

# 4D Server - Référence

 Introduction

 4D Server en 10 minutes

 Utilisation de 4D Server

 Fenêtre d'administration de 4D Server

 Méthodes base 4D Server

 Utilisation d'un 4D distant

 4D Server et le langage 4D

# Introduction

 Présentation

 Architecture de 4D Server

## Présentation

4D Server est le serveur de données et d'applications multi plate-forme de 4D.

Avec 4D Server, vous pouvez créer et utiliser des bases de données multi-utilisateurs ainsi que des applications personnalisées en architecture client/serveur. L'architecture client/serveur indépendante de plate-forme de 4D Server gère des applications pour les 4D sur PC Windows et Macintosh. 4D Server est doté de puissants outils de développement et de sécurité des données, est évolutif et se connecte à tous les systèmes d'entreprise.

L'architecture de 4D Server est totalement intégrée : le client et le serveur utilisent une application 4D unique, et les développeurs n'ont pas à concevoir deux développements spécifiques — un pour le serveur et un pour les clients. En outre, 4D Server est un serveur "zéro administration". Il est simple à installer, utiliser, administrer et permet de mettre en place des applications peu coûteuses et rapidement rentables.

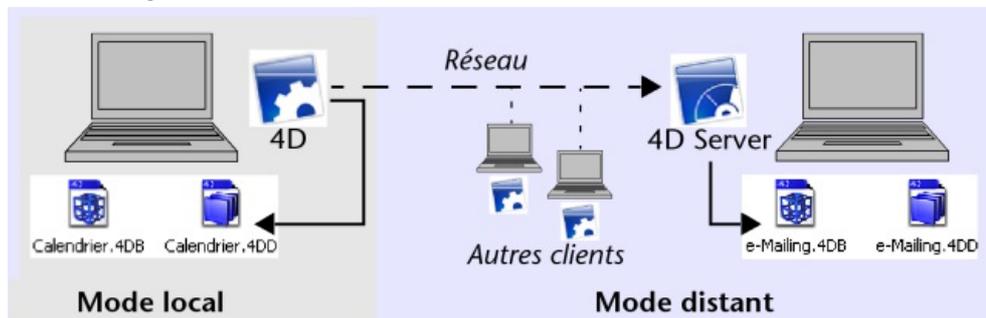
4D Server comble le fossé existant entre les systèmes de gestion basés sur le partage de fichiers, lents et peu efficaces, et les hôtes SQL, certes puissants mais complexes à concevoir, à déployer, et à faire évoluer dans le temps. Une application 4D Server peut s'intégrer facilement aux systèmes d'information existants dans les entreprises (tels que Oracle, Sybase, ou tout serveur compatible ODBC).

### Architecture client/serveur intégrée

Un système 4D Server fonctionne avec une application unique pour le serveur et le client. Le logiciel client et l'application serveur sont les composants d'un seul produit, 4D. L'application 4D Server se compose de deux éléments : 4D Server et 4D en mode distant, qui forment l'architecture client/serveur.

La partie 4D Server réside sur la machine serveur. Elle stocke et gère la base de données sur le serveur et permet aux utilisateurs de manipuler la base à partir de leur propre machine (la machine cliente — ou poste client).

L'application 4D réside sur chaque machine cliente. Elle peut être utilisée en mode local ou en mode distant. En mode local, les utilisateurs peuvent travailler avec une base de données ou une application 4D stockée localement sur leur poste. En mode distant, les utilisateurs s'en servent pour accéder à la base de données sur le serveur, et exécuter des opérations de base de données : ajouter de données, créer des états, ou encore modifier la structure de la base de données. Tout ce qui peut être réalisé avec 4D en local est également faisable avec 4D Server et 4D en mode distant.



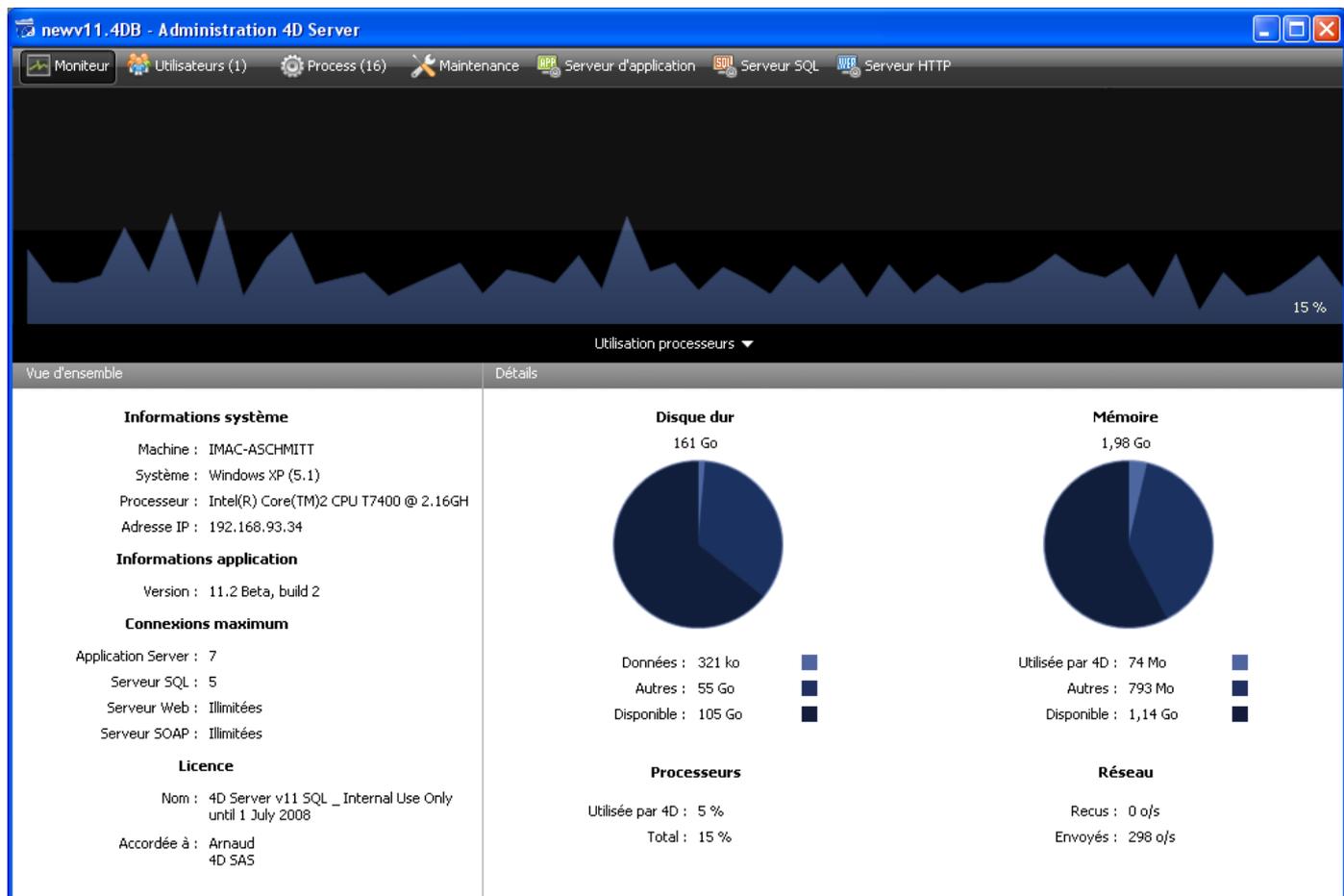
Aucun middleware ni développement supplémentaires ne sont nécessaires pour fonctionner en environnement client/serveur. 4D et 4D Server partagent les mêmes outils d'interface, le même langage ainsi que le même système de gestion de l'information.

Une application monoposte locale évolue facilement vers une configuration client/serveur pour le groupe de travail.

Réciproquement, lorsque cette application est développée avec 4D Server, elle est de fait une application monoposte avec 4D en mode local.

### Serveur de données et d'applications "Zéro Administration"

4D Server est, comme 4D, centré sur l'utilisateur. 4D Server est un système Plug & Play (PnP) complet.



Fenêtre d'administration de 4D Server

- **Administration centralisée en ligne, graphique :** La fenêtre d'administration de 4D Server affiche automatiquement toutes les informations essentielles concernant l'activité de l'application : mémoire totale allouée à 4D Server, cache de données, nombre et nom des utilisateurs connectés, nombre de process et statut de chacun, suivi des sauvegardes et des requêtes, activité des serveurs.
- **Auto-configurable et évolutif :** 4D Server est conçu pour intégrer de nouveaux protocoles, clients, plug-ins, et pour s'interfacer avec d'autres systèmes, sans qu'il soit nécessaire de le reconfigurer ou d'en modifier la conception.
- **Mise à jour automatique et dynamique des postes clients, contrôle de versions :** Tous les 4D connectés reçoivent automatiquement et de manière dynamique les nouveaux éléments d'une application chaque fois que la base évolue ou qu'un plug-in ou un composant est ajouté, supprimé ou modifié. En outre, si vous avez construit une application client-serveur personnalisée, vous avez la possibilité de télécharger automatiquement les nouvelles versions des clients 4D exécutables en cas de mise à jour de l'application 4D Server.
- **Connexions automatiques asynchrones via le protocole standard TCP/IP :** 4D Server et 4D communiquent de manière transparente via le protocole réseau TCP/IP, quelle que soit la plate-forme du client et du serveur. Le protocole TCP/IP étant intégré à tous les systèmes d'exploitation, son utilisation ne nécessite aucune installation particulière.
- **Gestion simultanée des sessions et des contextes des connexions 4D, SQL et HTTP :** Le moteur de base de données de 4D Server crée et maintient automatiquement un environnement de travail courant pour chaque combinaison table/process/utilisateur. Cette architecture basée sur les sessions permet à chaque process utilisateur de manipuler les données indépendamment et simultanément. Le serveur SQL de 4D Server prend en charge automatiquement les requêtes SQL internes ou externes. Le serveur HTTP de 4D Server répond aux requêtes HTTP et éventuellement aux requêtes SOAP.
- **Verrouillage automatique des enregistrements :** 4D verrouille et libère automatiquement les enregistrements. Cette fonction élimine les problèmes courants associés à la modification des enregistrements "en utilisation". Le verrouillage des enregistrements rend caducs les problèmes associés au verrouillage des pages ou fichiers qui sont généralement rencontrés dans d'autres systèmes.
- **Système de messagerie utilisateur intégrée :** 4D Server est né de la micro-informatique. Son interface utilisateur répond aux besoins des environnements de développement intégrés (IDE) modernes. Par exemple, 4D Server est capable d'informer les clients des actions d'administration en cours : déconnexions, sauvegardes, etc.
- **Méthodes de démarrage et de déconnexion automatisées :** 4D Server appelle automatiquement plusieurs méthodes base répondant à des événements spécifiques : Sur démarrage serveur, Sur arrêt serveur, Sur ouverture connexion serveur, Sur fermeture connexion serveur et Sur connexion Web. Par exemple, la **Méthode base Sur démarrage serveur** peut initialiser et charger automatiquement tous les objets qui seront utilisés pendant la session.

## Un jeu de fonctions inégalées

En plus des fonctionnalités de 4D, 4D Server possède les caractéristiques suivantes :

- **Gestion des données en environnement multi-utilisateurs :** Plusieurs utilisateurs peuvent simultanément effectuer des

opérations de bases de données sur la même table ou sur des tables différentes : ajouter, modifier, supprimer, rechercher, trier et imprimer des enregistrements. L'intégrité des données est assurée par un système interne de verrouillage des enregistrements.

- **Développement multi-utilisateurs** : Plusieurs utilisateurs peuvent simultanément développer et concevoir une base de données. Par exemple, les membres d'une équipe peuvent, en même temps, modifier les attributs d'une table, créer et modifier des formulaires et des méthodes. L'intégrité de la structure est protégée grâce à un système interne de verrouillage des objets.
- **Architecture client/serveur indépendante de plate-forme** : Cette architecture gère les performances de la base de données de manière identique pour les clients Windows et Mac OS, que ce soit en matière de multi-développement cross-plate-forme simultané ou pour les opérations de saisie et de modification de données par des postes clients sur des environnements matériels différents.
- **Architecture de plug-ins Windows et Mac OS** : La version Windows et la version Mac OS de 4D Server permettent d'installer à la fois des plug-ins Windows et Mac OS sur le poste serveur. Cette architecture simplifie la distribution de plug-ins 4D indépendants de plates-formes : ils sont traités de manière transparente par 4D et 4D Server, quelle que soit la plate-forme d'exécution du client.
- **Serveur HTTP intégré** : Tout comme 4D en mode local, 4D Server et chaque 4D en mode distant possèdent un moteur HTTP qui vous permet de publier les bases 4D sur le Web. Votre base de données peut être directement publiée sur le Web. Vous n'avez pas besoin de développer un système de base de données, un site Web ou une interface CGI entre eux. Votre base de données est votre site Web. Vous pouvez également transformer tout poste 4D distant en serveur Web. Pour plus d'informations concernant le moteur Web intégré de 4D Server et 4D, reportez-vous au chapitre **Présentation du serveur Web** dans le manuel Langage de 4D.
- **Sécurité des connexions via SSL (Secured Socket Layer)** : Vous pouvez configurer votre 4D Server de manière à ce que les connexions client/serveur s'effectuent en mode sécurisé, par l'intermédiaire du protocole SSL. Pour plus d'informations les connexions client/serveur sécurisées, reportez-vous à la section **Crypter les connexions client/serveur**.
- **Triggers** : Un trigger est une méthode associée à une table. C'est une des propriétés de la table. Vous n'appellez pas un trigger, il est automatiquement appelé par le moteur de la base de données chaque fois que vous manipulez des enregistrements (ajout, suppression et modification). Avec 4D Server, les triggers sont exécutés sur le poste serveur. Tout client, qu'il soit 4D ou connecté via ODBC, est assujéti aux règles de la base de données contrôlées par les triggers. Pour plus d'informations sur les triggers, reportez-vous à la section **Présentation des triggers** dans le manuel Langage de 4D.
- **Procédures stockées** : Vous pouvez écrire des méthodes 4D qui seront exécutées en local dans leur propre process sur la machine serveur ou sur un ou plusieurs postes clients que vous désignerez. Ces fonctions sont appelées les procédures stockées, pour utiliser une terminologie couramment employée dans l'industrie du client/serveur. Cependant, 4D possède une architecture qui dépasse le concept standard de procédures stockées. Avec 4D Server, une procédure stockée est en réalité un process serveur (ou un process client, cf. paragraphe suivant) qui exécute le code de manière asynchrone, et indépendamment de tous les autres process exécutés sur les postes client ou serveur. Dans une architecture client/serveur standard, une procédure stockée exécute et retourne (de manière synchrone ou asynchrone) un résultat. Avec 4D Server, vous pouvez démarrer une procédure stockée qui s'exécute pendant toute une session client/serveur, et qui répond, à la demande, aux messages qui lui sont transmis par les clients. Simultanément, vous pouvez lancer une autre procédure stockée qui n'aura aucune interaction avec les clients, mais qui se chargera de synchroniser les données avec un serveur SQL ou un autre 4D Server, via un plug-in de connectivité 4D ou ODBC. La seule limite au nombre de procédures stockées exécutables simultanément est celle de votre configuration matérielle. Une procédure stockée s'exécute dans son propre process. Donc, comme chaque process utilisateur, elle maintient son propre contexte de base de données (sélection courante). En outre, le langage de 4D possède des commandes qui permettent aux process clients (exécutés sur les postes clients) de lire et d'écrire des variables process dans toute procédure stockée (y compris les variables BLOB). Elles peuvent donc servir à mettre en place un système de communication efficace et puissant entre les clients et les procédures stockées. Avec les procédures stockées, vous pouvez ajouter des nouveaux services personnalisés dans 4D Server. Pour plus d'informations, reportez-vous à la section **Procédures stockées**.
- **Procédures stockées exécutées sur client** : 4D Server vous permet, à partir d'un poste client ou du serveur, d'exécuter des procédures stockées sur un ou plusieurs autres postes clients. Vous pouvez ainsi optimiser la répartition des charges de travail entre les postes clients et le serveur, ou construire des applications tirant parti des possibilités de communication inter-clients. Pour plus d'informations, reportez-vous à la section **Procédures stockées**.
- **Chemin d'accès au serveur** : Le chemin d'accès à une base de données serveur peut être enregistré avec un mot de passe utilisateur. Cette fonction permet à un utilisateur de se connecter à une base de données sur le serveur en double-cliquant simplement sur un document .4DLink. Pour plus d'informations, reportez-vous à la section **Connexion à une base 4D Server**.
- **Enregistrement comme service** : sous Windows, 4D Server peut être lancé comme un Service au démarrage.
- **Système de sauvegarde intégré** : 4D Server inclut un module complet de sauvegarde des bases de données et de récupération en cas d'incident. Ce module permet de sauvegarder une base en cours d'exploitation, sans qu'il soit nécessaire de quitter l'application. Les sauvegardes peuvent être déclenchées manuellement ou automatiquement, à intervalles réguliers et sans intervention de l'utilisateur. En cas d'incident, la restitution et/ou le redémarrage de la base peuvent également être déclenchés automatiquement.
- **Sauvegarde par miroir logique** : Dans le cadre d'applications critiques, il est possible de mettre en place un système de sauvegarde par miroir logique, permettant un redémarrage instantané en cas d'incident sur la base en exploitation.
- **Plug-ins de connectivité** : Avec les plug-ins de connectivité 4D tels que 4D ODBC Pro, 4D Server et 4D accèdent directement à des bases sur mini ou grands systèmes, tels que ORACLE ou toute source de données ODBC. 4D propose également un pilote 4D Server ODBC (Driver ODBC) qui permet à tout client ODBC de se connecter à 4D Server pour travailler avec les données.

## Architecture de 4D Server

Avec son architecture client/serveur, 4D Server ne se contente pas de stocker et de gérer la base de données, mais fournit également des services aux clients. Ces services fonctionnent à travers le réseau par l'intermédiaire d'un système de requêtes et de réponses.

Pour rechercher un ensemble d'enregistrements, par exemple, un poste client envoie une requête au serveur. Dès réception de la requête, 4D Server exécute la recherche en local (c'est-à-dire sur le poste serveur) et, lorsqu'elle est terminée, en retourne le résultat (les enregistrements trouvés).

L'architecture de 4D Server est basée sur le modèle client/serveur. Depuis de nombreuses années, le modèle d'architecture client/serveur s'est imposé, face à son concurrent plus ancien, le partage de fichiers, comme le plus efficace pour les bases de données multi-utilisateurs.

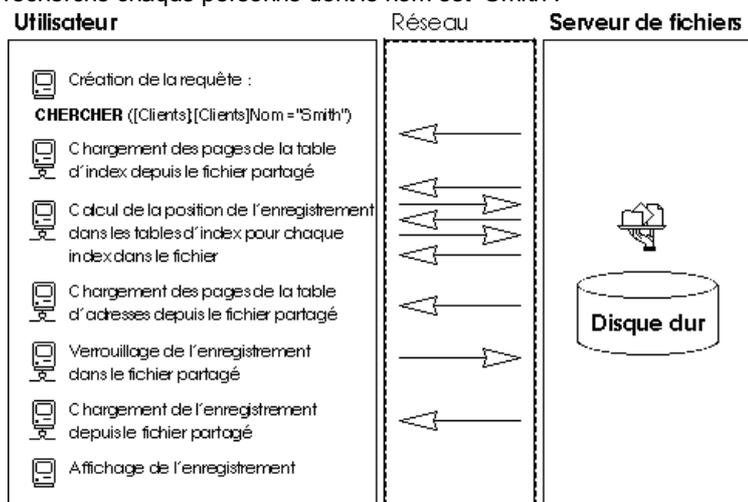
Le type d'architecture client/serveur de 4D Server est comparable à celui qui est utilisé dans le monde de la mini-informatique. De plus, 4D Server apporte deux innovations importantes :

- Une interface intuitive et graphique, présente à tous les niveaux de la base,
- Une architecture intégrée, permettant d'accroître l'efficacité et la vitesse.

## L'architecture Partage de fichiers

Avant l'apparition de l'architecture client/serveur, les systèmes multi-utilisateurs exploitaient le partage de fichiers comme modèle d'architecture réseau. Dans ce modèle, tous les utilisateurs partagent les mêmes données mais la gestion des données n'est pas contrôlée par un moteur de base de données central. Chaque poste client doit stocker une copie de la structure et du moteur de la base, tandis que le serveur n'est chargé que de la gestion du logiciel de partage de fichiers sur le réseau.

Dans le modèle du partage de fichiers, chaque station de travail effectue en local toutes les actions de modification sur les données. Cela a pour conséquence de créer un trafic réseau très important, car chaque requête nécessite de nombreuses communications à travers le réseau. Le schéma suivant présente un exemple de trafic réseau généré lorsqu'un utilisateur recherche chaque personne dont le nom est "Smith".



Autre inconvénient du modèle du partage de fichiers : l'impossibilité d'utiliser un cache mémoire pour conserver des enregistrements en mémoire. Si des enregistrements étaient conservés en mémoire, il pourrait exister différentes versions du même enregistrement stocké dans la mémoire cache, ce qui rendrait les données incohérentes. Par conséquent, chaque fois qu'un utilisateur accède à un enregistrement, celui-ci doit être téléchargé depuis le serveur de fichiers. Cela provoque un trafic réseau intense et augmente le temps d'accès aux données.

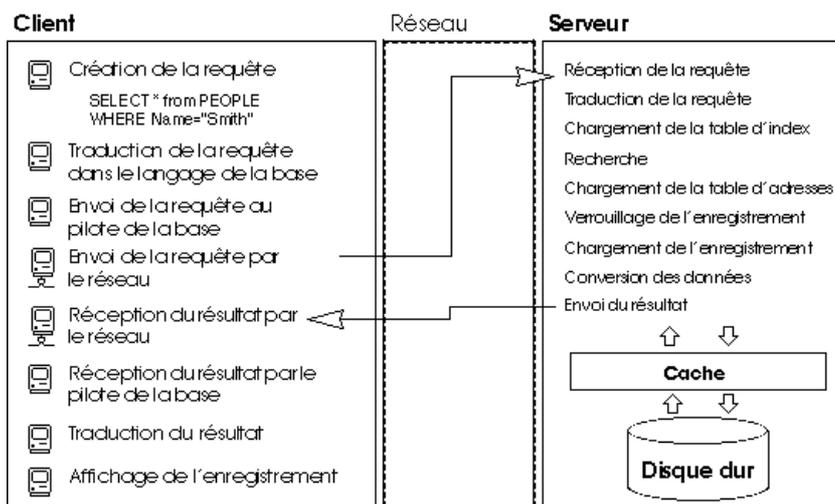
## L'architecture client/serveur hétérogène

L'architecture client/serveur est largement répandue dans le monde de la mini-informatique, pour la gestion de bases de données volumineuses, grâce à son efficacité et à sa vitesse. Avec cette architecture, le travail est divisé entre les clients et le serveur de manière à augmenter les performances.

Le serveur contient le moteur central de la base, qui stocke et gère les données. Le moteur de la base est le seul logiciel ayant accès aux données stockées sur le disque. Lorsqu'un client envoie une requête au serveur, le serveur retourne le résultat. Le résultat peut être de toute nature, depuis un simple enregistrement à modifier jusqu'à une liste triée d'enregistrements.

En général, la plupart des architectures client/serveur sont appelées architectures hétérogènes car les applications frontales exécutées sur les postes clients et le moteur de base de données exécuté sur le serveur sont deux produits différents. Dans cette situation, un pilote de base de données est requis pour servir de traducteur entre les clients et le serveur.

Par exemple, pour rechercher un enregistrement, un client envoie une requête au serveur. Comme la base est stockée sur le serveur, celui-ci exécute la commande en local et expédie le résultat au client. Le schéma suivant présente un exemple de trafic réseau généré lorsqu'un utilisateur recherche chaque personne dont le nom est "Smith" et affiche le premier enregistrement trouvé.



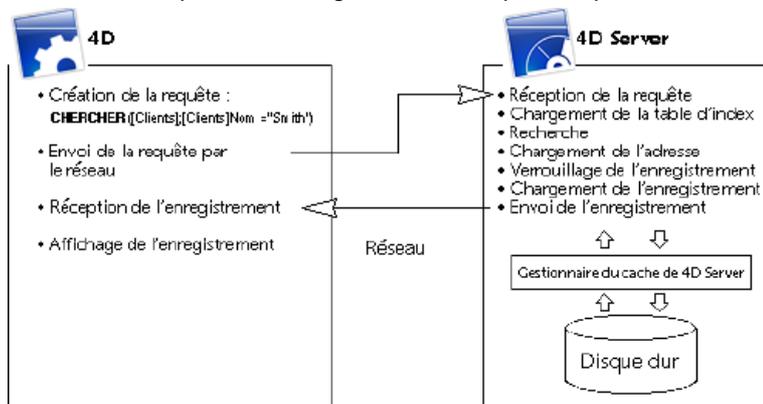
Cet exemple illustre deux différences majeures entre le partage de fichiers et l'architecture client/serveur :

- **L'architecture client/serveur autorise l'utilisation d'un cache** : Comme le moteur est le seul logiciel qui dispose de l'accès physique aux données, le serveur peut utiliser un cache qui conserve en mémoire les enregistrements modifiés jusqu'à ce qu'ils soient écrits sur le disque. Comme les données sont envoyées depuis un site central, les postes clients sont assurés de toujours recevoir la dernière version d'un enregistrement. En plus du contrôle de l'intégrité des données qu'il procure, le mécanisme de cache central accélère les opérations de base de données en remplaçant les accès disque par des accès mémoire. Avec le partage de fichiers, tous les accès sont des accès disque.
- **Les opérations de base de données de bas niveau sont effectuées sur le serveur** : L'architecture client/serveur permet une augmentation importante de la vitesse d'exécution, car les manipulations de bas niveau sur la base de données, telles que l'examen des tables d'index et d'adresses, sont exécutées localement sur le serveur, à la vitesse de la machine. Avec le partage de fichiers, les mêmes opérations sont ralenties par les transferts sur le réseau et les limites du poste client.

## L'architecture client/serveur intégrée de 4D Server

Dans la plupart des architectures client/serveur, l'application cliente et l'application serveur sont deux produits séparés, nécessitant une couche de communication pour pouvoir se comprendre entre eux. Avec 4D Server, l'architecture client/serveur est entièrement intégrée. 4D Server et 4D sont deux applications qui partagent la même structure et communiquent directement.

Comme 4D Server et 4D parlent la même langue, il est inutile de traduire les requêtes. La division du travail entre le client et le serveur est transparente et est gérée automatiquement par 4D Server.



La division du travail est organisée de telle manière qu'à une requête est associée une réponse. Comme vous pouvez le constater dans le schéma ci-dessus, le client est chargé de traiter les tâches suivantes :

- **Requêtes** : Le client 4D envoie des requêtes à 4D Server. Ces requêtes peuvent être construites à l'aide des éditeurs intégrés, tels que l'éditeur de recherches et l'éditeur de tris, à l'aide du langage intégré de 4D ou via le SQL. 4D dispose d'éditeurs dans lesquels les méthodes peuvent être créées et modifiées. Il gère également les éléments des méthodes telles que les variables et les tableaux.
- **Réception des réponses** : Le client 4D reçoit des réponses de 4D Server et en informe l'utilisateur par l'intermédiaire de l'interface utilisateur (des enregistrements différents sont affichés dans un formulaire, etc.). Par exemple, si le client recherche tous les enregistrements dont le nom est "Smith", 4D reçoit les enregistrements de 4D Server et les affiche dans un formulaire.

Le serveur est chargé de traiter les tâches suivantes :

- **Gestion des accès** : 4D Server gère toutes les connexions simultanées et les process créés par les clients. Cette gestion tire parti de l'architecture multi-tâche de 4D Server.
- **Objets de structure et de données** : 4D Server stocke et gère tous les objets de structure et de données, y compris les champs, les enregistrements, les formulaires, les méthodes, les barres de menus et les listes.
- **Cache** : 4D Server gère le cache contenant les enregistrements ainsi que des objets de données créés par les postes

clients, tels que les sélections et les ensembles.

- **Opérations de base de données de bas niveau** : 4D Server exécute les opérations de base de données dites "de bas niveau", telles que les recherches et les tris, qui impliquent l'utilisation des tables d'index et d'adresses.

Cette division du travail est extrêmement efficace grâce à l'intégration unique 4D Server et 4D. L'intégration de l'architecture de 4D Server est présente à chaque niveau :

- **Au niveau de la requête** : Lorsque 4D envoie à 4D Server une requête, telle qu'une recherche ou un tri, 4D envoie une description de l'opération de recherche ou de tri en utilisant la même structure interne que 4D Server.
- **Au niveau de la structure et des données** : Lorsque 4D et 4D Server échangent un objet de structure ou de données, les deux applications utilisent le même format interne. Lorsque 4D a besoin d'un enregistrement, par exemple, 4D Server envoie directement les données dans le format où elles sont stockées sur le disque ou dans le cache mémoire. De la même manière, lorsque 4D met à jour un enregistrement et envoie les données à 4D Server, celui-ci les stocke directement dans le cache exactement telles qu'il les a reçues.
- **Au niveau de l'interface utilisateur** : Lorsque 4D affiche une liste d'enregistrements, le formulaire utilisé joue un rôle dans l'architecture client/serveur. Par exemple, la fenêtre suivante montre le résultat d'une requête dans la table [Sociétés].



Réf société	Nom société	Téléphone	Fax	Réf ville
ALL	Allinone	+358 9 564		HEL
HYP	Hyperbureau	+33 01 456		PAR
MET	Metaltip	+39 02 789		MIL
ELCO	El Computador	+34 91 147		MAD
NIP	Nippon United	+81 3 258		TOK
GOR	Gorky Town	+00011144		MOS
HUI	Hychen Union	+654 3210		LUX
GOU	Gourdin Industries, Inc	+555 111 5		WAS
GLU	Glup Technologies	+00 44777		MIL
KOA	Koala Enterprises	+6545454		MEL
BRO	Broceliande	+789456		VAR
KLI	Klick	+666 555 4		MIA

Comme la taille de la fenêtre ne permet d'afficher que douze enregistrements et cinq champs à la fois, 4D Server envoie exactement douze enregistrements. Au lieu d'envoyer la totalité des enregistrements, 4D Server n'envoie que le nombre d'enregistrements et de champs pouvant être affichés dans la fenêtre. Si l'utilisateur fait défiler les enregistrements dans le formulaire, 4D Server envoie les enregistrements et les champs supplémentaires à mesure qu'ils apparaissent dans la fenêtre. Cette optimisation réduit le trafic réseau, car les enregistrements et les champs ne sont envoyés que lorsque c'est nécessaire.

## **4D Server en 10 minutes**

-  Vérification de l'installation
-  Créer une base serveur
-  Connexion à la base serveur avec un 4D distant
-  Définir la structure de la base
-  Traitement des données avec 4D Server
-  Créer une barre de menus personnalisée
-  Travailler simultanément avec plusieurs 4D distants

## Vérification de l'installation

---

Le chapitre d'initiation **4D Server en 10 minutes** vous permet de découvrir rapidement 4D Server. Vous verrez en particulier comment :

- Créer une base serveur
- Connecter un client à la base serveur
- Créer la structure de la base, comprenant des tables, des champs, des formulaires, des menus et des méthodes
- Connecter un second client et travailler simultanément

Pour effectuer ces exercices, vous devez disposer au minimum de deux ordinateurs :

- un ordinateur sur lequel 4D Server, 4D et un navigateur Web sont installés,
- un autre ordinateur sur lequel 4D est installé.

Avant de commencer à travailler pour la première fois avec 4D Server et 4D en mode distant, nous vous conseillons de vérifier votre installation. Pour cela, lisez la présente section.

### Eléments installés

---

Ce paragraphe précise l'emplacement des éléments installés sur votre disque à la suite d'une installation standard 4D + 4D Server.

### Windows

---

Les éléments ont été installés dans le dossier **Program Files\4D\4D v11 SQL release N** et apparaissent dans le menu **Démarrer**.

- *4D Server* : ce dossier contient l'application 4D Server ainsi que ses fichiers et dossiers associés. Pour lancer 4D Server, il vous suffit de double-cliquer sur le fichier *4D Server.exe*.
- *4D* : ce dossier contient l'application 4D ainsi que ses fichiers et dossiers associés. Pour lancer 4D, il vous suffit de double-cliquer sur le fichier *4D.exe*.

### Mac OS

---

Les éléments ont été installés dans le dossier **Applications:4D:4D v11 SQL release N** et apparaissent dans les applications.

- *4D Server* : progiciel (package) de 4D Server. Pour lancer 4D Server, il vous suffit de double-cliquer sur ce progiciel.
- *4D* : progiciel (package) de 4D. Pour lancer 4D, il vous suffit de double-cliquer sur ce progiciel.

Dans le cadre de cet exercice, vous devez effectuer une installation de 4D sur un poste supplémentaire.

### Et maintenant...

---

Notez que le protocole TCP/IP doit être correctement configuré pour que vos machines puissent communiquer à travers le réseau. Si 4D Server et 4D sont correctement installés, continuez avec la section **Créer une base serveur**. Sinon, si certains des fichiers listés ci-dessus sont manquants, reportez-vous au Guide d'installation et procédez à l'installation de ces fichiers.

## Créer une base serveur

Cette section décrit la création d'une base serveur à laquelle vous pourrez vous connecter, par l'intermédiaire du réseau, avec 4D en mode distant. Si vous utilisez 4D Server et 4D pour la première fois, n'hésitez pas en premier lieu à vérifier votre installation. Pour cela, reportez-vous à la section [Vérification de l'installation](#).

**Note :** Dans le cadre de cet exemple, nous supposons que vous avez déjà activé votre licence 4D Server, comme décrit dans le Guide d'installation. L'emploi de 4D en mode distant ne nécessite pas de licence sur le poste client. Les licences sont gérées sur la machine de 4D Server. Pour plus d'informations, reportez-vous au Guide d'installation.

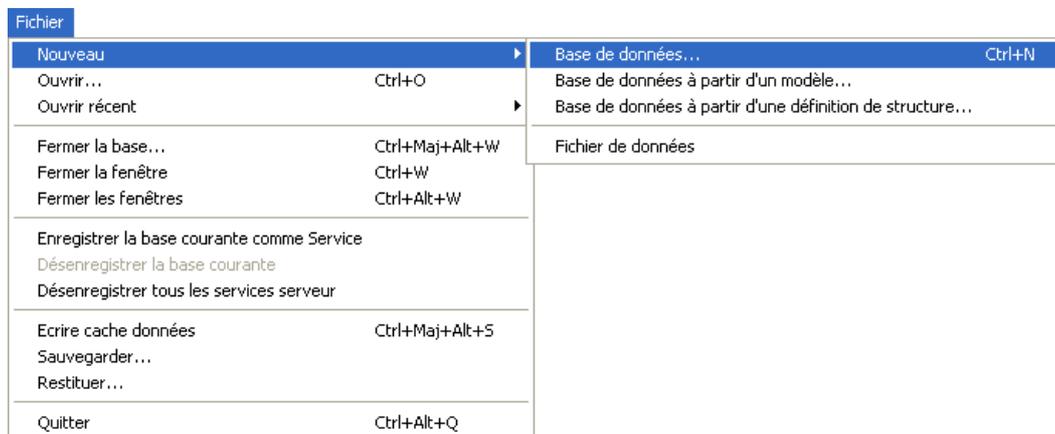
Pour créer ou ouvrir une base serveur, lancez 4D Server.

### 1. Lancez 4D Server en double-cliquant sur l'icône du programme.



Lors du tout premier lancement de l'application, la boîte de dialogue d'activation de l'application s'affiche. Par la suite, une fenêtre vide s'affichera à chaque démarrage. Vous pouvez configurer ce fonctionnement dans les Préférences de l'application. Dans cet exercice, nous allons créer une nouvelle base vierge.

### 2. Choisissez la commande Nouveau>Base de données... dans le menu Fichier de 4D Server.



Une boîte de dialogue standard d'enregistrement de fichier apparaît, vous permettant de définir le nom et l'emplacement de la base à créer.

### 3. Définissez un emplacement puis saisissez le nom de votre base.

Tapez **Employés**, puis cliquez sur le bouton **Enregistrer**.

4D Server crée automatiquement les fichiers et dossiers nécessaires au fonctionnement de la base, puis la fenêtre d'administration apparaît :



La **fenêtre d'administration** de 4D Server se compose de plusieurs pages, accessibles via des onglets. La page **Moniteur** affiche des informations dynamiques relatives à l'exploitation de la base de données ainsi que des informations sur le système et l'application 4D Server.

Parmi les autres pages, notez les pages **Utilisateurs** et **Process** indiquant respectivement le nombre d'utilisateurs connectés à la base et le nombre de process en cours d'exécution. A cet instant, le nombre d'utilisateurs connectés est zéro. Cela signifie qu'actuellement, aucun client n'est connecté à la base. Le nombre de process en cours d'exécution est de douze. Ces douze process sont créés automatiquement par le moteur de la base et les serveurs intégrés de 4D Server (serveur d'application, serveur HTTP, serveur SQL).

## Et maintenant...

A ce moment, la base est disponible pour les connexions distantes 4D Windows et/ou Macintosh. En revanche, la base n'est pas encore prête pour les connexions HTTP, car ces connexions ne sont pas autorisées par défaut.

Dans cette initiation, nous allons nous connecter dans un premier temps avec un 4D distant, définir la structure de la base, puis créer quelques enregistrements. Reportez-vous à la section **Connexion à la base serveur avec un 4D distant**.

## Connexion à la base serveur avec un 4D distant

Cette section aborde les sujets suivants :

- Connecter un 4D distant à la base serveur que vous avez créée,
- Créer la structure de la base ; la création des tables et des champs, l'ajout et la modification d'enregistrements sont présentés sous forme de petits exercices d'initiation.
- Connecter un second utilisateur,
- Travailler simultanément avec les deux clients distants.

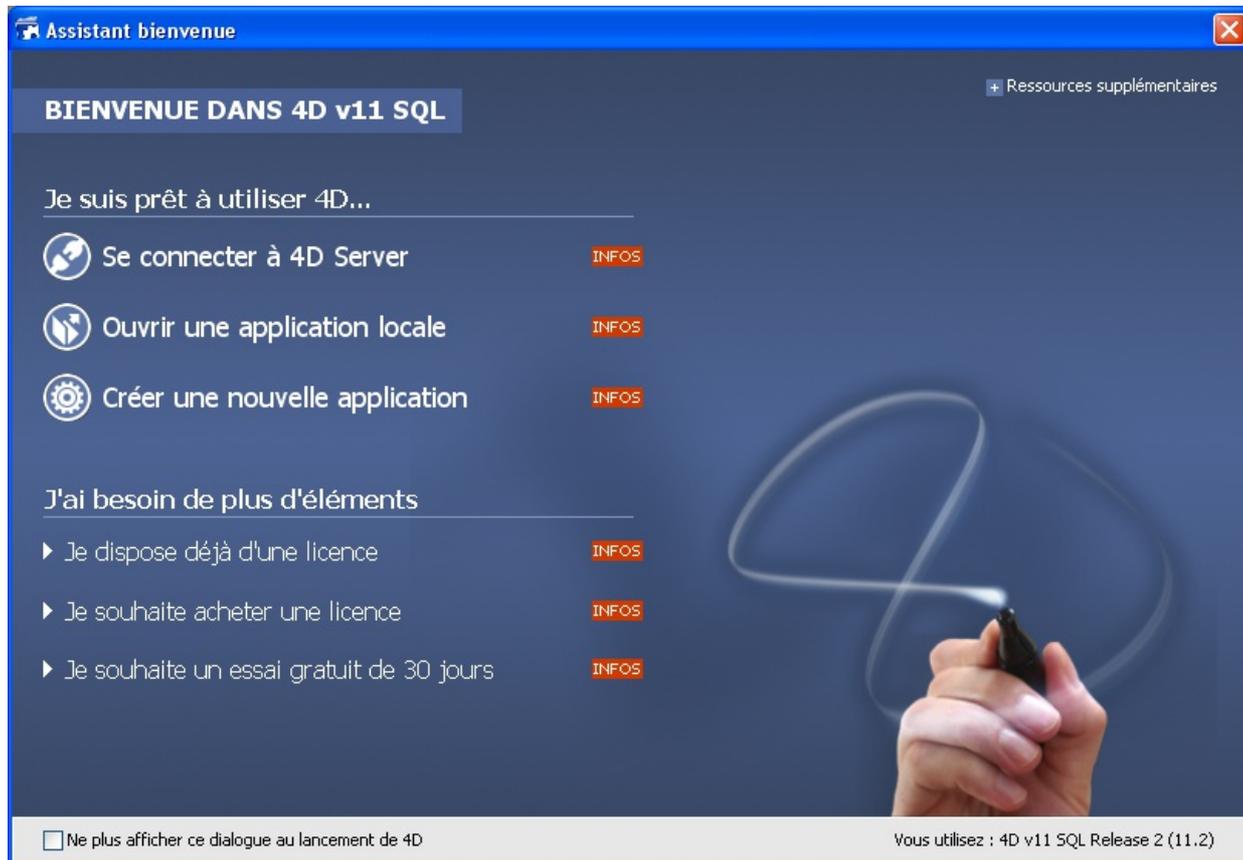
### Connexion à la base

Bien que vous ayez créé la base avec 4D Server (cf. section [Créer une base serveur](#)), toutes les modifications dans le développement et les données de la base sont réalisées à partir des postes clients. Ce paragraphe vous explique comment connecter un poste client à 4D Server et ouvrir la base serveur.

#### 1. Double-cliquez sur l'icône de l'application 4D distante.



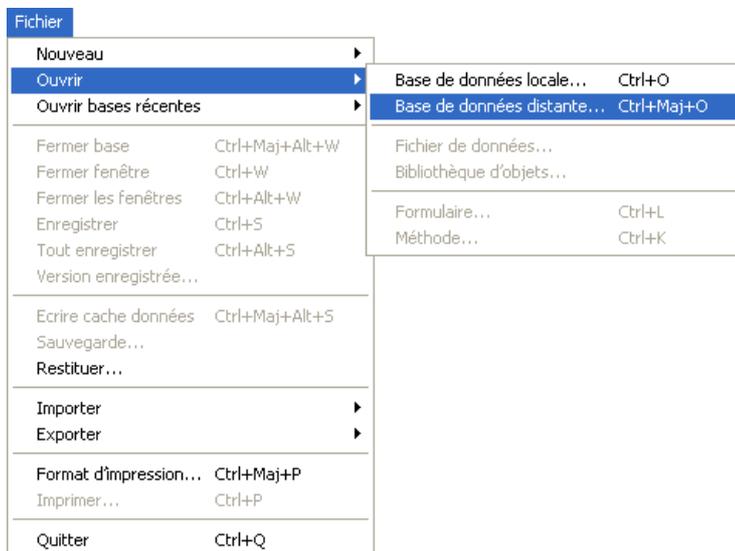
**Note :** Pour les besoins de cette initiation, vous pouvez utiliser une application 4D installée sur la même machine que 4D Server. S'il s'agit du premier lancement de l'application 4D ou si vous n'avez pas modifié les paramètres de démarrage, la boîte de dialogue de bienvenue apparaît :



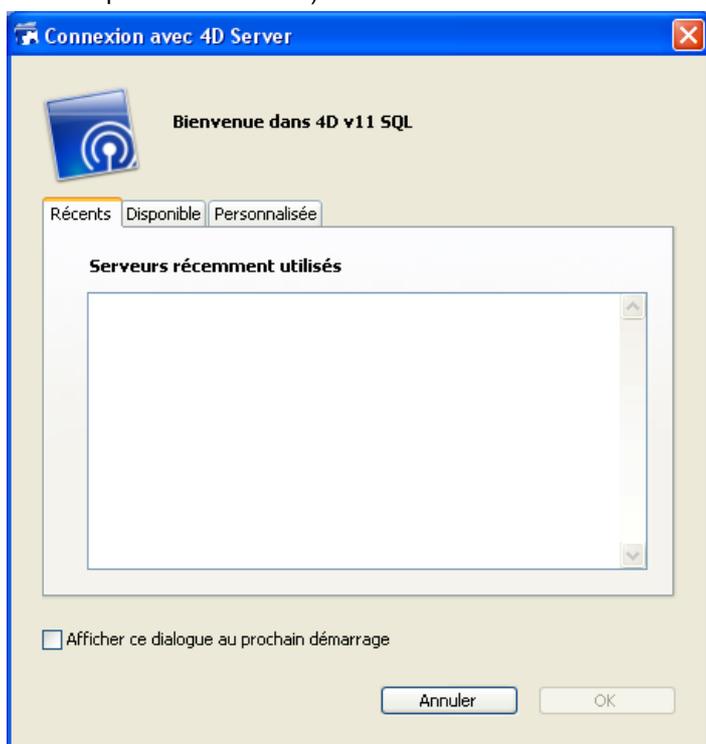
#### 2. Cliquez sur l'option "Se connecter à 4D Server".

OU BIEN :

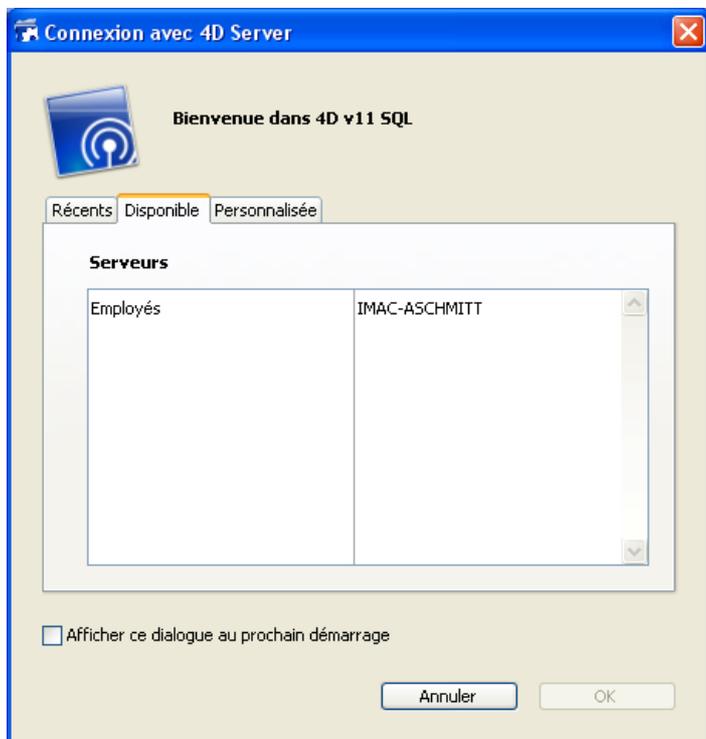
Si cette boîte de dialogue n'apparaît pas, choisissez la commande **Ouvrir>Base de données distante...** dans le menu **Fichier de 4D**



La boîte de dialogue de connexion apparaît. Elle affiche par défaut une page listant les serveurs récemment utilisés (la liste est vide lors de la première utilisation).



**3. Cliquez sur l'onglet Disponible afin d'afficher la liste des bases 4D publiées sur le réseau.**  
La base Employés doit apparaître dans la liste :



#### 4. Sélectionnez Employés et cliquez sur OK.

La base s'ouvre sur le poste distant, en mode Développement. Vous pouvez dès lors définir sa structure.

### Conseils de dépannage :

---

Si vous ne voyez pas le nom de la base que vous venez de créer avec 4D Server, vérifiez chaque point suivant :

- Est-ce que 4D Server est toujours exécuté sur l'autre poste ?
- Si vous utilisez une seconde machine, est-ce que les deux machines sont bien connectées au réseau ?
- Est-ce que le protocole réseau TCP/IP est bien configuré sur les deux postes ?
- Pour plus d'informations sur l'utilisation de la boîte de dialogue de connexion, reportez-vous à la section [Connexion à une base 4D Server](#).

### Activité du serveur

---

Si vous observez maintenant la fenêtre d'administration de 4D Server, vous pouvez noter que votre nom d'utilisateur apparaît dans la page correspondante, et que le nombre d'utilisateurs connectés est désormais un (1).



- le Process principal gère la fenêtre d'affichage des enregistrements et le mode Application.
- le Process développement gère le mode Développement. Chaque utilisateur supplémentaire provoquera la création d'au moins quatre process dans la liste.

Vous pouvez filtrer la liste des process affichés à l'aide des boutons Process utilisateurs, Process 4D et de la zone de filtre situés en haut à droite de la fenêtre d'administration.

## **Et maintenant...**

---

Maintenant que vous êtes connecté, vous pouvez travailler avec la base en ayant à votre disposition les mêmes fonctionnalités que si vous utilisiez 4D en mode local. Vous devez tout d'abord définir la structure de votre base : reportez-vous à la section **Définir la structure de la base**.

## Définir la structure de la base

---

Sur le poste 4D distant, après vous être connecté à la base serveur (cf. section [Connexion à la base serveur avec un 4D distant](#)), choisissez la commande **Structure** dans le menu **Développement**.

La fenêtre de Structure apparaît, vide par défaut. Nous allons créer une simple table.

### Créer la table [Employés]

---

1. Choisissez **Nouveau>Table** dans le menu **Fichier** ou dans la barre d'outils de 4D.

OU

Cliquez avec le bouton droit de la souris dans la fenêtre de Structure et choisissez **Ajouter table** dans le menu contextuel.

OU

Cliquez sur le bouton d'ajout de la fenêtre de Structure (en forme de +) et choisissez **Table**.

La table est créée.

2. Cliquez dans la zone de titre et renommez la table **Employés**.



3. Double-cliquez dans la zone de champs afin de créer un nouveau champ.

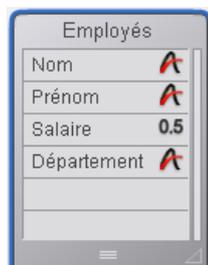


4. Renommez le champ **Nom** et conservez le type **Alpha (255)**.

Vous pouvez double-cliquer sur le champ afin d'afficher la palette de l'Inspecteur.

5. Ajoutez de la même manière les champs suivants dans la table [Employés]

Nom du champ	Type du champ
Prénom	Alpha (255 caractères)
Salaire	Réel
Département	Alpha (255 caractères)



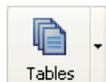
**Note :** Si d'autres applications 4D distantes travaillent simultanément avec la même base, les champs que vous venez de créer apparaissent sur leurs machines quelques instants plus tard. Les modifications sont effectuées sur le serveur en temps réel, mais n'apparaissent pas immédiatement sur les autres écrans afin d'éviter des redessins d'écrans trop fréquents.

### Créer des formulaires pour la table [Employés]

---

Une fois que vous avez défini le contenu de la table [Employés], vous devez créer des formulaires afin de pouvoir saisir et travailler avec des enregistrements pour cette table. Pour cela, vous pouvez utiliser l'**Assistant de création de formulaires** et créer des formulaires comme vous le souhaitez. Cependant, 4D vous propose un raccourci pratique pour créer rapidement des formulaires entrée et sortie par défaut.

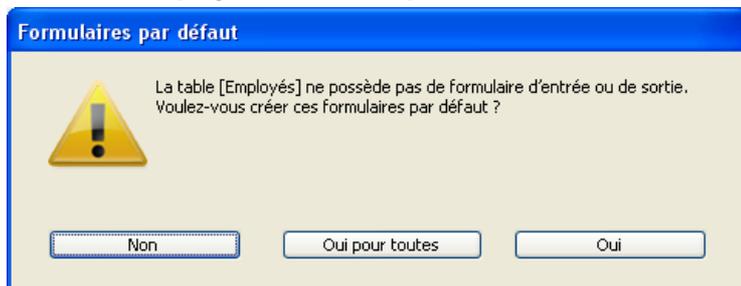
1. Cliquez sur le bouton Tables de la barre d'outils de 4D.



OU BIEN

Choisissez la commande **Afficher la table courante** dans le menu **Enregistrements**.

Vous affichez alors la fenêtre des enregistrements. 4D détecte que la table ne dispose d'aucun formulaire et vous demande si vous voulez laisser le programme les créer pour vous.



2. Cliquez sur le bouton **Oui**.

La table comporte désormais un formulaire entrée pour l'ajout et l'affichage individuel des enregistrements et un formulaire sortie pour l'affichage ou la saisie de plusieurs enregistrements en mode liste.

**Et maintenant...**

---

Votre base serveur est prête pour le traitement des données. Reportez-vous à la section **Traitement des données avec 4D Server**.

## Traitement des données avec 4D Server

Au cours de la section **Définir la structure de la base**, vous avez créé la table *[Employés]* et laissé 4D créer les formulaires par défaut pour cette table. Vous êtes désormais prêt pour saisir des enregistrements.

### Saisir des enregistrements

En mode Développement, 4D vous fournit des outils et éditeurs par défaut pour la saisie, la recherche, l'impression ou la modification des enregistrements. Par la suite, vous pourrez définir vos propres outils pour le mode Application.

#### 1. Choisissez **Nouvel enregistrement** dans le menu **Enregistrements**.

Le formulaire entrée apparaît, vide.

#### 2. Saisissez un premier enregistrement, comme dans l'écran ci-dessous.

Utilisez la touche **Tabulation** ou la souris pour vous déplacer parmi les champs.

Employés - Saisie pour Employés

Employés 0 sur 0

Nom : Poirier

Prénom : Alain

Salaire : 1950

Département : Ingénierie

#### 3. Cliquez sur le bouton de validation du formulaire pour enregistrer votre saisie.

Un formulaire entrée vide apparaît, vous permettant de créer un autre enregistrement.

#### 4. Saisissez cinq autres enregistrements avec les valeurs suivantes :

Nom	Prénom	Salaire	Département
Deleau	Christophe	1990	Production
Driart	Laurence	2100	Ingénierie
Hector	Marc	2050	Production
Leonard	Louise	2500	Production
Moisson	Franck	1890	Ingénierie

Une fois que vous avez validé le dernier enregistrement, cliquez sur l'icône d'annulation (comportant une croix), de manière à refermer le nouveau formulaire entrée vide. Le formulaire sortie apparaît.

#### 5. Si les six enregistrements ne sont pas affichés, choisissez **Tout montrer** dans le menu **Enregistrements** et redimensionnez la fenêtre ou les colonnes si nécessaire.

Votre fenêtre doit ressembler à celui-ci :

Employés - Employés : 6 enregistrement(s) sur 6			
Nom :	Prénom :	Salaire :	Département :
Poirier	Alain	1950	Ingénierie
Deleau	Christophe	1990	Production
Driart	Laurence	2100	Ingénierie
Hector	Marc	2050	Production
Leonard	Louise	2500	Production
Moisson	Franck	1890	Ingénierie

Les enregistrements sont maintenant stockés dans la base, sur le poste serveur. Si un second 4D distant était connecté au serveur, il pourrait afficher les enregistrements que vous venez de saisir. A l'inverse, si d'autres clients étaient en train de saisir des enregistrements, vous pourriez les afficher en choisissant **Tout montrer** dans le menu **Enregistrements**. Les enregistrements stockés sur le serveur sont accessibles à tous les utilisateurs.

## Chercher des enregistrements

Une fois que vous avez créé des enregistrements dans la table *[Employés]*, vous pouvez effectuer des recherches, des tris, des impressions, etc., avec ces enregistrements. Par exemple, recherchons les employés du département Ingénierie :

**1. Cliquez sur le bouton Requêtes dans la barre d'outils.**



L'éditeur de recherches apparaît.

Chercher

**Editeur de recherches**

**Liste des champs :**

Tables liées

- A Nom
- A Prénom
- 0.5 Salaire
- A Département

**Comparaisons :**

- est égal à
- est différent de
- est supérieur à
- est supérieur ou égal à
- est inférieur à
- est inférieur ou égal à
- contient

**Valeur**

**2. Cliquez sur "Département" dans la liste des champs, sur "est égal à" dans la liste des comparateurs, puis saisissez "Ingénierie" dans la zone Valeur.**

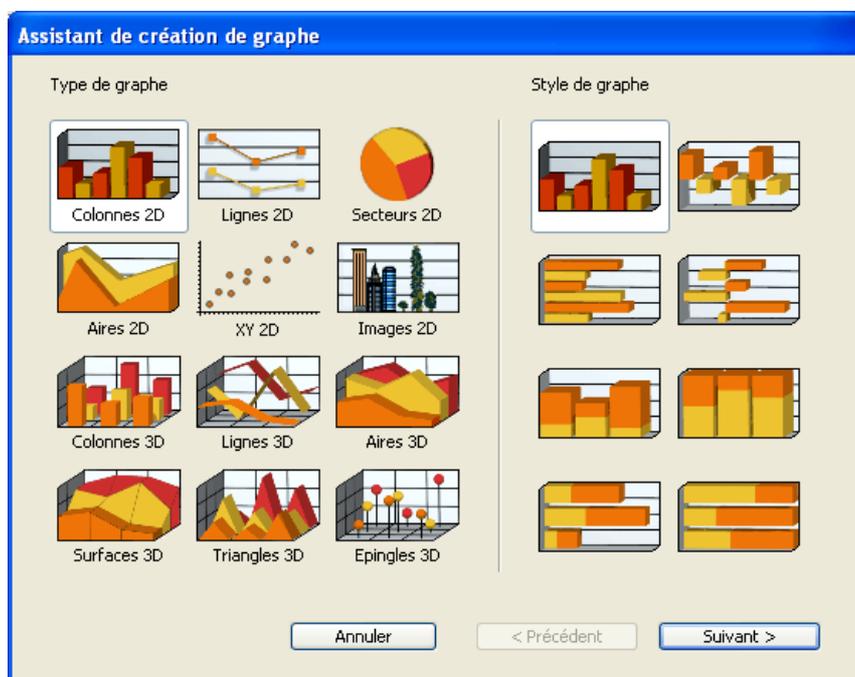
**3. Cliquez sur le bouton Rechercher.**

La requête est envoyée à 4D Server, puis 4D Server répond à 4D. Le formulaire sortie affiche alors uniquement les employés travaillant dans le département Ingénierie. Pour afficher tous les enregistrements, choisissez **Tout montrer** dans le menu **Enregistrements**.

## Créer un graphe

**1. Choisissez Graphes... dans le menu Outils.**

L'assistant de création de graphes apparaît :

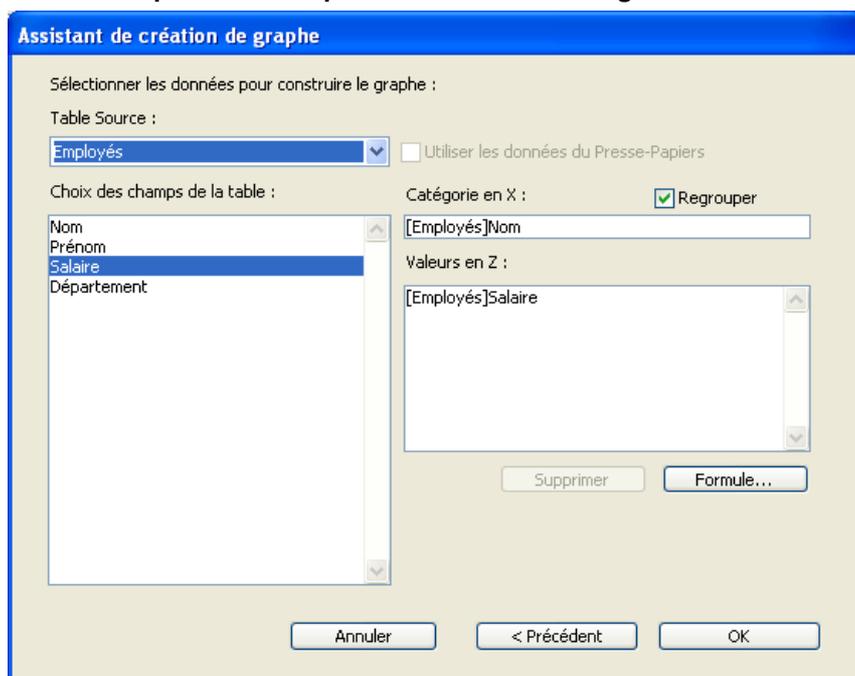


Nous allons créer un graphe en colonnes 2D (ce qui est le type de graphe sélectionné par défaut).

## 2. Cliquez sur le bouton Suivant >

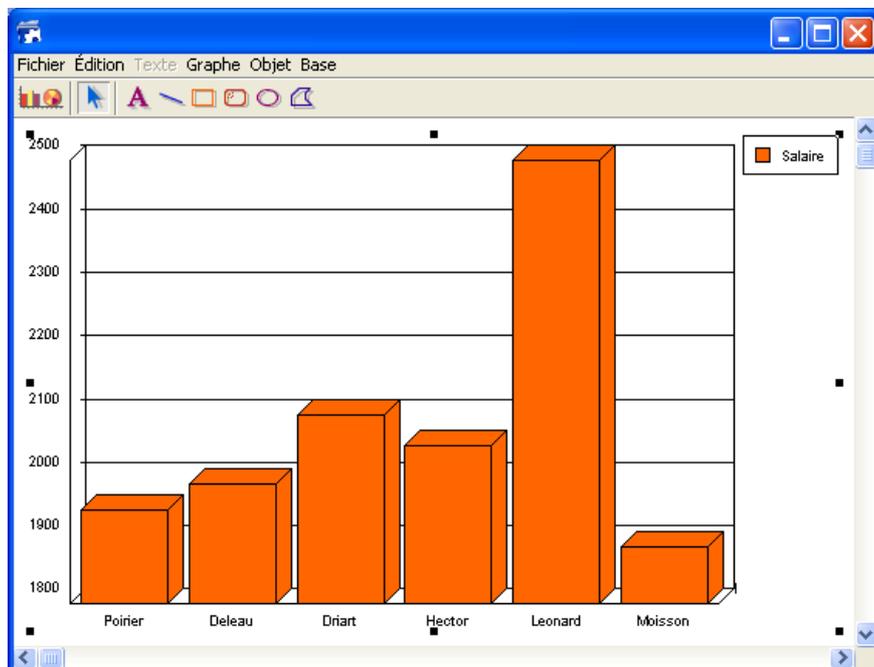
Vous devez sélectionner les données que vous souhaitez représenter graphiquement.

## 3. Glissez-déposez le champ Nom dans la zone Catégorie en X et le champ Salaire dans la liste Valeurs en Z :



## 4. Cliquez sur le bouton OK.

Une fenêtre de graphe apparaît, affichant le graphe issu de vos paramètres.



## Et maintenant...

---

En quelques minutes seulement vous avez créé une base serveur, défini une table, ajouté des enregistrements puis effectué une recherche et généré un graphe avec les données saisies dans la base.

Il est maintenant temps de créer une barre de menus personnalisée pour votre base. Reportez-vous à la section **Créer une barre de menus personnalisée**.

## Créer une barre de menus personnalisée

Dans cette section, vous allez créer deux méthodes et une barre de menus personnalisée. En résumé, vous allez créer une application 4D.

### Créer les deux méthodes

1. Sélectionnez **Nouveau > Méthode...** dans le menu **Fichier**.

La boîte de dialogue de création de méthode apparaît.

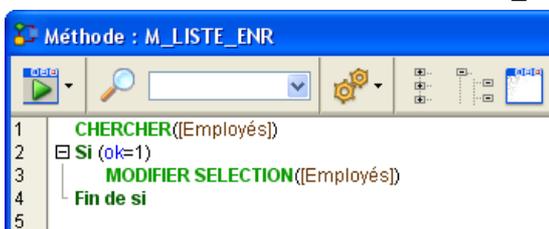
2. Nommez la méthode **"M\_AJOUT\_ENR"** et cliquez sur **OK**.

Une fenêtre de l'éditeur de méthodes apparaît.

3. Ecrivez le code de la méthode **M\_AJOUT\_ENR** ci-dessous :



4. Créez une seconde méthode nommée **"M\_LISTE\_ENR"** avec le code suivant :



Maintenant que les deux méthodes ont été créées, vous allez définir une barre de menus personnalisée et associer ces méthodes à des commandes de menus.

### Créer une barre de menus personnalisée

1. Sélectionnez **Boîte à outils > Menus** dans le menu **Développement**.

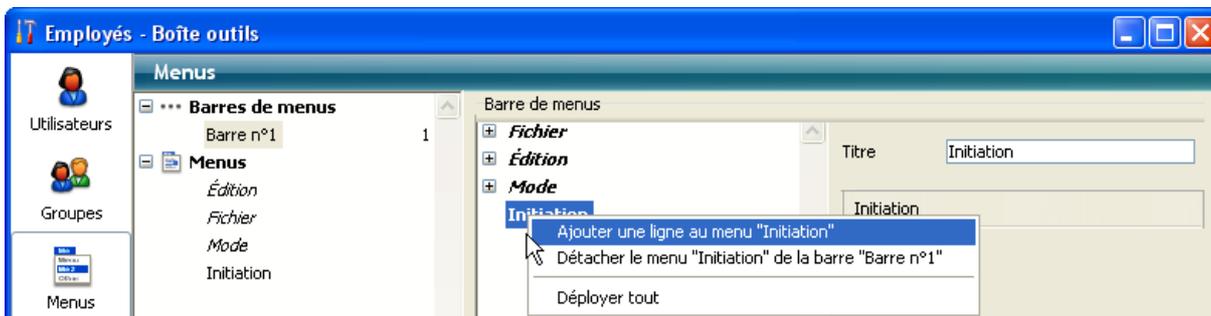
L'éditeur de barres de menus s'affiche. Il contient une barre de menus par défaut.

2. Sélectionnez le libellé **"Barre n°1"** et cliquez sur le bouton d'ajout dans la partie centrale de la fenêtre.



3. Saisissez **"Initiation"** comme libellé de menu et appuyez sur **Entrée**.

4. Cliquez avec le bouton droit de la souris sur le libellé **"Initiation"** et choisissez la commande **Ajouter une ligne au menu "Initiation"** :

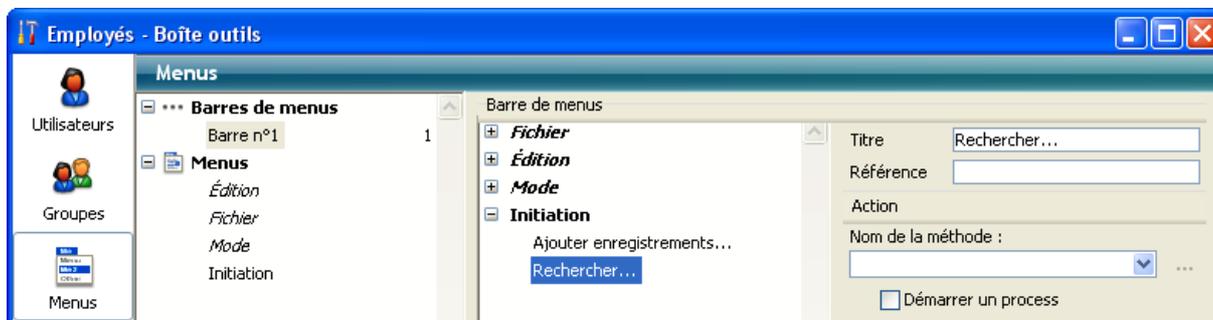


5. Saisissez **"Ajouter enregistrements..."** comme libellé et appuyez sur **Entrée**.

6. Cliquez à nouveau avec le bouton droit de la souris sur le libellé **"Initiation"** et ajoutez une seconde ligne au menu **"Initiation"**.

7. Saisissez **"Rechercher..."** comme libellé et appuyez sur **Entrée**.

La barre de menus n°1 doit maintenant avoir l'aspect suivant :



8. Cliquez sur l'intitulé "Ajouter enregistrements..." et sélectionnez "M\_AJOUT\_ENR" dans la combo box Nom de la méthode.

9. Cliquez sur l'intitulé "Rechercher..." et sélectionnez "M\_LISTE\_ENR" dans la combo box Nom de la méthode.  
La barre de menus n°1 apparaît alors ainsi :

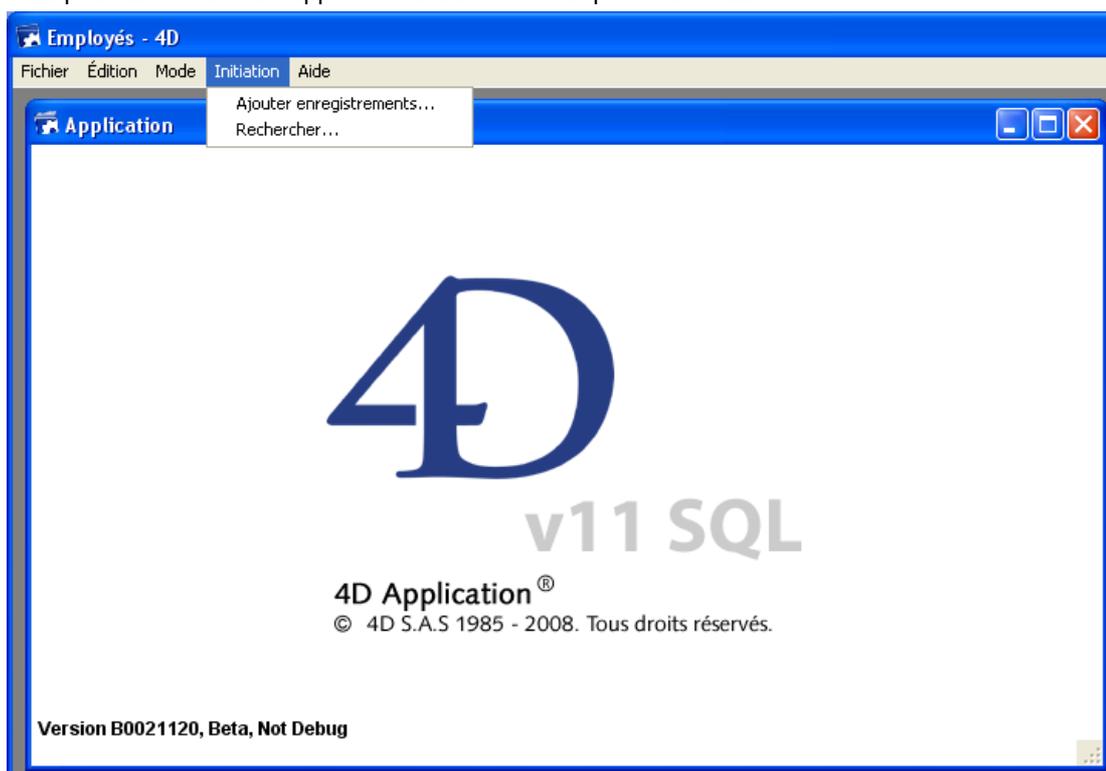


10. Refermez la fenêtre de la boîte à outils.

C'est tout !

11. Sélectionnez Tester l'application dans le menu Exécution.

Vous pouvez alors utiliser l'application avec les menus que vous venez de créer :



Si, par exemple, vous sélectionnez **Rechercher...** dans le menu **Initiation**, l'éditeur de recherches standard de 4D apparaît. Vous pouvez alors construire votre recherche, puis afficher et modifier les enregistrements trouvés.

Mais sachez surtout que, sans le savoir, vous venez de développer deux applications !

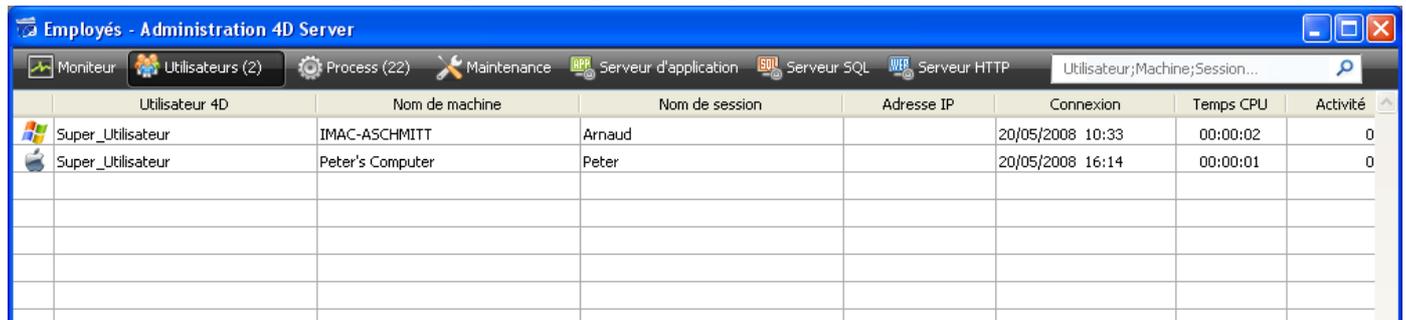
Pour savoir pourquoi, rendez-vous à la section [Travailler simultanément avec plusieurs 4D distants](#).

## Travailler simultanément avec plusieurs 4D distants

Si vous avez effectué les exercices de cette initiation sous Windows, vous pourriez utiliser la base serveur "telle que" sur Macintosh. Si vous avez effectué les exercices de cette initiation sur Macintosh, vous pourriez utiliser la base serveur "telle que" sous Windows.

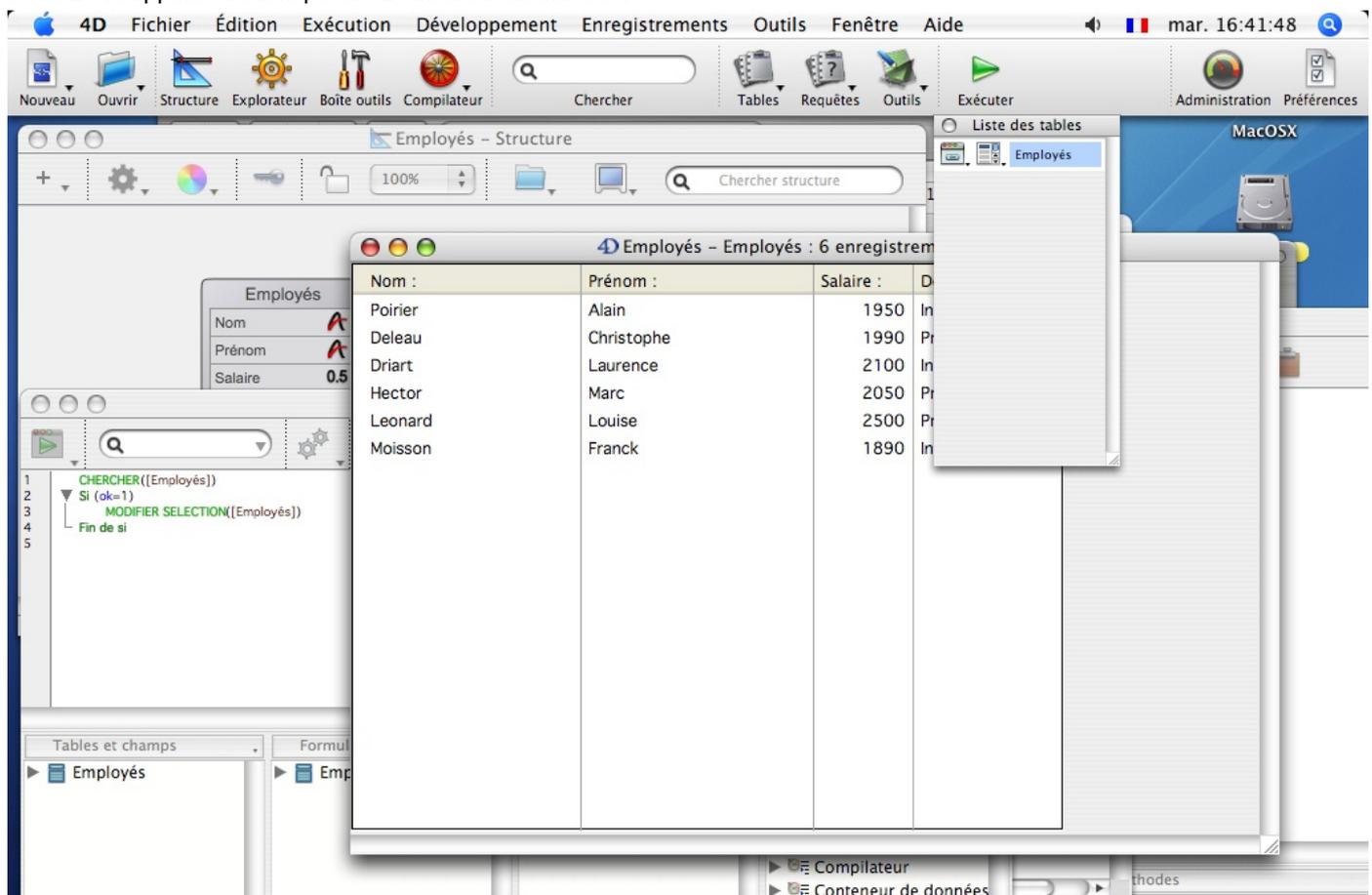
### Connexion à la base serveur avec un second utilisateur

Pour les besoins de cette initiation, nous nous connectons à la base serveur avec un 4D distant sous Windows et un 4D distant sous Mac OS. Dès que la connexion est établie, vous pouvez visualiser le second utilisateur dans la fenêtre d'administration de 4D Server (la première colonne indique le système d'exploitation du poste distant) :



Utilisateur 4D	Nom de machine	Nom de session	Adresse IP	Connexion	Temps CPU	Activité
Super_Utilisateur	IMAC-ASCHMITT	Arnaud		20/05/2008 10:33	00:00:02	0
Super_Utilisateur	Peter's Computer	Peter		20/05/2008 16:14	00:00:01	0

Sur chaque poste client, tout ce qui a été effectué sur la base est réutilisable instantanément et de manière transparente. Voici le mode Développement sur le poste 4D distant sous MacOS :



The screenshot shows the 4D development environment on a Mac OS. The main window displays a table of employee records with the following data:

Nom :	Prénom :	Salaire :	D
Poirier	Alain	1950	In
Deleau	Christophe	1990	Pr
Driart	Laurence	2100	In
Hector	Marc	2050	Pr
Leonard	Louise	2500	Pr
Moisson	Franck	1890	In

The script editor shows the following code:

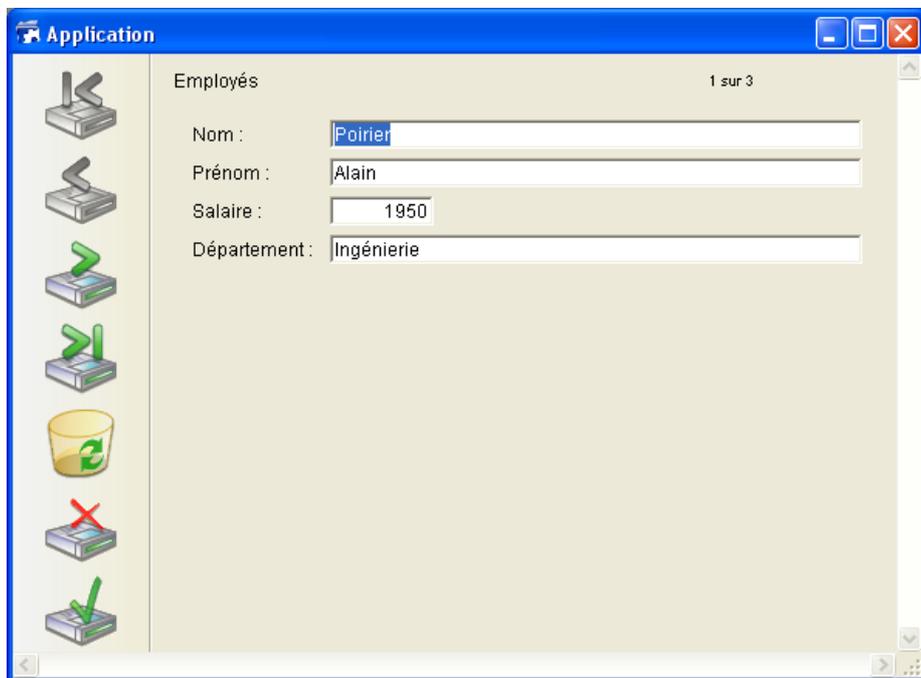
```
1  CHERCHER((Employés))
2  Si (ok=1)
3  MODIFIER SELECTION((Employés))
4  Fin de si
5
```

Vos six enregistrements et vos deux méthodes sont là !

### Travailler simultanément sur les enregistrements

Sur le premier poste distant, en mode "Test application", choisissez **Rechercher...** dans le menu **Initiation** et recherchez les enregistrements pour lesquels "Département est égal à Ingénierie". Faites la même chose sur le second poste distant. Sur les deux postes, vous obtenez une liste composée de trois enregistrements.

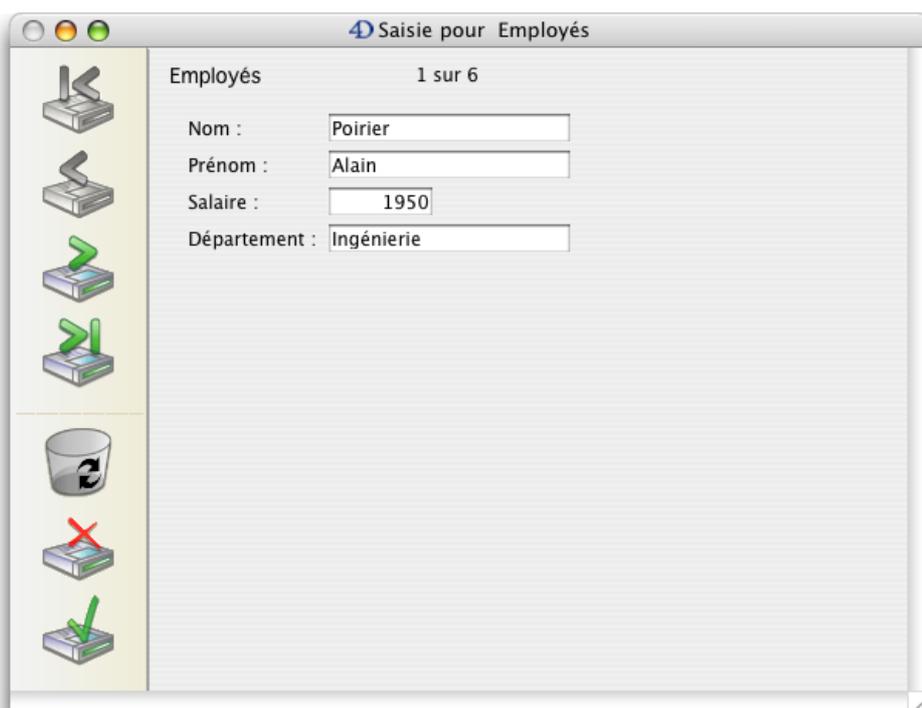
Sur la première machine, double-cliquez sur l'enregistrement "Poirier, Alain". Votre écran affiche la fenêtre suivante :



Faites la même chose sur le second poste. Le système de gestion intégré du verrouillage des données de 4D Server vous avertit que l'enregistrement est déjà en cours d'utilisation :



Vous avez tout de même accès à l'enregistrement en lecture seulement (vous pouvez le visualiser mais pas le modifier) :



Sur la première machine, modifiez le prénom en "Eric" et validez votre modification. Vous obtenez :

Nom :	Prénom :	Salaire :	Département :
Poirier	Eric	1950	Ingénierie
Driart	Laurence	2100	Ingénierie
Moisson	Franck	1890	Ingénierie

La liste a été mise à jour avec le nouveau prénom. Sur la seconde machine, cliquez sur le bouton d'annulation de l'enregistrement dans le formulaire entrée. Vous obtenez :

Nom :	Prénom :	Salaire :	Département :
Poirier	Eric	1950	Ingénierie
Driart	Laurence	2100	Ingénierie
Moisson	Franck	1890	Ingénierie

La liste a également été mise à jour avec le nouveau prénom !

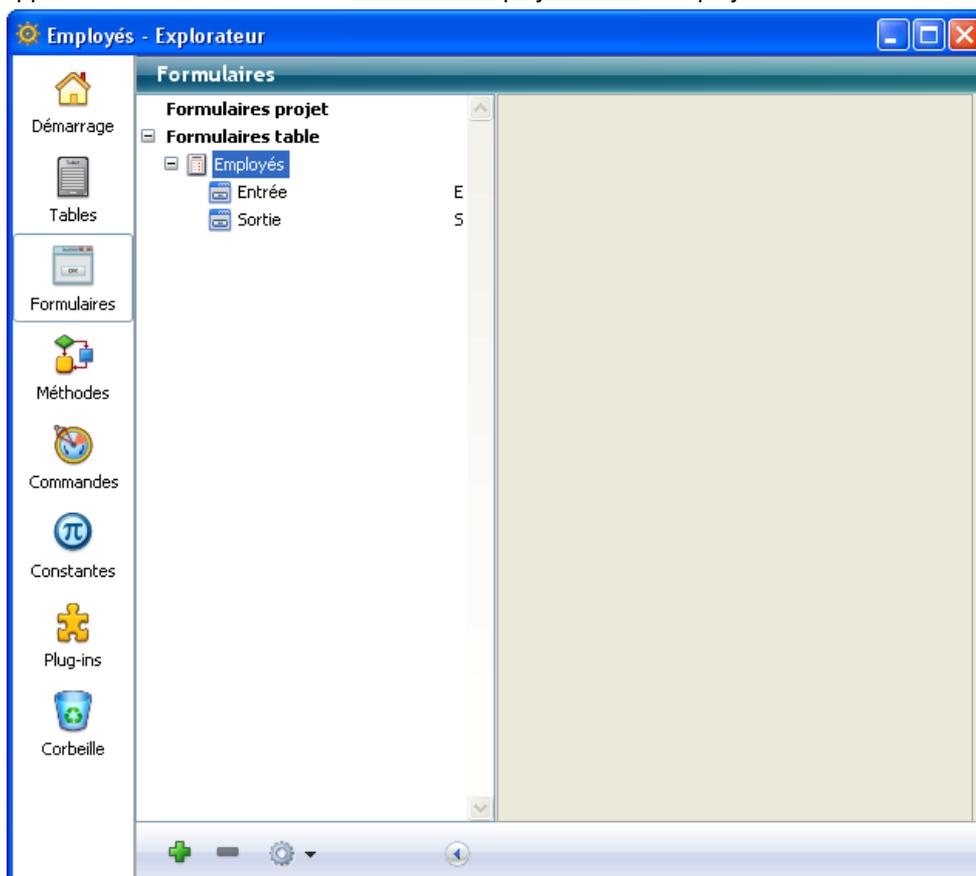
## Travailler simultanément avec des objets de structure

4D Server est un serveur de données **et** d'application. Nous allons voir ce que cela signifie.

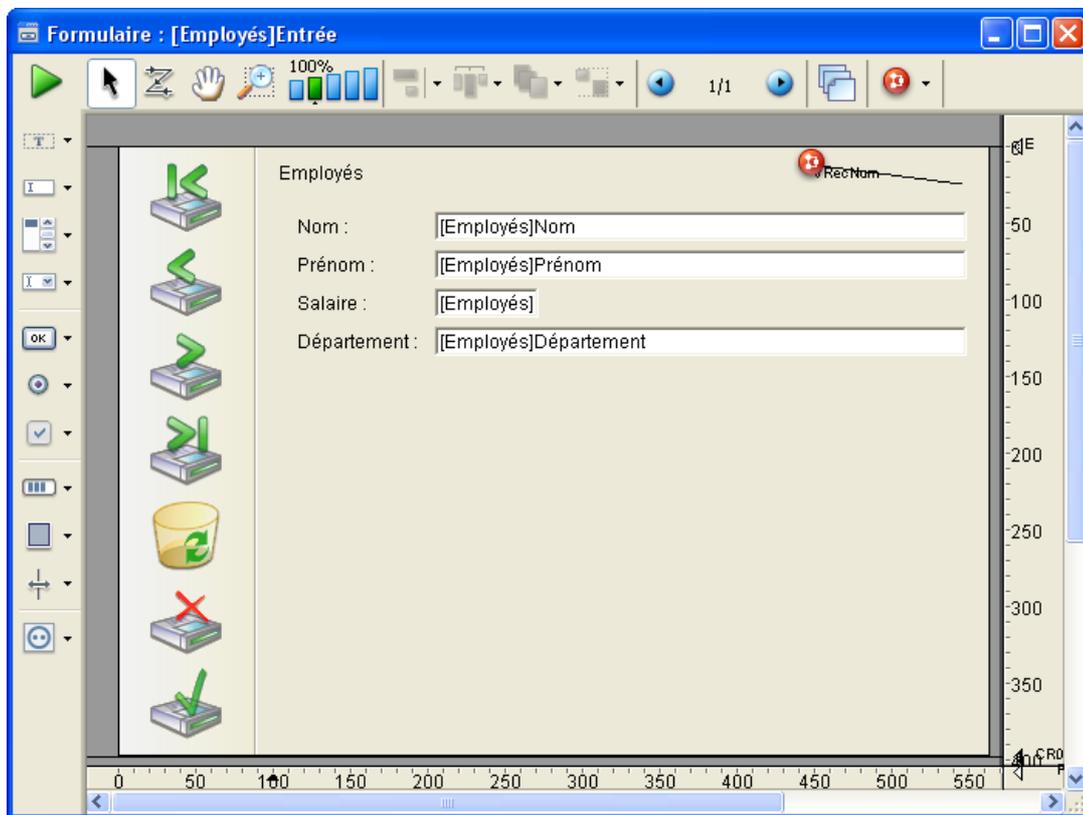
Sur la seconde machine, appuyez sur la touche **Echap**, puis choisissez la commande **Retour au mode Développement** dans le menu **Mode**.

Faites la même chose sur le premier poste.

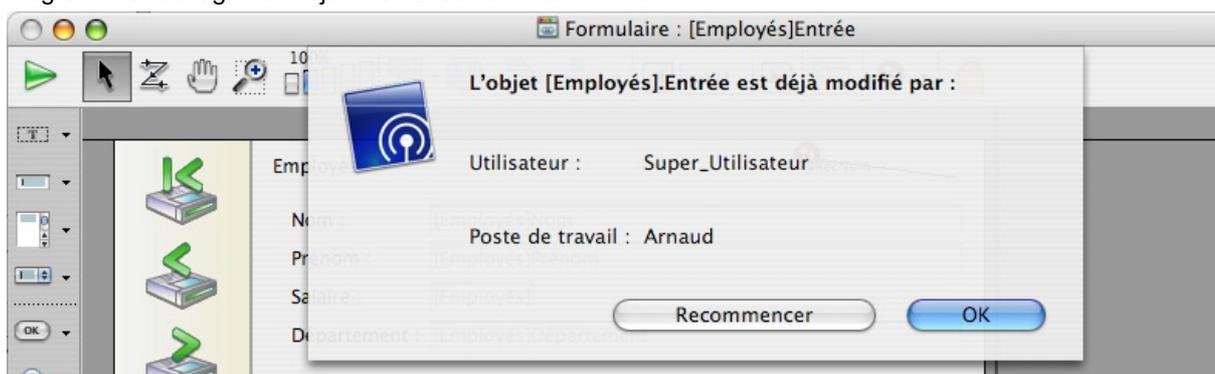
Sur la première machine, choisissez **Explorateur > Formulaires** dans le menu **Développement**. La fenêtre de l'Explorateur apparaît. Choisissez les formulaires table et déployez la table Employés :



Double-cliquez sur le formulaire entrée. L'éditeur de formulaires apparaît, il contient le formulaire entrée :

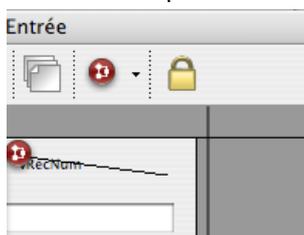


Faites la même chose sur le second poste. Comme le formulaire est déjà en mode modification sur l'autre machine, le mécanisme intégré de verrouillage des objets de 4D Server vous informe :



Malgré tout, vous pouvez ouvrir le formulaire sur la seconde machine, en mode visualisation seulement. Vous pouvez sélectionner et copier des objets vers d'autres formulaires, mais vous ne pouvez pas modifier le formulaire lui-même.

Notez l'icône représentant un cadenas dans l'angle supérieur droit du formulaire :



Elle signifie que vous ne pouvez pas modifier le formulaire.

Sur le premier poste, sélectionnez le libellé "Nom" à gauche du champ [Employés]Nom. Sélectionnez ensuite le menu hiérarchique **Objet>Couleur**. Affectez, par exemple, la couleur rouge à l'objet :



Sélectionnez ensuite **Sauvegarder le formulaire: [Employés]Entrée** dans le menu **Fichier**.

Sur la seconde machine, refermez puis ouvrez à nouveau le formulaire afin de le charger ; la modification de couleur que vous venez d'apporter au libellé apparaît alors :



4D Server vous permet de développer simultanément une base à plusieurs !

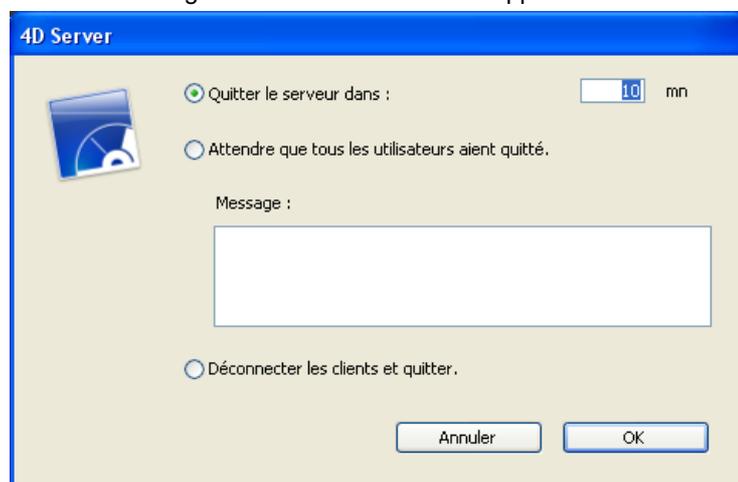
## Arrêt du serveur

---

En plus de l'information des utilisateurs 4D distants lors d'accès simultanés aux mêmes enregistrements ou objets, 4D Server gère en interne l'information des utilisateurs à travers le réseau lorsqu'il quitte.

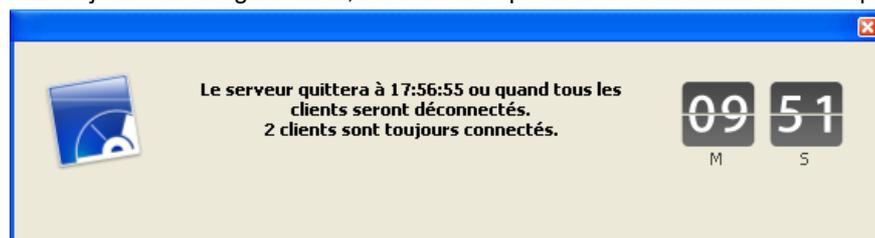
Tout en maintenant connectés vos deux clients, choisissez, sur le poste serveur, la commande **Quitter** dans le menu **Fichier** (Windows) ou dans le menu **4D Server** (MacOS).

La boîte de dialogue de fermeture de la base apparaît :



Cliquez sur le bouton **OK**.

Presque instantanément, les deux postes clients sont informés que le serveur va bientôt quitter. Si, par exemple, un client était en train d'ajouter un enregistrement, l'utilisateur disposerait de suffisamment de temps pour terminer et valider la saisie.



Cette boîte de dialogue d'alerte est répétée régulièrement sur chaque poste client.

**Note :** Vous pouvez également choisir de quitter l'application serveur en utilisant l'option "Attendre que tous les utilisateurs aient quitté" (en leur adressant éventuellement un message les invitant à se déconnecter dès que possible) ou forcer la déconnexion immédiate des clients via l'option "Déconnecter les clients et quitter".

Pendant que le serveur poursuit le processus de déconnexion, quittez 4D sur les deux machines distantes.

## Conclusion

---

A travers cette initiation, vous avez pu découvrir la simplicité et la facilité d'utilisation de 4D Server :

- Vous avez créé une nouvelle base
- Vous avez créé une table et laissé 4D Server construire les formulaires pour vous
- Vous avez ajouté et manipulé des enregistrements
- Vous avez personnalisé votre application avec votre propre barre de menus
- Vous avez utilisé la base serveur simultanément sous Windows et MacOS
- Vous avez quitté puis relancé le serveur

En définitive, vous avez créé deux applications personnalisées (Windows et Macintosh) alors que vous n'avez en fait effectué qu'un seul développement. De plus, si vous souhaitez utiliser la base en mode local, vous pouvez l'ouvrir directement avec 4D.

Pour en savoir plus à propos de 4D Server, reportez-vous aux sections d'introduction de ce manuel, ainsi qu'aux autres sections qui détaillent le fonctionnement de 4D Server.

Pour une information complète sur l'environnement 4D, référez-vous aux manuels suivants :

- Mode Développement pour savoir comment construire et utiliser des applications et des bases de données 4D.
- Langage pour connaître en détail les commandes du langage de 4D. Si, par exemple, vous souhaitez connaître les capacités de 4D Server en matière de Web, lisez la section **Présentation du serveur Web** dans le manuel Langage de 4D.

## **Utilisation de 4D Server**

-  Créer ou ouvrir une base 4D Server
-  Quitter 4D Server
-  Menus de 4D Server
-  Options réseau et Client-serveur
-  Configuration IP
-  Crypter les connexions client/serveur
-  Gestion du dossier Resources
-  Enregistrer une base comme service
-  Mise en place d'un miroir logique

## Créer ou ouvrir une base 4D Server

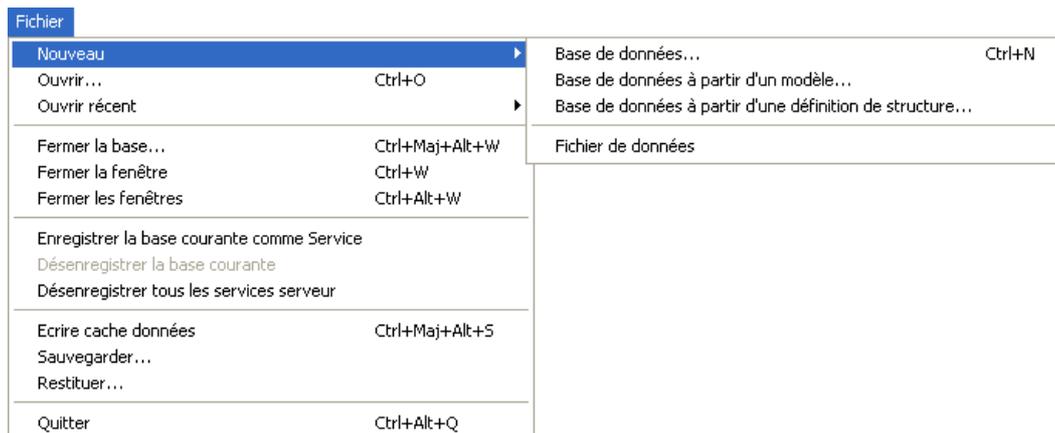
Pour créer une nouvelle base ou ouvrir une base existante, vous devez lancer 4D Server en double-cliquant sur l'icône de l'application.



Vous pouvez alors créer une nouvelle base ou ouvrir une base existante dans le menu **Fichier** de 4D Server.

### Créer une base

Pour créer une nouvelle base, choisissez une des commandes du sous-menu **Nouveau** :



- **Base de données** : permet de créer une base de données vierge, c'est-à-dire ne comportant ni table, formulaire ou interface prédéfini(e). Lorsque vous choisissez cette commande, une boîte de dialogue standard d'enregistrement de fichiers apparaît, vous permettant de définir le nom et l'emplacement de la base.
- **Base de données à partir d'un modèle** : crée une base en utilisant un modèle "prêt à l'emploi" que vous pourrez personnaliser par la suite. Pour que vous puissiez utiliser cette fonction, les bases modèles doivent être placées dans un dossier "4D Templates" ou "4D Modèles" au même niveau que le fichier 4D Server.exe (Windows) ou le progiciel 4D Server (Mac OS). Lorsque vous choisissez cette commande, la boîte de dialogue de sélection d'un modèle de base de données apparaît.
- **Base de données à partir d'une définition de structure** : permet de créer une base de données basée sur une description de structure au format XML. Cette définition peut provenir d'une structure exportée depuis 4D ou toute application de conception. Lorsque vous choisissez cette commande, une boîte de dialogue standard d'ouverture de fichiers apparaît, vous permettant de désigner le fichier XML à utiliser.

Pour plus d'informations sur ces options, reportez-vous au manuel Mode Développement.

### Ouvrir une base

Pour ouvrir une base existante, vous pouvez utiliser la boîte de dialogue standard d'ouverture de documents (commande **Fichier>Ouvrir...**) ou sélectionner directement une base précédemment ouverte (commande **Fichier>Ouvrir récent**).

Si une base était déjà ouverte au moment de la sélection d'une commande d'ouverture, elle est refermée au préalable. Si des postes clients étaient connectés, ils sont déconnectés avec le mode "Attendre que tous les utilisateurs aient quitté" (cf. section [Quitter 4D Server](#)).

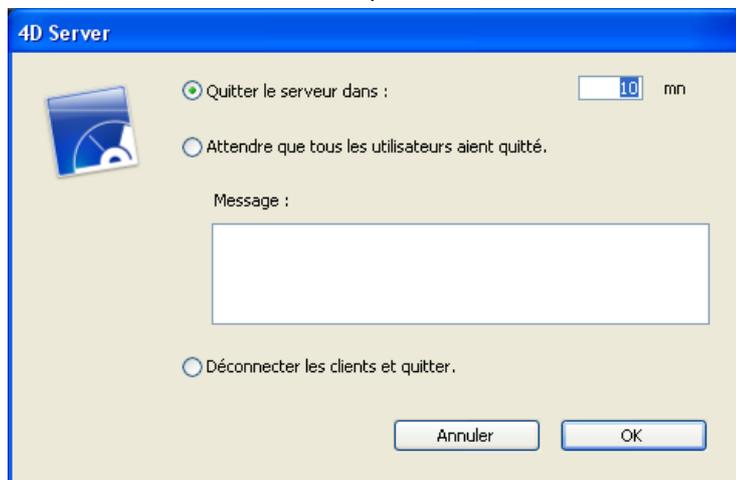
**Note** : Vous pouvez également ouvrir directement une base existante en effectuant un glisser-déposer d'un fichier de structure interprété ou compilé (.4db ou .4dc) ou d'un fichier de raccourci (.4dlink) sur l'icône de l'application 4D Server.

## Quitter 4D Server

Pour quitter l'application 4D Server, procédez ainsi :

**1. Sélectionnez la commande Quitter dans le menu Fichier de 4D Server (Windows) ou dans le menu 4D Server (Mac OS).**

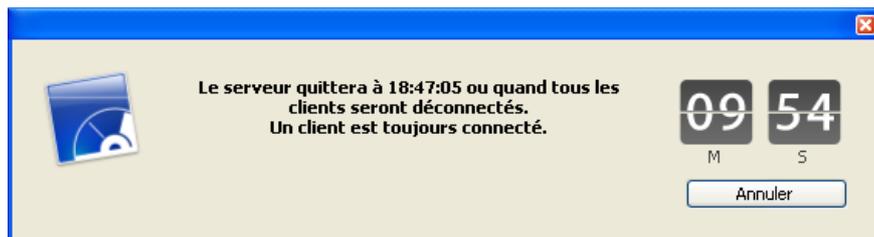
La fenêtre suivante s'affiche sur le poste serveur :



**2. Choisissez une des options de fermeture ainsi qu'un paramétrage complémentaire (nombre de minutes ou message destiné aux clients) puis cliquez sur OK.**

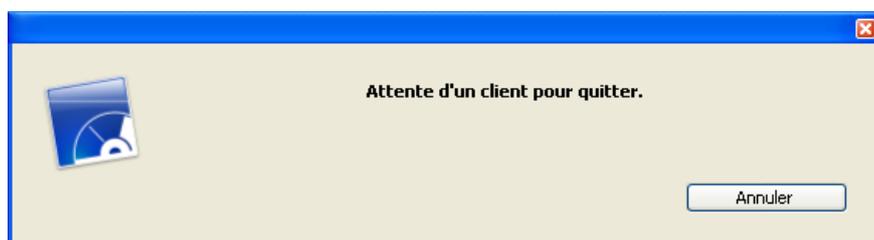
A partir de cet instant, plus aucun nouveau client ne peut se connecter au serveur.

- Si vous avez choisi l'option "Quitter le serveur dans XX mn", la fenêtre suivante apparaît sur le serveur :

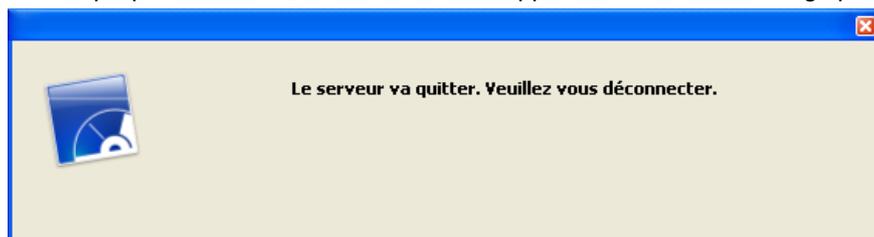


Une fenêtre identique apparaît sur chaque poste client 4D (le bouton **Annuler** n'apparaît pas sur les postes 4D). Cette fenêtre est répétée ou mise à jour sur chaque poste client toutes les 20 secondes environ, afin de l'inciter à quitter. A l'issue du délai, le serveur quitte même si des clients sont encore connectés.

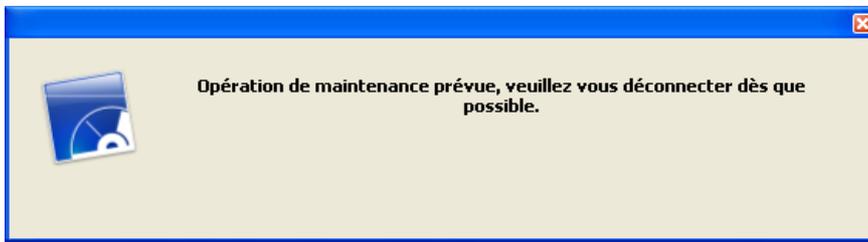
- Si vous avez choisi l'option "Attendre que tous les utilisateurs aient quitté" sans saisir de message, la fenêtre suivante apparaît sur le serveur, elle indique le nombre de clients encore connectés :



Sur chaque poste client 4D, la fenêtre suivante apparaît, affichant un message par défaut :



Si vous saisissez un message personnalisé dans la boîte de dialogue de fermeture de 4D Server, il est affiché à la place du message par défaut sur chaque poste client. Par exemple :



Le serveur quitte dès que tous les clients se sont déconnectés.

- Si vous avez choisi l'option "Déconnecter les clients et quitter", le serveur met fin à tous les process et toutes les connexions et quitte en quelques secondes.

**Notes :**

- Dans tous les cas, si aucun client n'est connecté au serveur au moment où la fenêtre de fermeture est validée, 4D Server quitte immédiatement.
- Si vous cliquez sur le bouton **Annuler** dans la fenêtre "Arrêt du serveur", le processus d'arrêt du serveur est annulé.
- Vous pouvez refermer la base (et déconnecter les clients) sans quitter l'application 4D Server, via la commande **Fermer la base...** Pour plus d'informations, reportez-vous à la section **Menus de 4D Server**.

## Menus de 4D Server

---

L'interface de l'application 4D Server se compose des menus **Fichier**, **Edition**, **Fenêtre**, **Aide**. Sous Mac OS, certaines commandes sont placées dans le menu **4D Server** (menu application).

### Fichier

---

Fichier	
Nouveau	
Ouvrir...	Ctrl+O
Ouvrir récent	
<hr/>	
Fermer la base...	Ctrl+Maj+Alt+W
Fermer la fenêtre	Ctrl+W
Fermer les fenêtres	Ctrl+Alt+W
<hr/>	
Enregistrer la base courante comme Service	
Désenregistrer la base courante	
Désenregistrer tous les services serveur	
<hr/>	
Ecrire cache données	Ctrl+Maj+Alt+S
Sauvegarder...	
Restituer...	
<hr/>	
Quitter	Ctrl+Alt+Q

### Nouveau

---

Cette commande hiérarchique donne accès à un sous-menu permettant de créer une base de données ou un nouveau fichier de données sur le poste serveur.

Les commandes de création de bases sont détaillées dans la section [Créer ou ouvrir une base 4D Server](#).

### Ouvrir..., Ouvrir récent

---

Ces commandes permettent d'ouvrir une base avec 4D Server. La commande **Ouvrir récent** affiche un sous-menu listant les bases récemment ouvertes par 4D Server. Pour réinitialiser le menu, choisissez la commande **Effacer le menu**.

Les commandes d'ouverture de bases sont détaillées dans la section [Créer ou ouvrir une base 4D Server](#).

### Fermer la base...

---

Cette commande de menu referme la base courante sans quitter l'application 4D Server. Lorsque vous choisissez cette commande, la boîte de dialogue de fermeture du serveur apparaît, permettant de définir le mode de déconnexion des clients éventuellement connectés (cf. section [Quitter 4D Server](#)).

### Fermer la fenêtre

---

Cette commande provoque la fermeture de la fenêtre de l'application 4D Server située au premier plan.

### Fermer les fenêtres

---

Cette commande provoque la fermeture de toutes les fenêtres de l'application 4D Server. A noter que dans ce cas seule l'activation de la commande **Fermer la base...** dans le menu **Fichier** vous permet de savoir que la base est toujours publiée.

### Enregistrer la base courante comme Service, Désenregistrer la base courante, Désenregistrer tous les services serveur

---

(Commandes disponibles sous Windows) Ces commandes permettent de gérer l'enregistrement de la base comme Service. Cette fonction est détaillée dans la section [Enregistrer une base comme service](#).

### Ecrire cache données

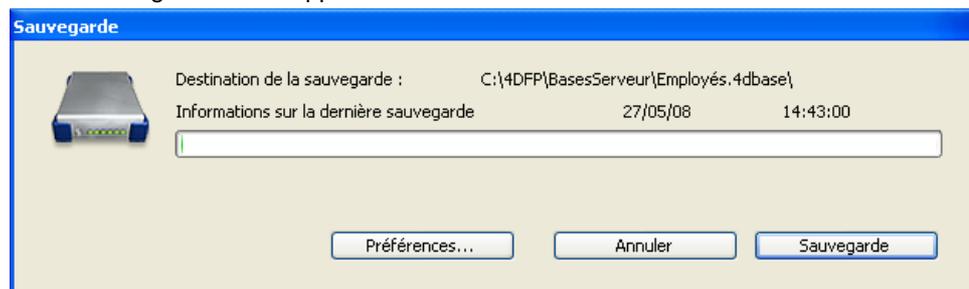
---

Cette commande permet de "forcer" l'enregistrement sur le disque des données placées dans le cache. Par défaut, 4D Server écrit automatiquement le cache sur le disque au bout du délai défini dans les préférences de la base (page Base de

## Sauvegarder...

---

Cette commande permet de déclencher une sauvegarde de la base à tout instant. Lorsque vous sélectionnez cette commande, la boîte de dialogue suivante apparaît :



- Le bouton **Sauvegarde** lance immédiatement une sauvegarde en tenant compte des paramètres définis dans la boîte de dialogue des Préférences de l'application (fichiers sauvegardés, emplacement des archives, nombre de jeux, etc.).
- Le bouton **Préférences** ouvre le thème "Sauvegarde" des Préférences, vous permettant de visualiser et éventuellement de modifier les paramètres de sauvegarde courants.
- Le bouton **Annuler** interrompt le processus de sauvegarde.

Pour plus d'informations sur le paramétrage des sauvegardes, reportez-vous au manuel Mode Développement de la documentation de 4D.

## Restituer...

---

Cette commande affiche une boîte de dialogue d'ouverture vous permettant de sélectionner une archive à restituer.

## Quitter

---

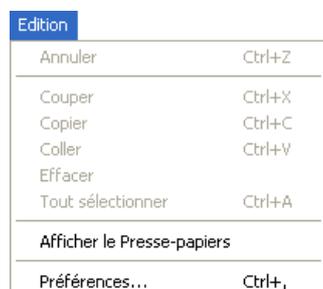
Cette commande permet de refermer l'application 4D Server. Pour plus d'informations, reportez-vous à la section **Quitter 4D Server**.

**Note** : Sous Mac OS X, la commande **Quitter** se trouve dans le menu **4D Server** (menu application).



## Edition

---



Le menu **Edition** de 4D Server comporte les commandes standard de copier/coller, la commande **Afficher le Presse-papiers**, etc.

Ce menu comporte également (sous Windows) la commande **Préférences...** Cette commande affiche la boîte de dialogue des Préférences de l'application, permettant de paramétrer de nombreux fonctionnements de la base. Pour plus d'informations sur cette boîte de dialogue, reportez-vous au manuel Mode Développement de la documentation de 4D. Les préférences spécifiques à 4D Server sont décrites dans les sections **Options réseau et Client-serveur** et **Configuration IP**.

**Note** : Sous Mac OS, la commande **Préférences...** est placée dans le menu **4D Server** (menu application).

## 4D Server

A propos de 4D Server@...	
Préférences...	⌘,
Services	▶
Masquer 4D Server	⌘H
Masquer les autres	⌘H
Tout afficher	
Quitter 4D Server	⌘Q

## Fenêtre

### Fenêtre

Réduire la fenêtre	Ctrl+M
Réduire toutes les fenêtres	Ctrl+Maj+Alt+M
Tout ramener au premier plan	
Cascade	
Administration	Ctrl+U
Explorateur d'exécution	

Le menu **Fenêtre** propose en premier lieu les commandes standard permettant d'organiser les fenêtres dans l'espace de travail (ces commandes diffèrent suivant la plate-forme).

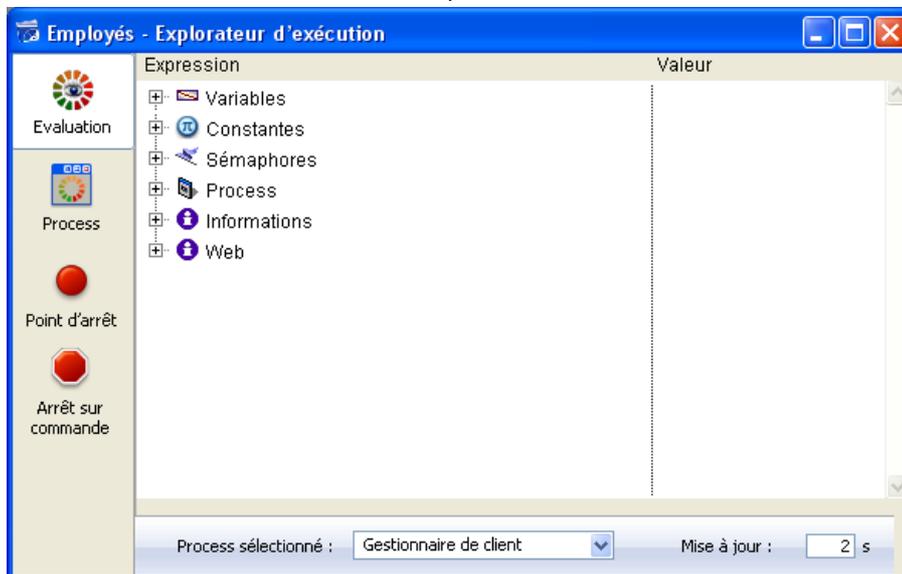
Il contient également les commandes d'affichage des fenêtres spécifiques de 4D Server :

## Administration

Cette commande affiche la fenêtre d'administration de 4D Server, si elle a été auparavant réduite ou fermée. Cette fenêtre est détaillée dans le chapitre Fenêtre d'administration de 4D Server (cf. section [Page Moniteur](#)).

## Explorateur d'exécution

Cette commande affiche la fenêtre de l'Explorateur d'exécution de 4D Server.



L'Explorateur d'exécution permet de visualiser le comportement des différents éléments structurels de la base et de vérifier que les ressources disponibles sont correctement exploitées. L'Explorateur d'exécution est particulièrement utile en phase de développement et d'analyse d'une base de données.

La fenêtre se compose de quatre pages, accessibles via les boutons correspondants : **Evaluation**, **Process**, **Point d'arrêt** et **Arrêt sur commande**. Le fonctionnement de l'Explorateur d'exécution sur 4D Server est identique à celui de 4D. Pour plus d'informations, reportez-vous au manuel Mode Développement.

## Aide

### Aide

Centre de sécurité et de maintenance
Mise à jour des licences...
A propos de 4D Server@...

## Centre de sécurité et de maintenance

---

Cette commande affiche la fenêtre du Centre de sécurité et de maintenance (CSM), regroupant tous les outils nécessaires au contrôle, à l'analyse, à la maintenance, à la sauvegarde et au compactage des fichiers de données et de structure. La commande est disponible même lorsqu'aucune base n'est ouverte par 4D Server : elle permet dans ce cas d'ouvrir une base en "mode maintenance" (elle affiche une boîte de dialogue standard d'ouverture de fichiers, permettant de désigner la base à ouvrir). Le mode maintenance est utilisé notamment pour les opérations telles que le compactage ou l'ouverture de bases endommagées. Pour plus d'informations sur le CSM, reportez-vous au manuel Mode Développement.

## Mise à jour des licences...

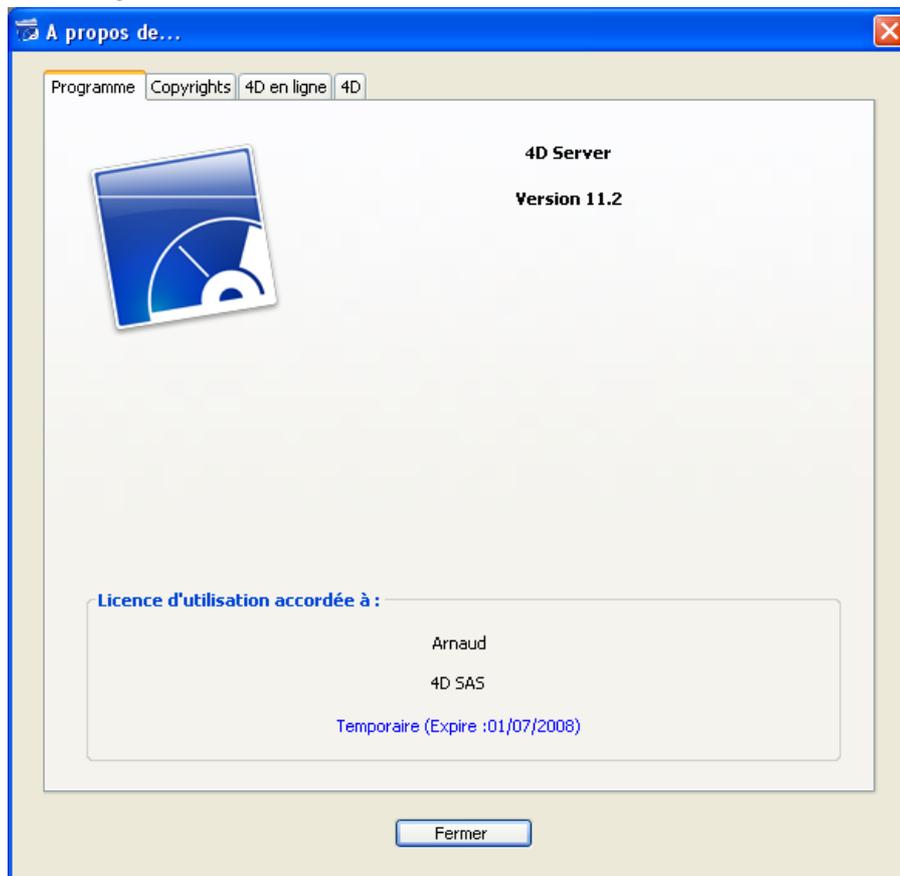
---

Cette commande de menu affiche la fenêtre permettant d'activer des licences supplémentaires dans votre environnement 4D. Pour plus d'informations sur cette boîte de dialogue, reportez-vous au Guide d'installation.

## A propos de 4D Server...

---

Cette commande affiche la fenêtre d'A propos de 4D Server, fournissant diverses informations classées par pages, accessibles via des onglets :



- Programme : version et licence 4D Server
- Copyrights : mentions légales
- 4D en ligne : ressources complémentaires liées à 4D et accessibles en ligne
- 4D : un aperçu de la société 4D SAS dans le monde.

**Note :** Sous Mac OS, la commande **A propos de 4D Server** est placée dans le menu **4D Server** (menu application).

## Options réseau et Client-serveur

Vous pouvez définir divers paramètres relatifs au réseau et à la communication client-serveur dans l'onglet "Options réseau" de la page **Client-Serveur** des Propriétés de la base (accessibles depuis 4D en mode distant et 4D Server) :

MaMusique - Propriétés de la base

Général Interface Compileur Base de données Déplacement Sauvegarde Client-Serveur Web SQL PHP Sécurité Compatibilité

Options réseau Configuration IP

Réseau

Publier la base au démarrage

Nom de publication : MaMusique

Numéro de port : 19813

Délai avant déconnexion Client-Serveur : 1 min. 5 min. 15 min. 30 min. 1 h Illimitée

Communication Client-Serveur

Inscrire les clients au démarrage pour Exécuter sur client

Crypter les connexions Client-Serveur  
Cette fonction nécessite 4DSL1.DLL

Mise à jour du dossier "Resources" en cours de session : Jamais

Ouvrir la structure en mode : Lecture écriture

Réglages d'usine Annuler OK

Ces paramètres sont décrits dans cette section.

### Réseau

#### Publier la base au démarrage

Cette option permet d'indiquer si la base 4D Server doit apparaître ou non dans la liste des bases publiées dans la boîte de dialogue de connexion.

- lorsque l'option est cochée (par défaut), la base est rendue publique, elle apparaît dans la liste des bases publiées (page **TCP/IP**).
- lorsque l'option est désélectionnée, la base n'est pas rendue publique, elle n'apparaît pas dans la liste des bases publiées. Pour se connecter, les utilisateurs doivent saisir manuellement l'adresse de la base dans la page **Personnalisée** de la boîte de dialogue de connexion.

**Note** : Si vous modifiez ce paramètre, vous devez redémarrer la base serveur afin qu'il soit pris en compte.

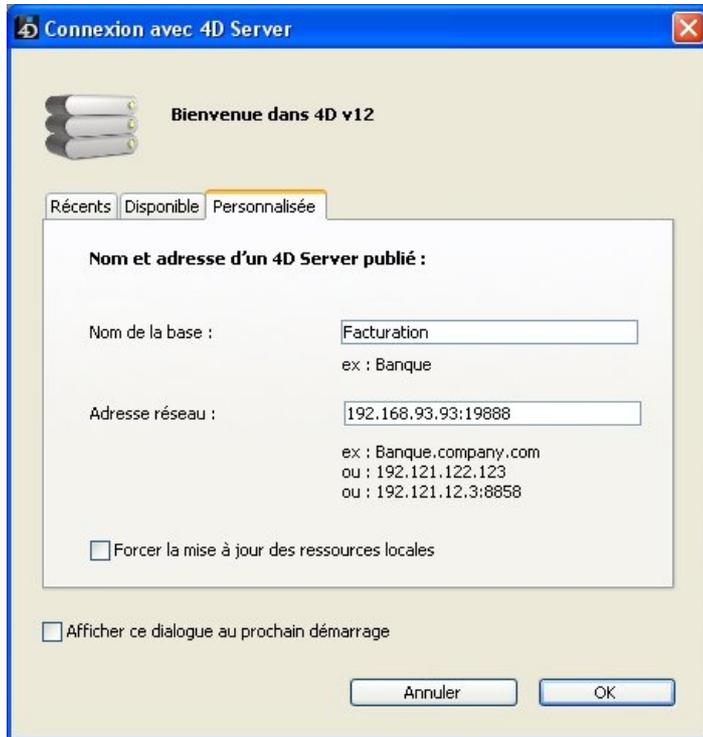
#### Nom de publication

Cette option permet de modifier le nom de publication d'une base publiée par 4D Server, c'est-à-dire le nom affiché dans la page de publication dynamique TCP/IP de la boîte de dialogue de connexion (cf. section **Connexion à une base 4D Server**). Par défaut, 4D Server utilise le nom du fichier de structure de la base. Vous pouvez saisir tout nom personnalisé.

**Note** : Ce paramètre n'est pas pris en compte dans le cadre des applications client-serveur personnalisées. En principe, l'application cliente se connecte directement à l'application serveur, sans passer par la boîte de dialogue de connexion. Toutefois, en cas d'erreur, cette boîte de dialogue apparaît ; dans ce cas, le nom de publication de l'application serveur est le nom de la base compilée.

## Numéro de port

Cette option permet de modifier le numéro de port TCP sur lequel 4D Server publie la base de données. Cette information est stockée dans la structure de la base et sur chaque poste client. Par défaut, le numéro de port TCP utilisé par 4D Server et 4D en mode distant est le 19813. La personnalisation de cette valeur est nécessaire lorsque vous souhaitez utiliser plusieurs applications 4D sur la même machine avec le protocole TCP ; dans ce cas, vous devez spécifier un numéro de port différent pour chaque application. Lorsque vous modifiez cette valeur depuis 4D Server ou 4D, elle est automatiquement répercutée sur tous les postes 4D connectés à la base. Pour mettre à jour les autres postes clients non connectés, il suffira, lors de leur connexion suivante, de saisir le nouveau numéro de port (précédé de deux-points) derrière l'adresse IP du poste serveur dans la page Personnalisée de la boîte de dialogue de connexion. Par exemple, si le nouveau numéro de port est le 19888 :



**Note :** Seules les bases publiées sur le même port que celui défini dans le client 4D sont visibles dans la page de publication dynamique TCP/IP.

## 4D Server et les numéros de port

4D Server utilise trois ports TCP pour les communications entre les serveurs internes et les clients :

- **Serveur SQL** : 19812 par défaut (modifiable via la page "SQL/Configuration" des Préférences).
- **Serveur d'application** : 19813 par défaut (modifiable via la page "Client-serveur/Configuration" des Préférences, cf. ci-dessus).
- **Serveur DB4D** (serveur de base de données) : 19814 par défaut. Ce numéro de port n'est pas modifiable directement mais il s'agit toujours du numéro de port du serveur d'application + 1. Lorsqu'un client 4D se connecte à 4D Server, il s'adresse au port TCP du serveur d'application (19813 ou port indiqué après ':' dans l'adresse IP indiquée dans la boîte de dialogue de connexion. La connexion aux autres serveurs via leur port respectif est ensuite automatique, il n'est pas nécessaire de les préciser. A noter que dans le cas d'accès via un routeur ou un firewall, les trois ports TCP doivent être ouverts explicitement.

## Délai avant déconnexion Client-Serveur

Ce thermomètre permet de définir le timeout (période d'inactivité au-delà de laquelle la connexion est fermée) entre 4D Server et les postes clients qui s'y connectent. L'option Illimité élimine le timeout. Lorsque cette option est sélectionnée, le contrôle d'inactivité du client est désactivé. Lorsqu'un délai est sélectionné, le serveur mettra un terme à la connexion d'un client s'il ne reçoit pas de requête de ce dernier dans l'intervalle de temps spécifié.

## Communication Client-Serveur

---

### Inscrire les clients au démarrage pour Exécuter sur client

Lorsque cette option est cochée, tous les postes 4D distants se connectant à la base peuvent exécuter des méthodes à distance. Ce mécanisme est détaillé dans la section [Procédures stockées sur les clients](#).

### Crypter les connexions Client-Serveur

Cette option permet d'activer le mode SSL (mode sécurisé) pour la communication entre le poste serveur et les postes 4D distants. Cette option est détaillée dans la section .

### Mise à jour du dossier Ressources en cours de session

Ce paramétrage permet de définir globalement le mode de mise à jour de l'instance locale du dossier **Resources** des postes 4D connectés en cas de modification du dossier **Resources** de la base au cours de la session (le dossier **Resources** est automatiquement synchronisé sur le poste distant à chaque ouverture de session). Trois paramétrages sont proposés :

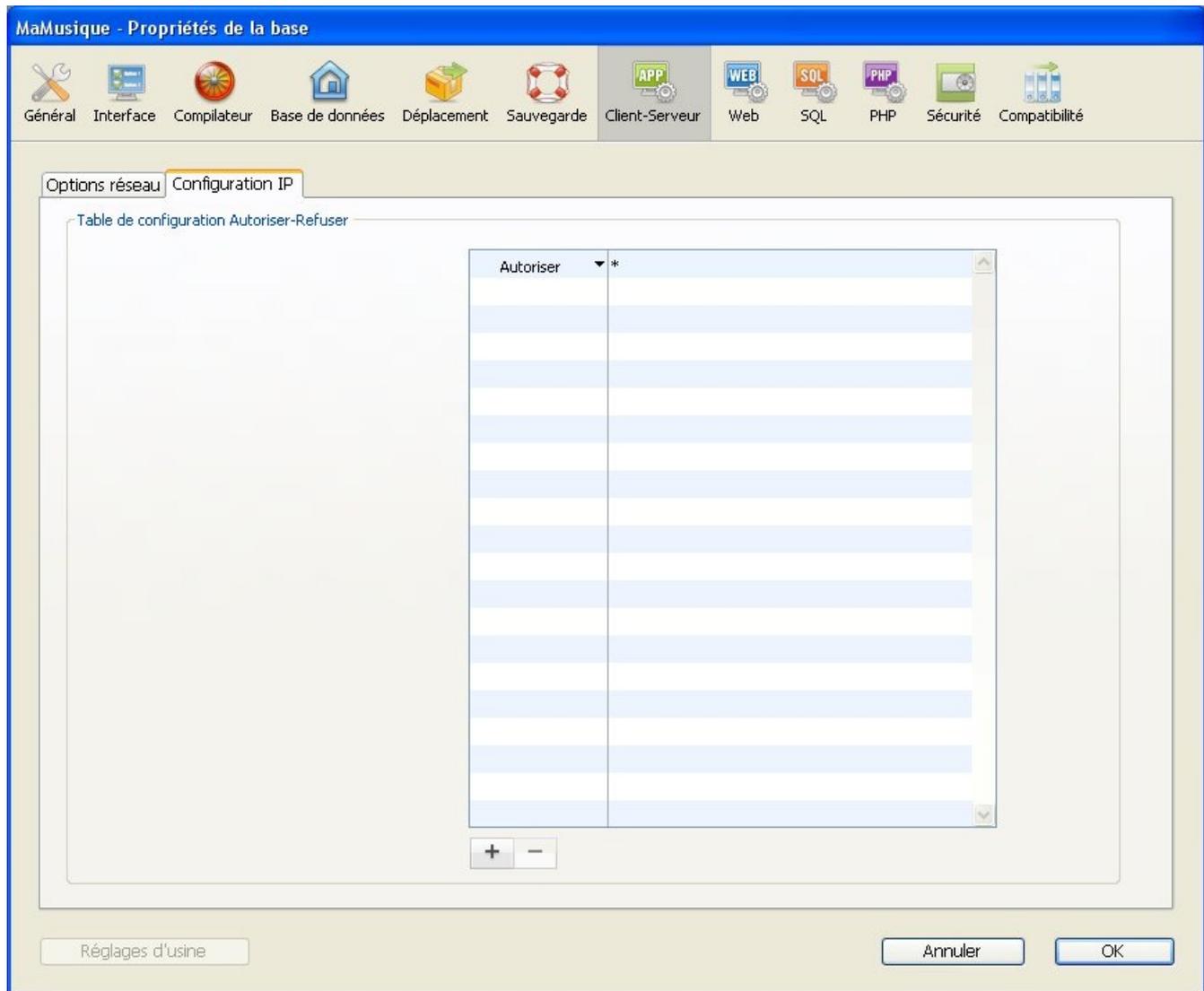
- **Jamais** : Le dossier **Resources** local n'est pas mis à jour en cours de session. La notification envoyée par le serveur est ignorée. Le dossier **Resources** local pourra toutefois être mis à jour manuellement via la commande **Mise à jour des ressources**.
- **Toujours** : La synchronisation du dossier **Resources** local est automatiquement effectuée en cours de session lorsque la notification est envoyée par le serveur.
- **Demander** : Lorsque la notification est envoyée par le serveur, une boîte de dialogue s'affiche sur les postes clients, signalant la modification. L'utilisateur peut accepter ou refuser la synchronisation du dossier **Resources** local. Le dossier **Resources** centralise les fichiers personnalisés nécessaires à l'interface de la base (fichiers de traduction, images, etc.). Des mécanismes automatiques ou manuels permettent de notifier chaque client lorsque le contenu de ce dossier a été modifié. Pour plus d'informations, reportez-vous à la section **Gestion du dossier Resources**.

### Ouvrir la structure en mode

Cette option permet de définir le mode d'ouverture de la structure de la base par les postes clients. Par défaut, le mode **Lecture écriture** est défini, mais vous pouvez configurer l'ouverture en mode **Lecture seulement** afin d'empêcher toute modification de la structure.

## Configuration IP

Vous pouvez définir les paramètres relatifs à la configuration réseau des bases 4D Server dans l'onglet "Configuration IP" de la page **Client-Serveur** des Propriétés de la base (accessibles depuis les postes 4D distants et 4D Server) :



Ces paramètres sont décrits dans cette section.

### Table de configuration Autoriser-Refuser

Cette table vous permet de définir des règles de contrôle d'accès à la base en fonction de l'adresse IP des postes 4D distants. Cette option permet de renforcer la sécurité par exemple pour des applications stratégiques.

**Note :** Cette table de configuration ne contrôle pas les connexions Web.

Le fonctionnement de la table de configuration est le suivant :

- La colonne "Autoriser-Refuser" permet de sélectionner le type de règle à appliquer (Autoriser ou Refuser) à l'aide d'un pop up menu. Pour ajouter une règle d'adresses, cliquez sur le bouton Ajouter. Une nouvelle ligne apparaît dans la table. Le bouton Supprimer permet de supprimer la ligne courante.
- La colonne "Adresse IP" permet de désigner la ou les adresse(s) IP concernées par la règle. Pour spécifier une adresse, cliquez dans la colonne et saisissez l'adresse sous la forme 123.45.67.89. Vous pouvez utiliser le caractère \* (étoile) pour spécifier des adresses du type "commence par". Par exemple, 192.168.\* indique toutes les adresses débutant par 192.168.
- L'application des règles s'effectue dans l'ordre d'affichage de la table. Si deux règles sont contradictoires, la priorité sera accordée à la règle située le plus haut dans le tableau. Vous pouvez réordonner les lignes en modifiant le tri courant (cliquez sur un en-tête de colonne pour alterner le sens de tri). Vous pouvez également déplacer des lignes par glisser-déposer.
- Pour des raisons de sécurité, seules les adresses correspondant à une règle d'autorisation explicite pourront se connecter. En particulier, si la table contient uniquement une ou plusieurs règle(s) de type Refuser, toutes les adresses seront refusées car aucune ne satisfera à au moins une règle. Si vous souhaitez refuser certaines adresses et autoriser toutes les autres, ajoutez une règle Autoriser \* à la fin de la table. Par exemple :

- Refuser 192.168.\* (refuser toutes adresses débutant par 192.168)
- Autoriser \* (et autoriser les autres)

Par défaut, aucune restriction de connexion n'est appliquée par 4D Server : la première ligne de la table contient le libellé Autoriser et le caractère \* (toute adresse).

## Crypter les connexions client/serveur

---

Vous pouvez configurer vos connexions client/serveur de manière à ce que 4D Server et les postes 4D distants communiquent en mode sécurisé.

La communication client/serveur sécurisée s'appuie sur le protocole SSL (Secured Socket Layer).

### Qu'est-ce que le protocole SSL dans le cadre des connexions client/serveur ?

---

Le protocole SSL a pour but de sécuriser les communications entre deux applications — généralement un serveur Web et un navigateur. Il comporte diverses fonctions permettant d'identifier les intervenants, de crypter les données ou encore de vérifier l'intégrité des données échangées. Pour une description détaillée du protocole SSL, reportez-vous, dans le manuel Langage de 4D, à la section **Utiliser le protocole SSL**.

Avec 4D Server et 4D Client, les mécanismes de génération de clés et d'authentification sont "transparents", ils sont gérés par 4D Server et ne nécessitent pas d'intervention de l'utilisateur.

Le cryptage des connexions client/serveur permet donc de renforcer la sécurité de votre application 4D Server, toutefois ce mode de fonctionnement ralentit les échanges.

### Configuration

---

Au niveau de l'architecture réseau, le protocole SSL s'insère entre la couche TCP/IP (bas niveau) et le protocole de haut niveau.

Pour pouvoir utiliser SSL dans une configuration client/serveur, assurez-vous de disposer, sur le poste serveur et sur chaque poste client, du fichier *4DSL.DLL*. Ce fichier constitue l'interface de la couche sécurisée (Secured Layer Interface) dédiée à la gestion du SSL. Il doit se trouver :

- sous Windows : à côté du fichier exécutable des applications 4D et 4D Server
- sous Mac OS : dans le sous-dossier [*4D Extensions*] des progiciels 4D et 4D Server.  
Ce fichier est installé par défaut.

Vous devez en revanche activer les connexions en mode sécurisé.

Pour cela, ouvrez la page "Client-Serveur/Options réseau" de la boîte de dialogue des Propriétés de la base et cochez l'option **Crypter les connexions Client-Serveur** dans la zone "Cryptage" (cf. section **Options réseau et Client-serveur**).

Par défaut, l'option n'est pas cochée.

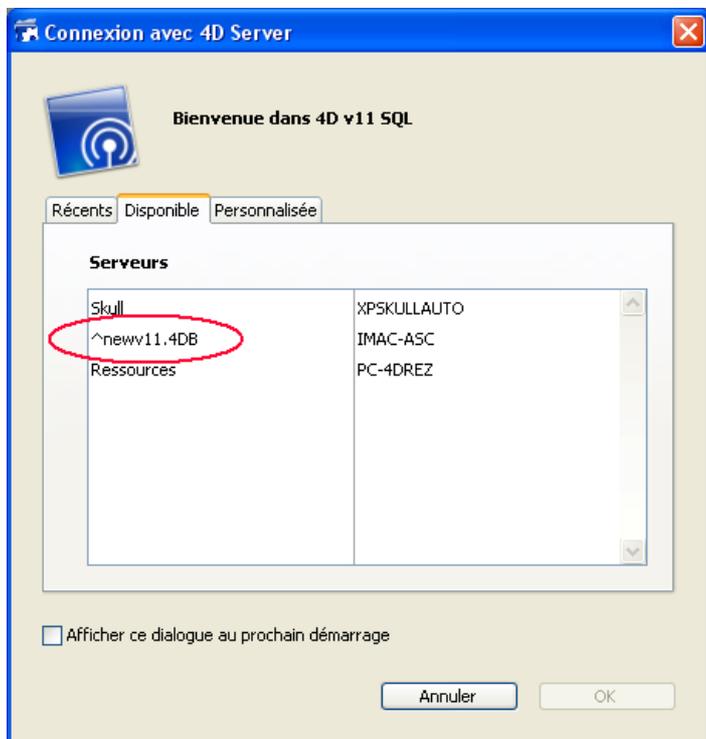
Vous devez ensuite quitter puis relancer l'application 4D Server pour que ce paramètre soit pris en compte.

Les postes 4D distants se connectent dès lors en mode sécurisé.

### Connexion en mode sécurisé

---

Le symbole "accent circonflexe" (^) apparaît devant le nom des bases publiées en mode SSL dans la page de connexion TCP/IP à 4D Server :



**Note :** Lorsque le nom d'une base n'apparaît pas dynamiquement dans la page de connexion TCP/IP, l'utilisateur peut le saisir manuellement dans la page **Personnalisée** (voir sections [Connexion à une base 4D Server](#) et [Configuration IP](#)). Dans ce cas, il faut impérativement faire débuter le nom par un ^ si la base est publiée en mode sécurisé, sinon la connexion sera refusée.

Le dossier **Resources** de la base permet de partager des données personnalisées (images, fichiers, sous-dossiers...) entre le poste serveur et tous les postes clients. Sur le poste serveur, le dossier **Resources** doit simplement être situé à côté du fichier de structure de la base.

Tous les mécanismes de référencement associé au dossier **Resources** sont pris en charge en mode client/serveur (dossier .lproj, XLIFF, images...). Pour plus d'informations sur ce point, reportez-vous au manuel Mode Développement de 4D.

Chaque client dispose en local d'une copie de ce dossier. Le contenu du dossier local est automatiquement synchronisé avec celui du serveur au moment de la connexion du client.

En outre, les postes clients peuvent être "notifiés" dynamiquement en cours de session lorsque le contenu du dossier **Resources** de la base serveur a été modifié par un développeur. Cette notification peut être déclenchée :

- automatiquement par le serveur, deux minutes après la dernière modification effectuée par un client (cette temporisation permet d'éviter les notifications intempestives en cas de copie de nombreux fichiers).
- manuellement via la commande **Aviser les clients** dans le menu d'action de l'Explorateur de ressources sur le poste client à l'origine de la modification.
- par programmation, via la commande **NOTIFIER MODIFICATION DOSSIER RESOURCES**. Cette commande est utile en cas de modification du contenu du dossier Resources sur le poste serveur via une procédure stockée.

Côté client, le mode de prise en compte de l'information de modification (la notification) dépend du paramétrage de la préférence "Mise à jour du dossier "Resources" en cours de session". Elle peut également être définie individuellement via la commande **FIXER PARAMETRE BASE**. Trois choix sont proposés : **ne jamais synchroniser**, **synchroniser automatiquement** ou **demander**. Pour plus d'informations, reportez-vous à la section **Options réseau et Client-serveur** et à la description de la commande **FIXER PARAMETRE BASE**.

Enfin, chaque poste client peut à tout moment se synchroniser avec le serveur via la commande **Mise à jour des ressources** du menu d'action de l'Explorateur de ressources. Pour plus d'informations sur l'Explorateur de ressources, reportez-vous au manuel Mode Développement de 4D.

**Note de compatibilité** : Dans les versions précédentes de 4D Server, le transfert de données personnalisées était effectué via un dossier nommé "Extras", placé à côté du fichier de structure. Ce dossier étant désormais obsolète, son utilisation est déconseillée. Il reste toutefois pris en charge par 4D Server afin de préserver la compatibilité des applications existantes.

## Enregistrer une base comme service

---

Sous Windows, 4D Server peut être lancé comme un Service.

**Note de compatibilité** : Cette fonction n'est plus disponible sous Mac OS à compter de la version 12 de 4D Server.

Une application 4D Server enregistrée comme service est automatiquement lancée au démarrage de la machine avec la base courante, avant même l'ouverture d'une session utilisateur. Elle n'est pas fermée lorsque l'utilisateur quitte sa session.

Ce fonctionnement permet de garantir la disponibilité d'une base 4D Server, même en cas d'incident nécessitant le redémarrage de la machine. La maintenance peut être effectuée à distance.

### Notes :

- Sur une plate-forme Windows 64 bits, une application 4D Server enregistrée comme service est exécutée sans interface (la fenêtre d'administration du serveur n'apparaît pas).
- Pour plus d'informations sur les mécanismes de gestion des Services, reportez-vous à la documentation de votre système d'exploitation.

Pour enregistrer la base de données 4D Server comme un Service, sélectionnez la commande **Enregistrer la base courante comme Service** dans le menu **Fichier** de 4D Server. Au prochain démarrage de la machine, 4D Server sera automatiquement lancé et la base courante ouverte. Vous pouvez enregistrer autant de bases de données que vous le souhaitez (mais une seule instance par base).

**Note** : Sous Windows Vista et Windows 7, cette commande peut être grisée car l'accès aux fonctions de gestion des services est restreint. Pour pouvoir utiliser cette commande, vous devez soit :

- Lancer 4D Server avec un niveau administrateur (pour cela, effectuez un clic droit sur l'icône de l'application et choisissez la commande **Exécuter en tant qu'administrateur** dans le menu contextuel).
- Désactiver le Contrôle de compte utilisateur (UAC) via le panneau de configuration "Comptes d'utilisateur" (pour cela, sous Windows 7 choisissez l'option "Ne jamais m'avertir", sous Windows Vista, désélectionnez l'option "Utiliser le Contrôle des Comptes d'utilisateurs pour vous aider à protéger votre ordinateur").

**Important** : Veillez à utiliser pour l'ouverture de session un compte d'utilisateur valide et ayant accès à une imprimante, sinon un message d'erreur s'affichera. L'accès aux paramètres des services s'effectue généralement via le panneau de configuration Outils d'administration/Services.

- Si vous souhaitez désenregistrer votre base de données courante, sélectionnez **Désenregistrer la base courante** à partir du menu **Fichier** de 4D Server. Cette commande est grisée si la base n'est pas enregistrée comme service.
- Si vous souhaitez désenregistrer toutes les bases de 4D Server à la fois, sélectionnez **Désenregistrer tous les services serveur** à partir du menu **Fichier** de 4D Server. Cette commande est grisée si aucun service 4D Server n'est actif.

Vous ne pouvez pas désenregistrer 4D Server à partir du menu de 4D Server si l'application a été lancée comme un service au démarrage car les trois commandes de menu sont désactivées. Il faut que vous passiez par le panneau de configuration des Services pour arrêter le services du serveur.

## Mise en place d'un miroir logique

---

4D Server propose une solution intégrée permettant de mettre en place un système de sauvegarde des données par miroir logique. Cette solution est basée sur deux commandes : **Nouveau fichier historique** et **INTEGRER FICHIER HISTORIQUE**.

### Qu'est-ce qu'un miroir logique ?

---

Le miroir logique est un mode de sauvegarde sophistiqué, principalement destiné aux bases de données critiques ou à forte charge d'exploitation.

Utiliser un miroir logique consiste à exploiter une base sur un premier poste, et à maintenir sur une deuxième machine une copie de cette base, périodiquement mise à jour. Les deux machines communiquent par le réseau, la machine en exploitation transmettant régulièrement à la machine miroir les évolutions de la base par l'intermédiaire du fichier d'historique.

De cette façon, en cas d'un incident sur la base en exploitation, il n'y a qu'à repartir de la base miroir pour reprendre très rapidement l'exploitation sans aucune perte de données. En outre, la base en exploitation n'est jamais "bloquée" par les opérations de sauvegarde.

### Pourquoi choisir la sauvegarde par miroir logique ?

---

L'emploi d'un miroir logique correspond à des besoins spécifiques. La stratégie standard basée sur une sauvegarde périodique et le fichier d'historique constitue dans la plupart des cas une solution simple, fiable et peu coûteuse. La base est sauvegardée régulièrement (toutes les 24 heures en général). Durant la sauvegarde, la base reste accessible en mode lecture seulement. Cette période d'indisponibilité partielle est très courte, et même pour les bases de données importantes (plus de 2 Go) elle ne dépasse pas 5 minutes. Cette opération peut même être programmée en-dehors des périodes d'utilisation de la base.

Cependant, pour certains organismes, comme par exemple les hôpitaux, les bases de données critiques doivent être entièrement opérationnelles 24h/24. La base ne peut pas être en lecture seulement, même durant un laps de temps très court. Dans ce cas, la mise en place d'un miroir logique est une solution adaptée.

**Note :** La base miroir ne reflète que les modifications apportées aux **données**. Ce mode de sauvegarde n'est donc pas adapté pour des bases en cours de développement, où les fréquentes modifications de structure rendront rapidement le miroir obsolète ou nécessiteront de multiples réactualisations de la structure de la base miroir.

### Principes de fonctionnement

---

La mise en place d'un système de sauvegarde par miroir logique s'appuie sur deux commandes : **Nouveau fichier historique** et **INTEGRER FICHIER HISTORIQUE**. Ces commandes sont décrites dans le manuel Langage de 4D.

Les principes mis en oeuvre sont les suivants :

- La base est installée sur le poste 4D Server principal (poste en exploitation) et une copie identique de la base est installée sur le poste 4D Server miroir.
- Un test au démarrage de l'application (par exemple la présence d'un fichier spécifique dans le dossier 4D Extensions) permet de distinguer chaque version (en exploitation et en miroir) et donc d'exécuter les opérations appropriées.
- Sur le poste 4D Server en exploitation, le fichier d'historique est "segmenté" à intervalle régulier à l'aide de la commande **Nouveau fichier historique**. Aucune sauvegarde n'étant effectuée sur le serveur principal, la base est en permanence disponible en lecture-écriture.
- Chaque "segment" de fichier d'historique est envoyé sur le poste miroir, où il est intégré à la base miroir à l'aide de la commande **INTEGRER FICHIER HISTORIQUE**.

La mise en place de ce système nécessite la programmation de code spécifique, notamment :

- un minuteur sur le serveur principal pour la gestion des cycles d'exécution de la commande **Nouveau fichier historique**,
- un système de transfert des "segments" de fichier d'historique entre le poste en exploitation et le poste miroir (utilisation de 4D Internet Commands pour un transfert via ftp ou messagerie, Web Services...),
- un process sur le poste miroir destiné à superviser l'arrivée de nouveaux "segments" de fichier d'historique et à les intégrer via la commande **INTEGRER FICHIER HISTORIQUE**,
- un système de communication et de gestion d'erreurs entre le serveur principal et le serveur miroir.

**ATTENTION :** La sauvegarde par miroir logique est incompatible avec les sauvegardes "standard" sur la base en exploitation car l'emploi simultané de ces deux modes de sauvegarde entraîne la désynchronisation de la base en exploitation et de la base miroir. Par conséquent, vous devez veiller à ce qu'aucune sauvegarde, automatique ou manuelle, ne soit effectuée sur la base en exploitation. En revanche, il est possible de sauvegarder la base miroir (cf. paragraphe suivant).

### Sauvegarde de la base miroir

---

4D Server permet d'effectuer des sauvegardes de la base sur le poste miroir.

Tous les moyens classiques peuvent être utilisés pour effectuer les sauvegardes sur le poste miroir : sauvegarde manuelle via la

commande du menu **Fichier**, sauvegarde périodique définie dans les Préférences ou sauvegarde programmée à l'aide des commandes du langage.

Pour éviter tout risque de désynchronisation avec le poste en exploitation, 4D verrouille automatiquement le poste miroir lorsqu'il effectue l'une ou l'autre des deux opérations fondamentales : intégration de l'historique en provenance du poste en exploitation et sauvegarde de la base miroir.

- Lorsqu'une intégration de l'historique est en cours, il n'est pas possible de déclencher une sauvegarde. Si la commande **SAUVEGARDER** est utilisée, l'erreur 1417 est générée (cf. section).

- Lorsqu'une sauvegarde est en cours, tous les process sont gelés, il n'est donc pas possible de démarrer une intégration d'historique.

## Scénario d'exploitation d'un miroir logique

Le scénario suivant illustre, du point de vue de chaque poste 4D Server, la mise en place d'un système de sauvegarde avec miroir :

Poste en exploitation	Poste miroir
Démarrage de l'application, sauvegarde du fichier de données et activation (si nécessaire) du fichier d'historique. 4D crée le fichier MaBase.journal On quitte l'application	
Copie de tous les fichiers de la base (fichier d'historique inclus) sur le poste miroir	
Démarrage de l'application et passage en exploitation.	Démarrage de l'application miroir. 4D Server demande le fichier d'historique : sélection du fichier MaBase.journal transféré depuis le poste en exploitation.
Décision de mettre à jour le miroir (par exemple, après une certaine durée d'exploitation)	
Exécution de la méthode contenant la commande Nouveau fichier historique. Le fichier sauvegardé est nommé MaBase[0001-0000].journal	
Envoi par programmation du fichier MaBase[0001-0000].journal au poste miroir (utilisation de 4DIC, Web Services, etc.)	
La base est en exploitation	Détection qu'un fichier est en attente d'intégration. Exécution de la méthode contenant la commande <b>INTEGRER FICHIER HISTORIQUE</b> pour intégrer le fichier MaBase[0001-0000].journal. Ce fichier devient le nouveau fichier d'historique courant.
Incident sur le poste, la base de données est inutilisable. Décision de passer sur la machine miroir.	
Copie du fichier d'historique courant MaBase.journal sur le poste miroir, dans le dossier de réception habituel.	
Réparation de la machine...	Détection qu'un fichier est en attente d'intégration. Exécution de la méthode contenant la commande <b>INTEGRER FICHIER HISTORIQUE</b> pour intégrer le fichier MaBase.journal. Ce fichier devient le nouveau fichier d'historique courant. La base est en exploitation
La machine est réparée	On quitte la base
Remplacement des fichiers de la base par ceux de la base miroir	
Démarrage de l'application. 4D Server demande le fichier d'historique : sélection du fichier MaBase.journal transféré depuis le poste miroir.	Démarrage de l'application miroir.
La base est en exploitation	

## Fenêtre d'administration de 4D Server

-  Page Moniteur
-  Page Utilisateurs
-  Page Process
-  Page Maintenance
-  Page Serveur d'application
-  Page Serveur SQL
-  Page Serveur HTTP

## Page Moniteur

4D Server bénéficie d'une fenêtre d'administration complète et ergonomique.

Cette fenêtre propose différents outils d'analyse et de contrôle de la base de données publiée. La fenêtre comporte sept pages, accessibles via une zone de boutons située dans la partie supérieure :



Chaque page est détaillée dans une section de ce chapitre.

**Note :** La fenêtre d'administration est accessible via un 4D distant. Pour plus d'informations sur ce point, reportez-vous à la section **Administration à distance**.

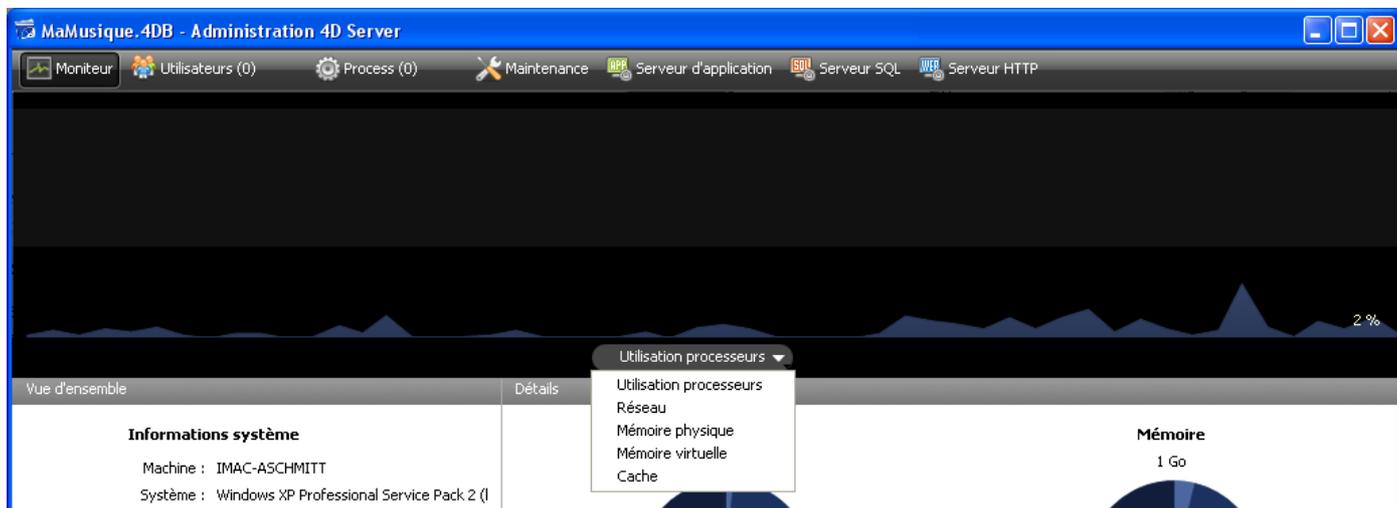
La Page Moniteur affiche des informations dynamiques relatives à l'exploitation de la base de données ainsi que des informations sur le système et l'application 4D Server :



**Note :** Sous Windows, l'affichage de ces informations est lié aux droits de l'utilisateur ayant ouvert la session. Pour plus d'informations, reportez-vous ci-dessous au paragraphe "Affichage des informations du Moniteur (Windows)".

## Zone graphique

La zone graphique permet de visualiser l'évolution en temps réel de plusieurs paramètres : le taux d'utilisation des processeurs, le trafic réseau et l'état de la mémoire. Vous sélectionnez le paramètre à afficher via le menu situé au centre la fenêtre :



- **Utilisation processeurs** : Taux d'utilisation globale du ou des processeur(s) de la machine, toutes applications confondues. La part spécifique de 4D Server dans ce taux d'utilisation est fournie dans la zone d'informations "Processeurs".
- **Réseau** : Nombre d'octets reçus par seconde par 4D Server. Le nombre d'octets envoyés par 4D Server est fourni dans la zone d'informations "Réseau".
- **Mémoire physique** : Quantité de mémoire RAM de la machine utilisée par 4D Server. Une vue plus détaillée de l'utilisation de la mémoire est fournie dans la zone d'informations "Mémoire".
- **Mémoire virtuelle** : Affiche dans la zone graphique la quantité de mémoire virtuelle utilisée par l'application 4D Server. Cette mémoire est allouée par le système en fonction des besoins de l'application. La valeur située en bas à droite de la zone indique la quantité de mémoire en cours d'utilisation. La valeur située en haut à gauche indique la quantité maximale de mémoire virtuelle utilisable. La valeur maximale est calculée dynamiquement en fonction des paramètres mémoire généraux de l'application.
- **Cache** : Affiche dans la zone graphique la quantité de mémoire cache utilisée par l'application 4D Server. La valeur située en bas à droite de la zone indique la quantité de mémoire en cours d'utilisation. La valeur située en haut à gauche indique la taille totale de la mémoire cache, telle que définie via les Propriétés de la base.  
A noter que lorsque cette option est sélectionnée, le défilement de la zone graphique est ralenti car une analyse efficace du cache s'effectue généralement sur une période d'observation assez longue.

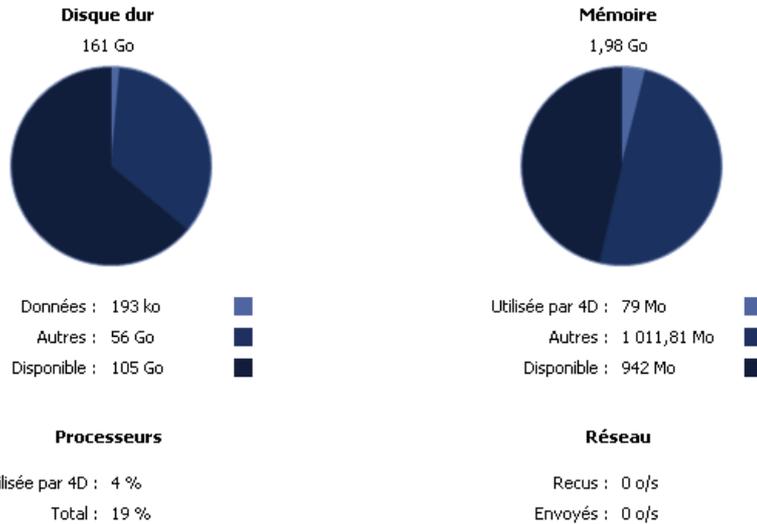
## Zone Vue d'ensemble

La zone "Vue d'ensemble" fournit diverses informations relatives au système, à l'application et aux licences installées sur la machine de 4D Server.

Vue d'ensemble	
<p><b>Informations système</b></p> <p>Machine : IMAC-ASCHMITT            Système : Windows XP (5.1)            Processeur : Intel(R) Core(TM)2 CPU T7400 @ 2.16GH            Adresse IP : 192.168.88.103</p>	Ordinateur, système et adresse IP du serveur
<p><b>Informations application</b></p> <p>Version : 11.2</p>	Numéro de version interne de 4D Server
<p><b>Connexions maximum</b></p> <p>Application Server : 2            Serveur SQL : Illimitées            Serveur Web : Aucune            Serveur SOAP : Aucune</p>	Nombre de connexions simultanées autorisées par type de serveur
<p><b>Licence</b></p> <p>Nom : 4D Server v11 SQL _ Internal Use Only            until 1 April 2008            Accordée à : Arnaud 4D SAS</p>	Description de la licence

## Zone Détails

La zone "Détails" reprend une partie des informations affichées dans la zone graphique et propose des informations complémentaires.



- **Disque dur** : Capacité globale du disque dur et répartition entre l'espace occupé par les données de la base (fichier de données + index des données), l'espace occupé par les autres fichiers et l'espace disponible.
- **Mémoire** : Mémoire RAM installée sur la machine et quantité de mémoire occupée par 4D Server, par les autres applications ainsi que mémoire disponible.  
La mémoire occupée par 4D Server peut également être affichée dynamiquement dans la zone graphique.
- **Processeurs** : Taux instantané d'occupation du ou des processeurs(s) de la machine par 4D Server et par les autres applications. Ce taux est recalculé en permanence.  
Le taux d'occupation par 4D Server peut également être affiché dynamiquement dans la zone graphique.
- **Réseau** : Nombre instantané d'octets reçus via le réseau par 4D Server nombre d'octets envoyés par l'application. Cette valeur est réactualisée en permanence.  
Le nombre d'octets reçus par 4D Server peut également être affiché dynamiquement dans la zone graphique.

## Affichage des informations du Moniteur (Windows)

Sous Windows, certaines informations système affichées dans la page Moniteur sont récupérées via les outils de l'"Analyseur de Performance" de Windows. L'accès à ces outils n'est permis que si l'utilisateur ayant ouvert la session à partir de laquelle a été lancée 4D Server dispose des autorisations nécessaires. Cet utilisateur doit soit :

- appartenir au groupe "Administrateurs",
- sous Windows Vista : appartenir au groupe "Utilisateurs de l'Analyseur de Performances" (pour un utilisateur non administrateur)

Pour placer un utilisateur non-administrateur dans le groupe "Utilisateurs de l'Analyseur de Performances" sous Windows Vista (vous devez utiliser un compte Administrateur pour effectuer les manipulations) :

1. Allez dans le Panneau de Configuration et ouvrez le panneau "Comptes d'utilisateurs".
2. Cliquez sur l'onglet "Options avancées" puis sur le bouton "Avancé" de la partie "Gestion avancée des utilisateurs".  
L'application "lusrmgr" s'exécute.
3. Double-cliquez sur le dossier "Groupes".
4. Double-cliquez sur le groupe "Utilisateurs de l'Analyseur de Performances".  
Une fenêtre nommée "Propriétés de Utilisateurs de l'Analyseur de Performances" apparaît.
5. Cliquez sur le bouton Ajouter... afin d'ajouter un utilisateur.
6. Dans la zone de texte intitulée "Entrez les noms des objets à sélectionner", saisissez les noms des utilisateurs à autoriser.
7. Cliquez sur OK (deux fois).  
Fermez l'application "lusrmgr" puis les "Comptes d'utilisateurs".



Il est possible de saisir plus d'une valeur à chercher. Utilisez un point-virgule pour séparer les valeurs. L'opérateur utilisé dans ce cas est du type OU.

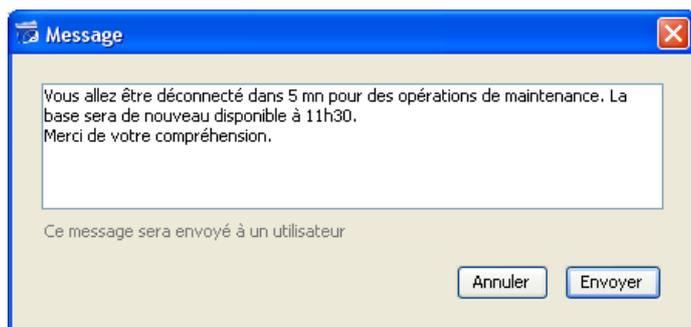
Par exemple, si vous saisissez "Jean;Marie;Pierre", seules les lignes comportant Jean OU Marie OU Pierre dans l'une des colonnes cibles seront conservées.

## Boutons d'administration

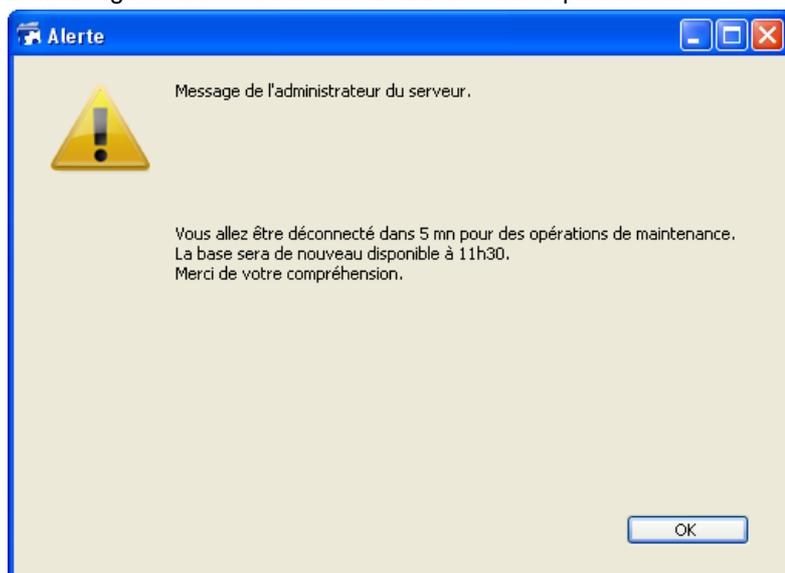
---

La page comporte trois boutons de commande. Ces boutons sont actifs si au moins une ligne est sélectionnée. Vous pouvez sélectionner plusieurs lignes en appuyant sur la touche **Maj** pour une sélection continue ou **Ctrl** (Windows) / **Commande** (Mac OS) pour une sélection discontinue.

- Envoyer message : Ce bouton permet d'envoyer un message aux utilisateurs 4D sélectionnés dans la fenêtre. Si aucun utilisateur 4D n'est sélectionné, le bouton est désactivé. Lorsque vous cliquez sur le bouton, une boîte de dialogue apparaît, vous permettant de saisir le message. La boîte de dialogue indique le nombre d'utilisateurs qui recevront le message ;



Le message sera affiché sous forme d'alerte sur les postes clients :



- Visualiser process : Ce bouton permet de visualiser directement les process du ou des utilisateur(s) sélectionné(s) dans la **Page Process** de la fenêtre d'administration. Lorsque vous cliquez sur ce bouton, 4D Server bascule sur la page Process et pré-remplit la zone de recherche/filtrage de cette page avec les noms des utilisateurs sélectionnés. Pour plus d'informations, reportez-vous à la description de cette page.
- Déconnecter : Ce bouton permet de forcer la déconnexion du ou des utilisateur(s) sélectionné(s). Lorsque vous cliquez sur ce bouton, une boîte de dialogue d'alerte apparaît, vous permettant de confirmer ou d'annuler l'opération.

**Note** : Vous pouvez déconnecter directement les utilisateurs sélectionnés sans afficher la boîte de dialogue de confirmation : pour cela, appuyez sur la touche **Alt** (Windows) ou **Option** (Mac OS) avant de cliquer sur le bouton **Déconnecter**.

La Page Process liste les process en cours d'exécution :

Type	Session	Nom de process	Num	État	Temps CPU	Activité
Serveur SQL		SHARED pool worker 1	0	En cours d'exécution	00:00:00	0,00 %
Serveur SQL		SQL Net Session Manager Thread	0	En cours d'exécution	00:00:00	0,00 %
Serveur d'application	-	Gestionnaire de client	3	En attente de sémaphore interne	00:00:05	0,00 %
Serveur d'application	-	Interface utilisateur	1	En attente d'entrée-sortie	00:03:05	34,29 %
Serveur d'application	-	Process minuteur interne	2	En cours d'exécution	00:05:58	0,00 %
Serveur d'application	-	Process passerelle interne	4	En attente de sémaphore interne	00:01:31	0,00 %
Process 4D Client	arnaud schmitt	arnaud schmitt	6	En attente d'entrée-sortie	00:00:02	0,00 %
Process base 4D Client	arnaud schmitt	arnaud schmitt	0	En cours d'exécution	00:00:00	0,00 %
Process base 4D Client	arnaud schmitt	P_11	0	En cours d'exécution	00:00:00	0,00 %
Process 4D Client	arnaud schmitt	P_11	8	En attente d'entrée-sortie	00:00:00	0,00 %
Process 4D Client	arnaud schmitt	Process développement	7	En attente d'entrée-sortie	00:00:02	0,00 %
Process base 4D Client	arnaud schmitt	Process développement	0	En cours d'exécution	00:00:00	0,00 %
Process base 4D Client	arnaud schmitt	Process principal	0	En cours d'exécution	00:00:00	0,00 %
Process 4D Client	arnaud schmitt	Process principal	5	En attente d'entrée-sortie	00:00:03	0,00 %

Le bouton "Process" indique entre parenthèses le nombre total de process en cours d'exécution à la base (ce nombre ne tient pas compte des éventuels filtres d'affichage appliqués à la fenêtre ni de l'état de l'option **Afficher les process par groupes**).



Vous pouvez modifier l'ordre des colonnes par simple glisser-déposer de la zone d'en-tête des colonnes. Vous pouvez également trier la liste sur les valeurs d'une colonne en cliquant sur son en-tête.

Comme la **Page Utilisateurs**, cette page contient une zone de recherche/filtrage dynamique, permettant de réduire le nombre de lignes affichées dans la liste à celles qui correspondent au texte saisi dans la zone de recherche. La recherche/le filtrage est effectué(e) dans les colonnes Session et Nom de process.



Vous disposez également de trois boutons-raccourcis permettant de filtrer par famille les process affichés dans la fenêtre :

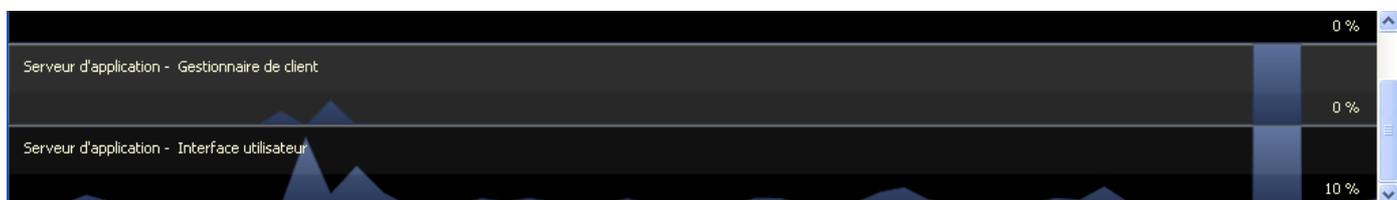


- **Process utilisateurs** : Process générés par et pour les sessions utilisateurs. Ces process sont précédés d'une icône en forme de personnage .
- **Process 4D** : Process générés par le moteur de 4D Server. Ces process sont précédés d'une icône en forme de roue crantée .
- **Process en attente** : Process inactifs mais conservés temporairement et pouvant être réutilisés à tout moment. Ce mécanisme permet d'optimiser la réactivité de 4D Server. Ces process sont précédés d'une icône grisée en forme de personnage .

L'option **Afficher les process par groupes** vous permet de regrouper les process internes de 4D Server ainsi que les process clients, pour plus de lisibilité. Lorsque vous cochez cette option :

- les process clients 4D "jumeaux" (Process client 4D principal et Process base 4D client, cf. paragraphe "Type du process") sont groupés en un seul,
- le groupe "Gestionnaires de tâches" est créé ; il inclut les process internes dédiés à la répartition des tâches (Shared balancer, Net session manager, Exclusive pool worker),
- le groupe "Gestionnaires clients" est créé ; il inclut les différents process internes clients.

La zone inférieure de la fenêtre permet d'afficher la représentation graphique de l'activité du ou des process sélectionné(s) :



**Note :** Vous pouvez sélectionner plusieurs lignes en appuyant sur la touche **Maj** pour une sélection continue ou **Ctrl** (Windows) / **Commande** (Mac OS) pour une sélection discontinue.

L'activité du process est le pourcentage du temps que 4D Server a consacré à ce process (ratio).

La fenêtre fournit les informations suivantes pour chaque process :

- Type du process (cf. ci-dessous),
- Session (vide dans le cas d'un process 4D et nom de l'utilisateur 4D dans le cas d'un process utilisateur)
- Nom du process,
- Numéro du process (tel que retourné par la fonction **Nouveau process** par exemple). Le numéro du process est le numéro attribué sur le serveur. Dans le cas d'un process global, ce numéro peut être différent de celui attribué sur le poste client.
- Etat courant du process,
- Temps (en secondes) d'exécution du process depuis sa création,
- Pourcentage du temps que 4D Server a consacré à ce process (ratio).

## Type du process

Chaque process est identifié par une icône et un type. La couleur et la forme de l'icône indiquent la famille du process :

	Serveur d'application
	Serveur SQL
	Serveur DB4D (moteur de base de données)
	Serveur Web
	Serveur SOAP
	Process client 4D protégé (process développement d'un 4D connecté)
	Process client 4D principal (process principal d'un 4D connecté. Process collaboratif, équivalent sur le serveur du process créé sur le poste client)
	Process base 4D client (process parallèle à un process 4D client. Process préemptif chargé de contrôler le process client 4D principal correspondant)
	Process en attente (ancien ou futur "Process client 4D base de données")
	Process 4D client (process tournant sur le 4D connecté)
	Procédure stockée (process lancé par un 4D connecté et tournant sur le serveur)
	Méthode Web (lancée par un 4D ACTION par exemple)
	Méthode SOAP (lancée par un Web Service)
	Méthode SQL (lancée par une requête SQL)

**Note :** Un process client 4D principal et son process base 4D client "jumeau" sont regroupés lorsque l'option **Afficher les process par groupes** est cochée.

## Boutons d'administration

La page comporte cinq boutons de commande permettant d'agir sur le ou les process sélectionné(s). A noter que vous ne pouvez agir que sur les process utilisateurs.



- **Tuer process :** permet de tuer le ou les process sélectionné(s). Lorsque vous cliquez sur ce bouton, une boîte de dialogue d'alerte apparaît, vous permettant de confirmer ou d'annuler l'opération.  
**Note :** Vous pouvez tuer directement les process sélectionnés sans afficher la boîte de dialogue de confirmation : pour cela, appuyez sur la touche Alt (Windows) ou Option (Mac OS) avant de cliquer sur le bouton.
- **Endormir process :** permet d'endormir le ou les process sélectionné(s).
- **Réactiver process :** permet de réactiver le ou les process sélectionné(s). Les process doivent avoir été auparavant endormis (via le bouton précédent ou par programmation) sinon le bouton est sans effet.
- **Tracer process :** permet d'ouvrir sur le poste serveur une ou plusieurs fenêtre(s) du débogueur pour le ou les process sélectionné(s). Lorsque vous cliquez sur ce bouton, une boîte de dialogue d'alerte apparaît, vous permettant de confirmer ou d'annuler l'opération. A noter que la fenêtre du débogueur ne s'affiche que lorsque du code 4D est effectivement exécuté sur

le poste serveur (par exemple dans le cadre d'un trigger ou de l'exécution d'une méthode ayant l'attribut "Exécuter sur serveur").

**Note :** Vous pouvez déboguer un process directement, sans afficher la boîte de dialogue de confirmation : pour cela, appuyez sur la touche **Alt** (Windows) ou **Option** (Mac OS) avant de cliquer sur le bouton.

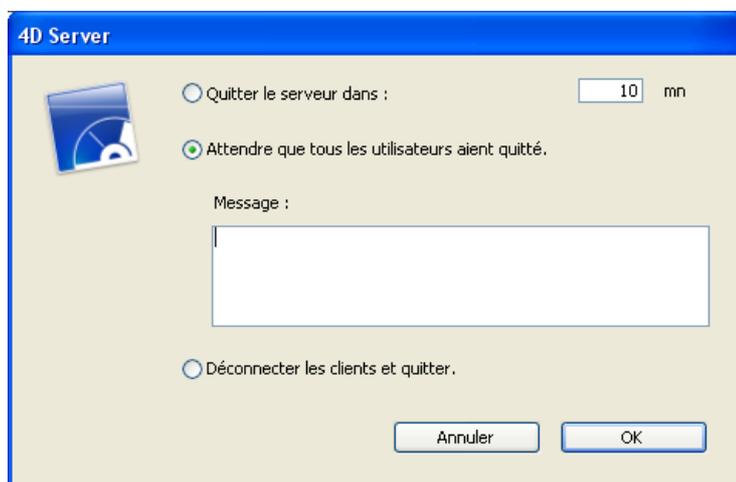
- Voir utilisateurs : permet d'afficher dans la page Utilisateurs tous les process du ou des utilisateur(s) sélectionné(s). Le bouton est actif lorsqu'un process utilisateur au moins est sélectionné.

La Page Maintenance fournit diverses informations relatives au fonctionnement courant de la base. Elle donne également accès aux fonctions de maintenance élémentaires :

The screenshot shows a web browser window titled "MyMusic" with a navigation bar containing "Moniteur", "Utilisateurs", "Process", "Maintenance", "Serveur d'application", "Serveur SQL", and "Serveur HTTP". The main content area is divided into several sections:

- Dernière vérification :** Date inconnue. Buttons: "Vérifier enregistrements et index", "Voir le compte rendu". Description: "La vérification peut vous aider à déceler des problèmes de performance ou de validité des index et des données."
- Dernier compactage :** Date inconnue. Buttons: "Compacter les données...", "Voir le compte rendu". Description: "En compactant la base vous réduisez la place prise par les données et optimisez leur organisation. Utilisez le compactage si vous constatez une dégradation des performances ou si vous voulez réduire le poids de vos données. Le compactage nécessite le redémarrage du serveur. Tous les utilisateurs seront déconnectés."
- Durée de fonctionnement :** 4 heures 0 minute. Button: "Redémarrer le serveur...". Description: "Le redémarrage du serveur entraînera la déconnexion de tous les utilisateurs."
- Dernière sauvegarde :** 05/12/2007 à 17:36. Prochaine sauvegarde : 00/00/00 à 00:00. Place nécessaire : / Place disponible : / Buttons: "Démarrer la sauvegarde", "Préférences...". Description: "Le serveur ne sera pas redémarré mais les utilisateurs seront bloqués pendant l'opération."
- Historique des requêtes :** 0 seconde enregistrées. Buttons: "Démarrer l'historique des requêtes", "Voir le compte rendu". Description: "Les performances du serveur pourront être légèrement altérées durant la génération de l'historique des requêtes. Cependant, l'analyse de cet historique permettra, par la suite, l'optimisation du serveur."

- **Dernière vérification :** Cette zone indique la date, l'heure et le statut de la dernière vérification des données effectuée sur la base. Pour plus d'informations sur la procédure de vérification des données, reportez-vous au manuel Mode Développement. Le bouton **Vérifier enregistrements et index** permet de lancer directement l'opération de vérification, sans interruption du serveur. A noter que le serveur peut être sensiblement ralenti durant l'opération. Tous les enregistrements et tous les index de la base sont vérifiés. Si vous souhaitez pouvoir cibler la vérification ou disposer d'options supplémentaires, vous devez utiliser le Centre de sécurité et de maintenance (CSM).  
A l'issue de la vérification, un fichier de compte-rendu est généré aux formats xml et html sur le serveur, dans le dossier Logs placé à côté du fichier de structure de la base. Le bouton **Voir le compte rendu** (nommé **Télécharger le compte rendu** si l'opération a été effectuée depuis un poste distant) vous permet d'afficher le fichier dans votre navigateur.
- **Dernier compactage :** Cette zone indique la date, l'heure et le statut du dernier compactage des données de la base. Pour plus d'informations sur la procédure de compactage des données, reportez-vous au manuel Mode Développement. Le bouton **Compacter les données** permet de lancer directement une opération de compactage des données. Cette opération nécessite de stopper le serveur : lorsque vous cliquez sur le bouton, la boîte de dialogue de fermeture de la base 4D Server apparaît, vous permettant de choisir le mode d'interruption de l'exploitation :



Pour plus d'informations sur cette boîte de dialogue, reportez-vous à la section [Quitter 4D Server](#).

Après l'interruption effective de la base, 4D Server effectue un compactage standard des données de la base. Si vous souhaitez disposer d'options supplémentaires, vous devez utiliser le Centre de sécurité et de maintenance (CSM).

Une fois le compactage terminé, 4D Server relance automatiquement la base. Les utilisateurs 4D peuvent alors se reconnecter.

**Note :** Si la demande de compactage a été effectuée depuis un client 4D distant, ce poste est automatiquement reconnecté par 4D Server.

Un fichier de compte-rendu est généré aux formats xml et html sur le serveur, dans le dossier Logs placé à côté du fichier de structure de la base. Le bouton **Voir le compte rendu** (nommé **Télécharger le compte rendu** si l'opération a été effectuée depuis un poste distant) vous permet d'afficher le fichier dans votre navigateur.

- **Durée de fonctionnement :** Cette zone indique la durée de fonctionnement du serveur depuis son dernier démarrage (jours, heures et minutes).

Le bouton **Redémarrer le serveur...** vous permet de provoquer un redémarrage immédiat du serveur. Lorsque vous cliquez sur ce bouton, la boîte de dialogue de fermeture de la base 4D Server apparaît, vous permettant de choisir le mode d'interruption de l'exploitation (cf. section [Quitter 4D Server](#)). Après le redémarrage, 4D Server relance automatiquement la base. Les utilisateurs 4D peuvent alors se reconnecter.

**Note :** Si la demande de redémarrage a été effectuée depuis un client 4D distant, ce poste est automatiquement reconnecté par 4D Server.

- **Dernière sauvegarde :** Cette zone indique la date et l'heure de la dernière sauvegarde de la base et fournit des informations relatives à la prochaine sauvegarde automatique, le cas échéant. Les sauvegardes automatiques sont paramétrées via la page "Périodicité" des préférences de la base.

- Prochaine sauvegarde : date et heure de la prochaine sauvegarde automatique.

- Place nécessaire estimée : estimation de la taille nécessaire pour la sauvegarde. La taille réelle du fichier de sauvegarde pourra varier en fonction des paramètres (compression...) et des variations du fichier de données.

- Place disponible : place disponible sur le volume de sauvegarde.

Le bouton **Sauvegarder la base** permet de démarrer une sauvegarde immédiate de la base en utilisant les paramètres de sauvegarde courants (fichiers sauvegardés, emplacement des archives, options, etc.). Vous pouvez visualiser ces paramètres en cliquant sur le bouton **Préférences...**

Pendant une sauvegarde sur le serveur, les postes clients sont "bloqués" (mais pas déconnectés) et il n'est pas possible à de nouveaux clients de se connecter.

- **Historique des requêtes :** Cette zone indique la durée d'enregistrement de l'historique des requêtes, lorsqu'il est activé. Le fichier d'historique des requêtes stocke diverses informations concernant les requêtes reçues par le serveur (hors requêtes Web) : heure, numéro de process, utilisateur, taille de la requête, durée de traitement, etc., permettant d'analyser le fonctionnement du serveur. Ce fichier est nommé 4DRequestsLog\_N (N étant le numéro séquentiel du fichier) et est stocké dans le dossier Logs de la base. Une fois que le fichier atteint une taille de 10 Mo, il est refermé et un nouveau fichier est généré, avec un numéro séquentiel incrémenté.

Le bouton **Démarrer l'historique des requêtes** permet de générer un nouveau fichier et d'activer le mode d'enregistrement des requêtes. Ce mode pouvant dégrader sensiblement les performances du serveur, il est à réserver à la phase de mise au point de l'application. Une fois l'enregistrement des requêtes activé, le libellé du bouton devient **Arrêter l'historique des requêtes**, permettant de stopper l'enregistrement des requêtes à tout moment. A noter qu'une reprise de l'enregistrement des requêtes après un arrêt "écrase" le fichier précédent.

**Note :** Il est possible de démarrer et de stopper l'historique des requêtes par programmation via la commande **FIXER PARAMETRE BASE**.

Le bouton **Voir le compte rendu** (nommé **Télécharger le compte rendu** si l'opération a été effectuée depuis un client distant) permet d'ouvrir une fenêtre système affichant le fichier d'historique des requêtes.

## Page Serveur d'application

La Page Serveur d'application regroupe les informations relatives à la base de données publiée par 4D Server et permet de gérer cette publication :

The screenshot shows the 'Administration 4D Server' window for the 'Employés.4DB' database. The interface includes a top navigation bar with icons for 'Moniteur', 'Utilisateurs (0)', 'Process (11)', 'Maintenance', 'Serveur d'application', 'Serveur SQL', and 'Serveur HTTP'. The main content area displays the following information:

- État : Démarré
- Date de démarrage : 12/12/2008 à 17:14
- Durée de fonctionnement : 2 minutes
- Un bouton 'Refuser les nouvelles connexions' est visible.
- Configuration section:
  - Fichier structure : "Employés.4DB" dans le volume "C:"
  - Fichier de données : "Employés.4DD" dans le volume "C:"
  - Fichier journal : Employés.4DB.journal
  - Mode : Interprété
  - Démarré comme service : Non
  - Adresse IP d'écoute : 192.168.88.138
  - Port : 19813
  - SSL activé : Non
- Mémoire section:
  - Mémoire cache utilisée : 2,32 Mo
  - Mémoire cache totale : 100 Mo
- Connexions maximum section:
  - Application Server : 10

### Informations de statut

La partie supérieure de la page fournit des informations sur le statut courant du serveur d'application de 4D Server.

- Etat : Démarré ou Arrêté
- Date de démarrage : Date et heure de lancement de la base serveur. Cette date correspond à l'ouverture de la base par 4D Server.
- Durée de fonctionnement : Durée écoulée depuis la dernière ouverture de la base.

### Bouton Refuser / Accepter nouvelles connexions

Ce bouton fonctionne en bascule. Il permet de gérer l'accès de nouveaux postes clients à l'application serveur.

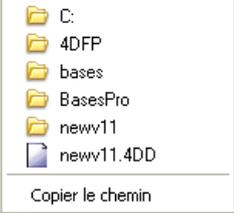
- Par défaut, lorsque la base est publiée :
  - Le libellé du bouton est "Refuser nouvelles connexions".
  - De nouveaux clients peuvent se connecter librement (dans les limites des connexions accordées par la licence).
  - Le nom de la base est publié dans la boîte de dialogue de connexion (si l'option "Publier le nom de la base au démarrage dans le dialogue de connexion" est cochée dans les Préférences).
- Si vous cliquez sur le bouton **Refuser nouvelles connexions** :
  - Le libellé du bouton devient "Accepter nouvelles connexions".
  - Plus aucun nouveau client ne peut alors se connecter.
  - Le nom de la base n'apparaît plus dans la boîte de dialogue de connexion.
  - Les clients déjà connectés ne sont pas déconnectés et peuvent continuer à travailler normalement.
  - Si vous cliquez sur le bouton **Accepter nouvelles connexions**, la base retourne dans l'état "par défaut".

Cette fonction permet par exemple à un administrateur, juste après avoir démarré le serveur, d'effectuer diverses opérations de maintenance (vérification, compactage...). S'il utilise une connexion cliente, il a la certitude d'être le seul à modifier les données. Il est également possible d'utiliser cette fonction en préparation d'une opération de maintenance nécessitant qu'aucun poste client ne soit connecté.

## Configuration

---

Cette zone fournit plusieurs informations sur la base de données 4D publiée par le serveur : nom et emplacement des fichiers de structure et de données et nom du fichier journal (fichier d'historique de la base). Vous pouvez cliquer sur le nom du fichier de structure ou de données afin de visualiser son chemin d'accès complet :

Fichier structure :	"neww11.4DB" dans le volume "C:"	▼
Fichier de données :	"neww11.4DD" dans le volume "C:"	▼
Fichier journal :		
arrêté comme service :		
adresse IP d'écoute :		
Port :		
SSL activé :	non	

Le champ "Mode" indique le mode d'exécution courant de la base : compilé ou interprété.

La partie inférieure de la zone indique les paramètres de configuration du serveur (démarré comme service, port et adresse IP) et l'activation du SSL pour les connexions client-serveur (ne concerne pas les connexions SQL ni Web).

## Mémoire

---

Cette zone indique la **mémoire cache totale** (paramètre défini dans les préférences de la base) et la **mémoire cache utilisée** (allocation dynamique par 4D Server en fonction des besoins).

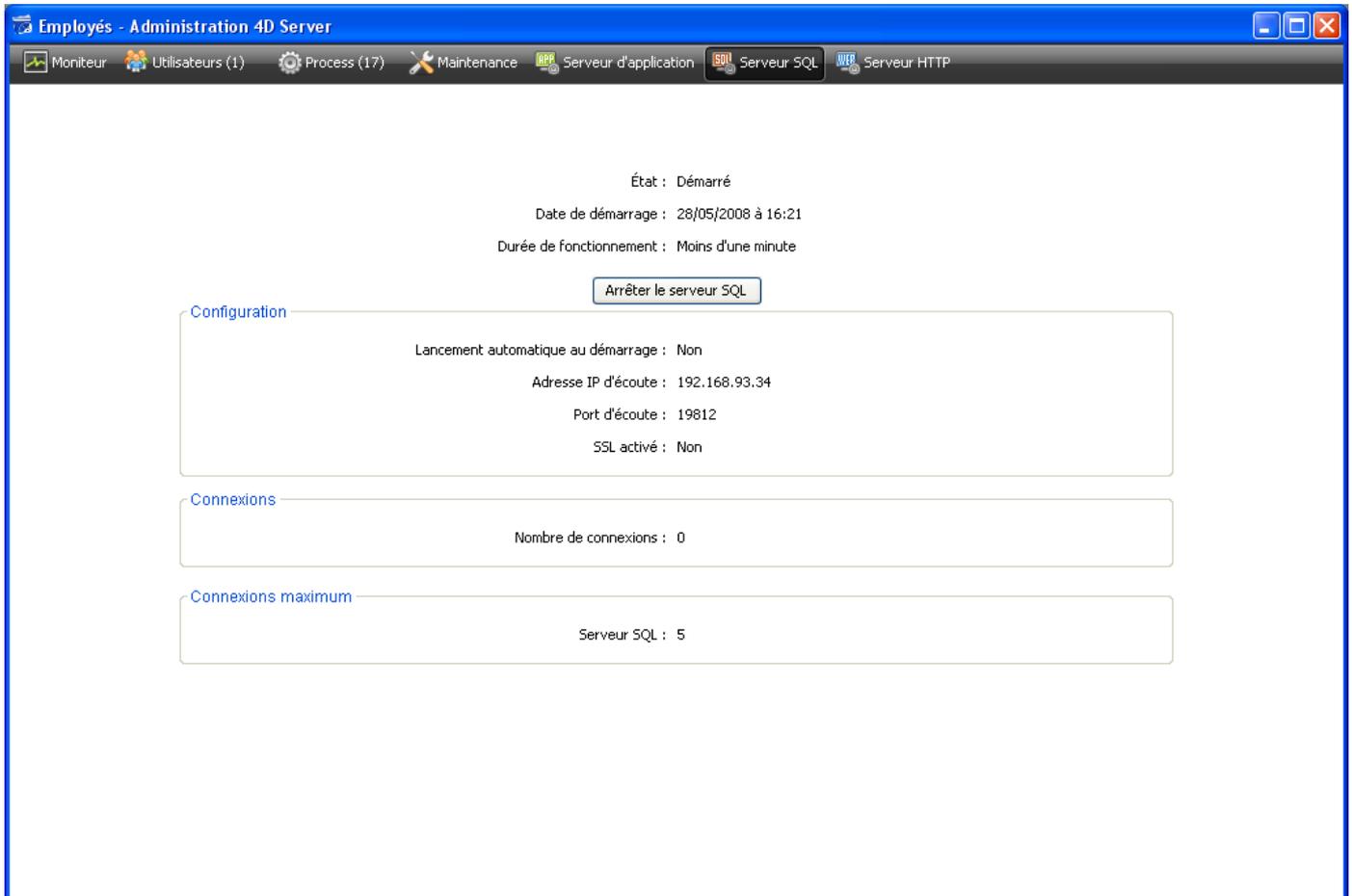
## Connexions maximum

---

Nombre maximum de connexions clientes simultanées autorisées. Cette valeur dépend de la licence installée sur le poste serveur.

## Page Serveur SQL

La Page Serveur SQL regroupe les informations relatives au serveur SQL intégré de 4D Server. Elle comporte également un bouton permettant de contrôler l'activation du serveur :



### Informations de statut

La partie supérieure de la page fournit des informations sur le statut courant du serveur SQL de 4D Server.

- Etat : Démarré ou Arrêté
- Date de démarrage : Date et heure du dernier lancement du serveur SQL. Cette valeur peut différer de celle du serveur d'application, si le lancement du serveur SQL ne s'effectue pas "au démarrage".
- Durée de fonctionnement : Délai écoulé depuis le dernier démarrage du serveur SQL.

### Bouton Démarrer / Arrêter le serveur SQL

Ce bouton fonctionne en bascule. Il permet de contrôler l'activation du serveur SQL de 4D Server.

- Lorsque l'état du serveur SQL est "Démarré", le bouton est libellé **Arrêter le serveur SQL**. Si vous cliquez sur ce bouton, le serveur SQL de 4D Server est immédiatement stoppé, il ne répond plus aux requêtes SQL externes reçues sur le port TCP désigné.
- Lorsque l'état du serveur SQL est "Arrêté", le bouton est libellé **Démarrer le serveur SQL**. Si vous cliquez sur ce bouton, le serveur SQL de 4D Server est immédiatement démarré, il répond aux requêtes SQL externes reçues sur le port TCP désigné. A noter que vous devez disposer d'une licence adéquate pour pouvoir exploiter le serveur SQL de 4D.

**Note :** Le serveur SQL peut également être lancé automatiquement au démarrage de l'application (option des Préférences) ou par programmation.

### Configuration

Cette zone fournit plusieurs informations sur les paramètres de configuration du serveur SQL : lancement automatique au démarrage, adresse IP d'écoute, port TCP (19812 par défaut) et activation du SSL pour les connexions SQL (ne concerne pas les connexions 4D ni Web).

Ces paramètres peuvent être modifiés via les Préférences de 4D.

## **Connexions**

---

Nombre de connexions SQL actuellement ouvertes sur 4D Server.

## **Connexions maximum**

---

Nombre maximum de connexions SQL simultanées autorisées. Cette valeur dépend de la licence installée sur le poste serveur.

## Page Serveur HTTP

La Page Serveur HTTP regroupe les informations relatives au fonctionnement du serveur Web et du serveur SOAP de 4D Server. Le serveur Web permet de publier du contenu Web tel que des pages HTML ou des images à destination de navigateurs Web. Le serveur SOAP gère la publication de Web Services. Ces deux serveurs s'appuient sur le serveur HTTP interne de 4D Server. La page comporte également des boutons permettant de contrôler l'activation des serveurs :

The screenshot shows the 'Administration 4D Server' window with the 'Serveur HTTP' tab selected. The main content area displays the following information:

- État : Démarré
- Date de démarrage : 28/05/2008 à 10:16
- Durée de fonctionnement : 6 heures 17 minutes
- Arrêter le serveur HTTP (button)
- Nombre de hits HTTP : 2

Below this, there are three sections:

- Informations Web**: Requetes Web : Acceptées; Connexion maximum : Illimitées
- Informations SOAP**: Requetes SOAP : Acceptées; Connexions maximum : Illimitées; Refuser les requêtes SOAP (button)
- Configuration serveur HTTP**: Lancement automatique au démarrage : Oui; Process serveur HTTP (en cours/total) : 0/2; Mémoire cache : 0 pages (%); Adresse IP d'écoute : 192.168.93.34; Port HTTP : 80; SSL activé : Oui; Port HTTPS : 443; Fichier journal : -; Format journal : -; Prochain backup du journal : 00/00/00 à 00:00

### Informations de statut

La partie supérieure de la page fournit des informations sur le statut courant du serveur HTTP de 4D Server.

- Etat : Démarré ou Arrêté
- Date de démarrage : Date et heure du dernier lancement du serveur HTTP. Cette valeur peut différer de celle du serveur d'application, si le lancement du serveur HTTP ne s'effectue pas "au démarrage".
- Durée de fonctionnement : Délai écoulé depuis le dernier démarrage du serveur HTTP.
- Nombre de hits HTTP : nombre de hits HTTP (bas niveau) reçus par le serveur HTTP depuis son démarrage.

### Bouton Démarrer / Arrêter le serveur HTTP

Ce bouton fonctionne en bascule. Il permet de contrôler l'activation du serveur HTTP de 4D Server.

- Lorsque l'état du serveur HTTP est "Démarré", le bouton est libellé **Arrêter le serveur HTTP**. Si vous cliquez sur ce bouton, le serveur HTTP de 4D Server est immédiatement stoppé, le serveur Web et le serveur SOAP n'acceptent plus aucune requête.
- Lorsque l'état du serveur HTTP est "Arrêté", le bouton est libellé **Démarrer le serveur HTTP**. Si vous cliquez sur ce bouton, le serveur HTTP de 4D Server est immédiatement démarré : les requêtes Web et les requêtes SOAP sont acceptées (à noter qu'il est possible de stopper séparément le serveur SOAP, cf. ci-dessous paragraphe "Informations SOAP").

### Notes :

- Vous devez disposer d'une licence appropriée pour pouvoir démarrer le serveur HTTP.
- Le serveur HTTP peut également être lancé automatiquement au démarrage de l'application (option des Préférences) ou par programmation.

## Informations Web

---

Cette zone fournit des informations spécifiques relatives au serveur Web de 4D Server.

- **Requêtes Web** : Acceptées ou Refusées. Cette information indique si le serveur Web est actif. Le serveur Web étant directement relié au serveur HTTP, les requêtes Web sont acceptées lorsque le serveur HTTP est démarré et refusées lorsqu'il est stoppé.
- **Connexions maximum** : Nombre maximum de connexions Web autorisées. Cette valeur dépend de la licence installée sur le poste serveur.

## Informations SOAP

---

Cette zone fournit des informations spécifiques relatives au serveur SOAP de 4D Server et contient un bouton de contrôle.

- **Requêtes SOAP** : Acceptées ou Refusées. Cette information indique si le serveur SOAP est actif. Pour que les requêtes SOAP soient acceptées, le serveur HTTP doit être démarré et le serveur SOAP doit explicitement accepter les requêtes (cf. bouton **Accepter/Refuser**).
- **Connexions maximum** : Nombre maximum de connexions SOAP autorisées. Cette valeur dépend de la licence installée sur le poste serveur.
- **Bouton **Accepter/Refuser les requêtes SOAP**** : Ce bouton fonctionne en bascule. Il permet de contrôler l'activation du serveur SOAP de 4D Server. Ce bouton modifie la valeur de l'option **Autoriser requêtes SOAP** dans la page "Web Services/SOAP" des préférences (et inversement).  
Si vous cliquez sur le bouton **Accepter les requêtes SOAP** et que le serveur HTTP est arrêté, 4D le démarre automatiquement.

## Configuration serveur HTTP

---

Cette zone fournit plusieurs informations sur les paramètres de configuration et le fonctionnement du serveur HTTP :

- **Lancement automatique au démarrage** : paramètre défini via les Préférences de 4D Server.
- **Process serveur HTTP (en cours/total)** : nombre de process HTTP créés sur le serveur (nombre courant de process / cumul de tous les process créés)
- **Mémoire cache (utilisée/totale)** : taille de la mémoire cache du serveur HTTP, lorsqu'elle est activée (taille réellement occupée par le cache / taille maximale théorique allouée au cache dans les Préférences).
- **Adresse IP d'écoute, Port TCP (80 par défaut), SSL activé pour les connexions HTTP (ne concerne pas les connexions 4D ni SQL) et Port HTTPS utilisé** : paramètres de configuration courants du serveur HTTP, définis dans la page "Web/Configuration" des Préférences (cf. section **Paramétrages du serveur Web** dans le manuel Langage de 4D) .
- **Informations sur le fichier journal** : emplacement, format et date de la prochaine sauvegarde automatique du journal du serveur HTTP (fichier logweb.txt).

## **Méthodes base 4D Server**

 Méthode base Sur démarrage serveur

 Méthode base Sur arrêt serveur

 Méthode base Sur ouverture connexion serveur

 Méthode base Sur fermeture connexion serveur

## Méthode base Sur démarrage serveur

---

La méthode base Sur démarrage serveur est appelée une fois sur le poste serveur lorsque vous ouvrez une base avec 4D Server. La méthode base Sur démarrage serveur n'est jamais exécutée dans un environnement autre que 4D Server.

La méthode base Sur démarrage serveur est l'emplacement idéal pour :

- Initialiser les variables interprocess utilisées pendant toute la session 4D Server.
- Démarrer automatiquement des **Procédures stockées** à l'ouverture de la base.
- Charger des préférences ou des paramétrages sauvegardé(e)s dans ce but lors de la précédente session 4D Server.
- Empêcher l'ouverture de la base si une condition n'est pas remplie (par exemple, absence de ressources système) par un appel explicite à **QUITTER 4D**.
- Placer toute action devant être automatiquement effectuée à chaque ouverture de la base.

Si vous souhaitez exécuter du code automatiquement sur un poste client lorsqu'un 4D distant se connecte au serveur, utilisez la **Méthode base Sur ouverture**.

**Note** : La méthode base Sur démarrage serveur est exécutée de façon atomique, ce qui signifie qu'aucun 4D distant ne peut se connecter tant que l'exécution de la méthode n'est pas terminée.

## Méthode base Sur arrêt serveur

---

La **Méthode base Sur arrêt serveur** est appelée une fois sur le poste serveur lorsque la base courante est refermée sur 4D Server. La **Méthode base Sur arrêt serveur** n'est appelée dans aucun environnement 4D autre que 4D Server.

Pour refermer la base courante sur le serveur, vous pouvez sélectionner la commande de menu **Fermer la base...** sur le serveur. Vous pouvez également choisir la commande **Quitter** ou appeler la commande **QUITTER 4D** au sein d'une procédure stockée exécutée sur le serveur.

Lorsque le processus de fermeture de la base a été engagé, 4D effectue les actions suivantes :

- S'il n'y a pas de **Méthode base Sur arrêt serveur**, 4D Server tue un à un chaque process en cours d'exécution, sans distinction.
- S'il existe une **Méthode base Sur arrêt serveur**, 4D Server exécute cette méthode dans un nouveau process local. Vous pouvez donc utiliser cette méthode base pour informer les autres process, via la communication interprocess, qu'ils doivent mettre fin à leur exécution. Notez que dans tous les cas 4D Server quittera en fin de compte —la **Méthode base Sur arrêt serveur** peut effectuer toutes les opérations de nettoyage et de fermeture que vous voulez, mais elle ne peut refuser de quitter, et finira par se terminer.

La **Méthode base Sur arrêt serveur** est l'emplacement idéal pour :

- Stopper les procédures stockées lancées automatiquement à l'ouverture de la base.
- Sauvegarder (localement, sur disque) des préférences ou des paramètres à réutiliser au début de la session suivante dans la **Méthode base Sur démarrage serveur**.
- Effectuer toute autre action que vous souhaitez déclencher automatiquement à chaque fois que vous quittez la base.

**Important :** Si vous utilisez la **Méthode base Sur arrêt serveur** pour refermer des procédures stockées, tenez compte du fait que le serveur quitte dès la fin de l'exécution de la **Méthode base Sur arrêt serveur** (et non des procédures stockées). Si des procédures stockées tournent encore à cet instant, elles sont purement et simplement tuées.

Par conséquent, si vous voulez être certain que les procédures stockées se terminent avant que le serveur ne les tue, il faut que la **Méthode base Sur arrêt serveur** leur signale qu'elles doivent mettre fin à leur exécution (par le test d'une variable interprocess, par exemple) mais aussi qu'elle leur laisse le temps de se refermer (boucle de n secondes ou test d'une autre variable interprocess).

Si vous voulez que du code soit exécuté automatiquement sur un poste client lorsqu'un 4D distant met un terme à sa connexion au serveur, utilisez la .

## Méthode base Sur ouverture connexion serveur

### Quand la Méthode base Sur ouverture connexion serveur est-elle appelée ?

---

La **Méthode base Sur ouverture connexion serveur** est appelée une fois sur la machine serveur chaque fois qu'un poste 4D distant démarre un process de connexion. La **Méthode base Sur ouverture connexion serveur** n'est appelée que par 4D Server, à l'exclusion de tout autre environnement 4D.

La **Méthode base Sur ouverture connexion serveur** est appelée à chaque fois que :

- un 4D distant se connecte (démarrage du process principal)
- un 4D distant ouvre l'environnement Développement (démarrage du process de développement)
- un 4D distant démarre un process non local (avec la commande **Nouveau process**, une commande de menu ou la boîte de dialogue "Exécuter une méthode").

Dans chaque cas, avec 4D Client, trois process démarrent. Un sur la machine client, les deux autres sur la machine serveur. Sur la machine client, le process exécute le code et envoie les requêtes à 4D Server. Sur la machine serveur, le **process 4D Client** gère l'environnement de base de données du process client (c.-à-d. les sélections courantes et le verrouillage des enregistrements pour le process utilisateur) et répond aux requêtes envoyées par le process exécuté sur la machine cliente. Le **process base 4D Client** est chargé de contrôler le process 4D Client correspondant.

**Important :** Les connexions Web et les connexions SQL ne provoquent **pas** l'exécution de la **Méthode base Sur ouverture connexion serveur**. Lorsqu'un navigateur Web se connecte à 4D Server, la **Méthode base Sur authentification Web** (si elle existe) et/ou la **Méthode base Sur connexion Web** sont appelées. Lorsque 4D Server reçoit une requête SQL, la **Méthode base Sur authentification SQL** (si elle existe) est appelée. Pour plus d'informations, reportez-vous à la description de ces méthodes base dans le manuel Langage de 4D.

**Important :** Lors du démarrage d'une procédure stockée, la **Méthode base Sur ouverture connexion serveur** n'est **pas** appelée. Les procédures stockées sont des process serveur et non des process 4D Client. Elles exécutent du code sur la machine serveur mais ne répondent pas aux requêtes échangées par 4D Client (ou d'autres clients) et 4D Server.

### Comment la Méthode base Sur ouverture connexion serveur est-elle appelée ?

---

La **Méthode base Sur ouverture connexion serveur** est exécutée sur le poste serveur dans le process 4D Client qui a provoqué l'appel de la méthode.

Si, par exemple, un 4D distant se connecte à une base 4D Server en mode interprété, il démarre le process utilisateur, le process de développement ainsi que (par défaut) le process d'inscription du client. La **Méthode base Sur ouverture connexion serveur** est donc exécutée trois fois de suite. La première fois dans le process principal, la deuxième fois dans le process d'inscription du client et la troisième fois dans le process de développement. Si les trois process sont respectivement les 6e, 7e et 8e process démarrés sur la machine serveur, et si vous appelez **Numero du process courant** dans la **Méthode base Sur ouverture connexion serveur**, le premier **Numero du process courant** retourne 6, le deuxième 7 et le troisième 8.

Notez que la **Méthode base Sur ouverture connexion serveur** s'exécute sur le poste serveur, à l'intérieur du process 4D Client sur le serveur. Elle ignore tout du process exécuté sur le client. En outre, au moment où la méthode est appelée, le process 4D Client n'est pas encore nommé (**INFORMATIONS PROCESS** ne retournera pas, à ce moment, le nom du process 4D Client).

La **Méthode base Sur ouverture connexion serveur** n'a pas accès à la table des variables process du process exécuté sur le client. Cette table réside sur le poste client, pas sur le serveur.

Lorsque la **Méthode base Sur ouverture connexion serveur** accède à une variable process, elle travaille avec une table de variables process particulière, créée dynamiquement pour le process 4D Client.

4D Server passe trois paramètres de type Entier long à la **Méthode base Sur ouverture connexion serveur** et attend un résultat Entier long. La méthode doit donc être explicitement déclarée avec trois paramètres Entier long ainsi qu'un retour de fonction Entier long :

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

Si vous ne retournez pas de valeur dans \$0 et donc laissez la variable indéfinie ou initialisée à zéro, 4D Server estime que la méthode base accepte la connexion. Si vous n'acceptez pas la connexion, retournez une valeur non nulle dans \$0.

Le tableau ci-dessous détaille les informations fournies par les trois paramètres passés à la méthode base :

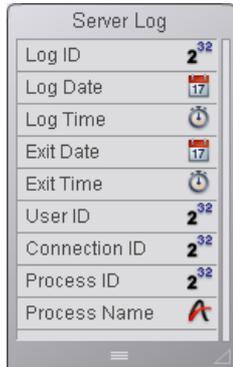
Paramètre	Description
\$1	Numéro d'utilisateur utilisé en interne par 4D Server pour identifier les utilisateurs
\$2	Numéro de connexion utilisé en interne par 4D Server pour identifier une connexion
\$3	Obsolète : Retourne toujours 0 (mais doit être déclaré)

Ces numéros de référence ne sont pas directement utilisables en tant « sources d'information » à passer, par exemple, comme paramètres à une commande 4D. Ils vous fournissent un moyen d'identifier de manière unique un process 4D Client entre la **Méthode base Sur ouverture connexion serveur** et la **Méthode base Sur fermeture connexion serveur**. La combinaison de ces valeurs est unique à tout moment d'une session 4D Server. Si vous stockez cette information dans une table ou un tableau interprocess, les deux méthodes base peuvent échanger des informations. Dans l'exemple présenté à la fin de cette section, les deux méthodes base utilisent cette information pour stocker l'heure et la date du début et de la fin d'une connexion dans le même

enregistrement d'une table.

## Exemple 1

L'exemple suivant montre comment maintenir un historique des connexions à la base de données en utilisant la **Méthode base Sur ouverture connexion serveur** et la **Méthode base Sur fermeture connexion serveur**. La table [Server Log] (ci-dessous) sert à garder la trace des process de connexion :



Field Name	Field Type	Field Length
Log ID	Integer	32
Log Date	Date	
Log Time	Time	
Exit Date	Date	
Exit Time	Time	
User ID	Integer	32
Connection ID	Integer	32
Process ID	Integer	32
Process Name	Text	

L'information stockée dans cette table est gérée par la **Méthode base Sur ouverture connexion serveur** et la **Méthode base Sur fermeture connexion serveur** listées ci-dessous :

```
\ Methode base Sur ouverture connexion serveur
C_ENTIER LONG($0;$1;$2;$3)
\ Créer un enregistrement [Server Log]
CREER ENREGISTREMENT([Server Log])
[Server Log]Log ID:=Numerotation automatique([Server Log])
\ Enregistrer l'historique Date et Heure
[Server Log]Log Date:=Date du jour
[Server Log]Log Time:=Heure courante
\ Enregistrer l'information sur la connexion
[Server Log]User ID:=$1
[Server Log]Connection ID:=$2
STOCKER ENREGISTREMENT([Server Log])
\ Ne retourne pas d'erreur, pour continuer la connexion
$0:=0
```

```
\ Methode base Sur fermeture connexion serveur
C_ENTIER LONG($1;$2;$3)
\ Chercher l'enregistrement [Server Log]
CHERCHER([Server Log];[Server Log]User ID=$1;)
CHERCHER([Server Log]; & ;[Server Log]Connection ID=$2;)
CHERCHER([Server Log]; & ;[Server Log]Process ID=0)
\ Enregistrer date et heure de déconnexion
[Server Log]Exit Date:=Date du jour
[Server Log]Exit Time:=Heure courante
\ Enregistrer informations process
[Server Log]Process ID:=Numero du process courant
INFORMATIONS PROCESS([Server Log]Process ID;$vsProcName;$vlProcState;$vlProcTime)
[Server Log]Process Name:=$vsProcName
STOCKER ENREGISTREMENT([Server Log])
```

Voici quelques entrées dans [Server Log] montrant plusieurs connexions distantes :

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

## Exemple 2

L'exemple suivant interdit toute nouvelle connexion entre 2 et 4 heures du matin.

```

\ Méthode base Sur ouverture connexion serveur
C_ENTIER LONG ($0;$1;$2;$3)

Si ((?02:00:00?<=Heure courante) & (Heure courante<?04:00:00?))
    $0:=22000
Sinon
    $0:=0
Fin de si

```

## Méthode base Sur fermeture connexion serveur

---

La **Méthode base Sur fermeture connexion serveur** est exécutée sur le poste serveur à chaque fois qu'un process 4D Client est refermé.

Comme pour la **Méthode base Sur ouverture connexion serveur**, 4D Server passe trois paramètres de type Entier long à la **Méthode base Sur fermeture connexion serveur**. En revanche, 4D Server n'attend pas de résultat en retour.

Par conséquent, la méthode doit contenir la déclaration explicite des trois paramètres Entier long :

```
C_ENTIER LONG ($1; $2; $3)
```

Le tableau suivant détaille les informations fournies par les trois paramètres passés à la méthode base :

Paramètre	Description
\$1	Numéro d'utilisateur utilisé en interne par 4D Server pour identifier les utilisateurs
\$2	Numéro de connexion utilisé en interne par 4D Server pour identifier une connexion
\$3	Obsolète : Retourne toujours 0 mais doit être déclaré

La **Méthode base Sur fermeture connexion serveur** est le pendant inverse de la **Méthode base Sur ouverture connexion serveur**. Pour plus d'informations sur ce point, ainsi que pour la description des **process 4D Client**, reportez-vous à la description de cette méthode base.

### Exemple

---

Reportez-vous au premier exemple de la **Méthode base Sur ouverture connexion serveur**.

## Utilisation d'un 4D distant

-  Connexion à une base 4D Server
-  Administration à distance
-  Compilation à distance

## Connexion à une base 4D Server

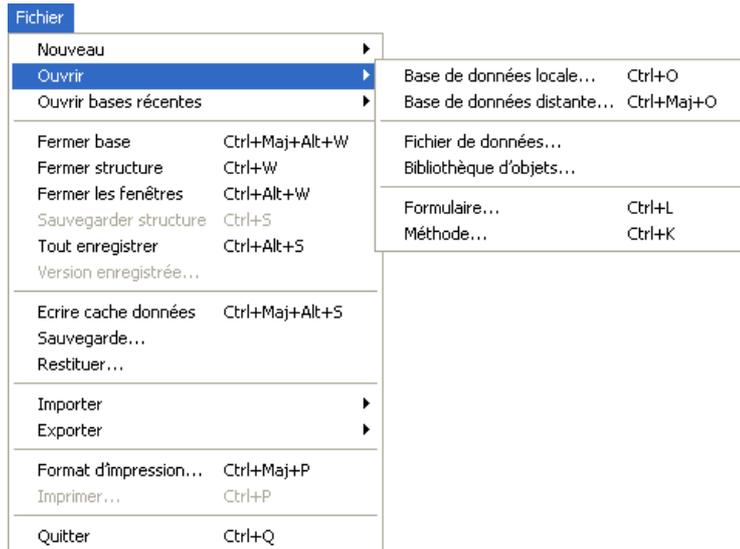
Pour vous connecter à une base 4D Server via un 4D distant, vous disposez de trois possibilités :

- Utiliser la boîte de dialogue de connexion
- Utiliser le menu **Ouvrir bases récentes**
- Utiliser un fichier 4DLink de raccourci contenant les paramètres d'accès à la base.

### Utiliser la boîte de dialogue de connexion

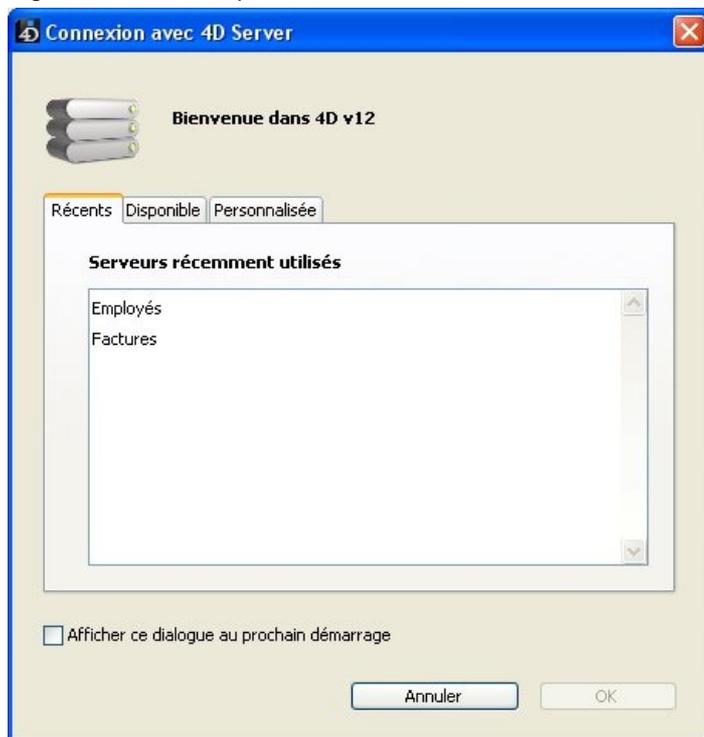
Pour afficher la boîte de dialogue de connexion à 4D Server, lancez l'application 4D.

La commande **Ouvrir** du menu **Fichier** (ou le bouton correspondant dans la barre d'outils de 4D) vous permet de sélectionner le mode d'ouverture de la base 4D :



Choisissez la commande **Ouvrir>Base de données distante...**

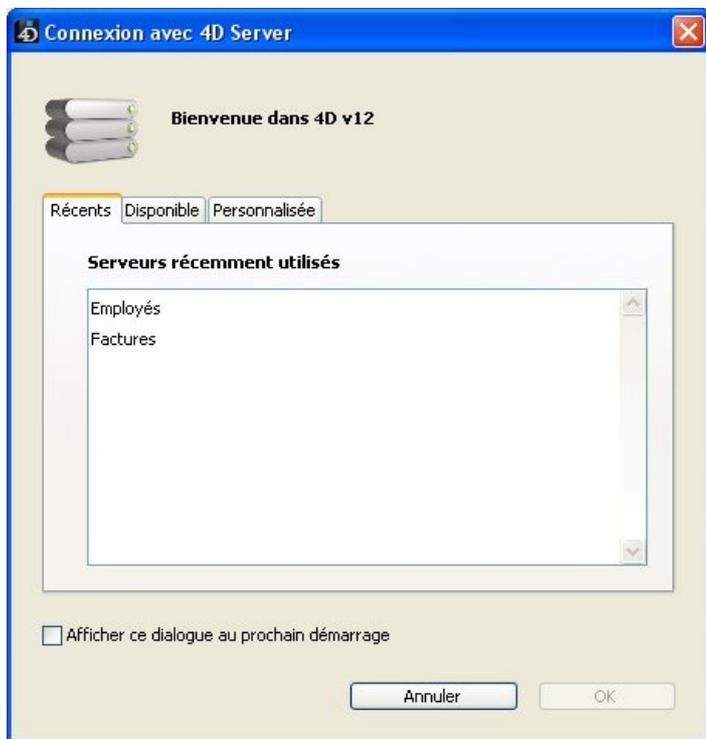
La boîte de dialogue de connexion à 4D Server apparaît. Cette boîte de dialogue comporte trois pages, accessibles via des onglets : Récents, Disponible et Personnalisée :



Si vous cochez l'option **Afficher ce dialogue au prochain démarrage**, cette boîte de dialogue sera automatiquement affichée au démarrage de l'application 4D.

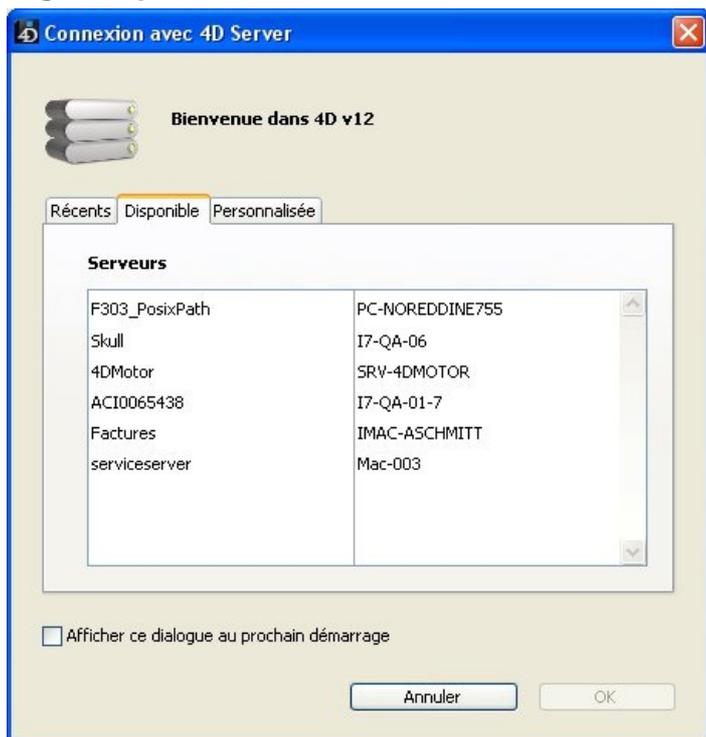
**Note :** Vous pouvez également afficher cette boîte de dialogue en cliquant sur le lien **Se connecter à 4D Server** dans le dialogue de bienvenue de 4D.

### Page "Récents"



Cette page mémorise la liste des bases 4D Server auxquelles vous vous êtes récemment connecté. La liste est triée par ordre alphabétique. Pour vous connecter à un serveur depuis cette liste, double-cliquez sur son nom ou sélectionnez-le et cliquez sur le bouton **OK**.

### Page “Disponible”



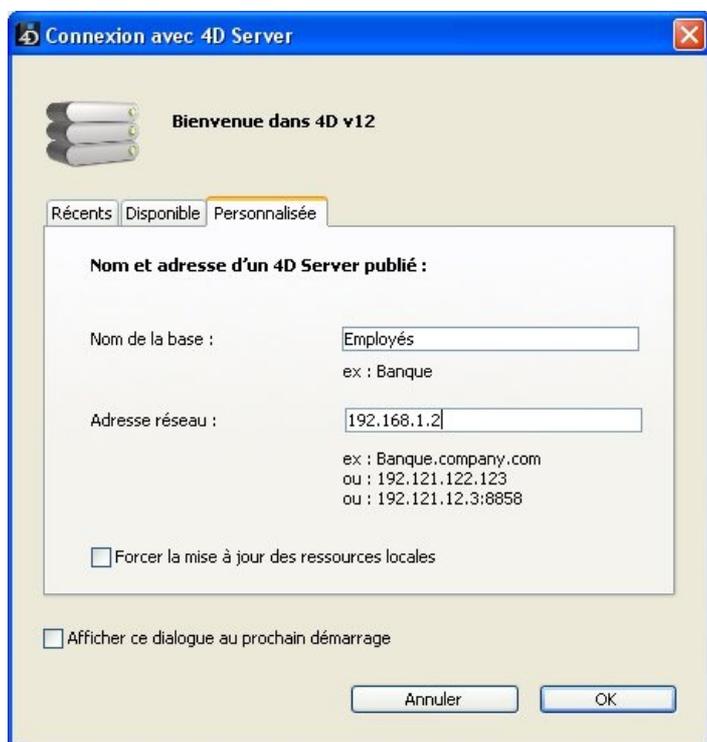
4D Server est doté d'un système de publication TCP/IP intégré, qui publie par défaut le nom des bases 4D Server disponibles sur le réseau. Ces noms apparaissent dans la page **Disponible** de la boîte de dialogue de connexion.

La liste est triée par ordre d'apparition et est mise à jour dynamiquement. Pour vous connecter à un serveur depuis cette liste, double-cliquez sur son nom ou sélectionnez-le et cliquez sur le bouton **OK**.

### Notes :

- Un accent circonflexe (^) apparaît devant le nom des bases publiées en mode sécurisé. Pour plus d'informations, reportez-vous à la section **Crypter les connexions client/serveur**.
- Il est possible d'interdire la publication dynamique du nom de la base sur le réseau (cf. section **Configuration IP**). Dans ce cas, la connexion ne peut être effectuée que manuellement à l'aide de la page “Personnalisée”.

### Page “Personnalisée”



La page **Personnalisée** permet de désigner, via l'adresse IP du serveur, une base publiée sur le réseau et de lui associer un nom personnalisé.

En effet, le système de publication TCP/IP de 4D Server peut être personnalisé afin que le nom de la base ne soit pas automatiquement publié sur le réseau (cf. section **Configuration IP**). Dans ce cas, ce nom n'apparaît pas dans la page "Disponible". Il est nécessaire dans ce cas de connaître l'adresse IP du serveur et de la saisir manuellement.

- **Nom de la base** : permet de définir le nom de la base 4D Server. Ce nom sera utilisé dans la page **Récents** pour désigner la base.
- **Adresse réseau** : permet de saisir l'adresse IP de la machine sur laquelle est lancé 4D Server. Si deux serveurs sont exécutés simultanément sur le même poste, l'adresse IP doit être suivie de deux-points et du numéro de port, par exemple : 192.168.92.104:19814

Par défaut, le port de publication d'un 4D Server est le 19813. Ce numéro peut être modifié dans les Préférences de l'application (cf. section **Options réseau et Client-serveur**).

**Note** : Si une base était sélectionnée dans les pages **Récents** ou **Disponible** au moment où vous avez cliqué sur l'onglet **Personnalisée**, les deux champs affichent les informations correspondantes.

Une fois que cette page désigne un serveur, cliquer sur le bouton **OK** vous permet de vous connecter au serveur. Par la suite, le nom du serveur sera listé dans la page **Récents**.

**Note** : Si la base est publiée en mode sécurisé, il faut impérativement faire débiter son nom par un accent circonflexe (^), sinon la connexion sera refusée. Pour plus d'informations, reportez-vous à la section **Crypter les connexions client/serveur**.

### Forcer la mise à jour des ressources locales

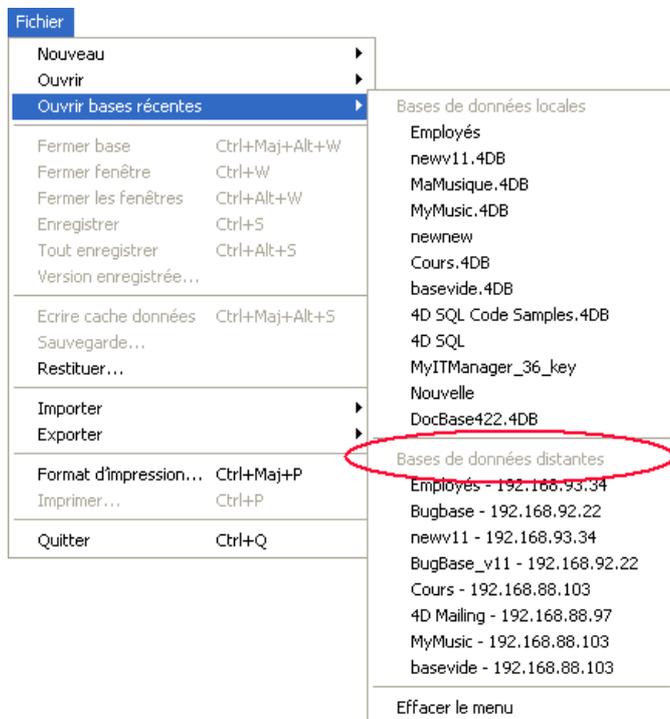
Cette option provoque la mise à jour systématique des ressources locales sur le poste client au moment de la connexion. Les ressources locales sont les informations structurelles relatives à la base, stockées sur chaque poste client.

La mise à jour des ressources locales est en principe automatique sur le poste distant au moment de sa connexion, lorsque la structure de la base a été modifiée depuis la dernière connexion. Dans la plupart des cas, cette option est inutile. Toutefois, dans certains cas spécifiques, il peut être nécessaire de forcer la mise à jour.

### Utiliser le menu Ouvrir bases récentes

La commande de menu **Ouvrir base récentes** vous permet de vous connecter directement à une base 4D Server à laquelle vous vous êtes déjà connecté.

Cette commande est située dans le menu **Fichier** de 4D. Si vous utilisez l'application 4D pour vous ouvrir des bases locales et pour vous connecter à des bases distantes, le menu liste les deux types de bases. Les bases distantes sont situées en bas du menu :



L'adresse IP du serveur est indiquée derrière le nom de la base.  
La commande **Effacer le menu** permet de réinitialiser le menu.

## Utiliser un fichier 4DLink

Vous pouvez générer des fichiers d'accès aux bases contenant des paramètres destinés à automatiser et simplifier l'ouverture ou la connexion à des bases 4D. Typiquement, un fichier d'accès peut enregistrer l'adresse d'une base 4D Server distante ainsi que les identifiants de connexion, évitant ainsi à l'utilisateur plusieurs opérations.  
Les fichiers d'accès peuvent également être utilisés pour l'ouverture de bases locales.

### Création des fichiers

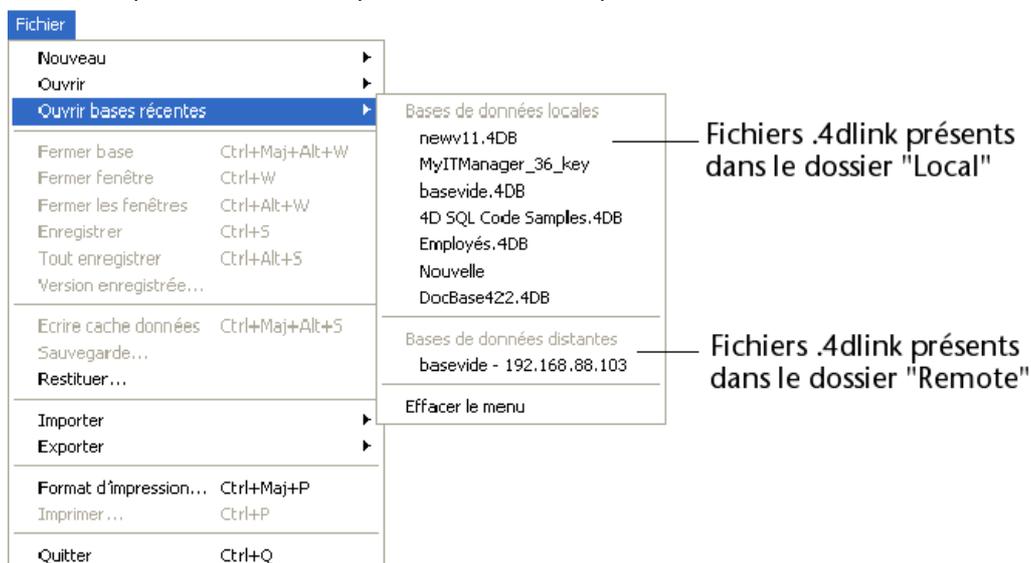
Les fichiers d'accès aux bases de 4D sont des fichiers XML. Ils comportent l'extension ".4DLink". 4D génère et utilise ce type de fichier pour construire le sous-menu des "bases récentes" : un fichier .4DLink est automatiquement généré par 4D à la première ouverture d'une base locale ou à la première connexion à un serveur.

Les fichiers .4DLink automatiquement créés par 4D sont placés dans le dossier des préférences locales de l'utilisateur. Dans ce dossier, deux répertoires sont créés : Local et Remote. Le dossier Local contient les fichiers ".4DLink" permettant de se connecter aux bases locales, le dossier Remote contient les fichiers ".4DLink" permettant de se connecter aux bases distantes.

Les dossiers de préférences locales sont situés à l'emplacement suivant :

- Windows Vista/Windows 7 : C:\Users\NomUtilisateur\AppData\Roaming\4D\Favorites v12\
- Windows XP : C:\Documents and Settings\NomUtilisateur\Application Data\4D\Favorites v12\
- Mac OS : Users/NomUtilisateur/Library/Preferences/4D/Favorites v12/

Les fichiers présents dans ces répertoires sont affichés par 4D dans le sous-menu **Ouvrir bases récentes** du menu **Fichier** :



Les fichiers ".4DLink" peuvent aussi être créés avec tout éditeur XML et contenir diverses informations personnalisées telles que

les identifiants de connexion (nom d'utilisateur et mot de passe) ou le mode d'ouverture de la base.

4D fournit une DTD décrivant les clés XML utilisables pour construire un fichier ".4DLink". Cette DTD est nommée *database\_link.dtd* et est située dans le sous-dossier *\Ressources\DTD\* de l'application 4D.

### Utilisation des fichiers

Un fichier d'accès .4DLink permet de lancer l'application 4D et d'ouvrir la base 4D Server cible. Il peut être utilisé :

- via un double-clic ou un glisser-déposer sur l'application 4D,
- via le sous-menu **Ouvrir bases récentes** (fichier placé dans le dossier de préférences local).  
Un même fichier .4DLink de type "base distante" (Remote) peut être copié et utilisé sur différentes machines.

**Note :** Il est également possible de sélectionner un fichier 4DLink dans la boîte de dialogue d'ouverture de 4D et de 4D Server (ouverture de bases locales uniquement).

## Administration à distance

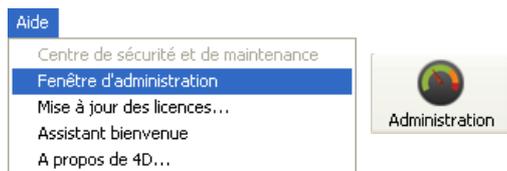
Vous pouvez administrer le poste 4D Server depuis un 4D distant (poste client). Le principe consiste à ouvrir la fenêtre d'administration de 4D Server (cf. section [Page Moniteur](#)) sur le poste client.

### Ouvrir la fenêtre d'administration sur un poste 4D distant

Pour ouvrir une fenêtre d'administration du serveur depuis un poste client, vous devez être connecté à la base distante en tant que Super\_Utilisateur ou Administrateur. Dans le cas contraire, si vous tentez d'ouvrir la fenêtre d'administration, une erreur de privilège (-9991) est générée.

L'accès à la fenêtre s'effectue par l'un des moyens suivants :

- Sélection de la commande **Fenêtre d'administration** dans le menu **Aide** ou cliquer sur le bouton correspondant dans la barre d'outils de 4D :



- Exécution de la commande **OUVRIR FENETRE ADMINISTRATION**.

Une fenêtre d'administration du serveur s'affiche alors sur le poste client :



### Spécificités de l'administration via un poste client

Un poste client affichant la fenêtre d'administration du serveur accède à toutes les informations disponibles et peut agir sur les process et le démarrage des serveurs. Il existe cependant des restrictions et des particularités de fonctionnement lorsque la fenêtre d'administration du serveur est affichée sur un poste client :

- Dans la [Page Process](#), il n'est pas possible de tracer un process utilisateur (la fenêtre du débogueur apparaîtrait sur le poste serveur).
- Dans la [Page Maintenance](#), il est possible d'exécuter des actions provoquant la déconnexion de tous les clients et le

redémarrage du serveur (compactage et redémarrage). Dans ce cas, le poste client ayant demandé l'opération est automatiquement reconnecté à l'issue du redémarrage.

- Dans la **Page Maintenance**, les boutons **Voir le compte rendu** sont renommés **Télécharger le compte rendu** après exécution d'une opération de maintenance. Ces fichiers sont téléchargés dans le dossier de la base sur le poste client avant d'être affichés.

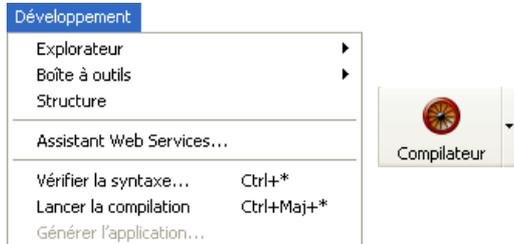
## Compilation à distance

---

Il est possible de compiler une application 4D à partir d'une connexion distante. Autrement dit, il est possible de compiler la base depuis un poste client 4D. Dans les versions de 4D Server antérieures à la v11 SQL, la compilation pouvait uniquement être effectuée depuis l'application monoposte.

**Note :** Il n'est pas possible de construire une application 4D personnalisée (monoposte ou client/serveur) à partir d'une connexion distante. Le Générateur d'applications n'est pas accessible dans cet environnement (la ligne de menu est grisée).

Côté client, l'interface et les principes de compilation sont identiques à ceux des versions monopostes. La compilation peut être déclenchée soit depuis le menu **Développement** ou la barre d'outils, soit depuis la fenêtre du compilateur :



**Note :** La licence "4D Team Server" est nécessaire côté 4D Server pour que les postes clients puissent accéder à la fonction de compilation.

Un seul poste client peut compiler la base à un instant donné. La compilation par un client verrouille la fonction pour les autres postes distants. Si un autre poste client tente de compiler la base au même moment, un message d'alerte apparaît.

Pendant qu'un poste client effectue une compilation, les autres clients peuvent continuer de travailler et de modifier des méthodes ou tout autre élément de structure. Le code compilé et le code interprété seront alors différents, ce qui nécessitera de recompiler la base ultérieurement.

Le code compilé est envoyé dans le fichier .4DB sur le serveur au fur et à mesure de la compilation. Côté client, après la fin de l'opération, il est possible de redémarrer le serveur en mode interprété ou en mode compilé en utilisant les commandes correspondantes dans le menu **Exécution**. Lorsqu'un 4D distant demande le redémarrage du serveur en compilé/en interprété, la boîte de dialogue standard d'arrêt du serveur apparaît, permettant d'accorder un délai ou d'envoyer un message d'avertissement aux autres clients (cf. section **Quitter 4D Server**). Lorsque le serveur a redémarré, le client à l'origine du redémarrage est automatiquement reconnecté.

Côté serveur, le redémarrage en compilé/interprété nécessite d'utiliser la boîte de dialogue standard d'ouverture de fichier (pop up menu associé au bouton **Ouvrir**).

**Note :** La compilation sur réseau WAN est déconseillée pour des raisons de performances (notamment dans le cas de bases comportant de nombreuses méthodes) car l'opération génère une quantité importante d'échanges réseau.

## **4D Server et le langage 4D**

-  4D Server et le langage 4D
-  4D Server, ensembles et sélections
-  Procédures stockées
-  Procédures stockées sur les clients
-  Import basé sur les procédures stockées (exemple)
-  Services basés sur les procédures stockées (exemple)
-  Attribut Exécuter sur serveur

Avec 4D Server, vous pouvez exécuter du code 4D sur le poste serveur. Il y a quatre situations dans lesquelles du code 4D peut être exécuté sur la machine serveur :

- Triggers
- Procédures stockées
- Méthodes projet avec attribut "Exécuter sur serveur"
- Méthodes base

### Triggers

---

Un trigger est une méthode associée à une table. Les triggers peuvent empêcher des opérations "illégalles" sur les enregistrements de votre base de données. Les triggers sont des outils très puissants qui permettent de restreindre les opérations sur une table ainsi que d'empêcher la perte ou la corruption accidentelle de données. Par exemple, dans un système de facturation, vous pouvez empêcher que ce soit d'ajouter un enregistrement sans indiquer le nom du client.

Les triggers sont exécutés sur la machine sur laquelle le moteur de la base de données est placé. Avec 4D Server, les triggers sont exécutés dans le contexte des process tournant sur la machine serveur, et non sur la machine cliente. Plus précisément, ils sont exécutés dans le contexte des process "jumeaux" des process utilisateurs qui appellent l'opération de base de données. Ces process jumeaux partagent avec le process utilisateur sur la machine cliente le contexte de base de données (notamment le statut des transactions et le verrouillage des enregistrements) mais pas le contexte du langage (variables process, ensembles, sélections courantes). A noter toutefois que l'enregistrement courant de la table du trigger est identique dans tous les contextes.

Pour plus d'informations sur les triggers, reportez-vous à la section [Présentation des triggers](#) dans le manuel Langage de 4D.

### Procédures stockées

---

Une procédure stockée 4D est une méthode projet exécutée dans un process séparé sur le poste serveur (ou sur tout poste client), par opposition au poste client qui est à l'origine de la méthode. Pour plus d'informations sur ce point, reportez-vous à la section [Procédures stockées](#).

### Méthodes avec attribut "Exécuter sur serveur"

---

Les méthodes projet disposant de l'attribut "Exécuter sur serveur" sont également exécutées sur le serveur. Toutefois, à la différence des procédures stockées, elles sont exécutées dans le process "jumeau" du process client et bénéficient alors de son contexte de base de données, comme pour les triggers. Pour plus d'informations, reportez-vous à la section [Attribut Exécuter sur serveur](#).

### Méthodes base

---

Quatre méthodes base sont exécutées uniquement sur la machine serveur :

- [Méthode base Sur démarrage serveur](#)
- [Méthode base Sur arrêt serveur](#)
- [Méthode base Sur ouverture connexion serveur](#)
- [Méthode base Sur fermeture connexion serveur](#)

Cinq méthodes base peuvent être exécutées sur la machine serveur ou sur un poste client en fonction du contexte :

- [Méthode base Sur authentification Web](#)
- [Méthode base Sur connexion Web](#)
- [Méthode base Sur authentification SQL](#)
- [Méthode base Sur démarrage sauvegarde](#)
- [Méthode base Sur arrêt sauvegarde](#)

Trois méthodes base peuvent être exécutées uniquement sur un poste client :

- [Méthode base Sur ouverture](#)
- [Méthode base Sur fermeture](#)
- [Méthode base Sur déposer](#)

Reportez-vous aux sections correspondantes dans ce manuel ainsi que dans le manuel Langage de 4D pour plus d'informations sur chacun des types de méthodes base.

## 4D Server et les variables

---

- 4D Server maintient une table de variables interprocess. La portée de ces variables est le poste serveur. Lorsque 4D Server exécute une base compilée, la définition de la table des variables interprocess est commune entre le serveur et les machines clientes, chaque machine possédant sa propre instance.
- Comme tout process, chaque procédure stockée, méthode base et trigger possède sa propre table de variables process. Ces variables process peuvent être créées et utilisées de manière dynamique pendant chaque phase de l'exécution.

## **4D Server, ensembles et sélections temporaires**

---

Avec 4D Server, la visibilité des ensembles et des sélections temporaires dépend de l'origine de la création (process serveur ou process client) et de la nature de ces objets (objets locaux, process ou interprocess). Pour plus d'informations sur ce point, reportez-vous à la section *4D Server, ensembles et sélections*.

## 4D Server, ensembles et sélections

Comme décrit dans les sections [Présentation des ensembles](#) et [Présentation des Sélections Temporaires](#) du manuel Langage de 4D, vous pouvez créer et utiliser des ensembles et des sélections temporaires interprocess, process et locaux :

- **Ensembles/Sélections temporaires process** : Un objet process n'est "visible" que par le process dans lequel il a été créé et, s'il a été créé dans un process client, par le process "jumeau" créé sur le serveur. Les ensembles process sont effacés à la fin de l'exécution de la méthode process. Le nom des objets process ne comporte aucun préfixe particulier.
- **Ensembles/Sélections temporaires interprocess** : Un objet interprocess est visible par tous les process sur le poste (client ou serveur) où il a été créé. Un objet est interprocess lorsque son nom est préfixé des symboles (<>) — le signe "inférieur à" suivi du signe "supérieur à".  
**Note** : Cette syntaxe est valable sous Windows et Mac OS. De plus, sous Mac OS uniquement, vous pouvez employer le symbole diamant (Option+v sur un clavier français).
- **Ensembles/Sélections temporaires locaux/clients** : Un objet local/client est visible uniquement dans le process qui l'a créé. Le nom d'un objet local/client est préfixé du symbole dollar (\$). A noter que l'ensemble système **UserSet**, bien que son nom ne débute pas par \$, est un ensemble local/client.

Le tableau suivant indique les principes de visibilité des sélections temporaires et des ensembles en fonction de leur lieu de création (le tableau est identique pour les deux types d'objets) :

	Process client	Autres process du client	Autres clients	Process serveur	Autres process du serveur
<b>Création dans un process client</b>					
\$test	x				
test	x			x (Trigger)	
<>test	x	x			
<b>Création dans un process serveur</b>					
\$test				x	
test				x	
<>test				x	x

*x = visible*

Vous devez garder à l'esprit cette matrice de visibilité en fonction des opérations que vous souhaitez effectuer. Par exemple, si vous voulez effectuer une opération de type **DIFFERENCE**, **INTERSECTION** ou **REUNION**, assurez-vous que tous les ensembles sont visibles sur le poste effectuant l'opération.

Pour des raisons d'optimisation, il est conseillé de choisir le lieu de création et la portée des objets en fonction de leur nécessité de visibilité.

### Qu'est-ce qu'une procédure stockée dans un système SQL ?

---

L'expression "Procédure Stockée" provient du monde des serveurs SQL. Lorsqu'une station cliente envoie une requête à un serveur SQL, elle envoie en réalité du texte en langage SQL au serveur SQL. Cette requête fait l'objet d'une analyse syntaxique (parsing) et est interprétée sur le serveur SQL avant d'être exécutée. De toute évidence, si le texte de la requête est important et si la requête est envoyée plusieurs fois au cours d'une session, il peut y avoir beaucoup de temps passé pour l'envoi de ce code sur le réseau, le parsing et l'interprétation de la requête. Ainsi est née l'idée de faire en sorte que la requête soit envoyée sur le réseau, analysée et interprétée en une seule fois, et qu'ensuite elle soit exécutée à chaque fois qu'elle est envoyée par la station cliente. La solution était de conserver le code source de la requête (en d'autres termes, une procédure) sur le serveur et que le client envoie une requête comportant seulement le nom de la procédure à exécuter. En conséquence, cette procédure est dite "stockée" sur le serveur, d'où le nom de Procédure Stockée. Notez qu'une procédure SQL est une procédure qui peut recevoir des paramètres venant de la station cliente, exécuter les tâches pour lesquelles elle est faite (de façon synchrone ou asynchrone) et le cas échéant retourner un résultat au client. Quand un client demande l'exécution d'une procédure stockée, ce qu'il fait, c'est dans une certaine mesure déléguer l'exécution du code sur le serveur.

### Qu'est-ce qu'une procédure stockée dans 4D Server ?

---

Bien que nous utilisions aussi le terme "procédures stockées" par analogie, puisqu'il est compréhensible par tous dans l'industrie, les objets que nous appelons ainsi dans 4D Server ont des fonctionnalités plus larges que celles que l'on trouve habituellement.

Avec 4D en mode local, lorsque par exemple vous utilisez la commande **Nouveau process**, vous pouvez ouvrir un process utilisateur dans lequel vous allez exécuter une méthode. Cette méthode est appelée une **méthode process** (voir à ce sujet la section **Méthodes projet** dans le manuel Langage de 4D).

Avec 4D Server, vous pouvez faire la même chose sur un poste client. De plus, en utilisant la commande **Executer sur serveur**, vous pouvez créer sur le serveur un process utilisateur dans lequel vous allez exécuter une méthode. De même, en utilisant la commande **EXECUTER SUR CLIENT**, vous pouvez exécuter une méthode dans un autre process sur un autre poste client. Dans les deux cas, cette méthode est appelée une **procédure stockée**, et par extension, le process démarré sur le serveur ou sur un autre poste client est aussi appelé procédure stockée.

**La différence principale entre une procédure stockée SQL et une procédure stockée 4D Server est que dans le premier cas vous exécutez une procédure SQL et dans le second, vous exécutez un process 4D autonome.**

### Architecture des procédures stockées de 4D

---

Comme un process 4D normal, une procédure stockée dispose de son propre environnement :

- **Une sélection courante par table** : chaque procédure stockée a sa propre sélection courante. Une table peut avoir une sélection courante différente dans chaque procédure stockée.
- **Un enregistrement courant par table** : chaque table peut avoir un enregistrement courant différent dans chaque procédure stockée.
- **Variables** : chaque procédure stockée a ses propres variables process. Les variables process ne sont reconnues que dans le contexte de la procédure stockée à laquelle elles appartiennent.
- **Table par défaut** : chaque procédure stockée a sa propre table par défaut.
- **Ensembles process** : chaque procédure stockée a ses propres ensemble process.
- **Appeler sur erreur** : chaque procédure stockée dispose d'une méthode de gestion d'erreur.
- **Fenêtre de débogage** : chaque procédure stockée dispose de sa propre fenêtre de débogage.

En termes d'interface utilisateur, une procédure stockée peut ouvrir des fenêtres et afficher des données (i.e. **AFFICHER ENREGISTREMENT**).

Une procédure stockée exécutée sur un poste client 4D permet la saisie de données.

En revanche, une procédure stockée exécutée sur le serveur ne permet pas la saisie de données.

Vous pouvez démarrer autant de procédures stockées que vous le souhaitez dans la mesure où votre matériel et la mémoire le permettent. En fait, la machine serveur doit être considérée comme une machine qui est non seulement capable de répondre à des clients 4D et à des navigateurs Web, mais aussi capable d'exécuter des process qui peuvent interagir avec d'autres process fonctionnant aussi bien sur le serveur que sur des 4D distants.

De la même façon que 4D fournit un environnement multitâche aux process utilisateurs fonctionnant sur le poste, 4D Server fournit un environnement multitâche aux procédures stockées. Par exemple, 4D Server maintient une table des variables interprocess qui peuvent être utilisées par les procédures stockées pour communiquer entre elles.

**Note** : La propriété de méthode "Exécuter sur serveur" permet elle aussi d'exécuter une méthode dans un process sur le serveur, mais la méthode utilise dans ce cas le process "jumeau" du process client, ce qui lui permet en particulier de bénéficier de l'environnement de ce process client. Il ne s'agit pas dans ce cas d'une procédure stockée 4D. Pour plus d'informations, reportez-vous à la section **Attribut Exécuter sur serveur**.

### Que peut faire une procédure stockée ?

---

Une grande partie fonctionnalités des process et des commandes décrites dans le manuel Langage de 4D s'appliquent aussi aux procédures stockées — à l'exception de la saisie de données pour les procédures stockées exécutées sur le serveur.

Une procédure stockée peut ajouter, rechercher, trier, mettre à jour ou détruire des enregistrements. Une procédure stockée peut utiliser des ensembles et des sélections temporaires, accéder à des documents sur disque, travailler avec des BLOBs, imprimer des enregistrements, etc. Pensez simplement qu'au lieu d'effectuer une tâche sur votre poste 4D local, vous l'effectuez sur le serveur ou sur un ou plusieurs autres postes clients.

Un avantage évident des procédures stockées exécutées sur le serveur est que précisément une procédure stockée s'exécute sur la machine serveur, là où se trouve le moteur de la base de données. Par exemple, un **APPLIQUER A SELECTION** n'est pas efficace sur le réseau, mais l'est à l'intérieur d'une procédure stockée. L'exemple proposé dans la section **Import basé sur les procédures stockées (exemple)** montre les optimisations remarquables que vous pouvez obtenir en utilisant une procédure stockée.

Les procédures stockées exécutées sur un ou plusieurs autres postes clients autorisent, quant à elles, l'optimisation de la répartition des tâches entre les clients, ou encore la communication inter-clients. La description de la commande **INSCRIRE CLIENT** fournit un exemple simple de messagerie utilisant des procédures stockées exécutées sur les clients.

Cependant, l'avantage le plus évident de l'architecture des procédures stockées se trouve dans la dimension supplémentaire qu'elles apportent à 4D Server : grâce aux procédures stockées, vous pouvez implémenter vos propres services 4D Server. La limite, c'est votre imagination. L'exemple fourni dans la section **Services basés sur les procédures stockées (exemple)** montre une procédure stockée capable de fournir des informations sur 4D Server à ses clients. Vous pouvez, par exemple, lister les volumes de la machine serveur. Cet exemple pourrait être étendu facilement, pour, notamment, renvoyer les informations sur les répertoires ou documents au client.

## **Que ne peut pas faire une procédure stockée (sur le serveur) ?**

---

De manière générale, les procédures stockées exécutées sur le serveur ne doivent pas effectuer d'opérations impliquant des éléments d'interface (menus, fenêtres, formulaires...). En effet, les mécanismes de gestion d'interface ne sont pas pris en charge sur le serveur.

Les commandes provoquant l'apparition de boîtes de dialogue sur le poste serveur ainsi que celles liées à la saisie de données sont également à proscrire.

Voici la liste des commandes ne devant PAS être utilisées dans le cadre des procédures stockées exécutées sur le serveur. Ces commandes sont classées en trois catégories :

- **Les commandes interdites sur le serveur**

La présence d'une de ces commandes dans une procédure stockée provoque l'apparition d'une boîte de dialogue d'alerte indiquant que la commande ne peut pas être exécutée sur 4D Server. L'erreur 67 est retournée, elle peut être interceptée à l'aide d'une méthode installée par la commande **APPELER SUR ERREUR**.

**ACTIVER LIGNE MENU**  
**AFFICHER BARRE MENUS**  
**AJOUTER ENREGISTREMENT**  
**AJOUTER LIGNE MENU**  
**AJOUTER SOUS ENREGISTREMENT**  
**APPELER PROCESS**  
**APPELER SUR EVENEMENT**  
**CACHER BARRE MENUS**  
**CHANGER LICENCES**  
**CHANGER PRIVILEGES**  
**CHERCHER PAR EXEMPLE**  
**Creer fenetre externe**  
**CREER FICHER DONNEES**  
**CUMULER SUR**  
**ECRIRE IMAGE DANS BIBLIOTHEQUE**  
**FILTREER EVENEMENT**  
**FIXER MARQUE LIGNE MENU**  
**FIXER RACCOURCI LIGNE MENU**  
**FIXER STYLE LIGNE MENU**  
**FIXER TEXTE LIGNE MENU**  
**GRAPHE SUR SELECTION**  
**INACTIVER LIGNE MENU**  
**INSERER LIGNE MENU**  
**Lire marque ligne menu**  
**Lire style ligne menu**  
**Lire texte ligne menu**  
**Lire titre menu**  
**Lire touche ligne menu**  
**Menu choisi**  
**MODIFIER ENREGISTREMENT**  
**MODIFIER SELECTION**

## MODIFIER SOUS ENREGISTREMENT

Niveau

## NIVEAUX DE RUPTURES

Nombre de lignes de menu

Nombre de menus

## OUVRIR FICHER DONNEES

Page impression

## PARAMETRES IMPRESSION

QR ETAT

SAUT DE PAGE

Sous total

## SUPPRIMER IMAGE DANS BIBLIOTHEQUE

SUPPRIMER LIGNE MENU

## UTILISER PARAMETRES IMPRESSION

VISUALISER SELECTION

- **Les commandes déconseillées sur le serveur**

L'utilisation de ces commandes dans des procédures stockées est fortement déconseillée car leur fonctionnement n'est pas adapté à une exécution sur le serveur. Ces commandes peuvent bloquer le serveur, provoquer des erreurs, et plus généralement ne produisent pas les effets escomptés. Aucun code d'erreur spécifique n'est retourné.

Activation

ACTIVER BOUTON

AFFICHER FENETRE

AJOUTER A LISTE

AJOUTER DONNEES AU CONTENEUR

AJOUTER SEGMENT DE DONNEES

ALLER A OBJET

Ancien

Appartient au groupe

Appel extérieur

Après

Avant

BLOB VERS DOCUMENT

BLOB vers liste

CACHER BARRE OUTILS

CACHER FENETRE

CACHER PROCESS

CHANGER COORDONNEES FENETRE

CHANGER ELEMENT

CHANGER MOT DE PASSE

CHANGER POINTEUR SOURIS

CHANGER PROPRIETES ELEMENT

CHANGER PROPRIETES LISTE

CHANGER UTILISATEUR COURANT

Charger liste

CHERCHER PAR FORMULE (2)

Chercher fenetre

CHERCHER (2)

COORDONNEES ECRAN

COORDONNEES FENETRE

Copier liste

Créer document (1)

Créer fichier ressources (1)

C\_GRAPHE

DÉPLACER FENETRE

Desactivation

DIALOGUE

Ecran principal

Ecrire proprietes groupe

Ecrire proprietes utilisateur

EFFACER CONTENEUR

EFFACER FENETRE

Element parent

Elements selectionnes

En entete

En pied

En rupture

Evenement formulaire  
EXPORTER DONNEES (1)  
Fenetre premier plan  
Fenetre suivante  
FILTRER FRAPPE CLAVIER  
FIXER IMAGE DANS CONTENEUR  
FIXER FICHER HISTORIQUE  
FIXER MINUTEUR  
FIXER PROFONDEUR ECRAN  
FIXER TEXTE DANS CONTENEUR  
FIXER TITRES CHAMPS  
FIXER TITRES TABLES  
FORM ALLER A PAGE  
FORM FIXER ENTREE  
FORM FIXER SORTIE  
FORM DERNIERE PAGE  
FORM PAGE PRECEDENTE  
FORM PAGE SUIVANTE  
FORM Lire page courante  
FORM LIRE PROPRIETES  
FORM PREMIERE PAGE  
Frappe clavier  
GENERER CLIC  
GENERER EVENEMENT  
GENERER FRAPPE CLAVIER  
Hauteur barre de menus  
Hauteur ecran  
IMPORTER DONNEES (1)  
INACTIVER BOUTON  
INFORMATION ELEMENT  
INSCRIRE CLIENT  
INSERER DANS LISTE  
INVERSER FOND  
Largeur ecran  
LIRE DONNEES CONTENEUR  
LIRE IMAGE DANS CONTENEUR  
LIRE LISTE GROUPE  
LIRE LISTE UTILISATEURS  
LIRE PROPRIETES ELEMENT  
LIRE PROPRIETES GROUPE  
LIRE PROPRIETES LISTE  
LIRE PROPRIETES UTILISATEUR  
Lire texte dans conteneur  
Lire texte edite  
LISTE DES POLICES  
LISTE FENETRES  
LISTE VERS BLOB  
Liste existante  
Majuscule enfoncee  
MARQUER ENREGISTREMENTS  
MAXIMISER FENETRE  
MINIMISER FENETRE  
Modifie  
MONTRER PROCESS  
NE PAS VALIDER  
Nom de police  
Nombre ecrans  
Nombre elements  
Nouvelle liste  
Numero de police  
OBJET DEPLACER  
OBJET FIXER COULEUR  
OBJET FIXER COULEURS RVB  
OBJET FIXER NOM ENUMERATION  
OBJET FIXER FILTRE SAISIE  
OBJET FIXER FORMATAGE  
OBJET FIXER SAISSABLE  
OBJET FIXER TITRE

OBJET FIXER VISIBLE  
OBJET LIRE COORDONNEES  
Objet focus  
Ouvrir document (1)  
Ouvrir fichier ressources (1)  
PARAMETRES DU GRAPHE  
PASSER AU PREMIER PLAN  
Pendant  
Pop up menu  
POSITION SOURIS  
Position déposer  
Position element liste  
Process de la fenetre  
Process de premier plan  
PROFONDEUR ECRAN  
PROPRIETES GLISSER DEPOSER  
REDESSINER  
REDESSINER FENETRE  
REDESSINER LISTE  
REFUSER  
SELECTIONNER ELEMENTS PAR POSITION  
SELECTIONNER ELEMENTS PAR REFERENCE  
SELECTIONNER TEXTE  
Selectionner dossier  
Self  
STOCKER LISTE  
SUPPRIMER DANS LISTE  
SUPPRIMER LISTE  
SUPPRIMER UTILISATEUR  
Table du formulaire courant  
Tester conteneur  
TEXTE SELECTIONNE  
Titre fenetre  
TRIER LISTE  
TRIER (2)  
Type fenetre  
Utilisateur courant  
Utilisateur supprime  
VALIDER  
Valider mot de passe

(1) Uniquement lorsque le premier paramètre est une chaîne vide.

(2) Uniquement lorsque la syntaxe utilisée provoque l'apparition de la boîte de dialogue (ex : **TRIER ([Table])**).

- **Les commandes sans effet sur le serveur**

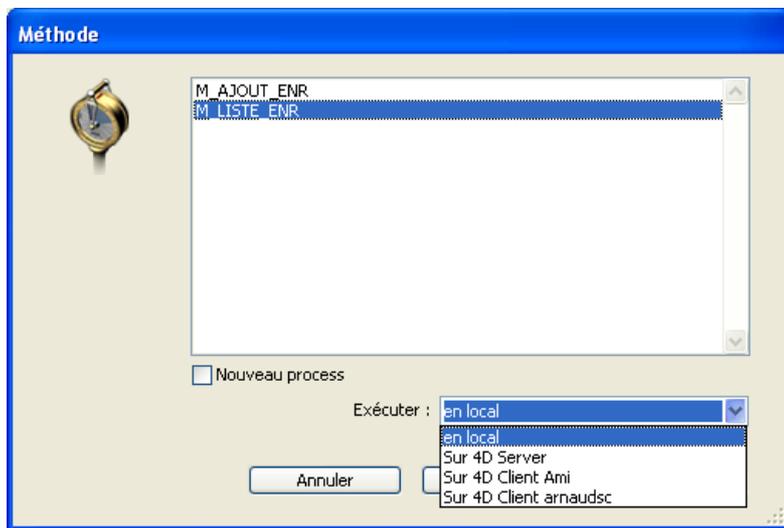
Ces commandes sont sans effet lorsqu'elles sont exécutées dans une procédure stockée sur le serveur. Aucun code d'erreur spécifique n'est retourné.

FIXER BARRE MENUS  
LAISSER MESSAGES  
SUPPRIMER MESSAGES  
GRAPHE  
AFFICHER BARRE OUTILS

## Comment lancer une procédure stockée ?

---

- Depuis 4D, vous pouvez lancer manuellement une procédure stockée dans la boîte de dialogue d'exécution de méthode :



Vous pouvez l'exécuter sur 4D Server ou sur un autre poste 4D client. Notez que, pour que les postes clients 4D apparaissent dans cette liste, ils doivent auparavant avoir été inscrits (cf. sections **Procédures stockées sur les clients** et commande **INSCRIRE CLIENT**).

- Toujours sur 4D, vous pouvez lancer une procédure stockée par programmation à l'aide des commandes **Executer sur serveur** ou **EXECUTER SUR CLIENT**.

**Note** : Il n'est pas possible d'utiliser les commandes de gestion des process **ENDORMIR PROCESS**, **SUSPENDRE PROCESS** et **REACTIVER PROCESS** à partir d'un 4D distant avec des procédures stockées sur le serveur.

- Une méthode exécutée sur 4D Server (méthode base du serveur, méthode avec attribut Exécuter sur serveur, trigger ou procédure stockée) peut elle-même lancer une procédure stockée à l'aide des commandes **Executer sur serveur**, **Nouveau process** ou **EXECUTER SUR CLIENT**.

## Communication interprocess entre les procédures stockées et les process utilisateurs

Les procédures stockées peuvent communiquer entre elles à l'aide des :

- variables interprocess
- sémaphores globaux ou locaux
- enregistrements
- ensembles interprocess et sélections temporaires interprocess
- commandes **LIRE VARIABLE PROCESS**, **ECRIRE VARIABLE PROCESS** et **VARIABLE VERS VARIABLE**.

Reportez-vous aux sections correspondantes du manuel Langage de 4D. Une fois encore, gardez à l'esprit que les commandes 4D agissent sur le poste qui exécute la procédure stockée (serveur ou clients) de la même manière qu'en local sur un poste client.

**Note** : Les mécanismes **APPELER PROCESS** et **Appel extérieur** n'ont pas de signification sur le poste serveur car les procédures stockées ne disposent pas d'interface utilisateur avec saisie de données.

En outre, notez cette fonctionnalité importante : les process utilisateur des clients (process tournant sur un poste client) peuvent lire et écrire les variables process (\*) d'une procédure stockée à l'aide des commandes **LIRE VARIABLE PROCESS**, **ECRIRE VARIABLE PROCESS** et **VARIABLE VERS VARIABLE**.

(\*) ainsi que les variables interprocess du poste serveur.

**Important** : La communication process "Intermachine" permise par les commandes **LIRE VARIABLE PROCESS**, **ECRIRE VARIABLE PROCESS** et **VARIABLE VERS VARIABLE** n'est possible que du client vers le serveur. C'est toujours un process client qui lit ou écrit les variables d'une procédure stockée.

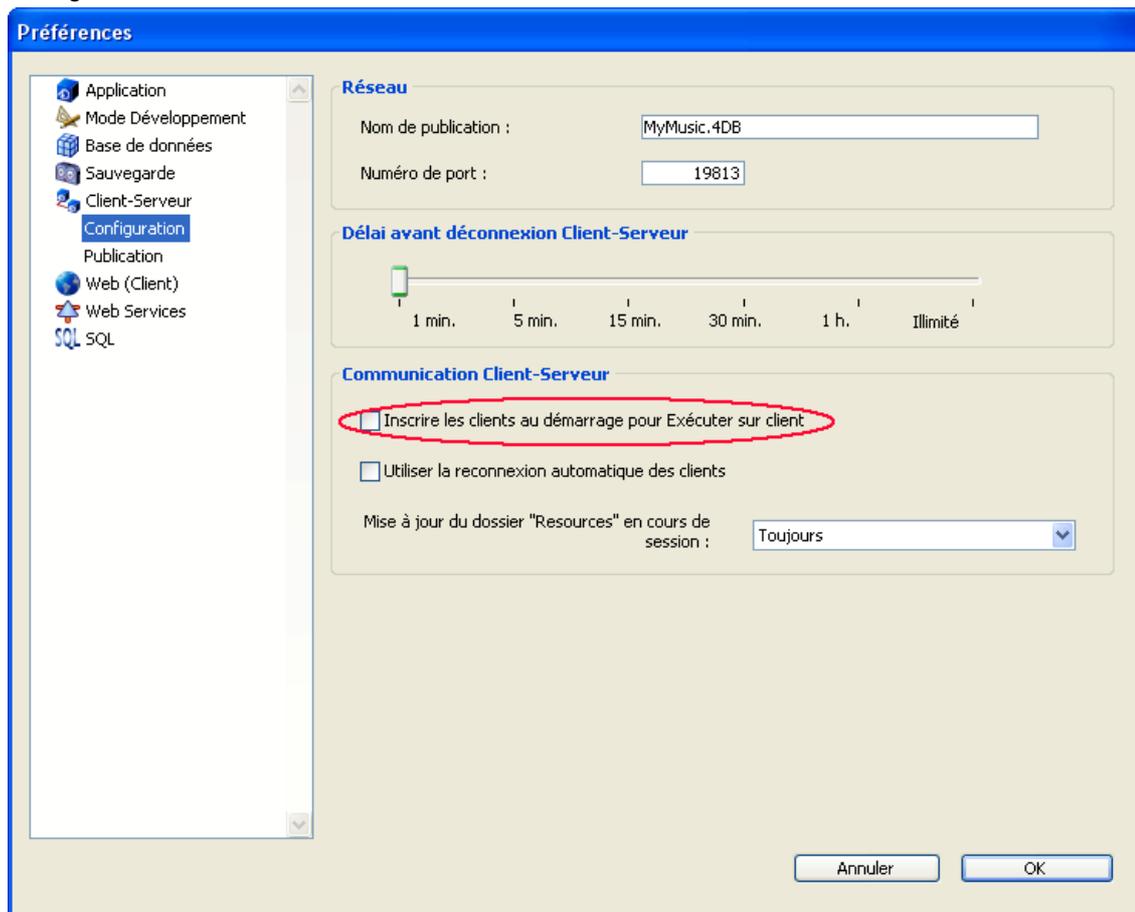
## Procédures stockées sur les clients

Vous pouvez exécuter des procédures stockées sur un ou plusieurs autres clients 4D. Le fonctionnement général des procédures stockées exécutées sur les clients est identique à celui des procédures stockées exécutées sur le serveur, à la différence près qu'une procédure stockée exécutée sur un 4D Client peut, elle, "saisir" des données. Ce fonctionnement est détaillé dans la section **Procédures stockées**.

Tout poste client devant exécuter des procédures à la demande du serveur ou d'un autre poste client doit être explicitement "inscrit" pour la session. Pour inscrire un client, vous disposez de deux solutions : inscrire automatiquement chaque client à la connexion, ou inscrire les clients par programmation.

### Inscrire automatiquement chaque client 4D qui se connecte à 4D Server

L'option "Inscrire les clients au démarrage pour Exécuter sur client" est disponible dans les Préférences de l'application, page "Configuration" du thème "Client-Serveur" :



Lorsque cette option est cochée, chaque poste client 4D qui se connecte à la base est automatiquement référencé auprès de 4D Server comme pouvant exécuter des procédures stockées. Un process de type 4D Client portant le nom de la machine du poste client est créé sur le serveur. Un process équivalent est également créé sur chaque poste client.

### Inscrire les 4D Client par programmation

Vous pouvez également inscrire un ou plusieurs postes clients 4D par programmation. Cette option vous permet de sélectionner les postes clients devant être inscrits et de choisir leur nom d'inscription.

Le thème "Process" comporte la commande **INSCRIRE CLIENT**, permettant d'inscrire un poste client sous le nom que vous voulez.

### Désinscrire un 4D Client

Quel que soit le mode d'inscription des postes clients, vous pouvez à tout moment désinscrire tout poste client du serveur pour la session courante. Pour cela, il vous suffit d'exécuter la commande **DESINSCRIRE CLIENT** (thème "Process") sur le(s) poste(s) client(s) à désinscrire.

Le process de gestion de l'inscription (portant le nom du client) disparaît alors du groupe de process de l'utilisateur sur le poste serveur, ainsi que sur le poste client.

**Note** : La commande **LIRE CLIENTS INSCRITS** vous permet d'obtenir la liste et la charge de travail (le nombre de méthodes

restant à exécuter) des clients inscrits dans la session.

## Import basé sur les procédures stockées (exemple)

L'exemple suivant démontre comment l'import de données peut être accéléré de manière spectaculaire en environnement client/serveur. La méthode **Import classique** listée ci-dessous vous permet de mesurer combien de temps prend un import d'enregistrements basé sur la commande **IMPORTER TEXTE** :

```
` Méthode projet Import classique
$vhDocRef:=Ouvrir document("")
Si (OK=1)
    FERMER DOCUMENT ($vhDocRef)
    FORMULAIRE ENTREE ([Table1];"Import")
    $vhStartTime:=Heure courante
    IMPORTER TEXTE ([Table1];Document)
    $vhEndTime:=Heure courante
    ALERTE ("L'opération a duré "+Chaine(0+($vhEndTime-$vhStartTime))+" secondes.")
Fin de si
```

Avec l'import de données classique, 4D analyse le fichier ASCII puis, pour chaque enregistrement, crée un nouvel enregistrement, remplit les champs avec les valeurs importées et envoie l'enregistrement au poste serveur afin qu'il soit ajouté à la base. Par conséquent, de nombreuses requêtes circulent sur le réseau. Afin d'optimiser l'opération, vous pouvez utiliser des procédures stockées pour effectuer l'import localement sur le poste serveur. Le poste client charge le document dans un BLOB et déclenche une procédure stockée en passant le BLOB comme paramètre. La procédure stockée place le BLOB dans un document sur le disque du poste serveur, puis importe le document en local. L'import des données est par conséquent effectué localement à une vitesse comparable à celle d'une version locale de 4D, car la plupart des requêtes transitant sur le réseau ont été éliminées. Voici la méthode projet **CLIENT IMPORT**. Lancée sur le poste client, elle déclenche l'exécution de la procédure stockée **SERVER IMPORT**, listée à la suite :

```
` Méthode projet CLIENT IMPORT
` CLIENT IMPORT ( Pointeur ; Texte )
` CLIENT IMPORT ( -> [Table] ; Formulaire entrée )

C_POINTEUR ($1)
C_TEXTE ($2)
C_HEURE ($vhDocRef)
C_BLOB ($vxData)
C_ENTIER LONG (spErrCode)

` Sélectionnez le document à importer
$vhDocRef:=Ouvrir document("")
Si (OK=1)
    ` Si un document était sélectionné, ne pas le garder ouvert
    FERMER DOCUMENT ($vhDocRef)
    $vhStartTime:=Heure courante
    ` Essayons de le charger en mémoire
    DOCUMENT VERS BLOB (Document;$vxData)
    Si (OK=1)
        ` Si le document a pu être chargé dans le BLOB,
        ` Démarrer la procédure stockée qui va importer les données sur le poste serveur
        $spProcessID:=Executer sur serveur ("SERVER IMPORT";32*1024;"Serveur Import
Services";Table ($1) ;$2;$vxData)
        ` Nous n'avons alors plus besoin du BLOB dans ce process
        EFFACER VARIABLE ($vxData)
        ` Attendons l'achèvement de l'opération effectuée par la procédure stockée
        Repeter
            ENDORMIR PROCESS (Numero du process courant;300)
            LIRE VARIABLE PROCESS ($spProcessID;spErrCode;spErrCode)
            Si (Indefinie (spErrCode) )
        ` Note: si la procédure stockée n'a pas initialisé sa propre instance
        ` de la variable spErrCode, il se peut qu'une variable indéfinie soit retournée
        spErrCode:=1
        Fin de si
        Jusque (spErrCode<=0)
    ` Envoyons un accusé de réception à la procédure stockée
    spErrCode:=1
    ECRIRE VARIABLE PROCESS ($spProcessID;spErrCode;spErrCode)
```

```

    $vhEndTime:=Heure courante
    ALERTE("L'opération a duré "+Chaine(0+($vhEndTime-$vhStartTime))+ " secondes.")
Sinon
    ALERTE("Il n'y a pas assez de mémoire pour charger le document.")
Fin de si
Fin de si

```

Voici la méthode projet **SERVER IMPORT** exécutée en tant que procédure stockée :

```

` Méthode projet SERVER IMPORT
` SERVER IMPORT ( Entier long ; Texte ; BLOB )
` SERVER IMPORT ( Numéro de table ; Formulaire entrée ; Données importées )

C_ENTIER LONG($1)
C_TEXTE($2)
C_BLOB($3)
C_ENTIER LONG(spErrCode)

` L'opération n'est pas encore terminée, affectons 1 à spErrCode
spErrCode:=1
$vpTable:=Table($1)
FORMULAIRE ENTREE($vpTable->;$2)
$vsDocName:="Fichier Import "+Chaine(1+Hasard)
Si(Sous Windows)
    $vsDocName:=$vsDocName+".txt" ` Sous Windows, l'extension est obligatoire
Fin de si
SUPPRIMER DOCUMENT($vsDocName)
BLOB VERS DOCUMENT($vsDocName;$3)
LECTURE ASCII($vpTable->;$vsDocName)
SUPPRIMER DOCUMENT($vsDocName)
` L'opération est terminée, affectons 0 à spErrCode
spErrCode:=0
` Attendons que le poste client à l'origine de la requête ait reçu les résultats
Repeter
    ENDORMIR PROCESS(Numero du process courant;1)
Jusque(spErrCode>0)

```

**Note :** La méthode projet **Sous Windows** est fournie dans la section **Présentation des documents système** du manuel Langage de 4D.

Une fois que ces deux méthodes projet ont été implémentées dans votre base, vous pouvez effectuer un import basé sur une procédure stockée, en écrivant par exemple :

```

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

```

Si vous réalisez quelques tests comparatifs, vous pourrez constater qu'avec ce type de méthode, l'import des enregistrements est jusqu'à 60 fois plus rapide qu'un import "classique".

## Services basés sur les procédures stockées (exemple)

Dans l'exemple présenté dans la section **Import basé sur les procédures stockées (exemple)**, une procédure stockée est lancée puis stoppée à chaque fois qu'une opération d'import de données est requise. Dans le présent exemple, une procédure stockée est lancée automatiquement lorsque la base serveur est ouverte et peut être stoppée et redémarrée à volonté par tout 4D connecté à la base. Dès qu'elle est exécutée, la procédure stockée peut répondre de manière asynchrone à de multiples requêtes envoyées par les clients connectés à la base.

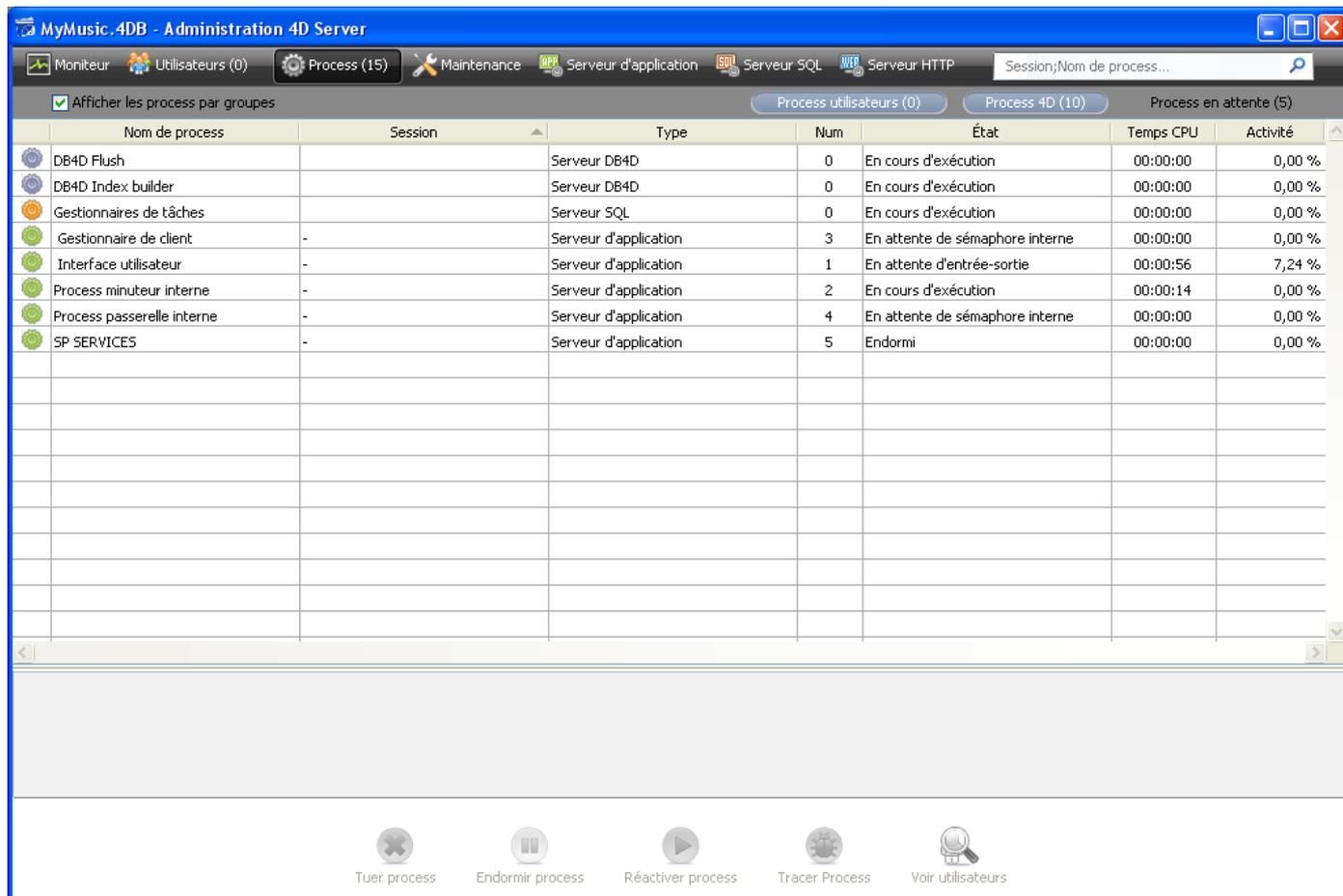
Alors que la section **Import basé sur les procédures stockées (exemple)** vous montre comment optimiser un service 4D Server existant, cet exemple vous montre comment créer des services nouveaux et personnalisés disponibles pour tous les 4D connectés. En outre, cet exemple peut être utilisé comme modèle pour la création de vos propres services.

### Démarrage automatique de la procédure stockée

La procédure stockée est lancée automatiquement par la **Méthode base Sur démarrage serveur** :

```
\` Méthode base Sur démarrage serveur  
START SP SERVICES
```

Comme la **Méthode base Sur démarrage serveur** lance la méthode projet **SP SERVICES** en tant que procédure stockée, cette dernière s'exécute dès que la base est ouverte avec 4D Server, que des client soient connectés à la base ou non. Ci-dessous, la fenêtre d'administration de 4D Server affiche la procédure stockée en cours d'exécution alors qu'aucun client n'est encore connecté.



Nom de process	Session	Type	Num	État	Temps CPU	Activité
DB4D Flush		Serveur DB4D	0	En cours d'exécution	00:00:00	0,00 %
DB4D Index builder		Serveur DB4D	0	En cours d'exécution	00:00:00	0,00 %
Gestionnaires de tâches		Serveur SQL	0	En cours d'exécution	00:00:00	0,00 %
Gestionnaire de client	-	Serveur d'application	3	En attente de sémaphore interne	00:00:00	0,00 %
Interface utilisateur	-	Serveur d'application	1	En attente d'entrée-sortie	00:00:56	7,24 %
Process minuteur interne	-	Serveur d'application	2	En cours d'exécution	00:00:14	0,00 %
Process passerelle interne	-	Serveur d'application	4	En attente de sémaphore interne	00:00:00	0,00 %
SP SERVICES	-	Serveur d'application	5	Endormi	00:00:00	0,00 %

### Démarrer et stopper à volonté la procédure stockée

Voici la méthode projet **START SP SERVICES** :

```
\` Méthode projet START SP SERVICES  
ÔvLSPServices:=Executer sur serveur ("SP SERVICES";32*1024;"SP SERVICES";*)
```

Comme la commande **Executer sur serveur** se comporte comme **Nouveau process** lorsqu'elle est appelée sur le serveur, la même méthode (**START SP SERVICES**) peut être utilisée sur le poste serveur ou sur tout poste client pour démarrer à volonté la méthode **SP SERVICES** en tant que procédure stockée sur le serveur.

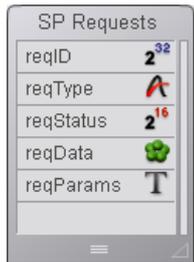
La méthode projet **STOP SP SERVICES** "ordonne" à la méthode projet **SP SERVICES** de s'arrêter.

```
 ` Méthode projet STOP SP SERVICES
  ECRIRE VARIABLE PROCESS (∅vlSPServices;vbStopSPServices;Vrai)
```

La méthode projet **SP SERVICES**, lorsqu'elle démarre, met la variable process *vbStopSPServices* à Faux puis s'exécute en boucle jusqu'à ce que cette variable booléenne devienne Vraie. Grâce à la commande **ECRIRE VARIABLE PROCESS**, tout process utilisateur exécuté sur le serveur ou sur un poste client peut modifier la valeur de la variable *vbStopSPServices*, et donc stopper la procédure quand il le souhaite.

## Communiquer avec la procédure stockée

La procédure stockée doit pouvoir, de manière asynchrone, recevoir des requêtes des clients et y répondre à tout moment et dans n'importe quel ordre. La moyen le plus simple et direct d'assurer cette communication est d'utiliser une table.



SP Requests	
reqID	2 <sup>32</sup>
reqType	A
reqStatus	2 <sup>16</sup>
reqData	(blob)
reqParams	T

La table *[SP Requests]* contient les champs suivants :

- *[SP Requests]reqID* est rempli à l'aide de la commande **Numerotation automatique**. Ce champ identifie chaque requête de manière unique.
- *[SP Requests]reqType* décrit le type de la requête.
- *[SP Requests]reqStatus* peut prendre une des valeurs suivantes :

Valeur	Description
1	la requête a été reçue mais pas encore traitée.
0	la requête a été traitée avec succès.
< 0	la requête a été traitée mais une erreur s'est produite.

**Note** : Ces valeurs ont été choisies arbitrairement pour cet exemple, elles ne sont pas issues de 4D.

- *[SP Requests]reqData* est un BLOB contenant les données de la requête. Il peut contenir les données envoyées par le demandeur ou les données retournées par la procédure stockée au demandeur.
- *[SP Requests]reqParams* contient éventuellement les valeurs de paramètres envoyés par le demandeur à la procédure stockée.

## Pourquoi utiliser une table ?

La communication entre un process client et une procédure stockée peut être établi par l'intermédiaire des commandes **LIRE VARIABLE PROCESS**, **ECRIRE VARIABLE PROCESS** et **VARIABLE VERS VARIABLE**. C'est, par exemple, la solution retenue dans la section **Import basé sur les procédures stockées (exemple)** ainsi que pour la méthode projet **STOP SP SERVICES** listée plus haut.

Ici, le système doit permettre à la procédure stockée de recevoir et de renvoyer des quantités variables de données. On pourrait utiliser des tableaux (y compris des tableaux Texte et Image), mais il existe deux raisons principales pour préférer l'emploi d'une table :

- L'algorithme de gestion des requêtes via des enregistrements est plus simple à créer. Envoyer d'une requête depuis un poste client consiste simplement à ajouter une requête dans la table. Répondre à la requête depuis la procédure stockée consiste simplement à modifier cette requête.
- Comme les requêtes sont stockées dans une table, elles le sont sur le disque. Par conséquent, si la taille d'une requête est importante, cela ne constituera pas un problème dans la mesure où elle peut être supprimée de la mémoire (contrairement aux données stockées dans des tableaux).

## Envoyer une requête depuis le poste client

La méthode projet **Client post request** est une méthode générique pour envoyer une requête :

```
 ` Méthode projet Client post request
  Client post request ( Chaîne { ; Texte } ) -> Entier long
  Client post request ( Type de requête { ; Parametres } ) -> Numéro de requête
  CREER ENREGISTREMENT ([SP Requests])
```

```

[SP Requests]reqID:=Numerotation automatique([SP Requests])
[SP Requests]reqType:=$1
[SP Requests]reqStatus:=1
Si(Nombre de parametres>=2)
    [SP Requests]reqParams:=$2
Fin de si
STOCKER ENREGISTREMENT([SP Requests])
$0:=[SP Requests]reqID

```

La méthode retourne le numéro de la requête, dont l'unicité est garantie par l'utilisation de la commande **Numerotation automatique**. Une fois que cet enregistrement a été ajouté dans la table *[SP Requests]*, le client n'a plus qu'à interroger régulièrement le champ *[SP Requets]redStatus* jusqu'à ce que la procédure stockée ait terminé le traitement de la requête.

## Tester le statut de la requête et récupérer le résultat sur le poste client

La méthode projet **Client get result** est une méthode générique pour tester le statut de la requête. Comme expliqué précédemment, dès que la valeur du champ *[SP Requets]redStatus* devient différente de 1, le client sait que la procédure stockée a traité (avec succès ou non) la requête.

```

` Méthode projet Client get result
` Client get result ( Entier long ; ->BLOB {;Entier long } ) -> Entier long
` Client get result ( Numéro de requête; ->Données {; Durée } ) -> Code d'erreur
C_ENTIER LONG($0;$1;$v1Delay)
$0:=1
$v1Delay:=0
Si(Nombre de parametres>=3)
    $v1Delay:=$3
Fin de si
LECTURE SEULEMENT([SP Requests])
Repeter
    CHERCHER([SP Requests];[SP Requests]reqID=$1)
    Si(Enregistrements trouves([SP Requests])>0)
        Si([SP Requests]reqStatus # 1)
            $2->:=[SP Requests]reqData
            LECTURE ECRITURE([SP Requests])
            Tant que(Enregistrement verrouille([SP Requests]))
                WAITING LOOP($v1Delay)
            CHARGER ENREGISTREMENT([SP Requests])
            Fin tant que
            SUPPRIMER ENREGISTREMENT([SP Requests])
            $0:=[SP Requests]reqStatus
        Fin de si
    Sinon
        ` L'enregistrement de la requête a été perdu !
        ` Cela ne devrait pas se produire. Mais fixons tout de même
        ` le code d'erreur -2 (valeur arbitraire)
        $0:=-2
    Fin de si
    ` La requête n'a pas encore été traitée
    Si($0=1)
        WAITING LOOP($v1Delay)
    Fin de si
Jusque($0 # 1)
LECTURE SEULEMENT([SP Requests])

```

Si la requête a été traitée avec succès par la procédure stockée, la méthode copie le résultat (s'il y en a un) de l'enregistrement vers le BLOB sur lequel un pointeur a été passé en paramètre. La méthode appelante analyse puis exploite les données du BLOB en fonction du type de la requête. Notez que c'est au client de supprimer l'enregistrement de *[SP Requests]* une fois que la requête est terminée.

La petite méthode projet **WAITING LOOP** effectue une boucle pendant un certain nombre de ticks :

```

` Méthode projet WAITING LOOP
` WAITING LOOP ( Entier long )
` WAITING LOOP ( Durée en ticks )
C_ENTIER LONG($1)
$v1StartTicks:=Nombre de ticks
Repeter

```

#### APPELER 4D

```
Jusque ((Nombre de ticks-$vlStartTicks)>=$1)
```

**Rappel :** **ENDORMIR PROCESS** n'a pas d'effet sur le process principal. Si vous utilisez la méthode projet **WAITING LOOP**, le process attendra pendant le temps défini même si la requête émane du process principal d'un poste client.

## La procédure stockée et ses sous-routines

La méthode projet **SP SERVICES** est la méthode exécutée en tant que procédure stockée sur le poste serveur. La structure générale de cette méthode, présentée ci-dessous, est très simple :

```
Initialisation d'une variable "stop"
Repeter
  Recherche des requêtes dont le champ [SP Requests]reqStatus est égal à 1
  Pour chaque requête
    En fonction du type de la requête, appeler une sous-routine
      qui stocke le résultat dans le champ [SP Requests]reqData
    Changer le statut de la requête pour que le client sache ce qui s'est passé
  Fin de boucle
  "Dormir" un peu avant de recommencer
Jusqu'à ce que la variable "stop" devienne vraie
```

Voici le code source réel :

```
` Méthode projet SP SERVICES
` La procédure stockée démarre
vbStopSPServices:=Faux
` La procédure stockée n'a pas besoin d'accès en écriture aux tables...
LECTURE SEULEMENT (*)
` ...sauf pour la table [SP Requests]
LECTURE ECRITURE ([SP Requests])
Repeter
` Recherche des requêtes non encore traitées
CHERCHER ([SP Requests];[SP Requests]reqStatus=1)
` Traitement de ces requêtes les unes après les autres
Boucle ($vlRecord;1;Enregistrements trouvés ([SP Requests]))
` Si l'enregistrement de la requête est verrouillé, attendre son déverrouillage
Tant que (Enregistrement verrouillé ([SP Requests]))
` Attendre une seconde avant d'essayer encore
ENDORMIR PROCESS (Numero du process courant;60)
` Essayer de charger l'enregistrement
CHARGER ENREGISTREMENT ([SP Requests])
Fin tant que
` Supposons que la requête sera traitée avec succès
[SP Requests]reqStatus:=0
Au cas ou
  : ([SP Requests]reqType="Server Information")
    SP DO SERVER INFORMATION
  : ([SP Requests]reqType="Volume List")
    SP DO VOLUME LIST
  : ([SP Requests]reqType="Browse Directory")
    SP DO BROWSE DIRECTORY ([SP Requests]reqParams)
` ...
` D'AUTRES TYPES DE REQUETES PEUVENT ETRE AJOUTES ICI !
` ...
Sinon
` Le type de requête est inconnu, retourner l'erreur -1 (valeur arbitraire)
[SP Requests]reqStatus:=-1
Fin de cas
` Forcer le statut de la requête à être différent de 1
` (pour le cas où une sous-routine lui a donné la valeur 1)
Si ([SP Requests]reqStatus=1)
  [SP Requests]reqStatus:=-3
Fin de si
` Mettre à jour l'enregistrement de la requête
STOCKER ENREGISTREMENT ([SP Requests])
` Passer à la requête non traitée suivante
```

```

ENREGISTREMENT SUIVANT ([SP Requests])
Fin de boucle
` Libérer le dernier enregistrement traité
LIBERER ENREGISTREMENT ([SP Requests])
` Attendre une seconde avant de continuer à répondre aux requêtes
ENDORMIR PROCESS (Numero du process courant; 60)
` Boucle jusqu'à ce qu'on ordonne à la procédure stockée de stopper son exécution
Jusque (vbStopSPServices)

```

La méthode projet **SP SERVICES** peut être utilisée comme modèle pour implémenter de nouveaux services dans une base. Dans ce document, nous détaillons les sous-routines **SP DO SERVER INFORMATION** et **SP DO VOLUME LIST**. La sous-routine **SP DO BROWSE DIRECTORY** (qui par ailleurs accepte comme paramètre le paramètre envoyé par le client dans le champ `[SP Requests]reqParams`) ne sera pas traitée ici.

En fonction du type de la requête, la méthode projet **SP SERVICES** appelle une sous-routine dont le rôle est de stocker les données résultantes dans le champ `[SP Requests]reqData` field. La sauvegarde de l'enregistrement et la modification du statut de la requête sont effectuées par la méthode projet **SP SERVICES**.

Voici la sous-routine **SP DO SERVER INFORMATION**. Elle stocke des informations relatives au serveur dans le BLOB. Une autre méthode projet extraira les données du BLOB en fonction du poste client.

```

` Méthode projet SP DO SERVER INFORMATION
TEXTE VERS BLOB (Version application (*); [SP Requests]reqData; UTF8 Chaîne en C)
TEXTE VERS BLOB (Fichier structure; [SP Requests]reqData; UTF8 Chaîne en C; *)
TEXTE VERS BLOB (Fichier donnees; [SP Requests]reqData; UTF8 Chaîne en C; *)
PROPRIETES PLATE FORME ($vlPlatform; $vlSystem; $vlMachine)
VARIABLE VERS BLOB ($vlPlatform; [SP Requests]reqData; *)
VARIABLE VERS BLOB ($vlSystem; [SP Requests]reqData; *)
VARIABLE VERS BLOB ($vlMachine; [SP Requests]reqData; *)

```

Voici la sous-routine **SP DO VOLUME LIST**. Elle stocke des informations relatives aux volumes dans le BLOB. Une autre méthode projet extraira les données du BLOB en fonction du poste client.

```

` Méthode projet SP DO VOLUME LIST
LISTE DES VOLUMES ($asVName)
$vlSize:=Taille tableau ($asVName)
TABLEAU REEL ($arVSize; $vlSize)
TABLEAU REEL ($arVUsedSpace; $vlSize)
TABLEAU REEL ($arVFreeSpace; $vlSize)
Boucle ($vlElem; 1; $vlSize)
  PROPRIETES DU
VOLUME ($asVName{$vlElem}; $arVSize{$vlElem}; $arVUsedSpace{$vlElem}; $arVFreeSpace{$vlElem})
Fin de boucle
VARIABLE VERS BLOB ($asVName; [SP Requests]reqData)
VARIABLE VERS BLOB ($arVSize; [SP Requests]reqData; *)
VARIABLE VERS BLOB ($arVUsedSpace; [SP Requests]reqData; *)
VARIABLE VERS BLOB ($arVFreeSpace; [SP Requests]reqData; *)

```

## Afficher les informations du serveur sur un poste client

Avec les méthodes projet génériques **Client post request** et **Client get result**, la méthode projet **M\_SERVER\_INFORMATION** affiche, sur le poste client, les informations sur le serveur retournées par la procédure stockée. Cette méthode pourrait être associée à une commande de menu ou appelée, par exemple, depuis la méthode objet d'un bouton :

```

` M_SERVER_INFORMATION
C_BLOB (vxData)
C_ENTIER LONG ($vlReqID; $vlErrCode; $vlOffset)
` Envoi de la requête
$vlReqID:=Client post request ("Server Information")
` Test du statut de la requête et réception du résultat
$vlErrCode:=Client get result ($vlReqID; ->vxData; 60)
` Si la requête est terminée avec succès, affichage du résultat
Si ($vlErrCode=0)
` Extraction de l'information résultante du BLOB
$vlOffset:=0
vsServerVersion:=BLOB vers texte (vxData; UTF8 Chaîne en C; $vlOffset)
vsStructureFile:=BLOB vers texte (vxData; UTF8 Chaîne en C; $vlOffset)
vsDataFile:=BLOB vers texte (vxData; UTF8 Chaîne en C; $vlOffset)

```

```

BLOB VERS VARIABLE (vxData; $vlPlatform; $vlOffset)
BLOB VERS VARIABLE (vxData; $vlSystem; $vlOffset)
BLOB VERS VARIABLE (vxData; $vlMachine; $vlOffset)
\ Analyse des propriétés de la plate-forme
vs4DPlatform:="Version de 4D Server inconnue"
vsSystem:="Version du système inconnue"
vsMachine:="Ordinateur inconnu"
\ ...
\ Ici est placé le code (non listé) qui analyse $vlSystem et $vlMachine
\ ( reportez-vous à l'exemple de la commande PROPRIETES PLATE FORME)
\ ...
\ Affichage de l'information résultante
DIALOGUE ([SP Requests]; "SERVER INFORMATION")
Sinon
ALERTE ("Erreur de requête "+Chaine($vlErrCode))
Fin de si
\ Le BLOB est désormais inutile
EFFACER VARIABLE (vxData)

```

Voici le formulaire [SP Requests];"SERVER INFORMATION" en exécution :

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: Annuler, OK

## Afficher la liste des volumes du serveur sur un poste client

Avec les méthodes projet génériques *Client post request* et *Client get result*, la méthode projet *M\_SERVER\_VOLUMES* affiche, sur le poste client, les informations sur la liste des volumes du serveur retournée par la procédure stockée. Cette méthode pourrait être associée à une commande de menu ou appelée, par exemple, depuis la méthode objet d'un bouton :

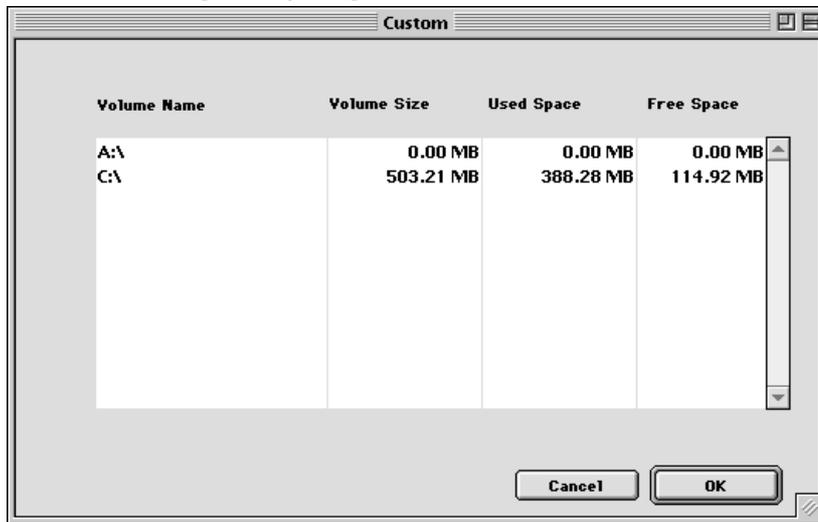
```

\ M_SERVER_VOLUMES
C_BLOB (vxData)
\ Envoi de la requête
$vlReqID:=Client post request("Volume List")
\ Test du statut de la requête et réception du résultat
$vlErrCode:=Client get result($vlReqID;->vxData;120)
\ Si la requête est terminée avec succès, affichage du résultat
Si ($vlErrCode=0)
\ Extraction de l'information résultante du BLOB
$vlOffset:=0
BLOB VERS VARIABLE (vxData; asVName; $vlOffset)
BLOB VERS VARIABLE (vxData; arVSize; $vlOffset)
BLOB VERS VARIABLE (vxData; arVUsedSpace; $vlOffset)
BLOB VERS VARIABLE (vxData; arVFreeSpace; $vlOffset)
Boucle ($vlElem; 1; Taille tableau(arVSize))
\ Conversion d'octets en méga-octets
arVSize{$vlElem}:=arVSize{$vlElem}/1048576
arVUsedSpace{$vlElem}:=arVUsedSpace{$vlElem}/1048576
arVFreeSpace{$vlElem}:=arVFreeSpace{$vlElem}/1048576
Fin de boucle

```

```
` Affichage de l'information résultante
DIALOGUE([SP Requests];"VOLUME LIST")
Si
  ALERTE("Erreur de requête "+Chaine($vlErrCode))
Fin de si
` Le BLOB est désormais inutile
EFFACER VARIABLE(vxData)
```

Voici le formulaire [SP Requests];"VOLUME LIST" en exécution :



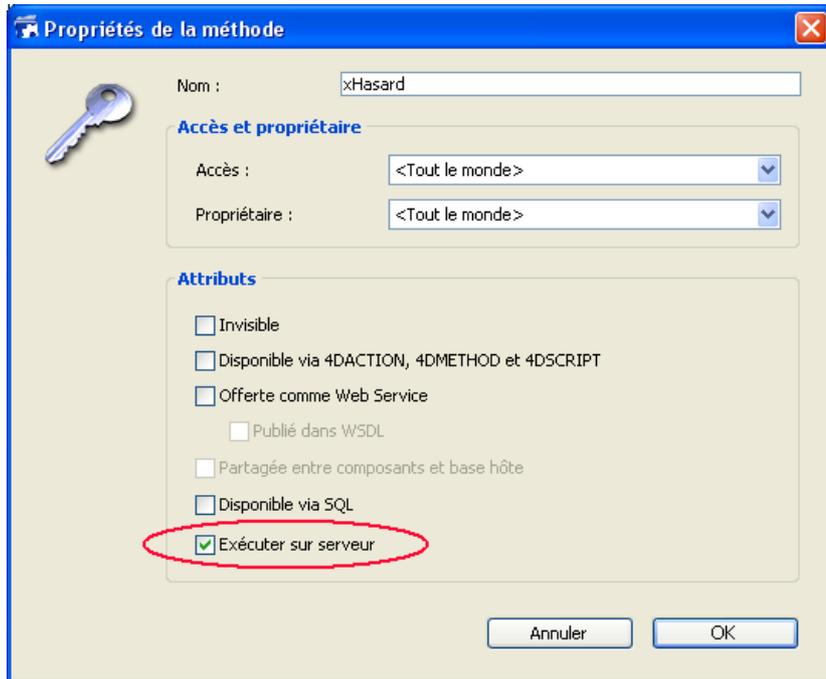
The screenshot shows a dialog box titled "Custom" with a table of volume information. The table has four columns: "Volume Name", "Volume Size", "Used Space", and "Free Space". The data is as follows:

Volume Name	Volume Size	Used Space	Free Space
A:\	0.00 MB	0.00 MB	0.00 MB
C:\	503.21 MB	388.28 MB	114.92 MB

At the bottom of the dialog box, there are "Cancel" and "OK" buttons.

## Attribut Exécuter sur serveur

L'attribut de méthode projet "Exécuter sur serveur" peut être défini dans la boîte de dialogue de modification globale des attributs ou dans la boîte de dialogue des propriétés de la méthode :



Lorsque cette option est cochée, la méthode projet est toujours exécutée sur le serveur, quel que soit le mode d'appel de la méthode.

**Note :** Cet attribut est pris en compte uniquement dans le cadre d'une application 4D exécutée en client/serveur.

## Contexte d'exécution

Lorsque cet attribut est coché, le contexte d'exécution de la méthode projet est le même que celui des triggers (cf. section **4D Server et le langage 4D**) : la méthode sur le serveur partage le même contexte de base de données que le contexte correspondant côté client pour le verrouillage d'enregistrements et les transactions, mais pas le même contexte de langage (variables process, ensembles, sélections courantes).

Tous les paramètres de la méthode (\$1, \$2, etc.) sont envoyés sur le serveur et la valeur du paramètre \$0, s'il est utilisé, est retournée sur le client.

A la différence de la commande **Executer sur serveur**, cette option ne provoque pas de création de process sur le serveur. 4D Server utilise le process "jumeau" du process client qui a demandé l'exécution.

En outre, cette option simplifie le principe de délégation de l'exécution d'une méthode sur le serveur car le transfert des paramètres s'effectue automatiquement dans les deux sens, comme pour un appel de méthode "normal".

La commande **Executer sur serveur**, elle, a un fonctionnement asynchrone et requiert donc davantage de programmation et le recours aux sémaphores pour la lecture des résultats.

## Commandes utilisables

Les méthodes ayant l'attribut "Exécuter sur serveur" sont soumises aux mêmes règles que les procédures stockées en matière d'usage des commandes du langage 4D. L'exécution de certaines commandes est interdite sur le serveur, d'autres sont déconseillées. Pour plus d'informations, reportez-vous au paragraphe "Que ne peut pas faire une procédure stockée (exécutée sur le serveur) ?" dans la section **Procédures stockées**.

## Pointeurs

Si vous passez un pointeur sur une variable (variable simple, tableau ou élément de tableau), la valeur pointée est également envoyée sur le serveur. Si la valeur pointée est modifiée sur le serveur par la méthode, la valeur modifiée est retournée sur le client pour mise à jour de la variable correspondante côté client.

Les pointeurs sur une table ou un champ sont envoyés sous forme de référence (numéro de table, numéro de champ). La valeur de l'enregistrement courant n'est pas échangée automatiquement.

**Note :** L'option fonctionne de la même manière en mode interprété et en mode compilé.

## Exemple

Voici le code la méthode projet **Monappli** ayant l'attribut "Exécuter sur serveur" :

```
C_POINTEUR($1) `Pointeur sur table
C_POINTEUR($2) `Pointeur sur champ
C_POINTEUR($3) `Pointeur sur tableau
C_TEXTE($4) `Valeur à rechercher
C_ENTIER LONG($0) `Résultat

`Rechercher et rapatrier des valeurs pour chaque enregistrement
CHERCHER($1->;$2->=$4)
Tant que(Non(Fin de selection($1->)))
    AJOUTER A TABLEAU($3->;maFormule($1))
    ENREGISTREMENT SUIVANT($1->)
Fin tant que
LIBERER ENREGISTREMENT($1->)
$0:=Enregistrements trouves($1->)
```

Côté client, l'appel de la méthode s'effectue ainsi :

```
TABLEAU TEXTE(monTab;0)
$vlnombre :=MonAppli(->[Table_1] ;->[Table_1]Champ_1 ;->monTab;"à trouver")
```