

МОБИЛЬНАЯ
РЕЛЯЦИОННАЯ
СУБД

ЛИНТЕР[®]

Linter Standard
Linter Bastion
Linter RealTime
Linter Multiversion

Командный интерфейс

НАУЧНО-ПРОИЗВОДСТВЕННОЕ ПРЕДПРИЯТИЕ

 **РЕЛАКС**[®]

Товарные знаки

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

Интеллектуальная собственность

Правообладателем продуктов ЛИНТЕР®, НЕВОД®, LAV™, ЛАКУНА является компания РЕЛЭКС (1990–2011). Все права защищены. Данный документ является собственностью компании РЕЛЭКС. Ни одна часть данного документа не может быть воспроизведена, передана, преобразована, сохранена в системе поиска информации, переведена на другой язык или компьютерный язык в какой-либо форме, какими-либо средствами, электронными, механическими, магнитными, оптическими, химическими, ручными или иными без предварительного разрешения компании РЕЛЭКС.

О документе

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

Адрес

394006, г. Воронеж, ул. 20-летия Октября, 119.
Тел./факс: (473) 2-711-711, 2-778-333.
e-mail: market@relex.ru.

Адрес для корреспонденции

394000, г. Воронеж, а/я 137.

Техническая поддержка

Отдел поддержки и сопровождения программных продуктов:

телефон: (473) 2-711-711 с 9:00 до 18:00 мск.
e-mail: support@relex.ru, market@relex.ru.

С целью повышения качества разрабатываемых программных средств и предоставляемых услуг в компании РЕЛЭКС действует автоматизированная система учёта и обработки рекламаций. Обо всех обнаруженных недостатках и ошибках в программном продукте и/или документации на него просим сообщать нам на Internet–странице [рекламация](#).

Оглавление

Предисловие	1
Назначение документа	1
Для кого предназначен документ	1
Необходимые предварительные знания.....	1
Принятые обозначения и соглашения	1
Назначение	4
Условия выполнения	4
Переменные окружения	4
Запуск	5
Ключи	6
Обработка SQL-операторов	9
Ввод.....	9
Результаты	10
Редактирование.....	13
SQL-операторы с параметрами.....	13
Команды	18
!.....	18
ALTER PROC	19
BROWSE.....	20
DISFILL.....	23
CODEPAGE	24
COUNT	25
CREATE PROC	26
CREATE TRIG	26
DISREP	27
DBINFO	28
ECHO.....	29
EDIT	30
_	32
PESSIMISTIC.....	33
EXEC	34

Оглавление

EXIT	35
FORMAT	35
HEADER	36
DISHEAD	37
HELP	38
IGNORE	40
LIST	41
OPTIMISTIC.....	42
OUTFIL.....	43
RESULT	44
PAGE.....	45
PRIORITY	46
SH.....	48
SHOW	49
SLEEP	51
TIME	51
UNLOAD	52
DISSEP	54
DISTAIL.....	55
USERNAME.....	56
Работа с BLOB-данными.....	57
Сообщения программы.....	59
Информационные сообщения	59
Диагностические сообщения	59
Коды завершения.....	60

Предисловие

Назначение документа

Документ содержит описание возможностей программы `inl`, реализующей командный интерфейс пользователя с СУБД ЛИНТЕР. Командный интерфейс обеспечивается для всех программных платформ, на которых функционирует СУБД ЛИНТЕР.

Приводится описание управляющих, информационных команд и команд настройки интерфейса. В приложениях приведены примеры интерактивных сеансов.

Документ может использоваться для работы с любой версией СУБД ЛИНТЕР. Особенности конкретных версий оговариваются по тексту.

Для кого предназначен документ

Документ предназначен для программистов и профессиональных пользователей СУБД ЛИНТЕР.

Командный интерфейс может использоваться:

- для проверки (отладки) синтаксиса и семантики SQL-запросов в процессе разработки пользовательских приложений;
- для получения и анализа временных характеристик обрабатываемых запросов;
- для выполнения SQL-скриптов как в интерактивном режиме, так и в командных файлах операционной системы;
- для форматирования и выдачи на экран (или на печать) результатов поисковых SQL-запросов.

Необходимые предварительные знания

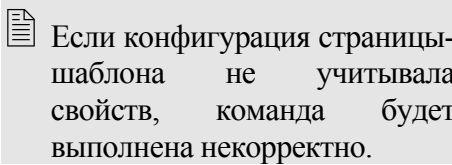

Для работы с командным интерфейсом необходимо;

- знать основы реляционных баз данных;
- владеть языком баз данных SQL для СУБД ЛИНТЕР;
- уметь работать в соответствующей операционной системе на уровне простого пользователя.

Принятые обозначения и соглашения

<u>Обозначение</u>	<u>Пример</u>	<u>Значение</u>
Курсив	<i>Растровым</i> называется изображение...	Новый термин в тексте
Полужирный шрифт	В этом случае необходимо переносить все физические файлы.	Выделение в тексте
Подчеркнутый шрифт	<u>Подробную</u> информацию о работе программы можно	Адреса страниц Internet

<u>Обозначение</u>	<u>Пример</u>	<u>Значение</u>
Текст, разделенный знаком ⇒	получить на сайте www.dmk.ru Выполните команду View ⇒ Properties (Вид ⇒ Свойства).	Последовательность выполнения команд
Текст, заключенный в <>, со знаком + между ними	<Ctrl>+<C>	В <> заключаются клавиши клавиатуры, знак + означает сочетание клавиш
Крупный моноширинный текст	SQL> _q	Текст командной строки
Мелкий моноширинный текст	Page Time Count	Текст программы
Заглавные буквы	BROWSE	Названия команд, слова, зарезервированные в SQL, ключевые слова
Курсив в <>	<return statement>	Определяемый элемент синтаксической конструкции
Символ ::=		Равенство по определению. Слева от знака стоит определяемое понятие, справа – собственно определение понятия
Квадратные скобки []	DBSTORE [-d -r -t -u]	Необязательные элементы конструкции. В данном примере ключи не являются обязательными элементами команды
Вертикальная черта	<return value> ::= <value expression> NULL	Указывает на то, что все предшествующие ей элементы списка являются необязательными и могут быть заменены любым другим элементом списка после этой черты
Фигурные скобки { }	CODEPAGE {866 1251 KOI8}	Указывают на то, что все находящееся внутри них является единым целым
Многоточие «...»	Характеристики столбца MAKE CHAR(20) MODEL CHAR(20) ... SQL>	Означает, что предшествующая часть может быть повторена любое количество раз
Многоточие, внутри которого		Указывает, что

Обозначение	Пример	Значение
находится запятая «.,..»		предшествующая часть оператора, состоящая из нескольких элементов, разделенных запятыми, может иметь произвольное число повторений
Текст со знаком  на сером фоне		Примечание

Назначение

Командный интерфейс (программа inl) может использоваться:

- 1) для проверки (отладки) синтаксиса и семантики SQL-запросов в процессе разработки пользовательских приложений;
- 2) для получения и анализа временных характеристик обрабатываемых SQL-запросов;
- 3) для выполнения SQL-скриптов как в интерактивном режиме, так и в командных файлах операционной системы;
- 4) для форматирования и выдачи на экран (или на печать) результатов поисковых SQL-запросов.

Условия выполнения

Для выполнения inl необходимы следующие условия:

- 1) СУБД ЛИНТЕР должна быть активна (кроме запуска inl с ключом -h);
- 2) в момент запуска inl СУБД ЛИНТЕР должна иметь в общем случае три свободных канала;
- 3) минимальный объем оперативной памяти, Мбайт – 8;
- 4) пользователь, от имени которого запускается inl, должен быть зарегистрирован в БД, к которой осуществляется доступ.
- 5) БД, к которой осуществляется доступ.

Переменные окружения

Утилита inl распознает следующие переменные окружения:

- LINTER_INLDEFCONNMODE – режим обработки транзакций по умолчанию.

Спецификация:

LINTER_INLDEFCONNMODE=optimistic | pessimistic |exclusive



Значение должно задаваться в нижнем регистре.

Если переменная окружения не задана, то действует autocommit.

- LINTER_INLDATEOUT – формат для вывода даты по умолчанию. Если задан, то для вывода даты используется указанный формат, иначе – формат по умолчанию. Спецификацию формата см. в документе «СУБД ЛИНТЕР. Библиотеки специальных типов данных», функция TICKTOSTRF.

Запуск

Запуск программы осуществляется стандартными средствами запуска задач, имеющимися в каждой операционной системе. Исполняемый файл утилиты – `inl`.






Командная строка:

```
inl [-u <имя пользователя>[/[<пароль пользователя>]]]  
[-n <имя сервера>]  
[-f <спецификация файла>]  
[-t]  
[-p <спецификация файла>]  
[-c <кодировка данных>]  
[-ci <кодировка интерфейса>]  
[-s]  
[-q]  
[-{h|?}]  
[-es]  
[-nf][-nc][-ns][-nl][-nh][-ia]  
[-i <код завершения>]  
[-version]
```

По умолчанию для соединения с БД используется кодировка OEM.


Ключи

Для передачи программе параметров используется набор **ключей**, позволяющих однозначно интерпретировать вид параметра. Все ключи имеют уникальное мнемоническое обозначение. Ключи в командной строке можно располагать в любой последовательности (за одним исключением: если используются оба ключа `-h` и `-c`, то ключ `-c` должен следовать первым, чтобы сообщения, выводимые ключем `-h`, отображались в указанной кодировке).

-  Команды и ключи допускается вводить как малыми, так и большими буквами.
-  При вводе значений ключей (имена, пароли, наименования таблиц и т.п.) малые и большие буквы различаются.
-  Если задан ключ, не относящийся к команде, то ошибка не фиксируется, а ключ программой не обрабатывается (игнорируется).
-  Признаком ключа является знак минус «-», альтернативный признак ключа / (обратная косая черта) допустим во всех ОС, кроме Unix.
-  Для получения справочной информации о ключах программы необходимо задать в командной строке символ `-?` или `-h`.

Ниже перечислены используемые утилитой ключи с описанием их применения.


<u>Ключ</u>	<u>Описание</u>
<code>-u</code>	Идентификация пользователя с аргументами <code><ИМЯ пользователя></code> и <code><пароль пользователя></code> . Если пароль не указан, он считается состоящим из одних пробелов
<code>-n</code>	Идентификация имени ЛИНТЕР-сервера. Имеет смысл только при работе в сети
<code>-f</code>	Определение имени файла, содержащего SQL-запрос (серию или пакет SQL-запросов), подлежащий выполнению
<code>-p</code>	При работе в пакетном режиме задает имя текстового файла с паролями пользователей. Формат строк файла: <code><имя пользователя>/[<пароль>]</code> Если <code><пароль></code> не задан, он будет запрошен интерактивно с консоли. Особенности использования ключа <code>-p</code> одновременно с ключом <code>-u</code> <code><имя пользователя>[/[<пароль пользователя>]</code> : 1) если после имени пользователя в ключе <code>-u</code> знак <code></></code> присутствует, то подразумевается, что пароль задан (он будет пустым, если после знака <code></></code> нет символов пароля), и в этом случае ключ <code>-p</code> игнорируется; 2) если после имени пользователя знака в ключе <code>-u</code> знака <code></></code> нет, то пароль не задан, и тогда при наличии ключа <code>-p</code> пароль берется из файла. Параметры файла паролей: <ul style="list-style-type: none">• максимальное количество строк: 100 (лишние строки


Ключ	Описание
	игнорируются); <ul style="list-style-type: none"> • максимальная длина строки: 100 байтов; • кодировка текста: ASCII. <p>Ключ <code>-p</code> может задаваться не только в паре с ключом <code>-f</code>, но и отдельно, в частности, при вводе в командной строке имени пользователя без пароля.</p> <p>Пример</p> <pre>inl -u SYSTEM/ -p pwd.txt</pre> <p>Inl выдаст сообщение о неверном пароле, если пароль у пользователя SYSTEM не пустой. Если в файле <code>pwd.txt</code> задан пароль для пользователя SYSTEM – он будет взят из файла. Если нет, то будет запрошен интерактивно</p>
<code>-t</code>	Отключение выдачи временной статистики
<code>-s</code>	Запрещает выдачу на экран диагностического сообщения в случае кода завершения 2202 при удалении объекта из БД (т. е. в БД нет удаляемого объекта). На все другие диагностические сообщения, связанные с операцией удаления объекта (например, синтаксическая ошибка в SQL-запросе, отсутствие привилегий и т.п.) действие ключа не распространяется. Также запрещает выдачу на экран диагностического сообщения с уведомлением об удалении несуществующих объектов
<code>-q</code>	Запрет выдачи на экран/печать информации об авторских правах
<code>-ia</code>	Заставляет игнорировать любой код завершения SQL-запроса и продолжать работу программы
<code>-i</code>	Запрещает/разрешает выдачу на экран диагностического сообщения для указанного <кода завершения>. Ключ действует как переключатель: если <код завершения> еще не установлен, то он устанавливается, если уже установлен, то сбрасывается
	 Список установленных по ключу <code>-i</code> кодов завершения можно посмотреть по команде <code>SHOW</code> .
<code>-es</code>	Заставляет выдавать на экран тексты синтаксически или логически неправильных SQL-запросов
<code>-nh</code>	Запрет выдачи заголовка
<code>-nc</code>	Запрет выдачи результата SQL-запроса по команде <code>count</code> (включая возвращаемое хранимой процедурой значение)
<code>-ns</code>	Запрет выдачи разделителя столбцов (см. команду <code>UNLOAD</code>)
<code>-nl</code>	Запрет выдачи разделителя столбцов в начале и конце строк ответа (см. команду <code>UNLOAD</code>)
<code>-nf</code>	Для столбцов с переменной длиной запрещает дополнять выдаваемое значение до ширины столбца
<code>-ci -c</code>	<p>Задаёт кодовую страницу для интерфейса утилиты.</p> <p>Если ключ не задан, по умолчанию используется язык операционной системы.</p> <p>Если кодовая страница задана неверно или не установлена в ОС, используется англоязычный интерфейс.</p>

Ключ	Описание
	Примеры: -сi ср866 (русскоязычный интерфейс). -сi ср437 (англоязычный интерфейс).
-h или ?	Выдает на экран справочную информацию и завершает выполнение программы. Если в командной строке заданы другие ключи, то они игнорируются
-version	Информация о версии программы

Принятые умолчания

- 1) ключи регистронезависимы;
- 2) если при запуске опущен ключ `-u` (вместе с аргументами), то прежде чем приступить к работе, `inl` запрашивает у пользователя имя и пароль;
- 3) если опущен ключ `-n`, то `inl` будет работать с сервером по умолчанию;
- 4) при отсутствии ключа `-f` программа переходит в интерактивный режим работы;
- 5) при отсутствии ключа `-t` выдается время начала и окончания обработки каждого SQL-запроса.
- 6) если ключ `-c` не задан, то берется значение переменной окружения `LINTER_CP`; если и она не задана, то возьмется национальная кодировка (`locale`), установленная в ОС.

 При вводе пароля вводимые символы маскируются.

 При запуске `inl` ей можно передавать команду для выполнения, например:
`echo 'select * from auto;' | INL -u SYSTEM/MANAGER`

Пример

Для запуска программы:

1. набрать имя программы и нажать клавишу `<Enter>`:

```
INL <Enter>
```

2. появится заставка программы и приглашение для ввода имени пользователя:

```
Интерактивный SQL в.6.0.0          СУБД ЛИНТЕР в.6.0  
Имя пользователя: SYSTEM
```

3. после ввода имени пользователя появится приглашение для ввода пароля пользователя:

```
Пароль пользователя:MANAGER
```

4. если регистрационные параметры введены правильно, появится подсказка готовности к работе:

```
SQL>
```

Обработка SQL-операторов

Выполнение SQL-операторов возможно в двух режимах: интерактивном и пакетном.

В интерактивном режиме текст SQL-оператора вводится с клавиатуры и сразу передается на обработку `inl`.

В пакетном режиме текст SQL-оператора предварительно записывается в текстовый файл (SQL-скрипте) с помощью любого текстового редактора имеющегося в операционной системе и поддерживающего тип файла `.txt`, либо с помощью `inl`, затем SQL-скрипт передается на обработку `inl`. SQL-скрипт может содержать любое количество SQL-операторов.

Для предоставления входных SQL-операторов и результатов из выполнения в требуемой кодировке надо:

- либо установить нужную кодировку с помощью переменной среды окружения `LINTER_CP`;
- либо предварительно выполнить SQL-оператор `SET NAMES <кодировка>`;

Ввод

Ввод SQL-операторов в интерактивном режиме выполняется по следующим правилам:

- 1) ввод SQL-оператора должен начинаться после приглашения `inl`;
`SQL>select count(*) from auto;`
- 2) если текст SQL-оператора не уместается на одной строке консоли, он может быть продолжен на следующих строках;
- 3) прекращение ввода текущей строки и переход на следующую строку выполняется по клавише `<Enter>`;
- 4) переход на следующую строку должен выполняться только на границах лексем SQL-оператора, т.е. разрыв ключевых слов, имен таблиц, столбцов, идентификаторов и т. п. не допускается;
- 5) продолжение ввода SQL-оператора должно выполняться после приглашения `inl` (выдается порядковый номер строк продолжения):
`SQL> update DEPT_SUMMARY s
1>set NUM_EMPS = (
2>select count(1)
3>from EMP E
4>where E.DEPTNO = S.DEPTNO);`
- 6) пробелы, введенные в начале и/или в конце текста SQL-оператора, игнорируются;
- 7) строки, состоящие полностью из пробелов и/или знаков табуляции, игнорируются;
- 8) текст SQL-оператора должен заканчиваться знаком «`>`» (в конце текущей строки либо на следующей строке продолжения);
- 9) неотображаемый на консоли комментарий SQL-скрипта задается символом «`>`» в начале строки, например:
`; Это неотображаемый комментарий`
- 10) отображаемый на консоли комментарий SQL-скрипта задается двойным символом дефиса, например:

-- Это отображаемый комментарий
или с помощью команды «!» утилиты inl;
! Это отображаемый комментарий

- 11) Если в конце оператора за знаком «;» стоит отображаемый комментарий типа
`SELECT * FROM AUTO; -- Пример SQL-запроса;`

или любой текст (например),


`Table AUTO; //проверить выполнение;`

то этот комментарий или текст:

- должен заканчиваться знаком «;», если он относится к законченному SQL-оператору;
- не должен заканчиваться знаком «;», если это не последний оператор текста хранимой процедуры. В противном случае (если знак «;» отсутствует), то будут проигнорированы все строки до первой последующей строки, которая заканчивается знаком «;» включительно, либо до конца входного файла.

- 12) ввод SQL-оператора `CREATE PROCEDURE...` имеет свои особенности, связанные с тем, что тело хранимой процедуры задается на процедурном языке СУБД ЛИНТЕР, в котором разделение операторов также выполняется по символу «;». Чтобы избежать коллизий и не начать преждевременное выполнение SQL-оператора при обнаружении во вводимом тексте символа «;», необходимо для всех таких операторов задавать признак продолжения «//» ввода тела хранимой процедуры, как, например, в этом примере:

```
SQL>create procedure Proc1(in p int) result int
1> code
2> return p+3; //
3> end;
```

 Если текст триггера (хранимой процедуры) записывается в SQL-скрипте, признаки продолжения тела триггера (процедуры) не ставятся (см раздел CREATE PROC).

Результаты

В общем случае выдается следующая информация:

- 1) длительность обработки SQL-оператора (время начала и окончания обработки с точностью до минуты) – если установлен режим выдачи статистики (см. команду COUNT);
- 2) код завершения и его расшифровка, указывающие на причину, по которой ядро СУБД ЛИНТЕР не смогло выполнить SQL-оператор (см. документ РСКЮ.10112-60 32 01-5 «Справочник кодов завершения»), например:

INL: состояние выполнения: 1503
Таблица уже существует

В частных случаях дополнительно выдается следующая информация:

<u>SQL-оператор</u>	<u>Дополнительная информация</u>
INSERT	Количество добавленных в таблицу записей INL: добавлено строк : nnn
DELETE	Количество реально удаленных из таблицы записей

<u>SQL-оператор</u>	<u>Дополнительная информация</u>
UPDATE	INL: удалено строк : nnn Количество реально измененных в таблице записей
SELECT	INL: изменено строк : nnn Количество реально выданных строк ответа в виде
GET EVENT	INL: выдано строк : nnn Текущее состояние указанного события (событие установлено/ событие не установлено)
TEST TABLE	Результаты тестирования таблицы

Результаты поисковых запросов выводятся по умолчанию в следующем формате (Таблица 1):

Таблица 1. Отображение по умолчанию типов данных

Тип данных	Представление по умолчанию
SMALLINT	Отрицательные значения выводятся со знаком '-', положительные – без знака. Для представления положительных чисел со знаком надо использовать функцию <code>to_char()</code> , например, <code>select to_char(weight,'s99999') from auto;</code>
INT	Аналогично <code>smallint</code>
BIGINT	Аналогично <code>smallint</code>
CHAR	Ширина выводимого столбца всегда равна размеру столбца в таблице
VARCHAR	По умолчанию ширина выводимого столбца всегда равна размеру столбца в таблице. Если количество выбранных символов меньше размера столбца в таблице, то недостающие символы отображаются в виде пробелов справа. Для отмены расширения столбца используется ключ <code>-nf</code>
BYTE	Ширина выводимого столбца всегда равна размеру столбца в таблице
VARBYTE	По умолчанию ширина выводимого столбца всегда равна размеру столбца в таблице. Если количество выбранных байтов меньше размера столбца в таблице, то недостающие байты отображаются в виде пробелов справа. Для отмены расширения столбца используется ключ <code>-nf</code>
REAL	Данные отображаются в формате <code>[s]99999999.9999</code> без выравнивания по десятичной точке, незначащие нули справа и слева отбрасываются, например, 123.456 123.4567 -12.06 0.6999 45

Тип данных	Представление по умолчанию
DOUBLE	<p>Данные отображаются в формате [s]9999999999999999.999999999 без выравнивания по десятичной точке, незначащие нули справа и слева отбрасываются, например,</p> <pre> 123.456 123.456789123 - 23.406 0.69 678885 </pre>
DECIMAL (NUMERIC)	<p>Данные отображаются в формате [s]9999999999999999999.999999999 с выравниванием по десятичной точке, незначащие нули справа (кроме первого после точки) и слева отбрасываются, например,</p> <pre> 123.456 123.456789123 12.06 0.6999 643.0 -796.00954 </pre>
DATE	<p>Данные отображаются в формате дд.мм.гггг:чч:ми:сс:тт, например,</p> <pre>24.08,2002:14:24:39:00</pre>
BOOLEAN	<p>Данные отображаются в формате Т или F: Т – true (0), F- false (1)</p>
NCHAR	Представление UNICODE-данных в текущей кодовой странице inl
NCHAR VARYING	<p>Представление UNICODE-данных в текущей кодовой странице inl. Если количество выбранных символов меньше размера столбца в таблице, то недостающие символы отображаются в виде пробелов справа. Для отмены расширения столбца используется ключ -nf</p>
BLOB	<p>Системный описатель BLOB-столбца в виде:</p> <ul style="list-style-type: none"> • размер BLOB-файла; • номер первой порции; • номер последней порции; • число BLOB-файлов; • дд.мм.гггг дата – дата создания фразового индекса (если он создан) или нулевая дата - в противном случае; • тип BLOB-данных (текст, графика, видео и т.п.), например: <pre> 0000572006 000002 000143 1 22.09.2002 003 </pre>
EXTFILE	Спецификация внешнего файла (символьная строка длиной 512 символов)

 Для просмотра данных типа CHAR, VARCHAR, BYTE, VARBYTE, NCHAR, NCHAR VARYING большой размерности рекомендуется использовать табличный режим (см. команду BROWSE).

Редактирование

Для редактирования текста SQL-оператора:

1. выполнить команду EDIT сразу после выполнения SQL-оператора, текст которого должен быть изменен;
2. откорректировать текст SQL-оператора;
3. выполнить команду EXEC для выполнения откорректированного SQL-оператора.

Для редактирования SQL-скрипта:

1. выполнить команду EDIT сразу после выполнения SQL-скрипта, текст которого должен быть изменен;
2. откорректировать текст SQL-скрипта;
3. сохранить, при необходимости, текст откорректированного SQL-скрипта в файле;
4. выполнить SQL-скрипт.

SQL-операторы с параметрами

Inl поддерживает обработку параметров для следующих SQL-операторов:

- SELECT;
- INSERT;
- DELETE;
- UPDATE;
- EXECUTE | CALL .

Общие правила

Обработка SQL-операторов с параметрами выполняется по следующим правилам:

- 1) допустимыми являются как именованные параметры (:<имя параметра>), так и неименованные (?), например:

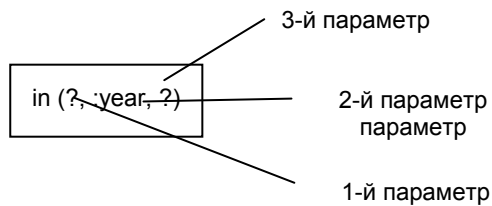
```
select make from auto where year=:year;
select make from auto where year>?;
```

- 2) SQL-оператор может включать одновременно именованные и неименованные параметры, например:

```
select make from auto where year between :beg_year and
:end_year and color=?;
```

- 3) подсчет параметров ведется слева направо по тексту SQL-оператора, например:

```
select make from auto where year in (?, :year, ?);
```



- 4) общее количество уникальных именованных параметров в одном SQL-операторе - не больше 255;
- 5) длина значения вводимого параметра не должна превышать 4 Кбайт;
- 6) максимальная суммарная длина значений всех параметров – 16 Кбайт;
- 7) при обработке SQL-оператора с параметрами `inl` запрашивает ввод значения для каждого параметра. При этом для именованных параметров в качестве подсказки выдается имя и в скобках тип и, если необходимо, длина параметра, а для неименованных – порядковый номер и тип, например:

```
SQL>SELECT count(model) FROM AUTO WHERE COLOR=?
```

```
1>AND YEAR = :YEAR;
```

```
Параметр 1 (CHAR(10))>YELLOW
```

```
YEAR (INTEGER)>71
```

```
|          38|
```

```
INL : выдано строк :1
```

```
SQL>select distinct make,year from auto
```

```
1>where ?(char(4))= make;
```

```
SQL>insert into auto( personid,make,year,model) values
```

```
1>(?, :make, :year, :model);
```

```
SQL>delete from auto where year<= ?;
```

- 8) если в SQL-операторе есть несколько одноименных именованных параметров, то запрос на ввод значения для данного именованного параметра выполняется один раз, например:

```
SQL>select count(model) from auto where weight between
```

```
1>:base_weight-500 and :base_weight+500;
```

```
BASE_WEIGHT (INTEGER)>1000
```

```
|          19|
```

```
SQL>exec
```

```
BASE_WEIGHT (INTEGER)>1500
```

```
|          54|
```

```
SQL>
```

- 9) присвоение параметру NULL-значения выполняется с помощью ввода строки `NULL` (все заглавные буквы). Для присвоения символьным строкам значения `'NULL'` необходимо использовать отличную от строки `NULL` транслитерацию, например, `null`, `Null`;

```
SQL>update auto set color=? where personid=:person;
```

```
Параметр1 (CHAR(10))>NULL
```

```
PERSON (INTEGER)>34
```

```
SQL>exec
```

```
Параметр1 (CHAR(10))>>null
```

```
PERSON (INTEGER)>35
```

```
SQL>select color from auto where personid in (34,35);
|           |
|null      |
```

- 10) для хранимых процедур запрашивается ввод значений только для параметров с атрибутами IN, INOUT. Список параметров процедуры является позиционным, поэтому, если в списке параметров присутствует параметр типа OUT, то в SQL-операторе с параметрами он должен быть позиционирован либо как необрабатываемый (пропущенный), знаком «запятая» (в конце списка параметры типа OUT можно не задавать), либо как обычный параметр (но запрос на ввод значения такого параметра выдаваться не будет), например:

<u>Список параметров процедуры</u>	<u>SQL-оператор с параметрами</u>
proc1 (in p1 int, out p2 int, inout p3 char(25), in p4 date);	execute proc1(?, ?, ?, ?); execute proc1(?, ?, ?, ?);
proc2 (out p1 int, out p2 int, inout p3 char(25), in p4 date);	execute proc2(, , :p3, ?); execute proc2(?, ?, :p3, ?);
proc3 (inout p1 int, in p2 int, out p3 char(25), out p4 date);	execute pro32(:p1, :p2);

- 11) нельзя использовать параметры для столбцов типа BLOB и EXTFILE;
- 12) если SQL-оператор с параметрами представлен в виде SQL-скрипта, то значения параметров должны задаваться в этом же скрипте сразу после текста SQL-оператора, каждое значение на отдельной строке. Для именованных повторяющихся параметров значение должно задаваться один раз, например:

Текст SQL-скрипта:

```
SELECT model, "Подзапрос5".name
FROM AUTO,
(SELECT name, personid FROM person where sex=?) as
"Подзапрос5"
where color=? and year=?
and auto.personid="Подзапрос5".personid;
M
YELLOW
70
```

Формат ввода параметров

В Таблица 2 приведены форматы ввода параметров.

Таблица 2. Форматы ввода параметров

Тип параметра	Формат ввода
SMALLINT	[s]999999
INT	[s]999999
BIGINT	[s]999999

Тип параметра	Формат ввода
CHAR	Символьная строка длиной не более 100 знаков, символы вводятся в текущей кодовой странице inl и не преобразуются (двойные кавычки для отмены преобразования не требуются)
VARCHAR	Символьная строка длиной не более 100 знаков, символы вводятся в текущей кодовой странице inl и не преобразуются (двойные кавычки для отмены преобразования не требуются)
BYTE	Символьная строка шестнадцатеричных цифр длиной не более 100 байт . Каждый байт должен быть представлен 2 символами шестнадцатеричных цифр. Между парами шестнадцатеричных цифр допускается любое число пробелов. Внутри пары шестнадцатеричных цифр разделители не допускаются. Общее количество шестнадцатеричных цифр должно быть четным
VARBYTE	Символьная строка шестнадцатеричных цифр длиной не более 100 байт. Правила ввода аналогичны формату byte
REAL	[s]99999999.9999 Незначащие нули можно не вводить
DOUBLE	[s]9999999999999999.999999999 Незначащие нули можно не вводить
DECIMAL (NUMERIC)	[s]99999999999999999999.999999999 Незначащие нули можно не вводить
DATE	Формат дд.мм.гггг[:чч:[ми:[сс:[тт]]]]
BOOLEAN	Символ T или F: T – true , F- false
NCHAR	Символьная строка длиной не более 100 цифровых символов, задающая UNICODE-значение в текущей кодовой странице inl (т.е. не более 50 UNICODE-символов)
NCHAR VARYING	Символьная строка длиной не более 100 цифровых символов, задающая UNICODE-значение в текущей кодовой странице inl (т.е. не более 50 Unicode-символов)

Пример 1

```
SQL>SELECT count(model) FROM AUTO WHERE COLOR=?
1>AND YEAR = :YEAR;
Параметр 1 (CHAR(10))>YELLOW
YEAR (INTEGER)>71
|          38|
INL : выдано строк   :1
SQL>exec
Параметр 1 (CHAR(10))>GREEN
YEAR (INTEGER)>70
|          20|
INL : выдано строк   :1
```

Пример 2

```
SQL>SELECT count(model)FROM AUTO WHERE ?=COLOR
1>AND year=:YEAR;
Параметр 1 (CHAR(10))>YELLOW
YEAR (INTEGER)>71
|                38|
```

Пример 3

```
SQL>SELECT * FROM AUTO WHERE COLOR=? AND YEAR = :YEAR;
Параметр 1 (CHAR(10))>ELLOW
YEAR (SMALLINT)>71
```

Пример 4

```
SQL>SELECT * FROM AUTO WHERE HORSEPWR=:A OR DSPLCMNT=:A;
A (SMALLINT)>250
```

Команды

!

Формат

! [символьная строка]

Назначение

Вывод на экран терминала или в выходной файл символьной строки.

Описание

При обработке SQL-скриптов часто возникает необходимость вести протокол выполнения SQL-скрипта, т.е. выдавать на экран терминала (или в выходной файл) сообщения о том, какая работа выполняется в текущий момент или уже была выполнена, Это можно сделать с помощью комментариев, которые явно записываются в SQL-скрипт и выводятся по команде !. Дополнительно команда ! может использоваться для форматирования результатов поисковых SQL-запросов, выводимых в файл в виде простых справок (отчетов).

Команда обрабатывается следующим образом:

- текст комментария выводится на экран терминала. Если используется средство операционной системы для перенаправления стандартного вывода – то в указанный выходной файл;
- если команда задается в интерактивном режиме, то текст комментария просто дублируется на экране. Таким способом можно протоколировать интерактивную работу пользователя;
- текст выводится в том виде, в каком он задан, начиная от символа ! и до конца строки. Только первый встретившийся в строке символ ! воспринимается как команда, все остальные – как обычные символы;
- команда ! воздействует только на одну строку SQL-скрипта. Перенос комментария не допускается (необходимо в каждой продолжающейся строке комментария повторять команду !);
- использование комментария внутри текстов SQL-запросов запрещается (в общем случае будет фиксироваться синтаксическая ошибка).

Примеры

1. Пусть содержимое SQL-скрипта `sample.sql` будет следующее:

```
! *** выполняется запрос - que1.sql
_que1.sql
! *** выполняется запрос - que2.sql
_que2.sql
! *** выполняется запрос - que3.sql
_que3.sql
```

Протокол выполнения скрипта `sample.sql` на экране терминала будет выглядеть следующим образом:

```
SQL> _sample
```

```

*** выполняется запрос - que1.sql
*** выполняется запрос - que2.sql
*** выполняется запрос - que3.sql
SQL>

```

Если SQL-скрипт sample.SQL запущен на выполнение как

```
INL -u SYSTEM/MANAGER -f c:\test_INL\sample >prtc01.txt
```

то протокол выполнения будет записываться в файл prtc01.txt, создаваемый в том же каталоге, в котором выполняется inl.

2. Пример использования команды ! для форматирования ответа поискового запроса. Пусть SQL-скрипт имеет следующий вид:

```

time
!                               Справка
! об автомобилях выпуска до 1970 года
select make as "Изготовитель", model as "Модель"
from auto where year <=70;


```

Команда TIME добавлена, чтобы исключить из справки временные параметры выполнения SQL-запроса. Тогда справка будет выдана в следующем виде:

```

                               Справка
об автомобилях выпуска до 1970 года
Изготовитель                   Модель
-                               -
|ALPINE                        |A-310                            |
|CHRYSLER                      |DODGE CORONET CUSTOM            |
|AMERICAN MOTORS              |GREMLIN X                        |
|GENERAL MOTORS                |OLDSMOBILE 98                    |
. . .

```

 Комментарии, задаваемые по команде !, являются исполняемыми, т.е. всегда выводятся на экран или в выходной файл. Для комментирования (документирования) текстов SQL-скриптов можно использовать неисполняемые комментарии. Они задаются следующим образом:

<однострочный SQL-оператор> [!]<комментарий>;

Например:

```
create role ROLE1 for XXXX ; Роль "Начальник отдела";
```

или

```
create role ROLE1 for XXXX; ! Роль "Начальник отдела";
```

причем и SQL-оператор и комментарий к нему должны размещаться на одной строке. Комментарий должен заканчиваться знаком '!'. Знак '!' (признак исполняемого комментария) можно не задавать – он все равно будет проигнорирован.


ALTER PROC

Формат

ALTER PROC <спецификация файла>

Назначение

Модификация хранимой процедуры из файла.

 Команда для модификации триггера не предусмотрена. Для выполнения этой операции триггер необходимо удалить, а затем создать заново.

Описание

Команда ALTER PROC заменяет текст существующей в БД хранимой процедуры.

<Спецификация файла> задает местоположение файла с новым исходным текстом хранимой процедуры.

Если тип расширения в <спецификации файла> не задан, подразумевается .sql.

Для выполнения команды необходимы следующие условия:

- 1) в БД должны присутствовать системные таблицы \$\$\$PRCD, \$\$PROC;
- 2) модифицируемая процедура должна существовать в БД;
- 3) текст модифицируемой процедуры в исходном файле должен начинаться с ключевого слова PROCEDURE. Другие ключевые слова (типа CREATE) недопустимы.

BROWSE

Формат

BROWSE

Назначение

Смена режима отображения информации.

Описание

Inl поддерживает два способа отображения информации, получаемой в результате выполнения поисковых SQL-операторов: строчный и табличный.

Специфика строчного режима:

- каждая строка ответа сразу выдается на экран терминала или записывается в выходной файл (если при запуске inl был сделано перенаправление стандартного вывода);
- если строка ответа не уместается на экране терминала, то она продолжается на следующих строках экрана;
- строки ответа выдаются на экран с той скоростью, с какой происходит выборка информации из базы данных;
- приостанов выдачи информации (например, для просмотра или анализа) выполняется стандартными для операционной системы средствами управления экраном терминала;
- если выданная строка ответа ушла «за экран», то доступ к ней уже невозможен.
- нет возможности просматривать данные типа BLOB и EXTFILE.

Пример выдачи ответов в строчном режиме

1) строка ответа полностью помещается в строке экрана (см. рис. 1):

```
SQL>select make, model, color from auto
1> where personid in (10,20,30);
```

```

MAKE              MODEL              COLOR
-----
!AMERICAN MOTORS  !MATADOR STATION  !WHITE
!FORD              !MERCURY MONTEREY !BLACK
!FORD              !LTD COUNTRY SQUIRE !GREEN
INL : выдано строк      : 3

```

Рис. 1 – Короткая строка ответа

2) строка ответа располагается в нескольких строках экрана (см. рис. 2)

```
SQL>select * from auto where personid in (10,20,30);
```

```

MAKE      DSPLCMNT  WEIGHT  MODEL      COLOR      YEAR  BODYTYPE  SERIALNO  CYLNDERS  HORSEPWR
HKMILE    PERSONID
-----
!AMERICAN MOTORS  !MATADOR STATION  !STATION WAGON  !
50!          304!      3725!WHITE  !          70!P298565792069809!  8!      720122!
!FORD              !MERCURY MONTEREY  !SEDAN HARDTOP  !
08!          430!      4500!BLACK  !          71!K670210003057005!  8!      720202!
!FORD              !LTD COUNTRY SQUIRE  !STATION WAGON  !
63!          352!      4505!GREEN  !          71!M787560194860222!  8!      720205!
INL : выдано строк      : 3

```

Рис. 2 – Длинная строка ответа

Очевидны отрицательные стороны этого режима:

- нет возможности вернуться к уже выданным ответам;
- отсутствует скроллинг по горизонтали в ответе, который представляет собой широкую таблицу, сложно проследить значения столбцов, не входящих (по ширине) в одну строку терминала.

Специфика табличного режима:

- inl сначала запоминает в собственном буфере столько ответов, сколько позволяет доступная оперативная память;
- после заполнения буфера выдает строки ответа в удобной экранной форме;
- поддерживается скроллинг по горизонтали/вертикали (не уместившиеся в буфере ответы подкачиваются по мере необходимости);
- поддерживается возможность манипулирования данными (добавление, удаление, модификация) для обновляемых SELECT-запросов;
- возможность просмотра данных типа BLOB и EXTFILE;

- форма представления информации удобна для визуализации ответа, но работа, в общем случае, выполняется несколько медленнее по сравнению со строчным режимом, особенно при первоначальном приеме ответов.

Если в строчном режиме вывод результата перенаправлен с экрана в выходной файл, то длинные ответы помещаются в файл полностью, без разделения на отдельные строки. В этом случае при просмотре такого файла стандартными средствами операционной системы возможен скроллинг по горизонтали/вертикали и строчный режим по своим визуальным возможностям приближается к табличному.

Табличный режим действует только на те SQL-запросы, которые потенциально возвращают множественный ответ, а именно:

- SELECT-запросы;
- хранимые процедуры, возвращающие в качестве ответа курсор;
- оператор TEST TABLE ..., возвращающий набор диагностических сообщений.

Для всех остальных SQL-операторов всегда используется строчный режим вывода информации (даже если явно установлен табличный режим).

Пример выдачи ответов в табличном режиме (рис. 3).

MAKE	MODEL	BODYTYPE	CYLINDERS	PRICE
AMERICAN MOTORS	PATRAGON STATION	STATION WAGON	4	154
ACURA	CORD	COUPE	4	244
CHRYSLER	BOBCE CORONET CUSTOM	STATION WAGON	0	255
PONCHIKO-BENZ	290 SE	SEDAN	6	154
AMERICAN MOTORS	DEERLIN X	SEDAN	4	144
GENERAL MOTORS	OLDSMOBILE 98	SEDAN HATCHTOP	4	225
GENERAL MOTORS	CADILLAC DE WILLE	SEDAN HATCHTOP	0	220
AMERICAN MOTORS	PATRAGON STATION	STATION WAGON	4	154
DE TOMARO	PANLEH	COUPE	4	285
GENERAL MOTORS	BUICK SKYLARK 40	COUPE HATCHTOP	4	154
AMERICAN MOTORS	JAWELIN 400	COUPE HATCHTOP	0	220
GENERAL MOTORS	BUICK SKYLARK 40	COUPE HATCHTOP	4	154
GENERAL MOTORS	OLDSMOBILE 98	SEDAN HATCHTOP	4	225
CHRYSLER	BOBCE CORONET CUSTOM	STATION WAGON	4	255
VOLVO	4 DR	COUPE	0	120
CHRYSLER	BOBCE CHALLENGER SIX	COUPE HATCHTOP	6	150
VA-HUNSCHE	HL 76	COUPE	6	184
FORD	HOUSTON MONTELU	SEDAN HATCHTOP	4	244

F1 HELP F2 TABLE/CARD F3-> DELETE F4-> APPEND ESC EXIT

Рис. 3 – Просмотр результатов в табличном режиме

Модификация данных в табличном режиме

В табличном режиме можно выполнять обновление данных (добавление, удаление, корректировку), полученных в результате выполнения обновляемого SELECT-запроса. SELECT-запрос считается обновляемым, если выполняются следующие условия:

- данные выбираются только из одного экземпляра таблицы или представления;
- таблица не является системной таблицей БД;
- представление порождено не из системной таблицы;
- результатом выполнения SELECT-запроса являются значения столбцов таблицы (представления), а не вычисляемые выражения агрегатных функций;
- запрос не содержит литералов.

Команда BROWSE работает как циклический двоичный переключатель, т.е. каждое выполнение BROWSE отменяет текущий режим отображения и устанавливает противоположный. Установленный режим сохраняется до конца работы inl.

При запуске `inl` по умолчанию устанавливается строчный режим отображения.

Для просмотра текущего режима отображения используется команда `show`:

- положение **ВКЛ.** – режим браузера (табличный режим);
- положение **ВЫКЛ.** – терминальный (строчный) режим.

DISFILL

Формат

DISFILL

Назначение

Запрещает/разрешает заполнение пробелами полей переменной длины строки ответа до максимальной длины столбца.

Описание

Команда `DISFILL` запрещает/разрешает дополнять пробелами столбец строки ответа переменной длины до заданной при создании таблицы максимальной длины.

Команда работает как переключатель – изменяет текущий режим на противоположный.

По умолчанию при запуске `inl` установлен режим вывода максимальной длины столбца.

Пример

```
create table tst(vc varchar(10), c char(5));
insert into tst (vc, c) values('1','1');
insert into tst (vc, c) values('11','11');
insert into tst (vc, c) values('111','111');
-- выдача столбцов ответа по максимальной длине (режим по
умолчанию)
select vc, c from tst;
vc      c
--      -
|1      |1      |
|11     |11     |
|111    |111    |
INL : выдано строк      : 3

-- выдача столбцов ответа по фактической длине
disfill
select vc, c from tst;
vc      c
--      -
|1|1    |      |
|11|11  |      |
|111|111|      |
INL : выдано строк      : 3
```

CODEPAGE

Формат

CODEPAGE <кодовая страница>

Назначение

Установка кодовой таблицы.

Описание


Для СУБД ЛИНТЕР версии до 5.9 включительно допустимыми являются кодовые страницы 866, 1251 или KOI8.

Результатом выполнения команды **CODEPAGE** является информирование сервера СУБД ЛИНТЕР о том, в какой кодировке работает программа `inl`. Это означает, что сервер ожидает от `inl` символьные данные в заданной кодировке и, соответственно, в этой же кодировке должен возвращать ей результаты SQL-запросов. При этом данные в самой БД могут храниться в любой другой кодировке. Ответственность за правильное кодирование входных данных возлагается на `inl`. СУБД ЛИНТЕР не проверяет соответствие декларированной кодовой страницы и фактически использованной, поэтому, в случае их несовпадения, в БД могут быть записаны неправильные символьные данные.

Команда **CODEPAGE** действует до конца текущего сеанса `inl` или до выполнения новой команды **CODEPAGE**.

По умолчанию при запуске `inl` используется кодовая страница 866.

Информация о текущей кодовой странице выдается по команде **LIST**.

 Если в операционной системе установлена переменная окружения `LINTER_CP`, то команда **codepage** игнорируется. Текущей кодовой страницей всегда будет являться кодовая страница, заданная `LINTER_CP`.

Примеры

1. Пусть в таблице `Tst_Code` символьные данные хранятся в кодировке KOI8-R. Для их корректного отображения необходимо установить эту же кодировку.

```
SQL>codepage KOI8-R
SQL>select * from "Tst_Code";
```

2. Получить список поддерживаемых СУБД кодовых страниц и установить нужную.

```
SQL>select name from $$$charset;
| DEFAULT
| CP866
| KOI8-R
| CP1251
| CP437
| CP1252
| CP8859-1
| CP8859-2
```

```
...
SQL>codepage KOI8-R
```

COUNT

Формат

COUNT

Назначение

Разрешение/запрет выдачи итоговой статистики.

Описание

Результаты обработки SQL-запросов, относящихся к обработке собственно данных, `inl` по умолчанию сопровождается дополнительной статистической информацией, позволяющей оценивать как временные характеристики выполнения запроса, так и его семантическую корректность (количество реально выбранных /обработанных записей), например:

`INL : удалено строк : 23`

или

`INL : выдано строк : 12`

Если вывод статистической информации нежелателен, то его можно запретить, установив переключатель итоговой статистики в положение **ВЫКЛ.** (выключен).

Если этот переключатель установлен в положение **ВКЛ.** (включен), то ответы на запросы обработки данных будут сопровождаться итоговой статистикой.

Управление переключателем итоговой статистики выполняет команда `COUNT`, которая меняет текущее состояние переключателя на противоположное, т.е. **ВКЛ./ВЫКЛ.** на **ВЫКЛ./ВКЛ.** соответственно.

Команда `COUNT` воздействует только на SQL-операторы:

- `SELECT`;
- `INSERT`;
- `DELETE`;
- `UPDATE`;
- `EXECUTE PROCEDURE` (если возвращается тип данных «курсор»).

Сразу после запуска `inl` переключатель итоговой статистики по умолчанию установлен в положение **ВКЛ.**

Для просмотра текущего состояния переключателя статистики используется команда `LIST`.

Примеры

1. В процессе отладки SQL-запроса для подсчета найденных записей можно вместо встроенной в SQL функции `count` использовать команду `COUNT`.

```
SQL>select count(personid) from auto where make='FORD';
|           118           |
INL : выдано строк           :1
```

```
SQL>select personid from auto where make='FORD';
|          1          |
|          20         |
|          22         |
|          30         |
|
|          981        |
|          983        |
|          997        |
INL  :  выдано строк      :118
```

2. Получить время выполнения SQL-запроса.

```
SQL>list
```

```
...
```

```
count      :  вкл.
```

```
...
```

```
SQL>update auto set year=year+1900;
```

```
INL: начальное время : 17:54:54 конечное время : 17.54.56
```

```
INL: изменено строк   : 1000
```

CREATE PROC

CREATE TRIG

Формат

```
CREATE {PROC |TRIG} <спецификация файла>
```

Назначение

Создание триггера или хранимой процедуры из файла.

Описание

Команда CREATE PROC создает в БД новую хранимую процедуру.

Команда CREATE TRIG создает в БД новый триггер таблицы.

<Спецификация файла> задает местоположение файла с исходным текстом триггера или хранимой процедуры.

Для выполнения команды необходимы следующие условия:

- 1) в БД должны присутствовать системные таблицы \$\$\$TRIG (при создании триггера) и \$\$\$PRCD, \$\$PROC (при создании процедуры);
- 2) имя создаваемого триггера (процедуры) должно быть уникальным;
- 3) текст процедуры в исходном файле должен начинаться с ключевого слова PROCEDURE, текст триггера – со слова TRIGGER. Другие ключевые слова (типа CREATE, ALTER) недопустимы.

Пример

```

Файл crt_proc.sql (исходный текст хранимой процедуры):
procedure test_proc ()
declare
    exception noresults for custom 100;
code

    execute direct "create or replace table tab (id1 int, id2
int, s char(10));";
    if errcode() <> 0 then
        signal noresults;
    endif
exceptions
    when others then
        resignal;
end

! Создание процедуры из файла
SQL>create proc crt_proc.sql

! Выполнение созданной процедуры
SQL>execute test_proc();

```

DISREP**Формат**

```
DISREP
```

Назначение

Запрещает/разрешает выводить результаты выполнения SQL-запросов и хранимых процедур.

Описание

Команда **DISREP** аналогична команде **COUNT**, но ее действие распространяется и на вывод результата хранимых процедур (если возвращается некурсорный тип данных).

Команда работает как переключатель – изменяет текущий режим на противоположный.

По умолчанию при запуске `inl` установлен режим вывода результатов.

Пример

```

SQL>create procedure proc(in p int) result int
1>code
2> return p+3;//
3>end;
SQL>execute proc(2);
return value = 5

```

```
SQL>disrep
SQL>execute proc(2);
SQL>
```

DBINFO

Формат

DBINFO

Назначение

Получение справочной информации о параметрах запуска СУБД.

Описание

Команда **DBINFO** выдает информацию о параметрах запуска ядра СУБД ЛИНТЕР. Она может использоваться для следующих целей:

- 1) эмпирический подбор параметров запуска СУБД ЛИНТЕР для оптимизации выполнения как отдельных клиентских приложений (автономная оптимизация), так и совокупности нескольких, одновременно работающих с БД, приложений (комплексная оптимизация). Это процесс выполняется, как правило, следующим образом:
 - предварительно теоретически, на основе анализа потока SQL-запросов клиентского приложения, оцениваются размеры очередей таблиц, столбцов, памяти ядра и других параметров, влияющих на эффективность работы ядра СУБД;
 - на макете (или на готовом клиентском приложении) практически оценивается эффективность выбранных параметров. Путем варьирования значений параметров и отслеживания их влияния на эффективность обработки подбираются оптимальные значения.
- 2) задавать правильные режимы работы с БД, например:
 - если при запуске СУБД был отключен режим транзакций (эта информация, в числе прочей, выдается по команде **DBINFO**), то отказ от ошибочно измененной в БД информации будет невозможен;
 - если выполняется отладка клиентского приложения, то иногда полезно вести протокол обращения к БД (файл `linter.log`), в котором фиксируются все операции с данными, выполненные ядром СУБД. При эксплуатации клиентских приложений ведение протокола обращений приводит только к дополнительным затратам времени на обработку SQL-запросов и поэтому не имеет смысла. Информация о протоколе обращения также выдается по команде **DBINFO**.

Пример

```
SQL>dbinfo
Информация о базе данных 'TicketTracking'
СУБД ЛИНТЕР версия                :6.0.472
Размер памяти ядра                 :500
Размер очереди каналов             :100
Размер очереди таблиц              :100
```

Размер очереди колонок	:500
Размер очереди файлов	:30
Размер очереди пользователей	:100
Журнал транзакций	:включен
Размер памяти сортировки	:100
Количество процессов сортировки	:1
Протокол обращений (LINTER.LOG)	:выключен

В строке «СУБД ЛИНТЕР версия» третье число обозначает номер сборки.

ЕCHO

Формат

ECHO {ON | OFF | ERROR}

Назначение

Трассировка SQL- запросов.

Описание

Команда позволяет осуществлять вывод в стандартный поток вывода всех или только ошибочных запросов выполняемого SQL-скрипта..

При указании опция ON выводятся тексты всех SQL-запросов до синтаксического разбора.

При указании опция OFF тексты SQL-запросов не выводятся (режим по умолчанию).

При указании опция ERROR выводятся тексты только тех SQL-запросов, при выполнении которых произошла ошибка. Если для SQL-запроса действует команда игнорирования ошибочного кода завершения (IGNORE), то такие запросы не трассируются.

Аналогично трассируются SQL-запросы их хранимых процедур.

Пример

```

файл tst.sql:
echo off;
! >>> трассировка не выполняется
drop table $$$LEVEL;
echo on;
! >>>выполняется трассировка всех запросов
drop table $$$LEVEL;
echo error;
! >>>выполняется трассировка тоько ошибочных запросов
drop table $$$LEVEL;

```

Запуск SQL-скрипта:

```

inl -u SYSTEM/MANAGER -f tst.sql > trace.sql
Результат выполнения скрипта (файл trace.sql):

```

```
>>> трассировка не выполняется
INL : начальное время : 14:54:07 конечное время : 14:54:08
INL : состояние выполнения : 83
нельзя явно изменять системную таблицу
>>> выполняется трассировка всех запросов
drop table $$$LEVEL;
INL : начальное время : 14:54:08 конечное время : 14:54:08
INL : состояние выполнения : 83
нельзя явно изменять системную таблицу
echo error;
>>> выполняется трассировка только ошибочных запросов
INL : начальное время : 14:54:08 конечное время : 14:54:08
INL : состояние выполнения : 83
drop table $$$LEVEL;
нельзя явно изменять системную таблицу
```

EDIT

Формат

EDIT

Назначение

Интерактивное создание/редактирование SQL-скрипта.

Описание

Команда **EDIT** используется для интерактивного (т.е. не выходя в операционную систему) создания или редактирования SQL-скрипта во внешнем буфере `inl`.

Все SQL-запросы и команды, которые влияют на отображение результатов выполнения SQL-запросов, `inl` запоминает и хранит в своем рабочем буфере размером 8 Кбайт (в MS DOS – 4 Кбайт). Таким образом, максимальная длина SQL-запроса, обрабатываемого программой `inl`, равна 8 Кбайт.

Рабочий буфер всегда (кроме самого первого запуска `inl` после установки СУБД ЛИНТЕР) содержит два объекта: последнюю выполненную команду (**outfil**, **header**, **unload** или **!**) и последний (введенный пользователем) или выполненный программой `inl` SQL-запрос.

После выполнения SQL-запроса его текст из буфера не удаляется, поэтому можно:

- повторить его выполнение необходимое число раз;
- отредактировать и выполнить;
- сохранить в отдельном файле для будущего использования.

Повторное выполнение SQL-запроса выполняется по команде **EXEC**.

Процесс подготовки и выполнения SQL-запроса в интерактивном режиме проходит в следующей последовательности:

1. выполнить команду **edit**.

```
SQL>edit
```

после чего inl:

- удаляет из текущего каталога файл inl.buf (если он там есть);
- открывает на запись новый файл inl.buf (все в том же текущем каталоге);
- переписывает из внутреннего буфера в файл inl.buf последний выполненный (введенный) SQL-запрос и, если есть, последнюю команду форматирования;
- закрывает файл inl.buf и вызывает на выполнение стандартный системный редактор текстовых файлов (см. рис. 4) (например, в MS Windows это Notepad), который открывает файл inl.buf.

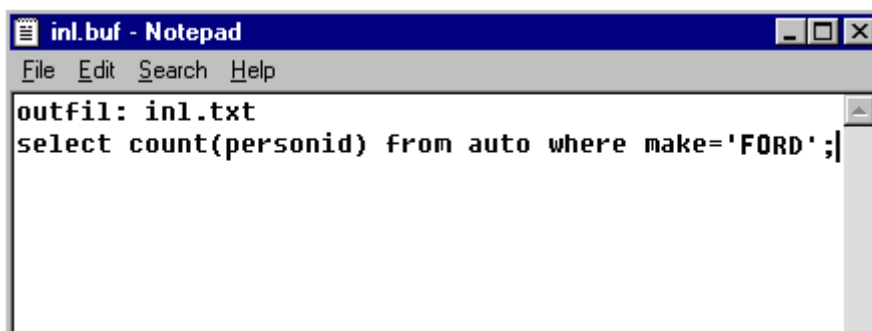


Рис. 4 – Окно текстового редактора

2. ввести (если файл inl.buf пуст) или откорректировать (при необходимости) с помощью стандартных средств редактора текст SQL-скрипта;
3. сохранить при необходимости созданный (откорректированный) SQL-скрипт в отдельном текстовом файле;
4. выйти из редактора. Inl переписывает содержимое inl.buf в свой внутренний буфер. После этого созданный (откорректированный) SQL-скрипт можно выполнить сразу же или позднее по команде EXEC;
5. при активизации редактора вместо открываемого по умолчанию файла inl.buf можно с помощью стандартных средств самого редактора открыть любой ранее созданный SQL-скрипт, текст которого будет сохранен при закрытии редактора по умолчанию в inl,buf.

Пример

```
SQL> _/usr/home/step9.sql
```

```
count
outfil:paper.txt
unload:|
```

```
header:                Утверждаю-
                        Директор предприятия-
                        "Виртуал" Прошан И.Я.-
```

```
-
```

```
                Штатное расписание отд. N 111-
```

```
-----
|Таб.N | ФИО   |Должность | Оклад  |-
-----
```

```
SELECT Tn ,Fio ,Dolgn ,Ocl FROM Kadry WHERE Notd =111;
```

```
...
```

```
count
```

—

Формат

`_<спецификация файла>`

<Спецификация файла> – задает каталог, имя и расширение текстового файла, содержащего выполняемый SQL-скрипт.

Если в <спецификация файла> не указан каталог, то файл ищется в текущем каталоге, если не задано расширение файла по умолчанию предполагается `.sql`.

Назначение

Выполнение SQL-скрипта.

Описание

Команда «`_`» (символ подчеркивания) заставляет `inl` выбирать команды для выполнения не с терминала, а из указанного файла. Файл для команды «`_`» может содержать SQL-запросы и/или команды `inl`.

По достижении конца файла `inl` автоматически переключится на прием команд с терминала.

Входной файл – это обычный текстовый файл, который можно подготовить любым текстовым редактором, имеющимся в операционной системе. По умолчанию в `inl` в качестве входного файла установлен терминал пользователя.

Правила подготовки входного файла максимально просты: команды набираются так, как если бы они набирались в диалоговом режиме.

Этот файл может готовиться, естественно, не только текстовым редактором, но и любой программой, в том числе и средствами `inl` (см. команду `EDIT`).

Примеры

1. Заданный SQL-скрипт (`systab.sql`) ищется в подкаталоге `dict` корневого каталога (ОС Unix)

```
SQL>_../dict/systab
```

2. Данный пример демонстрирует запуск командного файла (`append.sql`). Этот файл содержит три запроса на добавление (в таблицы `AUTO`, `FINANCE` и `PERSON`) и контрольный запрос на поиск:

```
Page
```

```
Time
```

```
Count
```

```
! ***** Insert Into Table AUTO *****
```

```
INSERT INTO AUTO VALUES('VOLVO', '4 DR', 'COUPE', 8, 120, 118,  
2900, 'BLACK', 70, 'O262593464330109', 720303, 98224, 10001);
```

```
! ***** Ok. *****
```

```
! ***** Insert Into Table FINANCE *****
```

```
INSERT INTO FINANCE VALUES('AMERICAN EXPRESS', 300, 800,  
'ARCO', 1600, 5, 'HARTFORD INSURANCE', 21000, 'RUTGERS', '',  
'SEASHORE PROP.', 'SECURITY PACIFIC NATIONAL L.A.',10001);
```

```
! ***** Ok. *****
```

```
! ***** Insert Into Table PERSON *****
```

```

INSERT INTO Person (FIRSTNAM, PERSONID)
VALUES ('JENNIFER',10001);
! ***** Ok. *****
! ***** Test Query *****
SELECT COUNT(*) FROM AUTO, FINANCE, PERSON
WHERE Person.PersonId =10001 AND
Auto.PersonId=Person.PersonId AND
Finance.PersonId=Person.PersonId;


```

Для выполнения этого запроса нужно подать команду запуска:

```

SQL>_append
*****      Insert Into Table AUTO      *****
*****      Ok.      *****
*****      Insert Into Table FINANCE    *****
*****      Ok.      *****
*****      Insert Into Table PERSON    *****
*****      Ok.      *****
*****      Test Query      *****
| 1          |
SQL>

```

 Если сразу после запуска файла `append.sql` подать команду `EXEC`, то еще раз выполнится контрольный запрос на поиск, но без комментария (`***** Test Query *****`).

PESSIMISTIC

Формат

```
PESSIMISTIC
```

Назначение

Команда изменения режима обработки запросов.

Описание

При использовании сложных (много взаимосвязанных таблиц) и/или распределенных транзакций (т.е. транзакций, которые включают изменения данных более чем на одном узле) рекомендуется **PESSIMISTIC**-режим обработки данных.

При этом режиме все модификации пишутся сразу в БД, но все измененные (первым пользователем-инициатором **PESSIMISTIC**-режима) записи блокируются. Другие пользователи в этот момент не могут читать или модифицировать записи, измененные первым пользователем до конца транзакции.

Транзакция, модифицирующая данные в **PESSIMISTIC**-режиме, работает с блокировками **Exclusive-Lock** на уровне записей (т.е. другие пользователи не имеют доступа к записи, пока работающая с ней **Exclusive**-транзакция не подаст запрос **COMMIT** или **ROLLBACK**).

При выполнении **COMMIT** модификации, произведенные транзакцией, остаются в БД (в системный журнал БД только ставится отметка о конце транзакции).

При выполнении **ROLLBACK** модификации, произведенные транзакцией, удаляются из БД в соответствии с системным журналом.

EXEC

Формат

EXEC

Назначение

Выполнение SQL-запроса.

Описание

По команда **EXEC inl** выполняет SQL-запрос из своего внутреннего буфера (если он не пуст). Внутренний буфер **inl** может:

- быть пустым (в начале работы **inl**);
- содержать последний выполненный в текущем сеансе SQL-запрос;
- содержать текст отредактированного, но еще не выполненного SQL-запроса.

Для просмотра текущего состояния внутреннего буфера программы **inl** используется команда **LIST**.

Если в буфере хранился SQL-запрос с параметрами, то при выполнении **EXEC** необходимо каждый раз вводить значения параметров в интерактивном режиме или явно задавать их в SQL-скрипте.

Пример

```
SQL>time
SQL>count
SQL>select count(*) from auto where year=70;
|          465|
; редактирование SQL-запроса
SQL>edit
; просмотр внутреннего буфера INL
SQL>list
...
Запрос : select count(*) from auto where year=71;
...
SQL>exec
|          535|
SQL>
```

EXIT

Формат

EXIT


Назначение

Завершение работы программы `inl`.

Описание

При выполнении `EXIT`:

- если необходимо, выполняется `COMMIT` незавершенной транзакции;
- разрывается соединение с БД;
- завершается выполнение `inl` и управление передается в операционную систему.

 В интерактивном сеансе завершить работу `inl` можно, нажав клавиши `<Ctrl>+<C>`, затем клавишу `<Enter>`.

Пример

```
...  
; в среде MS Windows  
SQL>exit  
c:\>
```

FORMAT

Формат

FORMAT

Назначение

Переключатель форматирования.

Описание

`Inl` позволяет отображать результаты выполнения `SELECT`-запросов в двух видах:

- естественном – информация выводится в соответствии с типом данных, хранящихся в БД;
- символьном – выводимая информация преобразуется к символьному виду независимо от ее хранения в БД.

Команда `FORMAT` управляет сменой режимов переключателя форматирования (меняет текущий режим форматирования на противоположный).

Если переключатель форматирования включен (положение **вкл.**), то `inl` переводит ответы в символьный вид.

Команды

Если переключатель форматирования выключен (положение **ВЫКЛ.**), то `inl` оставляет ответы в естественном двоичном виде. Это требуется обычно при выводе ответов в файл для последующей работы с ним другими средствами или для визуального просмотра.

Первоначально по умолчанию переключатель имеет положение **ВЫКЛ.**

Пример

```
SQL>list
    -- Текущие установки INL --
...

format    :выкл.
...
SQL>select make,year,weight from auto;
|Make      |YEAR|WEJGHT  |
|FORD      | 71 | 2900 |
|ALPINE    | 70 | 1826 |
...
SQL>format
SQL>list
    -- Текущие установки INL --
...

format    :вкл.
...
SQL>select make,year,weight from auto;
|Make      |YEAR|WEJGHT  |
|FORD      |G   |T   |
|ALPINE    |F   |"   |
...
```

HEADER

Формат

```
HEADER: [<символьная строка>] [-] { [<символьная строка>] [-]...}
```

Назначение

Определение заголовка ответа.

Описание

По умолчанию заголовков ответа при выполнении `SELECT`-запроса представляет собой имена столбцов (или их синонимы) результирующей выборки, разграниченные заданным символом-разделителем. Если выбирается значение агрегатной функции или вычисляемого выражения, то в качестве имени используются пробелы (или синонимы выбираемых значений, если заданы).

С помощью команды **HEADER** можно заменить однострочный заголовок таблицы ответов, формируемый `inl` по умолчанию, на любой другой по желанию.

Команда выполняется по следующим правилам:

- команда относится к **непосредственно следующему за ней** SELECT-запросу. Если между **HEADER** и SELECT-запросом выполнялись другие SQL-операторы, то значение **HEADER** теряется (однако не SQL-команды `inl` значения команды **HEADER** не сбрасывают, за исключением команд `inl BROWSE`, `DISHEAD`, `EXIT`, что следует из их предназначения);
- если команда задана, то заголовок по умолчанию SELECT-запроса заменяется на заголовок из команды **HEADER**;
- команда **EXEC** не распространяется на команды **HEADER**, т.е. при повторении выполнения SELECT-запроса команду **HEADER** надо явно повторить;
- в табличном режиме отображения данных команда не действует;
- максимальная длина формируемого по команде заголовка 800 символов;
- не уместившийся в одной строке текст заголовка можно перенести на следующую строку. Признаком переноса строки заголовка является символ дефиса '-' в последней позиции строки.
- можно создать заголовок, состоящий из нескольких строк. Признаком конца очередной строки заголовка является символ дефиса «-» в последней позиции строки. Максимальное количество строк заголовка – 10.

Пример

```
SQL> header:           Утверждаю-
2>                   Директор предприятия-
3>                   "Виртуал" Прошан И.Я.-
4> -
5>                   Штатное расписание отд. N 5-
6> -----
7> |Таб.N |   ФИО   | Должность| Оклад  |-
8> -----
SQL>select "Таб_N", "ФИО", "Должность", "Оклад"
>1 from "Штаты" where "N_Отдел"=5;
```

DISHEAD

Формат

DISHEAD

Назначение

Запрещает/разрешает выводить заголовок столбца строки ответа.

Описание

Команда **DISHEAD** запрещает/разрешает выводить заголовок столбца строки ответа, установленный по умолчанию или с помощью команды **HEADER**.

Команда работает как переключатель – изменяет текущий режим вывода заголовка на противоположный.

По умолчанию при запуске `inl` установлен режим вывода заголовка.

Пример

```
select make,model from auto fetch first 2;
MAKE                MODEL
----              -
| FORD              | MERCURY COMET GT V8 |
| ALPINE            | A-310                |
INL : выдано строк : 2
```

```
header: Изготовитель      Модель
select make,model from auto fetch first 2;
Изготовитель      Модель
| FORD            | MERCURY COMET GT V8 |
| ALPINE          | A-310                |
INL : выдано строк : 2
```

```
dishead
select make,model from auto fetch first 2;
| FORD            | MERCURY COMET GT V8 |
| ALPINE          | A-310                |
INL : выдано строк : 2
```

HELP

Формат

HELP

Назначение

Получение справочной информации о командах программы `inl`.

Описание

Команда выдает на экран полный список обрабатываемых программой `inl` команд с краткими пояснениями (Таблица 3).

Таблица 3. Результат выполнения команды HELP

Имя команды	Назначение команды
HELP	Показать список команд
TIME	Включить/выключить выдачу времени
FORMAT	Переводить/не переводить ответ в символьный вид

Имя команды	Назначение команды
USERNAME	Подключиться с другим именем пользователя
PAGE	Включить/выключить выдачу ответа по страницам
COUNT	Выводить/не выводить количество кортежей
Distail	Разрешить/запретить дополнение VAR типов до максимальной ширины
Disfill	Разрешить/запретить вывод лидирующего и хвостового ограничителя
Dissep	Разрешить/запретить вывод сепаратора
Disrep	Разрешить/запретить вывод результатов выполнения процедур
Dishead	Разрешить/запретить вывод заголовка ответа
BROWSE	Включить/выключить выдачу результата запроса в табличном режиме
–	Выполнение запроса из файла
OUTFIL :	Определение файла вывода ответа
HEADER :	Определение заголовка ответа
UNLOAD :	Определение разделителя
CREATE	Создание таблицы или индекса
CREATE PROC	Создание хранимой процедуры
CREATE TRIG	Создание триггера
ALTER PROC	Модификация хранимой процедуры
GRANT	Предоставление привилегий пользователю
ALTER	Изменить описание таблицы
DROP	Удалить таблицу или индекс
REVOKE	Отменить привилегии пользователя
PRESS	Перестроить все индексы и сжать номера записей
REBUILD	Восстановить конвертор таблицы
SELECT	Выбор строк из таблицы
INSERT	Вставка строк в таблицу
UPDATE	Замена строк
DELETE	Удаление строк из таблицы
LIST	Вывод текущих установок программы inl
!	Начало комментария для вывода на экран
EXEC	Выполнить запрос
EDIT	Редактировать запрос
OPTIMISTIC	Включить/выключить режим обработки транзакций <code>optimistic concurrency control</code>
EXCLUSIVE	Установить режим обработки транзакций <code>exclusive</code>

Имя команды	Назначение команды
COMMIT	Завершить текущую транзакцию
ROLLBACK	Отменить текущую транзакцию
PRIORITY:	Изменить приоритет канала
SH	Выполнить команду ОС
SHOW	Показать описание таблицы
EXIT	Завершение работы
CODEPAGE	Установить кодовую таблицу (866, 1251, KOI8)
SLEEP	Приостановить выполнение
DBINFO	Получить информацию о базе
IGNORE	Запретить/разрешить выдачу диагностического сообщения

IGNORE

Формат

IGNORE <код завершения>

<Код завершения> – возможный код завершения, возвращаемый ядром СУБД ЛИНТЕР при обработке SQL-запроса.

Назначение

Игнорирование заданного кода завершения при пакетной обработке SQL-запросов.

Описание

При пакетной обработке SQL-скрипта `inl` выдает на консоль диагностические сообщения для всех ошибочных ситуаций, возникающих при обработке SQL-запросов, и формирует код возврата обработки SQL-скрипта, который может быть получен и проанализирован внешней программой (например, командным процессором), вызвавшей `inl` на выполнение (см. раздел «Коды завершения»).

Если в процессе обработки SQL-скрипта СУБД вернула хотя бы один ненулевой код завершения, то код возврата всего скрипта будет также ненулевым, что подразумевает наличие ошибки в SQL-скрипте.

Команда IGNORE заставляет игнорировать заданный код завершения, обеспечивая тем самым нулевой код возврата скрипта.

Диагностическое сообщение, соответствующее коду завершения, указанному в команде IGNORE, выдается с пометкой «игнорируется», например:

```
INL: состояние выполнения: 2202 (игнорируется)
```

С помощью **IGNORE** можно задать только один код завершения. Для нескольких кодов завершения надо использовать несколько команд **IGNORE**.

Установленные значения сохраняются только на время текущего сеанса работы `inl`.

Повторное выполнение команды **IGNORE** с указанием того же самого кода завершения отменяет его игнорирование.

Просмотр текущих установок команды **IGNORE** выполняется с помощью команды **LIST**.

Пример

```
SQL> !запрет выдачи диагностического сообщения для кода 501
SQL> ignore 501 игнорирование кода завершения 501
SQL> ignore 73 игнорирование кода завершения 2202
SQL> ...
SQL> ignore 501 отмена игнорирования кода завершения 501
SQL> ...
```

LIST

Формат

`LIST`

Назначение

Вывод текущих установок программы `inl`.

Описание

Текущие установки программы `inl` содержат следующую информацию:

- 1) состояние переключателей режимов работы `inl`, задаваемых командами:
 - `TIME`;
 - `PAGE`;
 - `FORMAT`;
 - `COUNT`;
 - `OPTIMISTIC`;
 - `PESSIMISTIC`;
 - `BROWSE`.
- 2) символ разделителя (заданный командой `UNLOAD`;) и выходной файл (заданный командой `OUTFIL`);
- 3) текст последнего обработанного SQL-запроса (эта информация полезна при выполнении команды `EXEC`);
- 4) кодовая страница, в которой работает `inl`;
- 5) приоритет, по которому СУБД ЛИНТЕР обрабатывает SQL-запросы, поступающие от `inl`;
- 6) список кодов завершения СУБД ЛИНТЕР, информация о которых не должна выводиться на экран или в выходной файл.

Команды

Пример

```
;Вывод текущих установок на экран
SQL>
-- Текущие установки INL --

time          : Вкл.
page          : Вкл.
format        : ВЫкл.
count         : Вкл.
optimistic    : ВЫкл.
pessimistic   : ВЫкл.
priority      : 0
browse        : ВЫкл.
outfil        :
unload        : |
Код. стр.     : MS-DOS 866
Запрос: SELECT Tn ,Fio ,Dolgn ,Ocl FROM Kadry WHERE
Notd=111;
Игнорировать коды завершения :2202
SQL>

;Вывод текущих установок в файл
INL -u SYSTEM/MANAGER > output.txt
...
list
...
exit
```

OPTIMISTIC

Формат

OPTIMISTIC

Назначение

Переключатель режима обработки транзакций.

Описание

Эта команда переключает текущий режим работы СУБД на режим `optimistic Concurrency Control` и обратно.

Выключенный (положение **ВЫКЛ.**) переключатель `OPTIMISTIC` устанавливает режим работы СУБД ЛИНТЕР с возможностью «теплого» рестарта.

Сразу после запуска `inl` переключатель режима обработки транзакций установлен в положение **ВЫКЛ.**

OUTFIL

Формат

OUTFIL: [<спецификация файла>] | [CON]

Назначение

Спецификация выходного файла.

Описание

По умолчанию результаты выполнения команд и/или SQL-операторов `inl` выводит на экран видеотерминала пользователя (выходной файл по умолчанию). Для переназначения вывода используется команда `OUTFIL`. Параметр <спецификация файла> задает местоположение и имя выходного *текстового* файла в формате соответствующей операционной системы, например, `C:\linter\test\crt_table.tst`.

Если в качестве выходного файла указано `CON`, то текущий выходной файл закрывается и вывод осуществляется на видеотерминал.


В выходной файл записываются:

- 1) ответы на SQL-запрос (в форматированном или неформатированном виде);
- 2) заголовок для ответов (определенный командой `HEADER`);
- 3) итоговая статистика.

Команда выполняется по следующим правилам:

- устройство и каталог (каталоги), указанные в <спецификации файла> должны существовать на момент выполнения команды (автоматически не создаются);
- если не указано местоположение выходного файла, он создается в текущем каталоге;
- данные в выходной файл записываются в коде ASCII;
- результаты выполнения SQL-оператора будут направляться в файл, определенный командой `OUTFIL`, до тех пор, пока не будет произведено новое назначение выходного файла. При этом выходной файл, открытый предшествующей командой `OUTFIL`, закроется и откроется новый специфицированный файл;
- если в `OUTFIL` указан файл, который уже существует, то выходные данные *добавляются* в это файл;
- комментарии всегда выдаются на экран видеотерминала;
- команда `OUTFIL` может быть введена как с терминала, так и из SQL-скрипта;
- если параметр <спецификация файла> не указан, то в качестве выходного файла используется экран видеотерминала;
- максимальная длина параметра <спецификации файла> – 30 знаков.

 При запуске `inl` выходным файлом по умолчанию установлен экран видеотерминала.

 Узнать текущее имя выходного файла можно с помощью команды `SHOW`.

Примеры

```
SQL>outfil:/usr/home/Fill.ans      (в среде ОС Unix)

;создание выходного файла в текущем каталоге
SQL>outfil:tst
...

;создание другого выходного файла в заданном каталоге
SQL>outfil:d:/linter/script/crt_db.sql
...

;перенаправление вывода на экран видеотерминала
SQL>outfil:con
...

;продолжение вывода в файл tst
SQL>outfil:tst
...
```

RESULT

Формат

RESULT: [<спецификация файла>]

Назначение

Спецификация выходного файла для следующего SQL-запроса.

Описание

Команда RESULT позволяет записывать результаты выполнения SQL-запросов в индивидуальные файлы.

Она выполняется аналогично команде OUTFIL, но распространяется только на один SQL-запрос, выполняемый непосредственно за этой командой. После обработки такого запроса выдача результатов последующих SQL-запросов будет продолжена на экран видеотерминала или в файл, указанный в команде OUTFIL (если команда OUTFIL была ранее подана).

Пример

```
outfil:auto.tst
select count(*) from auto;
-- сформировать список моделей в файле model.lst
result:model.lst
select distinct model from auto;
-- сформировать список изготовителей в файле make.lst
result:make.lst
select distinct make from auto;
-- продолжить выдачу результатов в файл auto.tst
...
```

PAGE

Формат

PAGE

Назначение

Разрешить/запретить разбивку выводимой информации по страницам.

Описание

Команда PAGE выполняется только в строчном режиме функционирования `inl` и относится к SQL-запросам, потенциально возвращающим множественный ответ (несколько экранов видеотерминала):

- SELECT;
- EXECUTE PROCEDURE;
- TEST TABLE.

Установленный в положение **Вкл.** (включен) переключатель разбивки по страницам заставляет `inl` выдавать ответы SQL-запроса порциями по 20 записей ответа.



Если запись ответа занимает несколько строк экрана, то 20 записей ответа могут потребовать несколько экранов и в этом случае первые экраны ответа будут потеряны. В данной ситуации необходимо использовать команду **OUTFIL**: для вывода ответов в файл с последующим просмотром их системными средствами или перейти в режим табличный режим функционирования `inl`.

После выдачи очередной порции ответов `inl` вступает в диалог с пользователем и ждет указаний о дальнейшей работе:

`INL` : нажмите любую клавишу (q для выхода) :

Ввод символа 'q' эквивалентен отказу от выдачи следующей порции ответа и переходу к вводу новых команд для формирования другого SQL-запроса, нажатие любой другой клавиши продолжит выдачу ответов.

Выключенный (положение **Выкл.**) переключатель разбивки по страницам устанавливает режим выдачи всех ответов на запрос без разбивки по страницам.

Если ранее была выдана команда **OUTFIL**, то разбиение по страницам не выполняется.

Команда PAGE работает как циклический двоичный переключатель, т.е. каждое выполнение PAGE отменяет текущий режим и устанавливает противоположный. Установленный режим сохраняется до изменения его новой командой PAGE или конца работы `inl`.

Сразу после запуска `inl` переключатель разбивки по страницам по умолчанию установлен в положение **Вкл.**

Для просмотра текущего состояния переключателя режима разбивки страниц используется команда **LIST**.

Смена значения переключателя производится по команде PAGE.

Команды



В режиме приема команд из файла (а не с видеотерминала) команда PAGE игнорируется.

Пример

Выдать список служащих, получающих минимальную зарплату.

```
SQL>Select cast ' Минимальная зарплата: ' as char(20),
1>to_char(MIN(Salary)) from person
2>union
3>SELECT DISTINCT FirstNam,Name FROM Person
4>WHERE Salary =(SELECT MIN(Salary) FROM Person) order 5>by 1
asc;
```

Минимальная зарплата	
ANNETTE	PERREAULT
ART	SPIEGEL
BILL	MOUREAU
BRENDA	MOUREAU
CHARLES	WAGNER
CHARLY	FERRARI
CLARA	WAGNER
DALIAH	COLVILLE
EDDY	ALEXANDER
FORTUNA	RAEBIGER
FRANCOISE	QUIHLLAULT
GERARDIII	TERZI
JACK	LAWLER
JEFFERSON	LAWLER
JO	RAY
JOHN	QUILLION
LILIAN	KOLENCE
MARTHA	DAVENPORT
PUALA	HOROWITZ

INL : нажмите любую клавишу (q для выхода):

PRIORITY

Формат

PRIORITY: [<значение>]

Назначение

Изменение приоритета канала.

Описание

Команда устанавливает новый приоритет каналу, выделенному СУБД ЛИНТЕР для работы с inl.

Параметр <значение> – целочисленное положительное число в диапазоне от 0 до 240 (0 – минимальный приоритет, 249 – максимальный). Назначенный приоритет присваивается всем выполняемым SQL-запросам, подаваемым программой inl.

Установленный приоритет действует до назначения другого приоритета или до конца текущего сеанса inl.

Сразу после запуска inl приоритет канала по умолчанию установлен в 0.

Для просмотра текущего значения приоритета канала используется команда LIST.

Примеры

```
SQL>list
```

```
...
```

```
priority :0
```

```
...
```

Получить список владельцев автомобилей марки «FORD», чьи имена начинаются на D, и моделей их машин.

```
SQL>SELECT FirstNam, Name, Model FROM Person, Auto
```

```
1>>WHERE FirstNam LIKE 'D%' AND Make ='FORD' AND
```

```
2>Person.PersonId =Auto.PersonId;
```

```
INL : start time : 14:32:20 end time : 14:32:22
```

FIRSTNAM	NAME	MODEL
DIANA	HEAFNER	MUSTANG BOSS 351
DEBORAH	SPIEGEL	GRAN TORINO SPORT
		V8
DIANA	WATSON	PINTO RUNABOUT
DAISY	WOOLSEY	LTD COUNTRY SQUIRE
DIANA	COLVILLE	GRAN TORINO SPORT
		V8
DANIEL	SRC	MERCURY MONTEREY
DAN	TANIMOTO	CAPRI RS 2600
DANIEL	ALDEN	MERCURY MONTEREY

```
INL : number of rows shown: 8
```

```
SQL>
```

```
SQL>priority:240
```

```
SQL>list
```

```
...
```

```
priority :240
```

```
...
; этот же запрос будет обработан ядром с более высоким
; приоритетом
SQL>exec
...
; установка приоритета по умолчанию
SQL>priority:
SQL>list
...
priority :0
...
```

SH

Формат

SH <команда операционной системы>

Назначение

Выполнение команды операционной системы.

Описание

Команда позволяет, не выходя из программы `inl`, выполнить любую допустимую команду операционной системы.

Пример

Использование обращения к командному интерфейсу операционной системы.

```
SQL>SH ls -l
total 8
-rw-r--r--  1 root  sys  978   May  19 20:22  c.dir
-rw-rw-rw-  1 root  sys 2035   May  12 10:43  com_proc.c
-rw-rw-rw-  1 root  sys  992   May  15 12:55  com_proc.o
-rw-rw-rw-  1 root  sys 3128   May  12 10:43  dbc.h
-rw-rw-rw-  1 root  sys 23203  May  12 10:43  dbc_tcp.c
-rw-rw-rw-  1 root  sys 14400  May  15 12:55  dbc_tcp.o
-rw-rw-rw-  1 root  sys  2774  May  12 10:43  dbcs_err.h
-rw-rw-rw-  1 root  sys  2797  May  12 10:43  dbcscomm.h
SQL>
```

SHOW

Формат

SHOW <имя таблицы>

Назначение

Получение справочной информации о таблице.

Описание

<Имя таблицы> допускает символ обобщения – %. Для получения информации о всех таблицах необходимо подать запрос SHOW %.

В разделе «Характеристики столбца» (см. пример ниже) для каждого столбца таблицы выдается следующая информация:

- тип данных столбца и, в зависимости от типа данных, длина столбца и точность представления данных;
- атрибуты столбца (первичный ключ, ссылочный ключ, автоиндексация и т.п.);
- признак индексируемости столбца и параметры индекса: номер уровня характеризует объем индексного файла и, в общем случае, информирует о том, сколько страниц индексного файла будет предварительно считано при поиске значения данного столбца; номер вершины показывает относительный номер страницы индексного файла, где находится вершина данного столбца;
- о кодировках для версий, начиная с 6.0.

Если в таблице определены CHECK-ограничения, опции GENERATED-столбцов, атрибуты столбцов AUTOINC RANGE, AUTOINC INITIAL, то выводится информация об этих параметрах.



Для распределенной таблицы выдается дополнительная информация в виде:

* Узел : 'Имя_узла' (ID=Идентификатор_узла)

```

I
INTEGER CHECK(("I" > 0) AND ("I" < 10))
                DEFAULT 1

L
INTEGER AUTOINC INITIAL(1)
CH
CHAR(99)
NCH
NCHAR(21)
                DEFAULT 'abcd'

B
BIGINT CHECK("B" < 100)
                DEFAULT 100

CHECK(("I"+1) > 0)

```

Примеры

1 Справочная информация о таблице auto:

Описание таблицы «AUTO»

```
* Номер таблицы           : 245
* Предельный ROWID        : 1024
* Последний занятый ROWID : 1000
* Номер текущей строки     : 1000
* Процент заполнения страницы : 100
* Порог освобождения страницы : 0 (не установлен)
* Длина строки            : 113
* Количество столбцов     : 13
* Количество индексов     : 1
* Файлов индексов         : 1 ("SY00" 2)
* Файлов данных           : 1 ("SY00" 13)
```

INL : нажмите любую клавишу (q для выхода) :

Характеристики столбца

```
MAKE          CHAR(20)
MODEL         CHAR(20)
...          И Т.Д.
SQL>
```

2 Справочная информация о таблице с ограничениями целостности:

```
create or replace table "TA1" ("I" int default 1 check ( ("I" >
0) AND ("I" < 10) ),
"L" int autoinc default 1,
"CH" char(99),
"NCH" nchar(21) default 'abcd',
"B" bigint default 100 check ( "B" < 100 ), check (I+1 > 0));
select cast LINTER_DICT_INFO(1,$$$S11,5) as char(80) from
$$$$SYSRL where $$$S13='TA1';
```

Описание таблицы «TA1»

...

Характеристики столбца

```
I      INTEGER CHECK(("I" > 0) AND ("I" < 10)) DEFAULT 1
L      INTEGER AUTOINC INITIAL(1)
CH     CHAR(99)
NCH    NCHAR(21) DEFAULT 'abcd'
B      BIGINT CHECK("B" < 100) DEFAULT 100
CHECK(("I"+1) > 0)
```

SLEEP

Формат

SLEEP [<время>]

<время> – целочисленное неотрицательное значение.

Назначение

Приостановление выполнения программы `inl` на указанное в параметре <время> количество секунд. Если параметр <время> не задан, то по умолчанию пауза равна нулю.

Описание

В результате выполнения команды `SLEEP` на экране появляется сообщение:

INL: Пауза на <время> сек.

и функционирование программы приостанавливается на <время> секунд.

TIME

Формат

TIME

Назначение

Разрешить/запретить выдачу информации о длительности выполнения SQL-запроса.

Описание

Команда `TIME` управляет режимом отображения информации о длительности выполнения SQL-запроса. Она работает только в строчном режиме отображения информации. Если переключатель режима отображения времени включен (положение **вкл.**), то `inl` проводит замер и выдачу информации о времени начала и конца выполнения запроса (с точностью до секунды), например,

INL: начальное время : 16.08.42 конечное время :16.08.43

Для подсчета времени используется локальное время, установленное в компьютере (в операционной системе), на котором выполняется `inl`. Под замером времени следует понимать, что `inl` фиксирует время передачи запроса на выполнение и время получения *первого* ответа на этот запрос.


Если переключатель времени выключен (положение **выкл.**), то замер и, соответственно, выдача показаний времени обработки запроса не происходит.

Команда `TIME` работает как циклический двоичный переключатель, т.е. каждое выполнение `TIME` отменяет текущий режим отображения и устанавливает противоположный. Установленный режим сохраняется до конца работы `inl`.

Команды

Сразу после запуска inl переключатель режима отображения времени по умолчанию установлен в положение **ВКЛ.**

Для просмотра текущего состояния переключателя режима отображения времени используется команда **LIST**.

 Результаты выполнения команды **TIME** всегда выводятся на экран видеотерминала, т.е. в выходной файл, создаваемой по команде **OUTFIL**, они не попадают. Однако, если inl запущена с командой перенаправления вывода (например, `inl>out.txt`), то в этом случае результаты выполнения **TIME** будут помещены в указанный файл.

Примеры

```
...
SQL>list
           --Текущие установки INL--
time      :вкл.
...
SQL>select count (*) from auto
1>where color in ('YELLOW','GREEN','BLACK');
INL: начальное время : 10.55.58 конечное время :10.55.58
|          367|
INL : выдано строк  :1
SQL>time
SQL>list
           --Текущие установки INL--
time      :выкл.
...
SQL>select count (*) from auto
1>where color in ('YELLOW','GREEN','BLACK');
|          367|
INL : выдано строк  :1
SQL>
```

UNLOAD

Формат

UNLOAD: [<символ>]

Назначение

Определение разделителя.

Описание

При выдаче результатов выполнения SELECT-операторов в строчном режиме `inl` разделяет столбы выдаваемой информации с помощью специального символа – разделителя столбцов. По умолчанию разделителем является символ «|». С помощью команды `UNLOAD` пользователь может установить любой другой разделитель, например, `' : '`. Новый разделитель действует до его изменения командой `UNLOAD` либо до конца текущего сеанса `inl`.

Если параметр <символ> не задан, используется разделитель по умолчанию.

Пример

`;действие разделителя по умолчанию`

```
SQL>select make, year from auto;
```

```
|FORD          |      71|
```

```
|ALPINE        |      70|
```

...

`;разделитель – символ «:»`

```
SQL>unload::
```

```
SQL>select make, year from auto;
```

```
:FORD          :      71:
```

```
:ALPINE        :      70:
```

...

`; отказ от разделителя`

```
SQL>unload:<пробел>
```

```
SQL>select make, year from auto;
```

```
FORD          71
```

```
ALPINE        70
```

...

`; разделитель – символ «*»`

```
SQL>unload:*!%
```

```
SQL>select make, year from auto;
```

```
*FORD          *      71*
```

```
*ALPINE        *      70*
```

...

`; вернуться к разделителю по умолчанию`

```
SQL>unload:
```

```
SQL>
```

DISSEP

Формат

DISSEP

Назначение


Запрещает/разрешает выводить разделители в строке ответа.

Описание

Команда `DISSEP` запрещает/разрешает выводить разделители, установленные по умолчанию или с помощью команды `UNLOAD`.

Команда работает как переключатель – изменяет текущий режим вывода межстолбцовых разделителей на противоположный.

По умолчанию при запуске `inl` установлен режим вывода всех разделителей

 Проверить текущий режим вывода разделителей можно с помощью команды `LIST`. Если установлен запрет на вывод разделителей, то выдается сообщение `separators disabled`

Если вывод разделителей разрешен, сообщение об этом не выдается.

Пример

```
unload:!  
list  
...  
(по умолчанию вывод всех разделителей разрешен, поэтому сообщение об этом не выдается)  
...  
select personid, make, model from auto fetch first 2;  
  
PERSONID      MAKE                MODEL  
-----      -  
!              1!FORD                !MERCURY COMET GT V8 !  
!              2!ALPINE            !A-310                !  
INL : выдано строк      : 2  
  
-- отмена выдачи всех разделителей  
dissep  
list  
...  
separators disabled  
...  
select personid, make, model from auto fetch first 2;  
  
PERSONID      MAKE                MODEL  
-----      -  
              1FORD                MERCURY COMET GT V8  
              2ALPINE            A-310  
INL : выдано строк      : 2  
  
-- возобновление выдачи всех разделителей  
dissep
```

```
select personid, make, model from auto fetch first 2;
```

```
PERSONID      MAKE                MODEL
-----      -
!              1!FORD              !MERCURY COMET GT V8 !
!              2!ALPINE         !A-310                !
INL : выдано строк      : 2
```

DISTAIL

Формат

```
DISTAIL
```

Назначение


Запрещает/разрешает выводить в строке ответа концевые разделители.


Описание

Команда **DISTAIL** запрещает/разрешает выводить концевые разделители, установленные по умолчанию или с помощью команды **UNLOAD**.

Команда работает как переключатель – изменяет текущий режим вывода разделителей на противоположный.

По умолчанию при запуске **inl** установлен режим вывода концевых разделителей

 Проверить текущий режим вывода разделителей можно с помощью команды **LIST**. Если установлен запрет на вывод разделителей, то выдается сообщение **Limiters disabled**. Если вывод концевых разделителей разрешен, сообщение об этом не выдается.

 Команда **distail** устанавливает концевые разделители только в том случае, если они не были установлены предыдущей командой **dissep**.

Пример

```
unload:!!
list
select personid, make, model from auto fetch first 2;

PERSONID      MAKE                MODEL
-----      -
!              1!FORD              !MERCURY COMET GT V8 !
!              2!ALPINE         !A-310                !
INL : выдано строк      : 2

-- отмена выдачи граничных разделителей
distail
list
select personid, make, model from auto fetch first 2;
PERSONID      MAKE                MODEL
-----      -
!              1!FORD              !MERCURY COMET GT V8 !
```

```
                2!ALPINE                !A-310
INL  : выдано строк      : 2
```

USERNAME

Формат

```
USERNAME <имя пользователя>[/<пароль>]
```

Назначение

Инициирование работы под другим именем пользователя в текущей БД без выхода из inl.

Описание

Команда выполняется следующим образом:

- 1) если параметр <имя пользователя>[/<пароль>] задан полностью (т.е. одновременно имя и пароль), то это значение передается ядру СУБД, с которой в данный момент работает inl для проверки в БД пользователя с указанными регистрационными данными;
- 2) если в команде задано только <имя пользователя>, то после нажатия клавиши <Enter> в ответ на приглашение:

```
SQL>
```

необходимо ввести пароль пользователя (не более 18 символов) без двойных кавычек;
- 3) если имя пользователя и пароль введены правильно, inl продолжает работу с БД от имени нового пользователя (при этом текущий приоритет канала сбрасывается), иначе будет выдано сообщение об ошибке: «Неверное имя пользователя» или «Неверный пароль»;
- 4) если команда USERNAME завершилась неудачно (имя или/и пароль не совпали), текущий канал не закрывается и можно продолжать работать под текущими регистрационными данными.

Примеры

```
INL -u SYSTEM/MANAGER
SQL>time
SQL> select count(*) from AUTO;
|          1000|
INL  : выдано строк      :1
SQL>create user "Склад" identified by 'чы34эъ';
SQL>username "Склад"/"чы34эъ"
SQL> select count(*) from AUTO;
INL  : состояние выполнения :2202
неизвестная таблица
```

```

SQL> select count(*) from SYSTEM.AUTO;
|          1000|
INL  : выдано строк   :1
SQL>username SYSTEM
Пароль пользователя: ***** (введено MANAGER)
SQL> select count(*) from AUTO;
|          1000|
INL  : выдано строк   :1
SQL>

```

Работа с BLOB-данными

Для работы с BLOB-данными (просмотр, добавление, удаление, выгрузка) необходимо установить табличный режим просмотра (**Browse** вкл.), для чего воспользоваться командой **BROWSE** для перехода в этот режим (если текущее состояние **Browse** выкл.), и установить текущей ту строку таблицы, для которой будет выполняться операция с BLOB-данными. Для существующих в таблице строк текущая строка устанавливается с помощью команды **SELECT**; если предполагается добавлять BLOB-данные для новой строки таблицы, то предварительно должна быть выполнена команда **INSERT**, в которой значение BLOB-столбца пропущено или задано как NULL-значение. Вместе с BLOB-столбцом можно выбирать столбцы других типов данных. В появившемся окне перейти в BLOB-столбец (используя клавиши <стрелка влево/вправо> или <Ctrl>+<стрелка влево/вправо>), в результате высветится окно (см. рис. 5) для работы с BLOB-данными.



Рис. 5 – Окно для работы с BLOB-данными

В окне (рис. 5) в столбце BLOB-данных отображаются атрибуты загруженных BLOB-данных (в частности, первое число – это длина BLOB-данных в байтах). Панель внизу экрана содержит справочную информацию.

Клавишные команды работы с BLOB-данными (все они относятся только к выделенной строке загруженной таблицы):

- <Alt>+<S> - просмотр BLOB-данных. Независимо от характера данных, они высвечиваются в текстовом виде в стандартном окне просмотра файлов встроенного редактора операционной системы;
- <Alt>+<C> - удаление BLOB-данных. После удаления BLOB-столбец получает NULL-значение;
- <Alt>+<L> - загрузка BLOB-данных. При выполнении этой команды высвечивается стандартное окно для поиска и выбора файла, содержащего загружаемые данные;
- <Alt>+<U> - выгрузка BLOB-данных. При выполнении этой команды высвечивается стандартное окно для спецификации файла, в который должны быть выгружены данные.
- <Alt>+<I> - добавление BLOB-данных. При выполнении этой команды высвечивается стандартное окно для спецификации файла, который должен быть добавлен в конец BLOB-столбца.

Сообщения программы

Информационные сообщения

Информационные сообщения содержат справочную информацию и сведения о текущей работе программы inl. Смысл этих сообщений понятен из контекста интерактивного сеанса.

Диагностические сообщения

Диагностические сообщения подразделяются на две группы: сообщения, порожденные неправильными командами пользователя и сообщения, связанные с проблемами функционирования собственно программы inl в операционной системе (системные ошибки). В первом случае пользователь должен проанализировать сообщение об ошибке, исправить ее и повторить выполнение команды, во втором случае следует обратиться к администратору СУБД ЛИНТЕР.

Пользовательские ошибки

- INL : слишком длинная строка ответа
- INL : не открыт выводной файл. Код ошибки nppp
- INL : имя пользователя отсутствует в командной строке
- INL : командный файл отсутствует в командной строке
- INL : имя сервера отсутствует в командной строке
- INL : длинное имя сервера
- INL : неверное имя пользователя.
- INL : неверный пароль
- INL : канал не открыт. ЛИНТЕР не загружен
- INL : командный файл уже активен
- INL : неверная команда
- INL : длинное имя файла
- INL : неверный синтаксис
- INL : имя отношения не определено
- INL : неверное имя синонима
- INL : неверный синтаксис

Системные ошибки

Ниже под значением nppp понимается числовой код завершения, выдаваемый операционной системой для СУБД ЛИНТЕР.

- INL : канал не открыт. Код ошибки nppp
- INL : командный файл не открыт. Код ошибки nppp

Сообщения программы

INL : ошибка закрытия канала. Код ошибки nppn

INL : ошибка операционной системы. Код ошибки nppn

INL : ошибка открытия канала. Код ошибки nppn

INL : ошибка завершения транзакции. Код ошибки nppn

INL : ошибка отката транзакции. Код ошибки nppn

Коды завершения

Помимо диагностических сообщений, выдаваемых на видеотерминал или в выходной файл и предназначенных для визуального контроля выполнения программы, `inl` формирует код завершения (код возврата) для проверки результата своей работы другими программными средствами. Имя переменной, в которую помещается код возврата, зависит от операционной системы (см. соответствующую документацию).

Таблица 4. Дополнительная информация о SQL-запросах

Код завершения	Символьное обозначение кода завершения	Значение
0	<code>exit_success</code>	Нормальное завершение
1	<code>exit_help</code>	Выдача справочной информации
2	<code>exit_wrong_parameter</code>	Ошибка в параметрах командной строки
3	<code>exit_not_connected</code>	Нет соединения с БД
4	<code>exit_some_query_has_2202</code>	В БД нет заданного объекта (соответствует коду завершения 2202 СУБД)
5	<code>exit_some_query_has_error</code>	Ошибка в SQL-запросе
6	<code>exit_cannot_open_the_file</code>	Ошибка открытия файла
7	<code>exit_some_resource_locked</code>	Требуемая запись или таблица заблокированы

