СИСТЕМА
УПРАВЛЕНИЯ
БАЗАМИ
ДАННЫХ

ЛИНТЕР®

ЛИНТЕР БАСТИОН

ЛИНТЕР СТАНДАРТ

Запуск и останов СУБД ЛИНТЕР в среде ОС Windows

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

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

Назначение документа

Документ содержит описание процедуры запуска и останова СУБД ЛИНТЕР в среде ОС Windows.

Документ предназначен для СУБД ЛИНТЕР СТАНДАРТ 6.0 сборка 20.2, далее по тексту СУБД ЛИНТЕР.

Для кого предназначен документ

Документ предназначен для системных администраторов и пользователей СУБД ЛИНТЕР.

Необходимые предварительные знания

- Ознакомиться с первой главой документации «Архитектура СУБД».
- Иметь навыки работы и администрирования в соответствующей ОС.

Дополнительные документы

- Архитектура СУБД
- Сетевой администратор
- Установка СУБД ЛИНТЕР в среде ОС Windows
- Администрирование комплекса средств защиты данных
- Сетевые средства
- Создание и конфигурирование базы данных
- Справочник по SQL
- Процедурный язык
- Интерфейс нижнего уровня
- Миграция базы данных
- Конвертер баз данных
- Тестирование базы данных
- Архивирование и восстановление базы данных

Запуск СУБД ЛИНТЕР

Запуск СУБД ЛИНТЕР активизирует ядро СУБД, которое отвечает за управление данными во внешней памяти, управление буферами оперативной памяти, управление транзакциями и журнализацию. Кроме того, ядро включает в себя такие процессы, как SQL-транслятор, транслятор хранимых процедур и триггеров, процессор сортировки. Ядро является основной резидентной частью и основной составляющей серверной части СУБД. При запуске ядра СУБД ЛИНТЕР может использовать переменные среды окружения (см. подраздел Переменные среды окружения).

Режимы запуска СУБД ЛИНТЕР

В ОС семейства Windows NT ядро СУБД может быть запущено как приложение ОС или как сервис ОС.

Запуск ядра СУБД ЛИНТЕР как сервиса позволяет:

- запускать СУБД автоматически при старте операционной системы;
- работать СУБД в произвольном контексте безопасности, а не в контексте безопасности пользователя, зарегистрированного в данный момент в операционной системе.

Консольный запуск ядра описан в пункте Запуск как приложения ОС. Запуск СУБД ЛИНТЕР как сервиса ОС кратко описан в пункте Запуск как сервиса ОС. Более подробную информацию можно получить в документе «Сетевой администратор», раздел «Управление БД».

Режим запуска ядра СУБД ЛИНТЕР можно задать при установке СУБД ЛИНТЕР:

- если была выбрана опция «Запустить ядро СУБД ЛИНТЕР на демонстрационной базе данных», то сразу после установки ядро будет запущено локально на «Демонстрационной БД» либо как приложение ОС, либо как сервис ОС (если при установке была выбрана опция «Службы»);
- если была выбрана опция «Автоматически запускать ядро СУБД ЛИНТЕР после старта машины», то каждый раз после загрузки операционной системы будет запускаться ядро СУБД ЛИНТЕР на Демонстрационной БД как сервис ОС.

Примечание
Подробно особенности установки СУБД ЛИНТЕР описаны в документе «Установка СУБД ЛИНТЕР в среде ОС Windows», раздел «Проверка установки».

Запуск как приложения ОС

Запустить ядро СУБД как приложение можно одним из способов:

1) запустить исполняемый файл:
   - для 64-разрядной версии ядра СУБД:

```
linter64 [/<параметр> …]
linter64 [-<параметр> …]
```

   - для 32-разрядной версии ядра СУБД:

```
linternt [/<параметр> …]
linternt [-<параметр> …]
```

При этом каталог /bin должен быть включен в список каталогов переменной окружения PATH, иначе для запуска нужно использовать полный абсолютный или относительный путь к исполняемому файлу.

Указать путь к БД одним из следующих способов:

а) указать путь в команде запуска ядра СУБД ЛИНТЕР в ключе /BASE (наивысший приоритет);

б) установить переменную окружения SY00 так, чтобы ее значение указывало на каталог, в котором находится БД;

в) при отсутствии значений /BASE и SY00 будет выполнена попытка считывания значения пути к БД из параметров предыдущего запуска ядра СУБД, которое хранится в реестре ОС;

г) выбрать путь к БД в диалоговом окне, которое будет выдано в том случае, если СУБД не обнаружит БД.

2) выбрать пункт меню «СУБД ЛИНТЕР» (название пункта меню может быть другим, указанным при установке СУБД) из программной группы, которая была указана при установке. Например,

Пуск=>Программы=>СУБД ЛИНТЕР=>СУБД ЛИНТЕР

При успешном запуске на консоль выводятся параметры и установленные режимы работы ядра (рис. 1):

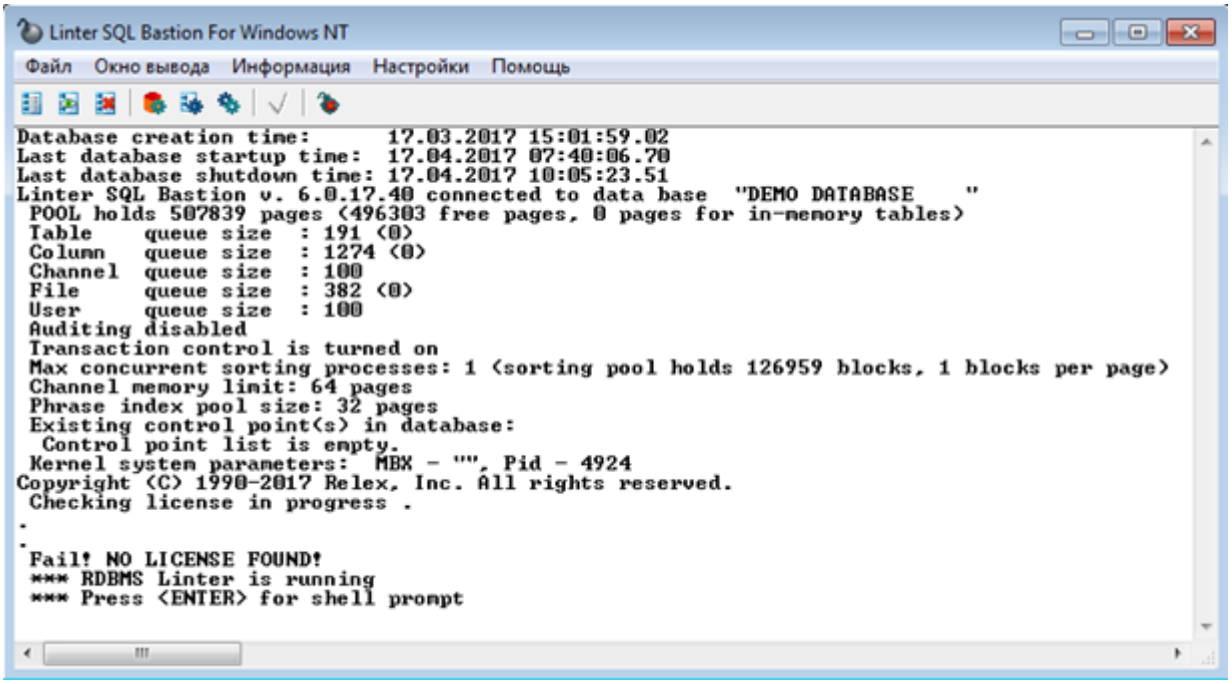

Рисунок 1. Окно параметров и установленных режимов работы ядра

Если ядро не смогло запуститься, то будет выдано сообщение об ошибке.

Примечание
Если при запуске ядра СУБД будет получено сообщение `error 5, can't lock sysrel index file`, значит ядро СУБД уже запущено.

Запуск как сервиса ОС

Для управления сервисами в состав СУБД ЛИНТЕР входит утилита «`Сетевой администратор`» (`linadm`, см. документ «Сетевой администратор», раздел «Запуск утилиты»). Запуск утилиты необходимо выполнить от имени администратора одним из способов:

1) выполнить **Пуск=>Программы=>СУБД ЛИНТЕР** (программная группа, которая была указана при установке СУБД) и в контекстном меню ярлыка **Администратор СУБД ЛИНТЕР** выбрать пункт **Запуск от имени администратора**;
2) в подкаталоге bin установочного каталога СУБД в контекстном меню исполняемого файла `linadm.exe` выбрать пункт **Запуск от имени администратора**.

Примечание
Подробно работа с утилитой описана в документе «Сетевой администратор».

Для запуска СУБД ЛИНТЕР как сервиса:

1) открыть список серверов и выделить в нем сервер БД, который нужно запустить;
2) щелкнув правой кнопкой мыши, выбрать в выпадающем меню пункт **Стартовать**. Появится диалоговое окно изменения параметров запуска БД. Если необходимо, измените значения параметров;
3) нажать кнопку **ОК**. Ввести имя и пароль создателя БД или пользователя БД с привилегией DBA.

Если «высветился» **зеленый** сигнал пиктограммы светофора, значит, локальный запуск ядра СУБД как сервиса ОС прошел успешно.

Если ядро СУБД не запустилось, то сигнал пиктограммы светофора остается красным и выдается диагностическое сообщение с кодом завершения.

Запуск СУБД с поддержкой Kerberos

Для запуска СУБД с поддержкой идентификации и аутентификации по Kerberos-протоколу необходимы следующие условия:

- ядро СУБД ЛИНТЕР должно поддерживать идентификацию и аутентификацию по Kerberos-протоколу;
- должна быть определена переменная окружения LINTER_KRB_SERVICE, задающая имя службы СУБД ЛИНТЕР. Значение переменной должно соответствовать имени сервиса СУБД ЛИНТЕР в терминах Kerberos-сервера. Это значение передается клиентскому приложению и используется как имя сервиса, для которого необходимо выполнить аутентификацию (приложение 1).

  Имя сервиса прописывается в виде linter@server.domain либо linter/server.domain, где linter – имя службы, server.domain – доменное имя сервера, на котором работает ядро СУБД ЛИНТЕР.

Добавлена возможность задать имя ЛИНТЕР-сервиса для kerberos через ключ /KRBSRVC, аналог переменной окружения LINTER_KRB_SERVICE, но с более высоким приоритетом. Основное назначение – запуск ядра СУБД ЛИНТЕР с этой опцией как службы в среде ОС Windows.

Сетевой режим работы

Для настройки работы СУБД ЛИНТЕР в сетевом режиме кроме запуска ядра СУБД необходимо запустить сетевые средства: сетевой драйвер сервера и сетевой драйвер клиента. Порядок запуска компонентов не имеет значения. Можно запускать в последовательности

```
ядро СУБД->драйвер сервера->драйвер клиента
```

либо

```
драйвер клиента->драйвер сервера->ядро СУБД
```

Драйвера, так же как и ядро СУБД, могут быть запущены как приложение ОС или как сервисы ОС.

Чтобы разрешить доступ клиентских приложений к удаленной БД, необходимо на ЛИНТЕР-сервере, где размещается эта БД, в режиме локального доступа к ней выполнить команду:

```
grant access on unlisted station to all;
```

(разрешить доступ к этой БД со всех компьютеров), либо команду на создание станции (разрешить доступ к этой БД только с конкретных компьютеров) (см. документ «Администрирование комплекса средств защиты данных», подраздел «Контроль доступа к БД с сетевых станций»).

Драйвер сервера

Драйвер сервера предназначен для обслуживания запросов удаленных клиентов на сервере базы данных.

Запустить сетевой драйвер сервера можно одним из способов:

- с помощью утилиты «Сетевой администратор» (см. документ «Сетевой администратор», подпункт «Запуск сетевого драйвера сервера»). При этом сетевой сервер будет запущен как служба ОС;
- указать при запуске ядра СУБД ключ `-tcp[=<номер порта>]` (по умолчанию номер порта 1060);
- выполнить из подкаталога /bin установочного каталога СУБД команду:

```
dbs_tcp -P=<номер порта>
```

По умолчанию номер порта 1060.

При этом сетевой сервер будет запущен как приложение ОС.

Драйвер клиента

Драйвер клиента предназначен для обслуживания запросов локальных клиентов и серверов на локальном сервере БД. Он должен запускаться на том компьютере, на

котором будет функционировать приложение. Драйвер можно запускать при активном или неактивном состоянии СУБД, но обязательно до запуска приложения. Перед запуском драйвера клиента необходимо внести в файл сетевой конфигурации `nodetab`, расположенный в подкаталоге bin установочного каталога СУБД, данные о серверах с которыми планируется работа (см. «Сетевые средства», пункт «Файл сетевой конфигурации клиента»).

Запустить сетевой драйвер клиента можно одним из способов:

- с помощью утилиты «Сетевой администратор» (см. документ «Сетевой администратор», подпункт «Запуск сетевого драйвера клиента»).

  При этом сетевой клиент будет запущен как служба ОС;
- выполнить из подкаталога /bin установочного каталога СУБД команду:

```
dbc_tcp -s=<имя ЛИНТЕР-сервера>
```

При этом сетевой драйвер клиента будет запущен как приложение ОС.

Ключи доступа к БД

```
/U=<имя>/<пароль>
```

Задает регистрационные данные пользователя БД. При старте ядра СУБД будет проверяться наличие данного пользователя. Если пользователь является создателем БД или зарегистрирован в БД с привилегией DBA, то ядро СУБД будет запущено. В противном случае происходит отказ в запуске ядра СУБД.

```
/BASE=<строка>
```

Задает полный путь к БД. Аналог переменной окружения SY00.

```
/CF=<спецификация файла>
```

Примечание
Поддерживается со сборки 6.0.17.92.

Задает имя и путь к текстовому файлу, содержащему ключи командной строки запуска ядра СУБД ЛИНТЕР.

Характеристика файла:

- файл может содержать несколько строк;
- строки, начинающиеся с символа #, считаются комментарием и не обрабатываются;
- считывание ключей из файла выполняется слева направо;
- если файл содержит одинаковые ключи или в командной строке заданы дополнительно отдельно от файла совпадающие ключи, то приоритет имеет самый последний обработанный ключ.

Возможные ошибочные ситуации при обработке ключа:

1) если файл не может быть прочитан, то ядро СУБД не запускается и выдается соответствующее диагностическое сообщение;

2) если строка в файле длиннее 4096 символов, то ядро СУБД не запускается и выдается соответствующее диагностическое сообщение;

3) если внутри файла встречается ключ `/CF`, то он игнорируется, ядро СУБД запускается с выдачей соответствующего предупреждения;

4) ключ не распознан (как в файле, так и в командной строке). Ядро СУБД запускается с распознанными ключами и выдается диагностическое сообщение длиной не более 256 символов со списком нераспознанных ключей. Если нераспознанных ключей на X больше, чем может уместиться в диагностическом сообщении, то добавляется текст «+X more». Если не умещается ни одного ключа, то выводится только часть первого ключа с символом ~ в конце;

5) ошибка при выделении памяти для загрузки файла.

С ошибками 1) и 2) выводится спецификация файла, вызвавшего ошибку (может быть выведено максимум 256 символов, остальные символы заменяются знаком ~).

Примеры.

```
linternt.exe /CF=c:\start_lin60.txt
Содержимое файла start_lin60.txt:
/pool=2000 /local /nooutfile
/tcorrect
/procprint

linternt.exe /CF=c:\start_lin60.txt /tracelog
```

Ключи доступа к защищенной БД

`/PASS=<пароль>`

Задает пароль доступа к защищенной БД.

Примечание
В некоторых ОС для символа $ требуется экранирование при указании в командной строке.

Пример

```
linternt.exe /local /pass=$gost$123456
```

`/SETPASS`

Задает интерактивный ввод пароля доступа к защищенной БД. Пароль доступа надо будет ввести в предложенном диалоговом окне или в строке ввода.

Пример

```
linternt.exe /local /setpass
```

`/PASSFILE=<спецификация файла>`

Задает имя и путь к текстовому файлу, содержащему пароль доступа к защищенной БД.

Пример

```
linternt.exe /local /passfile=d:\pass.txt
```

Содержимое файла `pass.txt` – строка: `$gost$123456`

Ключи управления оперативной памятью

`/POOL=<размер>`

Задает размер пула памяти ядра СУБД в страницах по 4 Кбайт. В пуле размещаются все очереди ядра: очередь файлов, очередь таблиц, очередь столбцов и т.д.

По умолчанию установлен минимальный размер пула ядра: 20000 страниц.

Примечания

1. Если ядро СУБД не может зарезервировать память указанного размера, то оно не стартует и на консоль будет выдано сообщение типа «`ERROR: no memory for pool NNN`», где NNN – заданный в ключе размер пула в страницах. Дополнительно выводится всплывающее диалоговое окно с предложением изменить указанный параметр запуска ядра СУБД.
2. Значение ключа имеет больший приоритет по сравнению со значением параметра AUTOCONFIG ON (см. документ «Создание и конфигурирование базы данных», пункт «Конфигурирование БД»).

`/SPOOL=<размер>`

Задает размер пула памяти одного процесса сортировки в страницах. Размер страницы по умолчанию 4 Кбайт, но его можно изменить с помощью команды ALTER DATABASE SET RECORD SIZE LIMIT (см. документ «Справочник по SQL», пункт «Ограничение длины записи»). В пуле процесса сортировки хранятся промежуточные результаты сортировки выборок в случае отсутствующих индексов или сложных запросов.

По умолчанию установлен минимальный размер пула сортировки: 4000 страниц.

Примечания

1. Рекомендуемое соотношение между параметрами pool и spool 4 к 1. Подробнее о значениях параметров, влияющих на эффективность работы ядра СУБД ЛИНТЕР, рассказано в документе «Архитектура СУБД», подраздел «Распределение оперативной памяти».
2. Значение ключа имеет больший приоритет по сравнению со значением параметра AUTOCONFIG ON (см. документ «Создание и конфигурирование базы данных», пункт «Конфигурирование БД»).

`/PPOOL=<размер>`

Задает размер пула подсистемы фразового поиска СУБД в страницах по 4 Кбайт.

По умолчанию – 2000 страницы, минимум – 400 страницы. Максимальное значение – 524287 страниц.

`/PCONTCACHE=<размер>`

Задает размер кэша в страницах по 4 Кбайт для контейнера, используемого при создании/модификации фразового индекса (не влияет на скорость поисковых операций с использованием фразовых индексов).

По умолчанию 1000.

Если задано 0, используется значение по умолчанию.

`/PBVCACHE=<размер>`

Задает размер кэша в страницах по 4 Кбайт для бит-вектора, используемого при создании/модификации фразового индекса (не влияет на скорость поисковых операций с использованием фразовых индексов).

По умолчанию 1000.

Если задано 0, используется значение по умолчанию.

`/INMEMPOOL=<размер>`

Задает максимально допустимое количество страниц в пуле ядра СУБД ЛИНТЕР, выделяемых для размещения таблиц «в памяти». Если этот ключ не задан, то использование таблиц «в памяти» запрещено.

Примечания

1. Страницы для размещения таблиц «в памяти» выделяются из пула памяти ядра СУБД (см. описание ключа /POOL).
2. Если размер пула страниц минус число страниц "в памяти" меньше 5000 страниц, то размер пула страниц будет увеличен на требуемое значение с выдачей сообщения на консоль и в файл `linter.out`.

`/LOCK`

Задает блокирование выделенной для работы ядра СУБД оперативной памяти (под очереди системных объектов, пул страниц и т.п.) на уровне ОС. В случае если память заблокировать невозможно, ядро СУБД не запускается. В этом режиме за счёт отсутствия вытеснения памяти ядра СУБД другими программами ускоряется его работа. Пользоваться этой возможностью нужно с осторожностью, чтобы не захватить практически всю физическую память и, таким образом, не повлиять отрицательно на общую производительность СУБД.

Ключи управления функционированием

`/KILL=<время>`

Задает временной промежуток (в секундах), через который проверяется «живучесть» программ-клиентов. Если через указанный промежуток времени обнаруживается, что какой-либо из клиентов закончил работу и не известил об этом ЛИНТЕР, то ядро автоматически освободит относящиеся к этому клиенту ресурсы.

По умолчанию 120 секунд, минимум 10 секунд.

/HIDE

Заставляет сворачивать консольное окно ядра в окошко SysTray (ядро должно быть запущено как приложение операционной системы).

/WLHB

Наличие этого ключа при запуске ядра СУБД ЛИНТЕР позволяет завершить сохранение системного журнала утилитой LHB при получении ядром команды останова.

Например, пусть ядро СУБД ЛИНТЕР запущено с ключом /WLHB. Если утилита архивирования баз данных (LHB) выполняет операцию сохранения БД в режиме WAIT, и в это время приходит команда на останов ядра, то ядро не завершит работу до тех пор:

1) пока утилита LHB не закончит сохранение системного журнала;
2) пока не наступит тайм-аут, определенный для утилиты LHB (т.е. прошло более 10 секунд после предыдущей команды от LHB);
3) пока не прошло 3000 секунд с момента запроса на останов ядра.

Если ядро запущено без ключа /WLHB, то останов ядра произойдет сразу, и архив не будет сформирован полностью.

/DEFCOMM[={ACK|ALL}[, …]]

Примечание
Поддерживается со сборки 6.0.17.95.

Ключ используется для работы с системой резервирования (server) и системой архивирования БД (lhb).

Возможна ситуация, когда клиент уже получил ответ на операцию commit, но, поскольку lhb работает в асинхронном режиме, эти данные не были сохранены в архив. В этом случае при отказе ЛИНТЕР-сервера невозможно будет восстановить данные, которые уже зафиксированы в БД. Это замечание относится, как к lhb, так и к server.

Для исключения такой ситуации вводится особый режим отложенного commit. Он заключается в том, что ответ клиенту на commit посылается не сразу, а только после того, как данные будут отосланы lhb. Таким образом, гарантируется сохранение данных lhb до получения ответа на запрос изменения БД клиентом.

Такое поведение замедляет работу клиента, особенно в случае работы нескольких lhb одновременно. Поэтому в зависимости от требований к быстродействию и надежности приложения могут применяться следующие варианты отложенного commit:

- DEFCOMM: отсылка ответа клиенту осуществляется непосредственно после отсылки данных одному из lhb. Подтверждение приема данных lhb не осуществляется. Данный режим используется по умолчанию;
- DEFCOMM=ACK: отсылка ответа клиенту осуществляется по приему подтверждения, что данные приняты от одного из lhb;
- DEFCOMM=ALL: отсылка ответа клиенту осуществляется при отсылке данных всем lhb;
- DEFCOM=ACK,ALL: отсылка ответа осуществляется при получении подтверждения о сохранении данных от всех lhb.

Примечание
Данный режим распространяется только на работающие в wait-режиме lhb.

Режим отложенного ответа включается не сразу по запуску lhb, а только после того момента, когда будет скачана вся БД.

Режим отключается при отсоединении всех процессов lhb, работающих в wait-режиме.

В режиме `TRUETYPECOMMIT OFF` (см. документ «Создание и конфигурирование базы данных», пункт «Конфигурирование БД») ответ будет задержан только в случае сохранения системного журнала СУБД на диск. Если в результате запроса сохранение на диск журнала не производилось, то и задержки ответа не будет.

`/IGNERROR`

Заставляет ядро СУБД игнорировать ошибки восстановления по системному журналу БД.

По умолчанию ядро СУБД во время восстановления БД (например, после отказа компьютера или из-за ошибки в самом ядре СУБД) при обнаружении ошибки прекращает дальнейшее восстановление БД. При задании ключа процесс восстановления прерываться не будет. Это позволяет запустить потом утилиту `testdb` (см. документ «Тестирование базы данных», раздел «Выполнение программы».)

`/RAPID`

Задает режим «догона» системного журнала, развернутого из архивного файла БД. Используется при запуске СУБД ЛИНТЕР в системе резервирования.

`/LOCAL`

Задает режим запуска ядра СУБД ЛИНТЕР в виде обычного приложения (не сервиса) операционной системы. В окне приложения выводится информация о параметрах, с которыми запущено ядро (путь к БД, размеры очередей, размеры пулов ядра и тайм-ауты). Существует возможность изменить эти параметры и сохранить изменения (т.е. последующий запуск ядра будет происходить уже с измененными параметрами), включить/отключить логирование и т.п.

Если запуск осуществляется без ключа /LOCAL, то ядро ЛИНТЕР сначала пытается стартовать как сервис ОС, и только, если это не удается, оно запускается как приложение ОС.

`/TMPDIR=<строка>`

Определяет местоположение (каталог) для размещения временных файлов.

`/[NO]JEXIT`

Ключ `/JEXIT` задает режим автоматического завершения работы ядра СУБД ЛИНТЕР в случае невозможности продолжения записи в системный журнал. Если ядро запущено с ключом `/NOJEXIT`, то работа ядра будет продолжаться, но без ведения системного журнала.

Примечание
В системный журнал заносится информация обо всех изменениях в БД. В случае аварийного завершения работы СУБД ЛИНТЕР при следующем запуске ядро определит

(по журналу) наличие прерванных транзакций и аннулирует все сделанные изменения. Если работа ведется без системного журнала, то СУБД ЛИНТЕР не сможет восстановить физическую и логическую целостность базы данных после аварии.

По умолчанию `/JEXIT`.

`/TCORRECT`

Заставляет ядро СУБД ЛИНТЕР игнорировать тот факт, что дата последнего запуска СУБД является «будущей» по сравнению с текущей датой ОС. Действует по умолчанию.

`/NOLARGE`

При запуске ядра СУБД определяет, поддерживает ли операционная система длинные файлы (это файлы, размер которых больше 2 Гбайт). Если не поддерживает, то запрещается стартовать на БД, содержащей такие файлы. Для исключения возникновения подобной ситуации необходимо задавать параметр /NOLARGE, который запрещает создание таблиц размером больше 2 Гбайт.

Если ядро СУБД запускается без ключа `/NOLARGE`, а ОС не поддерживает длинные файлы, то ядро запустится, но не будет работать с таблицами, которые расположены в длинных файлах. При попытке обратиться к таким таблицам будет выдано соответствующее сообщение.

Если ядро СУБД запускается с ключом `/NOLARGE`, и в БД присутствуют файлы размером больше 2 Гбайт, то ядро не будет запущено. При попытке расширить файл до размеров больше 2 Гбайт будет возвращен код, сигнализирующий о невозможности это сделать.

Запуск с ключом `/NOLARGE` позволяет пользователю создавать БД и работать с ней какое-то время на ОС, поддерживающей длинные файлы, а затем перенести БД на ОС, не поддерживающую длинные файлы.

`/RO`

Задает работу с БД в режиме «только чтение» (модификация БД в этом режиме невозможна). При запуске СУБД с этим ключом доступны следующие команды:

SELECT	ALTER EVENT
EXECUTE PROCEDURE	SET EVENT
TEST TABLE	CLEAR EVENT
LOCK TABLE	SET LOG
UNLOCK TABLE	SET NAMES
CREATE EVENT	SET SESSION BLOB LOG
DROP EVENT	SET SESSION BLOB GEOMETRY STORAGE {OFF|ON}
GET EVENT	SET SESSION QUANTUM
WAIT EVENT	SET COMPATIBILITY

При работе в режиме `read only` в каталоге временных файлов создается ряд файлов (файлы логирования, трассировки, временные рабочие файлы): `linter.out`, `LINTER.LOG`, `lintrace.log`, `phrase.idx`, `1.31`, `1.41`, `1.51`.

При завершении работы СУБД ЛИНТЕР перечисленные выше файлы удаляются.

Каталог временных файлов определяется следующим образом (по убыванию приоритета):

1) может быть задан в явном виде в параметрах запуска ядра с помощью ключа TMPDIR, например,

```
linternt.exe /local /base=D:\database /RO /TMPDIR=D:\TMP
```

2) путь к каталогу временных файлов берется из переменной среды окружения TEMP.

`/DEBUG`

Для процесса ядра СУБД создаётся консольное окно.

Примечание
Если ядро запущено с ключом /DEBUG, то при его останове оно может быть завершено некорректно и/или не дождавшись lhb, если на это потребуется больше времени, отведенного ОС на завершение приложения (около 10 сек). Поэтому такой режим применять не рекомендуется (а если применять, то иметь в виду этот момент).

`/C`

или

`/CONSOLE`

Запрещает создание окна консоли ядра СУБД и прочих диалоговых окон. Работа аналогично ключу /debug за исключением:

- в случае, если не найдена БД, то не выдаётся окно диагностического сообщения и диалог выбора пути расположения БД;
- не создается окно консоли ядра.

`/FIXCHAN[=<размер>]`

Примечание
Поддерживается со сборки 6.0.17.92.

При очистке канала выделенная память освобождаться не будет, а будет переиспользоваться для вновь открытого канала с тем же номером. При указании параметра `<размер>` – дополнительно будет резервироваться для канала указанный размер памяти в байтах при его открытии. Максимальное значение 65536.

`/EVENTLIMIT=<размер очереди активных событий>`

Примечание
Поддерживается со сборки 6.0.17.92.

Значение устанавливает максимальный размер очереди активных событий. При превышении значения будет выдан код завершения 90 (Нет памяти для размещения очереди сообщений). Значение по умолчанию и минимальное значение - 128, максимальное - 65535. При старте ядра на консоль и в `linter.out` будет выдано сообщение `Event queue size: NNN items`.

/NOEXTFILE

Примечание
Поддерживается со сборки 6.0.17.92.

Запрещает все обращения к внешним файлам, то есть допустимо заносить имена файлов в столбцы типа EXTFILE, но нельзя выполнять любые операции, при которых формируются полные имена файлов для их поиска в файловой системе.

/RESTRICTEXTFILE

Примечание
Поддерживается со сборки 6.0.17.92.

Запрещает все обращения к внешним файлам, которые лежат вне каталога базы данных, то есть нельзя использовать абсолютные пути и относительные пути, содержащие ссылку на родительский каталог "..", в том числе в пути для файлов по умолчанию.

/SAFE_MODE

Примечание
Поддерживается со сборки 6.0.17.95.

Ядро СУБД запускается в безопасном режиме. В данном режиме запрещены команды: ALTER TABLE DROP COLUMN, PRESS TABLE, CORRECT, RENAME TABLE, RENAME COLUMN, RENAME INDEX, SET SESSION BLOB LOG OFF/ON.

При подаче запрещенной команды в безопасном режиме будет выдан код завершения 97 "Операция запрещена в безопасном режиме".

/[NO]SYNC

При запуске ядра СУБД ЛИНТЕР с параметром /SYNC пул ядра начинает работать по механизму сквозной записи на диск. Т.е. результат операций сразу помещается на диск, минуя кэш.

Отключение данной опции осуществляется установкой параметра /NOSYNC. При этом сброс буферов происходит при завершении транзакции (применяется механизм отложенной записи на диск).

В синхронном режиме работа ядра СУБД ЛИНТЕР замедляется.

По умолчанию /NOSYNC.

Ключи сетевых настроек

/NAME=<строка>

Аналог переменной окружения LINTER_MBX.

/MBX=<параметр>

Аналог переменной окружения LINTER_MBX.

/JDBCP=<порт>

Задает запуск одновременно с запуском ядра СУБД сетевого драйвера сервера для JDBC-сервера с указанным номером <порта>.

/JDBCS

Задает запуск сетевого драйвера сервера для JDBC-сервера с номером порта по умолчанию 1070.

/TCP[=<порт>]

Задает режим запуска сетевого драйвера сервера, работающего по протоколу TCP/IP, одновременно с запуском ядра. <Порт> – номер порта, по которому будет осуществляться соединение клиента с сетевым сервером. Более подробно см. «Сетевые средства», подраздел «Драйвер сервера», пункт «Запуск драйвера».

Значение по умолчанию 1060.

/NONAME

Игнорировать параметры из реестра и параметры запуска сервиса по ключу /NAME.

/DEFAULT

Создание почтового ящика для работы ядра СУБД ЛИНТЕР по умолчанию (см. ключ /NAME).

Ключи совместимости с SQL-стандартом и другими СУБД

```
/COMPATIBILITY=<опция>[,<опция> [, …]]
<опция>::=STANDARD
          |CASTNOLENCHECK
          |CASTNOLTRM
          |GEOPREFIX
          |PGGEO
          |RESIGNAL_ALL
          |ORACLE
          |BROWSE_BLOB
          |NOREC_EXCEPTION
          |OPTIMISTIC
          |NAMES_UPPERCASE
```

Опция STANDARD

При запуске с опцией STANDARD ядро СУБД ЛИНТЕР начинает жёстко придерживаться стандарта SQL92. Поведение ядра, запущенного без ключа с данной опцией, в некоторых случаях не совпадает со стандартом.

Отличия в работе ядра, запущенного с ключом /COMPATIBILITY=STANDARD:

1) позиционные DML-операции и работа транзакций.

   При запуске с опцией STANDARD команды UPDATE CURRENT и DELETE CURRENT могут выполняться по собственному каналу, а не по тому каналу, по которому был подан SELECT-запрос. Это важно для согласованной работы транзакций.

   При запуске без опции STANDARD команды UPDATE CURRENT и DELETE CURRENT выполняются по тому каналу, по которому был подан соответствующий SELECT;

2) очистка канала после завершения транзакции.

   При запуске с опцией STANDARD очистка канала по командам COMT/RBAC (завершение транзакции) закрывает выборку. Исключение составляет выполнение команды PUTM (в СУБД ЛИНТЕР можно подавать команду COMT внутри потока команд пакетного добавления без завершения транзакции), в этом случае выборка не будет закрыта.

   При запуске без опции STANDARD после команд COMT/ROLB выборка не будет закрыта, следовательно, можно подавать команды GET* по тому же каналу;

3) код завершения в случае обработки 0 записей.

   При запуске с опцией STANDARD всеми DML-операциями (SELECT, DELETE, UPDATE, INSERT FROM SELECT) в случае обработки 0 записей возвращается код завершения «Нет данных» (код 2).

   При запуске без опции STANDARD в случае обработки 0 записей операциями DELETE, UPDATE, INSERT FROM SELECT возвращается код успешного завершения (код 0), а код завершения «Нет данных» возвращается только операцией SELECT;

4) выполнение операций 'CAST <выражение> AS CHAR' в том случае, когда <выражение> имеет тип REAL или DOUBLE.

   При запуске с опцией STANDARD результат выполнения описанной операции выводится в экспоненциальной форме с усеченными нулями и без знака '+'.

   При запуске без опции STANDARD результат выполнения описанной операции всегда выводится в форме с десятичной точкой и без экспоненты (так же, как для значений DECIMAL);

5) выполнение операций 'CAST <выражение> AS CHAR', где <выражение> имеет тип DECIMAL.

   При запуске с опцией STANDARD при преобразовании выражения типа DECIMAL с указанием точности в CHAR выводится столько символов после запятой, сколько указано точностью.

   Без опции STANDARD при преобразовании DECIMAL в CHAR концевые нули всегда усекаются;

6) привилегии для выполнения операций DELETE/UPDATE.

   При запуске с опцией STANDARD для выполнения операции DELETE/UPDATE над некоторой таблицей с отбором записей по WHERE необходима не только привилегия DELETE/UPDATE на эту таблицу, но и привилегия SELECT на нее.

   При запуске без опции STANDARD для выполнения операции достаточно привилегии DELETE/UPDATE;

7) усечение лишних концевых пробелов в константах.

   При запуске с опцией STANDARD перед проверкой совместимости типов происходит усечение лишних концевых пробелов в текстовых константах. Концевые пробелы в пределах длины типа не усекаются при операциях над значенями типов CHAR и NCHAR (как это делается в обычном режиме).

```
create or replace table tab2(c1 char(10), c2 char(10));
insert into tab2 values ('123', '456');
set compatibility 'STANDARD' on;
select c1 || c2 from tab2;
|123       456       |
set compatibility 'STANDARD' off;
select c1 || c2 from tab2;
|123456              |
```

   При запуске без опции STANDARD перед проверкой совместимости типов усечение лишних концевых пробелов не происходит, например, не удастся занести в CHAR(5) константу "A " ("буква и 5 пробелов");

8) обязательность условия для JOIN.

   При запуске с опцией STANDARD, если не задано условие соединения (ON или USING) для конструкции JOIN без конструкций NATURAL и UNION, выдается ошибка.

   При запуске без опции STANDARD условие соединения воспринимается, как если бы соединяемые по JOIN таблицы были перечислены во FROM через запятую;

9) ESC-символ по умолчанию.

   При запуске с опцией STANDARD ESC-символа по умолчанию нет.

   При запуске без опции STANDARD ESC-символом по умолчанию является символ '\'.

10) сравнение NULL-значения <значимого выражения условия> в опции CASE с NULL-значением <значимого выражения условия> в опции WHEN.

    При запуске с опцией STANDARD результат - FALSE.

    При запуске без опции STANDARD результат - TRUE.

11) удаление таблицы (синонима), на которую ссылается представление.

    Запрещено обычное удаление таблицы, на которую ссылаются представления (VIEW). При этом на консоль ядра СУБД и в файл linter.out выдаётся список объектов, ссылающихся на удаляемый объект и не позволяющих его удалить.

12) доступ личной последовательности.

    При запуске с опцией STANDARD только создателю последовательности, при запуске без опции STANDARD - всем пользователям.

13) тип данных результата функция SUM.

    При запуске с опцией STANDARD функция SUM с аргументом одного из типов SMALLINT, INT, BIGINT возвращает результат типа BIGINT, при работе в обычном режиме она в этих же случаях возвращает результат типа DECIMAL.

Опция `CASTNOLENCHECK`

Задаёт подавление вывода кода завершения 1063 «`Попытка усечения непустых символов`» при преобразовании числового значения в строковое с явным указанием длины результата в конструкции выполнении `CAST`.

По умолчанию данный код завершения выдается.

Опция `CASTNOLTRM`

Запрещает усечение ведущих пробелов при преобразовании чисел с фиксированной точкой в строковое значение в конструкции `CAST`.

При задании ключа с данной опцией ведущие пробелы в 32-символьном представлении DECIMAL-значения не усекаются, по умолчанию – усекаются.

Опция `GEOPREFIX`

Задаёт распознавание ключевых слов подсистемы геометрических данных только при наличии префикса LIN_ (например, LIN_ASTEXT вместо ASTEXT). По умолчанию воспринимаются ключевые слова без префикса LIN_, хотя некоторые из них распознаются только в определенном контексте (например, ASTEXT, если дальше следует признак функции (открывающая скобка "(")).

Опция `PGGEO`

В данном режиме координаты могут быть разделены пробелами и запятыми, в стандартном режиме они могут быть разделены только пробелами. При этом для каждой точки допустимо указание до 4-х координат из многомерного пространства, реально используются из которых только первые две.

Опция `RESIGNAL_ALL`

Заставляет некритичные исключения обрабатывать как критичные (т.е. при таком запуске ядра СУБД нет необходимости в каждом вложенном блоке явно выполнять оператор `RESIGNAL`).

Если опция `RESIGNAL_ALL` задана, то обработка некритичных исключения сразу передается в текущий <блок обработки исключений>, при его отсутствии – исключение будет передано на верхний уровень по иерархии.

Подробнее см. документ «Процедурный язык», подраздел «Формат блока кода».

Опция `ORACLE`

Заставляет обеспечивать идентичность обработки данных СУБД ЛИНТЕР и ORACLE по умолчанию в случаях, когда их обработка различается.

При указании данной опции:

- разность дат возвращается в виде значения типа NUMERIC (в днях), а не значения типа DATE;
- выдается информация о том, что в SELECT-запросе делается попытка вызова хранимой процедуры, содержащей вызовы запросов, отличных от SELECT (выполнение которых может выдавать различные результаты при последовательных вызовах с одними и теми же аргументами). На консоль ядра СУБД и в файл `linter.out` будут выдаваться сообщения типа

```
WARNING: user function "имя_процедуры" (#ROWID) possibly contains
 non-SELECT query calls
```

- результатом конкатенации символьных значений с NULL-значением будет исходная символьная строка (а не NULL-значение).
- Наличие псевдостолбца с именем «LEVEL» в выборке вида SELECT * FROM… зависит от ключа /COMPATIBILITY=ORACLE в команде запуска ядра СУБД:
  1) псевдостолбец будет присутствовать, если ключ не задан;
  2) псевдостолбец будет отсутствовать, если ключ задан.

Опция `BROWSE_BLOB`

Блокирует текущую запись, если в ней есть BLOB-значение по умолчанию так, как если бы был задан модификатор for browse, т.е. изменяет поведение при обнаружении конфликта доступа к BLOB-значению. В случае отсутствия ключа будет выдано диагностическое сообщение «`Строка при работе с BLOB не заблокирована`», а в случае его наличия будет выполнена блокировка записи.

Опция `NOREC_EXCEPTION`

При выполнении оператора FETCH процедурного языка при отсутствии обработанных строк в курсоре будет сгенерировано исключение 2 (NOREC).

Опция `OPTIMISTIC`

Разрешает использование режима OPTMISTIC.

Опция `NAMES_UPPERCASE`

Приводит все идентификаторы, в том числе указанные в двойных кавычках, к верхнему регистру.

Примечание
Поддерживается со сборки 6.0.17.97.

Ключи оптимизации обработки SQL-запросов

`/ANALYZE`

Оптимизатор СУБД ЛИНТЕР будет анализировать все SQL-запросы с точки зрения существования индексов, позволяющих ускорить их обработку. Список используемых и рекомендуемых к созданию индексов будет выведен на экран консоли ядра и в файл `linter.out` (описание файла приведено в приложении 2). Допустимо указывать один из ключей анализа SQL-запросов.

`/ANALYZE_NEED`

Оптимизатор СУБД ЛИНТЕР будет анализировать все SQL-запросы с точки зрения существования индексов, позволяющих ускорить их обработку. Список рекомендуемых к созданию индексов будет выведен на экран консоли ядра и в файл `linter.out` (описание файла приведено в приложении 2). Допустимо указывать один из ключей анализа SQL-запросов.

Примечание
Поддерживается со сборки 6.0.20.2.

/ANALYZE_USED

Оптимизатор СУБД ЛИНТЕР будет анализировать все SQL-запросы с точки зрения существования индексов, позволяющих ускорить их обработку. Список используемых индексов будет выведен на экран консоли ядра и в файл linter.out (описание файла приведено в приложении 2). Допустимо указывать один из ключей анализа SQL-запросов.

Примечание
Поддерживается со сборки 6.0.20.2.

/AUTOINDEX

Оптимизатор СУБД ЛИНТЕР будет анализировать все SQL-запросы с точки зрения существования индексов, позволяющих ускорить их обработку и создавать необходимые индексы для ускорения обработки SQL-запросов.

Ключи протоколирования работы ядра СУБД

/NOOUTFILE

Запрещает печать сообщений в файл linter.out (протокол работы СУБД).

По умолчанию файл linter.out создается.

Примечание
Структура файла linter.out описана в приложении 2.

/NOOUTPUT

Запрещает вывод в консольное окно информации о работе ядра СУБД.

/OUTFLIMIT=<размер>

Задает размер файла сообщений ядра СУБД (linter.out) в страницах по 4 Кбайт. По умолчанию размер файла неограничен.

Если в процессе работы размер файла linter.out превысит заданный размер, то он будет переименован в файл linter.ou1. После этого информация будет записываться в пустой файл linter.out. Т.е. в процессе работы СУБД могут существовать только два файла – linter.out и linter.ou1.

Примечание
Структура файла linter.out описана в приложении 2.

/LOGFLIMIT=<размер>

/TRACEFLIMIT=<размер>

Задают допустимые размеры файлов LINTER.LOG и lintrace.log соответственно. <Размер> задается в блоках по 4 Кб. Задание размеров файлов приводит к тому, что

перед проверкой ограничений пользователей осуществляется проверка размера файлов LINTER.LOG и lintrace.log. Если их размер превышает заданные границы, вся информация из текущих файлов LINTER.LOG и lintrace.log копируется в файлы с маской LINTER_YYYYMMDDHH24MISS.LOG и lintrace_YYYYMMDDHH24MISS.log, где YYYY – год, MM – месяц, DD – день, HH24 – час, MI – минуты, SS – секунды.

Вслед за этим открываются пустые файлы с именами LINTER.LOG и lintrace.log, и работа продолжается в обычном режиме вплоть до очередной проверки.

Примечания

1. Указанные ключами максимальные размеры файлов LINTER.LOG и lintrace.log могут быть несколько превышены, так как проверки выполняются через определенные промежутки времени.
2. Если ОС не поддерживает размер файлов более 2 Гб, то максимальный размер файла LINTER.LOG, если он не задан, автоматически устанавливается равным 1 Гб.

/[NO]LOG

Ключ /NOLOG – запрещает, ключ /LOG – разрешает вести протокол обработки SQL-запросов клиентских приложений (файл LINTER.LOG).

Если при запуске ядра ЛИНТЕР задан ключ /LOG, то ведется «краткий» протокол обработки запросов: текст SQL-запроса, количество ответов, код возврата.

По умолчанию /NOLOG.

При перезапуске ядра с включенным протоколированием обработки SQL-запросов старый файл будет переименован в файл с маской LINTER_YYYYMMDDHH24MISS.LOG и будет создан новый файл LINTER.LOG.

/LOGQUERY

Задает режим протоколирования обработки SQL-запросов клиентских приложений в файле LINTER.LOG.

При перезапуске ядра с включенным протоколированием обработки SQL-запросов старый файл будет переименован в файл с маской LINTER_YYYYMMDDHH24MISS.LOG и будет создан новый файл LINTER.LOG.

Примечание

Структура файла LINTER.LOG описана в приложении 3.

/LOGALL

Задает режим полного протоколирования обработки SQL-запросов клиентских приложений в файле LINTER.LOG. В этом режиме в файл протокола заносится дополнительная информация: время выполнения команды, сетевой адрес, идентификаторы процесса и нити, пославшие запрос, и др.

При перезапуске ядра с включенным протоколированием обработки SQL-запросов старый файл будет переименован в файл с маской LINTER_YYYYMMDDHH24MISS.LOG и будет создан новый файл LINTER.LOG.

```
/TRACE=DECOMP{[=FULL]|=DELAY[=<тики>]|=SPPAG}
| CHTRAN[=NOFLUSH]
| KRB
| LOCK[=LEVEL={0|1|2|3}|[TIME]]
| LOGIO[=([DEF|COMMT|ABSADR|PREPADR|HEX|BLOCK
| REC|STRUCT|DATA=<значение>|LEVEL=<значение>][, …])]
| KANCHN
| SORT[=TIME]
| WRBL[=READ]
```

Задает различные режимы трассировки обрабатываемых SQL-запросов в трассировочный файл `lintrace.log` (размещается в каталоге БД).

Конструкция `DECOMP` задает режим трассировки общей информации об обрабатываемом запросе. По умолчанию выполняется краткая трассировка.

Для получения более полной информации надо указывать опцию `FULL`.

Трассировка содержит следующую информацию:

- текст SQL-запроса, переданный на обработку ядру СУБД ЛИНТЕР (после его оптимизации SQL-транслятором);
- какие массивы данных (бит-вектора) были задействованы при обработке SQL-запроса. Используемые массивы влияют на время выполнения запроса;
- количество считанных/записанных блоков данных (физических/логических);
- количество считанных/записанных блоков системного журнала.

Пример трассировочной информации

```
C#4
QUERY:
SELECT
T_0."MSG"
FROM
<TABLE "SYSTEM"."ERRORS" AS T_0>
WHERE
T_0."NMRERR" == 1503;
C#4 DECOMP.C (Start_Cur_Dec): Now computing derived set #0.
C#4 OBRSTRAT.C (OBRSTRAT): Start set: TABLE("SYSTEM"."ERRORS" AS
T_0). Set included 1022 rows.
List of predicates:
Predicate [strategy #2(one index)]:
T_0."NMRERR" == 1503
C#4 DECOMP.C (End_Dekart): Derived set #0 is computed, Rows count:
1.
C#4 FORMOTW.C (FORMOTW): Read: 0 blocks, write: 0 blocks.
C#4 FORMOTW.C (FORMOTW): Read logical: 4 blocks, write logical: 0
blocks.
C#4 FORMOTW.C (FORMOTW): Journal read: 0 blocks, written: 0
 blocks.
```

Конструкция DECOMP=DELAY=<тики> задает режим трассировки задержек длительности выполнения отдельных квантов ядра СУБД на время, не меньшее указанного в параметре <тики>. По умолчанию трассируются задержки не менее 100 тиков (0.1 секунды).

Пример трассировочной информации

```
C#3 OBRSTRAT.C (OBRSTRAT): Start set: TABLE("SYSTEM"."LINEITEM"
AS T_1). Set included 6001215 rows.
List of predicates:
Predicate [strategy #1(full scan)]:
T_1."L_SHIPDATE" > '15.03.2013:00:00:00:00'
C#3 PROZA.C (PROZA): Strategy: #1(full scan).
C#3 Delay for 1.87 sec
C#3 Delay for 0.35 sec
C#3 RIDSTRAT.C (RIDSTRAT): Snap_Bv: 3241776 rows.
```

Конструкция CHTRAN задает режим трассировки информации об обрабатываемых транзакциях. В этом режиме выполняется трассирование операций open/ocur/clos/kill/ckil/comt/rbac интерфейса нижнего уровня, а также SQL-запросов COMMIT и ROLLBACK.

Для повышения производительности СУБД предусмотрена опция NOFLUSH. При ее задании трассировочная информация предварительно накапливается в оперативной памяти и записывается в файл lintrace.log в соответствии с настройками ОС (по мере вытеснения буфера файла на диск).

Пример трассировочной информации

```
C#5 OPEN: E=0: M=MVCC_OO
C#6 OCUR: E=0: M=MVCC_RC|MVCC_AUTOCOMMIT: EX=5
C#6 CLOS: E=0
C#5 COMT: E=0: M=MVCC_OO: S=JOURNAL_OPER_EXIST
C#5 TRID: E=0: T=23.09.2014 15:13:51.44: ID=410: M=MVCC_OO:
S=Open_exist|Tr_exist|WRT_OPTIMISTIC_EXIST
C#5 COMT: E=0: T=23.09.2014 15:13:51.44: ID=410: M=MVCC_OO:
S=Open_exist|JOURNAL_OPER_EXIST
C#5 CLOS: E=0
```

Конструкция KRB задает режим трассировки информации, формируемой Kerberos-сервером.

Пример трассировочной информации

Данный пример приведен для имени сервиса LINTER/srv.example.net, имени аутентифицирующегося пользователя test, домена EXAMPLE.NET и компьютера, на котором запущено ядро СУБД ЛИНТЕР, с именем srv.example.net.

```
security.dll was loaded
Use service name LINTER/srv.example.net
QuerySecurityPackageInfo status 0x0
```

```
AcceptSecurityContext status 0x0
User from ticket: 'test@EXAMPLE.NET'
Service: 'LINTER/srv.example.net@EXAMPLE.NET'
QuerySecurityContextToken status 0x0
KRB user name test
```

На клиенте был заранее получен тикет для пользователя test.

Mj и Mi – результат выполнения функции.

Конструкция LOCK добавляет вывод информации о заблокированных таблицах и запросах, вызвавших конфликты.

Опция LEVEL задает дополнительный уровень выдаваемой информации о блокировках. Поддерживаются следующие уровни выдаваемой информации о блокировках:

- LEVEL=0 – выдает в lintrace.log информацию об ошибках;
- LEVEL=1 – дополнительно выдает в lintrace.log информацию о блокировках таблиц (в том числе и тех, которые не вызывают конфликты блокировок);
- LEVEL=2 – дополнительно выдает в lintrace.log информацию о блокировках групп записей;
- LEVEL=3 – дополнительно выдает в lintrace.log информацию о блокировках отдельных записей.

Опция TIME выводит время для каждого сообщения (по умолчанию не выводится).

Пример трассировочной информации

```
TRACELOCK: PAUSE CHANNEL 6 by channel 8
Locked table: "T2"

C#6 QUERY:.
UPDATE.
<TABLE "SYSTEM"."T2" AS T_0>
SET.
T_0."I" = {(T_0."I")(1) <+> };
TRACELOCK: Found DEADLOCK. Channel 6 tries to pause channel 8.
Table to be locked: "T1"

C#8 QUERY:.
UPDATE.
<TABLE "SYSTEM"."T1" AS T_0>
SET.
T_0."I" = {(T_0."I")(1) <+> };
TRACELOCK: UNLOCK CHANNEL 6 paused by channel 8
```

Конструкция LOGIO (опции DEF, COMMT, ABSADR, PREPADR, HEX, BLOCK, REC, STRUCT, DATA, LEVEL) задает режим трассировки записи/чтения системного журнала.

Опция DEF задает установки по умолчанию (действует также, если задано /TRACE=LOGIO без опций).

Опция COMMT по умолчанию отключена. Устаревшая опция, в текущей версии не используется.

Опции ABSADR и PREPADR по умолчанию включены. Задают вывод журнальных адресов в дополнение к трассировке.

Опция HEX по умолчанию отключена. Задает вывод журнального адреса в шестнадцатеричном виде, иначе в десятичном.

Опция BLOCK по умолчанию включена. Задает вывод информации об операциях с журнальными блоками.

Опция REC по умолчанию включена. Задает вывод информации на уровне записей в журнале.

Опция STRUCT по умолчанию включена. Задает вывод информации о структуре журнальных записей.

Опция DATA по умолчанию имеет значение 24. Задает вывод буферов данных (выводится не более указанного числа байт).

Опция LEVEL по умолчанию имеет значение 3. Задает начальный уровень трассировки, который влияет на вывод или подавление вывода той или иной журнальной информации, этот уровень увеличивается и уменьшается в процессе работы.

Конструкция KANCHN задает режим трассировки изменения количества активных каналов.

Конструкция SORT задает режим трассировки обмена с процессами сортировки, опция TIME выводит время для каждого обмена (по умолчанию не выводится).

Конструкция WRBL задает режим трассировки записи на диск/чтения с диска страниц пула, страниц ядра.

Опция READ задает режим трассировки чтения страниц (по умолчанию только запись страниц).

Опция SPPAG предназначена для трассировки использования страниц файла 1.01. Это файл индексов таблицы $$$SYSRL, но в нем также хранятся и разные страницы специальных типов. Типы страниц этого файла такие: страницы индексов, страницы ограничений целостности (integrity), страницы CHECK, страницы диапазонов значений AUTOINC, страницы длинных VIEW, страницы описания составных ключей, страницы расширенной информации о файлах. В трассировке отображаются: выделение страницы с определенным номером в файле, занятие страницы с определенным номером под определенный тип страницы, освобождение страницы с определенным номером из-под определенного типа страницы. Трассировка предназначена для поиска проблем, связанных со страницами файла 1.01.

/TRACELOG[=1]

Задает вывод в файл linter.out диагностических сообщений об открытии и закрытии соединения с БД (с полной информацией об источнике команды) и ошибок выполнения запросов по соединениям.

При задании параметра 1 информация о соединении выводится с каждым ошибочным запросом.

/PROCPRINT

Разрешает хранимым процедурам выводить:

- на консоль ядра СУБД ЛИНТЕР и в файл протоколирования linter.out сообщения процедурной функции PRINT. Максимальный размер выводимого сообщения 980 символов.

  Сообщение выводится в виде:

  «*** Message from Stored Procedure: <текст сообщения>»

  По умолчанию выполнение функции PRINT игнорируется.

Примечание
Вывод сообщений в файл протоколирования linter.out поддерживается со сборки 6.0.17.95.

- на консоль ядра СУБД ЛИНТЕР и в файл протоколирования linter.out информацию об исключениях. Если хранимая процедура оттранслирована с отладкой, то выдается номер ошибочной строки в процедуре.

  Примеры вывода.

```
Exception DIVZERO caught in procedure "TEST" line 2, processed
Exception 903 caught in procedure "AAA" line 8, processed
Exception CUSTOM (1) caught in procedure "TEST", processed
Exception 905 caught in procedure "Trigger # 47#" line 3,
 processed
Exception BADPARAM caught in procedure "TEST3" line 2, resignaled
Exception DIVZERO caught in procedure "TEST", ignored
```

  Не протоколируются те исключения, которые игнорируются автоматически (без явной конструкции IGNORE).
- Функционал аналогичен подаче SQL-команды "SET PROCEDURE TRACE ON;", но не требуется перезапуск ядра.

/NOERRDIALOG

Запрещает отображать диалоговые окна с диагностическими сообщениями об ошибках запуска ядра СУБД, направляя эти сообщения на консоль ядра СУБД.

/PIDFILE=<файл>

Задает имя файла, в который будет записан Pid ядра СУБД. В случае корректного завершения работы СУБД этот файл удаляется.

Ключи LDAP-аутентификации

/LDAPSUFFIX=<строка>

Строка, добавляемая после имени пользователя, чтобы сформировать уникальное имя (DN) в БД LDAP-сервера.

Аналог переменной окружения LDAP_SUFFIX с более высоким приоритетом.

`/LDAPPREFIX=<строка>`

Строка, добавляемая перед именем пользователя, чтобы сформировать уникальное имя (DN) в БД LDAP-сервера.

Аналог переменной окружения LDAP_PREFIX с более высоким приоритетом.

`/LDAPTOUT=<целочисленное значение>`

Тайм-аут в секундах.

Аналог переменной окружения LDAP_TIMEOUT с более высоким приоритетом.

`/LDAPSRV=<адрес>`

Адрес LDAP-сервера в виде доменного имени.

Аналог переменной окружения LDAP_SERVER с более высоким приоритетом.

`/LDAPBASEDN=<строка>`

Корень поиска для LDAP-аутентификации с предварительным поиском.

Аналог переменной окружения LDAP_BASEDN с более высоким приоритетом.

`/LDAPFLTR=<строка>`

Фильтр поиска пользователя в БД LDAP.

Аналог переменной окружения LDAP_FILTER с более высоким приоритетом.

`/LDAPSRCHDN=<dn>`

Уникальное имя пользователя (DN), от которого будет производиться предварительный поиск в БД LDAP-сервера.

Аналог переменной окружения LDAP_SEARCHDN.

`/LDAPSRCHPW=<пароль>`

Пароль пользователя, от имени которого будет выполняться предварительный поиск в БД LDAP-сервера.

Аналог переменной окружения LDAP_SEARCHPW с более высоким приоритетом.

`/LDAPSRCHPWFILE=<спецификация файла>`

Путь к файлу, содержащему пароль DN для поиска в БД LDAP-сервера.

Аналог переменной окружения LDAP_SEARCHPW_FILE с более высоким приоритетом.

Ключи KERBEROS-аутентификации

`/KRBSRVC=<строка>`

Задает имя ЛИНТЕР-сервиса для Kerberos-сервера. Аналог переменной окружения LINTER_KRB_SERVICE, но с более высоким приоритетом.

Информационные ключи

`/BRIEFVERSION`

Задает вывод в диалоговое окно краткой информации о версии ядра СУБД (если ядро запускается как приложение ОС).

`/VERSION`

Задает вывод в диалоговое окно подробной информации о версии ядра СУБД (если ядро запускается как приложение ОС).

Переменные среды окружения

В своей работе ядро и утилиты СУБД ЛИНТЕР используют переменные среды окружения, которые могут быть установлены средствами ОС до запуска СУБД ЛИНТЕР.

Для установки значения переменной среды окружения для сеанса консольного файлового менеджера необходимо выполнить в нем команду:

```
set <имя переменной>=<значение переменной>
```

Пример.

```
set SY00=C:/Linter/db/DEMO
```

При этом для применения значений переменных для утилит СУБД они должны запускаться в командном режиме в текущем сеансе консольного файлового менеджера.

Общие переменные

В процессе работы СУБД ЛИНТЕР может использовать следующие переменные среды окружения:

1) **SY00** – значение переменной определяет путь до каталога основных файлов базы данных (файлов системных таблиц). По умолчанию в качестве значения переменной используется текущий каталог ОС. Этот путь может быть задан также из командной строки при запуске ядра;
2) **LINTER_EDIT** – переменная, определяющая текстовый редактор, используемый для редактирования, например, SQL-запросов (в утилите `inl`);

Примечание
Используется утилитами СУБД ЛИНТЕР.

3) **LINTER_MBX** – переменная, определяющая «имя почтового ящика» для обмена данными между ядром СУБД ЛИНТЕР и клиентскими приложениями. Под «почтовым ящиком» следует понимать некий межпроцессный механизм обмена

между ядром ЛИНТЕР и приложением. Изменение значения этой переменной может быть использовано для запуска нескольких ядер на одном компьютере.

Значение LINTER_MBX должно иметь строковый тип и быть уникальным для данного ядра СУБД ЛИНТЕР и приложений, работающих с этим ядром. По умолчанию "";

Пример.

а) в nodetab добавить строку:

```
DemoDb LOCAL Demo
```

б) в файловом менеджере, в котором будем запускать утилиты, выполнить команду:

```
SET LINTER_MBX=Demo
```

в) выполнить запуск ядра СУБД и сетевого драйвера клиента в файловом менеджере:

```
C:\Linter\bin>linternt /local /base=C:\Linter\db\Demo
C:\Linter\bin>dbc_tcp /local /ver=5
```

г) запустить новую сессию файлового менеджера, выполнить п.б и запустить утилиту командного интерфейса с подключением к БД DemoDb:

```
inl -u SYSTEM/MANAGER8 -n DemoDb
```

Успешный переход в интерактивный режим выполнения команд:

```
SQL>
```

является подтверждением успешного подключения к ядру СУБД;

д) завершить утилиту командного интерфейса командой `exit;`

е) завершить работу сетевого клиента: в сессии командного менеджера, в которой он запущен, нажать сочетание клавиш <Ctrl>+<C>;

ж) остановить ядро СУБД с помощью команды:

```
shut -u SYSTEM/MANAGER8
```

4) **LINTER_CP** – определяет кодировку данного клиентского приложения. Ядро СУБД будет работать с этим приложением в кодировке, определяемой переменной LINTER_CP. Для разных приложений, даже на одной машине, могут быть заданы различные значения переменной LINTER_CP.

   СУБД ЛИНТЕР поддерживает однобайтовые, многобайтовые кодировки, UNICODE, UTF8, а также предоставляет возможность загрузить в БД кодировку, необходимую клиентской задаче. Переменная LINTER_CP может принимать значение любой кодировки, загруженной в БД (список кодировок хранится в системной таблице $$ $CHARSET).

   Если значение переменной LINTER_CP не определено, то используется кодировка, соответствующая текущему значению locale.

   Для консольных утилит СУБД ЛИНТЕР используется кодировка по умолчанию 866, для графических утилит – 1251.

5) **NET_MBX** – переменная аналогична по своему назначению переменной LINTER_MBX, но для сетевого клиента. Т.е. NET_MBX определяет «имя почтового

ящика» для обмена данными между приложением и сетевым драйвером клиента. Значение по умолчанию: пустое.

Примечание
«Имя почтового ящика» формируется из предопределенного префикса и значения переменной NET_MBX. Поэтому даже если NET_MBX не определено, «имя почтового ящика» пустым не будет.

Переменные идентификации и аутентификации по LDAP-протоколу

Возможны два режима LDAP-аутентификации:

1) без предварительного поиска;
2) с предварительным поиском.

Для успешной аутентификации парольная аутентификация по LDAP-протоколу через LDAP-сервер требуется наличие пользователя в БД ЛИНТЕР и в БД LDAP-сервера, при этом имя пользователя БД ЛИНТЕР должно соответствовать одному из атрибутов пользователя в БД LDAP-сервера (для аутентификации без предварительного поиска данный атрибут должен входить в состав уникального имени пользователя в БД LDAP-сервера). Кроме того, чтобы для режима LDAP-аутентификации с предварительным поиском не требовалось задавать DN и пароль пользователя с правами поиска, необходимо разрешение на анонимный поиск в LDAP-базе.

В режиме без предварительного поиска сначала идет подключение к серверу LDAP_SERVER и попытка выполнить соединение с уникальным именем, сформированным как LDAP_PREFIX<имя>LDAP_SUFFIX, где <имя> и пароль определяются соответствующими параметрами команды OPEN (см. документ «Интерфейс нижнего уровня»). Удачное соединение означает удачную идентификацию и аутентификацию.

В режиме с предварительным поиском сначала идет подключение к серверу LDAP_SERVER и попытка выполнить соединение со специальными именем LDAP_SEARCHDN и паролем, который может быть задан одним из способов:

- напрямую через переменную LDAP_SEARCHPW;
- с помощью ключа LDAPSRCHPW;
- через файл, полный путь к которому передается в параметре LDAP_SEARCHPW_FILE;
- анонимно, если эти учетные данные не заданы.

После этого производится поиск пользователя в БД LDAP-сервера в каталоге LDAP_BASEDN с фильтром LDAP_FILTER. Найденное в результате поиска уникальное имя (если единственное) используется для попытки соединения с введенным пользователем паролем. Удачное соединение означает удачную идентификацию и аутентификацию.

Для настройки LDAP-аутентификации ядра СУБД ЛИНТЕР используются следующие переменные окружения:

- **LDAP_PREFIX** определяет строку, добавляемую в качестве префикса к имени пользователя БД ЛИНТЕР для формирования уникального имени (DN) для LDAP-аутентификации. Переменная служит для указания имени атрибута,

соответствующего выбранной схеме, отличной от значения по умолчанию. Если переменная LDAP_PREFIX не задана, по умолчанию используется строка "cn=". Например, для пользователя SYSTEM БД ЛИНТЕР будет использован атрибут с значением cn=SYSTEM (если LDAP_PREFIX ="cn=", т.е. стандартное имя пользователя). Если LDAP_PREFIX ="sn=", то имя пользователя БД СУБД ЛИНТЕР будет соответствовать атрибуту Surname LDAP-схемы.

Переменная игнорируется в режиме LDAP-аутентификации с предварительным поиском.

Значение переменной может задаваться с помощью ключа /LDAPPREFIX.

- **LDAP_SUFFIX** определяет строку, добавляемую в качестве суффикса к имени пользователя БД ЛИНТЕР для формирования уникального имени (DN) для LDAP-аутентификации. LDAP_SUFFIX выбирается при настройке LDAP-сервера и обычно соответствует доменному имени организации. Значение строки должно состоять из предопределенных имен атрибутов схемы LDAP-сервера (ou – организационная единица, c – страна, dc – домен и т.п.) и их значений. Указание переменной LDAP_SUFFIX является обязательным для режима LDAP-аутентификации без предварительного поиска. Если эта переменная не задана, то любой запрос на аутентификацию пользователей с включенной LDAP-аутентификацией будет отвергнут.

  Например,

```
LDAP_SUFFIX="ou=Users, dc=company, dc=com"
```

  DN строка формируется как конкатенация строк LDAP_PREFIX, имени пользователя СУБД ЛИНТЕР, символа ',' и LDAP_SUFFIX.

  Переменная игнорируется в режиме LDAP-аутентификации с предварительным поиском.

  Значение переменной может задаваться с помощью ключа /LDAPSUFFIX.

- **LDAP_SERVER** задает адрес LDAP-сервера в виде доменного имени. Если переменная не задана, по умолчанию предполагается работа с локальным LDAP-сервером (localhost). В переменной LDAP_SERVER может быть указан номер порта, отделяемый от доменного имени двоеточием. Можно задавать несколько серверов через пробел, в этом случае попытка соединиться с очередным LDAP-сервером будет предприниматься после неудачного соединения с предшествующим LDAP-сервером.

  Например,

```
LDAP_SERVER="ldap.server.domain.org"
LDAP_SERVER="ldap.server.domain.org:636"
LDAP_SERVER="ldap.server.relex.org"
LDAP_SERVER="ldap1.server.domain.org ldap2.server.domain.org:636
ldap3.server.domain.org"
```

  Значение переменной может задаваться с помощью ключа /LDAPSRV.

- **LDAP_TIMEOUT** задает интервал в секундах, в течение которого ожидается ответ от LDAP-сервера. Если в течение указанного интервала ответ не поступил, возвращается код завершения «Тайм-аут LDAP-сервера». По умолчанию интервал тайм-аута принимается в 10 секунд.

  Ожидание ответа от LDAP-сервера выполняется в чередующихся фазах активного и пассивного тайм-аута:

Фаза активного тайм-аута	Фаза пассивного тайм-аута	…	Фаза активного тайм-аута	Фаза пассивного тайм-аута

В фазе активного тайм-аута ЛИНТЕР-сервер ожидает ответ от LDAP-сервера в течение указанного в `LDAP_TIMEOUT` интервала времени. Если ответ не поступил, генерируется код завершения «Тайм-аут LDAP-сервера».

Если пользователь продолжает попытки соединиться с ЛИНТЕР-сервером, то в течение следующей фазы пассивного тайм-аута (также равного значению `LDAP_TIMEOUT`) все попытки соединения с LDAP-сервером немедленно отвергаются с выдачей кода завершения «Тайм-аут LDAP-сервера» (хотя к этому моменту соединение с LDAP-сервером, возможно, уже восстановлено).

После пассивной фазы тайм-аута повторяется фаза активного тайм-аута и т.д. Это сделано с целью предотвращения накопления очереди пользователей, ожидающих аутентификации, при сбоях в работе LDAP-сервера.

Пример

```
LDAP_TIMEOUT=15
```

Значение переменной может задаваться с помощью ключа /LDAPTOUT.

- **LDAP_BASEDN** задает режим LDAP-аутентификации с предварительным поиском уникального имени пользователя в БД LDAP-сервера. Значением переменной является корень поиска, производимого через заданный фильтр.

  Задание переменной обязательно для LDAP-аутентификации с предварительным поиском

  Значение переменной может задаваться с помощью ключа /LDAPBASEDN.

- **LDAP_FILTER** задает фильтр, используемый для поиска пользователя в БД LDAP-сервера. Строка фильтра может содержать служебное слово '%username%', которое при поиске заменяется на введенное при установлении соединения с СУБД ЛИНТЕР имя пользователя БД ЛИНТЕР. В найденной записи извлекается уникальное имя пользователя (DN), которое используется совместно с введенным пользователем паролем для попытки аутентификации. Если фильтр находит несколько записей (либо не находится ни одной), то аутентификация считается не пройденной.

  Переменная игнорируется в режиме LDAP-аутентификации без предварительного поиска.

  Значение по умолчанию "(uid=%username%)".

  Значение переменной может задаваться с помощью ключа /LDAPFLTR.

- **LDAP_SEARCHDN** задает уникальное имя (DN), с помощью которого будет производиться поиск в БД LDAP-сервера.

  Переменная игнорируется в режиме LDAP-аутентификации без предварительного поиска. Если значение переменной не задано или задана пустая строка, то будет использоваться анонимный поиск (т.е. без предоставления регистрационных данных пользователя БД LDAP-сервера).

  Значение переменной может задаваться с помощью ключа /LDAPSRCHDN.

- **LDAP_SEARCHPW** задает пароль для имени, от которого будет производиться поиск в БД LDAP-сервера.

  Переменная игнорируется в режиме LDAP-аутентификации без предварительного поиска или в случае анонимного поиска.

  Значение переменной может задаваться с помощью ключа /LDAPSRCHPW.
- **LDAP_SEARCHPW_FILE** задает путь к файлу, содержащему пароль `LDAP_SEARCHPW`. Переменная имеет более высокий приоритет, чем переменная `LDAP_SEARCHPW` и ключ `/LDAPSRCHPW`.

  Значение переменной может задаваться с помощью ключа /LDAPSRCHPWFILE.

Переменные идентификации и аутентификации по Kerberos-протоколу

Для идентификации и аутентификации по Kerberos-протоколу используются следующие переменные окружения:

- **LINTER_KRB_SERVICE** определяет имя службы ЛИНТЕР-сервера. Kerberos-сервер должен быть настроен на использование данной службы.

  Например,

```
LINTER_KRB_SERVICE="linter/linterserver.relex.ru"
LINTER_KRB_SERVICE="linter@linterserver.relex.ru"
```

- **KRB5_KTNAME** задает местоположение (полный путь и имя) файла с таблицей ключей. Может использоваться некоторыми реализациями Kerberos.

> **Примечание**
> Переменная окружения **KRB5_KTNAME** используется ядром СУБД ЛИНТЕР не напрямую, а опосредованно, через библиотеку Kerberos.

Значения размеров очередей, задаваемые по умолчанию

По умолчанию при запуске ядра СУБД ЛИНТЕР устанавливаются следующие значения:

- длина или размер очереди, содержащей описания таблиц – 100;
- длина или размер очереди, содержащей описания столбцов – 500;
- длина или размер очереди файлов – 30;
- количество каналов для связи приложений с ядром СУБД ЛИНТЕР – 100;
- квант обработки записей – число записей, просматриваемых системой без прерывания при обработке одного запроса – 10;
- квант обработки индексов – число индексов, просматриваемых без прерывания при обработке одного запроса – 1.

Консольное окно ядра СУБД ЛИНТЕР

При успешном запуске СУБД ЛИНТЕР в консольном режиме открывается окно, в которое выводятся параметры и установленные режимы работы ядра (см. рис. 1).

Консольный режим предоставляет возможности для установки параметров работы ядра, просмотра протокола обработки SQL-запросов, останова ядра и др.

Настройка параметров запуска ядра

Пункт меню Настройки позволяет просматривать и изменять значения параметров запуска ядра СУБД ЛИНТЕР (рис. 2).

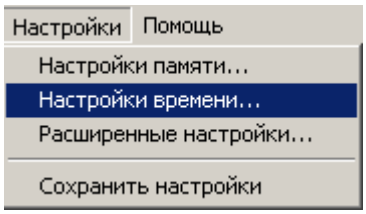

Рисунок 2. Изменение значений параметров ядра СУБД ЛИНТЕР

Измененные значения параметров вступят в силу после выполнения пункта меню **Сохранить настройки** и только после перезапуска ядра СУБД ЛИНТЕР. До останова ядро будет работать с теми значениями параметров, с которыми оно было запущено.

Управление размером пулов

Для управления размером пула ядра и пула сортировки предназначен пункт меню **Настройки=>Настройки памяти…** (рис. 3):

- в поле **Пул ядра** задается размер пула ядра (аналог параметра `/pool`);
- в поле **Пул сортировки** задается размер пула сортировки (аналог параметра `/spool`);
- при установленном флажке **В памяти** задается размер пула для таблиц «в памяти» (аналог параметра `/inmempool`);
- при установленном флажке **Фразовый индекс** задается размер пула памяти для фразового индекса (при использовании полнотекстового поиска) (аналог параметра `/ppool`);
- установленный флажок **Блокировать в физической памяти** задаёт блокирование выделенной для работы ядра оперативной памяти (под очереди системных объектов, пул страниц и т.п.) на уровне операционной системы (аналог параметра `/lock`);

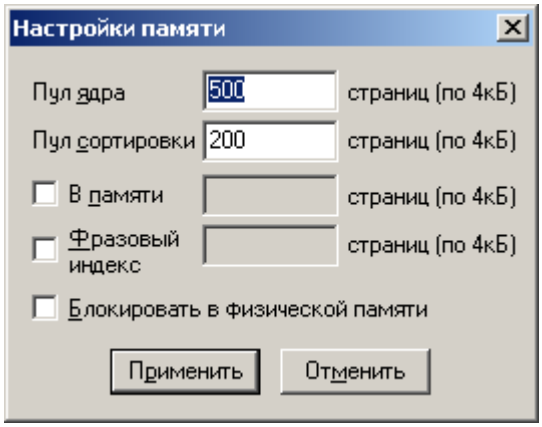

Рисунок 3. Настройка памяти ядра

Управление тайм-аутом

Пункт меню **Настройки=>Настройки времени…** (рис. 4) позволяет задавать периодичность проверки ядром «живучести» клиентских приложений (аналог параметра `/kill`).

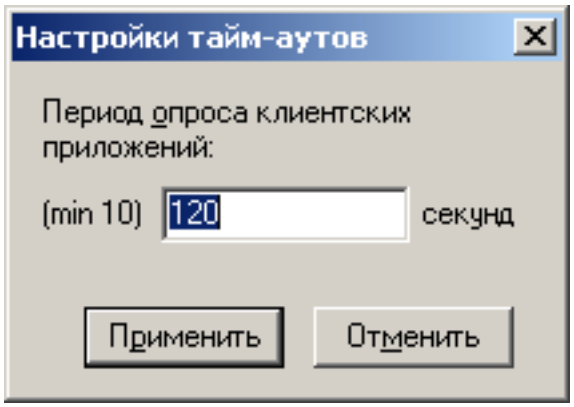

Рисунок 4. Управление тайм-аутами

Расширенные настройки

Пункт меню **Настройки=>Расширенные настройки…** (рис. 5) позволяет задавать дополнительные параметры функционирования ядра СУБД ЛИНТЕР. Установленные флажки действуют аналогично соответствующим параметрам запуска командной строки СУБД ЛИНТЕР.

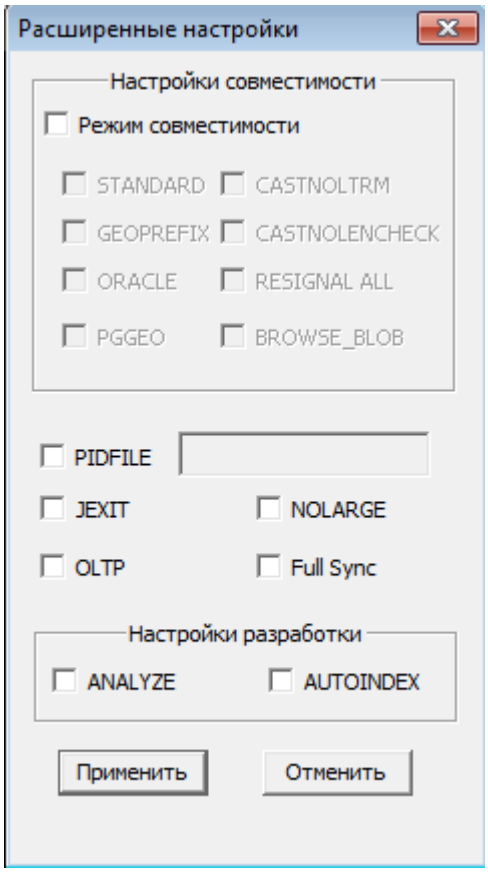

Рисунок 5. Расширенные настройки ядра

Управление работой ядра

Для управления работой ядра используется пункт меню **Файл** (рис. 6).

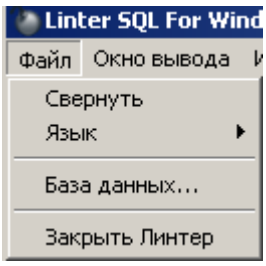

Рисунок 6. Управление работой ядра

Выбор БД

Для выбора БД, с которой должно работать ядро СУБД ЛИНТЕР:

1) выполнить команду **Файл=>База данных…** (см. рис. 6);
2) в появившемся окне (рис. 7) ввести вручную или выбрать с помощью кнопки **Обзор** каталог с нужной БД и подтвердить свой выбор.

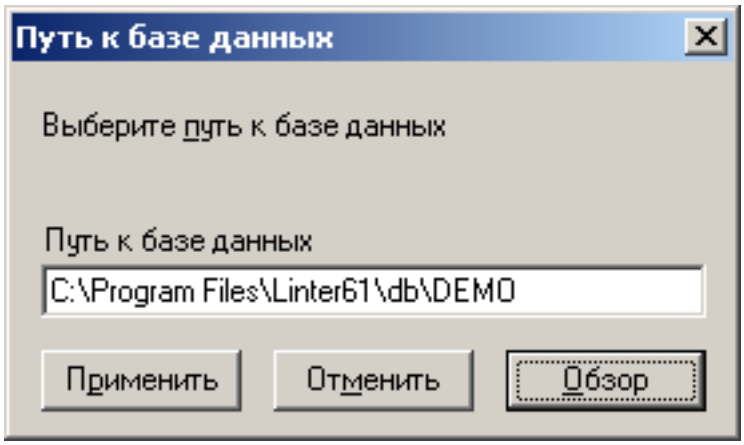

Рисунок 7. Окно выбора БД

Язык интерфейса

Для выбора языка консольного интерфейса (русский/английский) выполнить команду **Файл=>Язык** (см. рис. 6).

Свертывание программы в трей

Для свертывания консольного интерфейса ядра в область трея выполнить команду **Файл=>Свернуть** (см. рис. 6).

Завершение работы

Для завершения работы ядра выполнить команду **Файл=>Закрыть Линтер** (см. рис. 6).

После выбора этого пункта меню откроется стандартное диалоговое окно для подтверждения (или отказа) завершения работы ядра.

Управление выводом на консоль

Для управления консольным выводом ядра СУБД ЛИНТЕР используется пункт меню **Окно вывода** (рис. 8).

Команда **Запретить вывод** запрещает вывод консольной информации.

Команда **Очистить окно вывода** удаляет с консоли всю ранее выданную информацию.

Команда **Отметить** выводит на консоль маркер – строку в виде:

```
--------------------MARKED <дата> <время>--------------------
```

Маркеры используются для быстрого поиска в консольном выводе нужной информации.

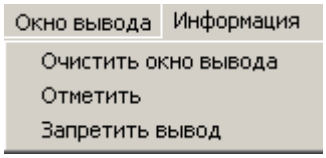

Рисунок 8. Управление выводом на консоль

Управление протоколированием работы ядра

Для управления протоколированием работы ядра СУБД ЛИНТЕР используется пункт меню **Информация** (рис. 9).

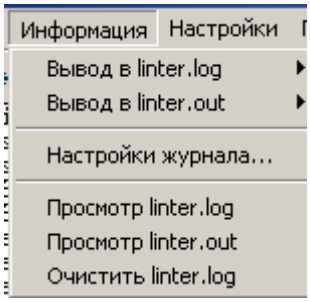

Рисунок 9. Управление протоколированием

Пункты меню **Вывод в linter.log**, **Вывод в linter.out** позволяют задать режим протоколирования (обычный/расширенный) или отменить протоколирование полностью (аналог параметров `/log` и `/nolog`). В файле `LINTER.LOG` ведется протокол обработки SQL-запросов, посланных клиентскими приложениями.

Пункты меню **Просмотр linter.log**, **Просмотр linter.out** позволяют просматривать соответствующие файлы протоколирования. Для этого будет открыто окно текстового редактора и в него загружен файл протокола.

Пункт меню **Очистить linter.log** удаляет все записи из файла `LINTER.LOG`.

Пункт меню **Настройки журнала …** (рис. 10) управляет выводом информации в системный журнал СУБД ЛИНТЕР. Установленные флажки действуют аналогично соответствующим параметрам запуска командной строки СУБД ЛИНТЕР.

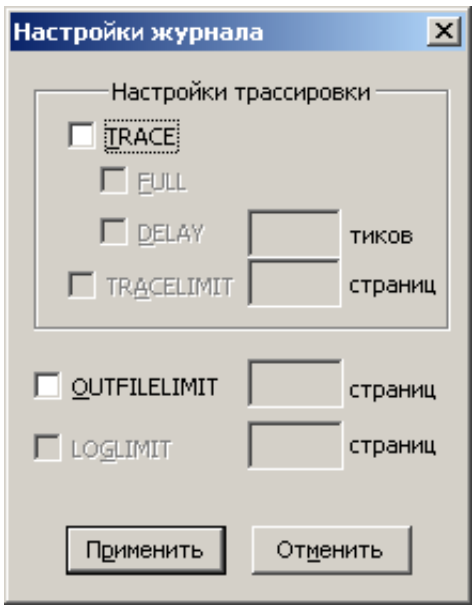

Рисунок 10. Настройки системного журнала

Останов СУБД ЛИНТЕР

Пользовательский останов ядра СУБД ЛИНТЕР

Останов ядра СУБД можно выполнить либо в консоли ядра СУБД (пункт **Файл => Закрыть Линтер**) либо с помощью специальной утилиты `shut`, которая подает команду останова ядру СУБД и ожидает, пока оно завершится.

Синтаксис команды

```
shut [<командная строка>]

<командная строка>::=[<имя>] [<пароль>] [<ЛИНТЕР-сервер>]
```

или

```
<командная строка>::={[-u имя/пароль] [-n <ЛИНТЕР-сервер>]
                      [-r] [-ci <кодировка>] | -version | -h}
```

Примечание

Если в процессе работы выполнялось расширение системных файлов (SYSWRK, SYSSRT, SYSWBV), то перед завершением работы ядро СУБД усекает их до размера, указанного при создании БД (или при её конфигурировании) и выдает об этом на консоль ядра СУБД и в файл `linter.out` информационное сообщение вида:

```
_Attention: file SYSWRK truncated from 16 to 4 pages

(Внимание: файл SYSWRK был усечен с 16 страниц до 4)
```

В связи с этим для исключения затрат СУБД на операции расширения рабочих файлов рекомендуется выполнить переконфигурирование БД, указав в качестве начальных размеров рабочих файлов те размеры, до которых они были автоматически расширены (см. документ «Создание и конфигурирование базы данных», пункт «Конфигурирование БД»).

Ключи управления утилитой

-u

Задает регистрационные данные (имя и пароль) создателя БД или пользователя БД с привилегией DBA.

-n

Задает имя узла удаленного ЛИНТЕР-сервера (должно присутствовать в файле `nodetab`). Если этот параметр не задан, то команда применяется к локальному узлу или к узлу по умолчанию.

-r

Заставляет выполнить завершение работы ядра, вне зависимости от наличия активных транзакций в данный момент. В случае наличия таких транзакций, произойдет их откат, а пользователи будут извещены о принудительном останове ядра. По команде останова ядра без ключа `-r` при наличии активных транзакций ядро не будет остановлено, и утилита `shut` получит код возврата 1012.

-ci <кодировка>

Задает кодовую страницу для интерфейса утилиты.

Если ключ не задан, по умолчанию используется язык операционной системы.

Если кодовая страница задана неверно или не установлена в ОС, используется англоязычный интерфейс.

Примеры:

```
-ci cp866 (русскоязычный интерфейс)
-ci cp437 (англоязычный интерфейс)
```

-version

Выдает информацию о версии программы shut.

-h

Выдается справочная информация о программе.

Системный останов ядра СУБД ЛИНТЕР

Системный останов ядра СУБД ЛИНТЕР инициируется с помощью системной команды ОС завершения работы программы.

В этом случае работа ядра завершается без уведомления пользователя.

Коды завершения

Коды завершения ядра СУБД

Если во время запуска ядра СУБД ЛИНТЕР происходит сбой, то анализируется причина сбоя, и работа ядра завершается. При этом выдается соответствующий код завершения.

Код завершения 1

Диагностическое сообщение

```
Wrong database version. Check version for database and kernel.
```

Причина ошибки

Версия запускаемого ядра СУБД не соответствует версии БД, на которой его пытаются запустить.

Рекомендации по устранению

Варианты:

1) выполнить запуск ядра СУБД с БД соответствующей версии.
2) выполнить перенос БД на новую версию СУБД ЛИНТЕР с помощью утилиты `migration` или `datariv`.

Для определения версии СУБД ЛИНТЕР и БД выполнить команды:

```
linternt -version
gendb -get "DATABASE VERSION"
```

См. документы:

- «Миграция базы данных»;
- «Конвертер баз данных»;
- «Создание и конфигурирование базы данных».

Код завершения 2

Диагностическое сообщение

```
Database is corrupt. Please repair.
```

Причина ошибки

Повреждение физической структуры БД.

Ответственный за устранение

Администратор БД.

Рекомендации по устранению

Выполнить рекомендованные действия в соответствии с документами:

- «Тестирование базы данных»;

- «Архивирование и восстановление базы данных».

Код завершения 3

Диагностическое сообщение

```
Not enough memory. Change kernel startup parameters.
```

Причина ошибки

Для запуска ядра СУБД ЛИНТЕР недостаточно доступной оперативной памяти.

Рекомендации по устранению

Варианты:

1) уменьшить размер пула ядра СУБД и/или уменьшить размер пула сортировки (см. описание ключей /POOL и /SPOOL в этом документе);
2) увеличить размер доступной ядру СУБД оперативной памяти либо за счет аппаратного увеличения оперативной памяти компьютера, либо уменьшить количество запущенных на компьютере программ.

Код завершения 4

Диагностическое сообщение

```
Low pool size. Change kernel startup parameters.
```

Причина ошибки

Заданный при запуске ядра СУБД размер пула ядра не достаточен для его работы.

Рекомендации по устранению

Увеличить размер пула ядра СУБД (см. описание ключа /POOL в этом документе).

Код завершения 5

Диагностическое сообщение

```
Can't initialize kernel. Syscall failed.
```

Причина ошибки

Возможные причины:

1) для БД, запущенной в режиме только для чтения (ключ /RO), длина пути к каталогу временных файлов превышает допустимую длину.

   Путь к каталогу временных файлов задается либо в ключе /DIR, либо в переменной среды окружения LINTER_TMP).

   Максимальная длина пути к каталогу задается в макросе ОС PATH_MAX (значение по умолчанию 32);

2) ошибка инициализации или установки дескриптора безопасности, создания мэйлслота, нити, объекта файлового отображения;

3) ошибка инициализации подсистемы репликации СУБД ЛИНТЕР;

4) ядро уже запущено на другой БД с тем же LINTER_MBX.

Рекомендации по устранению

В первом случае создать каталог для временных файлов БД с допустимым по длине именем.

Во втором и третьем случае следует установить причину ошибки, изучив диагностические сообщения в файле `linter.out` каталога базы либо системный код ошибки из сообщения об ошибке (отображаемое либо в диалоговом окне, либо в событии в журнале событий). В дальнейшем действовать исходя из полученной информации об ошибке.

В четвертом случае изменить значение переменной LINTER_MBX перед запуском ядра СУБД.

Код завершения 6

Диагностическое сообщение

```
Linter already running. Change kernel startup parameters.
```

Причина ошибки

Ядро СУБД на указанной БД уже запущено.

Рекомендации по устранению

Чтобы запустить несколько ядер СУБД ЛИНТЕР на одном компьютере, необходимо использовать ключ `/NAME` и переменную окружения `LINTER_MBX`.

См. документ:

- «Сетевые средства».

Код завершения 7

Диагностическое сообщение

```
Can't access database. Check database file presence and
permissions.
```

Причина ошибки

Возможные причины:

1) местоположение БД неизвестно;

2) БД отсутствует в указанном каталоге;

3) у пользователя нет прав для запуска ядра СУБД на указанной БД.

Рекомендации по устранению

В первом случае указать местоположение БД с помощью ключа `/LOCAL`, `/BASE` или переменной среды окружения `SY00`.

Во втором случае указать правильный каталог БД.

В третьем случае предоставить пользователю права DBA или предъявить пароль доступа к защищенной БД.

См. документ:

- «Справочник по SQL».

Код завершения 8

Диагностическое сообщение

```
Kernel abandon due to critical data write to disk fail.
```

Причина ошибки

Невозможно продолжать запись в системный журнал (ядро СУБД ЛИНТЕР запущено с ключом `/JEXIT`) из-за отсутствия свободного места на диске или исчерпания файлов системного журнала.

Рекомендации по устранению

1) Обеспечить наличие свободного пространства на диске.
2) Увеличить количество файлов системного журнала и/или их размер.

Количество файлов системного журнала и размер каждого из них задаются в свойствах БД с помощью утилиты `linadm` или `gendb` (параметры `SYSLOG COUNT`, `SYSLOG SIZE`).

См. документы:

- «Сетевой администратор»;
- «Создание и конфигурирование базы данных».

Код завершения 9

Диагностическое сообщение

```
Conflict between system time and time of last db shutting.
```

Причина ошибки

Дата последнего запуска ядра СУБД (по временному поясу GMT) является «будущей» по сравнению с текущей датой ОС.

Рекомендации по устранению

Запускать ядро СУБД ЛИНТЕР с ключом `/TCORRECT`.

Код завершения 10

Диагностическое сообщение

```
Can't find temporary directory or environment variable 'TEMP'
is not found.
```

Причина ошибки

Возможные причины:

1) для БД, запущенной в режиме «только чтение» (ключ `/RO`), невозможно определить временный каталог;
2) задано неверное значение переменной среды окружения `TEMP`.

Рекомендации по устранению

Указать правильный каталог для размещения временных файлов БД с помощью ключа запуска `/TMPDIR` или в переменной среды окружения `TEMP`.

Код завершения 11

Диагностическое сообщение

```
Database files locked by another application. May be Linter
already running on this database.
```

Причина ошибки

Ядро СУБД не может заблокировать индексный файл системной таблицы `$$$SYSREL`. Указанная БД уже заблокирована другим приложением (возможно, на ней уже запущено ядро СУБД).

Рекомендации по устранению

Завершить работу приложения, блокирующего файлы БД, на которой запускается ядро СУБД, и повторить запуск ядра СУБД.

Код завершения 12

Диагностическое сообщение

```
Wrong username/password specified.
```

Причина ошибки

Неверное имя пользователя или пароль в ключе `-u` запуска ядра СУБД.

Рекомендации по устранению

Ввести правильные регистрационные данные пользователя, от имени которого выполнятся запуск ядра СУБД.

Список пользователей БД можно получить с помощью SQL-запроса:

```
select $$$s34 from $$$usr;
```

(для этого надо запустить ядро СУБД от имени пользователя с правильными регистрационными данными).

Код завершения 13

Диагностическое сообщение

```
Kernel arguments error.
```

Причина ошибки

Ошибка работы с файлом, содержащим ключи командной строки запуска ядра СУБД ЛИНТЕР (ключ `/CF`).

Возможные причины:

1) неверная спецификация (файл не найден);
2) длина записей файла больше 4096 байт;
3) значение ключа `/NAME` или `/MBX` превышает 32 символа;
4) значение ключа `/TMPDIR` превышает значение, установленное в макросе `PATH_MAX` в ОС (по умолчанию 32).

Рекомендации по устранению

Исправить файл с ключами командной строки запуска ядра СУБД ЛИНТЕР (см. ключ `/CF` в этом документе) и повторить запуск ядра СУБД.

Код завершения 15

Диагностическое сообщение

```
License check failed.
```

Причина ошибки

Истек срок использования демонстрационной версии СУБД ЛИНТЕР.

Рекомендации по устранению

Приобрести лицензионную версию СУБД ЛИНТЕР.

Коды завершения программы останова СУБД

В таблице приведен перечень кодов завершения программы останова СУБД.

Таблица. Перечень кодов завершения программы останова СУБД

Код завершения	Описание
0	Успешное завершение.
1	Неполные или неверные аргументы.

Код завершения	Описание
2	Неверные имя пользователя/пароль или нет привилегий для завершения работы ядра.
3	Ядро СУБД или сетевой драйвер не загружены, не найден указанный сервер.
4	Неизвестная ошибка при обращении к СУБД ЛИНТЕР.
5	Ядро СУБД выполняет другие запросы.

Приложение 1
Пример настройки СУБД ЛИНТЕР для идентификации и аутентификации по Kerberos-протоколу

В примере используется программное средство Active Directory.

Исходные данные:

- Kerberos-сервер и ядро СУБД ЛИНТЕР находятся на одном компьютере с ОС Windows с адресом win-server.windows.server;
- клиентские приложения – на компьютерах под управлением ОС Linux и Windows.

Порядок настройки:

1) на сервере с Active Directory зарегистрировать с помощью программного средства SPN имя сервиса для того пользователя, от имени которого будет проверяться служба ядра СУБД ЛИНТЕР kerberos:

```
setspn -A LINTER/win-server.windows.server test
```

   где:

   `test` – имя пользователя домена;

   `LINTER` – имя сервиса (в данном примере используется LINTER, в общем случае может быть произвольным);

   `win-server.windows.server` – адрес компьютера, на котором будет запускаться ядро СУБД ЛИНТЕР (в данном примере это компьютер с именем WIN-SERVER в домене WINDOWS.SERVER);

2) на компьютере, где будет запускаться ядро СУБД ЛИНТЕР, создать переменную окружения, задающую имя зарегистрированного на предыдущем шаге сервиса:

```
LINTER_KRB_SERVICE="LINTER/win-server.windows.server"
```

   Запустить ядро СУБД ЛИНТЕР.
   Запуск может быть также осуществлен без установки переменной окружения. В этом случае имя сервиса передается в командной строке. Например:

```
linter64 /tcp /KRBSRVC=LINTER/win-server.windows.server
          /base=c:\linter\db\DEMO
```

   Запуск должен производится от администратора.
   Имена пользователей в БД Active Directory должны совпадать с именами пользователей в БД СУБД ЛИНТЕР;

3) создать в БД ЛИНТЕР пользователя с режимом идентификации и аутентификации по Kerberos-протоколу и тем же именем, что и в БД Kerberos-сервера:

```
inl -u SYSTEM/MANAGER8
> CREATE USER "test" IDENTIFIED BY KRB;
> CREATE USER "user" IDENTIFIED BY KRB;
```

4) на Linux-клиентах изменить (или сгенерировать заново) файл `/etc/krb5.conf` так, чтобы в качестве REALM выступал Windows-домен (в данном примере это WINDOWS.SERVER),

а в качестве KDC и admin-server – адрес компьютера с сервером Active Directory (в данном примере это win-server.windows.server).

После этого Kerberos-тикет можно получить с помощью программы `kinit`:

```
kinit
```

5) на компьютере с Windows-клиентом подключиться к Active Directory домену. Для этого в настройках сетевых подключений в качестве DNS-адреса должен быть указан IP-адрес компьютера с Active Directory (после этого можно изменить домен в свойствах компьютера на WINDOWS.SERVER);
6) на Windows-клиенте для доступа к Kerberos-серверу ввести имя и пароль из БД Active Directory, выбрав домен WINDOWS.SERVER. Kerberos-тикет в этом случае будет получен на этапе соединения с СУБД ЛИНТЕР;
7) клиентские Linux и Windows приложения готовы к идентификации и аутентификации по Kerberos-протоколу. Для доступа к СУБД ЛИНТЕР по Kerberos-протоколу необходимо указать пустые имя и пароль пользователя. Доступ к СУБД ЛИНТЕР будет предоставлен тому пользователю, который указан в полученном от Kerberos-сервера на предыдущем шаге Kerberos-тикете.

Приложение 2
Описание файла linter.out

В приложении приводится пример информации, содержащейся в файле сообщений ядра СУБД ЛИНТЕР (`linter.out`).

В этот файл заносится информация о таких параметрах запуска ядра, как размер очередей (Table, Column, Channel, File, User), размер пула ядра (POOL), количество процессов сортировок; а также значение переменной `LINTER_MBX`, состояние системных таблиц БД (Devices, Charset, Security, Procedures и т.д.). Сюда же вносятся предупреждения и сообщения об ошибках, возникающих в процессе работы СУБД.

Файл сообщений является дополняемым, т.е. при последующих запусках СУБД сообщения будут дописываться в конец файла `linter.out`.

При старте ядра СУБД ЛИНТЕР на уже созданной базе данных в файл `linter.out` заносятся сообщения о дате создания БД и датах последнего запуска и останова базы данных:

```
Database creation time: 22.10.2013 06:26:57.00
Last database startup time: 12.11.2013 08:59:02.05
Last database shutdown time: 12.11.2013 10:12:45.17
```

При завершении работы СУБД в файл `linter.out` заносится сообщение:

```
*** RDBMS Linter has been shut down ***
```

Пример файла `linter.out`.

```
22.10.2013
06:26:57 GENDB-I-CRFILE, Создается системная база данных ...
GENDB-I-SUCCRD,база данных создана
Database creation time: 22.10.2013 06:26:57.00
Last database startup time: Unknown
Last database shutdown time: Unknown
22.10.2013 06:26:59 Linter SQL v. 6.0.17.5 connected to data base
 "DEMO DATABASE"
22.10.2013 06:26:59 POOL holds 853 pages
22.10.2013 06:26:59 Table queue size : 100
22.10.2013 06:26:59 Column queue size : 500
22.10.2013 06:26:59 Channel queue size : 100
22.10.2013 06:26:59 File queue size : 30
22.10.2013 06:26:59 User queue size : 100
22.10.2013 06:26:59 Procedures table is absent
22.10.2013 06:26:59 Procedures Dictionary table is absent
22.10.2013 06:26:59 Security groups table is absent
22.10.2013 06:26:59 Security levels table is absent
22.10.2013 06:26:59 Devices table is absent
22.10.2013 06:26:59 Stations table is absent
22.10.2013 06:26:59 Replication rules table is absent
22.10.2013 06:26:59 Trigger table is absent
```

```
22.10.2013 06:26:59 Filter table is absent
22.10.2013 06:26:59 File extension table is absent
22.10.2013 06:26:59 Sequence table is absent
22.10.2013 06:26:59 Charset table is absent
22.10.2013 06:26:59 Translate table is absent
22.10.2013 06:26:59 Errors table is absent
22.10.2013 06:26:59 In-kernel backup table is absent
22.10.2013 06:26:59 Auditing disabled
22.10.2013 06:27:00 Transaction control is turned on
22.10.2013 06:27:00 Max concurrent sorting processes : 1
Existing control point(s) in database:
Control point list is empty.
22.10.2013 06:27:00 Kernel system parameters: MBX - "615", Pid -
 364
22.10.2013 06:27:00Copyright (C) 1990-2013 Relex, Inc. All rights
reserved.22.10.2013 06:27:00
22.10.2013 06:27:00 Fail! NO LICENSE FOUND!
22.10.2013 06:27:00 *** RDBMS Linter is running
22.10.2013 06:27:00 *** Press <ENTER> for shell prompt
22.10.2013 06:27:17 W-145 (OPENTH) : Invalid charset: CP437
22.10.2013 06:27:18 E-150 (CHARSET) : Charset #0 is used
22.10.2013 06:27:24 *** Linter is coming down ***
22.10.2013 06:27:24 *** PP processing ***
22.10.2013 06:27:24 *** Table queue processing ***
22.10.2013 06:27:24 *** Column queue processing ***
22.10.2013 06:27:24 *** User queue processing ***
22.10.2013 06:27:24 *** File queue processing ***
22.10.2013 06:27:24 *** RDBMS Linter has been shut down ***
Database creation time: 22.10.2013 06:26:57.00
Last database startup time: 12.11.2013 08:59:02.05
Last database shutdown time: 12.11.2013 10:12:45.17
12.11.2013 10:13:21Linter SQL v. 6.0.17.5 connected to data base
 "DEMO DATABASE"
12.11.2013 10:13:21 POOL holds 357 pages
12.11.2013 10:13:21 Table queue size : 100
12.11.2013 10:13:21 Column queue size : 500
12.11.2013 10:13:21 Channel queue size : 100
12.11.2013 10:13:21 File queue size : 29
12.11.2013 10:13:21 User queue size : 100
12.11.2013 10:13:21 In-kernel backup table is absent
12.11.2013 10:13:21 Auditing disabled
12.11.2013 10:13:21 Transaction control is turned on
12.11.2013 10:13:21 Max concurrent sorting processes : 1
Existing control point(s) in database:
Control point list is empty.
12.11.2013 10:13:21 Kernel system parameters: MBX - "", Pid - 1676
```

```
12.11.2013 10:13:21Copyright (C) 1990-2013 Relex, Inc. All rights
reserved.12.11.2013 10:13:21
12.11.2013 10:13:21 It's a DEMO license.
12.11.2013 10:13:21 *** RDBMS Linter is running
12.11.2013 10:13:21 *** Press <ENTER> for shell prompt
12.11.2013 15:08:23 WARNING: invalid channel number: 3
socket: 0 length = 1632 net = 0 cmd = RBAC. Continue work
12.11.2013 15:08:23 WARNING: invalid channel number: 3
socket: 0 length = 1632 net = 0 cmd = KILL. Continue work
12.11.2013 15:08:27 *** Linter is coming down ***
12.11.2013 15:08:27 *** PP processing ***
12.11.2013 15:08:27 *** Table queue processing ***
12.11.2013 15:08:27 *** Column queue processing ***
12.11.2013 15:08:27 *** User queue processing ***
12.11.2013 15:08:27 *** File queue processing ***
12.11.2013 15:08:28 *** RDBMS Linter has been shut down ***
```

Приложение 3
Описание файла LINTER.LOG

В приложении рассматриваются примеры информации, которая заносится в файл протоколирования обработки SQL-запросов пользователя (файл LINTER.LOG), и приводится расшифровка данной информации.

Файл LINTER.LOG является перезаписываемым, т.е. при запуске ядра (с параметром, разрешающим ведение протокола) все сообщения о предыдущем сеансе работы СУБД будут удалены из файла.

Пример 1. Пример файла LINTER.LOG в режиме краткого протоколирования.

```
?DESC:L=92:
!:E=0:C=0:
?OPEN:U="SYSTE<":P=16384:R=0:
!:E=1025 @&#:C=0:
?OPEN:U=H:P=16384:R=0:
!:E=1025 @&#:C=0:
?OPEN:U="SYSTEM":P=16384:R=0:
!:E=0:C=3:
?DESC:L=92:
!:E=0:C=3:
?DESC:L=92:
!:E=0:C=3:
?OCUR:C=3:P=16384:R=0:
!:E=0:C=4:
?SLCT:C=4:L=65535:P=0:
select * from "SYSTEM"."$$$AUDIT";;
!:E=0:C=4:A=1:
?GETA:C=4:L=0:
!:E=0:C=4:
?KILL:C=3:U=H:I=4:
!:E=0:C=3:
?CLOS:C=4:
!:E=1069 @&#:C=4:
OCUR:C=3:P=16384:R=0:
!:E=0:C=4:
?:C=4:L=65535:P=0:
execute "SYSTEM"."SAMPLE"(FALSE,FALSE,1);
#?OCUR:C=4:P=34:R=0:
#!:E=0:C=5:
#?:C=5:L=0:P=0:
drop table results;
#!:E=0:C=5:
#?:C=5:L=0:P=0:
create table results(lin char(100));
```

```
#!:E=0:C=5:
```

Пример 2. Пример файла `LINTER.LOG` в режиме полного протоколирования.

```
?SLCT:T=10:26:25.370:XPid=1732:XTid=1632:C=5:L=65535:P=0:
select * from auto;;;
!:E=0:T=10:26:25.450:XPid=1732:XTid=1632:C=5:A=1000:
?GETA:T=10:26:25.450:XPid=1732:XTid=1632:C=5:L=0:
!:E=0:T=10:26:25.450:XPid=1732:XTid=1632:C=5:
?GETA:T=10:26:25.450:XPid=1732:XTid=1632:C=5:L=3914:
!:E=0:T=10:26:25.460:XPid=1732:XTid=1632:C=5:
?OCUR:T=10:26:25.460:XPid=1732:XTid=1632:C=3:P=16384:R=0:
!:E=0:T=10:26:25.460:XPid=1732:XTid=1632:C=6:
?GETS:T=10:26:25.480:XPid=1732:XTid=1632:C=5:I=1:L=113:
!:E=0:T=10:26:25.480:XPid=1732:XTid=1632:C=5:
?GETS:T=10:26:25.480:XPid=1732:XTid=1632:C=5:I=1:L=113:
!:E=0:T=10:26:25.480:XPid=1732:XTid=1632:C=5:
```

Знак **?**, расположенный в начале строки, показывает, что информация относится к обработке запроса. Затем идёт команда из блока CBL (например, OCUR, SLCT, DESC и т.п.). Если ведется полное протоколирование работы, то будет показана также дополнительная информация: время выполнения запроса, сетевой адрес, идентификаторы процесса и нити, пославшие запрос. Далее идут параметры команд из блока CBL, которые описаны ниже. Затем (на следующей строке) расположены данные, которые необходимы ядру для выполнения команды (если они требуются): текст запроса, данные для загрузки в BLOB и т.д.

Знак **!**, расположенный в начале строки, указывает на то, что это информация об ответе ядра на запрос пользователя. Она включает в себя код возврата (:E=«код возврата») и признак ошибки (символы @&#, а затем параметры блока CBL), если была ошибка. Например, строка, сообщающая об ошибке при выполнении запроса, может выглядеть так !:E=1025 @&#:C=0: (см. пример 1).

Знак **#** в начале строки (перед **?** или **!**) – признак того, что запрос подан из триггера или хранимой процедуры.

Параметры блока CBL:

- C – номер канала;
- L – длина буфера ответа (поле LnBufRow);
- P – режим обработки команды внутреннего интерфейса;
- R – приоритет канала;
- U – имя пользователя/пароль;
- I – внутренний системный номер записи, которая была обработана последней (RowId);
- K – значение параметра, необходимого при обработке запроса. Например, номер BLOB-поля;
- S – размер запроса или обработанных данных;
- O – код ошибки, переданный СУБД операционной/сетевой средой при обработке запроса (SysErr);

- A – количество реально обработанных записей (RowCount).

Примечание
Подробно блок управления запросом CBL, описание его полей и параметров приводится в документе «Интерфейс нижнего уровня».

TRID – начало новой транзакции.

E – код завершения операции.

Обозначения для переменных канала:

- T – Tr_Time;
- ID – ChTransactionID;
- M – Tr_mode;
- S – Tr_Status.

Также используется дополнительное поле EX для расширенной информации.

Указатель ключей

U

V

W