СИСТЕМА
УПРАВЛЕНИЯ
БАЗАМИ
ДАННЫХ

ЛИНТЕР®

ЛИНТЕР БАСТИОН

ЛИНТЕР СТАНДАРТ

# Управление компонентами СУБД из клиентского приложения

НАУЧНО-ПРОИЗВОДСТВЕННОЕ ПРЕДПРИЯТИЕ

РЕЛЭКС

## Товарные знаки

РЕЛЭКС™, ЛИНТЕР® являются товарными знаками, принадлежащими АО НПП «Реляционные экспертные системы» (далее по тексту – компания РЕЛЭКС). Прочие названия и обозначения продуктов в документе являются товарными знаками их производителей, продавцов или разработчиков.

## Интеллектуальная собственность

Правообладателем продуктов ЛИНТЕР® является компания РЕЛЭКС (1990-2026). Все права защищены.

Данный документ является результатом интеллектуальной деятельности, права на который принадлежат компании РЕЛЭКС.

Все материалы данного документа, а также его части/разделы могут свободно размещаться на любых сетевых ресурсах при условии указания на них источника документа и активных ссылок на сайты компании РЕЛЭКС: relex.ru и linter.ru.

При использовании любого материала из данного документа несетевым/печатным изданием обязательно указание в этом издании источника материала и ссылок на сайты компании РЕЛЭКС: relex.ru и linter.ru.

Цитирование информации из данного документа в средствах массовой информации допускается при обязательном упоминании первоисточника информации и компании РЕЛЭКС.

Любое использование в коммерческих целях информации из данного документа, включая (но не ограничиваясь этим) воспроизведение, передачу, преобразование, сохранение в системе поиска информации, перевод на другой (в том числе компьютерный) язык в какой-либо форме, какими-либо средствами, электронными, механическими, магнитными, оптическими, химическими, ручными или иными, запрещено без предварительного письменного разрешения компании РЕЛЭКС.

## О документе

Материал, содержащийся в данном документе, прошел доскональную проверку, но компания РЕЛЭКС не гарантирует, что документ не содержит ошибок и пропусков, поэтому оставляет за собой право в любое время вносить в документ исправления и изменения, пересматривать и обновлять содержащуюся в нем информацию.

## Контактные данные

394006, Россия, г. Воронеж, ул. Бахметьева, 2Б.

Тел./факс: (473) 2-711-711, 2-778-333.

e-mail: info@linter.ru.

## Техническая поддержка

С целью повышения качества программного продукта ЛИНТЕР и предоставляемых услуг в компании РЕЛЭКС действует автоматизированная система учёта и обработки пользовательских рекламаций. Обо всех обнаруженных недостатках и ошибках в программном продукте и/или документации на него просим сообщать нам в раздел Поддержка на сайте ЛИНТЕР.

# Содержание

**Предисловие** ..... 2
Назначение документа ..... 2
Для кого предназначен документ ..... 2
Необходимые предварительные знания ..... 2
Дополнительные документы ..... 2
**Назначение и условия применения** ..... 3
Назначение ..... 3
Условия применения ..... 3
**Входные данные** ..... 4
Управляющая структура LINDBCtrl ..... 4
**Выходные данные** ..... 5
**Функции инициализации** ..... 6
Инициализация управляющей структуры ..... 6
Инициализация пути к БД ..... 8
Инициализация переменной окружения LINTER_MBX ..... 9
Инициализация пути к временным файлам ..... 11
Деинициализация управляющей структуры ..... 13
**Управление компонентами СУБД ЛИНТЕР** ..... 15
Функция управления ..... 15
Команды управления ..... 15
Создание БД ..... 15
Создание словаря БД ..... 18
Проверка возможности запуска ядра СУБД ЛИНТЕР ..... 21
Запуск ядра СУБД ЛИНТЕР ..... 21
Взять управление активным ядром СУБД ЛИНТЕР ..... 24
Проверка активности ядра СУБД ЛИНТЕР ..... 26
Останов ядра СУБД ЛИНТЕР ..... 28
**Указатель функций** ..... 32
**Указатель команд** ..... 33

**© Управление компонентами СУБД из клиентского приложения. АО НПП «РЕЛЭКС», 1990-2026**

# Предисловие

## Назначение документа

Документ содержит описание функций библиотеки `linctrl`, обеспечивающей управление ядром СУБД ЛИНТЕР в среде ОС Linux, ЗОСРВ Нейтрино из клиентских приложений.

Документ предназначен для СУБД ЛИНТЕР БАСТИОН 6.0 сборка 20.8, далее по тексту СУБД ЛИНТЕР.

## Для кого предназначен документ

Документ предназначен для программистов, разрабатывающих клиентские приложения на основе СУБД ЛИНТЕР с использованием языка программирования C/C++ в среде ОС Linux, ЗОСРВ Нейтрино.

## Необходимые предварительные знания

Для работы с библиотекой необходимо владеть:

- языками программирования C/C++;
- навыками работы с утилитами запуска/останова ядра СУБД ЛИНТЕР и создания базы данных СУБД ЛИНТЕР.

## Дополнительные документы

- Запуск и останов СУБД ЛИНТЕР в среде ОС Linux
- Создание и конфигурирование базы данных
- Сетевые средства
- Интерфейс нижнего уровня
- Библиотеки специальных типов данных


© Управление компонентами СУБД из клиентского приложения. АО НПП «РЕЛЭКС», 1990-2026

# Назначение и условия применения

## Назначение

Библиотека `linctrl` СУБД ЛИНТЕР предназначена для управления в среде ОС Linux, ЗОСРВ Нейтрино компонентами СУБД ЛИНТЕР (утилита создания БД, ядро СУБД и другие компоненты). Средства библиотеки позволяют создавать полностью автоматические (без участия человека) приложения БД для использования в автономных аппаратных комплексах реального времени или мобильных устройствах.

## Условия применения

Клиентское приложение должно быть написано на языке программирования C/C++. Библиотека поставляется как в виде исходного и заголовочного файлов (`linctrl.c` и `linctrl.h` соответственно), так и виде библиотеки `linctrl.a`. Указанные файлы размещаются в подкаталоге `intlib` установочного каталога СУБД ЛИНТЕР.

Для использования функций библиотеки в клиентском приложении следует включить в него заголовочный файл `linctrl.h`. Кроме того, как и для других клиентских приложений СУБД ЛИНТЕР, при трансляции приложения необходимо указывать специальные макросы версии дистрибутива и типа операционной системы. Версия дистрибутива задаётся макросом `_VER_MAX` и должна быть установлена в значение, равное 600 для 6.0.x и т.д. В случае, если этот макрос не указан, будет выдана ошибка трансляции. Кроме указания версии для корректной настройки на типы данных компилятора, надлежит указать ещё и вид операционной системы (ОС). Обычно все необходимые макросы заданы либо в файле `Defs` (в ОС типа Linux, ЗОСРВ Нейтрино), либо в самом компиляторе. Однако может потребоваться явное обозначение макросов ОС. Списки кодов ОС и макросов трансляции приведены в документе «Интерфейс нижнего уровня» приложения 1, 2 соответственно. При использовании в пользовательской программе специальных типов данных (десятичные числа с фиксированной точкой, тип данных «дата-время» и длинные целые числа), неподдерживаемых стандартным компилятором C/C++, необходимо включить в текст программы заголовочные файлы соответствующих библиотек (см. документ Библиотеки специальных типов данных) и подключить к программе в процессе сборки соответствующие библиотеки.

В дистрибутив СУБД ЛИНТЕР входит исходный файл библиотеки, который можно транслировать и собирать вместе с приложением для полного контроля за приложением в устройствах специального назначения.

**© Управление компонентами СУБД из клиентского приложения. АО НПП «РЕЛЭКС», 1990-2026**

# Входные данные

## Управляющая структура LINDBCtrl

Основным элементом входных данных является управляющая структура LINDBCtrl. Указатель на нее является **обязательным** аргументом всех функций библиотеки `linctrl`.

```
typedef struct
{
  char*     DbPath;
  char*     LinterPath;
  char*     LinterMbx;
  char*     TmpPath;
  int       LinterPid;
  L_LONG    Flg;
  L_LONG    OpenFlags;
  L_LONG    RetStatus;
  L_LONG    SysStatus
} t_LINDBCtrl;
```

Перед использованием управляющая структура LINDBCtrl должна быть инициализирована. Значение полям структуры должно присваиваться с помощью вызова специальных функций инициализации.

Поля управляющей структуры:

| **Поле** | **Описание** |
|---|---|
| DbPath | Путь к БД |
| LinterPath | Путь к установочному каталогу СУБД ЛИНТЕР |
| LinterMbx | Значение переменной окружения `LINTER_MBX` |
| TmpPath | Путь к каталогу временных файлов СУБД |
| LinterPid | Pid активного ядра СУБД |
| Flg | Флаги команд управления компонентами СУБД |
| OpenFlags | Для внутреннего использования |
| RetStatus | Библиотечный код завершения функции |
| SysStatus | Системный код завершения функции |

# Выходные данные

Выходными данными является управляющая структура LINDBCtrl, в которой обязательными для заполнения полями являются `RetStatus` и `SysStatus`. Остальные поля заполняются в зависимости от исполняемой функции (описание функций в разделах «Функции инициализации» и «Управление компонентами СУБД ЛИНТЕР»).

# Функции инициализации

## Инициализация управляющей структуры

### Назначение

Инициализация управляющей структуры t_LINDBCtrl.

### Синтаксис

```
L_LONG LINDBCtrlInit(t_LINDBCtrl* LINDBCtrl, char* LinterPath);
```

### Входные данные

Входными данными являются:

1) управляющая структура LINDBCtrl;
2) путь к установочному каталогу СУБД ЛИНТЕР LinterPath. Допустимые значения:
   - строка длиной не более 1024 байт (должна заканчиваться двоичным нулем);
   - NULL-значение.

### Выходные данные

Выходными данными является управляющая структура LINDBCtrl.

В управляющей структуре будут возвращены:

| Имя поля | Значение |
|---|---|
| DbPath | Путь к текущему каталогу |
| LinterPath | Путь к установочному каталогу СУБД ЛИНТЕР |
| LinterMbx | Пустая строка |
| TmpPath | Пустая строка |
| LinterPid | 0 |
| Flg | Флаг LINDB_INIT_FLG (структура инициализирована) |
| OpenFlags | 0 |
| RetStatus | Библиотечный код завершения |
| SysStatus | Системный код завершения |

### Описание

Функция выполняет следующие действия:

1) инициализирует поля структуры LINDBCtrl нулевыми значениями;
2) если аргумент LinterPath содержит строковое значение, проверяется заданный путь на существование;
3) если аргумент LinterPath содержит NULL-значение, производится попытка найти установочный каталог СУБД ЛИНТЕР на основании значения переменной окружения `PATH`. В данном случае для успешного поиска установочный каталог

(предполагается, что это каталог `linter/bin`) должен быть включен в список каталогов переменной окружения `PATH`;

4) в `PATH` путь к каталогу должен обязательно заканчиваться `linter/bin`;

5) проверяет наличие в предполагаемом установочном каталоге (в подкаталоге `bin`) исполняемого файла СУБД ЛИНТЕР;

6) выделяет динамическую память для хранения пути, сохраняет в выделенной памяти путь к дистрибутиву СУБД ЛИНТЕР и заполняет поле LinterPath управляющей структуры значением указателя на выделенную память;

7) выделяет динамическую память для текущего каталога, устанавливает значение текущего каталога в качестве пути к БД (поле DbPath).

Функция должна вызываться перед вызовом любой другой функции библиотеки `linctrl`.

Клиентское приложение может создавать необходимое количество управляющих структур.

### Возвращаемые значения

Функция возвращает:

- `0` – нормальное завершение;
- `-1` – структура LINDBCtrl не инициализирована.

### Библиотечные коды завершения

| Код | Описание |
|---|---|
| E_LINCTRL_ALREADYINIT | Управляющая структура LINDBCtrl уже инициализирована |
| E_LINCTRL_GETPATH | Отсутствие в ОС переменной окружения `PATH` |
| E_LINCTRL_NOTFOUND | Установочный каталог СУБД ЛИНТЕР не прописан в переменной окружения `PATH` |
| E_LINCTRL_NOTDIR | Найденный через переменную окружения `PATH` или заданный в аргументе LinterPath путь не является установочным каталогом СУБД ЛИНТЕР |
| E_LINCTRL_NOMEM | Невозможность выделения динамической памяти для хранения значения путей в структуре LINDBCtrl |

### Пример

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "linctrl.h"
#include "exlib.h"

void  PrintError(t_LINDBCtrl *LINDBCtrl);

void  main()
```

```
{
t_LINDBCtrl LINDBCtrl;
CHAR  LinterPath[]="/usr/linter";
LONGINT Err;
Err=LINDBCtrlInit(&LINDBCtrl,LinterPath);

If (Err != NORMAL)
  PrintError(&LINDBCtrl);
printf("Initialisation LINDBCtrl\n");

printf("End Example\n");
}
```

# Инициализация пути к БД

## Назначение

Сохранение в управляющей структуре LINDBCtrl пути к БД с целью последующего его использования для запуска СУБД.

## Синтаксис

```
L_LONG LINDBCtrlSetDbPath(t_LINDBCtrl* LINDBCtrl, char* DbPath);
```

## Входные данные

Входными данными являются:

1) инициализированная управляющая структура LINDBCtrl;
2) путь к БД. Допустимые значения:
   - строка длиной не более 1024 байт (должна заканчиваться двоичным нулем);
   - NULL-значение.

## Выходные данные

Выходными данными является управляющая структура LINDBCtrl.

В управляющей структуре будут возвращены:

| Имя поля | Значение |
|---|---|
| DbPath | Путь к каталогу БД |
| RetStatus | Библиотечный код завершения |
| SysStatus | 0 |

## Описание

Функция сохраняет в специально выделенной области динамической памяти, указатель на которую размещается в управляющей структуре LINDBCtrl, заданный в аргументе DbPath путь к БД.

Если аргумент DbPath равен NULL, в качестве пути к БД используется текущий каталог.

### Возвращаемые значения

Функция возвращает:

- `0` – нормальное завершение;
- `-1` – путь к БД не инициализирован.

### Коды завершения

| Код | Описание |
|---|---|
| E_LINCTRL_NOINIT | Управляющая структура LINDBCtrl не инициализирована |
| E_LINCTRL_NOMEM | Невозможность выделения динамической памяти для поля DbPath управляющей структуры |

### Пример

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "linctrl.h"
#include "exlib.h"

void  PrintError(t_LINDBCtrl *LINDBCtrl);

void  main()
  {
  t_LINDBCtrl LINDBCtrl;
  CHAR  LinterPath[]="/usr/linter";
  LONGINT Err;
  Err=LINDBCtrlInit(&LINDBCtrl,LinterPath);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Initialisation LINDBCtrl\n");
  Err= LINDBCtrlSetDbPath(&LINDBCtrl,NULL);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Set DBPath\n");
  printf("End Example\n");
  }
```

# Инициализация переменной окружения LINTER_MBX

### Назначение

Сохранение в управляющей структуре LINDBCtrl значения переменной окружения `LINTER_MBX`.

© Управление компонентами СУБД из клиентского приложения. АО НПП «РЕЛЭКС», 1990-2026

### Синтаксис

```
L_LONG LINDBCtrlSetLinterMbx(t_LINDBCtrl* LINDBCtrl, char*
 LinterMbx);
```

### Входные данные

Входными данными являются:

1) инициализированная управляющая структура LINDBCtrl;
2) значение переменной окружения `LINTER_MBX`. Допустимое значение: строка длиной не более 1024 байт (должна заканчиваться двоичным нулем).

### Выходные данные

Выходными данными является управляющая структура LINDBCtrl.

В управляющей структуре будут возвращены:

| Имя поля | Значение |
|---|---|
| LinterMbx | Значение переменной окружения `LINTER_MBX` |
| RetStatus | Библиотечный код завершения |
| SysStatus | 0 |

### Описание

Функция сохраняет в динамической памяти, указатель на которую помещается в поле `LinterMbx` управляющей структуры `LINDBCtrl`, значение почтового ящика для обмена, которое будет использовано при дальнейшей работе. Функция не затрагивает переменную окружения `LINTER_MBX`: не устанавливает её в ОС и не переопределяет ранее установленное в ОС значение.

Переменная окружения `LINTER_MBX` задает имя почтового ящика для обмена ядра СУБД ЛИНТЕР с клиентскими приложениями. Значение переменной окружения `LINTER_MBX` должно быть строковым числом в диапазоне от 1 до 65535 (для ОС Linux) или любым строковым значением, использующимся для пути к файлам, за исключением символа «/» (для ЗОСРВ Нейтрино). Оно не должно совпадать ни с одним из значений для существующих почтовых ящиков или других механизмов обмена в ОС. В случае неустановленного явно данной функцией значения поля `LinterMbx` в дальнейшем используется значение, установленное с помощью переменных окружения ОС или по умолчанию (20561 – для ОС Linux, linter – для ЗОСРВ Нейтрино).

Изменение значения данного поля может быть использовано для запуска нескольких ядер СУБД ЛИНТЕР на одном компьютере (см. документ «Сетевые средства»).

### Возвращаемые значения

Функция возвращает:

- `0` – нормальное завершение;
- `-1` – значение поля `LinterMbx` не инициализировано.

### Коды завершения

| Код | Описание |
|---|---|
| E_LINCTRL_NOINIT | Управляющая структура LINDBCtrl не инициализирована |


© Управление компонентами СУБД из клиентского приложения. АО НПП «РЕЛЭКС», 1990-2026

| Код | Описание |
|---|---|
| E_LINCTRL_NOMEM | Невозможность выделения динамической памяти для поля LinterMbx управляющей структуры |
| E_LINCTRL_INVARG | Недопустимое значение аргумента LinterMbx (NULL-значение) |

### Пример

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "linctrl.h"
#include "exlib.h"

void  PrintError(t_LINDBCtrl *LINDBCtrl);

void  main()
  {
  t_LINDBCtrl LINDBCtrl;
  CHAR  LinterPath[]="/usr/linter";
  CHAR  LinterMbx[]="30000";
  LONGINT Err;
  Err=LINDBCtrlInit(&LINDBCtrl,LinterPath);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Initialisation LINDBCtrl\n");
  Err= LINDBCtrlSetLinterMbx (&LINDBCtrl,LinterMbx);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Set LinterMbx\n");
  printf("End Example\n");
  }
```

# Инициализация пути к временным файлам

### Назначение

Сохранение в управляющей структуре LINDBCtrl пути к каталогу временных файлов СУБД ЛИНТЕР.

### Синтаксис

```
L_LONG LINDBCtrlSetTmpPath(t_LINDBCtrl* LINDBCtrl, char* TmpPath);
```

### Входные данные

Входными данными являются:

1) инициализированная управляющая структура `LINDBCtrl`;
2) путь к каталогу временных файлов. Допустимое значение: строка длиной не более 1024 байт (должна заканчиваться двоичным нулем).

## Выходные данные

Выходными данными является управляющая структура LINDBCtrl.

В управляющей структуре будут возвращены:

| Имя поля | Значение |
|---|---|
| RetStatus | Код завершения запроса к СУБД ЛИНТЕР |
| SysStatus | 0 |
| TmpPath | Путь к каталогу временных файлов |

## Описание

Функция сохраняет в поле TmpPath управляющей структуры `LINDBCtrl` указатель на область памяти, содержащую копию аргумента `TmpPath`.

Каталог временных файлов используется при работе с БД в режиме «только чтение» (запуск ядра СУБД ЛИНТЕР с ключом /RO) (см. документ «Запуск и останов СУБД ЛИНТЕР в среде ОС Linux»).

## Возвращаемые значения

Функция возвращает:

- `0` – нормальное завершение;
- `-1` – путь к каталогу временных файлов не инициализирован.

## Коды завершения

| Код | Описание |
|---|---|
| E_LINCTRL_NOINIT | Управляющая структура LINDBCtrl не инициализирована |
| E_LINCTRL_NOMEM | Невозможность выделения динамической памяти для поля TmpPath управляющей структуры |
| E_LINCTRL_INVARG | Недопустимое значение аргумента TmpPath (NULL-значение) |

## Пример

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "linctrl.h"
#include "exlib.h"

void  PrintError(t_LINDBCtrl *LINDBCtrl);

void  main()
  {
  t_LINDBCtrl LINDBCtrl;
  CHAR  LinterPath[]="/usr/linter";
```


© Управление компонентами СУБД из клиентского приложения. АО НПП «РЕЛЭКС», 1990-2026

```
  CHAR  TmpPath[]="d:/linter/tmp";
  LONGINT Err;
  Err=LINDBCtrlInit(&LINDBCtrl,LinterPath);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Initialisation LINDBCtrl\n");
  Err= LINDBCtrlSetTmpPath(&LINDBCtrl,TmpPath);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Set TmpPath\n");
  printf("End Example\n");
  }
```

# Деинициализация управляющей структуры

### Назначение

Деинициализация управляющей структуры LINDBCtrl.

### Синтаксис

```
L_LONG LINDBCtrlUninit(t_LINDBCtrl* LINDBCtrl);
```

### Входные данные

Входными данными является инициализированная управляющая структура LINDBCtrl.

### Выходные данные

Отсутствуют.

### Описание

Функция освобождает оперативную память, используемую для хранения значений полей управляющей структуры LINDBCtrl, после чего деинициализирует саму управляющую структуру.

Функция должна вызываться перед завершением клиентского приложения или перед повторной инициализацией управляющей структуры LINDBCtrl.

### Возвращаемые значения

Функция всегда возвращает `0`.

### Коды завершения

Отсутствуют.

### Пример

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
#include "linctrl.h"
#include "exlib.h"

void  PrintError(t_LINDBCtrl *LINDBCtrl);

void  main()
  {
  t_LINDBCtrl LINDBCtrl;
  CHAR  LinterPath[]="/usr/linter";
  CHAR  TmpPath[]="d:/linter/tmp";
  LONGINT Err;
  Err=LINDBCtrlInit(&LINDBCtrl,LinterPath);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Initialisation LINDBCtrl\n");
  …
  Err= LINDBCtrlUninit (&LINDBCtrl);
  printf("Uninit\n");
  printf("End Example\n");
  }
```

# Управление компонентами СУБД ЛИНТЕР

## Функция управления

### Назначение

Управление компонентами СУБД ЛИНТЕР.

### Синтаксис

```
L_LONG LinDatabaseControl(t_LINDBCtrl* LINDBCtrl,
  t_LinDatabaseCommand Command, L_LONG Flags, ...);
```

### Входные данные

Входными данными являются:

1) указатель t_LINDBCtrl на управляющую структуру LINDBCtrl;
2) Command – идентификатор команды;
3) флаги Flags, модифицирующие поведение команды; могут быть объединены операцией |.

Функция также может иметь один или два дополнительных аргумента типа char* после аргумента Flags.

Первые два аргумента являются обязательными, хотя некоторые могут быть нулевыми.

### Выходные данные

Выходными данными является управляющая структура LINDBCtrl.

### Описание

Функция выполняет заданную команду в соответствии с флагами и дополнительными аргументами. Подробное описание команд представлено ниже.

### Возвращаемые значения

Функция возвращает:

- `0` – нормальное завершение;
- `-1` – команда не выполнена.

# Команды управления

## Создание БД

### Назначение

Создание БД.

### Синтаксис

```
L_LONG LinDatabaseControl(t_LINDBCtrl* LINDBCtrl,
 LIN_DATABASE_CREATE, L_LONG Flags,char *AddComand);
```

## Входные данные

Входными данными являются:

1) указатель на инициализированную управляющую структура LINDBCtrl;
2) идентификатор `LIN_DATABASE_CREATE` команды «Создание БД»;
3) флаги Flags, модифицирующие поведение команды;
4) необязательный аргумент AddCommand с дополнительными командами создания БД (имеет смысл только одновременно с флагом `LIN_DATABASE_CREATE_GENDB_ADD`). Максимальная длина передаваемой утилите `gendb` полной команды (то есть в сумме с дополнительными командами) не должна превышать 1024 байт.

Допустимые флаги:

| Флаг | Значение |
|---|---|
| LIN_DATABASE_CREATE_PURGE | Предварительная очистка каталога, в котором будет создаваться БД, от файлов предыдущей БД |
| LIN_DATABASE_CREATE_GENDB_ADD | Функция рассматривает аргумент AddCommand как дополнительный набор команд для утилиты создания БД |
| LIN_DATABASE_CREATE_IGNORE | Создание новой БД поверх существующей |

## Выходные данные

Выходными данными является управляющая структура LINDBCtrl.

В управляющей структуре будут возвращены:

| Имя поля | Значение |
|---|---|
| RetStatus | Код завершения |
| SysStatus | Системный код завершения |

## Описание

Для создания БД функция запускает на выполнение утилиту `gendb`, на стандартный вход которой передает команду создания БД (см. документ «Создание и конфигурирование базы данных»).

Выбор каталога для создания БД осуществляется по следующим правилам:

1) если поле DbPath не инициализировано и нет дополнительных указаний о местоположении создаваемой БД в аргументе AddCommand, то БД создается в текущем каталоге;
2) если поле DbPath инициализировано и нет дополнительных указаний о местоположении создаваемой БД в аргументе AddCommand, то БД создается в каталоге, определяемом полем DbPath;
3) если поле DbPath инициализировано, а в аргументе AddCommand явно задано её местоположение, то БД создается в соответствии с параметрами AddCommand;
4) если поле DbPath не инициализировано, но есть дополнительные указания о местоположении создаваемой БД в аргументе AddCommand, то БД создается в соответствии с параметрами AddCommand.

Функция предполагает, что утилита gendb находится в подкаталоге /bin установочного каталога СУБД ЛИНТЕР. Путь к установочному каталогу берется из поля LinterPath управляющей структуры LINDBCtrl.

По умолчанию утилите gendb передается команда 'CREATE DATABASE' без параметров. Дополнительные параметры команды 'CREATE DATABASE' или другие команды утилиты gendb БД можно передать в аргументе AddCommand, при этом должен быть установлен флаг LIN_DATABASE_CREATE_GENDB_ADD, сигнализирующий о наличии дополнительных команд (параметров). В таком случае строка аргумента AddCommand просто добавляется к команде CREATE DATABASE.

Формат символьной строки дополнительных команд:

```
<команда gendb>;<CR>
<команда gendb>;<CR>
...
<команда gendb>;<CR>
```

<CR> – символ перевода строки.

Примеры значений аргумента AddCommand:

1)

```
"name \"Sale\" username \"SysAdm\" maxtab 100 maxcol 750 maxusr
 30;\n"
```

2)

```
";\nset  autoconfig  on;\n set sql users 120 sql columns 900;\n"
```

При установленном флаге LIN_DATABASE_CREATE_IGNORE БД создается даже в том случае, если в каталоге уже существует некоторая БД. Для этого в утилите gendb отключается интерактивный вывод запроса на удаление существующей БД. Неконтролируемое использование этого флага может быть опасным.

## Возвращаемые значения

Функция возвращает:

- 0 – нормальное завершение;
- -1 – БД не создана.

## Коды завершения

| Код | Описание |
|---|---|
| E_LINCTRL_PIPE | Неуспешная попытка создания программного канала ввода-вывода |
| E_LINCTRL_GENDBFAILED | Неудачное завершение работы утилиты gendb. Уточняющий код завершения см. в поле SysStatus управляющей структуры LINDBCtrl |
| E_LINCTRL_WRITE | Неуспешная запись в канал ввода-вывода |
| E_LINCTRL_RUN | Неуспешный запуск утилиты gendb |
| E_LINCTRL_INVARG | Список дополнительных команд слишком длинный (длина аргумента AddCommand больше 1024 байт) |

| Код | Описание |
|---|---|
| E_LINCTRL_NOINIT | Управляющая структура LINDBCtrl не инициализирована |
| E_LINCTRL_OPENDIR | Ошибка открытия каталога БД для получения списка файлов (opendir) |
| E_LINCTRL_REMOVE | Невозможность удаления файла БД |

### Пример

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "inter.h"
#include "exlib.h"

void  PrintError(t_LINDBCtrl *LINDBCtrl);

void  main()
  {
  t_LINDBCtrl LINDBCtrl;
  CHAR  LinterPath[]="/usr/linter";
  CHAR AddCommand[]="\"Продажи\" on DB01;\nset syslog count   3;
\n"
  LONGINT Err;
  Err=LINDBCtrlInit(&LINDBCtrl,LinterPath);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Initialisation LINDBCtrl\n");
  Err=LinDatabaseControl(LINDBCtrl, LIN_DATABASE_CREATE,
  LIN_DATABASE_CREATE_PURGE|LIN_DATABASE_CREATE_GENDB_ADD,
 AddCommand);
  If (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Create DB\n");
  printf("End Example\n");
  }
```

# Создание словаря БД

### Назначение

Создание словаря БД.

### Синтаксис

```
L_LONG LinDatabaseControl(t_LINDBCtrl* LINDBCtrl,
 LIN_DATABASE_DICT, L_LONG Flags, char* UserPassword [,char
 *AddComand]);
```

### Входные данные

Входными данными являются:

1) инициализированная управляющая структура LINDBCtrl;
2) идентификатор LIN_DATABASE_DICT команды «Создание словаря БД»;
3) флаги Flags, модифицирующие поведение команды;
4) UserPassword – регистрационные данные пользователя БД;
5) дополнительные параметры AddCommand для утилит `inl` и `loarel`, а также для самой команды. Максимальная длина передаваемой утилите `inl` строки не должна превышать 1024 байт.

Допустимые флаги:

| Флаг | Значение |
|---|---|
| LIN_DATABASE_DICT_NAME | В строке аргумента AddCommand содержится второй дополнительный параметр с указанием пути к sql-скрипту словаря БД |

### Выходные данные

Выходными данными является управляющая структура LINDBCtrl.

В управляющей структуре будут возвращены:

| Имя поля | Значение |
|---|---|
| RetStatus | Код завершения |
| SysStatus | Системный код завершения |

### Описание

В режиме по умолчанию (аргумент Flags не задан) функция выполняет следующие действия:

1) определяет путь к установочному каталогу СУБД ЛИНТЕР (извлекается из поля LinterPath управляющей структуры LINDBCtrl).
2) запускает утилиту `inl` для обработки sql-скриптов создания словарей БД.

   Для успешного запуска `inl` должны выполняться условия:

   - утилита находится в подкаталоге `/bin` установочного каталога СУБД ЛИНТЕР;
   - имя пользователя БД и его пароль, под которыми запускается утилита `inl`, должны быть заданы в аргументе UserPassword в формате `<Пользователь>/<Пароль>`, например,

```
"Петров"/123
```

   - sql-скрипты для создания словарей БД находятся в подкаталоге `/dict` установочного каталога СУБД ЛИНТЕР.

3) запускает утилиту `loarel` для загрузки данных в некоторые словари.

Для успешного запуска loarel должны выполняться условия:

- утилита находится в подкаталоге /bin установочного каталога СУБД ЛИНТЕР;
- имя пользователя БД и его пароль, под которыми запускается утилита loarel, должны быть заданы в аргументе UserPassword (используются те же значения, что и при запуске утилиты inl);
- файлы загрузки словарей находятся в подкаталоге /dict установочного каталога СУБД ЛИНТЕР.

Флаг LIN_DATABASE_DICT_NAME изменяет используемый по умолчанию алгоритм поиска данных для выполнения команды. Если флаг установлен, то в аргументе AddCommand должна быть задана спецификация местоположения единственного sql-скрипта для исполнения. При этом:

- если в спецификации нет символа «/», то предполагается, что это имя одного конкретного sql-скрипта, который должен быть выполнен для создания (или модификации) таблицы словаря. Этот файл будет искаться в подкаталоге /dict установочного каталога СУБД ЛИНТЕР.

Пример аргументов с установленным флагом LIN_DATABASE_DICT_NAME:

```
LINDBCtrl, LIN_DATABASE_DICT, LIN_DATABASE_DICT_NAME, "SYSTEM/
MANAGER8", "cerrors.sql"
```

- если в спецификации присутствует хотя бы один символ «/», то предполагается, что это полный или относительный путь к sql-скриптам создания словаря БД:

```
LINDBCtrl, LIN_DATABASE_DICT, LIN_DATABASE_DICT_NAME, "SYSTEM/
MANAGER8", "./1.sql"
```

### Возвращаемые значения

Функция возвращает:

- 0 – нормальное завершение;
- -1 – словарь БД не создан.

### Коды завершения

| Код | Описание |
|---|---|
| E_LINCTRL_INL | Неудачное завершение утилиты inl |
| E_LINCTRL_LOAREL | Неудачное завершение утилиты loarel |
| E_LINCTRL_RUN | Неудачный запуск утилиты inl или loarel |
| E_LINCTRL_NOMEM | Невозможность выделения динамической памяти для полей управляющей структуры LINDBCtrl |

### Пример

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "linctrl.h"
```

```
#include "exlib.h"

void  PrintError(t_LINDBCtrl *LINDBCtrl);

void  main()
  {
  t_LINDBCtrl LINDBCtrl;
  CHAR  LinterPath[]="/usr/linter";
  char UserPassword = "SYSTEM/MANAGER8";
  CHAR AddCommand[]="cerrors.sql"
  LONGINT Err;
  Err=LINDBCtrlInit(&LINDBCtrl,LinterPath);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Initialisation LINDBCtrl\n");
  Err= LinDatabaseControl (LINDBCtrl, LIN_DATABASE_DICT,
  LIN_DATABASE_DICT_NAME, USerPAssword, AddCommand);
  If (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Create dictionary\n");
  printf("End Example\n");
  }
```

# Проверка возможности запуска ядра СУБД ЛИНТЕР

В данной версии библиотеки команда проверки возможности запуска ядра СУБД ЛИНТЕР для данной БД (`LIN_DATABASE_RUN_CHECK`) не реализована.

Эмуляция команды может быть выполнена путем попытки запуска и последующего останова ядра СУБД ЛИНТЕР.

# Запуск ядра СУБД ЛИНТЕР

## Назначение

Запуск ядра СУБД ЛИНТЕР.

## Синтаксис

```
L_LONG LinDatabaseControl(t_LINDBCtrl* LINDBCtrl,
 LIN_DATABASE_RUN, L_LONG Flags [,char *AddComand]);
```

## Входные данные

Входными данными являются:

1) инициализированная управляющая структура LINDBCtrl;
2) идентификатор `LIN_DATABASE_RUN` команды «Запуск ядра СУБД ЛИНТЕР»;
3) флаги Flags, модифицирующие поведение команды;

4) необязательный аргумент AddCommand с дополнительными ключами запуска ядра СУБД ЛИНТЕР (задается только одновременно с флагом `LIN_DATABASE_RUN_ARG`). Максимальная длина передаваемой ядру полной команды запуска (то есть в сумме с дополнительными ключами) не должна превышать 1024 байт.

Допустимые флаги:

| Флаг | Значение |
|---|---|
| LIN_DATABASE_RUN_RDONLY | Запуск ядра в режиме «только чтение» (аналог ключа /RO) |
| LIN_DATABASE_RUN_TCP | Запуск одновременно с ядром сетевого агента (аналог ключа /TCP) |
| LIN_DATABASE_RUN_JDBC | Запуск одновременно с ядром сетевого драйвера сервера для JDBC-сервера с номером порта по умолчанию (аналог ключа /JDBCS) |
| LIN_DATABASE_RUN_ARG | Функция содержит аргумент AddCommand с дополнительным набором ключей запуска ядра |

## Выходные данные

Выходными данными является управляющая структура LINDBCtrl.

В управляющей структуре будут возвращены:

| Имя поля | Значение |
|---|---|
| LinterPid | Pid запущенного ядра СУБД ЛИНТЕР |
| RetStatus | Библиотечный код завершения |
| SysStatus | Системный код завершения |

## Описание

Функция определяет рабочую БД и запускает на ней ядро СУБД ЛИНТЕР. Алгоритм выбора каталога рабочей БД аналогичен выбору каталога при создании БД (см. пункт Создание БД).

Ядро запускается из каталога, прописанного в поле LinterPath управляющей структуры LINDBCtrl. Если поле LinterPath не инициализировано, по умолчанию ядро ищется в подкаталоге `/bin` установочного каталога СУБД ЛИНТЕР.

В качестве дополнительного указания о местоположении рабочей БД можно использовать аргумент AddCommand, задаваемый аналогично полю DbPath в управляющей структуре LINDBCtrl.

После запуска Pid запущенного ядра СУБД сохраняется в управляющей структуре LINDBCtrl.

Если флаги не заданы, по умолчанию ядру СУБД ЛИНТЕР передается пустой список ключей. При необходимости дополнительные ключи запуска ядра можно передать в аргументе AddCommand, при этом должен быть установлен флаг LIN_DATABASE_RUN_ARG, сигнализирующий о наличии ключей в дополнительном аргументе.

Формат символьной строки дополнительных команд:

```
<ключ><пробел><ключ><пробел>… <ключ><пробел>
```

Пример значений аргумента AddCommand:

```
"/pool=20000 /pidfile=c:\tmp.txt"
```

Если установлен флаг LIN_DATABASE_RUN_RDONLY, то должно быть инициализировано поле TmpDir управляющей структуры LINDBCtrl. Его значение будет передано ядру СУБД ЛИНТЕР в значении соответствующего ключа. Если поле не инициализировано, ядро СУБД ЛИНТЕР разместит временные файлы в каталоге, определяемом переменной среды окружения LINTER_TMP, а если и она не задана, то по умолчанию в каталоге /tmp/linter.

Если установлен флаг LIN_DATABASE_RUN_TCP, ядру СУБД ЛИНТЕР передается дополнительный ключ /TCP, и оно выполняет запуск сетевого агента (драйвера сервера dbs_tcp) для обеспечения доступа к данной БД удаленным клиентским приложениям.

Если установлен флаг LIN_DATABASE_RUN_JDBC, ядру СУБД ЛИНТЕР передается дополнительный ключ /JDBCS, и оно выполняет запуск JDBC-сервера для обеспечения доступа к данной БД удаленным JAVA-приложениям.

## Возвращаемые значения

Функция возвращает:

- 0 – нормальное завершение;
- -1 – не запущено ядро СУБД ЛИНТЕР. Если задан флаг LIN_DATABASE_RUN_TCP (LIN_DATABASE_RUN_JDBC), то данный код возвращается также при невозможности запустить сетевой драйвер сервера (сетевой драйвер JDBC-сервера).

## Коды завершения

| Код | Описание |
|---|---|
| E_LINCTRL_PIPE | Неуспешная попытка создания программного канала ввода вывода |
| E_LINCTRL_LINFAILED | Неудачный запуск ядра СУБД ЛИНТЕР. Уточняющий код завершения см. в поле SysStatus управляющей структуры LINDBCtrl |
| E_LINCTRL_RUN | Неудачная попытка запуска исполняемого модуля ядра СУБД ЛИНТЕР |
| E_LINCTRL_NOINIT | Управляющая структура LINDBCtrl не инициализирована |
| E_LINCTRL_NOMEM | Невозможность выделения динамической памяти для полей управляющей структуры LINDBCtrl |

## Пример

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "linctrl.h"
#include "exlib.h"
```

```
void  PrintError(t_LINDBCtrl *LINDBCtrl);

void  main()
  {
  t_LINDBCtrl LINDBCtrl;
  CHAR  LinterPath[]="/usr/linter";
  CHAR AddCommand[]="/pool=20000 /pidfile=c:\tmp.txt "
  LONGINT Err;
  Err=LINDBCtrlInit(&LINDBCtrl,LinterPath);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Initialisation LINDBCtrl\n");
  Err= LinDatabaseControl (LINDBCtrl, LIN_DATABASE_RUN,
  LIN_DATABASE_RUN_TCP | LIN_DATABASE_RUN_ARG, AddCommand);
  If (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Run DBMS Linter\n");
  printf("End Example\n");
  }
```

# Взять управление активным ядром СУБД ЛИНТЕР

### Назначение

Взять управление уже запущенным ядром СУБД ЛИНТЕР.

### Синтаксис

```
L_LONG LinDatabaseControl(t_LINDBCtrl* LINDBCtrl,
 t_LinDatabaseCommand Command, L_LONG Flags, …);
```

### Входные данные

Входными данными являются:

1) инициализированная управляющая структура LINDBCtrl;
2) идентификатор `LIN_DATABASE_ATTACH` команды «Запуск ядра СУБД ЛИНТЕР».

В управляющей структуре должны быть установлены:

| Имя поля | Значение |
|---|---|
| LinterPath | Путь к установочному каталогу СУБД ЛИНТЕР |
| LinterMbx | Значение переменной окружения `LINTER_MBX` |

### Выходные данные

Выходными данными является управляющая структура LINDBCtrl.

В управляющей структуре будут возвращены:

| Имя поля | Значение |
|---|---|
| LinterPid | Pid ядра СУБД ЛИНТЕР, взятого под управление |
| RetStatus | Код завершения |
| SysStatus | Системный код завершения |

### Описание

Функция подготавливает структуру LINDBCtrl для управления ранее запущенным ядром СУБД ЛИНТЕР. Ядро могло быть запущено как с помощью средств библиотеки `linctrl`, так и любыми иными средствами (командами ОС, клиентскими приложениями и т.п.).

Управляющая структура LINDBCtrl должна быть предварительно инициализирована (поля LinterPath и LinterMbx). Задаваемый при инициализации путь к установочному каталогу СУБД ЛИНТЕР не обязательно должен соответствовать каталогу, из которого запущено ядро СУБД ЛИНТЕР, над которым предполагается взять контроль. Это может быть корректный путь к любому установочному каталогу, однако в данном случае не исключено возникновение конфликта между запущенным ядром и теми программами СУБД ЛИНТЕР, которые, возможно, будут вызываться из этого каталога функциями библиотеки `linctrl`.

На компьютере могут быть активны несколько ядер СУБД ЛИНТЕР. Под управление будет взято ядро, которое было запущено с переменной окружения `LINTER_MBX`, соответствующей полю LinterMbx управляющей структуры LINDBCtrl. Если это поле не установлено, по умолчанию используется значение 20561. Значение LinterMbx ядра, над которым берется управление, клиентское приложение должно получить извне (из командной строки запуска, из интерактивного запроса, из файла общих ресурсов и т.п.).

Для получения Pid заданного активного ядра функция пытается прочитать сегмент разделяемой памяти ядра, соответствующего переменной окружения `LINTER_MBX`. В дальнейшем управление данным ядром (проверка активности, останов и т.п.) будет выполняться по его Pid, сохраненному в управляющей структуре LINDBCtrl.

### Возвращаемые значения

Функция возвращает:

- `0` – нормальное завершение;
- `-1` – взять под управление заданное ядро не удалось.

### Коды завершения

| Код | Описание |
|---|---|
| E_LINCTRL_SHMGET | Неудачный доступ к сегменту разделяемой памяти ядра |
| E_LINCTRL_NOINIT | Управляющая структура LINDBCtrl не инициализирована |

### Пример

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
#include "linctrl.h"
#include "exlib.h"

void  PrintError(t_LINDBCtrl *LINDBCtrl);

void  main()
  {
  t_LINDBCtrl LINDBCtrl;
  CHAR  LinterPath[]="/usr/linter";
  CHAR LinterMbx[]="20561"
  LONGINT Err;
  Err=LINDBCtrlInit(&LINDBCtrl,LinterPath);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Initialisation LINDBCtrl\n");
  printf("Введите значение LINTER_MBX\n");
  scanf (LinterMbx);
  Err=LINDBCtrlLinterMbx(&LINDBCtrl,LinterMbx);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Set LINTER_MBX\n");
  Err=LinDatabaseControl(LINDBCtrl, LIN_DATABASE_ATTACH, 0);
  If (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Attach DBMS Linter DB\n");
  printf("End Example\n");
  }
```

# Проверка активности ядра СУБД ЛИНТЕР

## Назначение

Проверка активности запущенного ранее ядра СУБД ЛИНТЕР.

## Синтаксис

```
L_LONG LinDatabaseControl(t_LINDBCtrl* LINDBCtrl,
 LIN_DATABASE_ALIVE,L_LONG Flags);
```

## Входные данные

Входными данными являются:

1) инициализированная управляющая структура LINDBCtrl;
2) идентификатор `LIN_DATABASE_ALIVE` команды «Проверка активности ядра СУБД ЛИНТЕР»;
3) флаги Flags, модифицирующие поведение команды.

Допустимые флаги:

| Флаг | Значение |
|---|---|
| LIN_DATABASE_ALIVE_SIG | Проверка активности ядра СУБД с помощью сигнала |

### Выходные данные

Выходными данными является управляющая структура LINDBCtrl.

В управляющей структуре будут возвращены:

| Имя поля | Значение |
|---|---|
| RetStatus | Библиотечный код завершения |
| SysStatus | Системный код завершения |

### Описание

На компьютере могут быть активны несколько ядер СУБД ЛИНТЕР. Функция проверяет активность того ядра, путь к которому прописан в поле LinterPath управляющей структуры LINDBCtrl. Если это поле не инициализировано, фиксируется ошибка.

Под активностью ядра понимается возможность получать данные от процессов ОС и посылать им ответы. Ядро, активное с точки зрения процессора (работает в непрерываемом цикле), не является активным с точки зрения данной функции.

По умолчанию проверка активности запущенного ранее ядра СУБД ЛИНТЕР (или ядра, к которому ранее было выполнено подключение командой `LIN_DATABASE_ATTACH`) выполняется путем запуска программы `chklinter`. Программа `chklinter` должна находиться в подкаталоге `/bin` каталога, заданного в поле LinterPath.

Если установлен флаг `LIN_DATABASE_ALIVE_SIG`, проверка активности ядра будет выполняться путем посылки 0 сигнала.

### Возвращаемые значения

Функция возвращает:

- `0` – нормальное завершение;
- `-1` – активность ядра СУБД не проверена, работа ядра остановлена.

### Коды завершения

| Код | Описание |
|---|---|
| E_LINCTRL_CHKFAILED | Программа `chklinter` не подтвердила активность ядра (неудачный код завершения `chklinter`) |
| E_LINCTRL_KILLFAILED | Неудачная проверка активности ядра с помощью сигнала |
| E_LINCTRL_RUN | Неудачный запуск программы `chklinter` |
| E_LINCTRL_NOMEM | Невозможность выделения динамической памяти для полей управляющей структуры LINDBCtrl |
| E_LINCTRL_LINPID | Pid ядра СУБД ЛИНТЕР неизвестен. Не выполнена команда LIN_DATABASE_ATTACH или LIN_DATABASE_RUN |

### Пример

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "linctrl.h"
#include "exlib.h"

void  PrintError(t_LINDBCtrl *LINDBCtrl);

void  main()
  {
  t_LINDBCtrl LINDBCtrl;
  CHAR  LinterPath[]="/usr/linter";
  LONGINT Err;
  Err=LINDBCtrlInit(&LINDBCtrl,LinterPath);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Initialisation LINDBCtrl\n");
  Err=LinDatabaseControl(LINDBCtrl, LIN_DATABASE_ALIVE,
  LIN_DATABASE_ALIVE_SIG);
  If (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Alive DBMS Linter\n");
  printf("End Example\n");
  }
```

# Останов ядра СУБД ЛИНТЕР

### Назначение

Останов ядра СУБД ЛИНТЕР.

### Синтаксис

```
L_LONG LinDatabaseControl(t_LINDBCtrl* LINDBCtrl,
 t_LinDatabaseCommand Command [,L_LONG Flags [,char *AddComand]]);
```

### Входные данные

Входными данными являются:

1) инициализированная управляющая структура LINDBCtrl;
2) идентификатор `LIN_DATABASE_SHUT` команды «Останов ядра СУБД ЛИНТЕР»;
3) флаги Flags, модифицирующие поведение команды;
4) необязательный аргумент AddCommand, содержащий командную строку для программы `shut`.

Допустимые флаги:

| Флаг | Значение |
|---|---|
| LIN_DATABASE_SHUT_IGNOREROLE | Разрешает выполнять останов «чужих» ядер СУБД ЛИНТЕР |
| LIN_DATABASE_SHUT_FORCE | Останов ядра выполнять с помощью посылки сигнала SIGKILL |
| LIN_DATABASE_SHUT_BY_SHUT_RUN | Останов ядра выполнять с помощью программы `shut` |

### Выходные данные

Выходными данными является управляющая структура LINDBCtrl.

В управляющей структуре будут возвращены:

| Имя поля | Значение |
|---|---|
| RetStatus | Библиотечный код завершения |
| SysStatus | Системный код завершения |

### Описание

В режиме умолчания (аргумент Flags не задан):

- останов ядра СУБД ЛИНТЕР выполняется путем посылки ему сигнала SIGTERM;
- остановлено может быть только ядро, запущенное данным клиентским приложением с использованием библиотеки `linctrl`;
- незавершенные транзакции не откатываются.

Если установлен флаг `LIN_DATABASE_SHUT_IGNOREROLE`, функция выполняет останов любого ядра СУБД ЛИНТЕР, ранее запущенного как с помощью библиотеки `linctrl`, так и с помощью средств ОС. В таком случае используемая в функции управляющая структура LINDBCtrl должна быть предварительно подготовлена с помощью команды `LIN_DATABASE_ATTACH`.

Если установлен флаг `LIN_DATABASE_SHUT_FORCE`, то останов ядра будет производиться путем посылки сигнала SIGKILL (вместо сигнала по умолчанию SIGTERM).

Если установлен флаг `LIN_DATABASE_SHUT_BY_SHUT_RUN`, то останов ядра будет производиться путем вызова программы `shut`, поэтому в аргументе AddCommand должна быть передана строка, содержащая имя пользователя БД и его пароль в БД, разделенные символом «/».

Программа `shut` должна находиться в подкаталоге `/bin` каталога, заданного в поле LinterPath управляющей структуры LINDBCtrl.

После останова ядра СУБД ЛИНТЕР pid уже остановленного ядра продолжает сохраняться в управляющей структуре LINDBCtrl. Это необходимо для отслеживания реального момента останова ядра командой `LIN_DATABASE_ALIVE`.

Для отката всех незаконченных транзакций перед остановом ядра следует задавать одновременно флаги `LIN_DATABASE_SHUT_FORCE` и `LIN_DATABASE_SHUT_BY_SHUT_RUN`.

### Возвращаемые значения

Функция возвращает:

- `0` – нормальное завершение;
- `-1` – ядро не остановлено.

### Коды завершения

| Код | Описание |
|---|---|
| E_LINCTRL_SHUTFAILED | Неудачный запуск программы `shut`. Уточняющий код завершения см. в поле SysStatus управляющей структуры LINDBCtrl |
| E_LINCTRL_KILLFAILED | Неудачная попытка посылки сигнала ядру |
| E_LINCTRL_RUN | Неудачный запуск программы `shut` |
| E_LINCTRL_NOMEM | Невозможность выделения динамической памяти для полей управляющей структуры LINDBCtrl |
| E_LINCTRL_LINPID | Неизвестный Pid останавливаемого ядра СУБД ЛИНТЕР (предварительно не выполнена команда LIN_DATABASE_ATTACH или LIN_DATABASE_RUN) |
| E_LINCTRL_ROLE | Попытка останова «чужого» ядра СУБД ЛИНТЕР без флага LIN_DATABASE_SHUT_IGNOREROLE |

### Пример

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "linctrl.h"
#include "exlib.h"

void  PrintError(t_LINDBCtrl *LINDBCtrl);

void  main()
  {
  t_LINDBCtrl LINDBCtrl;
  CHAR  LinterPath[]="/usr/linter";
  LONGINT Err;
  Err=LINDBCtrlInit(&LINDBCtrl,LinterPath);
  if (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Initialisation LINDBCtrl\n");
  Err=LinDatabaseControl(LINDBCtrl, LIN_DATABASE_SHUT,
  LIN_DATABASE_SHUT_FORCE);
  If (Err != NORMAL)
    PrintError(&LINDBCtrl);
  printf("Shutdown DBMS Linter\n");
  printf("End Example\n");
```

```
}
```

© Управление компонентами СУБД из клиентского приложения. АО НПП «РЕЛЭКС», 1990-2026

# Указатель функций

## L

LinDatabaseControl, 15
LINDBCtrlInit, 6
LINDBCtrlSetDbPath, 8
LINDBCtrlSetLinterMbx, 10
LINDBCtrlSetTmpPath, 11
LINDBCtrlUninit, 13

# Указатель команд

## L

LIN_DATABASE_ALIVE, 26
LIN_DATABASE_ATTACH, 24
LIN_DATABASE_CREATE, 15
LIN_DATABASE_DICT, 18
LIN_DATABASE_RUN, 21
LIN_DATABASE_SHUT, 28

© Управление компонентами СУБД из клиентского приложения. АО НПП «РЕЛЭКС», 1990-2026