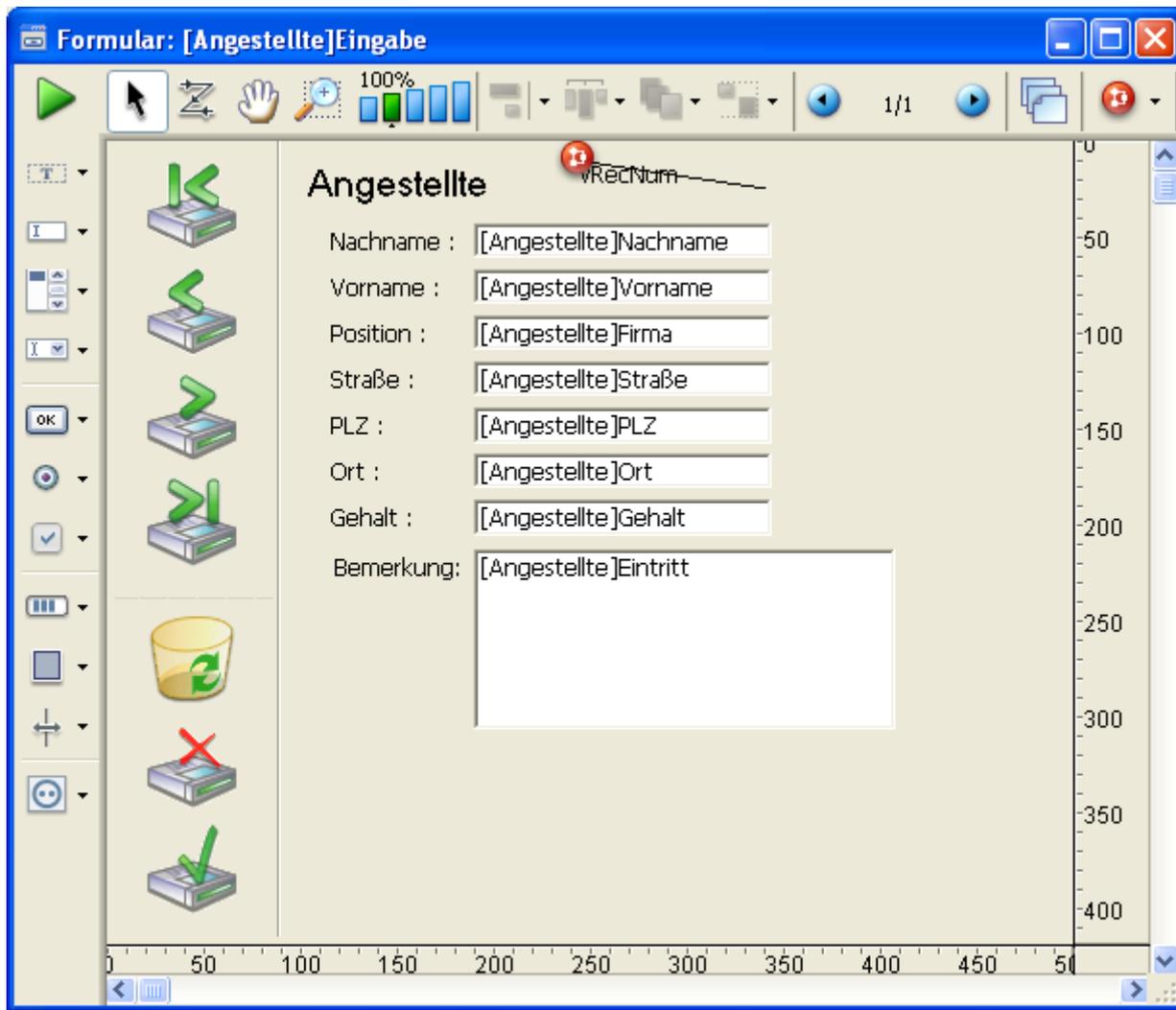
Auf dem Bildschirm erscheint das Explorer-Fenster.



**4. Wählen Sie das Tabellenformular und erweitern Sie die Tabelle [Angestellte].**

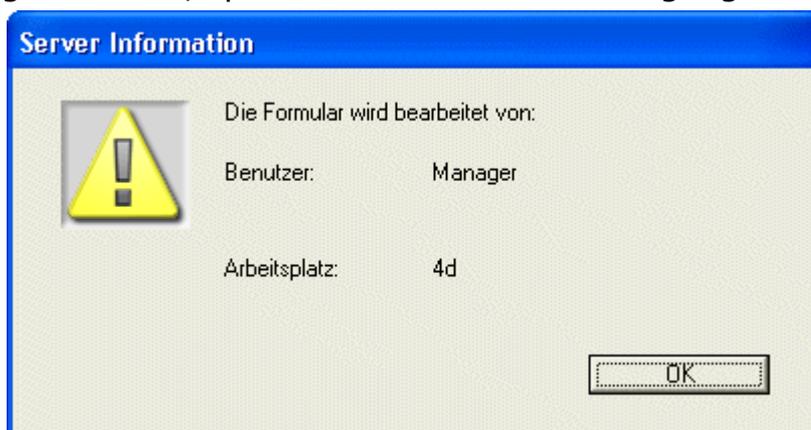
**5. Doppelklicken Sie auf das Eingabeformular und klicken auf die Schaltfläche Bearbeiten.**

Auf dem Bildschirm erscheint folgendes Fenster:



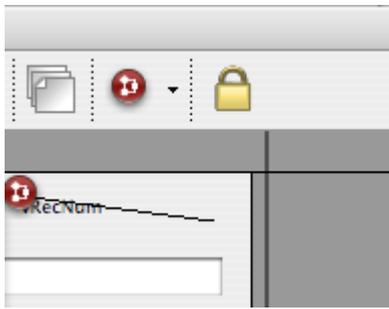
## 6. Wiederholen Sie diesen Vorgang auf dem zweiten Rechner.

Da dieses Formular bereits auf der ersten Arbeitsstation im Bearbeitungsmodus geöffnet ist, sperrt 4D Server diesen Vorgang. Sie erhalten folgende Meldung:



Sie können dieses Formular auf der zweiten Arbeitsstation zwar nicht bearbeiten, jedoch Objekte auswählen und in andere Formulare einsetzen.

Das Schloss in der Ecke oben rechts weist darauf hin, dass Sie keine Änderungen vornehmen können.



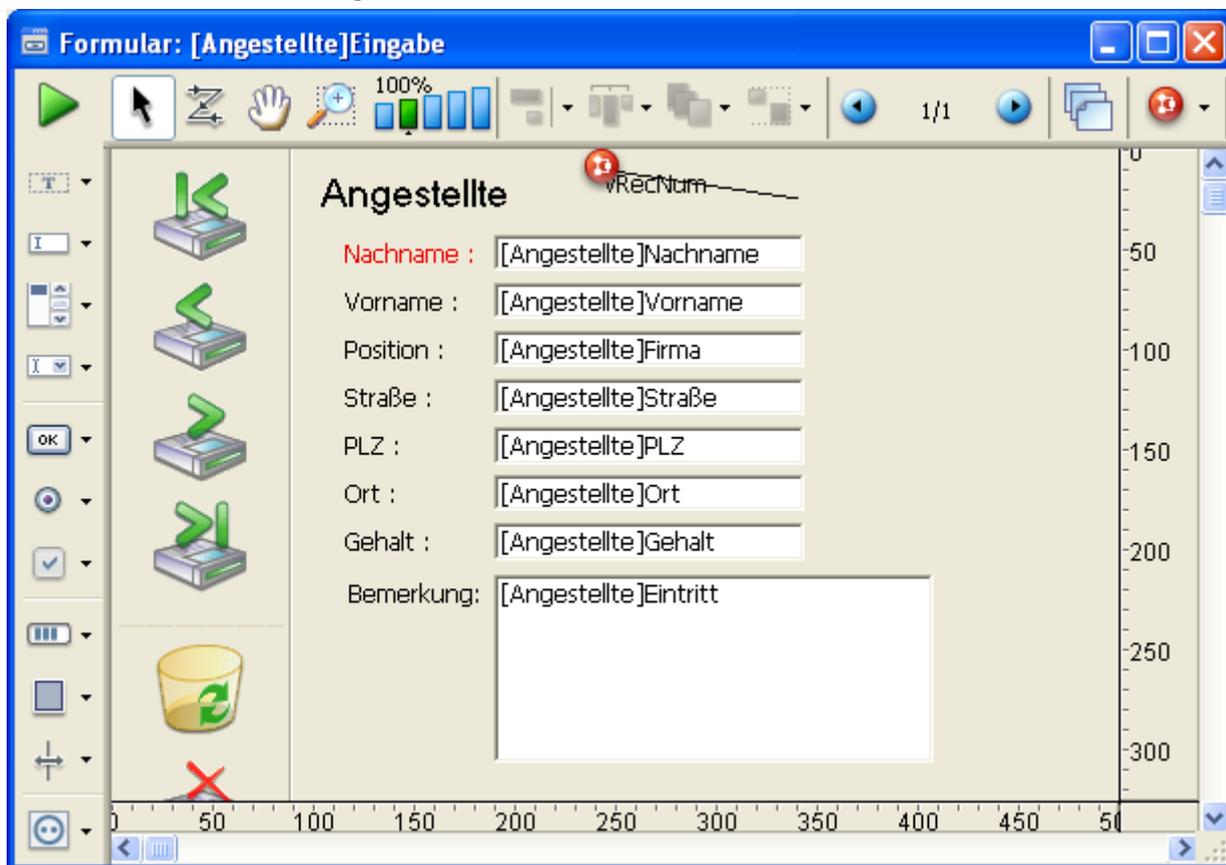
**7. Wählen Sie auf der ersten Arbeitsstation die Bezeichnung "Name" links neben der Tabelle [Angestellte].**

**8. Wählen Sie im Menü Objekt>Farbe in der Farbpalette einen roten Farbton, um dieses Objekt rot einzufärben.**

**9. Wählen Sie im Menü Datei/Ablage den Befehl Speichern Formular [Angestellte]Eingabe.**

**10. Schließen Sie das Formular auf der zweiten Arbeitsstation und öffnen es wieder, um es neu zu laden.**

Auf dem Bildschirm erscheint nun das Formular mit den Änderungen, die Sie auf der ersten Arbeitsstation gemacht haben.



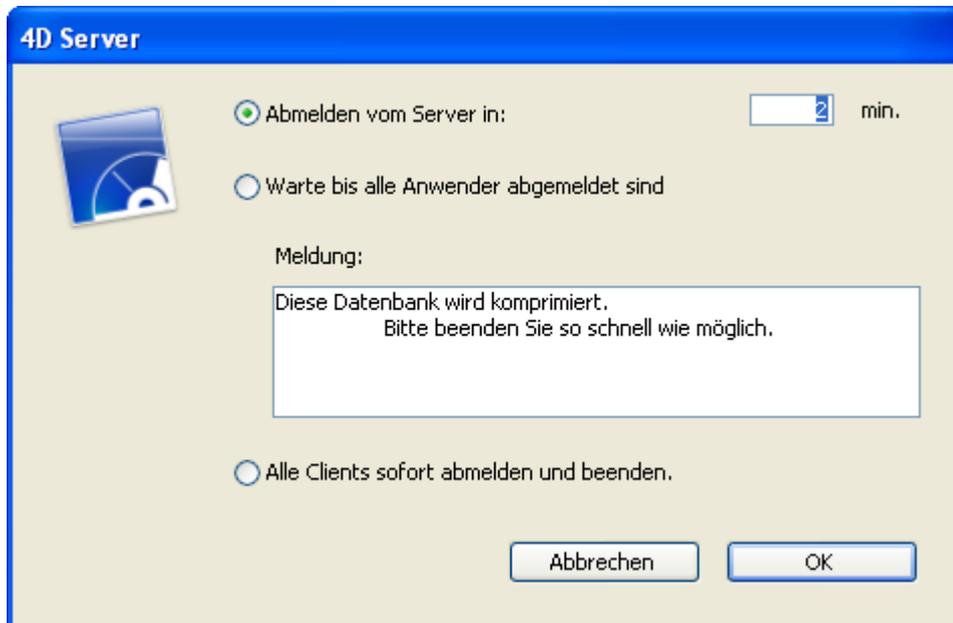
Wie Sie sehen, können mit 4D Server mehrere Benutzer gleichzeitig an derselben Datenbank arbeiten!

## Den Server abschalten

4D Server informiert nicht nur die Arbeitsstationen, wenn mehrere Benutzer gleichzeitig auf denselben Datensatz oder dasselbe Objekt zugreifen wollen, sondern auch wenn der Server abgeschaltet wird.

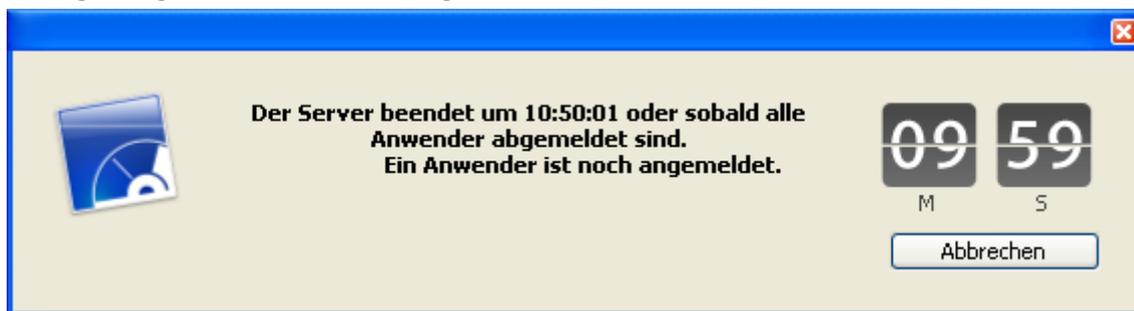
**1. Beide Arbeitsstationen bleiben an die Server-Datenbank angeschlossen. Wählen Sie auf dem Server-Rechner unter Windows im Menü Datei, auf Mac OS im Menü 4D Server den Befehl Beenden.**

Auf dem Bildschirm erscheint folgender Dialog:



## 2. Klicken Sie auf die Schaltfläche OK.

Die beiden Arbeitsstationen erhalten unmittelbar darauf die Meldung, dass der Server abgeschaltet wird. Legt ein Benutzer gerade einen neuen Datensatz an, bleibt ihm genügend Zeit, die Eingabe zu beenden und zu sichern.



Sie können beim Abschalten des Servers auch die Option "Warte bis alle Anwender abgemeldet sind" verwenden.

## 3. Während der Server abschaltet, beenden Sie 4D auf den beiden remote Rechnern.

## Zusammenfassung

---

In nur 10 Minuten – außer Sie hatten eine Kaffeepause – haben Sie gelernt, mit 4D Server zu arbeiten:

- Sie haben von Grund auf eine Datenbank erstellt.
- Sie haben eine Tabelle eingerichtet und 4D Server angewiesen, dafür Formulare zu erstellen.
- Sie haben Datensätze angelegt und damit gearbeitet.
- Sie haben Ihre Anwendung mit einer eigenen Menüleiste ausgestattet.
- Sie haben die Server Datenbank unter Windows und auf Macintosh eingesetzt.
- Sie haben den Server abgeschaltet und erneut gestartet.

Schließlich haben Sie zwei eigene Anwendungen erstellt (Windows und Macintosh), obwohl Sie eigentlich nur eine Anwendung entwickelt haben. Außerdem können Sie die Datenbank bei Bedarf auch im lokalen Modus einsetzen. Sie müssen die Datenbank dazu lediglich mit 4D öffnen.

Weitere Informationen zu 4D Server finden Sie in der Einleitung und den nachfolgenden Kapiteln.

Eine vollständige Beschreibung der 4D Umgebung finden Sie in folgenden Handbüchern:

- *4D Designmodus* erläutert die Strukturierung und Verwendung von 4D Anwendungen und Datenbanken.
- *4D Programmiersprache* beschreibt die Befehle der 4D Programmiersprache. Wollen Sie beispielsweise mehr über 4D Server als Web Server erfahren, lesen Sie den Abschnitt **Web Server, Überblick**.

# **4D Server verwenden**

-  Neue 4D Server Datenbank erstellen
-  4D Server beenden
-  4D Server Menüs
-  Netzwerk und Client-Server Optionen
-  IP Einstellungen
-  Client/Server Verbindungen verschlüsseln
-  Single Sign On (SSO) unter Windows
-  Ordner Resources verwalten
-  Datenbank als Dienst registrieren
-  Logischen Spiegel einrichten
-  Volume Shadow Copy Dienst unter Windows verwenden

## Neue 4D Server Datenbank erstellen

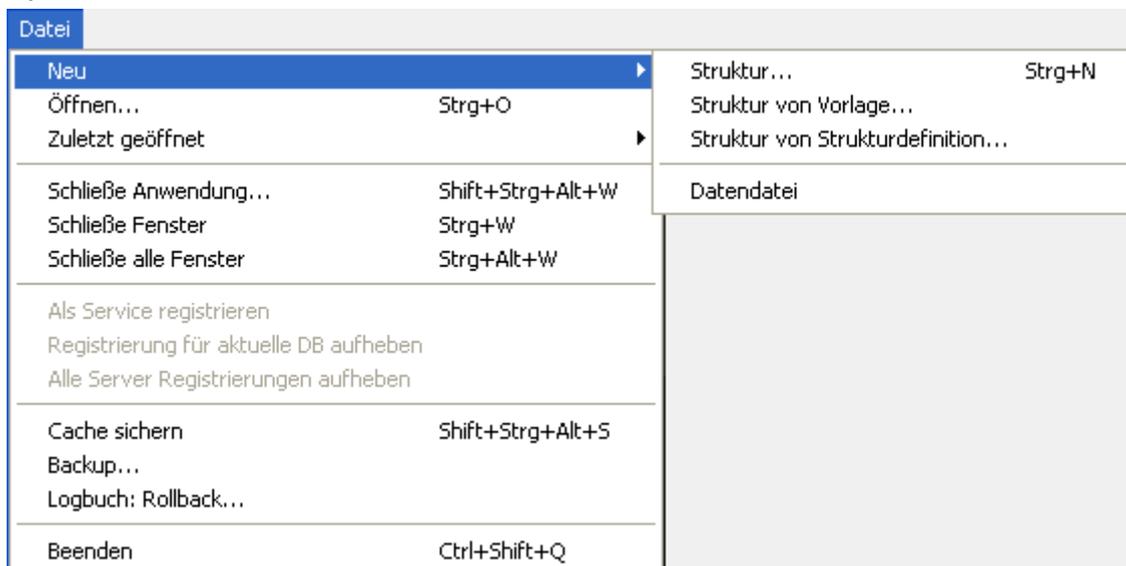
Um eine Server Datenbank zu erstellen bzw. eine vorhandene Datenbank zu öffnen, starten Sie 4D Server durch Doppelklick auf das Programm-Icon für 4D Server.



Über das Menü **Datei/Ablage** des 4D Server können Sie eine neue Datenbank erstellen oder eine bereits vorhandene öffnen.

### Datenbank erstellen

Um eine neue Datenbank zu erstellen, können Sie im Untermenü **Neu** zwischen drei Optionen wählen:



- **Struktur** erstellt eine leere Datenbank, d.h. sie enthält weder Tabellen, Formulare noch eine vordefinierte Oberfläche. Wählen Sie diesen Befehl, erscheint ein Standard-Sichern Dialog, in dem Sie Name und Speicherort der Datenbank festlegen können.
- **Struktur von Vorlage** erstellt eine Datenbank mit einer einsatzbereiten Vorlage, die Sie dann an Ihre eigenen Bedürfnisse anpassen können. Damit Sie diese Funktion nutzen können, muss die Datenbankvorlage in einem Ordner 4D Templates oder 4D Modelle abgelegt werden, der auf derselben Ebene wie die Datei 4D Server.exe (Windows), bzw. das 4D Server Software Paket (Mac OS) liegt. Wählen Sie diesen Menübefehl, erscheint ein Dialogfenster mit verschiedenen Vorlagen zur Auswahl.
- **Struktur von Strukturdefinition** erstellt eine Datenbank, die auf einer Strukturdefinition im XML Format basiert. Die Definition kann von einer Struktur stammen, die aus 4D oder einer beliebigen Design-Anwendung exportiert wurde. Wählen Sie diesen Menübefehl, erscheint ein Standard-Öffnen Dialog, in dem Sie die entsprechende XML Datei definieren können.

## Datenbank öffnen

---

Um eine vorhandene Datenbank zu öffnen, wählen Sie im Menü **Datei/Ablage>Öffnen** oder bei einer schon einmal geöffneten Datenbank im Menü **Datei/Ablage>Öffnen zuletzt verwendet**. Ist die Datenbank bereits geöffnet, wenn Sie den Befehl Öffnen wählen, wird sie zuerst geschlossen. Evtl. angemeldete Client-Rechner werden mit der Option "Warte bis alle Anwender abgemeldet sind" abgemeldet.

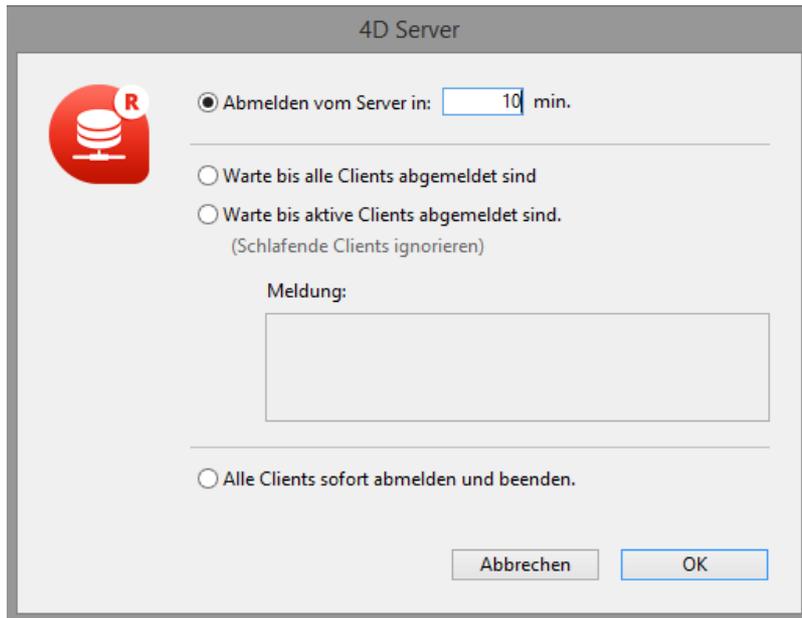
**Hinweis:** Sie können eine vorhandene Datenbank auch direkt öffnen, wenn Sie die interpretierte bzw. kompilierte Strukturdatei (.4db bzw. .4dc) oder die Datei .4dlink auf das Programm-Icon 4D Server ziehen.

## 4D Server beenden

Um 4D Server zu beenden, gehen Sie folgendermaßen vor:

**1. Wählen Sie unter Windows im Menü Datei von 4D Server, auf OS X im Menü 4D Server der Anwendung den Befehl Beenden.**

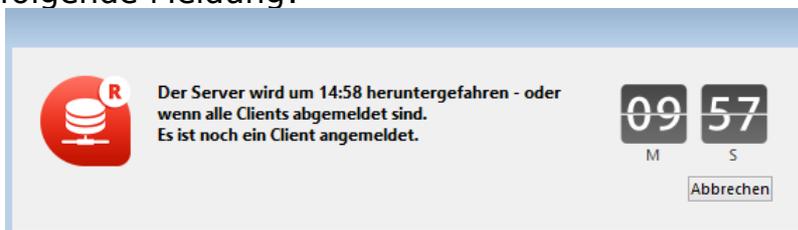
Auf dem Bildschirm des Server Rechners erscheint folgendes Dialogfenster:



**2. Geben Sie entweder die Zeitspanne an, nach der Server beenden soll oder wählen Sie die Option "Warte bis alle Clients abgemeldet sind".**

Sobald Sie eine Option gewählt haben, kann sich am Server kein neuer Client anmelden.

- Wählen Sie **Abmelden vom Server in ... min**, erscheint auf dem Bildschirm folgende Meldung:



Gleichzeitig erhalten alle Arbeitsstationen eine entsprechende Meldung. Der Benutzer kann die noch verbleibende Zeit weiterarbeiten. Dieses Dialogfenster wird alle 20 Sekunden wiederholt, um die Arbeitsstationen zum Beenden aufzufordern. Ist die Zeit abgelaufen, beendet der Server, auch wenn noch Client-Rechner angemeldet sind.

- **Warte bis alle Anwender abgemeldet sind**  
Der Server beendet erst, wenn sich alle Benutzer abgemeldet haben, inkl. Clients im Schlafmodus. Diese Option kann z.B. für Wartungsoperationen

während der Mittagspause ungeeignet sein, wenn sich mehrere Clients im Schlafmodus befinden.

- **Warte bis aktive Clients abgemeldet sind (Schlafende Clients ignorieren)**

Der Server beendet, wenn sich alle *aktiven* Clients abgemeldet haben, d.h. solche, die nicht im Schlafmodus sind. Clients im Schlafmodus gelten als nicht angemeldet. Mit dieser Option können Sie auch während der Mittagspause Wartungsoperationen durchführen.

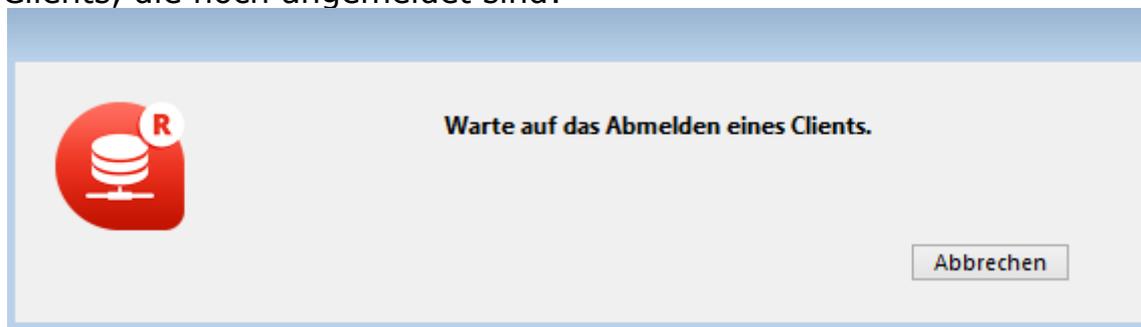
Clients im Schlafmodus erhalten dann beim Erwachen einen Verbindungsfehler.

**Hinweis:** Für diese Option ist die Netzwerkschicht *ServerNet* erforderlich.

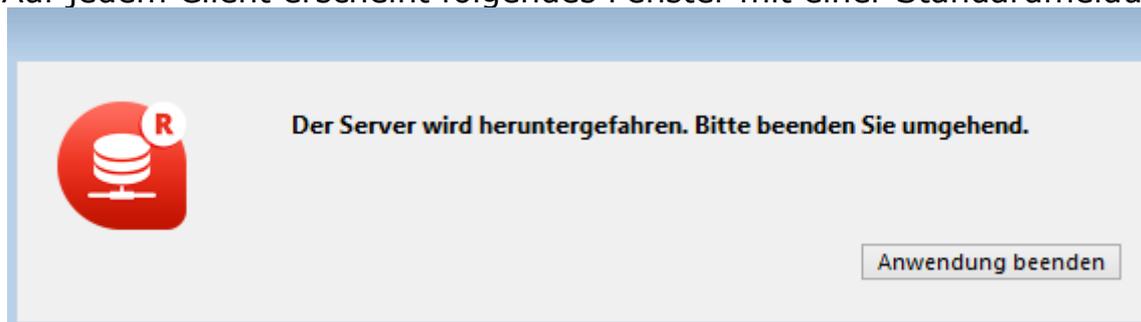
Weitere Informationen dazu finden Sie im Abschnitt **Neue Netzwerk-Schicht ServerNet (Kompatibilität)**.

**Hinweis:** Ein *schlafender Client* ist eine remote 4D Applikation auf einem Rechner, der in den Schlafmodus gewechselt hat, wenn die Verbindung zum Server aktiv ist. Mehr Informationen dazu finden Sie im Abschnitt **Handhabung schlafender Benutzer**.

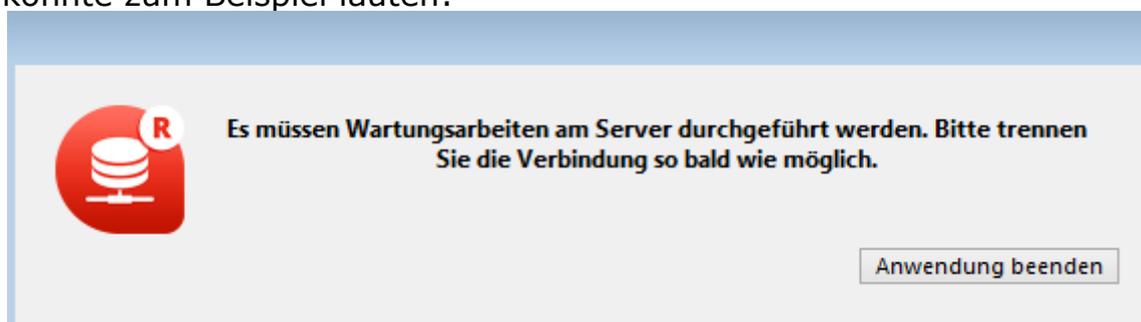
Wählen Sie eine dieser Optionen, erscheint ein Fenster mit der Anzahl der Clients, die noch angemeldet sind:



Auf jedem Client erscheint folgendes Fenster mit einer Standardmeldung:



Haben sie im Abmeldedialog auf dem Server eine eigene Meldung eingegeben, erscheint stattdessen diese Meldung auf jedem Client. Sie könnte zum Beispiel lauten:



- **Alle Clients sofort abmelden und beenden**

Der Server beendet alle Prozesse und Verbindungen und beendet nach wenigen Sekunden.

## 4D Server Menüs

---

Die Oberfläche von 4D Server enthält folgende Menüs: **Datei/Ablage**, **Bearbeiten**, **Fenster**, **Hilfe**

Auf Mac OS sind einige Befehle im Menü **4D Server** der Anwendung:

### Datei/Ablage

---



### Neu

---

Dieser Befehl hat Untermenüs, um eine Struktur oder eine neue Datendatei auf dem Server Rechner zu erstellen.

Weitere Informationen dazu finden Sie im Abschnitt **Neue 4D Server Datenbank erstellen**.

### Öffnen/Zuletzt geöffnet

---

Über diese Befehle können Sie eine Anwendung mit 4D Server öffnen.

Weitere Informationen dazu finden Sie im Abschnitt **Neue 4D Server Datenbank erstellen**.

### Zuletzt geöffnet

---

Dieser Befehl zeigt im Untermenü die zuletzt von 4D Server geöffneten Anwendungen. Markieren Sie die gewünschte Anwendung, um sie zu öffnen. Ist die Datenbank bereits geöffnet, wenn Sie diesen Befehl wählen, wird sie zuerst geschlossen. Evtl. angemeldete Client-Rechner werden mit der Option "Warte bis

alle Anwender abgemeldet sind" abgemeldet.

Um dieses Menü zurückzusetzen, wählen Sie den Befehl **Lösche Menü**.

## Schließe Anwendung

---

Damit schließen Sie die aktuelle Anwendung, ohne 4D Server zu beenden. Auf dem Bildschirm erscheint dann die Meldung zum Beenden des Server, um die entsprechende Option zum Abmelden der Clients zu wählen. Weitere Informationen dazu finden Sie im Abschnitt **4D Server beenden**.

## Schließe Fenster

---

Damit schließen Sie in 4D Server das Fenster im Vordergrund.

## Schließe alle Fenster

---

Damit schließen Sie alle Fenster im Programm 4D Server. Beachten Sie, dass in diesem Fall nur, wenn der Befehl **Schließe Anwendung** im Menü **Datei/Ablage** aktiviert ist, angibt, ob die Anwendung noch veröffentlicht ist.

## Als Service registrieren/Registrierung für aktuelle DB aufheben/Alle 4D Server Registrierungen aufheben

---

Über diese Befehle können Sie die Registrierung der Anwendung als Service verwalten. Weitere Informationen dazu finden Sie im Abschnitt **Datenbank als Dienst registrieren**.

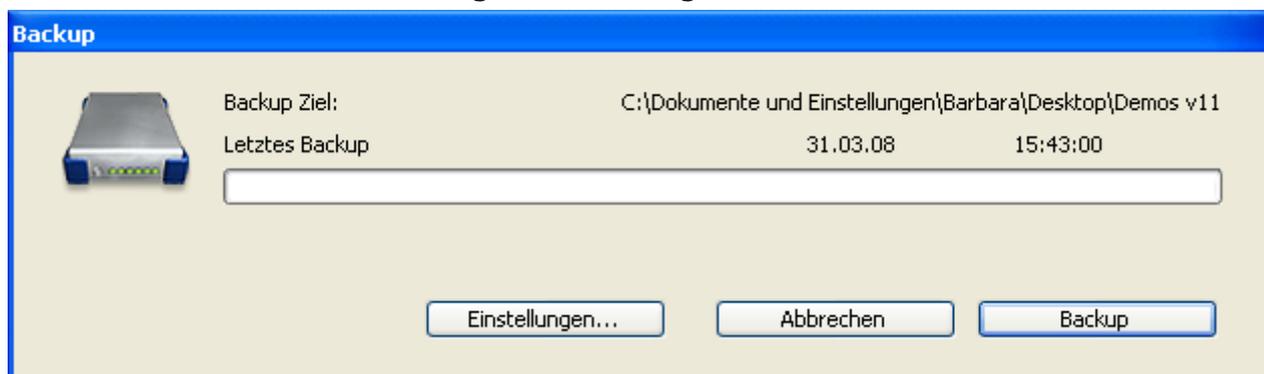
## Cache sichern

---

Dieser Befehl leert den Daten-Cache und schreibt die Daten auf die Festplatte. 4D Server leert den Cache automatisch nach der Zeitspanne, die in den Datenbank-Eigenschaften auf der Seite **Speicher** festgelegt wurde.

## Backup

Mit diesem Befehl können Sie jederzeit ein Backup der Datenbank starten. Auf dem Bildschirm erscheint dann folgendes Dialogfenster:



- Die Schaltfläche **Backup** startet sofort ein Backup mit den Parametern, die in den Datenbank-Eigenschaften definiert wurden (Dateien für das Backup, Zielverzeichnis, Anzahl der aufbewahrten Backup-Sätze, etc.)

- Die Schaltfläche **Einstellungen** öffnet die Einstellungen unter dem Thema **Backup**. Hier können Sie die aktuellen Einstellungen für das Backup ansehen und bei Bedarf verändern.
- Die Schaltfläche **Abbrechen** unterbricht den laufenden Prozess der Datensicherung.

Weitere Informationen zur Backup-Konfiguration finden Sie im Kapitel **Backup und Anwendung wiederherstellen** des Handbuchs *4D Designmodus*.

## Wiederherstellen

---

Dieser Befehl zeigt den Standard-Öffnen Dialog, um das Backup für die Wiederherstellung auszuwählen.

## Beenden

---

Mit diesem Befehl können Sie 4D Server abschalten. Weitere Informationen dazu finden Sie im Abschnitt **4D Server beenden**.

**Hinweis:** Auf OS X liegt der Befehl **Beenden** im Menü **4D Server** der Anwendung. Wählen Sie unter Windows im Menü **Datei**, auf Mac OS im Menü **4D Server** der Anwendung den Befehl **Beenden**.



## Bearbeiten

---

Das Menü **Bearbeiten** von 4D Server enthält die Standardbefehle **Kopieren/Einfügen**, den Befehl **Zwischenablage**, etc.



Unter Windows enthält dieses Menü auch den Befehl **Einstellungen**, der das Dialogfenster Datenbank-Eigenschaften anzeigt. Weitere Informationen dazu finden Sie im Handbuch *4D Designmodus*. Die spezifischen Einstellungen für 4D Server

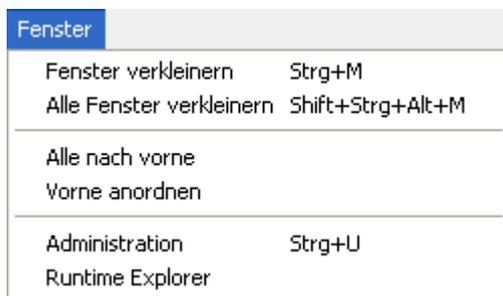
finden Sie in den Abschnitten **Netzwerk und Client-Server Optionen** und **IP Einstellungen**.

**Hinweis:** Auf Mac OS liegt der Befehl **Einstellungen** im Menü **4D Server** der Anwendung.



## Fenster

---



Der obere Teil enthält Standardbefehle zum Verwalten der Arbeitsfenster (die Befehle variieren je nach Plattform). Dieses Menü enthält auch Befehle zum Anzeigen serverspezifischer Fenster.

## Administration

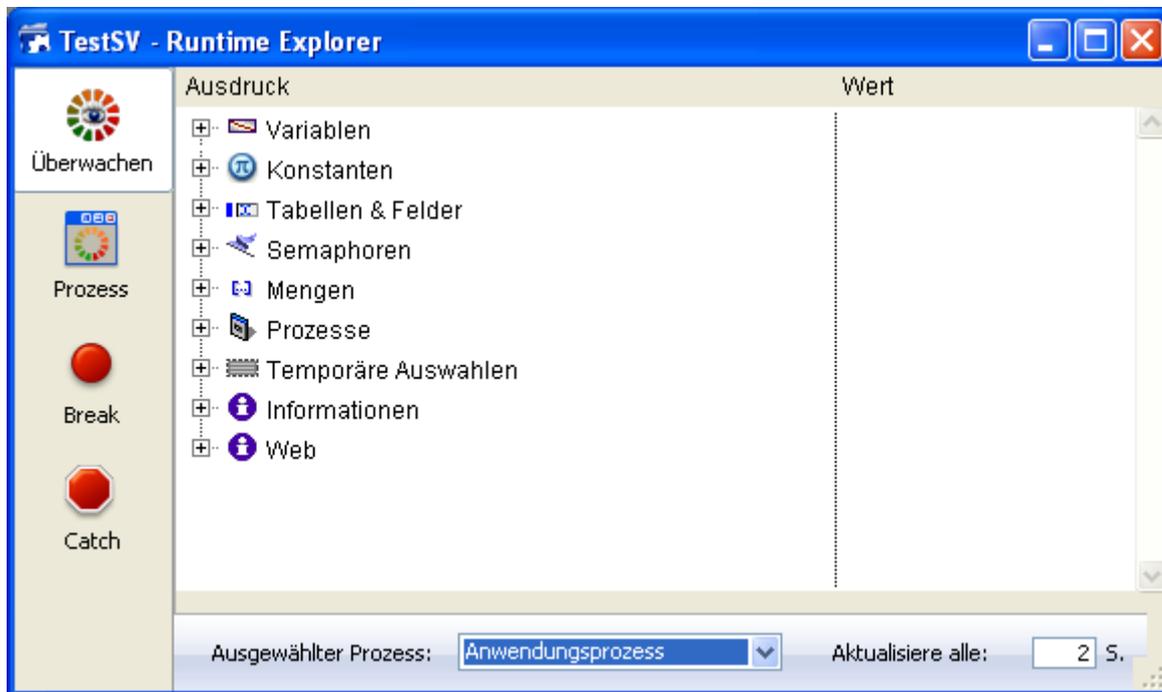
---

Dieser Befehl öffnet das 4D Server Verwaltungsfenster, wenn es geschlossen oder minimiert wurde. Weitere Informationen dazu finden Sie im Kapitel 4D Server Verwaltungsfenster (siehe **Seite Monitor**).

## Runtime Explorer

---

Dieser Befehl öffnet das Fenster Runtime Explorer von 4D Server.

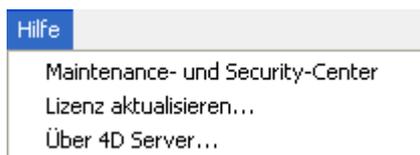


Mit dem Runtime Explorer können Sie den Status verschiedener Strukturelemente der Datenbank ansehen und prüfen, ob die verfügbaren Ressourcen korrekt verwaltet werden. Der Runtime Explorer ist besonders hilfreich beim Entwickeln oder Analysieren einer Datenbank.

Das Fenster enthält am linken Rand vier Schaltflächen, über die Sie auf die Seiten Überwachen, Prozess, Break und Catch gelangen. Er funktioniert in 4D Server auf dieselbe Art und Weise wie in 4D. Weitere Informationen dazu finden Sie im Handbuch *4D Designmodus* im Kapitel **Runtime Explorer**.

## Hilfe

---



## Maintenance und Security-Center

---

Dieser Befehl öffnet das Wartungs- und Sicherheits-Center (MSC). Es enthält die Werkzeuge zum Prüfen, Warten, Sichern und Komprimieren der Daten- und Strukturdateien.

Dieser Befehl ist auch verfügbar, wenn keine Anwendung von 4D Server geöffnet wurde. Dann lässt sich eine Anwendung im Wartungsmodus öffnen. Der Befehl zeigt den Standard-Öffnen Dialog an, in dem Sie die entsprechende Anwendung öffnen können. Im Wartungsmodus können Sie beschädigte Anwendungen öffnen oder komprimieren.

Weitere Informationen dazu finden Sie im Handbuch *4D Designmodus* im Kapitel **Maintenance und Security Center**.

## Lizenz aktualisieren

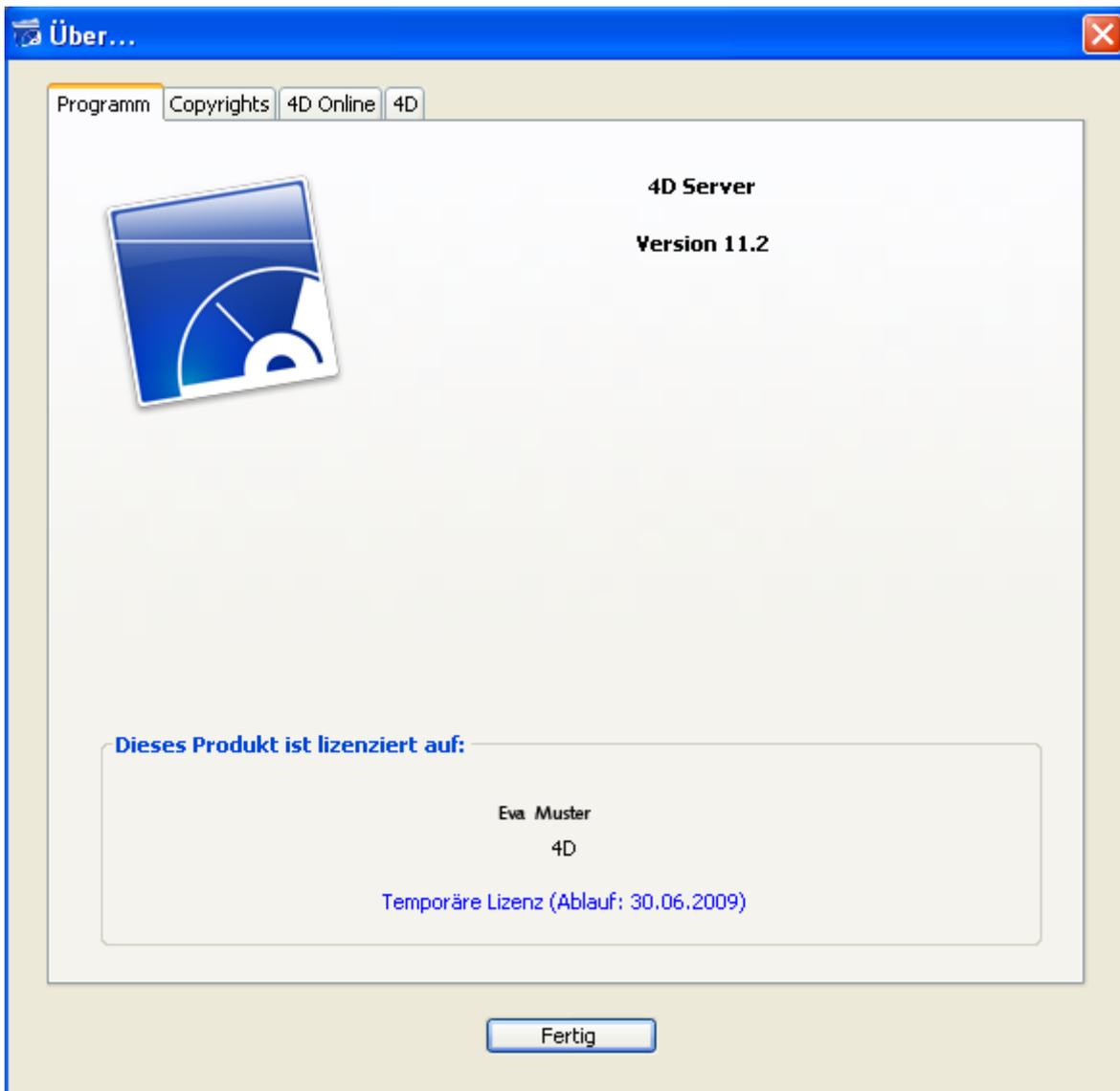
---

Dieser Befehl öffnet das Fenster zum Aktivieren zusätzlicher Lizenzen in Ihrer 4D Umgebung. Weitere Informationen dazu finden Sie im **Installationshandbuch**.

## Über 4D Server

---

Dieser Befehl zeigt das Fenster **Über 4D Server**. Es enthält Registerkarten, um verschiedene Seiten mit Informationen zum Server aufzurufen.



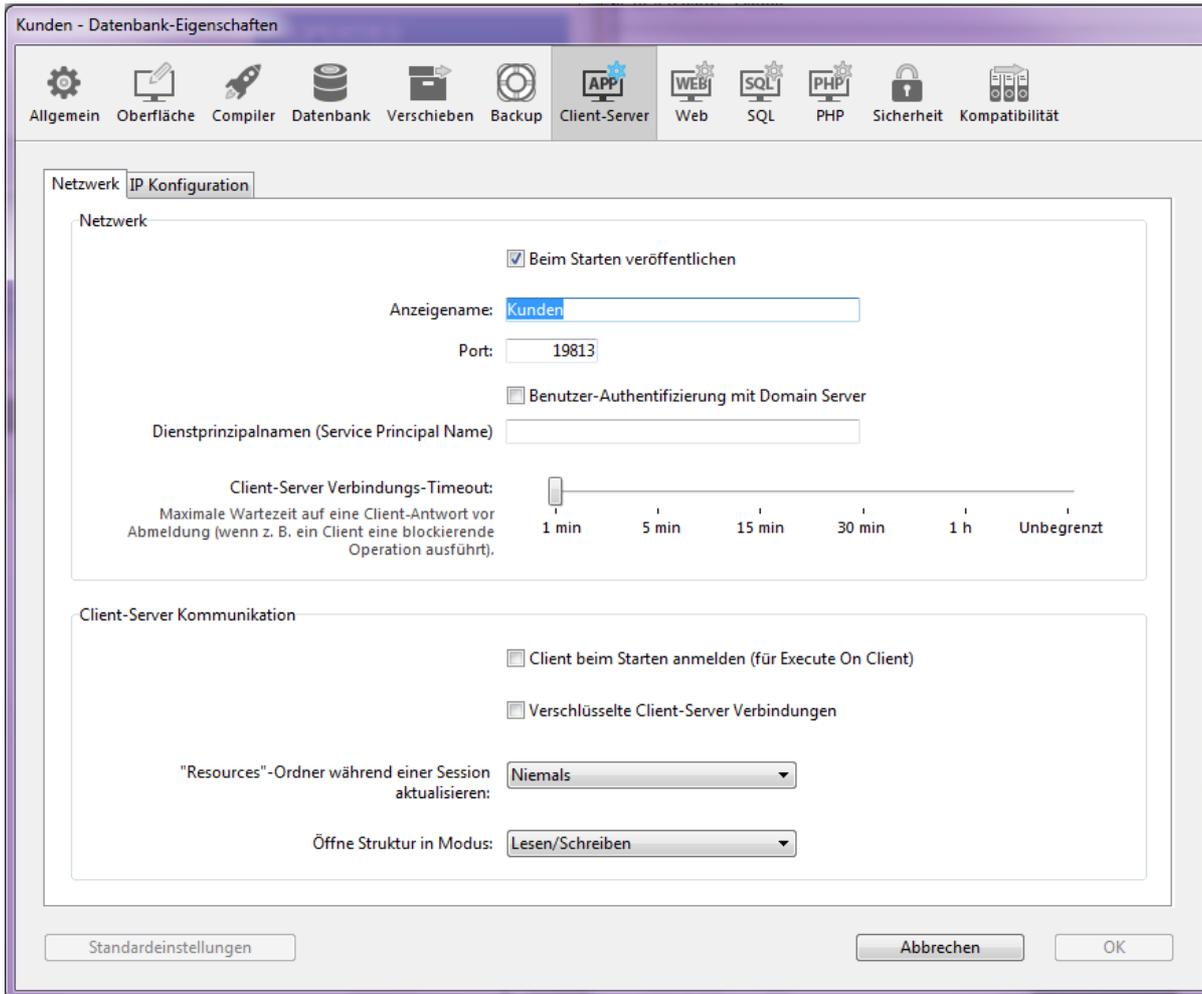
Programm: 4D Server Version und Lizenz

- Copyrights: Datenschutzbestimmungen
- 4D Online: Zusätzliche Ressourcen zu 4D, die online aufrufbar sind
- 4D: Überblick über das Unternehmen 4D SAS weltweit.

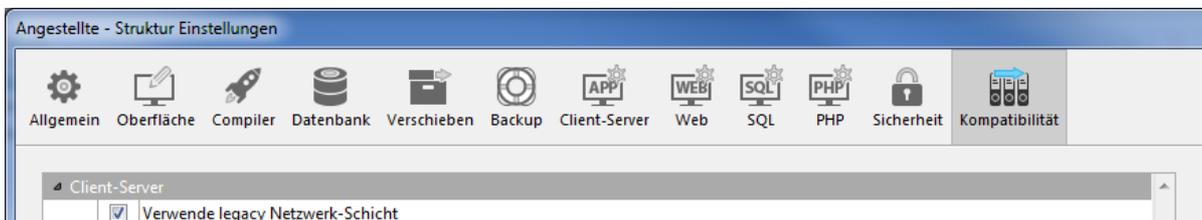
**Hinweis:** Auf Mac OS liegt der Befehl **Über 4D Server** im Menü **4D Server** der Anwendung.

## Netzwerk und Client-Server Optionen

Sie können in den Datenbank-Eigenschaften auf der Seite Client-Server die Parameter für das Netzwerk und die Client/Server Kommunikation festlegen. Sie ist für 4D im remote Modus und 4D Server zugänglich.



Außerdem gibt es ab 4D Server v14 R5 eine Option zur Kompatibilität, mit der Sie jederzeit die bisherige Netzwerk-Schicht aktivieren oder deaktivieren können:



Ausführliche Informationen dazu finden Sie im nächsten Abschnitt.

## Netzwerk

### Beim Starten veröffentlichen

Mit dieser Option geben Sie an, ob die 4D Server Datenbank im Verbindungsdialog in der Liste der veröffentlichten Datenbanken erscheint.

- Ist diese Option markiert (Standard), wird die Datenbank veröffentlicht und erscheint in der Liste der veröffentlichten Datenbanken (Seite **Verfügbar**).
- Ist diese Option nicht markiert, wird die Datenbank nicht veröffentlicht und erscheint nicht in der Liste. Benutzer müssen, um sich anzumelden, die Adresse der Datenbank manuell auf der Seite **Angepasst** des Verbindungsdialogs eingeben.

**Hinweis:** Verändern Sie diesen Parameter, müssen Sie die Server Datenbank neu starten, damit die Änderung berücksichtigt wird.

## Anzeigename

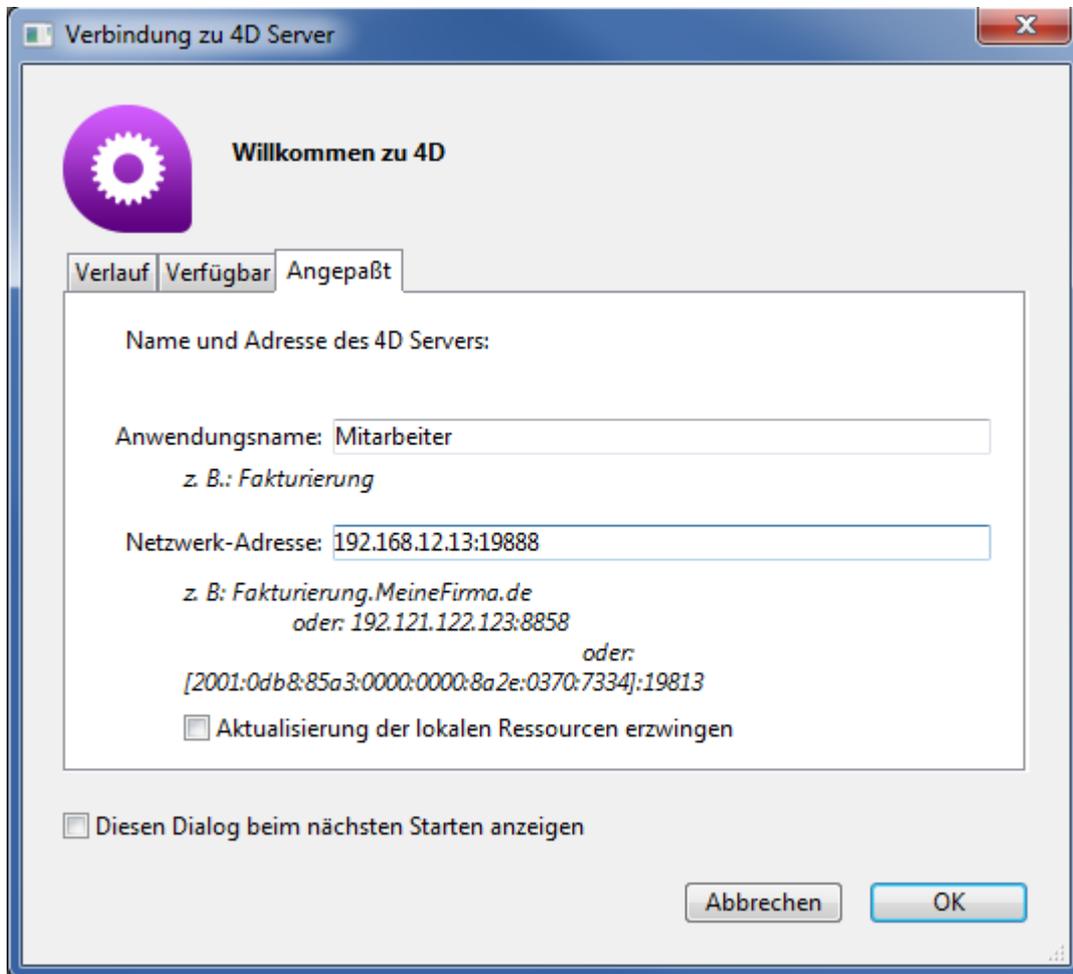
Der Name zum Veröffentlichen einer 4D Server Datenbank lässt sich ändern. Das kann z.B. der Name auf der dynamischen Veröffentlichungsseite **Verfügbar** im Verbindungsdialog sein (siehe Abschnitt **Anmeldung an eine 4D Server Datenbank**).

4D Server verwendet standardmäßig den Namen der Strukturdatei der Datenbank. Sie können einen beliebigen Namen eingeben.

**Hinweis:** Dieser Parameter wird bei eigenen Client-/Server-Anwendungen nicht berücksichtigt. Im Prinzip meldet sich die Client-Anwendung direkt an die Server Anwendung an, ohne über den Anmeldedialog zu gehen. Der Dialog erscheint jedoch, wenn ein Fehler auftritt; in diesem Fall ist der Publikationsname der Server-Anwendung der Name der kompilierten Datenbank.

## Port Nummer

Mit dieser Option können Sie die TCP Port Nummer ändern, unter der 4D Server die Datenbank veröffentlicht. Diese Information wird in der Strukturdatei der Datenbank und auf jedem Client-Rechner gespeichert. 4D Server und 4D im remote Modus verwenden standardmäßig die Port Nummer 19813 (+19814). Sie müssen diesen Wert anpassen, wenn Sie mehrere 4D Anwendungen auf demselben Rechner mit dem TCP Protokoll verwenden wollen; in diesem Fall müssen Sie für jede Anwendung eine andere Port Nummer vergeben. Ändern Sie diesen Wert auf 4D Server bzw. 4D, wird er automatisch auf alle 4D Rechner übertragen, die an die Datenbank angemeldet sind. Um andere nicht angeschlossene Client-Rechner zu aktualisieren, geben Sie bei der nächsten Anmeldung im Verbindungsdialog auf der Seite **Angepasst** die neue Port Nummer ein, getrennt durch Doppelpunkt. Sie wird an die IP-Adresse des Server-Rechners angehängt, getrennt durch Doppelpunkt. Ist die Nummer z.B. 19888, sieht der Eintrag folgendermaßen aus:



**Hinweis:** Verwendet ein Server IPv4, sind auf der Seite **Verfügbar** nur die auf Port 19813 veröffentlichten Datenbanken sichtbar.

## 4D Server und Port Nummern

4D Server verwendet drei TCP Port zur Kommunikation zwischen internen Servern und Clients:

- **SQL Server:** standardmäßig 19812 (lässt sich über die Seite "SQL" der Datenbank-Eigenschaften ändern).
- **Application Server:** standardmäßig 19813 (lässt sich in den Einstellungen auf der Seite "Client-Server/Konfiguration" verändern, siehe unten).
- **DB4D Server** (database server): standardmäßig 19814. Diese Port Nummer lässt sich nicht direkt ändern, sie besteht aber immer aus der Port Nummer des Application Server + 1.  
Meldet sich ein remote 4D an 4D Server an, verwendet es den TCP Port des Application Server (19813 oder den Port, der im Verbindungsdialog in der IP Adresse nach dem Doppelpunkt ':' angezeigt wird.). Die Anmeldung an andere Server über ihre entsprechenden Ports ist dann automatisch; sie müssen nicht mehr eigens spezifiziert werden.  
Beachten Sie, dass die drei TCP Port bei Zugriff über einen Router oder eine Firewall explizit geöffnet werden müssen.

## Benutzer-Authentifizierung mit Domain Server

Mit dieser Option können Sie Ihre Client-Server Lösungen unter Windows mit SSO (*Single Sign On* - Einmalanmeldung) ausrüsten. Markieren Sie diese Option, verbindet sich 4D direkt mit der Active Directory Zone des Windows Domain Server und erhält die verfügbaren Tokens der Authentifizierung.

Weitere Informationen dazu finden Sie im Abschnitt **Single Sign On (SSO) unter Windows**.

## Dienstprinzipalname

Ist Single Sign On (SSO) aktiviert (siehe oben), müssen Sie dieses Feld ausfüllen, wenn Sie Kerberos als Authentifizierungsprotokoll nutzen wollen.

Weitere Informationen dazu finden Sie im Abschnitt **Single Sign On (SSO) unter Windows**.

## Client-Server Verbindungs-Timeout

Im Ablaufbalken legen Sie das Timeout fest, d.h. die Dauer an Inaktivität, nach der die Verbindung zwischen 4D Server und den angeschlossenen Client-Rechnern beendet wird.

Bei **Ohne** gibt es kein Timeout. In diesem Fall erfolgt keine Kontrolle der Client-Aktivität, d.h ein inaktiver Client bleibt auf unbestimmte Zeit an den Server angeschlossen.

Ist ein Timeout eingestellt, beendet der Server die Verbindung zu einem Client, wenn er innerhalb der vorgegebenen Zeitspanne keine Anfragen von diesem Client erhält.

## Optionen

---

### Client beim Starten anmelden

Ist diese Option markiert, können alle Arbeitsstationen, die an die 4D Server Anwendung angemeldet sind, Methoden im remote Modus ausführen. Weitere Informationen dazu finden Sie im Abschnitt **Serverprozeduren auf Client-Rechnern**.

### Verschlüsselte Client-Server Verbindungen

Damit können Sie den gesicherten Modus zur Kommunikation zwischen Server Rechner und Rechnern mit remote 4D aktivieren. Weitere Informationen dazu finden Sie im Abschnitt **Client/Server Verbindungen verschlüsseln**.

### 'Resources'-Ordner während einer Session aktualisieren

Mit dieser Einstellung können Sie den Aktualisierungsmodus für den Ordner **Resources** auf den angeschlossenen Rechnern global setzen, wenn der Ordner **Resources** während einer Arbeitssitzung verändert wird. (Beim Öffnen einer neuen Arbeitssitzung wird er immer automatisch auf dem remote Rechner synchronisiert). Es gibt drei Möglichkeiten:

- **Nie:** Der lokale Ordner **Resources** wird während der Arbeitssitzung nicht aktualisiert. Die vom Server gesendete Meldung wird ignoriert. Dieser Ordner lässt sich bei Bedarf manuell über den Menüeintrag **Lokale Ressourcen aktualisieren** des Menüs "Aktionen" aktualisieren (siehe unter **Ressourcen Explorer verwenden**).
- **Immer:** Der lokale Ordner **Resources** wird während der Arbeitssitzung automatisch synchronisiert, wenn der Server eine Meldung sendet.
- **Fragen:** Sendet der Server eine Meldung, erscheint auf den Client-Rechnern ein Dialogfenster, das die Änderung meldet. Der Benutzer kann dann die Synchronisation des lokalen Ordners **Resources** bestätigen oder abweisen.

Der Ordner **Resources** bündelt die eigenen Dateien für die Oberfläche der Anwendungen, z.B. Bilder, Übersetzungsdateien. Die automatische oder manuelle Operation ermöglicht, jeden Client zu informieren, wenn sich der Inhalt dieses Ordners geändert hat. Weitere Informationen dazu finden Sie im Abschnitt **Ordner Resources verwalten**.

**Hinweis:** Diese Einstellmöglichkeit richtet sich an Team-Entwickler, sie ist für Anwender einer Datenbank belanglos.

## Öffne Struktur im Modus

Damit definieren Sie den Öffnen-Modus der Datenbankstruktur auf den Client-Rechnern. Standardmäßig ist der Modus **Lesen/Schreiben** eingestellt. Sie können aber auch den Modus **Nur Lesen** wählen, um Änderungen an der Struktur zu verhindern.

## IP Konfiguration

---

### Konfiguration Erlauben/Verweigern-Tabelle

Mit dieser Tabelle setzen Sie die Regeln für den Zugriff auf die Datenbank. Das richtet sich nach den IP-Adressen des remote-Rechners. Das sorgt z.B. bei Anwendungen mit sensiblen Daten für mehr Sicherheit.

**Hinweis:** Diese Tabelle steuert nicht die Web Verbindungen.

Diese Tabelle funktioniert folgendermaßen:

- In der Spalte "Erlauben-Verweigern" wählen Sie im PopUp-Menü die anzuwendende Regel, d.h. **Erlauben** oder **Verweigern**. Um eine Regel hinzuzufügen, klicken Sie auf die Schaltfläche **Hinzufügen**. In der Tabelle erscheint dann eine neue Zeile. Über die Schaltfläche **Löschen** können Sie die aktuelle Zeile entfernen.
- In der Spalte "IP Adresse" setzen Sie die IP-Adresse(n), für welche die Regel gelten soll. Dazu klicken Sie in die Spalte und geben die Adresse in folgender Form ein: 123.45.67.89 (IPv4 Format) oder 2001:0DB8:0000:85A3:0000:0000:AC1F:8001 (IPv6 Format). Sie können auch nur den Anfang der Adresse eingeben, gefolgt von einem \* (Stern). Beispiel: 192.168.\* gibt alle Adressen an, die mit 192.168 beginnen.
- Die Anwendung der Regeln richtet sich nach der Reihenfolge der Anzeige in der Tabelle. Bei zwei widersprüchlichen Regeln hat die zuerst angelegte Regel Priorität. Durch Ändern der aktuellen Sortierung können Sie die Zeilen neu ordnen (Klicken Sie auf den Kopfteil der Spalte, um die Sortierrichtung zu wechseln). Sie können die Zeilen auch per Drag&Drop bewegen.
- Aus Sicherheitsgründen ist die Anmeldung nur an Adressen möglich, die zu einer Regel passen, d.h. enthält die Tabelle eine bzw. mehrere Regeln Verweigern, werden alle Adressen abgewiesen, da keine zu mindestens einer Regel passt. Wollen Sie nur einige Adressen abweisen, andere jedoch zulassen, fügen Sie am Ende der Tabelle die Regel **Erlauben \*** hinzu.

#### Beispiel:

Verweigere 192.168.\* (weist alle Adressen ab, die mit 192.168 beginnen)  
Erlaube \* (lässt jedoch alle anderen Adressen zu)

4D Server verwendet standardmäßig keine Verbindungseinschränkungen, d.h. die erste Zeile der Tabelle enthält die Bezeichnung **Erlaube** und das Zeichen \* (alle Adressen sind zugelassen).

## Neue Netzwerk-Schicht ServerNet (Kompatibilität)

Ab dem Release v14 R5 enthalten 4D Applikationen die neue Netzwerk-Schicht (network layer) *ServerNet* für die Kommunikation zwischen 4D Server und remote 4D Rechnern (Clients). *ServerNet* basiert auf einer modernen und robusten API, ist leicht zu warten und ermöglicht uns, die neuesten Netzwerktechnologien zu implementieren, die auch für hohe Performance sorgen.

Die Verwendung von *ServerNet* ist für den Benutzer transparent. Beachten Sie, dass beim Verwenden von *ServerNet* die Namen der Datenbanken, die im gesicherten Modus veröffentlicht werden, nicht mit dem Zeichen "^" gekennzeichnet werden. Das Zeichen wird nur beim Verwenden der "legacy" Netzwerkschicht gesetzt. Weitere Informationen dazu finden Sie im Abschnitt **Client/Server Verbindungen verschlüsseln**.

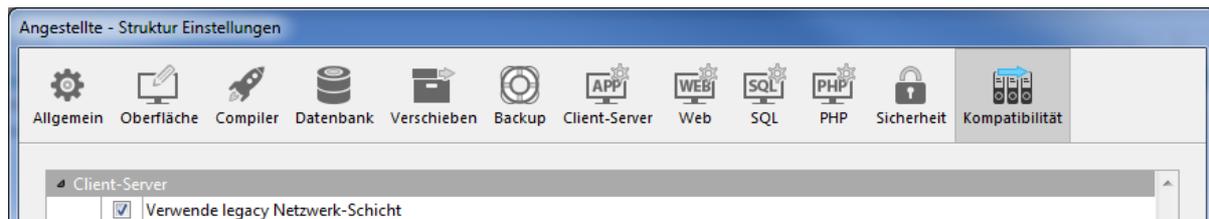
Die "legacy" Netzwerk-Schicht wird zur Wahrung der Kompatibilität mit vorhandenen Anwendungen weiter unterstützt. Neu angelegte Datenbanken verwenden automatisch *ServerNet*.

Es gibt Optionen, um *ServerNet* zu aktivieren/deaktivieren. Wir empfehlen, *ServerNet* in all Ihren Lösungen nach und nach zu aktivieren, damit Ihre Anwendungen von zukünftigen Netzwerk Evolutionen profitieren.

### Die legacy Netzwerk-Schicht aktivieren oder deaktivieren

Sie können die legacy Netzwerk-Schicht in Ihrem 4D Server jederzeit aktivieren bzw. deaktivieren. Sie können entweder:

- Die Konstante Use legacy network layer mit dem Befehl **SET DATABASE PARAMETER** verwenden oder
- Die Option **Verwende legacy Netzwerk-Schicht** in den Datenbank-Eigenschaften (siehe **Seite Kompatibilität**):



Für die Netzwerk-Schicht *ServerNet* gilt standardmäßig:

- Sie ist in neuen Datenbanken, die mit 4D v14 R5 oder höher erstellt wurden, automatisch aktiviert
- Sie ist in konvertierten Datenbanken automatisch deaktiviert

### Doppelklickbare 4D Clients migrieren

Aktivieren Sie die Schicht *ServerNet* in Ihrer 4D Server Applikation, können sich nur dazu passende 4D Client Anwendungen anmelden:

- Client Anwendungen in Version 15 (beginnend mit 4D v14 R4) können sich ohne Änderung anmelden
- Client Anwendungen älterer Versionen (v14.x und alle vorherigen v14 R-Versionen) müssen erst aktualisiert werden, damit sie sich an den Server anmelden können.

Arbeitet Ihre Anwendung mit doppelklickbaren 4D Volume Desktop Clients älter als v14 R4, und wollen Sie den automatischen Mechanismus von 4D Server verwenden,

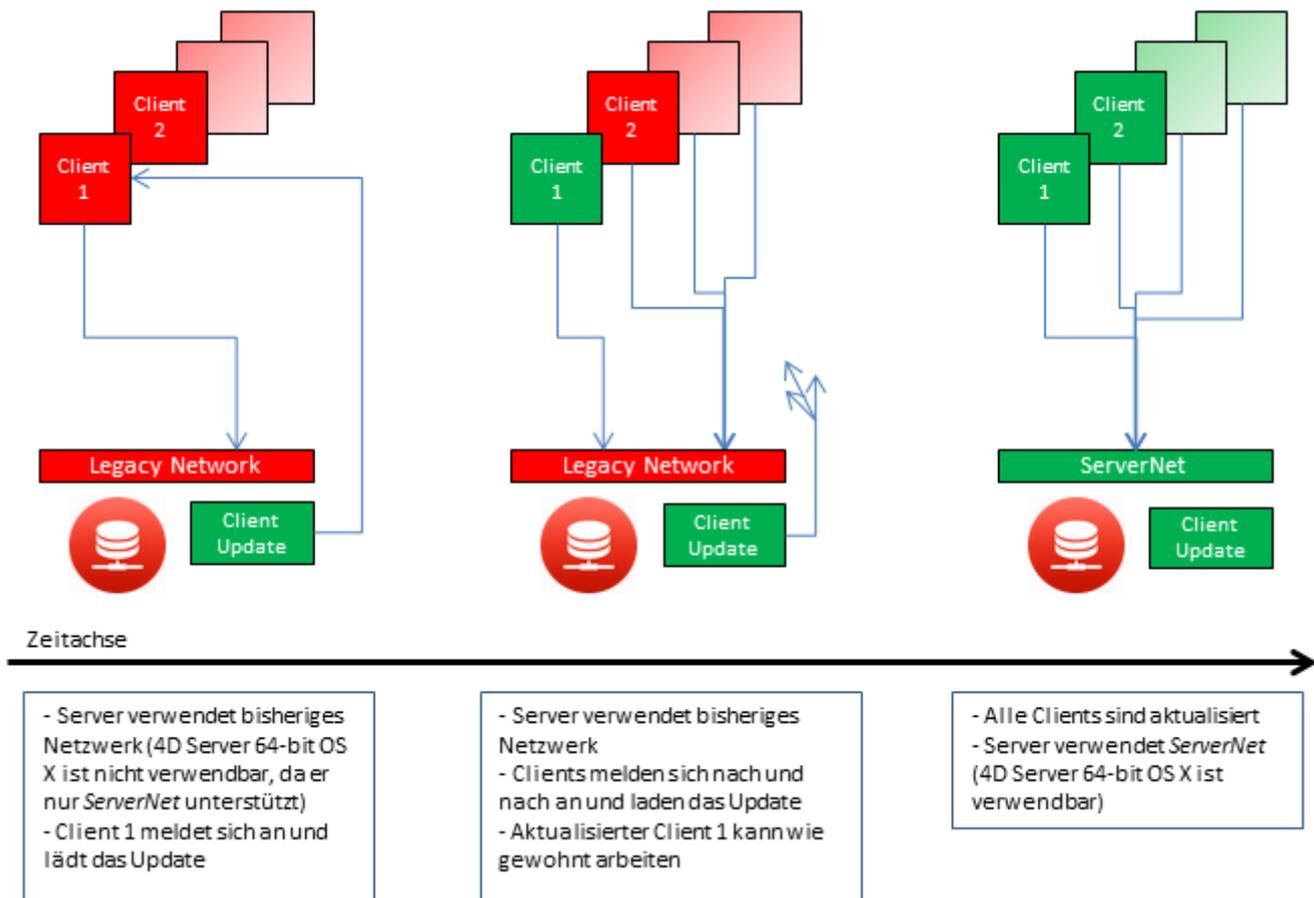
um Ihre aktualisierten Client Anwendungen über das Netzwerk zu verteilen, müssen Sie eine Strategie zum Migrieren einrichten. Dabei gelten folgende Prinzipien:

- Nicht-kompatible Clients können sich nur an einen 4D Server anmelden, der die legacy Netzwerk-Schicht verwendet.
- Aktualisierte Clients können ihre Protokolle dynamisch anpassen und sich so an 4D Server v15 und höher anmelden, egal, welche Netzwerk-Schicht der Server verwendet.

Die Strategie zum Migrieren sollte in folgenden Schritten erfolgen:

1. Erstellen Sie aktualisierte Client Anwendungen mit 4D v15 oder höher.
2. Starten Sie 4D Server v15 oder neuer mit der markierten Einstellung "Verwende legacy Netzwerk-Schicht".  
Mit dieser Konfiguration können sich alle Clients anmelden.
3. Warten Sie einen gewissen Zeitraum ab, bis sich alle Clients angemeldet und die neue Version geladen haben.  
Das kann einen Tag, eine Woche oder länger dauern. Während dieser Übergangsphase können sich sowohl bisherige als auch aktualisierte Clients an den Server mit der legacy Netzwerk-Schicht anmelden.
4. Sind alle Clients aktualisiert, können Sie die legacy Netzwerk-Schicht deaktivieren und schließlich auf dem Server auf *ServerNet* wechseln.

Nachfolgende Darstellung veranschaulicht diese Strategie:



## Client Anfragen protokollieren

Es empfiehlt sich, während dem Migrationsprozess das Logbuch "Diagnostic log recording" zu aktivieren. Hier protokolliert 4D Server dann jede Update-Anfrage des Client, so dass Sie den Prozess überwachen können. Dieses Logbuch wird nicht standardmäßig aktiviert. Sie müssen dazu den Befehl **SET DATABASE PARAMETER** aufrufen und die Konstante Diagnostic log recording auf 1 setzen.

Für jede Update Anfrage wird folgende Information protokolliert:

- IP Client
- Client Version
- Ereignis "Update Client"

Die Überprüfung dieses Logbuchs ist auch *nach* der Umstellung des Server auf die Netzwerkschicht *ServerNet* hilfreich, um sicherzustellen, dass alle Clients korrekt aktualisiert wurden. Versucht ein nicht-kompatibler Client, sich anzumelden, protokolliert der Server wie folgt:

- IP Client
- Client Version
- Ereignis "Anmeldung fehlgeschlagen"

In diesem Fall können Sie beispielsweise entscheiden, den Client manuell zu aktualisieren.

## IP Einstellungen

### Unterstützung von IPv6

---

Ab v16 R4 unterstützt der 4D Application Server die IPv6 Adressen Notation. Die Unterstützung von IPv6 ist transparent für Benutzer und 4D Entwickler: 4D Server akzeptiert ohne Unterschied entweder IPv6 oder IPv4 Verbindungen.

Die Unterstützung von IPv6 Adressen richtet sich jedoch nach den Versionen von 4D Server und 4D remote (32-bit oder 64-bit). Nachfolgende Tabelle zeigt die unterstützten Konfigurationen:

			Client					
			32-bit			64-bit		
			IPv4	IPv6	Both	IPv4	IPv6	Both
Server	32-bit	IPv4	IPv4		IPv4	IPv4		IPv4
		IPv6					IPv6	IPv6
		Both	IPv4		IPv4	IPv4	IPv6	IPv6
	64-bit	IPv4	IPv4		IPv4	IPv4		IPv4
		IPv6					IPv6	IPv6
		Both	IPv4		IPv4	IPv4	IPv6	IPv6

IPv4	Supported (IPv4)
IPv6	Supported (IPv6)
	Not supported

**Hinweis zur Kompatibilität:** IPv6 Unterstützung ist nur verfügbar, wenn auf 4D Server die Netzwerkschicht **ServerNet** aktiviert ist (*siehe **Neue Netzwerk-Schicht ServerNet (Kompatibilität)***).

Ausführliche Informationen über IPv6 finden Sie unter [RFC 2460](#).

## Client/Server Verbindungen verschlüsseln

---

Sie können Client/Server-Verbindungen so einstellen, dass die Arbeitsstationen mit remote 4D im gesicherten Modus mit 4D Server kommunizieren. Die gesicherte Client/Server Kommunikation basiert auf dem TLS Protokoll (Transport Layer Security), dem Nachfolger von SSL (Secured Socket Layer).

**Hinweis:** Einen allgemeinen Überblick über die Sicherheitsvorkehrungen in 4D finden Sie im [4D Security guide](#).

### **TLS Protokoll und Client/Server Verbindungen**

---

Das TLS Protokoll sichert die Kommunikation zwischen zwei Anwendungen ab, das sind z.B. Web Server und Browser. Das TLS Protokoll bürgt für die Identität von Sender und Empfänger, sowie für die Vertraulichkeit und Vollständigkeit der ausgetauschten Informationen. Eine ausführliche Beschreibung dieses Protokolls finden Sie im Handbuch *4D Programmiersprache* im Abschnitt **TLS Protokoll verwenden (HTTPS)**.

4D Server verwaltet die Operationen zur Authentifizierung und Datenintegrität zwischen 4D Server und remote 4D eigenständig. Der Benutzer muss keine weiteren Einstellungen vornehmen.

**Hinweis:** Das Verschlüsseln von Client und Server macht die Verbindung langsamer.

### **Einstellungen**

---

Auf Netzwerkebene wird das TLS Protokoll zwischen der TCP/IP Schicht (low level) und dem "high level" Protokoll eingefügt.

Zum Verwenden von TLS in einer klassischen Client/Server Architektur müssen Zertifikatsdateien vorhanden sein.

- *key.pem* (Dokument mit dem Schlüssel zur privaten Verschlüsselung)
  - *cert.pem* (Dokument mit Zertifikat).
- Diese Dateien müssen im Unterordner *Resources* der 4D und 4D Server Anwendungen liegen. Sie müssen auf dem Rechner mit 4D Server und auf allen Rechnern mit remote 4D vorhanden sein.
- Standardmäßige *key.pem* und *cert.pem* Dateien werden mit 4D geliefert. Zur Erhöhung der Sicherheitsebene empfehlen wir, diese Dateien durch Ihre eigenen Zertifikate zu ersetzen. Weitere Informationen dazu finden Sie im Abschnitt **Zertifikat erhalten** der Dokumentation zu 4D Web Server (die Verfahrensweise ist identisch).

Sie müssen auch die Verbindungen im gesicherten Modus aktivieren. Dazu öffnen Sie in den Datenbank-Eigenschaften die Seite **Client-Server** und wählen im Bereich "Client-Server Kommunikation" die Option **Verschlüsselte Client/Server Verbindungen** aus (siehe Abschnitt **Netzwerk und Client-Server Optionen**).

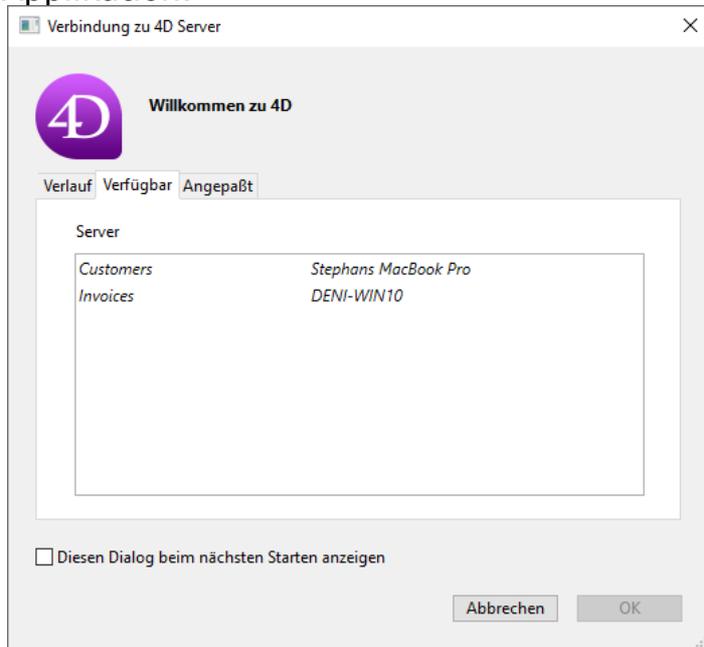
Diese Option ist standardmäßig nicht markiert. Sie wird erst nach Beenden und Neustart von 4D Server aktiv. Alle remote 4D Arbeitstationen melden sich dann im gesicherten Modus an.

## 4D Client Verbindung im gesicherten Modus

---

- **ServerNet verwenden**

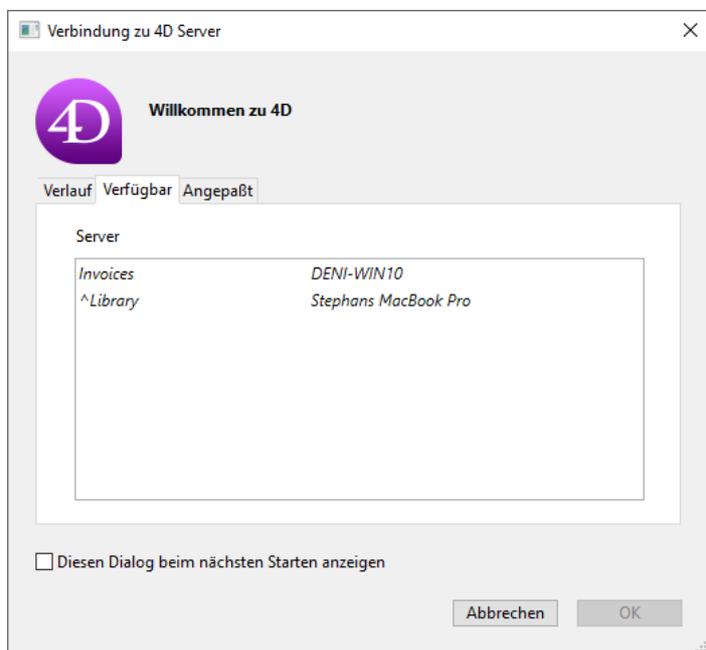
Bei Verwendung der neuen Netzwerkschicht *ServerNet* ist die Aktivierung des gesicherten Protokolls auf dem Server transparent, d.h. der Veröffentlichungsmodus wird im Verbindungsdialog nicht angezeigt. Die Verbindung und der Wechsel in den gesicherten Modus erfolgen nach einer internen Absprache zwischen der Server Applikation und der remote Applikation.



Weitere Informationen dazu finden Sie im Abschnitt **Netzwerk und Client-Server Optionen**.

- **Legacy Netzwerkschicht verwenden**

Bei Verwendung der bisherigen Netzwerkschicht wird ein Circumflex (^) vor den Namen der Datenbanken gesetzt, die im TLS Modus veröffentlicht werden. Sie erscheinen auf der Seite TCP/IP des Verbindungsdialogs von 4D Server:



**Hinweis:** Erscheint die Datenbank im Verbindungsdialog nicht automatisch, kann der Benutzer auf der Seite **Angepasst** ihren Namen eingeben (siehe Abschnitte **Anmeldung an eine 4D Server Datenbank** und **IP Einstellungen**). Läuft sie im gesicherten Modus, muss ein Circumflex (^) vor den Datenbanknamen gesetzt werden, sonst wird die Verbindung zurückgewiesen.

## Single Sign On (SSO) unter Windows

4D Server ermöglicht, Ihre Client-Server Lösungen unter Windows mit SSO (*Single Sign On* - Einmalanmeldung) auszurüsten.

Ist SSO in Ihre 4D Lösung integriert, können 4D Benutzer auf die 4D Anwendung unter Windows zugreifen und müssen ihr Kennwort nicht erneut eingeben, wenn sie bereits im Windows Domain ihres Unternehmens angemeldet sind (über Active Directory). Hinter den Kulissen kann die 4D Server Applikation auf das Login zur Authentifizierung der Windows Sitzung zugreifen, das Sie über Ihre standardmäßige Login-Methode zum Einloggen der 4D Benutzer in die Anwendung verwenden können.

**Hinweis:** Einen Überblick über die Authentifizierung und andere Sicherheitsvorkehrungen in 4D finden Sie im [4D Security\\_guide](#).

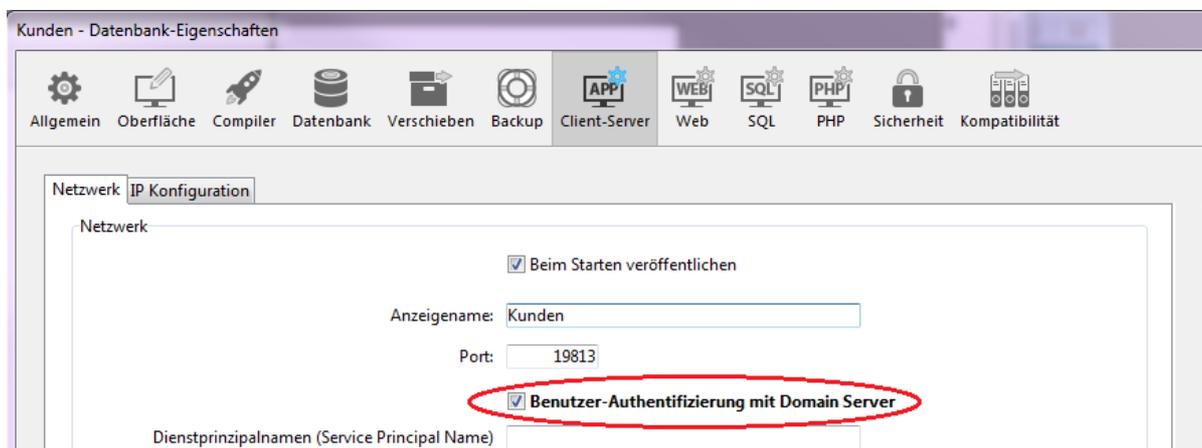
## Anforderungen

Das SSO Feature ist verfügbar:

- Mit 4D Server Anwendungen unter Windows (4D Einzelplatzanwendungen unterstützen nicht SSO)
- Mit der Netzwerkschicht **ServerNet**. Weitere Informationen dazu finden Sie im Abschnitt **Neue Netzwerk-Schicht ServerNet (Kompatibilität)**.

## SSO Features aktivieren

SSO Features sind in 4D Server standardmäßig nicht aktiviert. Um diese Funktionalität zu nutzen, müssen Sie in den Datenbank-Eigenschaften von 4D Server auf der Seite **Client-Server/Netzwerk** die Option **Benutzer-Authentifizierung mit Domain Server** setzen:



Markieren Sie diese Option, verbindet sich 4D direkt mit der Active Directory Zone des Windows Domain Server und erhält die verfügbaren Tokens der Authentifizierung.

Diese Option bietet Standard Authentifizierung über das NTLM Protokoll. 4D unterstützt NTLM und Kerberos Protokolle. Es wird gemäß der aktuellen Konfiguration von 4D automatisch ausgewählt (siehe **Anforderungen für SSO**). Um das Kerberos Protokoll zu verwenden, müssen Sie zusätzlich den Bereich Dienstprinzipalname ausfüllen (siehe unten).

## Kerberos aktivieren

Wollen Sie Kerberos als Authentifizierungsprotokoll nutzen, müssen Sie auf der Seite **Client-Server/Netzwerk** der Datenbank-Eigenschaften auch die Option **Dienstprinzipalnamen** ausfüllen:

Benutzer-Authentifizierung mit Domain Server

Dienstprinzipalnamen (Service Principal Name)

Diese Option deklariert SPN als Set in der Active Directory Konfiguration. Ein Dienstprinzipalname ist eine einmalige Kennung der Service Instanz. Die Authentifizierung über Kerberos nutzt SPNs, um einem Service Logon Account eine Service Instanz zuzuweisen. So kann eine Client Anwendung den Service auffordern, ein Account zu authentifizieren, auch wenn der Client ohne Account Name ist. Weitere Informationen dazu finden Sie [auf der msdn Web Site unter SPN](#).

Der SPN Identifier muss folgendermaßen aufgebaut sein:

- "ServiceName/FQDN\_user", wenn SPN ein Computer-Attribut ist
- "ServiceName/FQDN\_computer", wenn SPN ein Benutzer-Attribut ist

Definitionen:

- *ServiceName* ist der Name des Service, für den der Client authentifiziert werden möchte.
- Der "Fully Qualified Domain Name" (FQDN) ist ein Domain Name, der die exakte Position in der Baumhierarchie der Active Directory Zone für Computer und Benutzer angibt.

In 4D Anwendungen wird SPN an folgender Stelle eingerichtet:

- in den Struktureinstellungen der Datenbank zur Verwendung mit 4D Server
- oder in den Benutzereinstellungen für Deployment Zwecke (Datei *settings.4DSettings* im Ordner Preferences der Datenbank)

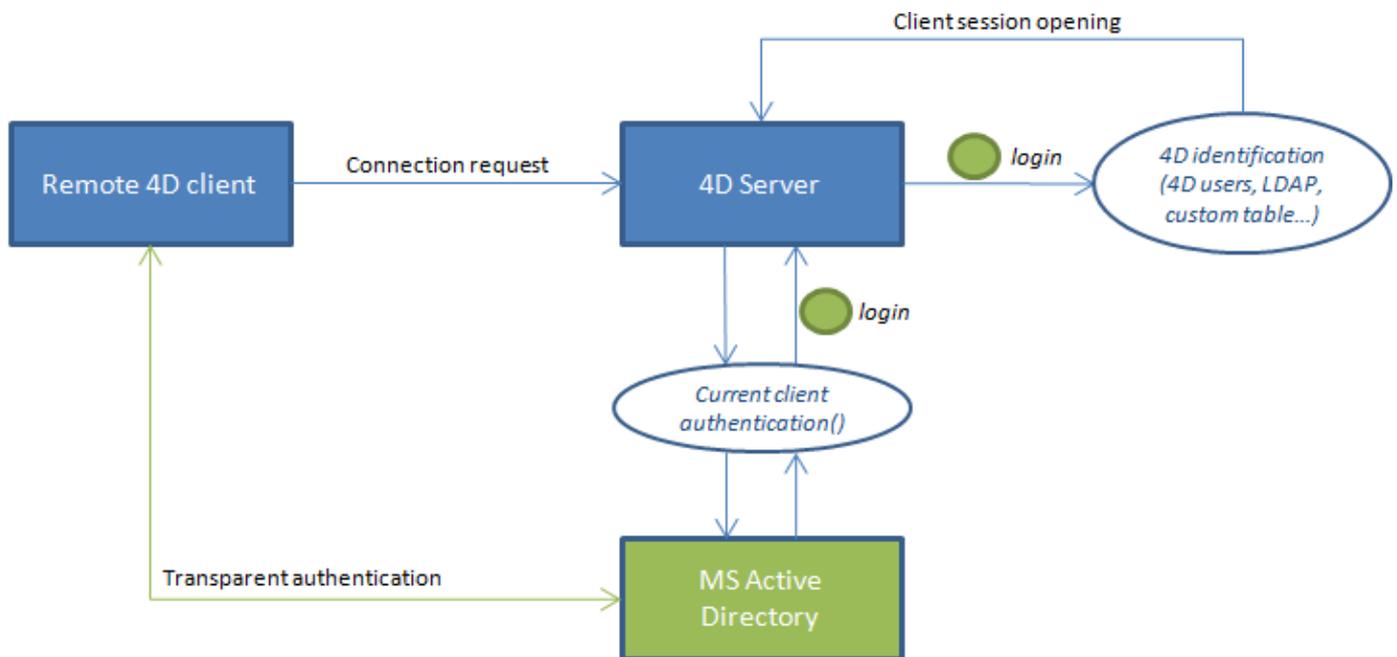
## SSO einfügen

---

Sind SSO Features aktiviert (siehe oben), ist die Benutzer-Authentifizierung, die auf den Anmeldedaten zur Windows Sitzung basiert, zum Öffnen einer Sitzung auf 4D Server zuverlässig.

Beachten Sie, dass das SSO Feature nur das authentifizierte Login liefert, Sie müssen es in Ihre Standard 4D Login-Methode übernehmen. Versucht eine 4D remote Anwendung, sich an den Server anzumelden, müssen Sie die 4D Funktion **Current client authentication** aufrufen, die das Benutzer-Login gemäß der Definition in der Active Directory Zone zurückgibt. Dieses Login können Sie dann in Ihr eigenes Identifikationssystem übernehmen (über die eingebauten Benutzer und Gruppen, LDAP Befehle oder einen eigenen Mechanismus), um in Ihrer 4D Anwendung die passende Sitzung für den remote Benutzer zu öffnen.

Folgende Übersicht zeigt die Vorgehensweise:



Die Funktion **Current client authentication** muss in der **Datenbankmethode On Server Open Connection** aufgerufen werden. Sie wird jedes Mal aufgerufen, wenn ein remote 4D eine neue Verbindung zur 4D Server Datenbank öffnet. Schlägt die Authentifizierung fehl, können Sie in \$0 einen nicht-Nullwert zurückgeben, um die Anmeldung abzuweisen.

## Die Funktion Current client authentication verwenden

Die Funktion **Current client authentication** rufen Sie mit folgender Syntax auf:

```
login:=Current client authentication(Domain;Protokoll)
```

- *login* ist die vom Client verwendete ID zum Einloggen in die Active Directory Zone (Wert vom Typ Text). Diesen Wert müssen Sie verwenden, um den Benutzer in Ihrer Anwendung zu identifizieren. Wurde der Benutzer nicht korrekt authentifiziert, wird ein leerer String zurückgegeben und es wird kein Fehler angezeigt.
- *Domain* und *Protokoll* sind optionale Textparameter. Sie werden von der Funktion gefüllt und ermöglichen, Verbindungen anhand dieser Werte zuzulassen oder abzuweisen:
  - *Domain* ist der Domain Name der Active Directory Zone
  - *Protokoll* ist der Name des Protokolls, das Windows zum Authentifizieren des Benutzers verwendet.

Weitere Informationen dazu finden Sie unter der Funktion **Current client authentication**.

## Anforderungen für SSO

4D Server verwaltet verschiedene SSO Konfigurationen, die sich nach der aktuellen Architektur und Einstellungen richten. Das Protokoll zur Authentifizierung (NTLM oder Kerberos) sowie die von der Funktion **Current client authentication** zurückgegebene Information richten sich nach der aktuellen Konfiguration, wenn alle Anforderungen erfüllt sind. Der Parameter *Protokoll* von **Current client authentication** gibt das aktuell verwendete Protokoll für die Authentifizierung zurück.

Nachfolgende Übersicht zeigt die verschiedenen Situationen:

	<b>NTLM</b>	<b>Kerberos</b>
4D Server und 4D remote auf verschiedenen Rechnern	ja	ja
4D Server Benutzer ist in Domain	ja	ja
4D remote in der gleichen AD Zone wie 4D Server Benutzer	ja oder nein(*)	ja
SPN in 4D Server eingetragen	nein	ja(**)
<b>von Current client authentication zurückgegebene Information, wenn die Anforderungen berücksichtigt werden</b>	<i>user=erwartetes Login, domain=erwartetes Domain, protocol="NTLM"</i>	<i>user=erwartetes Login, domain=erwartetes Domain, protocol="Kerberos"</i>

(\*) Folgende spezifische Konfiguration wird unterstützt: der 4D remote Benutzer ist ein lokaler Account auf dem Rechner, der zur selben AD Zone wie 4D Server gehört. In diesem Fall wird im Parameter *Domain* der Name des Rechners mit 4D Server eingesetzt. Beachten Sie, dass die Anzeige von den aktuellen Benutzereinstellungen abhängt: Ist keine verfügbar, werden leere Strings zurückgegeben.

(\*\*) Wurden alle Kerberos Anforderungen berücksichtigt, die Funktion **Current client authentication** gibt jedoch in *Protokoll* "NTLM" zurück, hat das folgende Gründe:

- Die SPN Syntax ist nicht gültig, d.h. sie berücksichtigt nicht die [von Microsoft vorgeschriebenen Einschränkungen](#).
- oder SPN hat Duplikate in der AD Zone. Dieses Problem kann nur der Administrator von Active Directory beheben.

**Hinweis:** Eine gültige Syntax bedeutet nicht, dass die SPN Deklaration ansich korrekt ist; existiert SPN nicht in der AD Zone, gibt **Current client authentication** leere Strings zurück.

## Ordner Resources verwalten

---

Der Ordner **Resources** der Datenbank ermöglicht, eigene Daten auf dem Server-Rechner und allen Client-Rechnern gemeinsam zu nutzen, z.B. Bilder, Dateien, Unterordner. Auf der Server-Seite muss der Ordner **Resources** nur neben der Strukturdatei der Datenbank liegen.

Alle diesem Ordner zugewiesenen Referenzmechanismen werden im Client/Server Modus unterstützt (Ordner .lproj, XLIFF, Bilder, etc.). Weitere Informationen dazu finden Sie im Handbuch *4D Designmodus*.

Jeder Client hat eine lokale Kopie dieses Ordners. Der Inhalt wird, immer wenn sich der Client anmeldet, mit dem Ordner auf dem Server automatisch abgeglichen.

Darüberhinaus können Client-Rechner während einer Arbeitssitzung automatisch eine Meldung erhalten, wenn ein Entwickler den Inhalt des Ordners **Resources** in der Server Datenbank verändert. Die Meldung kann folgendermaßen erfolgen:

- Automatisch vom Server, zwei Minuten nach der letzten Änderung durch einen Client. Dieser Zeitversatz verhindert unnötige Meldungen, wenn z.B. viele Dateien kopiert werden.
- Manuell über den Befehl **Clients benachrichtigen** im Menü **Aktion** des Ressourcen Explorers auf dem Client-Rechner, von dem die Änderung ausgeht.
- Per Programmierung über den Befehl **NOTIFY RESOURCES FOLDER MODIFICATION**. Dieser Befehl ist hilfreich, wenn der Inhalt des Ordners Resources auf dem Server-Rechner über eine Serverprozedur geändert wird.

Auf der Client-Seite richtet sich die Ausgabe der Meldung nach der Option "'Resources-Ordner' während einer Arbeitssitzung aktualisieren", die in den Einstellungen der Datenbank definiert wurde. Dies lässt sich auch individuell über den Befehl **SET DATABASE PARAMETER** einrichten. Es gibt drei Möglichkeiten: Nie, immer oder Fragen. Weitere Informationen dazu finden Sie im Abschnitt **Netzwerk und Client-Server Optionen** und in der Beschreibung zum Befehl **SET DATABASE PARAMETER**. Über den Befehl **Lokale Ressourcen aktualisieren** im Menü **Aktion** des Ressourcen Explorer kann jeder Client den Abgleich mit dem Server auch selbst ausführen. Weitere Informationen dazu finden Sie im Handbuch *4D Designmodus* im Kapitel **Ressourcen Explorer**.

**Hinweis zur Kompatibilität:** In bisherigen Versionen von 4D Server erfolgte die Übertragung eigener Daten über den Ordner **Extras**, der neben der Strukturdatei lag. Dieser Ordner ist veraltet und sollte nicht mehr verwendet werden. Zur Wahrung der Kompatibilität mit vorhandenen Anwendungen unterstützt 4D Server diesen Ordner jedoch weiterhin.

## Datenbank als Dienst registrieren

---

Sie können 4D Server als Dienst unter Windows starten.

**Hinweis zur Kompatibilität:** Diese Funktion ist ab Version 12 von 4D Server auf Mac OS nicht mehr verfügbar.

Eine 4D Server Anwendung, die als Dienst registriert ist, wird beim Starten des Rechners automatisch mit der aktuellen Datenbank hochgefahren, und zwar noch vor Öffnen einer Benutzersitzung. Die Anwendung wird nicht geschlossen, wenn der Benutzer die Sitzung beendet.

Diese Vorgehensweise sorgt dafür, dass die 4D Server Datenbank auch bei einem Zwischenfall verfügbar bleibt, der den Neustart des Rechners erfordert. Die Wartung kann aus der Ferne erfolgen.

### **Hinweise:**

- Auf einer Windows Plattform mit 64 Bit wird eine 4D Server Anwendung, die als Dienst registriert ist, ohne Oberfläche ausgeführt, d.h. das Server Verwaltungsfenster erscheint nicht.
- Weitere Informationen zum Steuern von Diensten finden Sie in der Dokumentation Ihres Betriebssystems.

Um die 4D Server Datenbank als Service zu registrieren, wählen Sie im 4D Server Menü **Datei/Ablage** den Befehl **Als Service registrieren**. Beim nächsten Hochfahren des Rechners wird 4D Server automatisch gestartet und die aktuelle Datenbank geöffnet. Sie können beliebig viele Datenbanken registrieren, jedoch immer nur eine Instanz pro Datenbank.

**Hinweis:** Unter Windows kann dieser Befehl in Grauschrift sein, wenn der Zugriff auf die Funktionen zur Service-Verwaltung eingeschränkt ist. Damit Sie diesen Befehl verwenden können, müssen Sie 4D Server auf Administrator-Ebene starten. Klicken Sie dazu mit rechter Maustaste auf das Programm-Icon und wählen im Kontextmenü den Befehl **Als Administrator ausführen**.

**Wichtig:** Stellen Sie sicher, dass Sie beim Öffnen einer Sitzung ein gültiges Benutzerkonto verwenden, andernfalls erscheint eine Fehlermeldung. 4D Server wird standardmäßig im lokalen Systemkonto ausgeführt, das u.U. nicht die erforderlichen Einstellungen zum Benutzen Ihrer Anwendung hat. Insbesondere zum Drucken müssen Sie die Sitzung mit einem Benutzerkonto mit den standardmäßigen Druckeinstellungen öffnen. Dieses Vorgehen ist ähnlich, wenn Sie auf Netzwerk Volumes zugreifen wollen. Dazu empfehlen wir, das Konto zu wechseln: Gehen Sie zu **Systemsteuerung > System und Sicherheit > Administrative Tools > Dienste**. Klicken Sie in der Liste Dienste mit der rechten Maustaste auf 4D Server, wählen die Option **Eigenschaften**, gehen zur Registerkarte **Anmelden** und geben das Konto an, unter dem der Server laufen soll. Die Einstellung wird mit dem nächsten Start aktiviert.

Um die Registrierung einer Datenbank aufzuheben, wählen Sie im Menü **Datei/Ablage** von 4D Server den Befehl **Registrierung für aktuelle DB aufheben**. Dieser Befehl erscheint in Grauschrift, wenn die Datenbank nicht als Dienst registriert ist.

Um die Registrierung aller Datenbanken aufzuheben, wählen Sie im Menü **Datei/Ablage** von 4D Server den Befehl **Alle Server Registrierungen aufheben**. Dieser Befehl erscheint in Grauschrift, wenn kein 4D Server Dienst aktiviert ist. Startet die Anwendung beim Öffnen als Dienst, können Sie diesen Status nicht in 4D Server ändern; die drei Menübefehle sind inaktiv. Um den Dienst zu stoppen, rufen Sie über das Verwaltungsprogramm Dienste in Ihrer Systemsteuerung auf.

## Logischen Spiegel einrichten

---

4D Server enthält eine integrierte Lösung, um ein Backup-System durch logisches Spiegeln einzurichten. Dafür gibt es die beiden Routinen **New log file** und **INTEGRATE MIRROR LOG FILE**.

### Was ist ein logischer Spiegel?

---

Ein logischer Spiegel ist ein ausgeklügeltes Backup-System, das besonders für kritische Datenbanken oder solche im 24-Stunden Betrieb sinnvoll ist.

Bei einem logisches Spiegel läuft die Datenbank auf einem Rechner. Eine Kopie dieser Datenbank wird auf einem 2. Rechner gehalten und in bestimmten Zeitabständen aktualisiert. Beide Rechner kommunizieren über das Netzwerk miteinander. Der Rechner im laufenden Betrieb überträgt über das Logbuch alle Änderungen in der Datenbank an den Spiegelrechner.

So kann bei einem Zwischenfall die Spiegeldatenbank zum raschen Wiederherstellen von Daten und Operationen dienen, ohne dass Daten verloren gehen. Außerdem wird die laufende Datenbank nie durch Backup-Operationen blockiert.

### Warum Backup durch logisches Spiegeln?

---

Die Verwendung eines logischen Spiegels ist nur für spezifische Situationen sinnvoll. In den meisten Fällen reicht die standardmäßige Strategie aus, d.h. die periodische Sicherung und das Mitlaufen eines Logbuchs. Das ist eine einfache, zuverlässige und kostengünstige Lösung. Die Datenbank wird regelmäßig gesichert, im allgemeinen alle 24 Stunden. Während dem Backup sind alle Prozesse eingefroren. Diese Einschränkung ist jedoch nur von kurzer Dauer. Sie dauert selbst bei umfangreichen Datenbanken mit mehr als 2 GB gerade mal 5 Minuten. Das Backup lässt sich auch so programmieren, dass es außerhalb der normalen Betriebszeiten der Datenbank durchgeführt wird.

Für bestimmte Bereiche, z.B. Krankenhäuser, ist als Sicherungsmethode jedoch ein logischer Spiegel angebracht. Denn hier müssen kritische Datenbanken im 24-Stunden Betrieb laufen. Sie können nicht im Backup-Prozess sein, nicht einmal für eine kurze Zeitspanne.

**Hinweis:** Die Spiegeldatenbank gibt nur Änderungen an den Daten wieder. Sie ist für Datenbanken in der Entwicklungsphase nicht geeignet. Der Spiegel wäre aufgrund häufiger Änderungen an der Struktur schnell überholt. Die Struktur der Spiegeldatenbank müsste laufend aktualisiert werden.

### Funktionsweise

---

Das Sicherungssystem für logisches Spiegeln wird mit den beiden Routinen **New log file** und **INTEGRATE MIRROR LOG FILE** verwaltet. Weitere Informationen dazu finden Sie im Handbuch *4D Programmiersprache*.

Es funktioniert folgendermaßen:

- Die Datenbank für den laufenden Betrieb wird auf dem 4D Server Hauptrechner eingerichtet, ein Duplikat der Datenbank wird auf dem Spiegelrechner installiert.
- Über einen Test beim Starten des Programms, z.B. Suchen nach einer bestimmten Datei in einem Unterordner des 4D Server Programms, wird zwischen den beiden Versionen unterschieden (in Betrieb und Spiegel), um die jeweils passenden Operationen auszuführen.
- Auf dem 4D Server Rechner in Betrieb wird das Logbuch über die Funktion **New log file** in regelmäßigen Abständen "segmentiert". Da auf dem Hauptrechner kein Backup ausgeführt wird, bleibt die Datenbank ununterbrochen im Lesen-Schreiben Modus verfügbar.
- Jedes "Segment" des Logbuchs wird an den Spiegelserver gesendet und über den Befehl **INTEGRATE MIRROR LOG FILE** in die Spiegeldatenbank integriert.

Für dieses System ist spezifischer Programmiercode erforderlich. Folgendes wird benötigt:

- Ein Timer auf dem Hauptserver, der die Ausführungsphasen der Funktion **New log file** verwaltet,
- Ein Übertragungssystem für die "Segmente" des Logbuchs zwischen dem laufenden Rechner und dem Spiegelrechner. Das können 4D Internet Befehle für die Übertragung via FTP oder E-Mail, Web-Services, o.ä. sein.
- Ein Prozess auf dem Spiegelserver, der das Eintreffen neuer Segmente des Logbuchs überwacht und diese über den Befehl **INTEGRATE MIRROR LOG FILE** integriert,
- Ein System zur Kommunikation und Fehlerverwaltung zwischen dem Hauptserver und dem Spiegelserver.

**Warnung:** Das Backup durch logisches Spiegeln ist nicht mit den anderen Backup-Arten kompatibel, da die gleichzeitige Verwendung von zwei Systemen zu einer Desynchronisierung der laufenden und der Spiegeldatenbank führen würde. Sie müssen also sicherstellen, dass auf dem Hauptrechner keine Backups ausgeführt werden, sei es automatisch oder manuell. Es ist jedoch möglich, ein Backup von der Spiegeldatenbank oder ein Spiegel vom Spiegel zu erstellen (siehe nächster Absatz).

## **Backup einer Spiegeldatenbank**

---

Über 4D Server lassen sich Backups der Datenbank auf dem Spiegelrechner ausführen.

Auf dem Spiegelrechner sind alle gängigen Verfahren möglich: manuelle Backups über den Befehl im Menü **Datei/Ablage**, geplante Backups in den Einstellungen der Datenbank oder programmierte Backups über Befehle der 4D Programmiersprache.

Um eine mögliche Desynchronisation mit dem Hauptrechner zu vermeiden, sperrt 4D automatisch den Spiegelrechner bei einer der beiden Grundoperationen: Integration des Logbuchs aus dem Hauptrechner und Backup der Spiegeldatenbank.

- Wird gerade ein Logbuch integriert, lässt sich kein Backup ausführen. Wird der Befehl **BACKUP** verwendet, wird der Fehler 1417 erzeugt (siehe *Handbuch 4D Programmiersprache* im Abschnitt **Backup-Verwaltung Systemfehler (1401 -> 1421)**).
- Läuft gerade ein Backup, werden alle Prozesse eingefroren und es ist unmöglich, die Integration eines Logbuchs zu starten.

Ab 4D v14 können Sie auch das aktuelle Logbuch auf dem Spiegelrechner aktivieren, d.h. Sie können den Spiegel eines Spiegels, eine Reihe von Spiegeln oder auch eine "hub-and-spoke" Spiegelarchitektur, d.h. mehrere Spiegel für die gleiche Datenbank im laufenden Betrieb einrichten. Im ersten Fall wird das aktuelle Logbuch des Spiegels im Turnus an einen anderen Spiegel (Spiegel vom Spiegel) zur Integration gesendet und, falls vorhanden, fortlaufend zu den weiteren Spiegeln der Reihe. Im zweiten Fall wird das aktuelle Logbuch direkt an mehrere identische Spiegelserver gesendet. Diese Art Redundanz stellt die kontinuierliche Verfügbarkeit des Servers sicher, selbst wenn der Server und der Hauptspiegel gleichzeitig ausfallen sollten.

## **Operationschema für logisches Spiegeln**

---

Nachfolgende Übersicht zeigt den Ablauf beim Einrichten des Backup-Systems logisches Spiegeln für beide 4D Server Rechner:

## Schritt Produktionsrechner

## Spiegelrechner

**1** Anwendung starten,  
Backup der Datendatei.  
Das Logbuch wird  
standardmäßig aktiviert.  
Es sollte für bessere  
Sicherheit auf einer  
separaten Festplatte  
gespeichert werden. 4D  
erstellt die Datei  
MeineDatenbank.journal.

Anwendung beenden.  
Alle Dateien der  
Datenbank inkl. Logbuch  
auf den Spiegelrechner  
kopieren.

**2** Anwendung neu starten  
und Operation beginnen  
(prüfen Sie, dass kein  
Vollbackup per  
Programmiersprache oder  
Planer durchgeführt wird).

**3** Der Spiegelserver soll  
aktualisiert werden, z.B.  
nach einer bestimmten  
Zeitspanne.

Ausführung der Methode,  
welche die Funktion **New  
log file** enthält. Die  
gesicherte Datei lautet  
MeineDatenbank[0001-  
0001].journal

Die Datei  
MeineDatenbank[0001-  
0001].journal per  
Programmierung an den  
Spiegelserver senden  
(über 4DIC, Web  
Services, etc.)

Server ist betriebsbereit.

**4** Eine zu integrierende Datei wird erhalten.  
Ausführung der Methode, die den Befehl  
**INTEGRATE MIRROR LOG FILE** enthält,  
um die Datei MeineDatenbank[0001-  
0001].journal zu integrieren. Bei einem  
Spiegel-Spiegel System läuft die  
Ausführung einer Operation auf dem  
Spiegelrechner wie in Schritt 3 ab. Sie  
muss für jedes Integrieren eines Logbuchs  
wiederholt werden.

**5** Auf dem Rechner passiert  
ein Zwischenfall, die

Datendatei ist nicht mehr verwendbar.  
Administrator entscheidet:  
Der Spiegelrechner soll als Ersatzrechner verwendet werden.

Kopieren Sie das aktuelle Logbuch  
MeineDatenbank.journal  
manuell auf den Zielordner des Spiegelrechners.

**6**

Zwischenfall analysieren und Rechner reparieren.

Eine Datei, die auf Integration wartet, wird gefunden. Ausführung der Methode, die den Befehl **\_o\_INTEGRATE LOG FILE** enthält, um die Datei MeineDatenbank.journal zu integrieren.

Server ist betriebsbereit.

**7**

Der Rechner ist repariert.  
Die beschädigten Dateien der Datenbank mit denen der Spiegeldatenbank ersetzen. Die Anwendung starten: 4D Server fordert das Logbuch an. Wählen Sie die von der Spiegeldatenbank übertragene Datei.

Die Datenbank wird beendet. Zurück zu Schritt 2.

## **Volume Shadow Copy Dienst unter Windows verwenden**

### **Überblick**

---

4D Server Windows 64-bit enthält das spezifische Programm **VSS Writer** zur automatischen Verwaltung von Anfragen für Versionsstände (sog. Snapshots), die über den Windows Volume Shadows Copy Service (VSS) gesendet werden.

VSS wird vom Windows Server bereitgestellt, damit Backup Programme zu bestimmten Zeitpunkten im laufenden Betrieb Versionsstände (Schattenkopien) von beliebigen Dateien oder ganzen Festplatten machen können. Mit Hilfe dieser Technologie können Sie z.B. eine 4D Server Anwendung mit genau dem Status wiederherstellen, wie er im Moment des Snapshots war. Dafür müssen die laufenden Anwendungsdateien zum Zeitpunkt des Snapshots in einem konsistenten Zustand sein. Zu diesem Zweck muss ein VSS-taugliches Programm einen VSS Writer und Requestor enthalten. Zur Herstellung eines konsistenten Zustands werden die "Writer" benötigt. Ein VSS-Writer übernimmt die Aufgabe, die Dateien entsprechend "einzufrieren" und nach der Erstellung des Snapshots wieder in den Normalzustand zu versetzen. Ein "Requestor" ist üblicherweise ein Backup-Client, der den Snapshot anfordert, die Daten sichert und den Snapshot auch wieder entfernt.

**Hinweis:** Einen allgemeinen Überblick über die Sicherheitsvorkehrungen in 4D finden Sie im [4D Security guide](#).

### **Anforderungen für den "Virtualizer"**

---

Die Host-Seite unterstützt folgende VSS Requestoren:

- **VMware ESXI** auf jeder Plattform
- **Microsoft Hyper-V Server 2016**

### **Anforderungen für den "guest" (4D Server)**

---

4D Server unterstützt den Microsoft Windows VSS mit folgender Konfiguration:

- 4D Server (doppelklickbar oder nicht)
- Version 64-bit
- Produktreihe Windows Server Version 2012 R2 oder höher

Hier installiert 4D Server den VSS Service automatisch.

Andere Windows Versionen, die VSS unterstützen, sind kompatibel aber nicht zertifiziert; in diesen Versionen muss der VSS Service beim Einrichten aktiviert werden:

- Windows 7
- Windows 8.1
- Windows 10

### **VSS aktivieren**

---

Das VSS Feature wird beim Starten der 4D Server Applikation automatisch installiert/upgedated. Der VSS Writer Application Service startet unter folgenden Bedingungen:

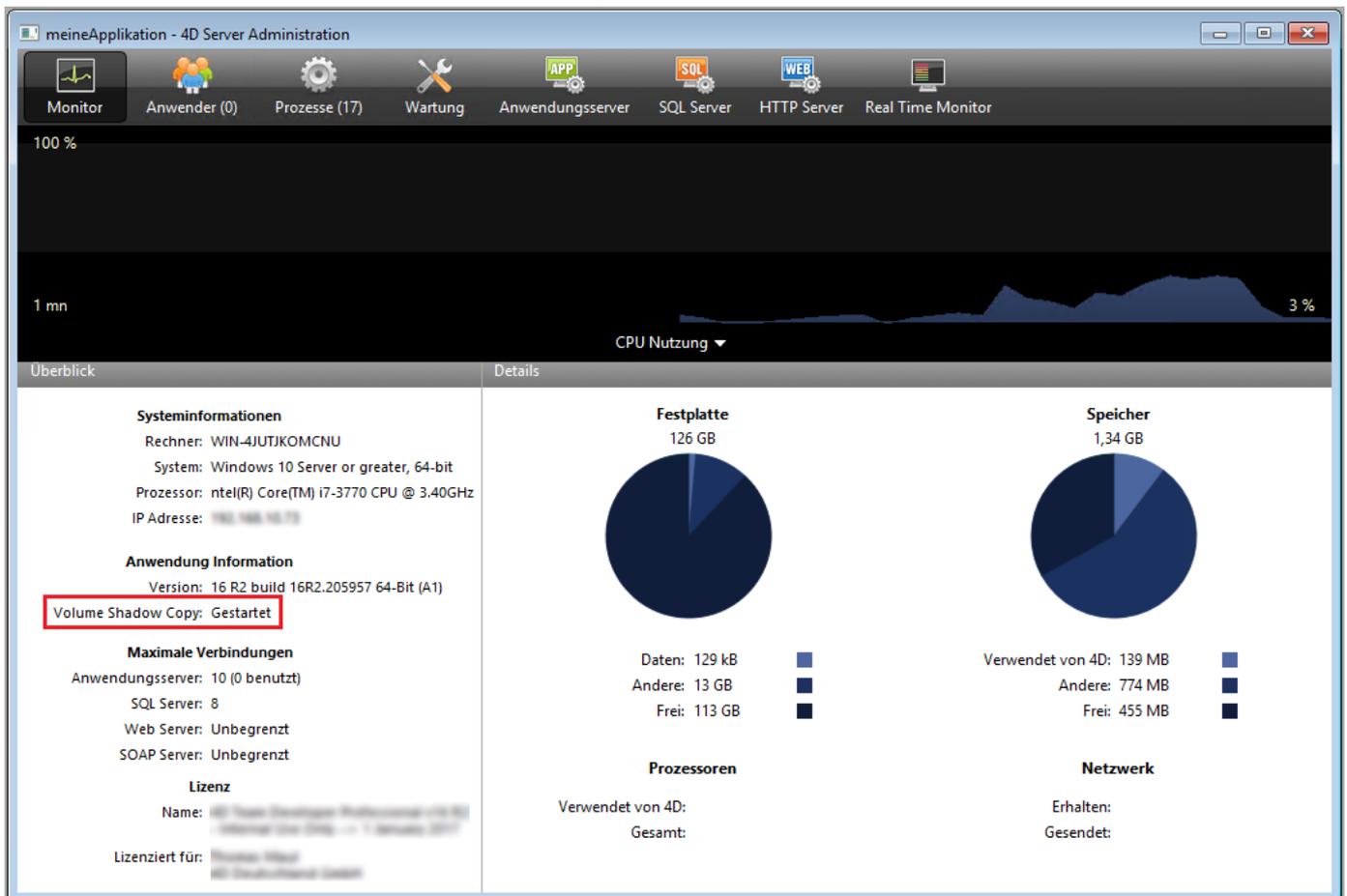
- die 4D Server Konfiguration ist kompatibel (siehe oben)
- Der Benutzer der Sitzung hat Admin-Rechte

In der Regel läuft beim Starten folgendes ab:

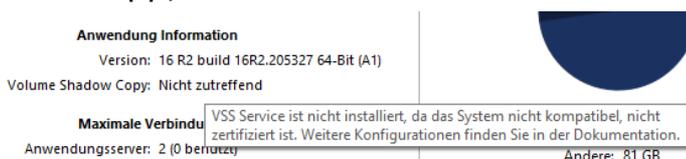
1. 4D Server oder gemeinsam genutzer Application Server wird zum ersten Mal gestartet
2. Beim Starten ohne Admin-Rechte erscheint ein Warn-Icon
3. Beenden und 4D Server oder den gemeinsam genutzen Application Server als Administrator erneut starten  
Der 4D VSS Dienst wird dann automatisch ausgeführt und in VSS registriert
4. (Optional) 4D Server oder den gemeinsam genutzen Application Server erneut mit Standardberechtigung starten

Der ausführbare VSS Writer wird als Service mit dem Namen "VSS <appName>" gestartet. Ein VSS Dienst läuft für alle 4D Server Instanzen und jeweils für jede weitere doppelklickbare Applikation (unterschiedlicher Name), die auf dem Rechner läuft (siehe unten).

Die **Seite Monitor** des 4D Server Verwaltungsfensters zeigt den Status des VSS Writer Dienstes im Bereich Anwendungsinformation:



Zusätzliche Information zum Status des Volume Shadow Copy Dienstes erscheint in einem Tipp, wenn Sie mit der Maus über diesen Bereich ziehen:



## Über VSS Writer

---

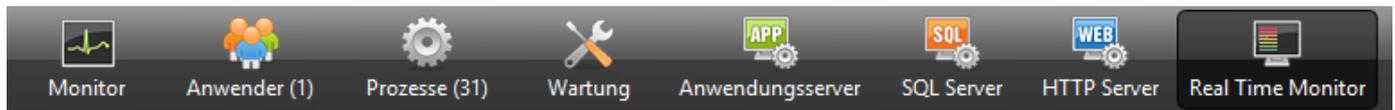
Die Applikation **vss\_writer.exe** dient zum Verwalten des Volumeschattenkopie-Dienstes (VSS) für 4D Anwendungen.

**Hinweis:** Die 4D VSS Verwaltung wird über ein eigenes Programm gesteuert, da es mit Administrator-Rechen laufen muss.

4D Server installiert den ausführbaren 4D VSS Writer automatisch beim ersten Start. Der 4D VSS Writer Dienst verwaltet und überträgt VSS Meldungen an 4D Server. Diese Meldungen werden im Diagnose-Logbuch von 4D Server und in der Ereignisansicht von Windows protokolliert.

# 4D Server Verwaltungsfenster

4D Server hat ein verständliches und benutzerfreundliches Verwaltungsfenster. Es bietet verschiedene Tools zum Analysieren und Steuern für über den Server veröffentlichte Anwendungen. Über die Icons in der Leiste am oberen Rand können Sie direkt die gewünschte Seite öffnen:

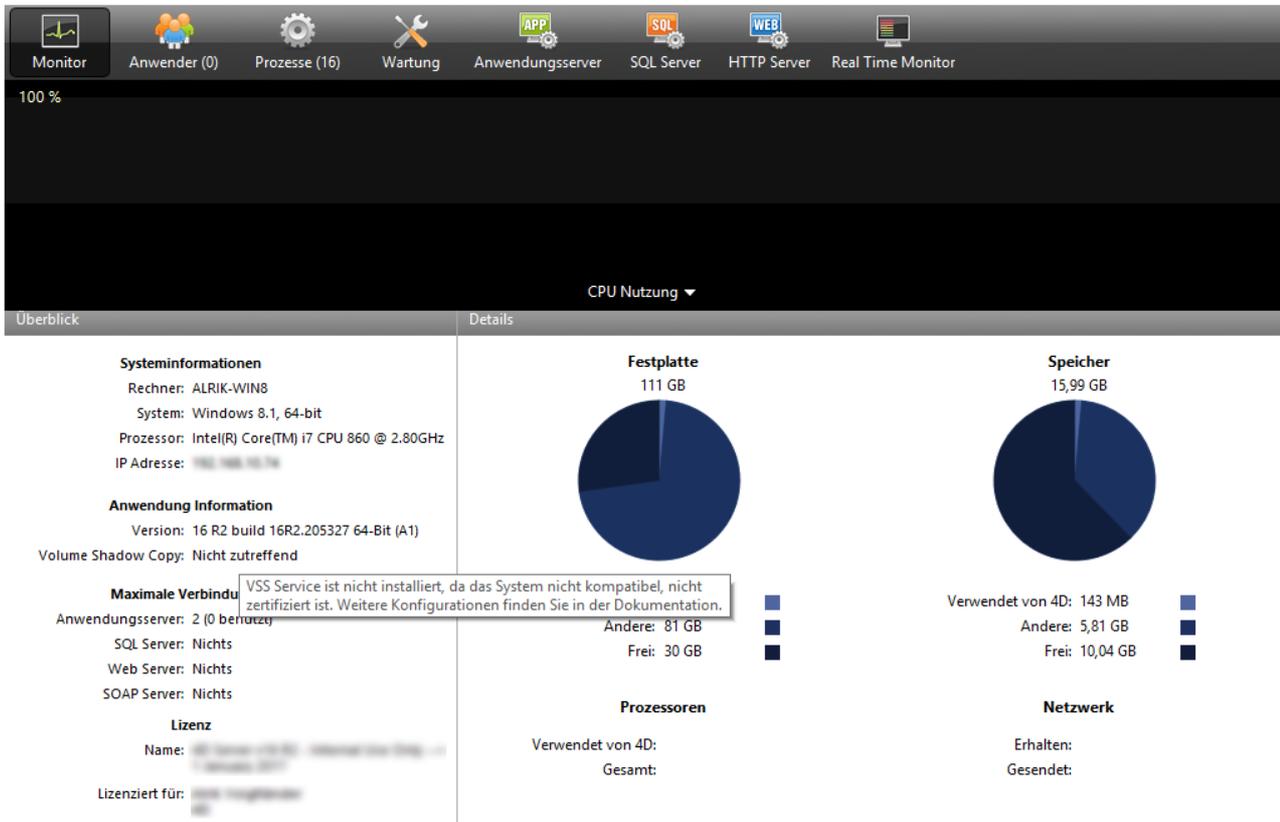


**Hinweis:** Das Verwaltungsfenster ist auch über einen Rechner mit remote 4D zugänglich. Weitere Informationen dazu finden Sie im Abschnitt **Administration über Remote Rechner**.

-  Seite Monitor
-  Seite Anwender
-  Seite Prozesse
-  Seite Wartung
-  Seite Anwendungsserver
-  Seite SQL Server
-  Seite HTTP Server
-  Seite Real Time Monitor

## Seite Monitor

Die Seite Monitor zeigt dynamische Angaben zur verwendeten Datenbank, sowie Informationen zum System und der 4D Server Anwendung:



**Hinweis:** Unter Windows richtet sich die Anzeige nach den Zugriffsrechten des Benutzers, der die Arbeitssitzung geöffnet hat. Weitere Informationen dazu finden Sie im unteren Abschnitt **Anzeige der Monitor Information (Windows)**.

## Grafische Darstellung

Hier sehen Sie in Echtzeit das Ablufen der Parameter CPU Nutzung, Netzwerkverkehr und Speicher. Über das DropDown-Menü in der Mitte des Fensters können Sie den jeweiligen Parameter auswählen:



- **CPU Nutzung:** Hier sehen Sie in Echtzeit das Ablufen der Parameter CPU Nutzung, Netzwerkverkehr und Speicher. Über das DropDown-Menü in der Mitte

des Fensters können Sie den jeweiligen Parameter auswählen.

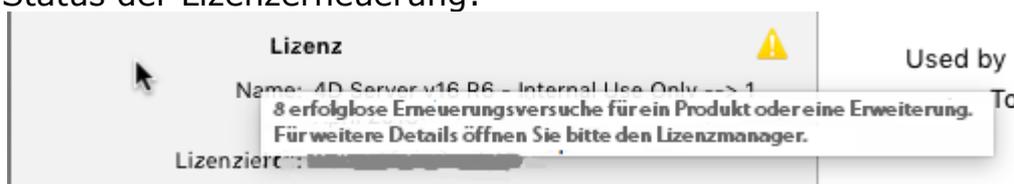
- **Netzwerk:** Anzahl Bytes, die pro Sekunde von 4D Server empfangen werden. Die von 4D Server gesendete Anzahl Bytes erscheint im Bereich "Netzwerk".
- **Speicher:** Anteil des RAM Speichers, den 4D Server auf dem Rechner nutzt. Die genaue Übersicht des verwendeten Speichers erscheint im Bereich "Speicher".

## Überblick

Dieser Bereich enthält Informationen zum System, zur Version und zu den Lizenzen, die auf dem 4D Server Rechner installiert sind.



- **Systeminformationen:** Computer, System and IP Adresse des Server
- **Anwendung Information:** Interne Versionsnummer von 4D Server und Volume Shadow Copy Status (siehe Abschnitt **Volume Shadow Copy Dienst unter Windows verwenden**).
- **Maximale Verbindungen:** Anzahl gleichzeitig erlaubter Verbindungen für Servertyp
- **Lizenz:** Angaben zur Lizenz  
Endet die Produktlizenz oder eine ihrer dazugehörigen Expansions in weniger als 10 Tagen, z.B. bei einer Subscription-Lizenz, versucht 4D Server die Lizenz automatisch über das 4D Benutzerkonto zu erneuern. Schlägt die automatische Erneuerung aus irgendeinem Grund fehl (Verbindungsfehler, ungültiger Account-Status, nicht-verlängerter Vertrag), erscheint neben der Lizenz ein Warn-Icon, um den Server-Administrator darauf aufmerksam zu machen. Ziehen Sie mit der Maus über den Bereich, erscheinen zusätzliche Angaben zum Status der Lizenzerneuerung:

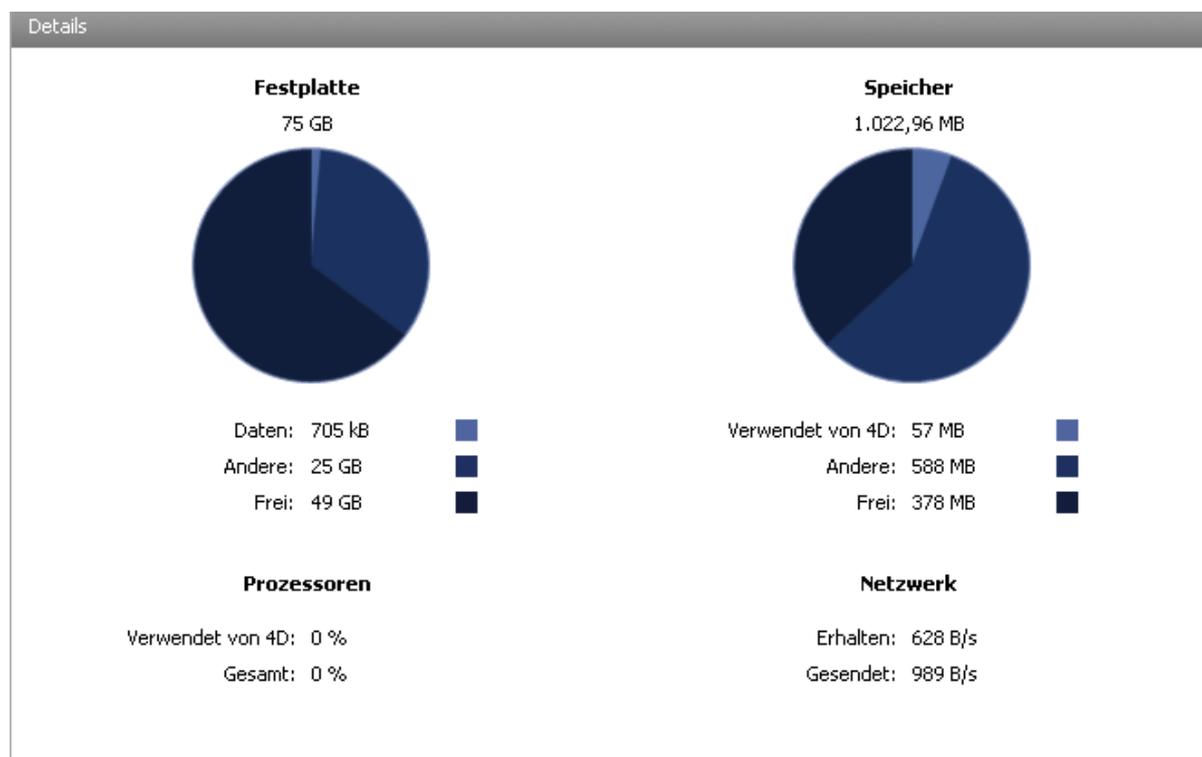


In der Regel müssen Sie zum Überprüfen den Lizenz-Manager öffnen. Weitere Informationen dazu erhalten Sie im Abschnitt .

## Details

---

Der Bereich "Details" wiederholt einige Angaben aus dem grafischen Teil und zeigt zusätzliche Informationen.



- **Festplatte:** Gesamte Kapazität der Festplatte und Anteile, die die Daten der Datenbank (Datendatei + Datenindex) und andere Dateien benutzen, sowie verfügbarer freier Speicherplatz.
- **Speicher:** RAM Speicher des Rechners und Anteile, die 4D Server bzw. andere Programme nutzen, sowie freier Platz.  
Der von 4D Server genutzte Speicherplatz lässt sich im grafischen Bereich auch dynamisch anzeigen.
- **Prozessoren:** Anteile, die 4D Server und andere Programme momentan für Prozessoren auf dem Rechner belegen. Diese Rate wird laufend neu berechnet. Die von 4D Server genutzte Auslastung lässt sich im grafischen Bereich auch dynamisch anzeigen.
- **Netzwerk:** Anzahl Bytes, die 4D Server momentan vom Netzwerk erhält und Anzahl Bytes, die das Programm versendet. Dieser Wert wird laufend aktualisiert.  
Die von 4D Server empfangenen Bytes lassen sich im grafischen Bereich auch dynamisch anzeigen.

## Anzeige der Monitor Information (Windows)

---

Einige Systeminformationen von der Seite Monitor lassen sich auch über das Werkzeug "Leistung" der Windows-Verwaltung anzeigen. Es steht jedoch nur zur Verfügung, wenn der Benutzer, der die Arbeitssitzung auf 4D Server geöffnet hat, die entsprechende Berechtigung hat. Er muss:

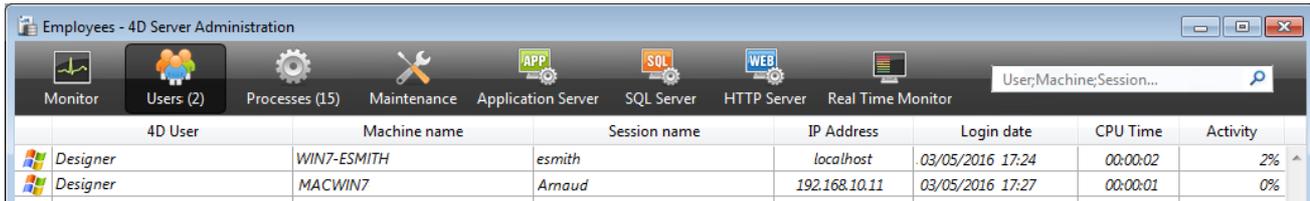
- entweder zur Gruppe "Administratoren" gehören,
- unter Windows Vista zur Gruppe "Hauptbenutzer" gehören (für einen Benutzer, der nicht Administrator ist)

Um einen Benutzer, der nicht Administrator ist, unter Windows Vista in die Gruppe "Hauptbenutzer" aufzunehmen:

1. Gehen Sie in den Bereich Systemsteuerung und öffnen die Option "Benutzerkonten".
2. Klicken Sie auf die Registerkarte "Erweitert", dann im Bereich "Erweiterte Benutzerverwaltung" auf die Schaltfläche "Erweitert".  
Die Option "Lokale Benutzer und Gruppen" wird aktiviert.
3. Doppelklicken Sie auf den Ordner "Gruppen".
4. Doppelklicken Sie auf den Ordner "Hauptbenutzer".  
Auf dem Bildschirm erscheint das Fenster "Eigenschaften von Hauptbenutzern"
5. Klicken Sie auf die Schaltfläche **Hinzufügen**, um einen Benutzer hinzuzufügen.
6. Geben Sie im Textbereich "Geben Sie die zu verwendenden Objektnamen ein" die Benutzernamen ein, die Sie autorisieren möchten.
7. Klicken Sie in den Dialogfenstern jeweils auf die Schaltfläche **OK** (zweimal).  
Schließen Sie das Dialogfenster "Lokale Benutzer und Gruppen", dann das Fenster "Benutzerkonten".

## Seite Anwender

Die Seite Anwender zeigt die Liste der 4D Benutzer, die an die Datenbank angemeldet sind:



4D User	Machine name	Session name	IP Address	Login date	CPU Time	Activity
Designer	WIN7-ESMITH	esmith	localhost	03/05/2016 17:24	00:00:02	2%
Designer	MACWIN7	Arnaud	192.168.10.11	03/05/2016 17:27	00:00:01	0%

Die Schaltfläche "Anwender" zeigt in Klammern die Gesamtanzahl der Benutzer, die an die Datenbank angemeldet sind. Diese Nummer berücksichtigt keine Anzeigefilter, die auf das Fenster angewendet werden.



Diese Seite enthält auch einen dynamischen Suchbereich und Schaltflächen zum Steuern.

Sie können die Anordnung der Spalten durch Markieren und Ziehen des Titelbereichs verändern. Sie können auch die Werte in den Spalten durch Klicken in den Titelbereich sortieren:



Klicken Sie mehrmals, um aufsteigend/absteigend zu sortieren

## Liste der Anwender

Für jeden angemeldeten Benutzer erscheinen folgende Informationen:

- **System:** Typ des Client Rechners (Mac OS oder Windows) als Icon.
- **4D Anwender:** Name der 4D Anwender. Sind keine Kennwörter aktiviert, haben alle Benutzer den Namen "Designer."
- **Rechner:** Name des Client Rechners.
- **Session-Name:** Name der Arbeitssitzung, die auf dem Client Rechner geöffnet wird.
- **IP Adresse:** IP Adresse des Client Rechners.
- **Anmeldedatum:** Anmeldedatum und -uhrzeit des Client Rechners.
- **CPU Zeit:** CPU Zeit, die der Anwender seit der Anmeldung verbraucht hat.
- **Aktivität:** Zeitspanne, die 4D Server diesem Benutzer widmet (dynamische Anzeige). "Schlafend", wenn der Client Rechner in den Schlafmodus gewechselt hat (siehe unten)

## Handhabung schlafender Benutzer

Geht das Betriebssystem einer angemeldeten 4D remote Anwendung in den Schlafmodus, erhält 4D Server automatisch eine Meldung über die vorübergehende Abmeldung.

Der angemeldete Benutzer erhält auf dem Server den Status **schlafend**:

4D Anwender	Rechner	Session-Name	IP Adresse	Anmeldedatum	CPU Zeit	Aktivität
Designer	ALRIK-WIN8	Alrik	localhost	21.03.2016 14:52	00:00:01	0%
Designer	Administrators MacBook Pro	Alrik	192.168.10.61	21.03.2016 14:52	00:00:01	schlafend

Dieser Status gibt Ressourcen auf dem Server frei. Bei Beenden des Schlafmodus nimmt die 4D remote Anwendung die Verbindung zu 4D Server automatisch wieder auf.

Der Ablauf ist wie folgt: Ein remote Benutzer unterbricht seine Arbeit für eine Weile, z.B. für die Mittagspause, bleibt aber am Server angemeldet. Sein Rechner geht in den Schlafmodus. Ist der Benutzer zurück, wacht der Rechner wieder auf und die 4D remote Anwendung stellt automatisch die Verbindung zum Server sowie den Kontext der Sitzung wieder her.

## Bereich Suchen/Filtern

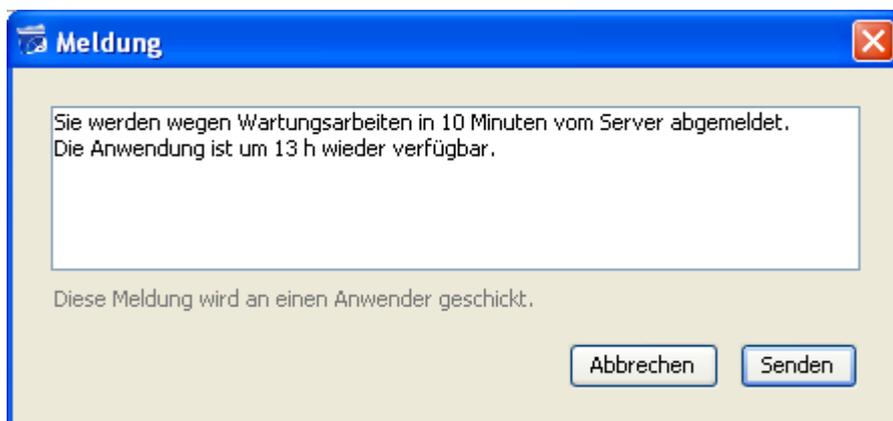
Hier können Sie die angezeigten Zeilen in der Liste auf den im Suchbereich eingegebenen Text beschränken. Er gibt die Spalten an, in denen Suchen/Filtern ausgeführt wird. Das sind auf der Seite Anwender die Spalten Anwender; Rechner und Session.

Die Liste wird unmittelbar aktualisiert, wenn Sie Text im Bereich eingeben. Sie können auch mehrere Werte, getrennt durch Strichpunkt, eingeben. Dann wird zum Filtern der Operator ODER verwendet. Geben Sie z.B. "Hans;Susi;Petra," ein, werden in den Zielspalten nur Zeilen mit Hans ODER Susi ODER Petra beibehalten.

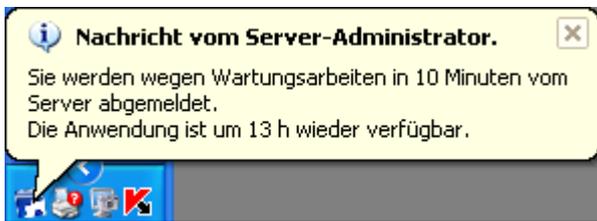
## Schaltflächen zum Verwalten

Der untere Bereich enthält drei Schaltflächen. Sie sind aktiv, wenn mindestens eine Zeile ausgewählt ist. Mit gedrückter **Großschreibtaste** können Sie zusammenhängende Zeilen, mit gedrückter **Strg-Taste** unter Windows, **Befehlstaste** auf Mac OS nicht-zusammenhängende Zeilen auswählen.

- **Nachricht senden:** Damit senden Sie eine Meldung an die in der Liste ausgewählten 4D Benutzer. Ist kein Benutzer markiert, ist diese Schaltfläche nicht aktiv.  
Klicken Sie auf diese Schaltfläche, erscheint ein Dialogfenster zum Eingeben der Meldung.  
Es zeigt die Anzahl der Benutzer an, die diese Meldung erhalten werden:



Auf den Client Rechnern erscheint folgende Meldung am unteren Rand:



- **Prozesse überwachen:**

Damit können Sie direkt die Prozesse der gewählten Benutzer auf der Seite **Prozesse** im Verwaltungsfenster anzeigen. Klicken Sie auf diese Schaltfläche, wechselt 4D Server auf die Seite Prozesse und gibt die ausgewählten Benutzernamen im Bereich Suchen/Filtern dieser Seite ein. Weitere Informationen dazu finden Sie im Abschnitt **Seite Prozesse**.

- **Anwender abmelden:** Damit können Sie den/die ausgewählten Benutzer direkt abmelden.

Klicken Sie auf diese Schaltfläche, erscheint eine Meldung, um die Operation zu bestätigen oder abzubrechen.

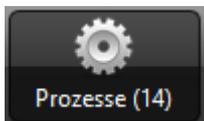
**Hinweis:** Sie können den/die gewählten Benutzer auch direkt abmelden, d.h. ohne Dialog zum Bestätigen. Klicken Sie dazu unter Windows mit gedrückter **Alt-Taste**, auf Mac OS mit gedrückter **Wahltaste** auf diese Schaltfläche.

## Seite Prozesse

Die Seite Prozesse listet alle Prozesse in Betrieb auf:

Prozessname	Session	Typ	Num	Status	CPU-Zeit	Aktivität
Arbeitsplatz-Manager	-	Anwendungsserver	3	Warte auf Ereignis	00:00:02	0 %
DB4D CRON	-	DB4D Server	0	Ausführung	00:00:00	0%
DB4D Flush	-	DB4D Server	0	Ausführung	00:00:00	0%
DB4D Index builder	-	DB4D Server	0	Ausführung	00:00:01	0%
DB4D Server	-	DB4D Server	0	Ausführung	00:00:00	0%
DB4D Sockets	-	DB4D Server	0	Ausführung	00:00:00	0%
Garbage Handler	-	DB4D Server	0	Ausführung	00:00:00	0%
Interner Timer Prozess	-	Anwendungsserver	2	Ausführung	00:00:01	0 %
Reused spare process	-	4D Client Datenbank-Prozess	0	Ausführung	00:00:01	0%
Serverfenster	-	Anwendungsserver	1	Warte auf Ereignis	00:00:04	10 %
Task-Managers	-	SQL Server	0	Ausführung	00:00:00	0%
TCP connection listener	-	SQL Server	0	Ausführung	00:00:00	0%
Anwendungsprozess	Alrik	4D Client Prozess	4	Warte auf Ereignis	00:00:01	0%

Die Schaltfläche "Prozesse" gibt in Klammern die Gesamtanzahl der in der Datenbank laufenden Prozesse an. Diese Nummer berücksichtigt weder Anzeigefilter, die auf das Fenster angewandt werden, noch den Status der Option **Reserve Prozesse**.



Sie können die Anordnung der Spalten durch Markieren und Ziehen des Titelbereichs verändern. Sie können auch die Werte in den Spalten durch Klicken in den Titelbereich sortieren.

Diese Seite enthält, wie die **Seite Anwender**, einen dynamischen Bereich zum Suchen/Filtern. Hier können Sie die angezeigten Zeilen in der Liste auf den im Suchbereich eingegebenen Text beschränken. Er gibt die Spalten an, in denen Suchen/Filtern ausgeführt wird. Das sind auf der Seite Prozesse die Spalten Session und Prozessname.

Session;Prozessname...

Darunter gibt es auch drei Schaltflächen, um die Anzeige der Prozesstypen zu filtern:



- **Anwender Prozesse:** Prozesse, die durch und für Benutzersitzungen erzeugt werden. Sie sind mit einem Icon in Form einer Figur gekennzeichnet.

- **4D Prozesse:** Prozesse, die von der 4D Server Engine erzeugt werden. Sie sind mit einem Icon in Form eines Zahnrads gekennzeichnet.
- **Reserve Prozesse:** Inaktive Prozesse, die temporär beibehalten werden und jederzeit wieder verwendbar sind. Das optimiert die Reaktionszeit von 4D Server. Sie sind mit dem jeweiligen Icon in Grau gekennzeichnet.

Mit der Option **Prozesse gruppiert anzeigen** können Sie die internen Prozesse von 4D Server und die Client-Prozesse für eine bessere Lesbarkeit gruppieren. Markieren Sie diese Option, passiert folgendes:

- Die 4D Client „Zwillingsprozesse“ (4D Client Hauptprozess und 4D Client Basisprozess - siehe Abschnitt "Prozesstypen") werden zu einem Prozess gruppiert,
- Eine Gruppe „Task Managers“ wird angelegt; sie enthält die internen Prozesse, die die einzelnen Tasks aufteilen (Shared balancer, Net session manager, Exclusive pool worker),
- Eine Gruppe „Client Managers“ wird angelegt; sie enthält verschiedene interne Client Prozesse.

Der untere Bereich stellt die Aktivität des/der gewählten Prozesse in grafischer Form dar:



**Hinweis:** Sie können auch mehrere Zeilen auswählen. Mit gedrückter **Umschalttaste** (Shift-Taste) wählen Sie zusammenhängende Zeilen, mit gedrückter **Strg-Taste** unter Windows, **Befehlstaste** auf OS X nicht-zusammenhängende Zeilen aus.

Die Aktivität des Prozesses ist die Zeit in Prozent, die 4D Server diesem Prozess widmet (ratio).

Das Fenster zeigt für jeden Prozess folgende Informationen:

- Prozesstyp (siehe unten)
- Session (bei einem 4D Prozess leer, zeigt bei einem Anwenderprozess den Namen des 4D Benutzers)
- Prozessname
- Prozessnummer (z.B. von der Funktion zurückgegeben) Prozessnummer ist die Nummer, die auf dem Server zugewiesen wird. Bei einem globalen Prozess kann diese Nummer anders sein, als die Nummer, die auf dem Client-Rechner zugewiesen wurde.
- Aktueller Status des Prozesses
- Laufzeit des Prozesses (in Sekunden) seit seiner Erstellung
- Zeit in Prozent, die 4D Server diesem Prozess widmet (ratio)

## Prozesstypen

---

Jedem Prozess ist ein Icon zugeordnet; Farbe und Form kennzeichnen die Art des Prozesses

- ⚙ Anwendungsserver
- ⚙ SQL Server
- ⚙ DB4D Server (Datenbank-Engine)
- ⚙ Web Server
- ⚙ SOAP Server
- 👤 Geschützter 4D Client Prozess (Prozess für Entwicklungsumgebung eines Client)
- 👤 4D Client Hauptprozess (Hauptprozess eines Client), Spiegelprozess eines 4D Client auf dem Server
- 👤 4D Client Basisprozess (Prozess existiert parallel zu einem 4D Client Prozess; preemptive Prozess zur Kontrolle des jeweiligen 4D Client Hauptprozesses)
- 👤 Reserve Prozess (vorheriger oder zukünftiger 4D Client Prozess)
- SQL Server Worker Prozess
- HTTP Server Worker Prozess
- 👤 4D Client Prozess (auf einem Client gestarteter Prozess)
- 👤 Serverprozedur (von einem angemeldeten Client gestarteter Prozess auf dem Server)
- 🌐 Web Methode (z.B. über 4DACTION gestartet)
- 🌐 Web Methode (preemptive)
- 🌐 SOAP Methode (über einen Web Dienst gestartet)
- 🌐 SOAP Methode (preemptive)
- 📄 Logger
- 👁 TCP Connection Listener
- 🕒 TCP Session Manager
- 👤 Anderer Prozess
- Worker Prozess (kooperativ)
- 👤 Serverprozedur (preemptive Prozess)
- Worker Prozess (preemptive)

**Hinweis:** Jeder 4D Client Hauptprozess und sein „Zwilling“ 4D Client Basisprozess werden zusammengelegt, wenn die Option *Prozesse gruppiert anzeigen* markiert ist.

## Schaltflächen zum Verwalten

---

Der untere Bereich enthält fünf Schaltflächen, die bei ausgewählten Prozessen aktiv werden. Beachten Sie jedoch, dass sie nur für Anwenderprozesse gelten.



Prozess löschen



Prozess anhalten



Prozess fortsetzen



Prozess debuggen



Anwender beobachten

- **Prozess löschen:** Damit können Sie den/die gewählten Prozesse löschen. Klicken Sie auf die Schaltfläche, erscheint ein Dialogfenster mit einer Meldung, um die Operation zu bestätigen oder zu annullieren.  
**Hinweis:** Sie können den/die gewählten Prozesse auch direkt löschen, d.h. ohne Dialog zum Bestätigen. Klicken Sie dazu unter Windows mit gedrückter **Alt-Taste**, auf OS X mit gedrückter **Wahltaste** auf diese Schaltfläche.
- **Prozess anhalten:** Damit können Sie den/die gewählten Prozesse anhalten.
- **Prozess fortsetzen:** Damit können Sie den/die gewählten Prozesse wieder aktivieren. Dazu muss der Prozess zuvor angehalten worden sein, entweder

über die Schaltfläche oder per Programmierung. Sonst hat die Schaltfläche keine Auswirkung.

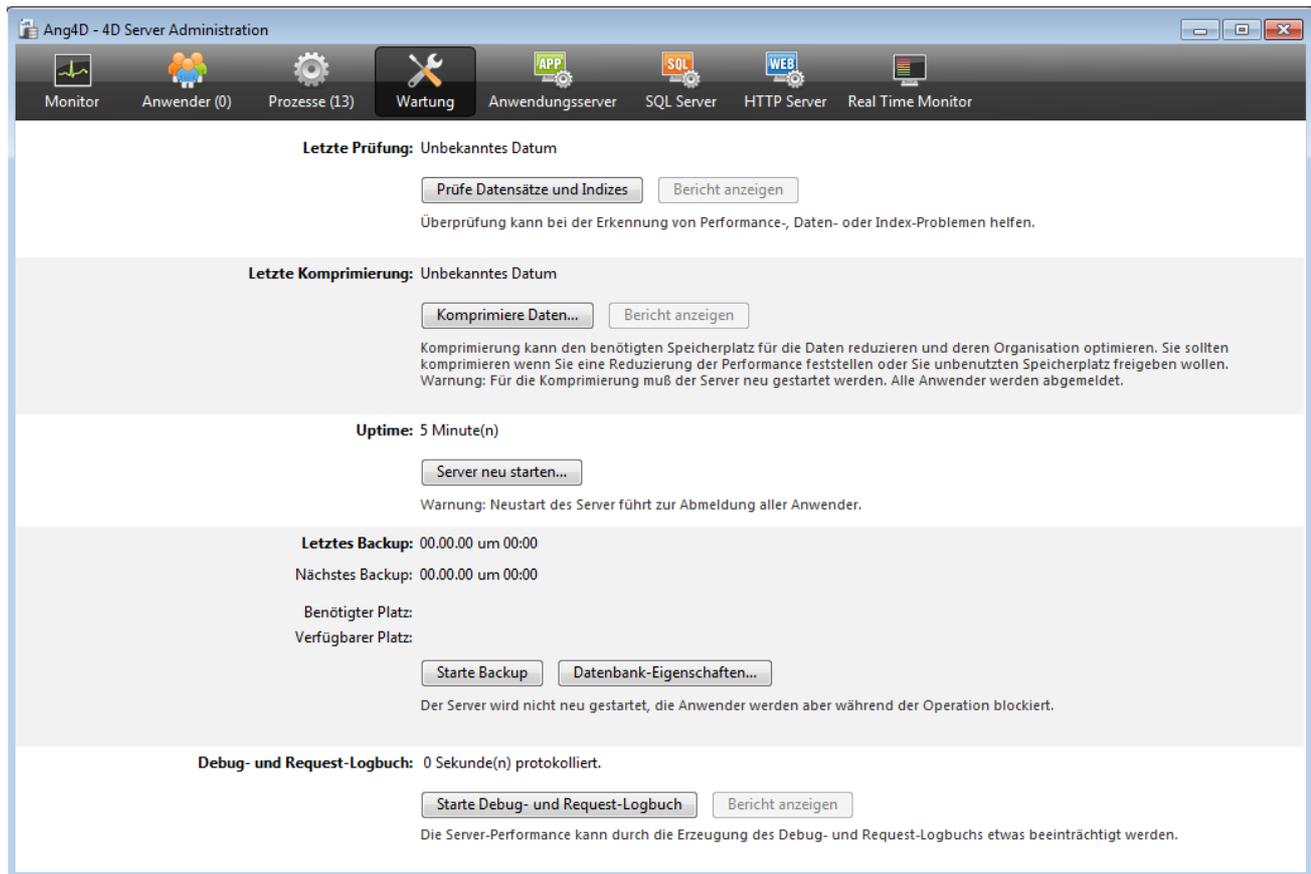
- **Prozess debuggen:** Damit können Sie auf dem Server Rechner ein oder mehrere Debugger Fenster für den/die gewählten Prozesse öffnen. Klicken Sie auf die Schaltfläche, erscheint ein Dialogfenster, um die Operation zu bestätigen oder zu annullieren.

Das Debugger Fenster wird nur angezeigt, wenn der 4D Code gerade auf dem Server Rechner ausgeführt wird, z.B. im Rahmen eines Triggers oder beim Ausführen einer Methode mit dem Attribut "Auf Server ausführen".

**Hinweis:** Sie können einen Prozess auch direkt debuggen, d.h. ohne das Dialogfenster zum Bestätigen anzuzeigen. Klicken Sie dazu unter Windows mit gedrückter **Alt-Taste**, auf OS X mit gedrückter **Wahltaste** auf die Schaltfläche.

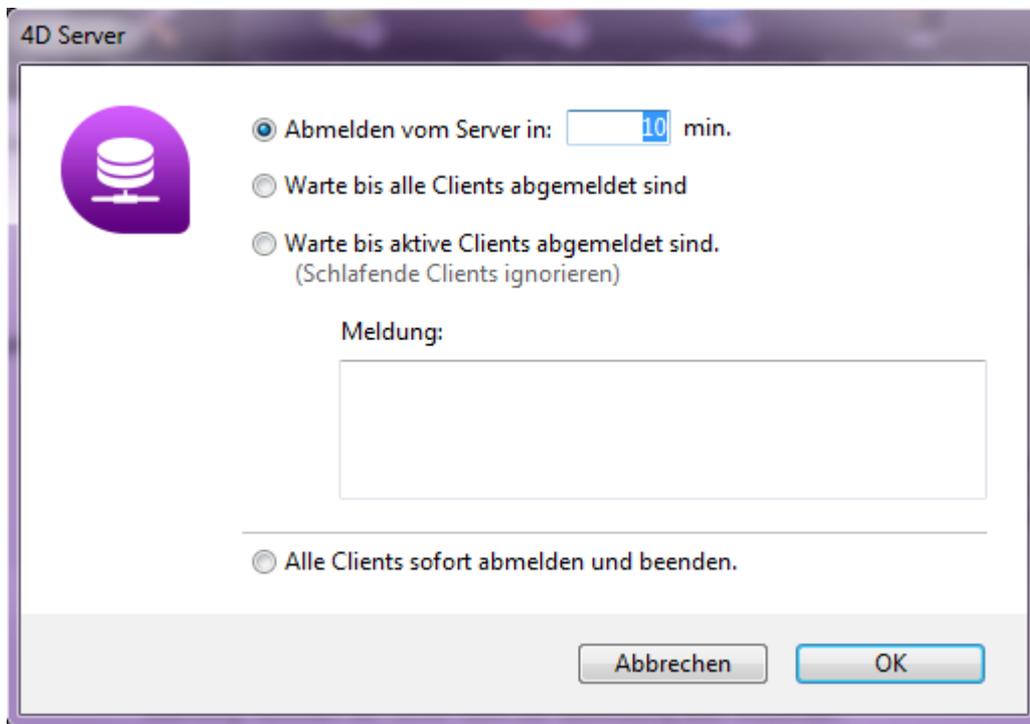
- **Anwender beobachten:** Damit können Sie auf der Anwenderseite alle Prozesse des/der gewählten Benutzer anzeigen. Diese Schaltfläche ist aktiv, wenn mindestens ein Anwenderprozess ausgewählt wurde.

Die Seite Wartung zeigt Informationen zur aktuellen Operation der Datenbank und bietet Zugriff auf die wichtigsten Wartungsfunktionen:



- **Letzte Prüfung:** Dieser Bereich zeigt Datum, Uhrzeit und Status der zuletzt ausgeführten Überprüfung der Anwendung. Weitere Informationen dazu finden Sie im Handbuch 4D Designmodus. Über die Schaltfläche **Prüfe Datensätze und Indizes** können Sie die Überprüfung direkt starten, ohne den Server zu unterbrechen. Bedenken Sie jedoch, dass dies u.U. den Server-Betrieb signifikant verlangsamen kann. Alle Datensätze und alle Indizes der Datenbank werden überprüft. Für eine gezielte Überprüfung oder weitere Optionen müssen Sie das Maintenance- und Security-Center (MSC) aufrufen. Nach der Prüfung wird auf dem Server im Ordner Logs (neben der Strukturdatei der Anwendung) ein Bericht im Format XML und HTML angelegt. Über die Schaltfläche **Zeige Logbuch** können Sie den Bericht in Ihrem Browser anzeigen. Wird die Operation auf einem Client Rechner ausgeführt, lautet sie **Logbuch laden**.
- **Letzte Komprimierung:** Dieser Bereich zeigt Datum, Uhrzeit und Status der zuletzt ausgeführten Komprimierung der Anwendung. Weitere Informationen dazu finden Sie im Handbuch 4D Designmodus. Über die Schaltfläche **Komprimiere Daten** können Sie die Komprimierung der Daten direkt starten. Für diese Operation muss der Server jedoch beendet

werden. Klicken Sie auf diese Schaltfläche, erscheint das Dialogfenster zum Abmelden, wo Sie festlegen können, wie der Server-Betrieb beendet wird:



Weitere Informationen zu diesem Dialogfenster finden Sie im Abschnitt **4D Server beenden**

Wurde die Anwendung unterbrochen, führt 4D Server eine standardmäßige Komprimierung der Daten aus. Für weitere Optionen müssen Sie das Maintenance- und Security-Center (MSC) aufrufen.

Nach dem Komprimieren startet 4D Server die Anwendung automatisch neu. Die 4D Anwender können dann wieder angeschlossen werden.

**Hinweis:** Wurde die Anfrage zur Komprimierung von einem Remote 4D Client Rechner ausgeführt, meldet 4D Server diesen Rechner automatisch wieder an.

Auf dem Server wird im Ordner **Logs** (neben der Strukturdatei der Anwendung) ein Bericht im Format XML und HTML angelegt. Über die Schaltfläche **Zeige Logbuch** können Sie den Bericht in Ihrem Browser anzeigen. Wird die Operation auf einem Client Rechner ausgeführt, lautet sie **Logbuch laden**.

- **Uptime:** Dieser Bereich zeigt die Dauer des Server-Betriebs seit dem letzten Start in Tagen, Stunden und Minuten.  
Über die Schaltfläche **Server neu starten** können Sie den Server sofort neu starten. Klicken Sie auf diese Schaltfläche, erscheint das Dialogfenster zum Abmelden, wo Sie festlegen können, wie der Server-Betrieb unterbrochen wird (siehe Abschnitt **4D Server beenden**).  
Nach dem Neustart startet 4D Server die Anwendung automatisch neu. Die 4D Anwender können dann wieder angeschlossen werden.  
**Hinweis:** Wurde die Anfrage zum Neustart von einem Remote Client Rechner ausgeführt, meldet 4D Server diesen Rechner automatisch wieder an.
- **Letztes Backup:** Dieser Bereich zeigt Datum und Uhrzeit des letzten automatischen Backup der Anwendung und meldet, wann das nächste Backup geplant ist. Automatische Backups richten Sie in den Einstellungen der Anwendung auf der Seite **Backup>Planer** ein.
  - Nächstes Backup: Datum und Uhrzeit des nächsten automatischen Backup.
  - Benötigter Platz: Geschätzter Platz, der für das Backup benötigt wird. Die aktuelle Größe der Backup-Datei kann je nach den Einstellungen (Komprimierung, etc.) und den Veränderungen der Datendatei unterschiedlich

sein.

- Verfügbarer Platz: Platz, der auf dem Backup Volume verfügbar ist  
Über die Schaltfläche **Starte Backup** können Sie sofort ein Backup der Anwendung mit den aktuellen Backup Einstellungen (Backup-Inhalt, Zielverzeichnis, etc.) durchführen. Um die Einstellungen anzusehen, klicken Sie auf die Schaltfläche **Voreinstellungen**. Während dem Backup auf dem Server werden die Client-Rechner blockiert, aber nicht abgekoppelt. Neue Clients können sich in dieser Zeit nicht anmelden.

- **Debug- und Request-Logbuch:** Dieser Bereich zeigt die Dauer zum Protokollieren von Anfragen und Debug Ereignisse, wenn sie aktiviert sind.
  - Das **Request-Logbuch** speichert Informationen zu den Anfragen, die der Server empfängt (ohne Web Anfragen): Zeit, Prozessnummer, Anwender, Größe der Anfrage, Prozesszeit, etc.. Diese Daten dienen zur Analyse der Server-Operationen. Das Logbuch hat den Namen *4DRequestsLog\_X*, wobei X die fortlaufende Nummer der Datei ist, und wird im Ordner **Logs** der Anwendung abgelegt. Bei einer Größe von 10 MB wird es geschlossen und ein neues Logbuch mit der nächsthöheren Nummer angelegt.
  - Das **Debug-Logbuch** speichert jede Ausführung einer Methode, eines 4D Befehls oder Plug-In Befehls in einer Datei mit Namen *4DDebugLog\_X.txt*. Sie wird automatisch in den Unterordner **Logs** der Anwendung neben der Strukturdatei abgelegt. Jedes Ereignis wird systematisch vor der Ausführung in der Datei protokolliert. So wird sichergestellt, dass es im Logbuch vorhanden ist, selbst wenn die Anwendung unerwartet endet. Beachten Sie, dass diese Datei bei jedem Starten der Anwendung entfernt und neu geschrieben wird. Über den Befehl **SET DATABASE PARAMETER** können Sie diese Datei konfigurieren.

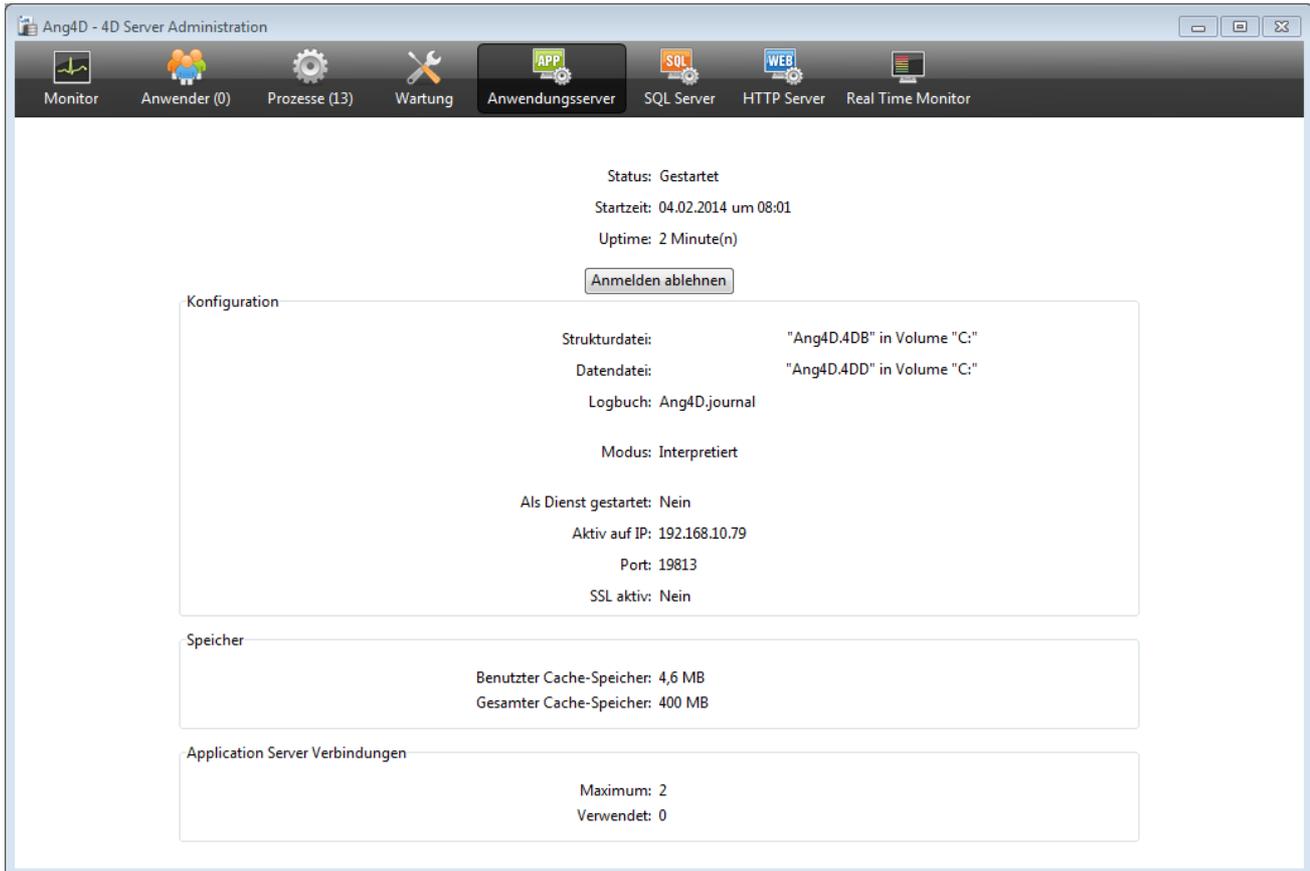
Die Schaltfläche **Starte Debug- und Request-Logbuch** startet das Protokollieren der Anfragen und speichert Debug Ereignisse. Da dies die Server-Leistung entscheidend beeinträchtigen kann, sollte die Operation nur in der Entwicklungsphase der Anwendung ausgeführt werden. Ist das Logbuch aktiviert, wechselt die Schaltfläche in **Stopp Debug- und Request-Logbuch**, so dass sich das Protokollieren jederzeit stoppen lässt. Bedenken Sie jedoch, dass erneutes Starten des Logbuchs die vorige Datei überschreibt.

**Hinweis:** Mit dem Befehl **SET DATABASE PARAMETER** lassen sich die Logbücher per Programmierung starten und stoppen.

Über die Schaltfläche **Bericht anzeigen** können Sie ein Systemfenster öffnen, das die Datei Request Logbuch anzeigt. Wird die Operation auf einem Client-Rechner ausgeführt, lautet die Schaltfläche **Bericht laden**.

## Seite Anwendungsserver

Die Seite Anwendungsserver enthält Informationen über die von 4D Server veröffentlichte Anwendung und ermöglicht, dies zu steuern:



The screenshot shows the 'Ang4D - 4D Server Administration' window. The 'Anwendungsserver' tab is selected. The status is 'Gestartet' (Started) with a start time of '04.02.2014 um 08:01' and an uptime of '2 Minute(n)'. A button labeled 'Anmelden ablehnen' is visible. The configuration section includes: Strukturdatei: "Ang4D.4DB" in Volume "C:."; Datendatei: "Ang4D.4DD" in Volume "C:."; Logbuch: Ang4D.journal; Modus: Interpretiert; Als Dienst gestartet: Nein; Aktiv auf IP: 192.168.10.79; Port: 19813; SSL aktiv: Nein. The Speicher section shows: Benutzer Cache-Speicher: 4,6 MB; Gesamter Cache-Speicher: 400 MB. The Application Server Verbindungen section shows: Maximum: 2; Verwendet: 0.

### Status Information

Der obere Teil zeigt Informationen über den aktuellen Status des Anwendungsserver von 4D Server.

- **Status:** Gestartet oder gestoppt.
- **Startzeit:** Datum und Uhrzeit, zu der die Server-Anwendung gestartet wurde. Dieser Zeitpunkt entspricht dem Öffnen der Anwendung durch 4D Server.
- **Uptime:** Zeitspanne seit dem letzten Öffnen der Anwendung.

### Schaltfläche Anmelden erlauben/ablehnen

Diese Schaltfläche wechselt zwischen **Anmelden erlauben** und **Anmelden ablehnen**. Damit wird der Zugriff neuer Client Rechner auf den Anwendungsserver verwaltet.

- Wird die Anwendung veröffentlicht, gilt standardmäßig folgendes:
  - Die Schaltfläche lautet "Anmelden verweigern."

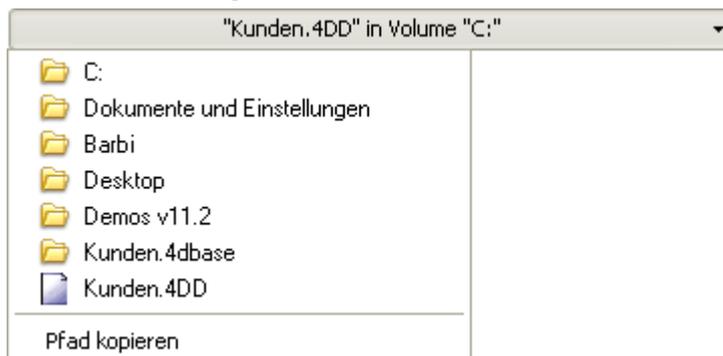
- Neue Clients können sich frei anmelden, soweit die erworbene Lizenz dies zulässt.
- Der Datenbankname wird im Anmeldedialog angezeigt. Dazu muss in den Einstellungen der Anwendung auf der Seite **Anwendung>Optionen** die Option "4D Server Verbindungsdialog anzeigen" ausgewählt sein.
- Klicken Sie auf die Schaltfläche **Anmelden ablehnen**:
  - Die Schaltfläche wechselt in "Anmelden erlauben".
  - Es kann sich kein neuer Client anmelden.
  - Der Name der Anwendung erscheint nicht mehr im Anmeldedialog.
  - Bereits angemeldete Clients werden nicht abgekoppelt und können normal weiterarbeiten.
  - Klicken Sie auf die Schaltfläche **Anmelden erlauben**, kehrt die Anwendung in den standardmäßigen Status zurück.

Diese Funktionalität ermöglicht dem Administrator z.B. verschiedene Wartungsoperationen, wie Überprüfen, Komprimieren, o.ä., direkt nach Starten des Servers auszuführen. Bei Verwendung einer Client-Verbindung kann er sicher sein, dass er der einzige ist, der Daten verändert. Er kann diese Funktionalität auch zur Vorbereitung von Wartungsarbeiten verwenden, für die kein Client Rechner angemeldet sein darf.

## Konfiguration

---

Dieser Bereich zeigt Informationen über die vom Server veröffentlichte 4D Anwendung: Name und Speicherort der Strukturdatei, der Datendatei und des Logbuchs. Sie können auf den Namen der Strukturdatei oder Datendatei klicken, um den vollständigen Pfadnamen zu sehen:



Der Bereich Modus zeigt den aktuellen Ausführungsmodus an, d.h. ob die Datenbank interpretiert oder kompiliert läuft.

Der untere Bereich zeigt die Parameter der Server-Konfiguration: Als Dienst gestartet, IP Adresse und Port und ermöglicht SSL für Client/Server-Verbindungen (gilt weder für SQL noch Web Verbindungen).

## Speicher

---

Dieser Bereich zeigt den Wert für Gesamter Cache-Speicher (definiert in den Einstellungen der Anwendung) und den Wert für **Benutzter Cache-Speicher** (dynamische Zuweisung durch 4D Server nach Bedarf).

## Maximale Verbindungen

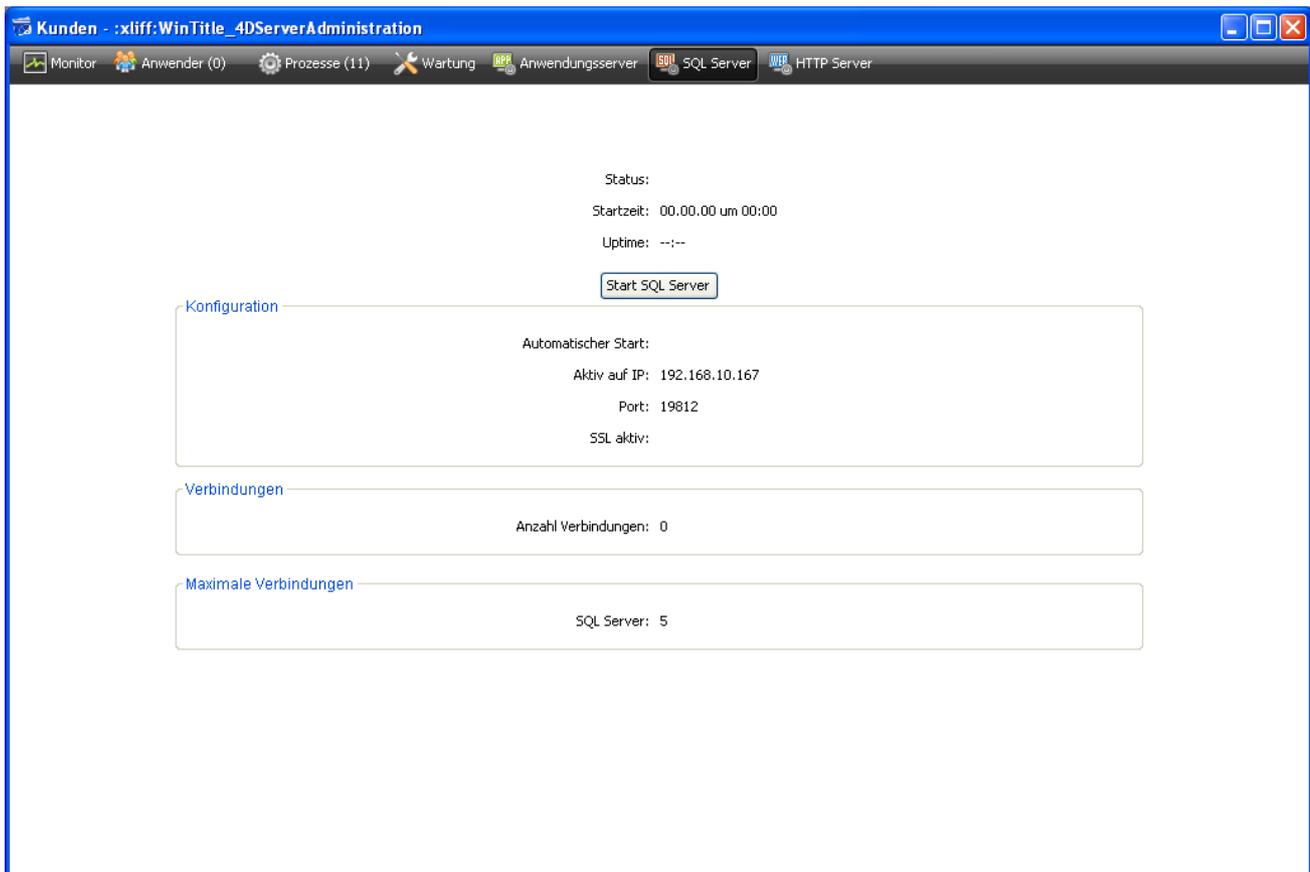
---

**Maximum** gibt die maximal zugelassene Anzahl gleichzeitig angemeldeter Clients für den Applikation Server an. Dies richtet sich nach der Lizenz, die auf dem Server Rechner installiert ist.

**Used** gibt die aktuelle Anzahl gerade gleichzeitig verwendeter Verbindungen an.

## Seite SQL Server

Die Seite SQL Server enthält Informationen über den in 4D Server integrierten SQL Server und eine Schaltfläche zum Steuern der Aktivierung:



## Status Information

Der obere Teil zeigt Informationen über den aktuellen Status des SQL Server von 4D Server.

- **Status:** Gestartet oder gestoppt
- **Startzeit:** Datum und Uhrzeit, zu der der SQL Server gestartet wurde. Diese Angaben können sich vom Startzeitpunkt des Anwendungsserver unterscheiden, wenn der SQL Server nicht gleichzeitig mitgestartet wird.
- **Uptime:** Zeitspanne seit dem letzten Start des SQL Server.

## Schaltfläche Start / Stop SQL Server

Diese Schaltfläche wechselt. Damit steuern Sie die Aktivierung des SQL Server von 4D Server.

- Ist der Status des SQL Server "Gestartet", lautet die Schaltfläche **Stop SQL Server**. Klicken Sie auf die Schaltfläche, wird der SQL Server von 4D Server

sofort gestoppt; er antwortet nicht mehr auf externe SQL Anfragen, die auf dem angegebenen TCP Port empfangen werden.

- Ist der Status des SQL Server "Gestoppt," lautet die Schaltfläche **Start SQL Server**. Klicken Sie auf die Schaltfläche, wird der SQL Server von 4D Server sofort gestartet; er antwortet auf externe SQL Anfragen, die auf dem angegebenen TCP Port empfangen werden. Beachten Sie, dass Sie zum Nutzen des 4D SQL Server eine passende Lizenz benötigen.

**Hinweis:** Der SQL Server lässt sich auch automatisch beim Starten der Anwendung (Option in den Einstellungen der Anwendung) oder per Programmierung starten.

## Konfiguration

---

Dieser Bereich zeigt Informationen über die Parameter des SQL Server: Automatisches Hochfahren beim Start, aktive IP Adresse, TCP Port (standardmäßig 19812) und ermöglicht SSL für SQL-Verbindungen (gilt weder für 4D noch Web Verbindungen).

Diese Parameter lassen sich in den 4D Einstellungen der Anwendung verändern.

## Verbindungen

---

Anzahl der aktuell auf 4D Server geöffneten SQL Verbindungen.

## Maximale Verbindungen

---

Maximal zugelassene Anzahl gleichzeitiger SQL Verbindungen. Dies richtet sich nach der Lizenz, die auf dem Server Rechner installiert ist.

## Seite HTTP Server

Die Seite HTTP Server enthält Informationen über den Web Server und den SOAP Server von 4D Server und Schaltflächen zum Steuern ihrer Aktivierung. Der Web Server ermöglicht, Web Inhalt wie HTML Seiten oder Bilder für Web Browser zu veröffentlichen. Der SOAP Server steuert das Veröffentlichen von Web Services. Beide Server gehören zum internen HTTP Server von 4D Server:

## Status Information

Der obere Teil zeigt Informationen über den aktuellen Status des HTTP Server von 4D Server.

- **Status:** Gestartet oder gestoppt
- **Startzeit:** Datum und Uhrzeit, zu der der HTTP Server gestartet wurde. Diese Angaben können sich vom Startzeitpunkt des Anwendungsserver unterscheiden, wenn der HTTP Server nicht gleichzeitig mitgestartet wird.
- **Uptime:** Zeitspanne seit dem letzten Start des HTTP Server
- **HTTP Hit Gesamt:** Anzahl der HTTP Hits (low level), die seit dem Start des HTTP Server empfangen wurden.

## Schaltfläche Start / Stop HTTP Server

Diese Schaltfläche wechselt. Damit steuern Sie die Aktivierung des HTTP Server von 4D Server.

- Ist der Status des HTTP Server "Gestartet", lautet die Schaltfläche **Stop HTTP Server**. Klicken Sie auf die Schaltfläche, wird der HTTP Server von 4D Server sofort gestoppt; Web Server und SOAP Server akzeptieren keine Anfragen mehr.
- Ist der Status des HTTP Server "Gestoppt," lautet die Schaltfläche **Start HTTP Server**. Klicken Sie auf die Schaltfläche, wird der HTTP Server von 4D Server sofort gestartet; Web und SOAP Anfragen werden akzeptiert. Sie können den SOAP Server auch separat stoppen. Weitere Informationen dazu finden Sie im unteren Abschnitt **SOAP Information**.

#### **Hinweise:**

- Beachten Sie, dass Sie zum Starten des HTTP Server eine passende Lizenz benötigen.
- Der HTTP Server lässt sich auch automatisch beim Starten der Anwendung (Option in den Voreinstellungen) oder per Programmierung starten.

## **Web Information**

---

Dieser Bereich enthält spezifische Angaben zum Web Server von 4D Server.

- **Web Anfragen:** Angenommen oder Verweigert. Diese Information zeigt, ob der Web Server aktiviert ist. Da der Web Server direkt mit dem HTTP Server verbunden ist, werden Web Anfragen angenommen, wenn der HTTP Server gestartet ist und verweigert, wenn er gestoppt ist.
- **Maximale Verbindungen:** Maximale Anzahl der zugelassenen Web Verbindungen. Dieser Wert richtet sich nach der Lizenz, die auf dem Server Rechner installiert ist.

## **SOAP Information**

---

Dieser Bereich enthält spezifische Angaben zum SOAP Server von 4D Server sowie eine Schaltfläche zum Steuern der SOAP Anfragen.

- **SOAP Anfragen:** Angenommen oder Verweigert. Diese Information zeigt, ob der SOAP Server aktiviert ist. Damit SOAP Anfragen angenommen werden, muss der SOAP Server gestartet sein und die Anfragen müssen explizit angenommen werden (siehe Schaltfläche SOAP Anfragen annehmen/ablehnen).
- **Maximale Verbindungen:** Maximale Anzahl der zugelassenen SOAP Verbindungen. Dieser Wert richtet sich nach der Lizenz, die auf dem Server Rechner installiert ist.
- **SOAP Anfragen annehmen/ablehnen:** Diese Schaltfläche wechselt. Damit steuern Sie die Aktivierung des SOAP Server von 4D Server. Diese Schaltfläche ändert den Wert der Option **Web Service Abfragen erlauben** auf der Seite "Web Services" in den Datenbank-Eigenschaften (und umgekehrt). Klicken Sie auf die Schaltfläche **SOAP Anfragen annehmen** und wird der HTTP Server gestoppt, startet 4D ihn automatisch.

## **HTTP Server Konfiguration**

---

Dieser Bereich zeigt Informationen über die Konfiguration und Operation des HTTP Server:

- **Automatischer Start:** Dies wird in den Voreinstellungen von 4D Server festgelegt.
- **HTTP Server Prozesse (benutzt/gesamt):** Anzahl der HTTP Prozesse, die auf dem Server erstellt werden (aktuelle Anzahl und gesamte Anzahl aller erstellten Prozesse).
- **Cache-Speicher:** Größe des Cache Speichers des HTTP Server, wenn er aktiviert ist (Der aktuell vom Cache verwendete Platz und die Gesamtgröße werden in der Regel in den Voreinstellungen zugewiesen.).
- **Aktiv auf IP, TCP Port (standardmäßig 80),** SSL aktiv für HTTP Verbindungen (nicht 4D oder SQL Verbindungen) und verwendeter HTTPS Port: Aktuelle Konfigurationsparameter des HTTP Server, der in den Datenbank-Eigenschaften auf der Seite **Web>Konfiguration** angegeben ist. Weitere Informationen dazu finden Sie im Abschnitt **QR DELETE COLUMN**.
- **Logbuch Information:** Speicherort, Format und Datum des nächsten automatischen Logbuch Backup des HTTP Server (Datei logweb.txt)

## Seite Real Time Monitor

Die Seite Real Time Monitor überwacht das Fortschreiten "langer" Operationen in der Anwendung in Echtzeit. Das sind z.B. sequentielle Suchläufe, Ausführen von Formeln, etc.:

Startzeit	Dauer (ms)	Information
2014-06-27 11:00:49:395	176.623	Sequentielle Suche von Companies: 1325694 von 1535650 Daten...
2014-06-27 11:03:28:918	17.100	Lösche Datensätze: 3 von 10
2014-06-27 11:03:30:648	15.370	Array to selection: 4 von 10000
2014-06-27 11:03:44:378	1.640	Lade Daten
2014-06-27 11:03:44:498	1.520	Lade Daten

**Erzeugt auf Client**

**Operation Details**

Operationstyp: Lösche Datensätze  
Tabelle: Companies

**Prozess-Details**

Client Prozess Num: 7  
Prozess-Name: P\_11  
4D Anwender: Designer  
Session Name: Barbara  
Computer Name: BARBARA-WIN7

Angehalten Snapshot Fortsetzen

Diese Seite ist im Verwaltungsfenster des Server Rechners und auf Rechnern mit remote 4D verfügbar. Auf einem remote Rechner zeigt diese Seite Daten zu Operationen, die auf dem Server Rechner ausgeführt werden.

Jede lange Operation, die mit Daten ausgeführt wird, erscheint in einer eigenen Zeile. Sie verschwindet automatisch, wenn die Operation abgeschlossen ist. Sie können die Option **Operationen für mindestens 5 Sekunden anzeigen** markieren, damit Operationen von kurzer Dauer mindestens fünf Sekunden lang angezeigt werden (siehe unten):

Jede Zeile zeigt folgende Informationen:

- **Startzeit:** Startzeit der Operation im Format: "dd/mm/yyyy - hh:mm:ss"
- **Dauer (ms):** Dauer in Millisekunden der ablaufenden Operation
- **Information:** Titel der Operation
- **Details:** Dieser Bereich zeigt detaillierte Information, die je nach Typ der ausgewählten Operation variiert. Es gibt folgende Angaben:
  - **Erzeugt auf:** Gibt an, ob die Operation von einer Operation auf dem Client stammt (**Erzeugt auf Client**) oder explizit auf dem Server gestartet wurde, entweder über eine Serverprozedur oder über die Option "Auf dem Server ausführen" (**Erzeugt auf Server**).
  - **Operation Details:** Art der Operation und bei Suchoperationen Suchplan.

- **Unteroperationen** (wenn vorhanden): Unteroperationen zur gewählten Operation, z.B. verknüpfte Datensätze vor einem Hauptdatensatz löschen
- **Prozess-Details:** Je nach Operationsart zusätzliche Information zu Tabelle, Feld, Prozess oder Client.

**Hinweis:** Die Seite Real Time Monitor verwendet den Befehl **GET ACTIVITY SNAPSHOT** intern. Weitere Informationen dazu finden Sie unter diesem Befehl.

Die Seite ist aktiv und wird, sobald sie angezeigt wird, kontinuierlich aktualisiert. Bitte beachten Sie, dass diese Operation die Ausführung der Anwendung signifikant verlangsamen kann. Um die Aktualisierung dieser Seite auszusetzen, können Sie:

- auf die Schaltfläche **Pause** klicken
- in die Liste klicken
- Die Taste für Leerzeichen drücken

Steht die Seite auf Pause, erscheint die Meldung "Angehalten" und die Bezeichnung der Schaltfläche wechselt auf **Fortsetzen**.

Wollen Sie das Überwachen der Operationen wieder einschalten, führen Sie dieselbe Aktion wie für Pausieren aus.

**Hinweis:** Die Überwachung von Operationen in Echtzeit lässt sich auch über den Befehl **GET ACTIVITY SNAPSHOT** einstellen.

## Erweiterter Modus

Die Seite RTM kann jetzt bei Bedarf für jede gelistete Operation zusätzliche Information anzeigen.

Drücken Sie dazu die **Umschalttaste (Shift-Taste)** und wählen die gewünschte Operation. Die verfügbare Information erscheint ungefiltert im Bereich Details, d.h. so wie sie vom Befehl **GET ACTIVITY SNAPSHOT** zurückgegeben wird. Je nach der gewählten Operation sind unterschiedliche Informationen verfügbar.

Hier ein Beispiel für angezeigte Information im Standardmodus:

Startzeit	Dauer (ms)	Information	Details
2014-06-27 11:06:47:941	187.605	Sequentielle Suche von Companies: 1040648 von 1585560 Daten...	<b>Erzeugt auf Client</b>
2014-06-27 11:09:24:034	31.512	Lösche Datensätze: 8 von 10	<b>Operation Details</b>
2014-06-27 11:09:25:124	30.422	Array to selection: 10 von 10000	Operationstyp: Suche Query Plan: Companies.ID == 5
2014-06-27 11:09:51:836	3.710	Lade Daten	<b>Prozess-Details</b>
2014-06-27 11:09:54:146	1.400	Lade Daten	Client Prozess Num: 5 Prozess-Name: P_9 4D Anwender: Designer Session Name: Barbara Computer Name: BARBARA-WIN7

Im erweiterten Modus (Operation mit gedrückter **Umschalttaste** markieren) erscheint zusätzliche Information:

23-Real\_time\_AP - 4D Server Administration

Monitor Anwender (1) Prozesse (17) Wartung Anwendungsserver SQL Server HTTP Server Real Time Monitor

Operationen für mindestens 5 Sekunden anzeigen

Startzeit	Dauer (ms)	Information
2014-06-27 11:06:47:939	99.470	Sequentielle Suche von Companies: 983488 von 1585560 Datens...
2014-06-27 11:08:05:815	21.594	Lösche Datensätze: 5 von 10
2014-06-27 11:08:07:347	20.062	Array to selection: 7 von 10000
2014-06-27 11:08:25:989	1.420	Lade Daten
2014-06-27 11:08:26:049	1.360	Lade Daten

**Erzeugt auf Client**

**Operation Details**

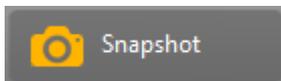
Operationstyp: Suche  
Query Plan: Companies.ID == 5

**Prozess-Details**

Client Prozess Num: 5  
Prozess-Name: P\_9  
4D Anwender: Designer  
Session Name: Barbara  
Computer Name: BARBARA-WIN7  
Client UID: 7C031E951893FC439B7ABC8A84BEA7B1  
Client Version: v14 R3

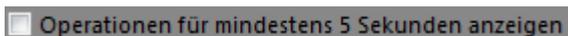
## Schaltfläche Snapshot

Über die Schaltfläche **Snapshot** können Sie alle Operationen, die im RTM Bereich angezeigt werden, sowie dazugehörige Details (Prozess-Details und Unteroperationen) in die Zwischenablage kopieren:



## Operationen für mindestens 5 Sekunden anzeigen

Markieren Sie die Option **Operationen für mindestens 5 Sekunden anzeigen**, werden alle auf der Seite gelisteten Operationen mindestens fünf Sekunden lang angezeigt, selbst wenn die Ausführung der Operation bereits beendet ist:



Beibehaltene Operationen erscheinen in der Auflistung in Grauschrift.

Diese Option ist hilfreich, um Information über nur kurz ablaufende Operationen zu erhalten.

# **Datenbankmethoden**

-  Datenbankmethode On Server Startup
-  Datenbankmethode On Server Shutdown
-  Datenbankmethode On Server Open Connection
-  Datenbankmethode On Server Close Connection

## Datenbankmethode On Server Startup

Datenbankmethode On Server Startup

Dieser Befehl benötigt keine Parameter

---

Die **Datenbankmethode On Server Startup** wird einmal auf dem Server-Rechner aufgerufen, wenn Sie eine Datenbank mit 4D Server öffnen. Die **Datenbankmethode On Server Startup** wird ausschließlich in der 4D Server Umgebung aufgerufen.

Die **Datenbankmethode On Server Startup** ist der richtige Ort, um:

- Interprozessvariablen zu initialisieren, die Sie während der gesamten 4D Server Sitzung verwenden.
- **Serverprozeduren** automatisch beim Öffnen der Datenbank zu starten.
- Voreinstellungen bzw. Einstellungen, die während der letzten 4D Server Sitzung gesichert wurden, zu laden.
- Durch explizites Aufrufen von **QUIT 4D** das Öffnen der Datenbank nicht zuzulassen, wenn eine Bedingung nicht erfüllt ist (z.B. fehlende System-Ressourcen)
- Weitere Aktionen auszuführen, die bei jedem Öffnen der Datenbank automatisch ablaufen sollen.

Um Code automatisch auf einem Client-Rechner auszuführen, wenn sich ein remote 4D am Server anmeldet, verwenden Sie die **Datenbankmethode On Server Startup**.

**Hinweis:** Die **Datenbankmethode On Server Startup** wird automatisch ausgeführt, d.h. ein remote 4D kann sich erst anmelden, wenn die Ausführung der Methode abgeschlossen ist.

## Datenbankmethode On Server Shutdown

Datenbankmethode On Server Shutdown

Dieser Befehl benötigt keine Parameter

---

Die **Datenbankmethode On Server Shutdown** wird einmal auf dem Server-Rechner aufgerufen, wenn die aktuelle Datenbank auf dem Server beendet wird. Diese Datenbankmethode wird ausschließlich in der 4D Server Umgebung aufgerufen.

Um die aktuelle Datenbank auf dem Server zu schließen, können Sie auf dem Server den Menübefehl **Schließe Anwendung** oder **Beenden** wählen oder in einer Serverprozedur, die auf dem Server ausgeführt wird, den Befehls **QUIT 4D** aufrufen.

Beim Verlassen der Datenbank führt 4D folgende Aktionen durch:

- Ist keine **Datenbankmethode On Server Shutdown** vorhanden, beendet 4D Server ohne Unterscheidung der Reihe nach jeden Prozess.
- Ist eine **Datenbankmethode On Server Shutdown** vorhanden, führt 4D Server diese Methode in einem neu angelegten lokalen Prozess durch. Sie können so via Interprozesskommunikation anderen Prozessen mitteilen, die Ausführung zu beenden. Beachten Sie, dass 4D Server evtl. beendet wird – die **Datenbankmethode On Server Shutdown** kann zwar alle gewünschten Operationen zum Aufräumen oder Schließen durchführen, aber nicht das Beenden an sich verweigern, und so an einem gewissen Punkt enden.

Die **Datenbankmethode On Server Shutdown** ist der richtige Ort, um:

- Serverprozeduren zu beenden, die beim Öffnen der Datenbank automatisch starten.
- Voreinstellungen bzw. Einstellungen (lokal oder auf Festplatte) zu speichern, um sie beim Starten der nächsten Sitzung in der **Datenbankmethode On Server Startup** wiederzuverwenden.
- Andere Aktionen auszuführen.
- Andere Aktionen auszuführen, die bei jedem Beenden der Datenbank automatisch ablaufen sollen.

**Wichtig:** Beachten Sie beim Schließen von Serverprozeduren über die **Datenbankmethode On Server Shutdown**, dass der Server abschaltet, wenn die Ausführung dieser Datenbankmethode (und nicht der Serverprozeduren) beendet ist. Noch laufende Serverprozeduren werden abgebrochen.

Wollen Sie sicherstellen, dass die Serverprozeduren vollständig ausgeführt sind, bevor der Server abschaltet, muss diese Datenbankmethode eine Anweisung zum Beenden geben, z.B. über Testen einer Interprozessvariablen und die Zeit zum Beenden gewähren. Dazu können Sie eine Schleife von n-Sekunden oder einen Test durch eine andere Interprozessvariable einrichten.

Um Code automatisch auf einem Client-Rechner auszuführen, wenn sich ein remote 4D am Server abmeldet, verwenden Sie die **Semaphore**.

## ⚙️ Datenbankmethode On Server Open Connection

\$1, \$2, \$3 -> Datenbankmethode On Server Open Connection -> \$0

Parameter	Typ	Beschreibung
\$1	Lange Ganzzahl	← ID Nummer für Benutzer, die 4D Server intern zum Identifizieren der Anwender verwendet.
\$2	Lange Ganzzahl	← ID Nummer für Verbindung, die 4D Server intern zum Identifizieren einer Verbindung verwendet.
\$3	Lange Ganzzahl	← Obsolet: Gibt immer 0 zurück, muss aber deklariert werden.
\$0	Lange Ganzzahl	➡ 0 oder weggelassen = Verbindung akzeptiert; anderer Wert = Verbindung verweigert

### Wann wird diese Datenbankmethode aufgerufen?

---

Die **Datenbankmethode On Server Open Connection** wird einmal auf dem Server-Rechner aufgerufen, wenn sich ein remote 4D anmeldet. Diese Datenbankmethode wird ausschließlich in der 4D Server Umgebung aufgerufen.

Die **Datenbankmethode On Server Open Connection** wird immer aufgerufen, wenn:

- Ein remote 4D sich anmeldet (weil der Prozess Anwendung startet)
- Ein remote 4D die Designumgebung öffnet (weil der Prozess Design startet)
- Ein remote 4D einen globalen Prozess startet (Name beginnt nicht mit \$), der das Erstellen eines kooperativen Prozesses auf dem Server erfordert (\*). Dieser Prozess lässt sich über die Funktion **New process**, über einen Menübefehl oder das Dialogfenster Methode ausführen erstellen.

Bei den Aktionen mit remote 4D starten verschiedene Prozesse — Einer auf dem Client-Rechner und ein bzw. bei Bedarf zwei weitere auf dem Server-Rechner. Auf dem Client-Rechner führt der Prozess Code aus und sendet Anfragen an 4D Server. Auf dem Server-Rechner verwaltet der **4D Client-Prozess** (präemptiver Prozess) die Datenbankumgebung für den Client-Prozess (z.B. aktuelle Auswahl und Datensatz sperren für Benutzerprozesse) und beantwortet Anfragen, die vom Prozess auf dem Client-Rechner gesendet wurden. Der **4D Client Datenbankprozess** (kooperativer Prozess) ist für die Überwachung des entsprechenden 4D Client Prozesses zuständig.

(\*) Ab 4D v13 werden die Server Prozesse (ein präemptiver Prozess für den Zugriff auf die Datenbank-Engine und ein kooperativer Prozess für den Zugriff auf die Programmiersprache) aus Optimierungsgründen nur erstellt, wenn sie beim Ausführen des Code auf Client-Seite notwendig sind. Hier ein Beispiel für eine 4D Code Sequenz, die in einem neuen Client Prozess läuft:

```
// globaler Prozess beginnt ohne einen neuen Prozess auf dem Server, wie ein lokaler  
Prozess.  
CREATE RECORD([Table_1])  
[Table_1])field1_1:="Hallo Welt"  
SAVE RECORD([Table_1]) // erstellt hier präemptiven Prozess auf Server
```

```
$serverTime:=Current time(*) // erstellt hier kooperativen Prozess auf Server
//Aufruf von On Server Open Connection
```

**Wichtig:** Web-Verbindungen und SQL-Verbindungen lösen nicht die **Datenbankmethode On Server Open Connection** aus. Meldet sich ein Web Browser an 4D Server an, startet die **Datenbankmethode On Web Authentication** - sofern vorhanden, und/oder die **Datenbankmethode On Web Connection**. Empfängt 4D Server eine SQL Anfrage, wird die **Datenbankmethode On SQL Authentication** - sofern vorhanden, aufgerufen.

**Wichtig:** Startet eine Serverprozedur, wird NICHT die **Datenbankmethode On Server Open Connection** ausgelöst. **Serverprozeduren** sind Server- und keine Client-Prozesse. Sie führen Code auf dem Server-Rechner aus, beantworten jedoch keine Anfragen von remote 4D oder anderen Clients an 4D Server.

## Wie wird diese Datenbankmethode aufgerufen?

Die **Datenbankmethode On Server Open Connection** wird auf dem 4D Server-Rechner innerhalb des 4D Client-Prozesses ausgeführt, der die Anweisung gibt, die Methode aufzurufen.

**Beispiel:** Meldet sich ein remote 4D an eine interpretierte Datenbank auf 4D Server an, starten standardmäßig die Prozesse Benutzer, Design und Registrierung für diesen Client. Die **Datenbankmethode On Server Open Connection** wird also dreimal ausgeführt — einmal innerhalb des Anwendungsprozesses, ein zweites Mal innerhalb des Client Registrierungsprozesses und ein drittes Mal innerhalb des Prozesses Design. Sind diese Prozesse jeweils der sechste, siebte und achte gestartete Prozess auf dem Server-Rechner und rufen Sie **Current process** innerhalb der **Datenbankmethode On Server Open Connection** auf, gibt die Funktion **Current process** jeweils 6, 7 und 8 zurück.

Beachten Sie, dass die **Datenbankmethode On Server Open Connection** auf dem Server-Rechner ausgeführt wird. Sie läuft im 4D Client Prozess ab, der auf dem Server läuft, unabhängig vom Prozess auf der Client-Seite. Außerdem ist der 4D Client Prozess in dem Moment, wo die Methode ausgelöst wird, noch nicht benannt. (Der Befehl **PROCESS PROPERTIES** gibt zu diesem Zeitpunkt nicht den Namen des 4D Client Prozesses zurück).

Die **Datenbankmethode On Server Open Connection** hat keinen Zugriff auf die Tabelle Prozessvariablen des Prozesses auf der Client-Seite, da diese Tabelle auf dem Client-Rechner und nicht auf dem Server-Rechner liegt.

Greift die **Datenbankmethode On Server Open Connection** auf eine Prozessvariable zu, läuft sie in einer eigenen und dynamisch erstellten Tabelle Prozessvariablen für den 4D Client Prozess ab.

4D Server übergibt der **Datenbankmethode On Server Open Connection** drei Parameter vom Typ Lange Ganzzahl und erwartet ein Ergebnis vom Typ Lange Ganzzahl. Die Methode muss deshalb ausdrücklich mit drei Parametern vom Typ Lange Ganzzahl und mit dem Funktionsergebnis vom Typ Lange Ganzzahl deklariert werden:

```
C_LONGINT($0;$1;$2;$3)
```

Geben Sie keinen Wert in *\$0* zurück, d.h. die Variable ist undefiniert oder gegen Null initialisiert, nimmt 4D Server an, dass die Datenbankmethode die Verbindung

akzeptiert. Akzeptieren Sie die Verbindung nicht, geben Sie in  $\$0$  einen Wert zurück, der nicht Null ist.

Die drei Parameter für die Datenbankmethode geben folgende Informationen zurück:

Parameter	Beschreibung
$\$1$	ID Nummer für Benutzer, die 4D Server intern zur Identifikation von Benutzern verwendet.
$\$2$	ID Nummer für Verbindung, die 4D Server intern zur Identifikation einer Verbindung verwendet.
$\$3$	Obsolet: Gibt immer 0 zurück, muss aber deklariert werden.

Diese ID Nummern sind nicht direkt als Informationsquellen einsetzbar, z.B. um sie in einem 4D Befehl als Parameter zu übergeben. Sie ermöglichen jedoch, einen 4D Client Prozess zwischen der **Datenbankmethode On Server Open Connection** und der **Datenbankmethode On Server Close Connection** eindeutig zu identifizieren. Die Kombination dieser Werte ist zu jedem Moment einer Server Sitzung einmalig. Ist diese Information in einer Interprozess-Array bzw. Tabelle gespeichert, können beide Datenbankmethoden Informationen austauschen. Im Beispiel am Ende dieses Abschnitts verwenden die beiden Datenbankmethoden diese Information, um Datum und Uhrzeit für Beginn und Ende einer Verbindung zum gleichen Datensatz einer Tabelle zu speichern.

## Beispiel 1

---

Dieses Beispiel zeigt, wie mit der **Datenbankmethode On Server Open Connection** und der **Datenbankmethode On Server Close Connection** in der Datenbank ein Logbuch der Verbindung verwaltet wird. Die Tabelle [Server Log] wird zum Auffinden der Verbindungsprozesse verwendet.



Server_Log	
Log_Nr	2 <sup>32</sup>
Anmeldedatum	17
Anmeldezeit	
Abmeldedatum	17
Abmeldezeit	
Benutzer_Nr	2 <sup>32</sup>
Verbindung_Nr	2 <sup>32</sup>
Prozess_Nr	2 <sup>32</sup>
Prozessname	A

Die hier gespeicherte Information wird von der **Datenbankmethode On Server Open Connection** und der **Datenbankmethode On Server Close Connection** wie folgt verwaltet:

```
  ` Datenbankmethode On Server Open Connection
C_LONGINT( $\$0$ ; $\$1$ ; $\$2$ ; $\$3$ )
  ` Erstelle eine Datensatz [Logbuch]
CREATE RECORD([Server_Log])
[Server_Log]Log_Nr:=Sequence number([Server_Log])
  ` Sichere Anmeldedatum und Anmeldezeit
[Server_Log]Anmeldedatum:=Current date
[Server_Log]Anmeldezeit:=Current time
```

‣ Sichere die Verbindungsinformation

[Server\_Log]Benutzer\_Nr:=\$1

[Server\_Log]Verbindung\_Nr:=\$2

**SAVE RECORD**([Server\_Log])

‣ Gibt keinen Fehler zurück, so dass die Verbindung weiterlaufen kann

**\$0:=0**

‣ Datenbankmethode On Server Close Connection

**C\_LONGINT**(\$1;\$2;\$3)

‣ Finde wieder Datensatz [Server\_Log]

**QUERY**([Server\_Log];[Server\_Log]Benutzer\_Nr=\$1;\*)

**QUERY**([Server\_Log]; & ;[Server\_Log]Verbindung\_Nr=\$2;\*)

**QUERY**([Server\_Log]; & ;[Server\_Log]Prozess\_Nr=0)

‣ Sichere Abmeldedatum und -zeit

[Server\_Log]Abmeldedatum:=**Current date**

[Server\_Log]Abmeldezeit:=**Current time**

‣ Sichere die Prozessinformation

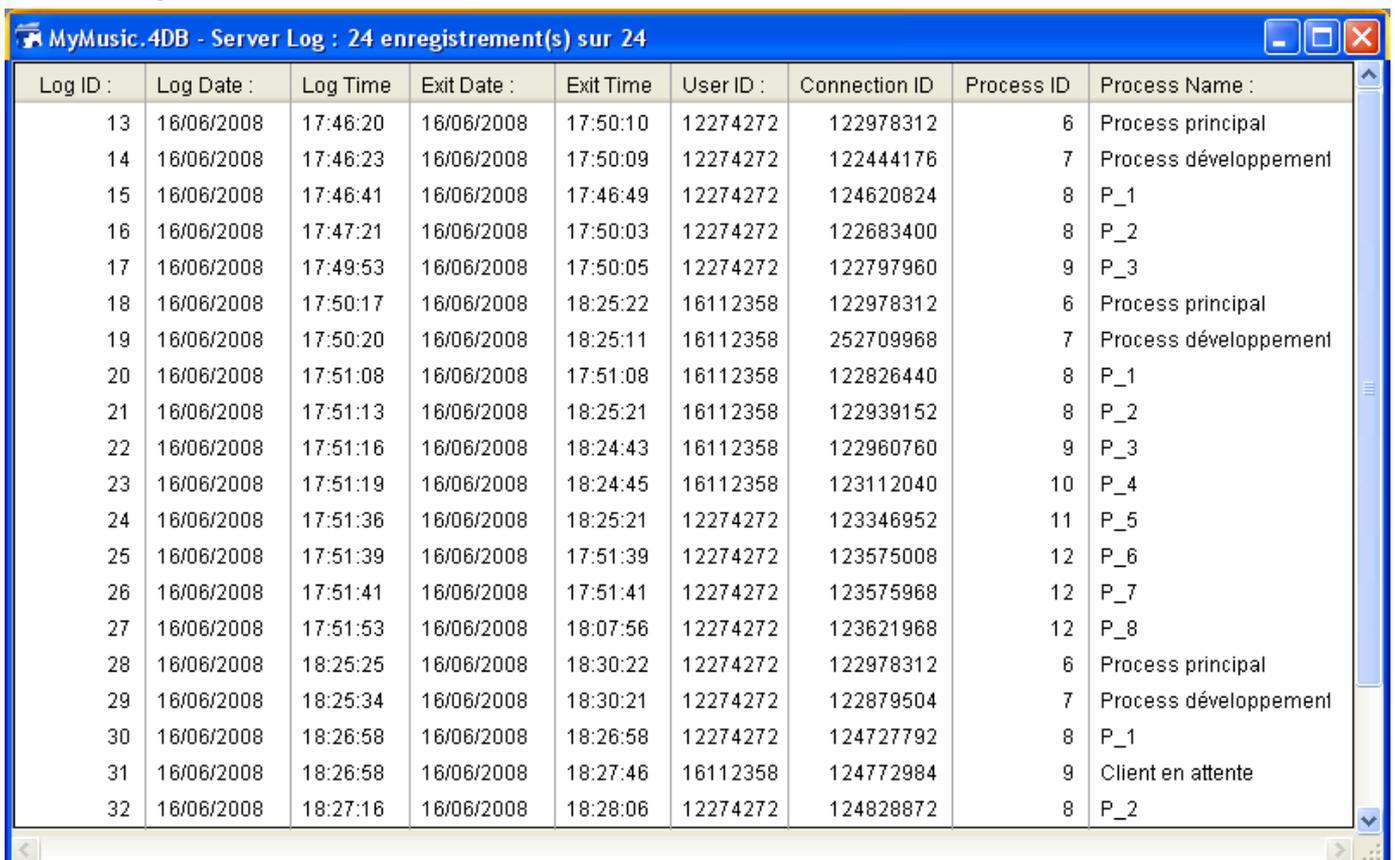
[Server\_Log]Prozess\_Nr:=**Current process**

**PROCESS PROPERTIES**([Server\_Log]Prozess\_Nr;\$vsProzName;\$vlProzStatus;\$vlProzZeit)

[Server\_Log]Prozessname:=\$vsProzName

**SAVE RECORD**([Server\_Log])

Nachfolgend sehen Sie einige Einträge in [Server\_Log] mit mehreren remote Anbindungen:



Log ID :	Log Date :	Log Time	Exit Date :	Exit Time	User ID :	Connection ID	Process ID	Process Name :
13	16/06/2008	17:46:20	16/06/2008	17:50:10	12274272	122978312	6	Process principal
14	16/06/2008	17:46:23	16/06/2008	17:50:09	12274272	122444176	7	Process développement
15	16/06/2008	17:46:41	16/06/2008	17:46:49	12274272	124620824	8	P_1
16	16/06/2008	17:47:21	16/06/2008	17:50:03	12274272	122683400	8	P_2
17	16/06/2008	17:49:53	16/06/2008	17:50:05	12274272	122797960	9	P_3
18	16/06/2008	17:50:17	16/06/2008	18:25:22	16112358	122978312	6	Process principal
19	16/06/2008	17:50:20	16/06/2008	18:25:11	16112358	252709968	7	Process développement
20	16/06/2008	17:51:08	16/06/2008	17:51:08	16112358	122826440	8	P_1
21	16/06/2008	17:51:13	16/06/2008	18:25:21	16112358	122939152	8	P_2
22	16/06/2008	17:51:16	16/06/2008	18:24:43	16112358	122960760	9	P_3
23	16/06/2008	17:51:19	16/06/2008	18:24:45	16112358	123112040	10	P_4
24	16/06/2008	17:51:36	16/06/2008	18:25:21	12274272	123346952	11	P_5
25	16/06/2008	17:51:39	16/06/2008	17:51:39	12274272	123575008	12	P_6
26	16/06/2008	17:51:41	16/06/2008	17:51:41	12274272	123575968	12	P_7
27	16/06/2008	17:51:53	16/06/2008	18:07:56	12274272	123621968	12	P_8
28	16/06/2008	18:25:25	16/06/2008	18:30:22	12274272	122978312	6	Process principal
29	16/06/2008	18:25:34	16/06/2008	18:30:21	12274272	122879504	7	Process développement
30	16/06/2008	18:26:58	16/06/2008	18:26:58	12274272	124727792	8	P_1
31	16/06/2008	18:26:58	16/06/2008	18:27:46	16112358	124772984	9	Client en attente
32	16/06/2008	18:27:16	16/06/2008	18:28:06	12274272	124828872	8	P_2

## Beispiel 2

Im folgenden Beispiel kann zwischen 2 und 4 Uhr a.m. keine neue Verbindung hergestellt werden:

```
` Datenbankmethode On Server Open Connection
```

```
C_LONGINT($0;$1;$2;$3)
```

```
If((?02:00:00?<=Current time)&(Current time<?04:00:00?))
```

```
    $0:=22000
```

```
Else
```

```
    $0:=0
```

```
End if
```

## ⚙️ Datenbankmethode On Server Close Connection

\$1, \$2, \$3 -> Datenbankmethode On Server Close Connection

Parameter	Typ	Beschreibung
\$1	Lange Ganzzahl	← ID Nummer für Benutzer, die 4D Server intern zur Identifikation des Benutzers verwendet.
\$2	Lange Ganzzahl	← ID Nummer für Verbindung, die 4D Server intern zur Identifikation einer Verbindung verwendet.
\$3	Lange Ganzzahl	← Obsolet: Gibt immer 0 zurück, muss aber deklariert werden.

### Beschreibung

---

Die **Datenbankmethode On Server Close Connection** wird jedes Mal auf dem Server-Rechner aufgerufen, wenn ein 4D Client Prozess endet.

Analog zur **Datenbankmethode On Server Open Connection** übergibt 4D Server in der **Datenbankmethode On Server Close Connection** drei Parameter vom Typ Lange Ganzzahl, er erwartet jedoch kein Ergebnis.

Die Methode muss deshalb ausdrücklich mit drei Parametern vom Typ Lange Ganzzahl definiert werden:

```
C_LONGINT($1;$2;$3)
```

Die drei Parameter für die Datenbankmethode geben folgende Informationen zurück:

Parameter	Beschreibung
\$1	ID Nummer für Benutzer, die 4D Server intern zur Identifikation des Benutzers verwendet.
\$2	ID Nummer für Verbindung, die 4D Server intern zur Identifikation einer Verbindung verwendet.
\$3	Obsolet: Gibt immer 0 zurück, muss aber deklariert werden.

Die **Datenbankmethode On Server Close Connection** ist das exakte Gegenstück zur **Datenbankmethode On Server Open Connection**. Weitere Informationen und eine Beschreibung der remote 4D Prozesse finden Sie in der Beschreibung dieser Datenbankmethode.

### Beispiel

---

Siehe erstes Beispiel unter der **Datenbankmethode On Server Open Connection**.

# **Remote 4D verwenden**

-  Anmeldung an eine 4D Server Datenbank
-  Administration über Remote Rechner
-  Kompilieren über Remote Rechner

## Anmeldung an eine 4D Server Datenbank

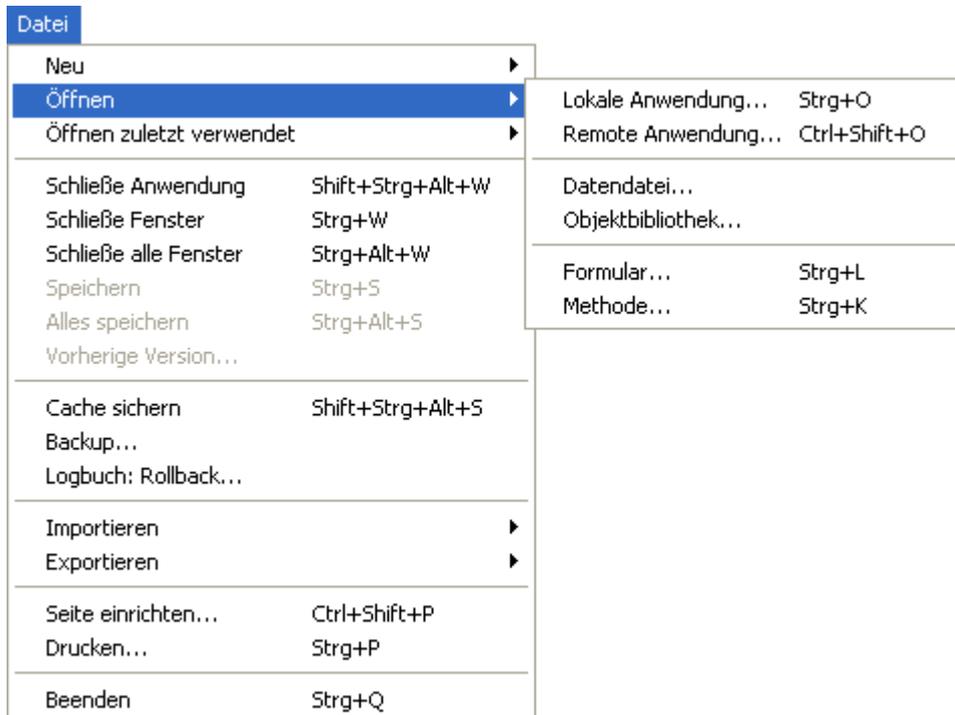
Es gibt drei Wege, sich über remote 4D an eine 4D Server Datenbank anzumelden:

- Über den Anmeldedialog
- Über den Befehl **Öffnen zuletzt verwendet**
- Über eine Verknüpfungsdatei 4D Link mit den Zugriffparametern für die Datenbank.

### Über den Anmeldedialog

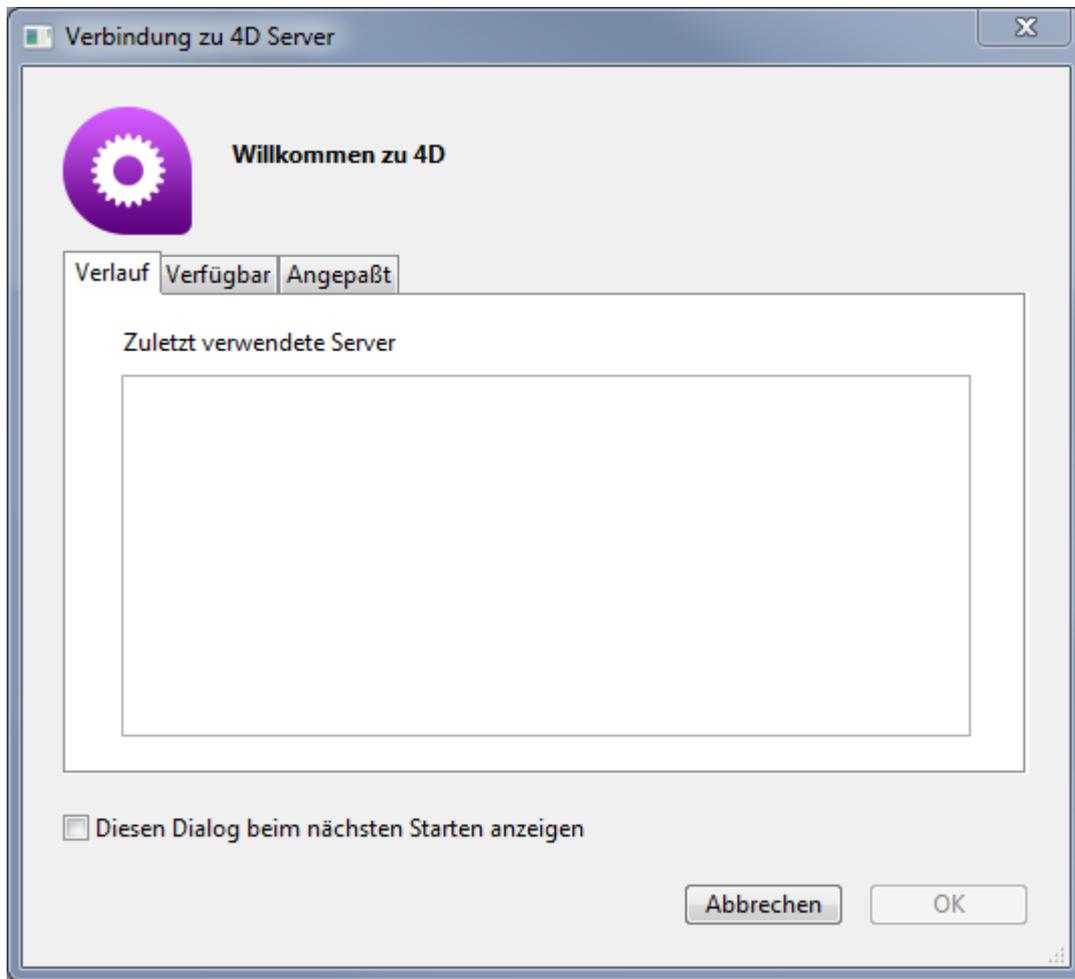
Um den Anmeldedialog für 4D Server anzuzeigen, starten Sie zuerst die 4D Anwendung.

Über das Menü **Datei/Ablage>Öffnen** bzw. die entsprechende Schaltfläche in der 4D Toolbar können Sie die Art zum Öffnen der 4D Anwendung auswählen:



Wählen Sie den Befehl **Öffnen>Remote Anwendung**.

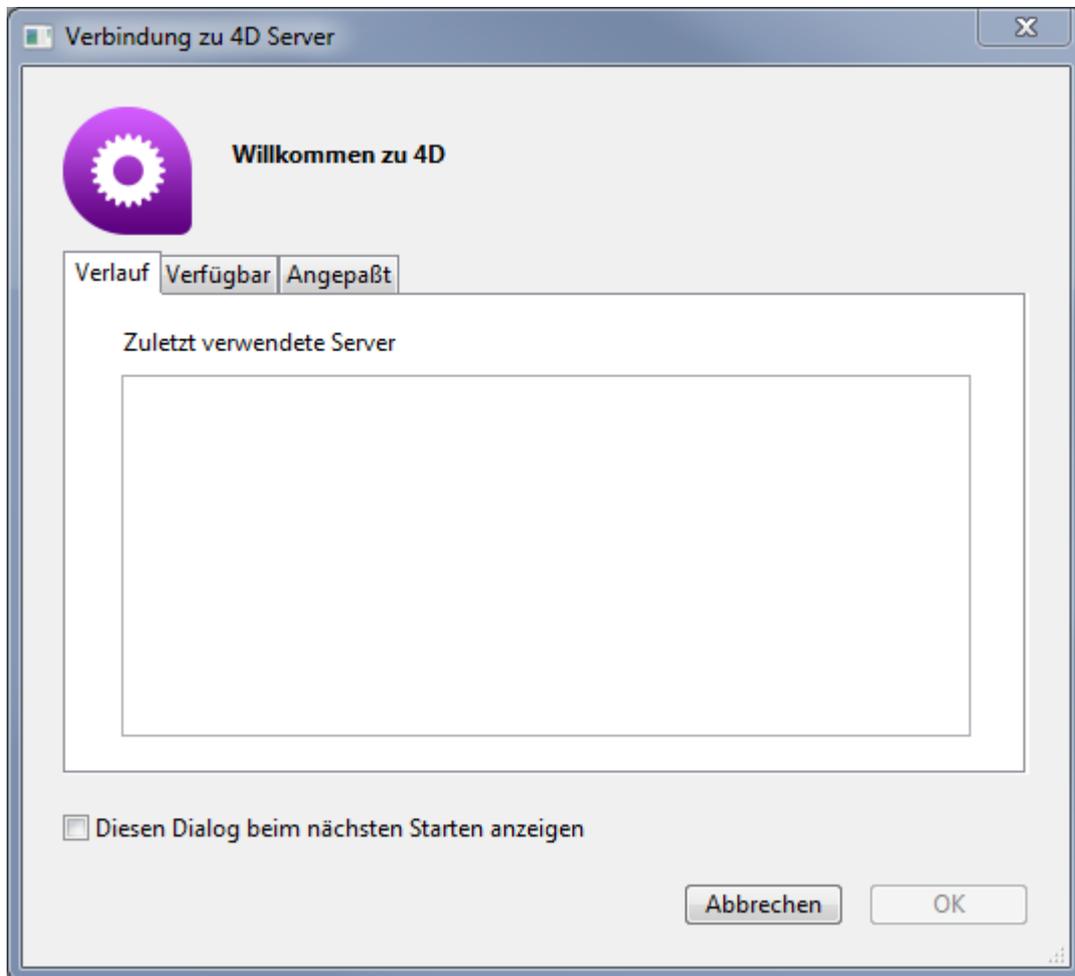
Auf dem Bildschirm erscheint der Anmeldedialog für 4D Server. Er enthält 3 Registerkarten, über die Sie auf folgende Seiten zugreifen können: **Verlauf**, **Verfügbar** und **Angepasst**.



Markieren Sie die Option **Diesen Dialog beim nächsten Starten anzeigen**, erscheint das Fenster automatisch beim Starten der 4D Anwendung.

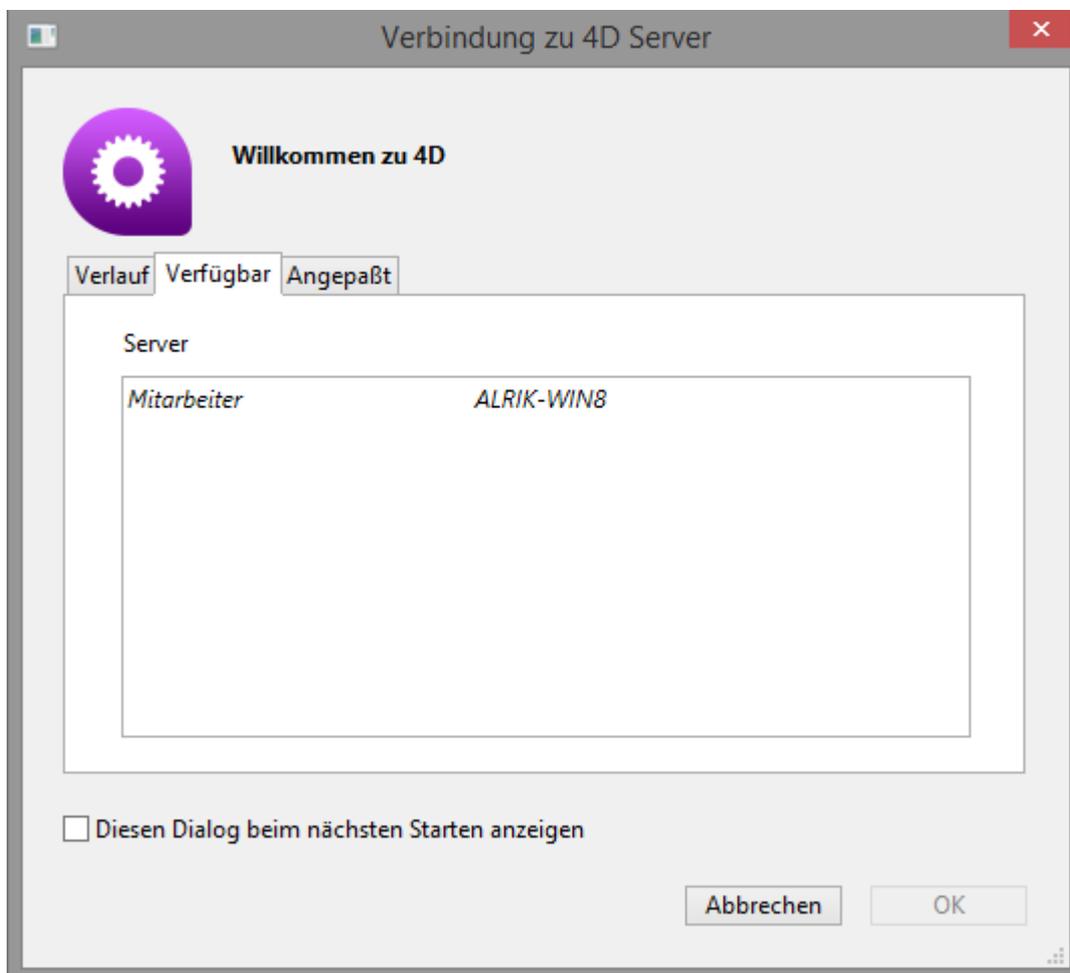
**Hinweis:** Sie können dieses Dialogfenster auch öffnen, wenn Sie im 4D Willkommen-Assistent auf den Link **Anmelden an 4D Server** klicken.

## Seite Verlauf



Die Seite **Verlauf** zeigt die Liste der zuletzt verwendeten 4D Server in alphabetischer Reihenfolge. Um sich an einen Server in der Liste anzumelden, doppelklicken Sie auf seinen Namen oder markieren ihn und klicken auf die Schaltfläche **OK**.

## **Seite Verfügbar**



4D Server enthält ein integriertes System, das automatisch die Namen der 4D Server Datenbanken anzeigt, die über das Netzwerk verfügbar sind. Die Namen erscheinen im Verbindungsdialog auf der Seite **Verfügbar**.

Die Liste ist in der Reihenfolge der Veröffentlichung sortiert und wird dynamisch aktualisiert. Wollen Sie sich über diese Liste an einen Server anmelden, doppelklicken Sie auf seinen Namen oder markieren ihn und klicken auf die Schaltfläche **OK**.

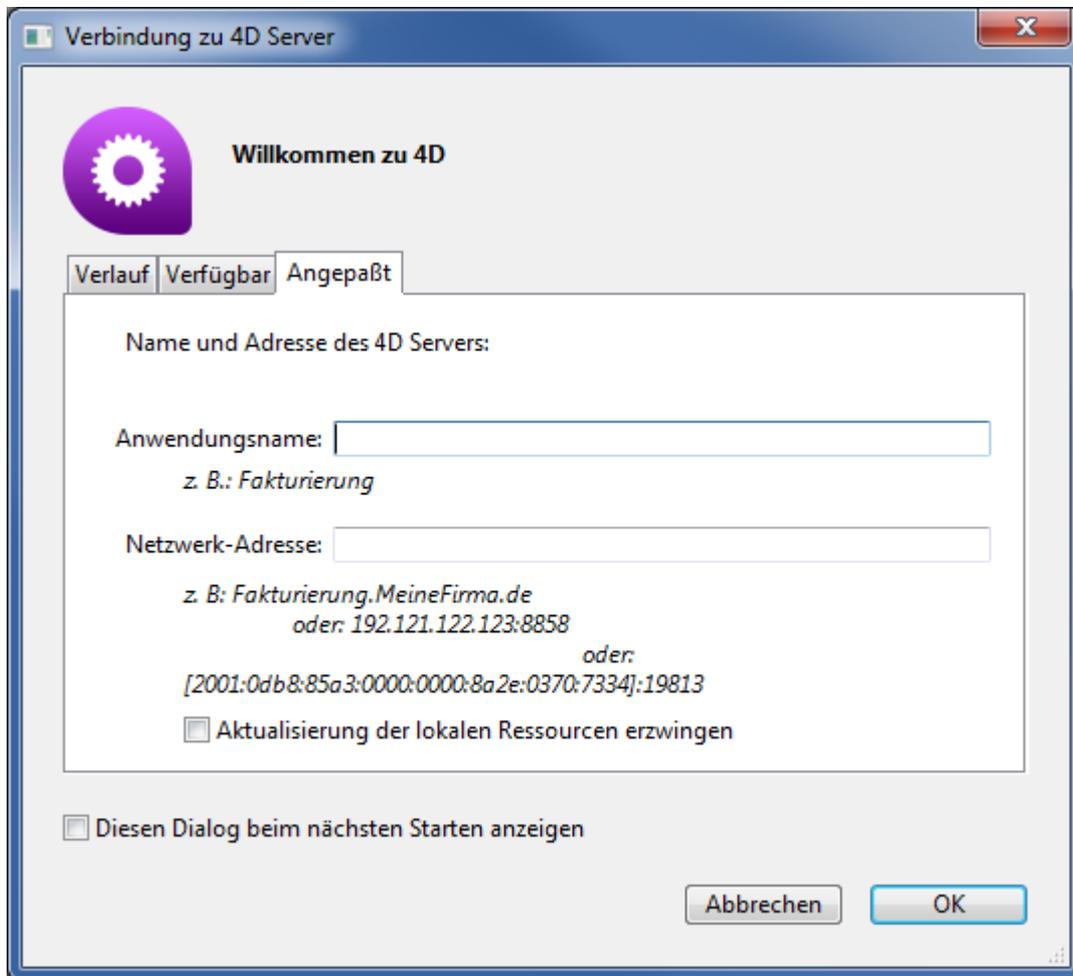
#### **Hinweise:**

- Ein Circumflex (^) vor dem Namen der Datenbanken gibt an, dass sie verschlüsselt publiziert werden. Weitere Informationen dazu finden Sie im *4D Server Handbuch* im Abschnitt **Client/Server Verbindungen verschlüsseln**.
- Es ist möglich, den Datenbanknamen im Netzwerk nicht zu veröffentlichen. Dazu deaktivieren Sie auf dem Server unter **Datenbank-Eigenschaften > Netzwerk** die Option **Beim Starten veröffentlichen**. Dann lässt sich die Verbindung nur manuell über die Seite "Angepasst" herstellen.

**Hinweis zur Kompatibilität:** Verwenden Sie IPv4, sind auf der Seite **Verfügbar** nur auf dem Standard Port (19813) veröffentlichte Datenbanken sichtbar. Ist IPv6 aktiviert (*siehe Unterstützung von IPv6*), sind auf der Seite **Verfügbar** alle im Netzwerk veröffentlichten 4D Server Datenbanken sichtbar. Soll eine Datenbank nicht erscheinen, müssen Sie die Option zum Veröffentlichen deaktivieren (siehe Hinweis oben).

#### **Seite Angepasst**

Auf dieser Seite können Sie einen im Netzwerk veröffentlichten Server über seine IP Adresse zuweisen und einen eigenen Namen vergeben:



Sie können 4D Server so anpassen, dass die Namen der Datenbanken nicht automatisch über das Netzwerk publiziert werden (siehe Abschnitt **Beim Starten veröffentlichen**). In diesem Fall erscheinen die Namen nicht auf der Seite **Verfügbar** und Sie können dann auf der Seite **Angepasst** die IP-Adresse manuell eingeben.

- **Anwendungsname:** Hier definieren Sie den Namen der 4D Server Datenbank. Dieser Name wird auf der Seite **Verlauf** verwendet, wenn Sie Bezug auf die Datenbank nehmen.
- **Netzwerk-Adresse:** Hier geben Sie die Netzwerk-Adresse des Rechners an, auf dem der 4D Server gestartet wurde. Der Port zum Veröffentlichen eines Servers ist standardmäßig 19813. In diesem Fall müssen Sie ihn in der Netzwerkadresse nicht angeben. Jedoch kann sich der Port, insbesondere wenn auf einem Rechner zwei Server gleichzeitig laufen, in den Datenbank-Eigenschaften geändert haben (siehe Abschnitt **Netzwerk und Client-Server Optionen**). In diesem Fall müssen an die IP Adresse ein Doppelpunkt und eine Port-Nummer angehängt werden, zum Beispiel 192.168.92.104:19814 (IPv4 Format) oder [2001:0db8:0000:85a3:0000:0000:ac1f:8001]:19814 (IPv6 Format)..

**Hinweis:** Ist auf den Seiten **Verlauf** oder **Verfügbar** eine Datenbank ausgewählt, wenn Sie auf die Registerkarte **Angepasst** klicken, enthalten die beiden Felder die entsprechende Information.

Sobald diese Seite einen Server zuweist, können Sie auf die Schaltfläche **OK** klicken, um sich an den Server anzumelden. Der Server erscheint dann auf der Seite **Verlauf**.

**Hinweis:** Wird die Datenbank verschlüsselt veröffentlicht, müssen Sie vor den Namen ein Circumflex (^) setzen. Andernfalls wird die Anmeldung zurückgewiesen.

Weitere Informationen dazu finden Sie im *4D Server Handbuch* im Abschnitt **Client/Server Verbindungen verschlüsseln**.

## Aktualisieren der lokalen Ressourcen erzwingen

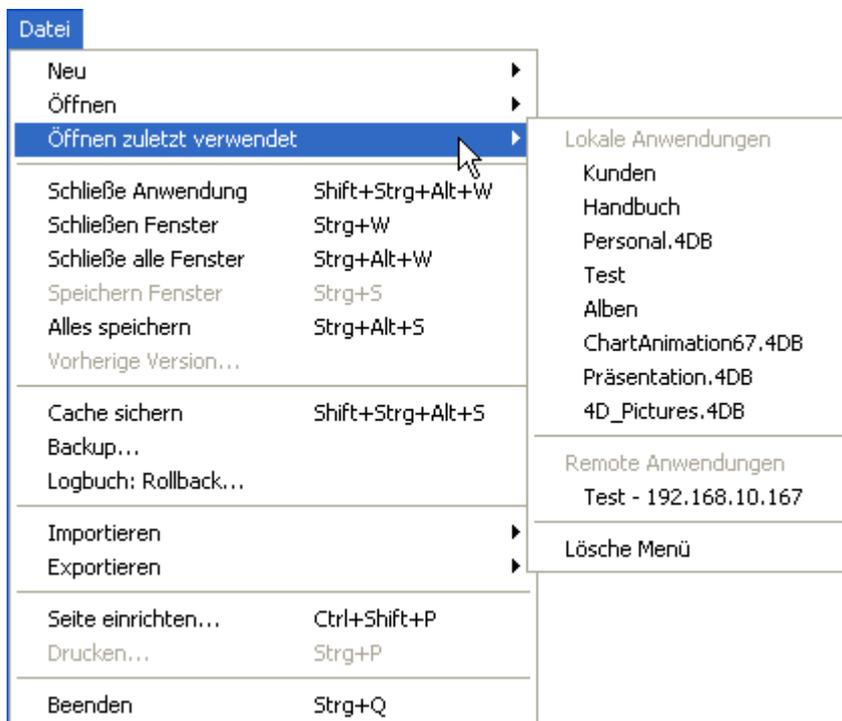
Ist diese Option markiert, werden die lokalen Ressourcen auf dem Client-Rechner beim Anmelden systematisch aktualisiert. Das sind Informationen zur Struktur der Datenbank, die auf jedem Client-Rechner gespeichert sind.

In der Regel werden die lokalen Ressourcen auf dem remote Rechner automatisch bei jeder Anmeldung aktualisiert, wenn die Struktur der Datenbank geändert wurde. In den meisten Fällen ist diese Option nicht erforderlich. In bestimmten Fällen kann es jedoch erforderlich sein, die Aktualisierung zu erzwingen.

## Über den Befehl Öffnen zuletzt verwendet

---

Über das Menü **Datei/Ablage>Öffnen zuletzt verwendet** können Sie sich direkt an eine 4D Server Datenbank anmelden, die sie zuvor schon verwendet haben. Verwenden Sie die 4D Anwendung zum Öffnen lokaler und zum Anmelden an remote Anwendungen, zeigt das Untermenü beide Arten an. Die remote Anwendungen erscheinen im unteren Teil:



Die IP-Adresse des Server erscheint neben dem Namen der Anwendung. Über den Befehl **Lösche Menü** können Sie die Einträge entfernen.

## Über die Verknüpfungsdatei 4DLink

---

Sie können Zugriffsdateien für Datenbanken anlegen. Sie enthalten Parameter, um das Öffnen oder Anmelden an 4D Datenbanken zu automatisieren und zu vereinfachen. Eine Zugriffsdatei kann in der Regel die Adresse einer remote Anwendung sowie die Anmelde-Identifikation sichern, wodurch für den Benutzer mehrere Operationen wegfallen.

Über diese Zugriffsdateien können Sie entweder eine lokale Datenbank öffnen oder sich an eine von 4D Server veröffentlichte Datenbank anmelden.

## Dateien verwenden

Über eine Zugriffsdatei .4DLink können Sie das 4D Programm starten und die 4D Zieldatenbank öffnen. Es gibt folgende Möglichkeiten:

- Doppelklick oder Drag&Drop auf das 4D Programm,
- Über das Untermenü **Zuletzt verwendet** (Datei im lokalen Ordner Preferences).  
Eine Datei .4DLink vom Typ "remote Anwendung" können Sie kopieren und auf verschiedenen Rechnern verwenden.

**Hinweis:** Sie können eine Datei .4DLink auch im Öffnen-Dialog von 4D und 4D Server auswählen (nur beim Öffnen lokaler Anwendungen).

## Zugriffsdatei einrichten

Die Zugriffsdateien von 4D Anwendungen sind XML Dateien mit der Endung ".4DLink". 4D erstellt und verwendet diesen Dateityp zum Einrichten des Untermenüs **Zuletzt verwendet**: 4D erstellt automatisch die Datei .4DLink, wenn eine Datenbank zum ersten Mal auf dem Rechner geöffnet wird oder beim ersten Anmelden an den Server.

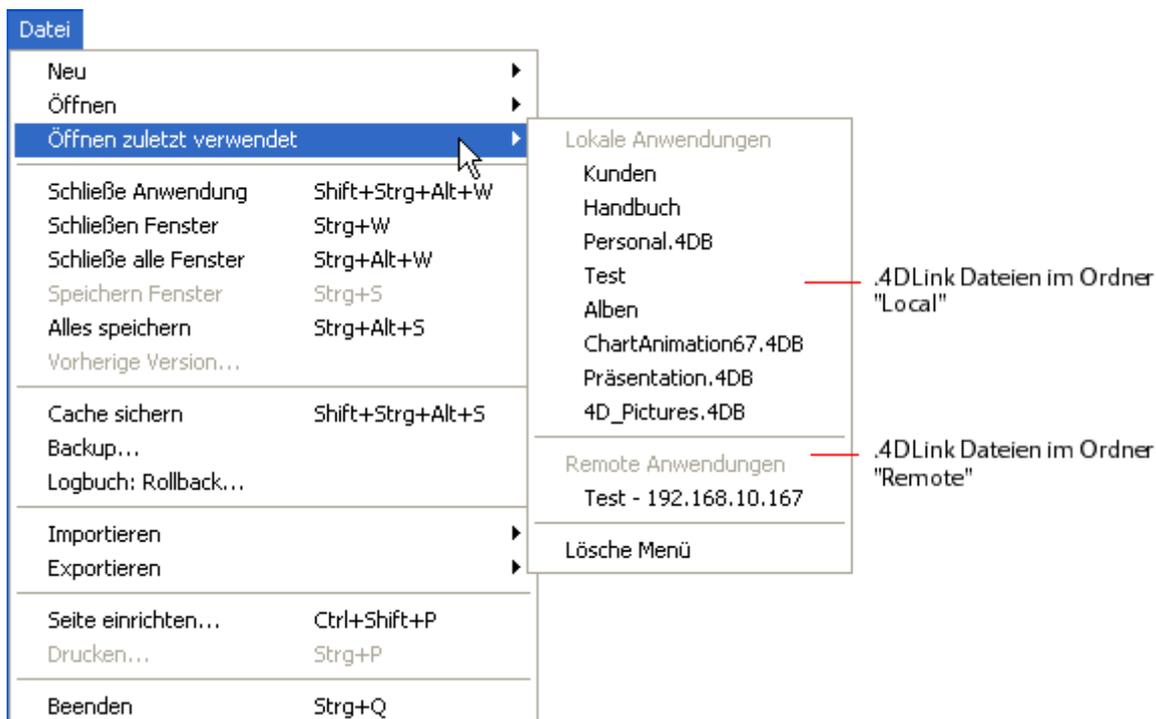
Die automatisch angelegten Dateien .4DLink werden im lokalen Ordner Preferences des Benutzers abgelegt. Es werden zwei Verzeichnisse angelegt: Local und Remote. Der Ordner Local enthält die Dateien ".4DLink" zum Anmelden an Datenbanken auf dem Rechner, der Ordner Remote die Dateien ".4DLink" zum Anmelden an remote Datenbanken.

Die lokalen Ordner Preferences liegen an folgender Stelle:

- Windows 7 und höher: C:\Users\UserName\AppData\Roaming\4D\Favorites vXX\  
vXX\
- Mac OS: Users/UserName/Library/Preferences/4D/Favorites vXX

... XX ist die Versionsnummer des Programms (zum Beispiel "Favorites v14" für 4D v14).

4D zeigt die hier gefundenen Dateien im Menü **Datei/Ablage** im Untermenü **Öffnen zuletzt verwendet** an:



Die Dateien ".4DLink" lassen sich auch mit einem XML Editor erstellen. Sie können auch angepasste Information, wie die Anmelde-Identifikation oder den Öffnungsmodus der Datenbank enthalten.

4D liefert ein DTD mit der Beschreibung der XML Keys, mit der Sie eine ".4DLink" Datei einrichten können. Die DTD lautet `database_link.dtd` und liegt im Unterordner `\Resources\DTD\` des 4D Programms.

## Administration über Remote Rechner

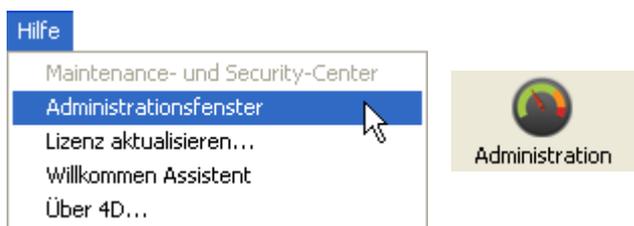
Sie können den 4D Server Rechner über ein remote 4D, d.h. über einen Client Rechner verwalten. Dazu müssen Sie nur das 4D Server Verwaltungsfenster auf dem Client-Rechner öffnen (siehe [Seite Monitor](#)).

### Das Verwaltungsfenster auf einem remote 4D Rechner öffnen

Um den 4D Server Rechner über ein remote 4D zu verwalten, müssen Sie in der remote Anwendung als **Designer** oder **Administrator** angemeldet sein. Andernfalls erscheint der Fehler -9991 - Keine Zugriffsrechte.

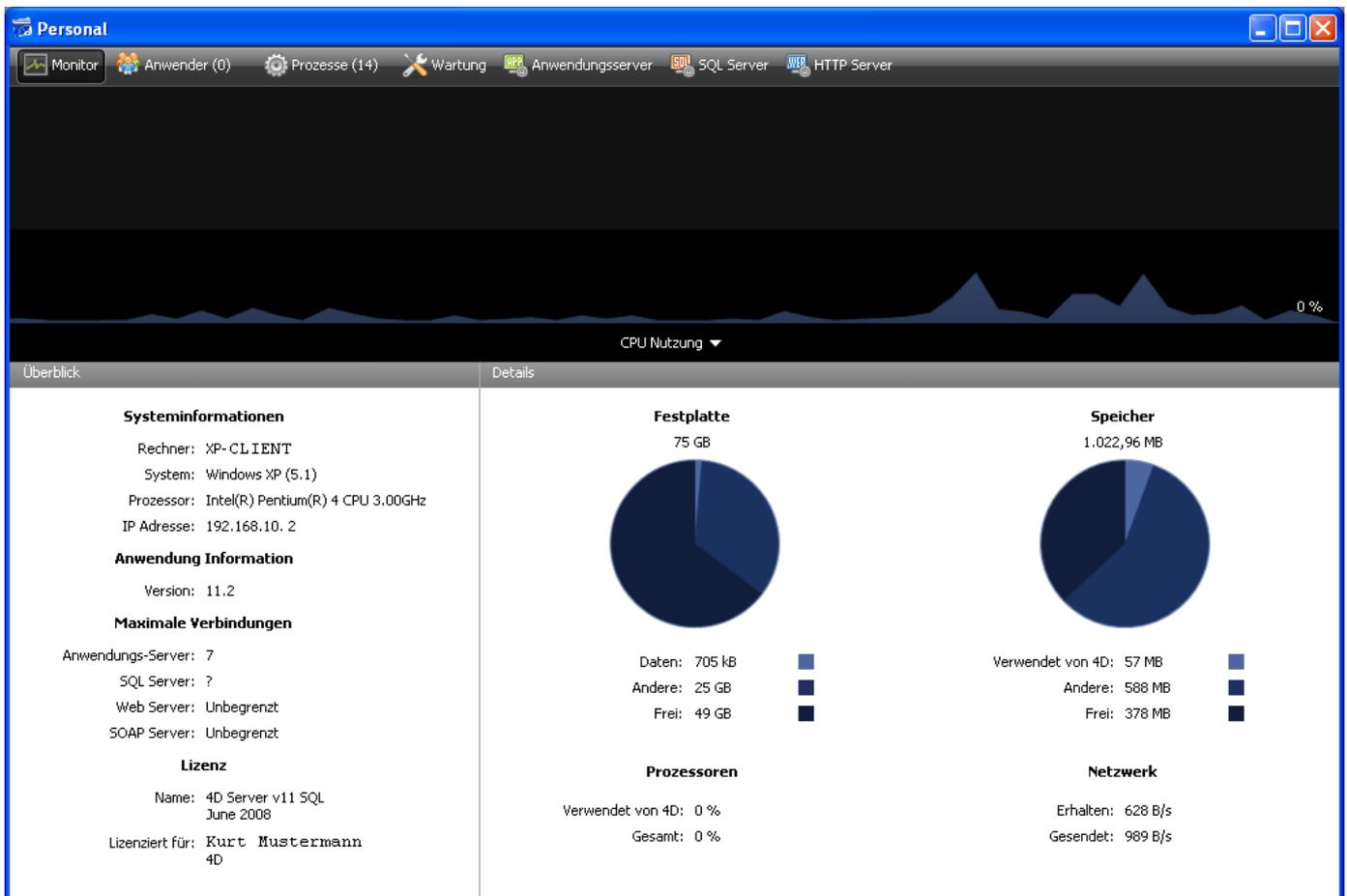
Zum Öffnen gibt es folgende Möglichkeiten:

- Wählen Sie im Menü **Hilfe** den Befehl **Administrationsfenster** oder klicken Sie in der 4D Werkzeugleiste auf die entsprechende Schaltfläche:



- Führen Sie den Befehl **OPEN ADMINISTRATION WINDOW** aus.

Auf dem Bildschirm des Client-Rechners erscheint das Server Verwaltungsfenster:



## Besonderheiten beim Verwalten über einen Client

Ein Client-Rechner, der dieses Fenster anzeigt, hat Zugriff auf alle verfügbaren Informationen. Er kann auf die Prozesse einwirken und die verschiedenen Server starten bzw. stoppen. Es gibt jedoch ein paar Einschränkungen und spezifische Vorgehensweisen:

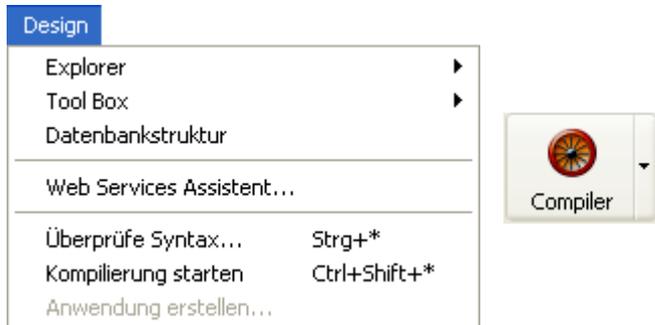
- Auf der **Seite Prozess** lässt sich ein Benutzerprozess nicht debuggen, da das Debugger-Fenster auf dem Server-Rechner angezeigt wird.
- Auf der **Seite Wartung** lassen sich Aktionen, wie Komprimieren und Neustart ausführen, die alle Clients abkoppeln und den Server neu starten. In diesem Fall wird der ausführende Client bei Neustart automatisch wieder angeschlossen.
- Auf der **Seite Wartung** lauten die Schaltflächen **Logbuch anzeigen** nach Ausführen einer Wartungsoperation **Logbuch laden**. Diese Dateien werden in den Anwendungsordner des Client-Rechners geladen, bevor sie angezeigt werden.

## Kompilieren über Remote Rechner

Eine 4D Anwendung lässt sich über eine remote Verbindung kompilieren, also über einen Client-Rechner. Das war vor 4D Server v11 SQL nur mit der 4D Einzelplatz-Version möglich.

**Hinweis:** Über eine remote Verbindung lässt sich jedoch keine eigene 4D Anwendung (Einzelplatz oder Client/Server) erstellen. Der Application Builder ist in dieser Umgebung nicht verfügbar, der Menübefehl erscheint in Grauschrift.

Die Oberfläche und Vorgehensweise zum Kompilieren auf der Client-Seite sind dieselben wie im Einzelplatz. Der Compiler lässt sich über das Menü **Design** oder das entsprechende Icon in der Werkzeugleiste aufrufen:



**Hinweis:** Damit die Client-Rechner die Funktionen zum Kompilieren nutzen können, ist auf 4D Server eine "4D Team Server" Lizenz erforderlich.

Es kann immer nur ein Client-Rechner kompilieren. Die Aktion ist dann für andere Client-Rechner gesperrt. Versucht ein anderer Client Rechner, zur gleichen Zeit zu kompilieren, erhält er eine Meldung.

Während die Kompilierung auf einem Client-Rechner läuft, können die anderen Clients weiterarbeiten und Methoden oder andere Strukturelemente ändern. Kompilierter und interpretierter Code sind dann unterschiedlich, d.h. die Anwendung muss erneut kompiliert werden.

Der kompilierte Code wird während der Aktion schrittweise an die Datei .4DB auf dem Server gesendet.

Nach Abschluss der Aktion lässt sich der Server auf dem Client im interpretierten oder kompilierten Modus starten. Wählen Sie dazu im Menü **Start** den entsprechenden Befehl. Auf dem Bildschirm erscheint das Abmeldefenster des Server, um eine Wartezeit einzugeben oder eine Meldung an die anderen Clients zu senden (siehe Abschnitt **4D Server beenden**). Wurde der Server neu gestartet, wird der Client, der die Aktion auslöst, automatisch wieder angemeldet.

Auf dem Server muss zum Neustart im kompilierten/interpretierten Modus der Standard-Öffnen Dialog verwendet werden (PopUp-Menü zur Schaltfläche **Öffnen**).

**Hinweis:** Zur Wahrung der Performance ist die Kompilierung des WAN Netzwerks nicht zu empfehlen, besonders bei Anwendungen mit zahlreichen Methoden. Denn das würde zu umfangreichem Netzwerkverkehr führen.

# **4D Server und Programmiersprache**

-  Einführung
-  4D Server, Mengen und temporäre Auswahlen
-  Serverprozeduren
-  Serverprozeduren auf Client-Rechnern
-  Import mit Serverprozeduren (Beispiel)
-  Service mit Serverprozeduren (Beispiel)
-  Attribut Auf Server ausführen

4D Server bietet vier Möglichkeiten, den 4D Code auf dem Server-Rechner auszuführen:

- Trigger
- Serverprozeduren
- Methoden mit dem Attribut "Auf Server ausführen"
- Datenbankmethoden

### Trigger

---

Trigger sind Tabellenereignisse, die ablaufen, wenn bestimmte Ereignisse auftreten, die das Sichern, Löschen und Laden von Datensätzen betreffen. Sie unterbinden unzulässige Operationen in den Datensätzen und schützen vor unabsichtlichem Löschen. In einem Rechnungssystem kann ein Trigger zum Beispiel dafür sorgen, dass eine Rechnung erst dann vollständig ist, wenn auch der dazugehörige Kunde eingetragen wurde.

Trigger werden auf dem Rechner ausgeführt, der die Datenbank-Engine enthält. Auf 4D Server werden Trigger im Kontext von Prozessen ausgeführt, die auf dem Server-Rechner laufen, und nicht auf dem Client-Rechner. Sie werden insbesondere im Kontext der "Zwillingsprozesse" von Benutzerprozessen ausgeführt, welche die Datenbankoperation aufrufen. Diese Zwillingsprozesse nutzen den Datenbankkontext gemeinsam mit dem Benutzerprozess auf dem Client-Rechner (im einzelnen den Status von Transaktionen und das Sperren von Datensätzen), sie nutzen dagegen nicht gemeinsam den Programmiersprachekontext (Variablen, Prozesse, Mengen, aktuelle Auswahlen) Beachten Sie jedoch, dass der aktuelle Datensatz der Tabelle des Trigger in allen Kontexten gleich ist.

Weitere Informationen dazu finden Sie im Handbuch *4D Programmiersprache* im Abschnitt **ARRAY REAL**.

### Serverprozeduren

---

4D Serverprozeduren sind Projektmethoden, die Methoden in Prozessen ausführen, die auf dem Server-Rechner oder auf jedem Client-Rechner laufen, anstatt auf dem Client-Rechner, der die Methode gestartet hat. Weitere Informationen dazu finden Sie im Abschnitt **Serverprozeduren**.

### Methoden mit dem Attribut "Auf Server ausführen"

---

Projektmethoden mit dem Attribut "Auf Server ausführen" werden ebenfalls auf dem Server ausgeführt. Sie werden jedoch im Gegensatz zu Serverprozeduren im Zwillingsprozess des Client-Prozesses ausgeführt und nutzen so, wie Trigger, dessen Datenbankkontext. Weitere Informationen dazu finden Sie im Abschnitt **Attribut Auf Server ausführen**.

## Datenbankmethoden

---

Vier Datenbankmethoden werden nur auf dem Server-Rechner ausgeführt:

- **Datenbankmethode On Server Startup**
- **Datenbankmethode On Server Shutdown**
- **Datenbankmethode On Server Open Connection**
- **Datenbankmethode On Server Close Connection**

Fünf weitere Datenbankmethoden lassen sich sowohl auf dem Server-Rechner als auch auf dem Client-Rechner ausführen:

- **Datenbankmethode On Web Authentication**
- **Datenbankmethode On Web Connection**
- **Datenbankmethode On SQL Authentication**
- **Datenbankmethode On Backup Startup**
- **Datenbankmethode On Backup Shutdown**

Drei Datenbankmethoden lassen sich nur auf dem Client-Rechner ausführen:

- **Datenbankmethode On Startup**
- **Semaphore**
- **Datenbankmethode On Drop**

Weitere Informationen dazu finden Sie in den folgenden Abschnitten und im Handbuch *4D Programmiersprache*.

## 4D Server und Variablen

---

4D Server verwaltet eine Tabelle Interprozessvariablen. Diese Variablen liegen auf dem Server-Rechner. Bei einer kompilierten Datenbank ist die Tabellendefinition für Interprozessvariablen für Server und alle Client-Rechner gleich, jeder Rechner hat jedoch einen eigenen Variableninhalt.

Analog zu Prozessen haben auch jede Serverprozedur, Datenbankmethode und Trigger eine eigene Tabelle Prozessvariablen. Diese Prozessvariablen lassen sich in jeder Ausführungsphase erstellen und dynamisch verwenden.

## 4D Server, Mengen und temporäre Auswahlen

---

Mit 4D Server richtet sich die Sichtbarkeit von Mengen und temporären Auswahlen danach, wo sie erstellt wurden (Server-Prozess oder Client-Prozess) und nach der Art dieser Objekte (lokal, Prozess oder Interprozess). Weitere Informationen dazu finden Sie im Abschnitt **4D Server, Mengen und temporäre Auswahlen**.

## 4D Server, Mengen und temporäre Auswahlen

Sie können mit Interprozess-, Prozess- und lokalen Mengen sowie temporären Auswahlen arbeiten. Weitere Informationen dazu finden Sie im Handbuch *4D Programmiersprache* in den Kapiteln **Mengen** und **Temporäre Auswahl**.

- **Prozessmengen/temporäre Auswahlen:** Auf ein Prozessobjekt lässt sich nur über den Prozess zugreifen, der es erzeugt hat. Wurde es in einem Client Prozess erstellt, kann nur der auf dem Server angelegte Zwillingprozess darauf zugreifen. Prozessmengen werden gelöscht, sobald die Prozessmethode beendet ist. Prozessobjekte benötigen keine spezifische Vorsilbe im Namen.
- **Interprozessmengen/temporäre Auswahlen:** Ein Interprozessobjekt ist für alle Prozesse auf dem Rechner (Server oder Client) sichtbar, der sie erzeugt hat. Eine Menge oder temporäre Auswahl ist ein Interprozessobjekt, wenn das Zeichen <> ("kleiner als" gefolgt von "größer als") vorangestellt ist.  
**Hinweis:** Diese Schreibweise gilt für Windows und Macintosh. Auf Macintosh können Sie auch das Rautezeichen benutzen. Tippen Sie dazu bei gedrückter **Umschalt- und Wahltaste** den Buchstaben v ein.
- **Lokale Mengen/Client Mengen/temporäre Auswahlen:** Ein lokales/Client Objekt ist nur in dem Prozess sichtbar, der es angelegt hat. Der Name von lokalen bzw. Client Objekten beginnt mit dem Dollarzeichen (\$). Auch die Systemmenge *UserSet* gehört zu den lokalen bzw. Client Mengen, obwohl sie nicht mit einem Dollarzeichen \$ beginnt.

Nachfolgende Tabelle zeigt, wie temporäre Auswahlen sichtbar sind, je nachdem, wo sie erstellt wurden. Die Tabelle ist für beide Objektarten gleich:

Sichtbarkeit temporärer Auswahlen und Mengen					
	Client Prozess	Andere Client Prozesse	Andere Clients	Server Prozess	Andere Server Prozesse
<b>Erstellung in einem Client Prozess</b>					
\$test	x				
test	x			x (Trigger)	
<>test	x	x			
<b>Erstellung in einem Server Prozess</b>					
\$test				x	
test				x	
<>test				x	x

x = sichtbar

Sie müssen die Sichtbarkeit von Prozessen je nach der Art der Operationen, die Sie ausführen wollen, berücksichtigen. Wollen Sie z.B. eine Operation vom Typ **DIFFERENCE**, **INTERSECTION** oder **UNION** ausführen, stellen Sie sicher, dass alle Mengen auf dem Rechner sichtbar sind, der diese Operation ausführt.

Zur Optimierung sollten Sie den Erstellungsort und die Reichweite der Objekte jeweils nach der erforderlichen Sichtbarkeit wählen.

### **Was ist eine SQL Serverprozedur?**

---

Der Ausdruck Serverprozedur stammt aus der Server-Welt, die auf SQL basiert. Sendet eine Arbeitsstation eine Anfrage an einen SQL Server, sendet sie die Anfrage als Volltext in SQL Sprache an diesen Server. Die Anfrage wird vor der Ausführung auf dem SQL Server analysiert und interpretiert. Ist der Quellcode umfangreich und wird die Anfrage während einer Sitzung mehrere Male gesendet, beansprucht das natürlich viel Zeit für die Übertragung über das Netzwerk und die stets neue Analyse und Interpretation.

Deshalb sollte ein Weg gefunden werden, diese Anfrage zuerst über das Netzwerk zu senden, zu analysieren und zu interpretieren und nur noch auszuführen, wenn sie von der Arbeitsstation gesendet wird. Die Lösung war, den Quellcode der Anfrage (mit anderen Worten eine Prozedur) auf der Server-Seite zu lassen; die Arbeitsstation sendet dann als Anfrage nur den Namen dieser Prozedur. Da diese Prozedur auf dem Server gespeichert wird, nennen wir solche Prozeduren "Serverprozeduren".

Beachten Sie, dass eine Serverprozedur, die auf SQL basiert, Parameter von einer Arbeitsstation empfangen kann, die damit verbundenen Tasks synchron oder asynchron ausführt und evtl. ein Ergebnis an die Arbeitsstation zurückgibt. Löst eine Arbeitsstation die Ausführung einer Serverprozedur aus, überlässt die Station die Ausführung von Code bis zu einem gewissen Maß dem Server-Rechner.

### **Was ist eine 4D Server Serverprozedur?**

---

Selbst wenn wir den üblichen Namen benutzen, bieten Serverprozeduren unter 4D Server weit mehr als reguläre Serverprozeduren.

Mit 4D im lokalen Modus können Sie beim Verwenden einer Routine, z.B. **New process** einen Benutzerprozess starten, in dem eine Methode abläuft. Diese Methode heißt **Prozessmethode**. Weitere Informationen dazu finden Sie im Handbuch *4D Programmiersprache* im Abschnitt **Projektmethoden**.

Dasselbe können Sie mit 4D Server auf einer Arbeitsstation durchführen. Sie können zusätzlich mit der Funktion **Execute on server** auf dem Server-Rechner einen Benutzerprozess starten, in dem eine Methode abläuft. Über den Befehl **EXECUTE ON CLIENT** können Sie eine Methode in einem anderen Prozess auf einem anderen Client laufen lassen. In beiden Fällen heißt die Methode **Serverprozedur**.

Der Prozess, der auf dem Server-Rechner oder einem anderen Client gestartet wird, heißt ebenfalls **Serverprozedur**.

**Wichtig:** Der Hauptunterschied zwischen einer Serverprozedur, die auf SQL-basiert und einer Serverprozedur unter 4D Server ist, dass im ersten Fall eine SQL Prozedur, im zweiten Fall ein eigenständiger 4D Prozess abläuft.

### **Architektur der 4D Serverprozeduren**

---

Eine Serverprozedur hat wie ein normaler Prozess eine eigene Umgebung:

- **Aktuelle Auswahl pro Tabelle:** Jede Serverprozedur hat eine eigene aktuelle Auswahl. Eine Tabelle kann in verschiedenen Serverprozeduren auch verschiedene Auswahlen haben.
- **Aktueller Datensatz pro Tabelle:** Jede Tabelle kann in jeder Serverprozedur einen anderen aktuellen Datensatz haben.
- **Variablen:** Jede Serverprozedur hat ihre eigenen Prozessvariablen. Prozessvariablen werden nur im Bereich der jeweiligen Serverprozedur erkannt.
- **Standardtabelle:** Jede Serverprozedur hat eine eigene Standardtabelle.
- **Prozessmengen:** Jede Serverprozedur hat ihre eigenen Prozessmengen.
- **On Error Call:** Jede Serverprozedur hat ihre eigene Methode **On Error Call**.
- **Debugger Fenster:** Jede Serverprozedur kann ein eigenes Debugger Fenster haben.

Eine Serverprozedur kann in der Benutzeroberfläche Fenster öffnen und Daten anzeigen (z.B. **DISPLAY RECORD**). Sie kann jedoch im Gegensatz zu Benutzerprozessen auf der Arbeitsstation keine Dateneingabe auslösen (z.B. **ADD RECORD**), da es keinen Eingabe-Kernel auf dem Server-Rechner gibt.

Sie können so viele Serverprozeduren starten wie es Hardware und Speicher des Systems zulassen. Im Grunde genommen antwortet der 4D Server-Rechner nicht nur auf Anfragen der Arbeitsstation und von Web Browsern, er führt auch Prozesse aus, die mit anderen Prozessen interagieren, die auf dem Server-Rechner und remote 4D Rechnern laufen.

Ebenso wie 4D eine Multi-Tasking Umgebung für Benutzerprozesse auf einer Arbeitsstation bietet, bietet 4D Server eine Multi-Tasking Umgebung für Serverprozeduren. 4D Server verwaltet zum Beispiel eine Tabelle mit Interprozessvariablen, die Serverprozeduren für die Interprozesskommunikation benutzen können.

**Hinweis:** Die Methodeneigenschaft "Auf Server ausführen" lässt sich für eine Methode anwenden, die in einem Prozess auf dem Server ausgeführt wird. Die Methode verwendet dann jedoch einen Zwillingprozess des Client-Prozesses, d.h. sie kann die Umgebung dieses Client Prozesses nutzen. Es handelt sich dann nicht um eine 4D Serverprozedur. Weitere Informationen dazu finden Sie im Abschnitt **Attribut Auf Server ausführen**.

## **Was macht eine Serverprozedur?**

---

Fast alles, was im Handbuch *4D Programmiersprache* über die Funktionsweise von Prozessen und Befehlen beschrieben wird, gilt auch für Serverprozeduren. Davon ausgenommen ist die Dateneingabe.

Eine Serverprozedur kann Datensätze hinzufügen, suchen, sortieren, aktualisieren und löschen. Sie kann Mengen und Auswahlen verwenden, auf Dokumente auf der Festplatte zugreifen, mit BLOBs arbeiten, Datensätze drucken, u.v.m. Stellen Sie sich einfach vor, dass etwas nicht auf dem lokalen 4D Rechner, sondern auf dem Server-Rechner bzw. auf einer oder mehreren Arbeitsstationen ausgeführt wird.

Ein offensichtlicher Vorteil ist, dass eine Serverprozedur lokal auf dem Server-Rechner ausgeführt wird, also dort, wo auch die Datenbank-Engine liegt. So ist z.B. der Befehl **APPLY TO SELECTION** innerhalb einer Serverprozedur effizienter als über das Netzwerk. Das Beispiel im Abschnitt **Import mit Serverprozeduren (Beispiel)** zeigt, wie stark sich die Leistung über clever eingesetzte Serverprozeduren optimieren lässt.

Serverprozeduren, die auf einer oder mehreren Arbeitsstationen ausgeführt werden, ermöglichen, Aufgaben und Kommunikation auf mehrere Client Rechner zu verteilen.

Ein Beispiel dazu finden Sie im Handbuch *4D Programmiersprache* unter dem Befehl **REGISTER CLIENT**.

Der wichtigste Vorteil von Serverprozeduren sind jedoch die zusätzlichen Möglichkeiten für 4D Server. Mit Serverprozeduren können Sie Ihre eigenen 4D Server Dienste einrichten. Ihrer Phantasie sind keine Grenzen gesetzt. Das Beispiel im Abschnitt **Service mit Serverprozeduren (Beispiel)** zeigt eine Serverprozedur, die Clients mit Information über 4D Server bzw. den Server-Rechner ausstattet. Sie können z.B. die Volumens des Server-Rechners auflisten. Dieses Beispiel lässt sich leicht erweitern, so dass ein Client auch Information über ein Verzeichnis oder ein Dokument erhält.

## **Was eine Server Prozedur (ausgeführt auf dem Server) nicht macht?**

---

Generell sollten Serverprozeduren, die auf dem Server ausgeführt werden, nicht mit Oberflächenelementen arbeiten, wie z.B. Menüs, Fenster, Formulare, etc. Die Oberfläche wird ja auch nicht auf der Server-Seite verwaltet. Befehle, die Dialogfenster auf dem Server anzeigen, sowie Dialogfenster zur Dateneingabe, sollten vermieden werden.

Es folgt die Liste der Befehle, die Sie NICHT in Serverprozeduren verwenden sollten, die auf dem Server ausgeführt werden. Sie ist in drei Gruppen gegliedert:

- **Auf dem Server nicht zulässige Befehle**

Enthält eine Serverprozedur einen dieser Befehle, erscheint eine Meldung, dass der Befehl nicht auf 4D Server ausgeführt werden kann. Der Fehler #67 wird zurückgegeben. Sie können ihn mit dem Befehl **ON ERR CALL** verwalten.

**ACCUMULATE**

**ADD RECORD**

**\_o\_ADD SUBRECORD**

**APPEND MENU ITEM**

**BREAK LEVEL**

**POST OUTSIDE CALL**

**CHANGE LICENSES**

Count menu items

Count menus

**CREATE USER FORM**

**DELETE MENU ITEM**

**DELETE USER FORM**

**DISABLE MENU ITEM**

**DISPLAY SELECTION**

**EDIT ACCESS**

**EDIT FORM**

**ENABLE MENU ITEM**

**FILTER EVENT**

Get menu item

Get menu item key

Get menu item mark

Get menu item style

Get menu title

**SET PICTURE TO LIBRARY**

**\_o\_GRAPH TABLE**

**INSERT MENU ITEM**

**Level**  
**LIST USER FORMS**  
**Menu selected**  
**MODIFY RECORD**  
**MODIFY SELECTION**  
**\_o\_MODIFY SUBRECORD**  
**ON EVENT CALL**  
**\_o\_Open external window**  
**PAGE BREAK**  
**PAGE SETUP**  
**PRINT SETTINGS**  
**QUERY BY EXAMPLE**  
**QR REPORT**  
**Printing page**  
**REMOVE PICTURE FROM LIBRARY**  
**SET MENU ITEM**  
**SET MENU ITEM SHORTCUT**  
**SET MENU ITEM MARK**  
**SET MENU ITEM STYLE**  
**SET PICTURE TO LIBRARY**  
**SHOW MENU BAR**  
**Subtotal**

- **Auf dem Server ungeeignete Befehle**

Wir raten dringend, die folgenden Befehle nicht in Serverprozeduren zu verwenden, da sie für Methoden, die auf dem Server ausgeführt werden, ungeeignet sind. Sie können den Server blockieren und Fehler erzeugen und werden in jedem Fall nicht korrekt ausgeführt. Es erscheint keine explizite Fehlermeldung.

**ACCEPT**  
**Activated**  
**\_o\_ADD DATA SEGMENT**  
**After**  
**APPEND DATA TO PASTEBOARD**  
**APPEND TO LIST**  
**Before**  
**BLOB TO DOCUMENT**  
**BLOB to list**  
**BRING TO FRONT**  
**\_o\_C\_GRAPH**  
**CANCEL**  
**CHANGE CURRENT USER**  
**CHANGE PASSWORD**  
**CLEAR LIST**  
**CLEAR PASTEBOARD**  
**Copy list**  
**Count list items**  
**Count screens**  
**Create document(1)**  
**\_o\_Create resource file(1)**  
**Current form table**  
**Current user**  
**Deactivated**  
**DELETE FROM LIST**  
**DELETE USER**

**DIALOG**  
**\_o\_DISABLE BUTTON**  
**DRAG AND DROP PROPERTIES**  
**DRAG WINDOW**  
**Drop position**  
**\_o\_During**  
**\_o\_ENABLE BUTTON**  
**ERASE WINDOW**  
**EXPORT DATA(1)**  
**FILTER KEYSTROKE**  
**Find window**  
**Focus object**  
**FONT LIST**  
**\_o\_Font name**  
**\_o\_Font number**  
**Form event**  
**FORM FIRST PAGE**  
**FORM Get current page**  
**FORM GET PROPERTIES**  
**FORM GOTO PAGE**  
**FORM LAST PAGE**  
**FORM NEXT PAGE**  
**FORM PREVIOUS PAGE**  
**FORM SET INPUT**  
**FORM SET OUTPUT**  
**Frontmost process**  
**Frontmost window**  
**Get edited text**  
**GET GROUP LIST**  
**GET GROUP PROPERTIES**  
**GET HIGHLIGHT**  
**GET LIST ITEM**  
**GET LIST ITEM PROPERTIES**  
**GET LIST PROPERTIES**  
**GET MOUSE**  
**GET PASTEBOARD DATA**  
**GET PICTURE FROM PASTEBOARD**  
**Get text from pasteboard**  
**GET USER LIST**  
**GET USER PROPERTIES**  
**GET WINDOW RECT**  
**Get window title**  
**GOTO OBJECT**  
**GRAPH SETTINGS**  
**HIDE PROCESS**  
**HIDE TOOL BAR**  
**HIDE WINDOW**  
**HIGHLIGHT RECORDS**  
**HIGHLIGHT TEXT**  
**IMPORT DATA(1)**  
**In break**  
**In footer**  
**In header**  
**INSERT IN LIST**  
**\_o\_INVERT BACKGROUND**

**Is a list**  
**Is user deleted**  
**Keystroke**  
**List item parent**  
**List item position**  
**LIST TO BLOB**  
**Load list**  
**MAXIMIZE WINDOW**  
**Menu bar height**  
**Menu bar screen**  
**MINIMIZE WINDOW**  
**Modified**  
**New list**  
**Next window**  
**OBJECT GET COORDINATES**  
**OBJECT MOVE**  
**OBJECT SET LIST BY NAME**  
**OBJECT SET COLOR**  
**OBJECT SET ENTERABLE**  
**OBJECT SET FILTER**  
**OBJECT SET FORMAT**  
**OBJECT SET RGB COLORS**  
**OBJECT SET TITLE**  
**OBJECT SET VISIBLE**  
**Old**  
**Open document(1)**  
**Open resource file(1)**  
**ORDER BY(2)**  
**Outside call**  
**Pasteboard data size**  
**Pop up menu**  
**POST CLICK**  
**POST EVENT**  
**POST KEY**  
**QUERY BY FORMULA(2)**  
**QUERY(2)**  
**REDRAW**  
**\_o\_REDRAW LIST**  
**REDRAW WINDOW**  
**REGISTER CLIENT**  
**REJECT**  
**SAVE LIST**  
**SCREEN COORDINATES**  
**SCREEN DEPTH**  
**Screen height**  
**Screen width**  
**Select folder**  
**SELECT LIST ITEMS BY POSITION**  
**SELECT LIST ITEMS BY REFERENCE**  
**SELECT LOG FILE**  
**Selected list items**  
**Self**  
**SET CURSOR**  
**SET FIELD TITLES**  
**Set group properties**

**SET LIST ITEM**  
**SET LIST ITEM PROPERTIES**  
**SET LIST PROPERTIES**  
**SET PICTURE TO PASTEBOARD**  
**SET SCREEN DEPTH**  
**SET TABLE TITLES**  
**SET TEXT TO PASTEBOARD**  
**SET TIMER**  
**Set user properties**  
**SET WINDOW RECT**  
**Shift down**  
**SHOW PROCESS**  
**SHOW WINDOW**  
**SORT LIST**  
**User in group**  
**Validate password**  
**Window kind**  
**WINDOW LIST**  
**Window process**

(1) Nur wenn der erste Parameter ein leerer String ist.

(2) Nur wenn die Syntax die Anzeige eines Dialogfensters auslöst, z.B. **ORDER BY**([Tabelle]).

- **Befehle ohne Auswirkung auf dem Server**

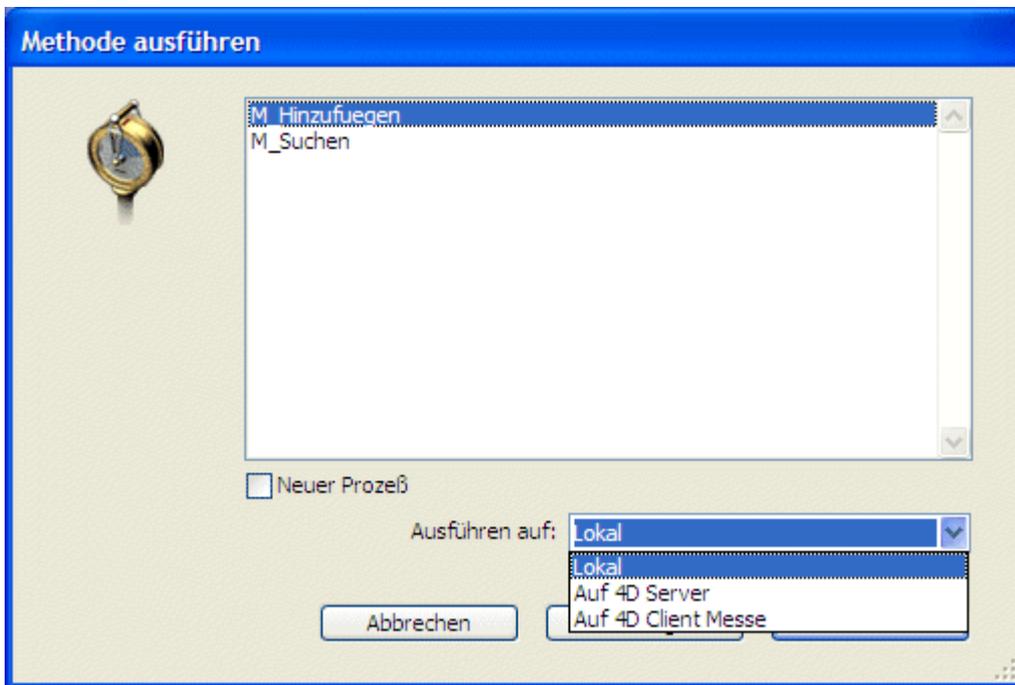
Diese Befehle haben keine Auswirkung, wenn sie in Serverprozeduren auf dem Server ausgeführt werden. Es erscheint keine explizite Fehlermeldung.

**GRAPH**  
**MESSAGES OFF**  
**MESSAGES ON**  
**SET MENU BAR**  
**SHOW TOOL BAR**

## **Serverprozedur starten**

---

- Von 4D aus können Sie eine Serverprozedur im Dialogfenster **Methode ausführen** manuell starten:



Sie können sie auf dem 4D Server- oder einem Client-Rechner ausführen. Beachten Sie, dass nur die Client-Rechner in der Liste erscheinen, welche Sie zuvor registriert haben. Weitere Informationen dazu finden Sie im Abschnitt **Serverprozeduren auf Client-Rechnern** und unter dem Befehl **REGISTER CLIENT**.

- Sie können in 4D eine Serverprozedur auch per Programmierung starten: Verwenden Sie dazu die Routinen **Execute on server** oder **EXECUTE ON CLIENT**.

**Hinweis:** Sie können von remote 4D aus keine Befehle zur Prozessverwaltung für Serverprozeduren auf dem Server verwenden. Das sind die Befehle **DELAY PROCESS**, **PAUSE PROCESS** und **RESUME PROCESS**.

- Sie können eine Serverprozedur über eine Methode aufrufen, die auf 4D Server ausgeführt wird (Server Datenbankmethode, Methode mit dem Attribut **Auf Server ausführen** oder Serverprozedur): Verwenden Sie dazu die Funktionen **Execute on server**, **New process** oder **EXECUTE ON CLIENT**.

## Interprozess Kommunikation zwischen Serverprozeduren und Benutzerprozessen

---

Serverprozeduren können miteinander kommunizieren durch:

- Interprozessvariablen
- lokale oder globale Semaphoren
- Datensätze
- Interprozessmengen und temporäre Auswahlen auf Interprozessebene
- die Befehle **GET PROCESS VARIABLE**, **SET PROCESS VARIABLE** und **VARIABLE TO VARIABLE**

Weitere Informationen dazu finden Sie im Handbuch *4D Programmiersprache* in der Beschreibung zu den Befehlen. Nochmal zur Erinnerung: Die 4D Befehle arbeiten im Bereich des Server-Rechners genauso wie im Bereich eines Client-Rechners.

**Hinweis:** Die Befehle **POST OUTSIDE CALL** und **Outside call** haben auf dem Server-Rechner keine Bedeutung, da Serverprozeduren keine Benutzeroberfläche mit Dateneingabe haben.

Es gibt jedoch ein anderes wichtiges Feature: Benutzerprozesse, die auf dem Client-Rechner laufen, können mit den Befehlen **GET PROCESS VARIABLE**, **SET PROCESS VARIABLE** und **VARIABLE TO VARIABLE** Prozessvariablen (\*) von einer Serverprozedur lesen und schreiben.

(\*) ebenso wie die Interprozessvariable auf dem Server-Rechner.

**Wichtig:** Die Prozesskommunikation zwischen mehreren Rechnern über die Befehle **GET PROCESS VARIABLE**, **SET PROCESS VARIABLE** und **VARIABLE TO VARIABLE** ist nur vom Client zum Server möglich. Es ist immer ein Client-Prozess, der die Variablen einer Serverprozedur liest oder schreibt.

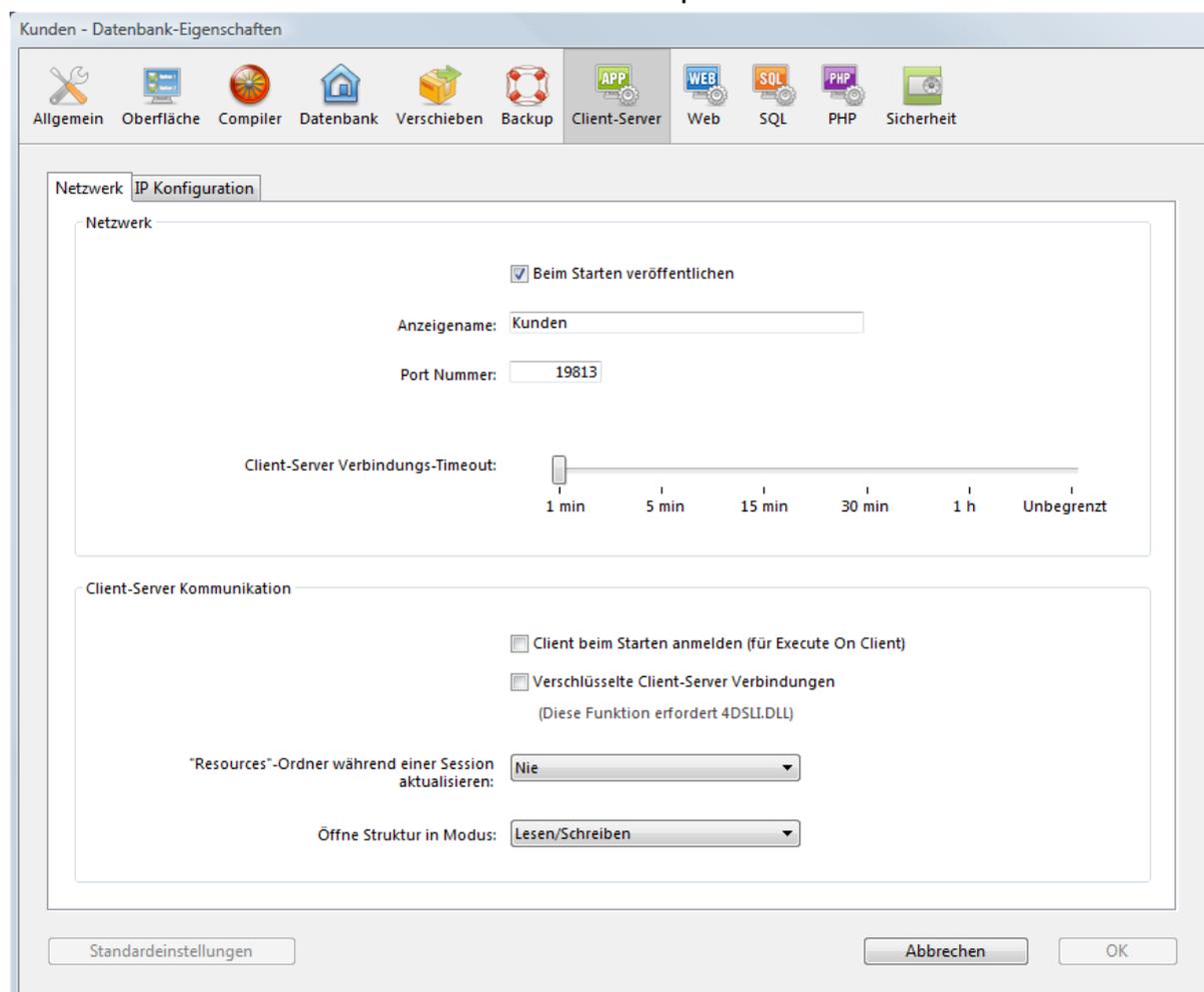
## Serverprozeduren auf Client-Rechnern

Sie können Serverprozeduren auf einem oder mehreren Client-Rechnern ausführen. Sie arbeiten wie die Serverprozeduren auf dem Server. Der einzige Unterschied ist, dass die Serverprozedur auf einem Client auch die Dateneingabe zulässt. Weitere Informationen dazu finden Sie im Abschnitt **Serverprozeduren**.

Jede Arbeitsstation, die auf Anfrage des Servers oder eines anderen Clients Serverprozeduren ausführen soll, muss explizit für die Arbeitssitzung registriert sein. Dafür gibt es zwei Möglichkeiten: Jeden Client automatisch beim Anmelden oder per Programmierung registrieren.

### Jeden Client-Rechner automatisch beim Anmelden an 4D Server registrieren

Markieren Sie dazu in den Datenbank-Eigenschaften auf der Registerkarte **Netzwerk** der Seite **Client-Server** die Option "Client beim Starten anmelden":



Ist die Option markiert, gilt jeder Client-Rechner, der sich an die Datenbank anmeldet, in 4D Server als Client, der Serverprozeduren ausführen kann. Auf dem

Server wird ein Prozess mit dem Namen des Client Rechners angelegt. Ein entsprechender Prozess wird auch auf jedem Client-Rechner erstellt.

## **Client per Programmierung registrieren**

---

Sie können auch einen oder mehrere Client-Rechner per Programmierung registrieren. Auf diese Weise können Sie die Arbeitsstationen, die Sie registrieren wollen, selbst auswählen und eigene Namen vergeben. Verwenden Sie dazu den Befehl **REGISTER CLIENT**.

## **Registrierung von Client aufheben**

---

Unabhängig von der Art der Registrierung können Sie jede Registrierung der Arbeitsstation für die aktuelle Sitzung wieder aufheben. Rufen Sie dazu für die entsprechende Arbeitsstation den Befehl **UNREGISTER CLIENT** auf. Der Registrierungsprozess – er hat den Namen des Client – wird dann auf dem Server aus der Gruppe Benutzerprozesse und auf dem Client entfernt.

**Hinweis:** Mit dem Befehl **GET REGISTERED CLIENTS** erhalten Sie die Liste der für die Sitzung registrierten Clients und ihr jeweiliges Arbeitsaufkommen, d.h. die Anzahl der noch auszuführenden Methoden.

Weitere Informationen zu diesen Befehlen finden Sie im Handbuch *4D Programmiersprache*.

## Import mit Serverprozeduren (Beispiel)

Folgendes Beispiel zeigt, wie sich der Datenimport im Client/Server-Betrieb enorm beschleunigen lässt. Mit der Projektmethode **Regulärer Import** können Sie testen, wie lange es dauert, mit dem Befehl **IMPORT TEXT** Datensätze auf der Arbeitsstation zu importieren:

```
` Regulärer Import
$vhDocRef:=Open document("")
If(OK=1)
  CLOSE DOCUMENT($vhDocRef)
  INPUT FORM([Table1];"Import")
  $vhStartTime:=Current time
  IMPORT TEXT([Table1];Document)
  $vhEndTime:=Current time
  ALERT("Es dauerte "+String(0+($vhEndTime-$vhStartTime))+ " Sekunden.")
End if
```

Mit dieser Methode analysiert 4D die Textdatei, erstellt dann für jeden Datensatz einen neuen Datensatz, füllt die Datenfelder mit den importierten Daten und sendet den Datensatz zum Server-Rechner, damit er der Datenbank hinzugefügt wird. Demzufolge laufen viele Anfragen über das Netzwerk. Diese Operation lässt sich mit einer Serverprozedur optimieren, da sie dann lokal auf dem Server-Rechner ausgeführt wird. Der Client-Rechner lädt das Dokument in ein BLOB, startet dann eine Serverprozedur, die das BLOB als Parameter übergibt. Die Serverprozedur speichert das BLOB in einem Dokument auf der Festplatte des Server-Rechners und importiert dann das Dokument lokal. Der Datenimport wird demnach lokal ausgeführt (mit einer Geschwindigkeit, die mit 4D im lokalen Modus vergleichbar ist), da die meisten Anfragen über das Netzwerk eliminiert wurden.

Es folgt die Projektmethode **CLIENT IMPORT**. Sie wird auf dem Client-Rechner ausgeführt und ruft die Serverprozedur **SERVER IMPORT** auf:

```
` Projektmethode CLIENT IMPORT
` CLIENT IMPORT ( Pointer ; Text)
` CLIENT IMPORT ( -> [Table] ; Input form )

C_POINTER($1)
C_TEXT($2)
C_TIME($vhDocRef)
C_BLOB($vxData)
C_LONGINT(spErrCode)

` Wähle das zu importierende Dokument
```

```

$vhDocRef:=Open document("")
If(OK=1)
  ` Wurde ein Dokument ausgewählt, schlieÙe es
    CLOSE DOCUMENT($vhDocRef)
    $vhStartTime:=Current time
  ` Versuche, es in den Speicher zu laden
    DOCUMENT TO BLOB(Document;$vxData)
    If(OK=1)
  ` Konnte das Dokument in das BLOB geladen werden,
  ` Starte die Serverprozedur, die die Daten auf dem Server-Rechner importiert.
    $spProcessID:=Execute on server("SERVER IMPORT";32*1024;
    "Server Import Services";Table($1);$2;$vxData)
  ` Wir benötigen das BLOB nicht länger in diesem Prozess
    CLEAR VARIABLE($vxData)
  ` Warte bis die Operation durch die Serverprozedur vollständig ausgeführt ist
    Repeat
      DELAY PROCESS(Current process;300)
      GET PROCESS VARIABLE($spProcessID;spErrCode;spErrCode)
      If(Undefined(spErrCode))
    ` <strong>Hinweis:</strong> Hat die Serverprozedur nicht ihre eigene Instanz der
    Variablen spErrCode
    ` initialisiert, wird evtl. eine undefinierte Variable zurückgegeben
      spErrCode:=1
    End if
    Until(spErrCode<=0)
  ` Teile der Serverprozedur mit, dass wir bestätigen
    spErrCode:=1
    SET PROCESS VARIABLE($spProcessID;spErrCode;spErrCode)
    $vhEndTime:=Current time
    ALERT("Es dauerte "+String(0+($vhEndTime-$vhStartTime))+ " Sekunden.")
  Else
    ALERT("Es gibt nicht genügend Speicher, um das Dokument zu laden.")
  End if
End if

```

Es folgt die Projektmethode **SERVER IMPORT**, ausgeführt als Serverprozedur:

- ` Projektmethode SERVER IMPORT
- ` SERVER IMPORT (Long; Text; BLOB)
- ` SERVER IMPORT (Table Number; Input form; Import Data)

```

C_LONGINT($1)
C_TEXT($2)
C_BLOB($3)
C_LONGINT(spErrCode)

```

- ` Operation ist noch nicht beendet, setze spErrCode auf 1.

```

spErrCode:=1
$vpTable:=Table($1)
INPUT FORM($vpTable->;$2)
$vsDocName:="Import File "+String(1+Random)
If(On Windows)
    $vsDocName:=$vsDocName+".txt" ` Auf Windows ist die Dateieindung zwingend
End if
DELETE DOCUMENT($vsDocName)
BLOB TO DOCUMENT($vsDocName;$3)
IMPORT TEXT($vpTable->;$vsDocName)
DELETE DOCUMENT($vsDocName)
    ` Operation ist beendet, setze spErrCode auf 0.
spErrCode:=0
    ` Warte, bis der anfragende Client das Ergebnis erhalten hat.
Repeat
    DELAY PROCESS(Current process;1)
Until(spErrCode>0)

```

**Hinweis:** Die Projektmethode *On Windows* finden Sie im Handbuch *4D Programmiersprache* im Abschnitt **Einführung in Systemdokumente**.

Sind diese beiden Projektmethoden in einer Datenbank integriert, können Sie den Import über eine Serverprozedur ausführen, wenn Sie beispielsweise schreiben:

```

CLIENT IMPORT(->[Table1];"Import")

```

Bei einigen Benchmarks werden Sie feststellen, dass der Import von Datensätzen mit dieser Methode bis zu 60 mal schneller läuft als über den regulären Import.

## Service mit Serverprozeduren (Beispiel)

---

Im Beispiel aus dem vorigen Abschnitt **Import mit Serverprozeduren (Beispiel)** startet oder endet eine Serverprozedur immer, wenn ein Datenimport angefordert wird. In diesem Beispiel startet eine Serverprozedur automatisch beim Starten der Serverdatenbank und kann von jedem remote 4D, der sich an die Datenbank anmeldet, beliebig beendet oder wieder gestartet werden. Sobald die Serverprozedur läuft, kann sie asynchron auf die verschiedenen Anfragen antworten, welche die an die Datenbank angeschlossenen Clients senden.

Der vorige Abschnitt zeigt, wie Sie einen in 4D Server bestehenden Service zur Optimierung von Operationen integrieren können. Dieser Abschnitt erläutert, wie Sie neue Services integrieren können, die für alle angemeldeten Client-Rechner verfügbar sind. Sie können das Beispiel auch als Vorlage zum Einrichten eigener Services verwenden.

### Automatisches Starten der Serverprozedur

---

Die Serverprozedur wird automatisch gestartet mit der **Datenbankmethode On Server Startup**:

```
` Datenbankmethode On Server Startup  
START SP SERVICES
```

Da die **Datenbankmethode On Server Startup** die Projektmethode **SP SERVICES** als Serverprozedur startet, läuft sie, sobald die Datenbank mit 4D Server gestartet wird, unabhängig davon, ob derzeit Clients an die Server-Datenbank angemeldet sind. Das hier abgebildete Verwaltungsfenster von 4D Server zeigt die aktive Serverprozedur, ohne dass ein Client angemeldet ist.

The screenshot shows the '4D Mailing - 4D Server Administration' window. At the top, there are tabs for 'Monitor', 'Anwender (0)', 'Prozesse (13)', 'Wartung', 'Anwendungsserver', 'SQL Server', and 'HTTP Server'. Below the tabs, there are buttons for 'Anwender Prozesse', '4D Prozesse (13)', and 'Reserve Prozesse (0)'. The main area contains a table with the following columns: 'Prozessname', 'Session', 'Typ', 'Num', 'Status', 'CPU-Zeit', and 'Aktivität'.

Prozessname	Session	Typ	Num	Status	CPU-Zeit	Aktivität
DB4D Flush		DB4D Server	0	Ausführung	00:00:01	0,00 %
DB4D Index builder		DB4D Server	0	Ausführung	00:00:00	0,00 %
Task-Managers		SQL Server	0	Ausführung	00:00:02	0,00 %
Arbeitsplatz-Manager	-	Anwendungsserver	3	Warte auf Semaphore	00:00:00	0,00 %
Internal Bridge Prozess	-	Anwendungsserver	5	Warte auf Semaphore	00:00:00	0,00 %
Internal Timer Prozess	-	Anwendungsserver	2	Ausführung	00:00:02	1,61 %
Serverfenster	-	Anwendungsserver	1	Warte auf Ereignis	00:00:10	3,23 %
SP SERVICES	-	Anwendungsserver	6	Warte auf Ereignis	00:00:00	0,00 %
Web Server	-	Web Server	4	Ausführung	00:00:00	0,00 %

Below the table, there is a detailed view for 'Anwendungsserver - SP SERVICES' showing '0 %' activity. At the bottom, there are five icons for process management: 'Prozess löschen', 'Prozess anhalten', 'Prozess fortsetzen', 'Prozess debuggen', and 'Anwender beobachten'.

## Serverprozeduren individuell starten und beenden

Die Projektmethode **START SP SERVICES** sieht folgendermaßen aus:

```
\ Projektmethode START SP SERVICES
\dvSPServices:=Execute on server("SP SERVICES";32*1024;"SP SERVICES";*)
```

Da die Funktion **Execute on server** wie **New process** arbeitet, wenn sie auf dem Server-Rechner aufgerufen wird, kann auf dem Server-Rechner bzw. auf jedem Client-Rechner dieselbe Methode (**START SP SERVICES**) verwendet werden, um die Methode **SP SERVICES** als Serverprozedur auf dem Server-Rechner aufzurufen.

Die Projektmethode **STOP SP SERVICES** stoppt die Projektmethode **SP SERVICES**

```
\ Projektmethode STOP SP SERVICES
SET PROCESS VARIABLE(\dvSPServices;vbStopSPServices;True)
```

Startet die Projektmethode **SP SERVICES**, setzt sie die Prozessvariable *vbStopSPServices* auf Falsch und durchläuft die Schleife, bis diese Variable vom Typ Boolean den Wert Wahr annimmt. Mit dem Befehl **SET PROCESS VARIABLE** lässt sich in jedem Benutzerprozess auf dem Server-Rechner bzw. auf jedem Client-Rechner der Wert der Variablen *vbStopSPServices* verändern und so die Serverprozedur nach eigenen Wünschen stoppen.

## Mit Serverprozeduren kommunizieren

Die Serverprozedur muss Anfragen des Clients jederzeit und in beliebiger Reihenfolge asynchron empfangen und beantworten können. Das ist am einfachsten mit einer Tabelle möglich.



SP Anfragen	
Anf_Nr	2 <sup>32</sup>
Anf_Typ	A
Anf_Status	2 <sup>16</sup>
Anf_Daten	🟩
Anf_Parameter	T

Die Tabelle *[SP Anfragen]* enthält folgende Datenfelder:

- *[SP Anfragen]Anf\_Nr* arbeitet mit der Funktion **Sequence number**. Sie dient nur zum Identifizieren der Anfragen.
- *[SP Anfragen]Anf\_Typ* beschreibt die Art der Anfrage.
- *[SP Anfragen]Anf\_Status* kann folgende Werte annehmen:

#### **Wert Beschreibung**

- |     |   |
|-----|---|
| 1   | Die Anfrage wurde gesendet, jedoch noch nicht bearbeitet            |
| 0   | Die Anfrage wurde erfolgreich bearbeitet.                           |
| < 0 | Die Anfrage wurde bearbeitet, es ist jedoch ein Fehler aufgetreten. |

**Hinweis:** Die Werte wurden willkürlich für dieses Beispiel gewählt, sie sind nicht von 4D vorgegeben.

- *[SP Anfragen]Anf\_Daten* ist ein BLOB mit den Daten der Anfrage. Sie kann sowohl Daten enthalten, die der Anfrager sendet als auch Daten, die die Serverprozedur an den Anfrager zurückgibt.
- *[SP Anfragen]Anf\_Parameter* enthält optional Parameterwerte, die der Anfrager an die Serverprozedur sendet.

## **Warum eine Tabelle verwenden?**

---

Die Kommunikation zwischen einem Client-Prozess und einer Serverprozedur lässt sich mit den Befehlen **GET PROCESS VARIABLE**, **SET PROCESS VARIABLE** und **VARIABLE TO VARIABLE** herstellen. Diese Befehle werden z. B. im Abschnitt **Import mit Serverprozeduren (Beispiel)** und in der Projektmethode **STOP SP SERVICES** verwendet.

Hier muss das System jedoch zulassen, dass die Serverprozedur variable Datenmengen empfängt und zurücksendet. Sie könnten zwar auch Arrays, einschließlich Arrays vom Typ Text und Bild verwenden, zwei Gründe sprechen jedoch für die Verwendung einer Tabelle:

- Der Algorithmus zum Bearbeiten von Anfragen via Datensätzen lässt sich einfacher einrichten. Wollen Sie eine Anfrage von einem Client-Rechner aus senden, müssen Sie in der Tabelle lediglich eine Anfrage hinzufügen. Wird die Anfrage von der Serverprozedur aus beantwortet, muss diese Anfrage lediglich geändert werden.
- Anfragen in einer Tabelle werden auch auf der Festplatte gespeichert. Von daher ist die Größe einer umfangreichen Anfrage kein Problem, da sie aus dem

Speicher entfernt werden kann (im Gegensatz zu Daten, die in Arrays gespeichert sind).

## Eine Anfrage vom Client-Rechner senden

---

Die Projektmethode **Client post request** ist eine generische Methode zum Senden einer Anfrage:

```
` Projektmethode Client post request
` Client post request ( String { ; Text } ) -> Long
` Client post request ( Anfrage_Typ { ; Parameter } ) -> Anfrage_Nr
CREATE RECORD([SP Anfragen])
[SP Anfragen]Anf_Nr:=Sequence number([SP Anfragen])
[SP Anfragen]Anf_Typ:=$1
[SP Anfragen]Anf_Status:=1
If(Count parameters>=2)
    [SP Anfragen]Anf_Parameter:=$2
End if
SAVE RECORD([SP Anfragen])
$0:= [SP Anfragen]Anf_Nr
```

Die Methode gibt die Nummer der Anfrage zurück. Die Funktion **Sequence number** stellt ihre Einzigartigkeit sicher. Wurde der Datensatz in der Datenbank [SP Anfragen] hinzugefügt, kann der Client das Datenfeld [SP Anfragen]Anf\_Status befragen und abwarten, bis die Serverprozedur die Anfragen vollständig bearbeitet hat.

## Den Status der Anfrage abfragen und das Ergebnis auf dem Client-Rechner erhalten

---

Die Projektmethode **Client get result** ist eine generische Methode zum Abfragen des Status der Anfrage. Wie bereits erwähnt, weiss der Client, sobald das Datenfeld [SP Anfragen]Anf\_Status einen anderen Wert als 1 hat, dass die Serverprozedur die Anfrage bearbeitet hat (erfolgreich oder nicht erfolgreich).

```
` Projektmethode Client get result
` Client get result ( Long ; ->BLOB { ; Long } ) -> Long
` Client get result ( Anfrage-Nr; ->Daten { ; Delay } ) -> Error Code
C_LONGINT($0;$1;$vIDelay)
$0:=1
$vIDelay:=0
If(Count parameters>=3)
    $vIDelay:=$3
End if
READ ONLY([SP Anfragen])
Repeat
    QUERY([SP Anfragen];[SP Anfragen]AnfID=$1)
    If(Records in selection([SP Anfragen])>0)
```

```

If([SP Anfragen]Anf_Status#1)
    $2->:=[SP Anfragen]Anf_Daten
    READ WRITE([SP Anfragen])
    While(Locked([SP Anfragen]))
        WAITING LOOP($vIDelay)
        LOAD RECORD([SP Anfragen])
    End while
    DELETE RECORD([SP Anfragen])
    $0:=[SP Anfragen]Anf_Status
End if
Else
    ` Datensatz für die Anfrage ging verloren!
    ` Das sollte zwar nicht passieren. Setze trotzdem den Fehler auf -2 (willkürlicher Wert)
    $0:=-2
End if
    ` Die Anfrage wurde noch nicht bearbeitet
    If($0=1)
        WAITING LOOP($vIDelay)
    End if
Until($0&NBSP;#&NBSP;1)
READ ONLY([SP Anfragen])

```

Hat die Serverprozedur die Anfrage erfolgreich bearbeitet, kopiert die Methode das Ergebnis (falls vorhanden) aus dem Datensatz in das BLOB, dessen Zeiger als Parameter übergeben wurde. Die aufrufende Methode analysiert und verwendet dann die BLOB Daten je nach Art der Anfrage. Beachten Sie, dass der Client für das Löschen des Datensatzes [SP Anfragen] zuständig ist, sobald die Anfrage vollständig bearbeitet ist.

Die kleine Projektmethode **WAITING LOOP** durchläuft die Schleife, bis eine Anzahl Ticks vergangen ist:

```

    ` Projektmethode WAITING LOOP
    ` WAITING LOOP ( Long )
    ` WAITING LOOP ( Delay in ticks )
C_LONGINT($1)
$vIStartTicks:=Tickcount
Repeat
    IDLE
Until(((Tickcount-$vIStartTicks)>=$1)|Process aborted)

```

**Zu Erinnerung:** Der Befehl **DELAY PROCESS** hat im Anwendungsprozess keine Auswirkung. Mit der Projektmethode **WAITING LOOP** wartet der Prozess die erforderliche Zeitspanne ab, selbst wenn die Anfrage von einem Benutzerprozess auf einem Client-Rechner ausging.

## Serverprozeduren und dazugehörige Unterrountinen

Die Projektmethode **SP SERVICES** läuft als Serverprozedur auf dem Server-Rechner. Der gesamte Aufbau dieser Methode, hier gezeigt in Pseudocode, ist denkbar einfach:

```
Initialisiere eine Variable "stop"
Wiederhole
    Suche nach Anfragen mit dem Wert des Datenfeldes [SP Anfragen]Anf_Status ist
gleich 1
    Für jede Anfrage
        Rufe je nach Art der Anfrage eine Unterroutine auf,
        die das Ergebnis im Datenfeld [SP Anfragen]Anf_Daten speichert
        Ändere den Status der Anfrage, so dass der Client weiß, was passiert ist
    Ende für
    "Schlafe" ein wenig vor erneutem Starten
Bis die Variable "stop" den Wert wahr hat.
```

Hier ist der echte Source Code:

```
` Projektmethode SP SERVICES
` Die Serverprozedur startet
vbStopSPServices:=False
` Die Serverprozedur benötigt auf Tabellen etc. keinen Zugriff im Lese/Schreibmodus
READ ONLY(*)
` ...außer der Tabelle [SP Anfragen]
READ WRITE([SP Anfragen])
Repeat
` Suche nach noch nicht bearbeiteten Anfragen
QUERY([SP Anfragen];[SP Anfragen]Anf_Status=1)
` Bearbeite diese Anfragen eine nach der anderen
For($vlRecord;1;Records in selection([SP Anfragen]))
` Ist der Datensatz Anfrage gesperrt, warte bis er freigegeben wird
While(Locked([SP Anfragen]))
` Warte eine Sekunde vor erneutem Versuch
DELAY PROCESS(Current process;60)
` Versuche, Zugriff im Lese/Schreibmodus zu erhalten
LOAD RECORD([SP Anfragen])
End while
` Nimm an, dass die Anfrage erfolgreich bearbeitet wird
[SP Anfragen]Anf_Status:=0
Case of
    :([SP Anfragen]Anf_Typ="Server Information")
        SP DO SERVER INFORMATION
    :([SP Anfragen]Anf_Typ="Volume List")
        SP DO VOLUME LIST
    :([SP Anfragen]Anf_Typ="Browse Directory")
        SP DO BROWSE DIRECTORY([SP Anfragen]Anf_Parameter)
` ...
` HIER KÖNNEN ANDERE ANFRAGETYPEN HINZUGEFÜGT WERDEN!
```

```

\ ...
    Else
\ Der Anfragetyp ist unbekannt, gibt Fehler -1 zurück (willkürlicher Wert)
    [SP Anfragen]Anf_Status:=-1
    End case
\ Sorge dafür,dass der Anfragestatus ungleich 1 ist
\ (falls eine Unterroutine ihn auf 1 setzt)
    If([SP Anfragen]Anf_Status=1)
        [SP Anfragen]Anf_Status:=-3
    End if
\ Aktualisiere den Datensatz Anfrage
    SAVE RECORD([SP Anfragen])
\ Gehe zur nächsten unbearbeiteten Anfrage
    NEXT RECORD([SP Anfragen])
End for
\ Gib den zuletzt bearbeiteten Datensatz Anfrage frei
    UNLOAD RECORD([SP Anfragen])
\ Warte eine Sekunde vor erneutem Starten der Antwort auf die Anfrage
    DELAY PROCESS(Current process;60)
\ Bleibe in der Schleife, bis die SP angewiesen wird, die Ausführung zu stoppen
Until(vbStopSPServices)

```

Die Projektmethode **SP SERVICES** kann als Vorlage zur Integration neuer Services in einer Datenbank dienen. In diesem Abschnitt betrachten wir näher die Unterroutinen **SP DO SERVER INFORMATION** und **SP DO VOLUME LIST**. Die Unterroutine **SP DO BROWSE DIRECTORY**, die den Parameter übernimmt, den der Client im Datenfeld `[SP Anfragen]Anf_Parameter` übersendet, wird hier nicht erläutert.

Die Projektmethode **SP SERVICES** ruft je nach Anfragetyp eine Unterroutine auf, welche die resultierenden Daten im Datenfeld `[SP Anfragen]Anf_Daten` speichert. **SP SERVICES** sichert den Datensatz und ändert den Status der Anfrage.

Hier folgt die Unterroutine **SP DO SERVER INFORMATION**. Sie speichert die server-bezogene Information im BLOB. Eine andere Projektmethode entnimmt die BLOB Daten entsprechend auf dem Client-Rechner.

```

\ Projektmethode SP DO SERVER INFORMATION
TEXT TO BLOB(Application version(*);[SP Anfragen]Anf_Daten;UTF8 C string)
TEXT TO BLOB(Structure file;[SP Anfragen]Anf_Daten;UTF8 C string;*)
TEXT TO BLOB(Data file;[SP Anfragen]Anf_Daten;UTF8 C string;*)
PLATFORM PROPERTIES($vIPlatform;$vISystem;$vIMachine)
VARIABLE TO BLOB($vIPlatform;[SP Anfragen]Anf_Daten;*)
VARIABLE TO BLOB($vISystem;[SP Anfragen]Anf_Daten;*)
VARIABLE TO BLOB($vIMachine;[SP Anfragen]Anf_Daten;*)

```

Hier folgt die Unterroutine **SP DO VOLUME LIST**. Sie speichert die volume-bezogene Information im BLOB. Eine andere Projektmethode entnimmt die BLOB Daten entsprechend auf dem Client-Rechner.

```

  ` Projektmethode SP DO VOLUME LIST
VOLUME LIST($asVName)
  $vISize:=Size of array($asVName)
  ARRAY REAL($arVSize;$vISize)
  ARRAY REAL($arVUsedSpace;$vISize)
  ARRAY REAL($arVFreeSpace;$vISize)
  For($vIElem;1;$vISize)
    VOLUME ATTRIBUTES($asVName{$vIElem};$arVSize{$vIElem};
      $arVUsedSpace{$vIElem};$arVFreeSpace{$vIElem})
  End for
  VARIABLE TO BLOB($asVName;[SP Anfragen]Anf_Daten)
  VARIABLE TO BLOB($arVSize;[SP Anfragen]Anf_Daten;*)
  VARIABLE TO BLOB($arVUsedSpace;[SP Anfragen]Anf_Daten;*)
  VARIABLE TO BLOB($arVFreeSpace;[SP Anfragen]Anf_Daten;*)

```

## Die Server Information auf einem Client-Rechner zeigen

Mit den generischen Projektmethoden *Client post request* und *Client get result* zeigt die Projektmethode **M\_SERVER\_INFORMATION** die Serverinformation an, die die Serverprozedur auf dem Client-Rechner zurückgibt. Diese Methode kann einem Menübefehl zugeordnet werden oder z.B. über eine Objektmethode für eine Schaltfläche aufgerufen werden:

```

  ` M_SERVER_INFORMATION
  C_BLOB(vxData)
  C_LONGINT($vAnf_Nr;$vIErrCode;$vIOffset)
  ` Stelle die Anfrage
  $vAnfID:=Client post request("Server Information")
  ` Frage den Anfragestatus ab und erhalte das Ergebnis
  $vIErrCode:=Client get result($vAnfID;->vxDaten;60)
  ` Ist die Anfrage erfolgreich abgeschlossen, zeige das Ergebnis an
  If($vIErrCode=0)
  ` Entnehme die Information aus dem BLOB
  $vIOffset:=0
  vsServerVersion:=BLOB to text(vxDaten;UTF8 C string;$vIOffset)
  vsStructureFile:=BLOB to text(vxDaten;UTF8 C string;$vIOffset)
  vsDataFile:=BLOB to text(vxDaten;UTF8 C string;$vIOffset)
  BLOB TO VARIABLE(vxDaten;$vIPlatform;$vIOffset)
  BLOB TO VARIABLE(vxDaten;$vISystem;$vIOffset)
  BLOB TO VARIABLE(vxDaten;$vIMachine;$vIOffset)
  ` Analysiere die Plattform Eigenschaften
  vs4DPlatform:="Unbekannte 4D Server Version"
  vsSystem:="Unbekannte System Version"
  vsMachine:="Unbekannter Rechner"
  ` ...
  ` Hier ist der Code (nicht aufgelistet) der $vISystem und $vIMachine analysiert

```

```

` (siehe Beispiel für den Befehl PLATFORM PROPERTIES)
`
...
` Zeige das Ergebnis
  DIALOG([SP Anfragen];"SERVER INFORMATION")
Else
  ALERT("Anfrage error "+String($vErrCode))
End if
` BLOB wird nicht länger benötigt
CLEAR VARIABLE(vxDaten)

```

Hier ist das Formular `[SP Anfragen];"SERVER INFORMATION"` in der Ausführung:

The screenshot shows a dialog box with the following fields and values:

- 4D Server Version Number:** F0031120
- Structure File Pathname:** C:\4DFP\bases\BasesPro\MyMusic\MyMusic.4DB
- Data File Pathname:** C:\4DFP\bases\BasesPro\MyMusic\MyMusic.4DD
- 4D Server Platform Version:** 4D Server Windows version 5.1
- OS Version:** Windows™ XP
- Server Machine hardware:** INTEL Compatible

Buttons: Cancel, OK

## Laufwerke des Server-Rechners auf einem Client-Rechner anzeigen

Mit den generischen Projektmethoden *Client post request* und *Client get result* zeigt die Projektmethode **M\_SERVER\_VOLUMES** auf dem Client-Rechner die Laufwerke des Server-Rechners an, die die Serverprozedur zurückgibt. Diese Methode kann einem Menübefehl zugeordnet werden oder z.B. über eine Objektmethode für eine Schaltfläche aufgerufen werden:

```

` M_SERVER_VOLUMES
C_BLOB(vxDaten)
` Stelle die Anfrage
$vReqID:=Client post request("LAUFWERK")
` Frage den Anfragestatus ab und erhalte das Ergebnis
$vErrCode:=Client get result($vAnf_Nr;->vxDaten;120)
` Ist die Anfrage erfolgreich abgeschlossen, zeige das Ergebnis an
If($vErrCode=0)

```

```

` Entnehme die Information aus dem BLOB
  $vOffset:=0
  BLOB TO VARIABLE(vxDaten;asVName;$vOffset)
  BLOB TO VARIABLE(vxDaten;arVSize;$vOffset)
  BLOB TO VARIABLE(vxDaten;arVUsedSpace;$vOffset)
  BLOB TO VARIABLE(vxDaten;arVFreeSpace;$vOffset)
  For($vElem;1;Size of array(arVSize))
` Konvertiere von bytes in MB
  arVSize{$vElem}:=arVSize{$vElem}/1048576
  arVUsedSpace{$vElem}:=arVUsedSpace{$vElem}/1048576
  arVFreeSpace{$vElem}:=arVFreeSpace{$vElem}/1048576
  End for
` Zeige das Ergebnis
  DIALOG([SP Anfragen];"LAUFWERK")
Else
  ALERT("Anfrage error "+String($vErrCode))
End if
` BLOB wird nicht länger benötigt
CLEAR VARIABLE(vxDaten)

```

Hier ist das Formular `[SP Anfragen];"LAUFWERK"` in der Ausführung:

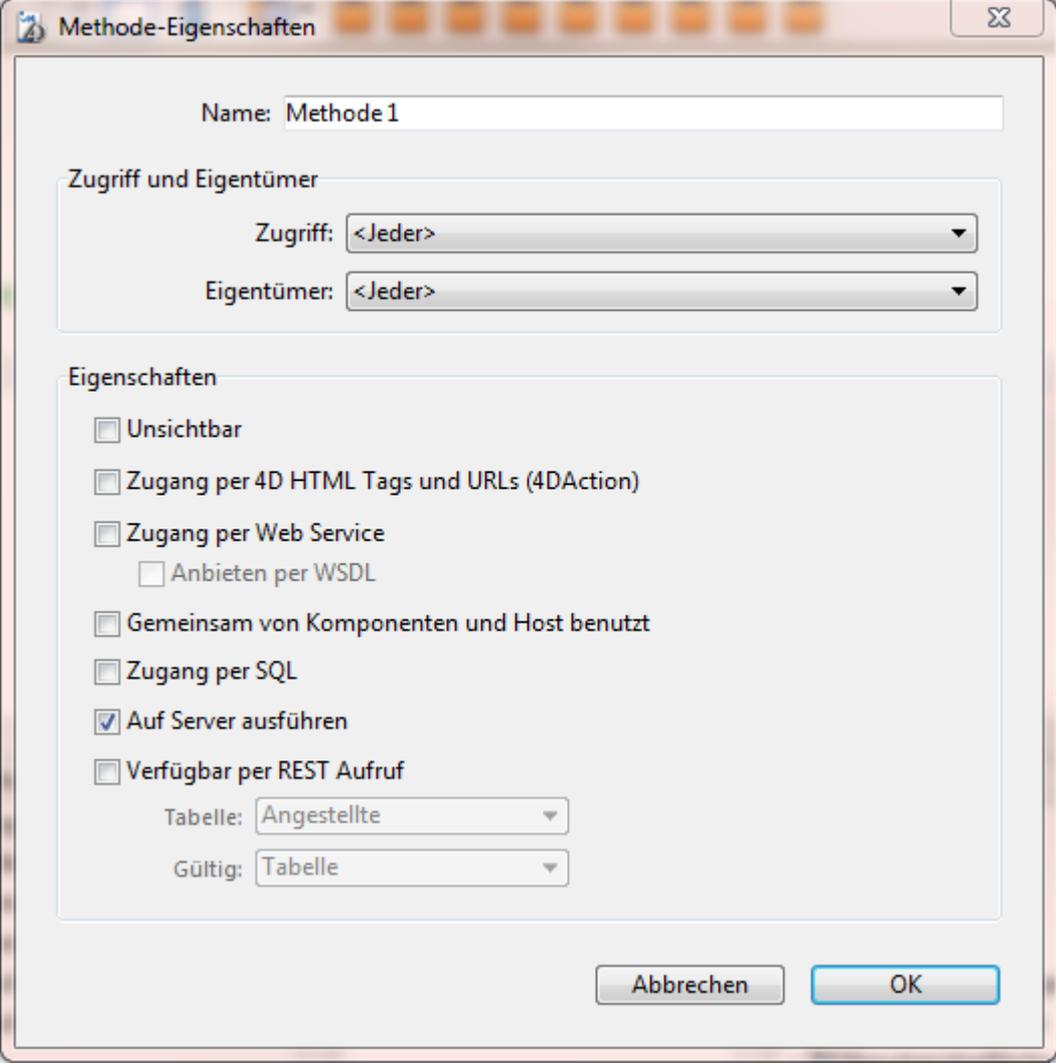
The screenshot shows a dialog box titled "Anwendung" with a table of disk space information. The table has four columns: "Laufwerk", "Speicher", "Benutzter Speicher", and "Freier Speicher". The data is as follows:

Laufwerk	Speicher	Benutzter Speicher	Freier Speicher
R:\	0,00 MB	0,00 MB	0,00 MB
C:\	503,21 MB	388,28 MB	114,92 MB

At the bottom right of the dialog box, there are two buttons: "Abbrechen" and "OK".

## **Attribut Auf Server ausführen**

Die Option "Auf Server ausführen" können Sie im Dialogfenster Methodenauswahl-Eigenschaften und Methode-Eigenschaften festlegen:



The screenshot shows the 'Methode-Eigenschaften' dialog box. The 'Name' field is 'Methode1'. The 'Zugriff und Eigentümer' section has 'Zugriff' and 'Eigentümer' both set to '<Jeder>'. The 'Eigenschaften' section has the following options:

- Unsichtbar
- Zugang per 4D HTML Tags und URLs (4DAction)
- Zugang per Web Service
  - Anbieten per WSDL
- Gemeinsam von Komponenten und Host benutzt
- Zugang per SQL
- Auf Server ausführen
- Verfügbar per REST Aufruf

Below these options, there are two dropdown menus: 'Tabelle' set to 'Angestellte' and 'Gültig' set to 'Tabelle'. At the bottom, there are 'Abbrechen' and 'OK' buttons.

Die Eigenschaft wird nur für eine 4D Anwendung im Client/Server-Modus berücksichtigt.

Ist diese Option markiert, wird die Projektmethode immer auf dem Server ausgeführt, unabhängig davon, wie sie aufgerufen wird.

## **Kontext der Ausführung**

Der Kontext der Ausführung für Projektmethoden ist ähnlich wie für Trigger, siehe **Einführung**: Die Methode auf dem Server verwendet in der Datenbank denselben Kontext zum Sperren von Datensätzen und für Transaktionen wie auf der Client-Seite, aber nicht denselben Kontext der Programmiersprache (Prozessvariablen, Mengen, aktuelle Auswahlen). Im Gegensatz zu Triggern nutzt eine Methode, die auf dem Server ausgeführt wird, den aktuellen Datensatz nicht gemeinsam mit dem Client-Kontext.

Alle Parameter der Methode (\$1, \$2, etc.) werden an den Server gesendet und der Wert des Parameters \$0 - falls übergeben - wird an den Client zurückgegeben.

Im Gegensatz zur Funktion **Execute on server** erstellt diese Option keinen Prozess auf dem Server. 4D Server verwendet den "Zwillingsprozess" des Client-Prozesses, der die Ausführung anfordert.

Diese Option vereinfacht außerdem das Delegieren der Ausführung einer Methode auf dem Server, da die Parameter automatisch in beiden Richtungen übertragen werden, wie beim normalen Aufruf einer Methode.

Die Funktion **Execute on server** arbeitet asynchron, benötigt deshalb mehr Programmierung und verwendet Semaphoren zum Lesen der Ergebnisse.

## Verwendbare Befehle

---

Für Methoden mit dem Attribut "Auf Server ausführen" gelten dieselben Regeln wie für Serverprozeduren. Für einige Befehle ist die Ausführung auf dem Server nicht erlaubt, für andere wird die Ausführung auf dem Server nicht empfohlen. Weitere Informationen dazu finden Sie im Abschnitt **Serverprozeduren**.

## Zeiger

---

Übergeben Sie einen Zeiger für eine Variable (einfache Variable, Array oder Array-Element), wird der Wert mit Zeiger auch an den Server gesendet. Wird er auf dem Server durch eine Methode geändert, wird der geänderte Wert an den Client zurückgesendet, um die entsprechende Variable auf der Client-Seite zu aktualisieren.

Zeiger auf eine Tabelle oder ein Datenfeld werden als Referenzen gesendet (Tabellennummer, Datenfeldnummer). Der Wert des aktuellen Datensatzes wird nicht automatisch ausgetauscht.

**Hinweis:** Diese Option arbeitet im interpretierten und im kompilierten Modus gleich.

## Beispiel

---

Hier ein Beispiel für die Projektmethode Myappli mit dem Attribut "Auf Server ausführen":

```
C_POINTER($1) ` Zeiger auf Tabelle
C_POINTER($2) ` Zeiger auf Feld
C_POINTER($3) ` Zeiger auf Array
C_TEXT($4) ` Wert, nach dem gesucht wird
C_LONGINT($0) ` Ergebnis
```

```
` Suchen a Werte für jeden Datensatz zurücksenden
```

```
QUERY($1->;$2->=$4)
While(Not(End selection($1->)))
  APPEND TO ARRAY($3->;myFormula($1))
  NEXT RECORD($1->)
End while
```

```
UNLOAD RECORD($1->)
```

```
$0:=Records in selection($1->)
```

Auf der Client-Seite wird die Methode folgendermaßen aufgerufen:

```
ARRAY TEXT(myArray;0)
```

```
$vlnum :=MyAppli(->[Table_1] ;->[Table_1]Field_1 ;->myArray;"to find")
```