ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ИНФОРМАЦИОННАЯ ТЕХНОЛОГИЯ

ЯЗЫК БАЗ ДАННЫХ SQL С РАСШИРЕНИЕМ ЦЕЛОСТНОСТИ

Издание официальное

БЗ 2—93/146

gost.e-ecolog.ru

ГОССТАНДАРТ РОССИИ
Москва

ГОСТ Р ИСО/МЭК 9075—93

ИНФОРМАЦИОННАЯ ТЕХНОЛОГИЯ

ЯЗЫК БАЗ ДАННЫХ SQL С РАСШИРЕНИЕМ ЦЕЛОСТНОСТИ

Издание официальное

МОСКВА — 1993

Предисловие

1 РАЗРАБОТАН И ВНЕСЕН Техническим комитетом по стандартизации ТК22 «Информационная технология»

2 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Постановлением Госстандарта России от 08.07.93 № 169

Настоящий стандарт подготовлен на основе аутентичного текста международного стандарта ИСО/МЭК 9075—89 «Системы обработки информации. Язык баз данных SQL с расширением целостности»

3 ВВЕДЕН ВПЕРВЫЕ

© Издательство стандартов, 1993

Настоящий стандарт не может быть полностью или частично воспроизведен, тиражирован и распространен без разрешения Госстандарта России

СОДЕРЖАНИЕ

1 Область применения . . . 1
2 Ссылки . . . 2
3 Определения . . . 2
3.1 Структура . . . 2
3.2 Нотация . . . 3
3.3 Условные обозначения . . . 3
3.4 Согласование . . . 4
4 Понятия . . . 5
4.1 Множества . . . 5
4.2 Типы данных . . . 6
4.3 Столбцы . . . 7
4.4 Таблицы . . . 7
4.5 Ограничения целостности . . . 8
4.6 Схемы . . . 9
4.7 База данных . . . 9
4.8 Модули . . . 9
4.9 Процедуры . . . 10
4.10 Параметры . . . 10
4.11 Стандартные языки программирования . . . 10
4.12 Курсоры . . . 10
4.13 Операторы . . . 12
4.14 Встроенный синтаксис . . . 12
4.15 Привилегии . . . 13
4.16 Транзакции . . . 13
5 Общая лексика . . . 14
5.1 <символ> . . . 14
5.2 <литерал> . . . 15
5.3 <лексема> . . . 17
5.4 <имена> . . . 18
5.5 <тип данных> . . . 20
5.6 <спецификация значения> и <спецификация цели> . . . 21
5.7 <спецификация столбца> . . . 23
5.8 <спецификация функции набора> . . . 24
5.9 <выражение значения> . . . 26
5.10 <предикат> . . . 28
5.11 <предикат сравнения> . . . 29
5.12 <предикат интервала> . . . 30
5.13 <предикат принадлежности> . . . 30
5.14 <предикат подобия> . . . 31
5.15 <предикат нуля> . . . 33
5.16 <квантифицированный предикат> . . . 33
5.17 <предикат существования> . . . 34
5.18 <условие поиска> . . . 35
5.19 <табличное выражение> . . . 36
5.20 <спецификатор отображения> . . . 37
5.21 <спецификатор выборки> . . . 38
5.22 <спецификатор группировки> . . . 39
5.23 <спецификатор выборки групп> . . . 40
5.24 <подзапрос> . . . 41
5.25 <спецификация запроса> . . . 43

6 Язык определения схемы 46
6.1 <схема> 46
6.2 <определение таблицы> 47
6.3 <определение столбца> 49
6.4 <спецификатор умолчания> 49
6.5 <определение ограничений для таблиц> 50
6.6 <определение ограничения уникальности> 51
6.7 <определение ограничения на ссылки> 52
6.8 <определение ограничения проверки> 54
6.9 <определение представления> 54
6.10 <определение привилегий> 56
7 Модульный язык 58
7.1 <модуль> 58
7.2 <спецификатор имени модуля> 59
7.3 <процедура> 59
8 Язык манипулирования данными 63
8.1 <оператор закрытия> 63
8.2 <оператор блокировки> 64
8.3 <объявление курсора> 64
8.4 <оператор удаления: по положению> 68
8.5 <оператор удаления: поиск> 69
8.6 <оператор выборки> 70
8.7 <оператор вставки> 71
8.8 <оператор открытия> 74
8.9 <оператор отката> 75
8.10 <оператор выбора> 75
8.11 <оператор корректировки: по положению> 77
8.12 <оператор корректировки: поиск> 80
9 Уровни 82
Приложение A. <Встроенная общая программа SQL> 86
Приложение B. <объявление встроенной особой ситуации> . . . 92
Приложение C. <программа встроенного SQL КОБОЛ> . . . 94
Приложение D. <программа встроенного SQL ФОРТРАН> . . . 96
Приложение E. <программа встроенного SQL ПАСКАЛЬ> . . . 98
Приложение F. <программа встроенного SQL ПЛ/1> . . . 100
Указатель 102

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информационная технология

ЯЗЫК БАЗ ДАННЫХ SQL С РАСШИРЕНИЕМ ЦЕЛОСТНОСТИ

Information processing systems — Database
Language SQL with Integrity Enhancement

Дата введения 1994—07—01

1 ОБЛАСТЬ ПРИМЕНЕНИЯ

Настоящий стандарт определяет синтаксис и семантику двух языков, ориентированных на работу с базами данных (языков баз данных):

1) Язык описания схемы (SQL — DDL) для объявления структур и ограничений целостности базы данных SQL.

2) Модульный язык и язык манипулирования данными (SQL — DML) для объявления процедур работы с базами данных и выполняемых операторов специальной программы применения базы данных.

Настоящий стандарт определяет логические структуры данных и базовые операции для базы данных SQL. Он обеспечивает функциональные возможности для разработки, доступа, сохранения, управления и защиты базы данных.

Настоящий стандарт представляет средство обеспечения мобильности определения баз данных и прикладных программ между соответствующими реализациями.

Стандарт определяет два уровня и отдельно свойство расширения целостности. Уровень 2 — это полный язык базы данных за исключением свойства расширения целостности. Уровень 1 — это подмножество уровня 2, определенное в пункте 9 раздела «Уровни».

Примечание — Дополнительный язык SQL планируется в последующих дополнениях к этому стандарту. Основные темы, рассматриваемые в таких дополнениях, — это расширенная обработка транзакций, задание определенных правил, определяемых разработчиком, расширенные средства обработки символов и средства обеспечения множеств национальных символов.

Издание официальное

Свойство расширения целостности представляет собой средство для задания:

1) ограничения на обращения между таблицами, которые должны соблюдаться;

2) проверочные ограничения, которые следует применять для строк таблицы;

3) величина по умолчанию для столбца, когда строка вставляется в таблицу.

Приложения к этому стандарту определяют встроенный синтаксис для включения оператора языка манипулирования данными SQL в стандартную в других отношениях прикладную программу. Такой встроенный синтаксис по определению является сокращенной записью для стандартной прикладной программы, в которой встроенные операторы SQL заменены в явной форме «вызовами» процедур базы данных, которые содержат оператор SQL.

Настоящий стандарт применяется для реализаций, существующих в среде, которая может включать в себя языки прикладных программ, языки запросов конечного пользователя, системы генерирования сообщений, системы словарей данных, системы библиотеки программ и системы распределенной связи, а также различные средства для создания баз данных, организации прохождения данных и оптимизации характеристик.

2 ССЫЛКИ

ГОСТ 28141—89 Язык программирования ФОРТРАН (ISO 1539 Programming Languages — FORTRAN)

ГОСТ 22558—89 Язык программирования КОБОЛ (ISO 1989 Programming Languages — COBOL)

ISO 6160—79, Язык программирования ПЛ/1 * (Programming Languages — PL/1)

ГОСТ 28140—89 Язык программирования ПАСКАЛЬ (ISO 7185 Programming Languages — PASCAL)

3 ОПРЕДЕЛЕНИЯ

3.1 **Структура**

Структура настоящего стандарта следующая:

1) 3.2 «Нотация» и 3.3 «Условные обозначения» определяют обозначения и условные обозначения в данном стандарте.

---

* Данный документ может быть получен по запросу из ВНИИКИ Госстандарта России

2) 3.4 «Согласование» определяет критерии согласования.

3) Раздел 4 «Понятия» определяют термины и представляет понятия, использованные в определении SQL.

4) Раздел 5 «Общие элементы» определяет элементы языка, появляющиеся в нескольких частях языка SQL.

5) Раздел 6 «Язык определения схемы» определяет способности SQL для задания базы данных.

6) Раздел 7 «Модульный язык» определяет модули и процедуры.

7) Раздел 8 «Язык манипулирования данными» определяет операторы по управлению данными в SQL.

8) Раздел 9 «Уровни» определяет оба уровня SQL и средства расширения целостности.

3.2 Нотация

В настоящем стандарте используется синтаксическая запись БНФ («нормальная форма Бэкуса», «форма Бэкуса — Наура») со следующими расширениями:

1) прямоугольные скобки ([ ]) показывают необязательные элементы;

2) многоточие (...) показывают элементы, которые могут повторяться один или более раз;

3) фигурные скобки ({}) охватывают последовательности элементов.

В синтаксисе БНФ символ продукции <A> определен как «содержащий» символ продукции <B>, если <B> занимает некое место в расширении <A>. Если <A> содержит <B>, то <B> «содержится» в <A>. Если <A> содержит <B>, то <A> является «содержащим» <A> символом продукции для <B>

3.3 Условные обозначения

Синтаксические элементы настоящего стандарта определяются с помощью понятия:

1) Функции: краткое указание цели элемента.

2) Формата: определение БНФ синтаксиса данного элемента.

3) Синтаксических правил: дополнительные синтаксические ограничения, не выраженные в БНФ, которым должен удовлетворять данный элемент.

4) Общим правилам: последовательное определение эффекта в ходе вычисления данного элемента.

В синтаксических правилах термин «должен» определяет условия, которые должны быть правильными для синтаксически под-

чиненного SQL языка. Обработка языка SQL, который не подчиняется форматам или синтаксическим правилам, определяется разработчиком.

В общих правилах термин «должен» определяет условия, которые проверяются во время прохода при выполнении оператора SQL. Если все такие условия истинны, то оператор успешно выполняется и устанавливается параметр SQLCODE (код SQL) в определенном неотрицательном числе. Если любое такое условие неправильно, то указание не выполняется успешным образом, выполнение указания не оказывает влияния на базу данных, и параметр SQLCODE приобретает заданное разработчиком отрицательное число.

Согласующая реализация не требуется для выполнения точной последовательности действий, определенных в Общих правилах. а будет оказывать такое же влияние на базу данных, что и эта последовательность. Термин «эффективно» используется в Общих правилах для подчеркивания действий, эффект от которых мог бы быть достигнут другими способами путем реализации.

Термин «устойчивый объект» используется для охарактеризования таких объектов, как <модуль>-ли и <схема>-мы, которые создаются и разрушаются с помощью определенных разработчиком механизмов.

3.4 Согласование

Настоящий стандарт определяет согласующий язык SQL и согласующие реализации SQL. Согласующий язык SQL будет твердо придерживаться формата БНФ и соответствующих синтаксических правил. Согласующая реализация SQL будет обрабатывать стандартный согласующий язык SQL в соответствии с Общими правилами.

Реализация, подтверждающая согласование SQL — DDL, будет обрабатывать SQL — DDL (<схема>) на уровне 1 или 2.

Реализация, подтверждающая согласование SQL — DML, будет проводить обработку на уровне 1 или 2:

1) прямого вызова указаний языка управления данными SQL (<оператор SQL>); и/или

2) модульного языка (<модуль>); и/или

3) одного или более:

a) встроенного SQL КОБОЛ (<встроенная программа SQL КОБОЛ>)

b) встроенного SQL ФОРТРАН (<встроенная программа SQL ФОРТРАН>)

c) встроенного SQL ПАСКАЛЬ (<встроенная программа SQL ПАСКАЛЬ>)

d) встроенного SQL ПЛ/1 (<встроенная программа SQL ПЛ/1>).

Реализация, подтверждающая полное согласование SQL, будет обеспечивать либо на уровне 1, либо на уровне 2 согласования SQL — DDL и SQL — DML.

Согласующая реализация может предоставлять дополнительные средства или варианты, не обусловленные в данном стандарте. Реализация остается согласующей даже в том случае, если она предоставляет пользователю варианты для работы с несогласующим языком SQL или для работы с согласующим языком SQL несогласованным способом.

Объявления согласования с настоящим стандартом будут устанавливать:

1) Какие из следующих типов согласования объявляются:
   a) полное согласование SQL с уровнем 1;
   b) полное согласование SQL с уровнем 2;
   c) согласование SQL — DDL с уровнем 1;
   d) согласование SQL — DDL с уровнем 2;
   e) согласование SQL — DML с уровнем 1;
   f) согласование SQL — DML с уровнем 2.
2) Какие из следующих средств реализованы:
   a) прямая обработка указаний языка управления данными SQL;
   b) модульный язык (<модуль>);
   c) встроенный SQL КОБОЛ (<встроенная программа SQL КОБОЛ>);
   d) встроенный SQL ФОРТРАН (<встроенная программа SQL ФОРТРАН>);
   e) встроенный SQL ПАСКАЛЬ (<встроенная программа SQL ПАСКАЛЬ>);
   f) встроенный SQL ПЛ/1 (<встроенная программа SQL ПЛ/1>).
3) Реализованы ли средства расширения целостности.

Настоящий стандарт не определяет метод или время связи между прикладными программами и компонентами системы управления базой данных.

4 ПОНЯТИЯ

4.1 Множества

Множество есть неупорядоченный набор определенных объектов.

Мультимножество — неупорядоченный набор объектов, которые не обязательно являются определенными.

Последовательность — это упорядоченный набор объектов, которые не обязательно являются различными.

Мощность набора — это количество объектов в этом наборе. Если не определено специально, любой набор может быть пустым.

4.2 Типы данных

Тип данных является множеством представляемых значений. Логическое представление значения — это <литерал>. Физическое представление значения определяется разработчиком.

Значение является базисным элементом, то есть оно не имеет логического подразделения в данном стандарте. Значение является нулевым или ненулевым значением.

Нулевое значение является специальным значением, которое задается разработчиком в зависимости от типа, то есть оно отличается от всех ненулевых значений этого типа.

Ненулевое значение является либо строкой символов, либо числом. Строка символов и число не являются сопоставимыми значениями.

4.2.1 Строки символов

Строка символов состоит из последовательности символов из множества символов, определенных разработчиком. Строка символов имеет длину, которая является положительным целым, определяющим количество символов в последовательности.

Все строки символов сопоставимы. Строка символов идентична другой строке символов, если и только если она равна этой строке символов в соответствии с правилами сопоставления, заданными в 5.11 «<предикат сравнения>».

4.2.2 Числа

Число является либо точным численным значением либо приближенным численным значением. Все числа являются сопоставимыми значениями.

Точное численное значение имеет точность и масштаб. Точность представления является положительным целым, которое определяет число значащих десятичных цифр. Масштаб является неотрицательным целым. Масштаб О указывает, что данное число является целым. Для масштаба N точное численное значение является целым значением значащих цифр, умноженных на 10 в степени $-N$.

Приближенное численное значение состоит из мантиссы и экспоненты. Мантисса является отмеченным знаком численным значением, а порядок — это отмеченное знаком целое, которое задает величину мантиссы. Приближенное численное значение имеет точность. Точность является положительным целым, которое определяет число значащих двоичных цифр в мантиссе.

Когда точное численное значение приписано элементу данных или параметру, представляющему точное численное значение, то приближение его значения, сохраняющее первые значащие цифры, представляется в типе элемента данных. Это значение преобразуется с тем, чтобы оно имело точность и масштаб элемента данных.

Когда точное или приближенное численное значение приписывается элементу данных или параметру, представляющему приближенное численное значение, то приближение его значения представляется в типе элемента данных. Это значение преобразуется с тем, чтобы оно имело точность цели.

4.3 Столбцы

Столбец является мультимножеством значений, которые могут изменяться со временем. Все значения в одном и том же столбце имеют один и тот же тип данных и являются значениями в одной и той же таблице. Значение столбца является наименьшей единицей данных, которая может быть выбрана из таблицы, и наименьшей единицей данных, которую можно обновить.

Столбец имеет описание и заданное положение в таблице. Описание столбца включает его тип данных и указание, ограничен ли столбец содержанием только ненулевых значений. Описание столбца строки символов задает длину строки символов. Описание приближенного численного столбца определяет точность его чисел. Описание точного численного столбца определяет точность и масштаб чисел в столбце.

Поименованный столбец является столбцом поименованной таблицы или столбцом, который наследует описание поименованного столбца. Описание поименованного столбца включает его имя.

4.4 Таблицы

Таблица является мультимножеством строчек. Строчка является непустой последовательностью значений. Каждая строчка одной и той же таблицы имеет одну и ту же мощность и содержит значение каждого столбца этой таблицы. *i*-е значение в каждой строчке таблицы представляет собой значение *i*-го столбца этой таблицы. Строчка является наименьшим элементом данных, который может быть вставлен в таблицу и удален из таблицы.

Порядком таблицы является количество столбцов в этой таблице. В любой момент времени порядок таблицы — это то же самое, что количество элементов (мощность) каждой из ее строк, а мощность таблицы — это то же самое, что и мощность каждого из ее столбцов.

Таблица имеет описание. Это описание включает в себя описание каждого из ее столбцов.

Базовая таблица — это поименованная таблица, определенная с помощью <определения таблицы>. Описание базовой таблицы включает в себя ее имя.

Производная таблица — это таблица, производная прямо или косвенно от одной или более других таблиц путем вычисления <спецификации запросов>. Значения производной таблицы являются значениями таблиц, лежащих в ее основе при ее получении.

Представляемая таблица является поименованной таблицей, которая определена <определением представления>. Описание представляемой таблицы включает ее имя.

Таблица является либо обновляемой, либо только считываемой. Операции вставки, корректировки и удаления разрешены для обновляемой таблицы и не разрешены для только считываемой.

Группированная таблица — это множество групп, выведенных при вычислении <спецификатора группировки>. Группой является мультимножество строк, в которых все значения группирующих столбцов (-а) равны. Группированные таблицы могут рассматриваться как набор таблиц. Установочные функции могут действовать на отдельные таблицы в рамках группированной таблицы.

Группированное представление — это представляемая таблица, выведенная из группированной таблицы.

4.5 **Ограничения целостности**

Ограничения целостности определяют достоверные состояния базы данных путем ограничения значений в базовых таблицах.

Ограничения целостности эффективно проверяются после выполнения каждого <оператора SQL>. Если базовая таблица, связанная с ограничениями целостности, не удовлетворяет этому ограничению целостности, то <оператор SQL> не оказывает воздействия, и параметр SQLCODE устанавливается в заданное разработчиком отрицательное число.

<Определение ограничения уникальности> требует, чтобы никакие две строки в таблице не имели одинаковых значений в заданных столбце или столбцах.

Определение NOT NULL (нулевое) требует, чтобы ни одно из значений в столбце не было нулевым значением.

<Определение ограничения на ссылки> требует, чтобы для каждой строки одной заданной таблицы, «обращающейся таблицы», значения заданного столбца или столбцов либо имели хотя бы одно нулевое значение, либо были такими же, как и значения заданного столбца или столбцов в некоторой строке другой задан-

ной таблицы, «таблицы, к которой обращаются». Обращающаяся таблица может быть такой же, как и таблица, к которой обращаются.

<Проверочное определение ограничения> требует, чтобы заданное <условие поиска> не было ложным в любой строке таблицы.

4.6 Схемы

<Схема> является устойчивым объектом, который задается с помощью языка определения схемы. Она состоит из <спецификатора полномочий схемы> и всех <определения (-ий) таблицы>, <определения (-ий) представления> и <определения (-ий) привилегий>, известных для системы для заданного <идентификатора полномочий> в данной среде. Понятие среды определяется разработчиком.

Таблицы, представления и привилегии, определенные <схемой>, считаются «присвоенными» или «созданными» <идентификатором полномочий>, заданным для данной <схемы>.

Примечание — Реализация может предоставлять средства (такие как DROP TABLE, DROP VIEW, ALTER TABLE и ALTER VIEW), которые дают возможность определения таблиц, представлений и привилегий для данного <идентификатора полномочий>, который создается, разрушается и модифицируется постепенно со временем. Однако этот стандарт только адресует <схему> (-ы), которые представляют определения, известные системе в данный момент времени.

4.7 База данных

База данных — это набор всех данных, определяемых <схемой> (-ами) в среде. Понятие среды задается разработчиком.

4.8 Модули

<Модуль> — это устойчивый объект, заданный на модульном языке. <Модуль> состоит из произвольного <имени модуля>, <спецификатора языка>, <спецификатора полномочий модуля>, нулевого или большего количества указателей, заданных <объявлением курсора>, и одной или более <процедурой> (-ами).

Прикладная программа является сегментом исполнительного кода, возможно, состоящего из множества подпрограмм. Одиночный <модуль> связан с прикладной программой во время ее выполнения. Прикладная программа будет связана по крайней мере с одним <модулем>. Способ задания этой связи, включая возможное требование к выполнению некоторого определенного разработчиком указания, определяется разработчиком.

4.9 **Процедуры**

<Процедура> состоит из <имени процедуры>, последовательности <объявления (-ий) параметров> и одиночного <оператора SQL>.

Прикладная программа, связанная с <модулем>, может обращаться к <процедуре> (-рам) этого <модуля> с помощью «запрашивающего» оператора, задающего <имя процедуры> для данной <процедуры>, и дает последовательность значений параметров, соответствующих по числу и по <типу данных> <объявлению параметра> (-ров) <процедуры>. Вызов <процедуры> обуславливает <оператор SQL>, содержащийся в ней, для исполнения.

4.10 **Параметры**

Параметр объявляется в <процедуре> <объявлением параметра> <Объявление параметра> задает <тип данных> его значения. Параметр либо предполагает, либо предоставляет значение соответствующего аргумента в вызове этой <процедуры>.

4.10.1 Параметр SQLCODE

Параметр SQLCODE является специальным целочисленным параметром. Его значение присваивается коду состояния, который либо показывает, что вызов <процедуры> успешно завершен, либо что имели место особые условия во время выполнения <процедуры>.

4.10.2 Параметры-признаки

Параметр-признак является целочисленным параметром, который задается после другого параметра. Его главная цель — указать, является ли нулевым значение, которое другой параметр приобретает или передает.

4.11 **Стандартные языки программирования**

Настоящий стандарт задает действие <процедуры> (-р) в <модуле> (-лях), когда эти <процедуры> вызываются программами, которые соответствуют заданным стандартным языкам программирования. Термины «стандартная программа КОБОЛ», «стандартная программа ФОРТРАН», «стандартная программа ПАСКАЛЬ», «стандартная программа ПЛ/1» относятся к программам, которые соответствуют критериям соответствия стандартам, приведенным в разделе 2 «Ссылки».

4.12 **Курсоры**

Курсор задается <объявлением курсора>.

Для каждого <объявления курсора> в <модуле> курсор эффективно создается, когда транзакция (см. 4.16 «Транзакция»),

обращающаяся к модулю, начинается, и разрушается, когда эта транзакция заканчивается.

Курсор находится либо в открытом, либо в закрытом состоянии. Начальным состоянием курсора является закрытое состояние. Курсор помещается в открытое состояние <оператором открытия> и возвращается в закрытое состояние <оператором закрытия>, <оператором блокировки> или <оператором возврата>.

Курсор в открытом состоянии обозначает таблицу, упорядочение строк в этой таблице и положение относительно этого упорядочения. Если <объявление курсора> не включает <спецификатор упорядочения> или если оно включает <спецификатор упорядочения>, который не полностью задает упорядочение строк, то строки в таблице имеют порядок, полностью или частично определяемый разработчиком.

В пределах одной транзакции, где упорядочение строк не определено или определено не полностью <спецификатором упорядочения>, то относительное положение двух строк будет в общем случае одинаковым в любой момент времени в период, когда курсор открыт. Упорядочение может изменяться от одного момента времени к другому, если значения базы данных или значения параметров, запрашиваемых в <спецификации курсора>, в <объявлении курсора>, различны.

В отдельных транзакциях, даже если значения баз данных и значения параметров, запрашиваемых в <спецификации курсора> в <объявлении курсора>, одинаковы, порядок строк, определенный одной и той же <спецификацией курсора> и <оператором открытия>, может быть различным.

Курсор располагается в открытом состоянии либо перед определенной строкой, на определенной строке, либо после последней строки. Если курсор находится на строке, то эта строка является текущей строкой для курсора. Курсор может находиться перед первой строкой или после последней строки, даже если таблица пуста.

<Оператор выборки> перемещает положение открытого курсора к следующей строке по порядку следования курсора и отыскивает значения столбцов этой строки. <Оператор корректировки: по положению> корректирует текущую строку курсора. <Оператор удаления: по положению> удаляет текущую строку курсора.

Если курсор находится перед строкой, и в это положение вставляется новая строка, то влияние, если оно есть, на положение курсора задается разработчиком.

Если курсор находится на строке или перед строкой, и эта строка удаляется, то курсор располагается перед строкой, находящейся непосредственно после положения удаляемой строки. Если такая строка не существует, то курсор располагается после последней строки.

Если происходит ошибка во время выполнения <оператора SQL>, который идентифицирует открытый курсор, то влияние, если оно есть, на положение или состояние этого курсора определяется разработчиком.

Рабочая таблица — это таблица, появляющаяся при открытии курсора. Разработчик определяет, приводит открытие курсора к появлению рабочей базовой таблицы или рабочей представленной таблицы.

Каждая строка рабочей представленной таблицы появляется только тогда, когда курсор устанавливается на эту строку.

Рабочая базовая таблица возникает, когда курсор открывается, и исчезает, когда курсор закрывается.

4.13 Операторы

<Оператор SQL> задает работу базы данных или работу курсора. <Оператор выборки> делает выборку значений из таблицы. <Оператор вставки> вставляет строки в таблицу. <Оператор корректировки: поиск> или <оператор корректировки: по положению> корректирует значения в строках таблицы. <Оператор удаления: поиск> или <оператор удаления: по положению> удаляет строки из таблицы.

4.14 Встроенный синтаксис

Встроенная общая программа SQL (<встроенная программа SQL КОБОЛ>, <встроенная программа SQL ФОРТРАН>, <встроенная программа SQL ПАСКАЛЬ> или <встроенная программа SQL ПЛ/1>) является прикладной программой, которая состоит из текста языка программирования и текста SQL. Текст языка программирования должен соответствовать требованиям заданного стандартного языка программирования. Текст SQL состоит из одного или более <операторов встроенного SQL> и, произвольно, одного или более разделов <встроенных разделов объявления SQL>. Это позволяет выражать прикладные базы данных в гибридной форме, в которой <оператор SQL> встроен непосредственно в прикладную программу. Такая гибридная прикладная программа по определению должна быть эквивалентной стандартной прикладной программе, в которой <оператор SQL> заменен стандартной процедурой или подпрограммой ВЫЗОВ <процедур> (-ы) SQL в отдельном <модуле> SQL.

4.15 Привилегии

Привилегия разрешает выполнение данной категории <действия> на заданной таблице или просмотр с помощью заданного <идентификатора полномочий>. Можно задать <операции> INSERT, DELETE, SELECT, UPDATE и REFERENCES.

<Идентификатор полномочий> задается для каждой <схемы> и <модуля>.

<Идентификатор полномочий>, заданный для <схемы>, должен отличаться от <идентификатора полномочий> любой другой <схемы> в данной среде. <Идентификатор полномочий> <схемы> является «владельцем» всех таблиц и представлений, определенных в этой <схеме>.

Таблицы и представления обозначаются <табличными именами>. <Табличное имя> состоит из <идентификатора полномочий> и <идентификатора>. <Идентификатор полномочий> идентифицирует <схему>, в которой была определена таблица или представление, <табличным именем>. Таблицы и представления, определенные в различных <схемах>, могут иметь один и тот же <идентификатор>.

Если обращение к <табличному имени> не содержит в явной форме <идентификатор полномочий>, то <идентификатор полномочий> содержащей его <схемы> или <модуля> задается по умолчанию.

<Идентификатор полномочий> <схемы> имеет все привилегии на таблицах и представлениях, определенные в данной <схеме>.

<Схема> с данным <идентификатором полномочий> может содержать <определения привилегий>, предоставляющие привилегии другим <идентификаторам полномочий>. Предоставленные привилегии могут применяться к таблицам и представлениям, определенным в текущей <схеме>, либо они могут быть привилегиями, которые предоставлены данному <идентификатору полномочий> другими <схемами>. Предложение WITH GRANT OPTION <определения привилегий> определяет, может ли получатель привилегии передать ее другим.

<Модуль> определяет <идентификатор полномочий>, <идентификатор полномочий модуля>, который должен иметь привилегии, заданные для каждого <оператора SQL> в <модуле>.

4.16 Транзакции

Транзакция является последовательностью операций, включая операции базы данных, которые являются элементарными по отношению к восстановлению и параллелизму. Транзакция начинает-

ся при вызове <процедуры> и при том, что в этот момент ни одна транзакция не выполняется. Транзакция заканчивается <оператором фиксации> или <оператором отката>. Если транзакция заканчивается <оператором фиксации>, то все изменения, произведенные в базе данных этой транзакцией, становятся доступными для всех параллельных транзакций. Если транзакция заканчивается <оператором отката>, то все изменения, которые произвела данная транзакция, уничтожаются. Заблокированные изменения не могут быть уничтожены. Изменения, произведенные в базе данных транзакцией, могут быть восприняты этой транзакцией, но пока эта транзакция не закончится <оператором блокировки>, они не могут восприниматься другими транзакциями.

Исполнение параллельных транзакций гарантированно переходит в последовательную форму. Последовательное исполнение определяется как исполнение операций параллельно выполняемых транзакций, которое оказывает такое же влияние, как и некое последовательное выполнение тех же самых транзакций. Последовательное выполнение таково, что в нем каждая транзакция выполняется так, чтобы выполнение закончилось до начала следующей транзакции.

Исполнение <оператора SQL> в пределах транзакции не оказывает влияния на базу данных, отличающегося от влияния, которое установлено в Общих правилах для этого <оператора SQL>.

Наряду с последовательным выполнением, это означает, что все операции считывания воспроизводимы в пределах транзакции, за исключением:

1) Влияние изменений на базу данных и ее содержание осуществляется исключительно самой транзакцией.

2) Влияние различий в значениях параметров, подвергаемых процедурам обработки, открывает курсоры для последовательного поиска с помощью этих курсоров.

5 ОБЩАЯ ЛЕКСИКА

5.1 <символ>

Функция

Определяет графические символы и элементы символьных строк, допустимые в языке.

Формат

<символ> : : =
<цифра> | <буква> | <специальный символ>

<цифра> : : =
0|1|2|3|4|5|6|7|8|9

<буква> : : =
<прописная буква>|<строчная буква>

<прописная буква> : : =
A|B|C|D|E|F|G|H|I
J|K|L|M|N|O|P|Q|R
S|T|U|V|W|X|Y|Z

<строчная буква> : : =
a|b|c|d|e|f|g|h|i
j|k|l|m|n|o|p|q|r
s|t|u|v|w|x|y|z

<специальный символ> : : =
см. Правило синтаксиса 1.

Правила синтаксиса

1) <специальный символ> — это любой символ из набора символов, определенного разработчиком реализации, отличный от <цифры> или <буквы>. Если разработчиком реализации в качестве указателя конца строки определен символ, то он исключается из набора специальных символов.

Примечание — См. 5.3 <лексема>, <новая строка> «Формат».

2) Набор <специальный символ> должен включать в себя все символы, отличные от <буквы> и <цифры>, включенные в наборы графики терминальных устройств, используемых при работе с языком SQL. В набор специальных символов должны обязательно входить символы «процент» и «подчеркивание».

Общие правила

Не имеется.

5.2 **<литерал>**

Функция

Определяет значение (кроме неопределенных).

Формат

<литерал> : : =
<строковый литерал>
|<числовой литерал>

<строковый литерал> : : =
'<символьное представление> ...

<символьное представление> : : =
<символ, отличный от кавычек>
|<кавычки>

<символ, отличный от кавычек> : : =
см. Правила синтаксиса 1.

<кавычки> : : =
«
<числовой литерал> : : =
<точный числовой литерал>
|<приближенный числовой литерал>
<точный числовой литерал> : : =
[+|—]{<целое без знака> [<целое без знака>]
|<целое без знака>.
| <целое без знака>}
<приближенный числовой литерал> : : =
<мантисса>E<порядок>
<мантисса> : : = <точный числовой литерал>
<порядок> : : = <целое со знаком>
<целое со знаком> : : = [+|—]<целое без знака>
<целое без знака> : : =
<цифра> ...

Правила синтаксиса

1) <символ, отличный от кавычки> — это любой символ, за исключением символа «одиночная кавычка (апостроф)» (').

2) Тип данных, к которому относится <строковый литерал>— это строка символов. Длина <строкового литерала> определяется как число <символьных представлений> в этом литерале. <кавычки> в <строковом литерале> — это символы «одиночная кавычка», определяющие как значение, так и длину <строкового литерала>.

3) В <точном числовом литерале> без десятичной точки (.) десятичная точка подразумевается после последней <цифры>.

4) <точный числовой литерал> относится к точному числовому типу данных. Точность <точного числового литерала> определяется количеством <цифр>, которые в него входят. Дробная часть <точного числового литерала> характеризуется количеством <цифр> справа от десятичной точки.

5) <приближенный числовой литерал> относится к приближенному числовому типу данных. Точность <приближенного числового литерала> — это точность его <мантиссы>.

Общие правила

1) Значением <строкового литерала> является последовательность содержащихся в нем <символов>.

2) Числовое значение <точного числового литерала> определяется путем обычной математической интерпретации, принятой в позиционной десятичной системе счисления для чисел со знаком.

3) Числовое значение <приближенного числового литерала> определяется как произведение точного числового значения, заданного в <мантиссе>, и числа, полученного путем возведения числа 10 в степень, заданную <порядком>.

5.3 **<лексема>**

Функция

Определяет лексические единицы.

Формат

<лексема> : : =
<лексема, не являющаяся ограничителем>
|<лексема-ограничитель>

<лексема, не являющаяся ограничителем> : : =
<идентификатор>
|<ключевое слово>
|<числовой литерал>

<идентификатор> : : =
<прописная буква>|{[<подчеркивание>|<буква или цифра>}...]

<подчеркивание> : : =

<буква или цифра> : : =
<прописная буква>|<цифра>

<ключевое слово> : : =
ALL|AND|ANY|AS|ASC|AUTHORIZATION|AVG
|BEGIN|BETWEEN|BY
|CHAR|CHARACTER|CHECK|CLOSE|COBOL|COMMIT
|CONTINUE|COUNT|CREATE|CURRENT|CURSOR
|DEC|DECIMAL|DECLARE|DEFAULT|DELETE|DESC|
|DISTINCT|DOUBLE
|END|ESCAPE|EXEC|EXISTS
|FETCH|FLOAT|FOR|FOREIGN|FORTRAN|FOUND|FROM
|GO|GOTO|GRANT|GROUP|HAVING
|IN|INDICATOR|INSERT|INT|INTEGER|INTO|IS
|KEY|LANGUAGE|LIKE
|MAX|MIN|MODULE|NOT|NULL|NUMERIC
|OF|ON|OPEN|OPTION|OR|ORDER
|PASCAL|PL1|PRECISION|PRIMARY|PRIVILEGES|
|PROCEDURE|PUBLIC
|REAL|REFERENCES|ROLLBACK
|SCHEMA|SECTION|SELECT|SET|SMALLINT|SOME
|SQL|SQLCODE|SQLERROR|SUM

|TABLE|TO|UNION|UNIQUE|UPDATE|USER
|VALUES|VIEW|WHENEVER|WHERE|WITH|WORK

<лексема-ограничитель> : : =
<строковый литерал>
| . | (|) |<|>|. |:|=| * |+|—|/|<>|>=|<=

<разделитель> : : =
{<комментарий>|<пробел>|<новая строка>}...

<комментарий> : : =
<начало комментария> [<символ>...] <новая строка>

<новая строка> : : =
признак конца строки, определенный разработчиком реализации

<пробел> : : = символ «пробел»

Правила синтаксиса

1) <лексема>, не являющаяся <строковым литералом>, не должна содержать <пробел>.

2) За каждой <лексемой> может следовать <разделитель>. За <лексемой, не являющейся ограничителем>, должен следовать <разделитель> или <лексема-ограничитель>. Если синтаксис не позволяет, чтобы за <лексемой, не являющейся ограничителем>, следовала <лексема-ограничитель>, то за такой <лексемой, не являющейся ограничителем>, должен следовать <разделитель>.

3) <идентификатор>может включать в себя не более 18 <символов>.

4) <идентификатор> не должен совпадать с <ключевым словом>.

5) <начало комментария> — это последовательность из двух или более следующих друг за другом дефисов (-), не разделенных ни <пробелом>, ни <новой строкой>, которая не входит в состав <строкового литерала>.

Общие правила

Не имеется.

5.4 **<имена>**

Функция

Задает имена.

Формат

<имя таблицы> : : =
[<идентификатор полномочий>.]<идентификатор таблицы>
<идентификатор полномочий> : : = <идентификатор>

<идентификатор таблицы> :: = <идентификатор>
<имя столбца> :: = <идентификатор>
<соотнесенное имя> :: = <идентификатор>
<имя модуля> :: = <идентификатор>
<имя курсора> :: = <идентификатор>
<имя процедуры> :: = <идентификатор>
<имя параметра> :: = <идентификатор>

Правила синтаксиса

1) <имя таблицы> идентифицирует именованную таблицу.

2) Если <имя таблицы> не включает в себя <идентификатор полномочий>, то:

a) если <имя таблицы> содержится в <схеме>, то неявным образом используется <идентификатор полномочий>, указанный в качестве <идентификатора полномочий схемы>;

b) если <имя таблицы> содержится в <модуле>, то неявным образом используется <идентификатор полномочий>, указанный в качестве <идентификатора полномочий модуля>.

3) Два <имени таблицы> считаются идентичными тогда и только тогда, когда совпадают их <идентификаторы таблицы> и <идентификатор полномочий>, независимо от того, в какой форме задан <идентификатор полномочий> — явной или неявной.

4) <имя таблицы> задается в <определении таблицы> или в <определении представления>.

5) В <SQL-операторе> <имя таблицы> должно указывать таблицу, определенную в <схеме>.

6) <идентификатор полномочий> представляет собой идентификатор полномочий доступа.

7) <соотнесенное имя> задается с помощью <идентификатора> и используется для указания таблицы в определенных контекстах. <соотнесенное имя> применяется в таких контекстах как <оператор выборки>, <подзапрос> или <спецификация запроса> (см. 5.20 <спецификатор отображения>). Эти контексты могут быть вложенными. Одно и то же <соотнесенное имя> в различных контекстах может относиться и к разным таблицам, и к одной таблице.

8) <имя столбца> идентифицирует именованный столбец. <имя столбца> задается в виде <идентификатора> либо в <определении таблицы>, либо в <определении представления>.

9) <имя модуля> идентифицирует <модуль>.

10) <имя курсора> идентифицирует <указатель>.

11) <имя процедуры> идентифицирует <процедуру>.

12) <имя параметра> идентифицирует параметр.

Общие правила
Не имеется.

5.5 <тип данных>

Функция

Задает тип данных.

Формат

```
<тип данных> ::=
                  <строковый тип>
                 |<точный числовой тип>
                 |<приближенный числовой тип>
<строковый тип> ::=
                CHARACTER [(<длина>)]
                |CHAR [(<длина>)]
<точный числовой тип> ::=
                |NUMERIC [(<точность>[, <масштаб>])]
                |DECIMAL [(<точность>[, <масштаб>])]
                |DEC [(<точность>[, <масштаб>])]
                |INTEGER
                [INT
                |SMALLINT
<приближенный числовой тип> ::=
                 FLOAT [(<точность>)]
                |REAL
                |DOUBLE PRECISION
<длина> ::= <целое без знака>
<точность> ::= <целое без знака>
<масштаб> ::= <целое без знака>
```

Правила синтаксиса

1) CHAR является синонимом CHARACTER. DEC является синонимом DECIMAL. INT является синонимом INTEGER.

2) <длина> и <точность> должны задаваться <целым без знака>, значение которого больше 0.

3) Если <длина> не указана, то она принимается за 1. Если не указан <масштаб>, то он принимается за 0. Если не указана <точность>, то принимается значение, определенное разработчиком реализации.

4) В <точном числовом типе> значение <масштаба> не должно превышать значения <точности>.

5) CHARACTER задает тип данных строки символов, длина которой указана в <длине>.

6) NUMERIC задает тип данных точного числа, точность и масштаб которого задается <точностью> и <масштабом>.

7) DECIMAL задает тип данных точного числа, масштаб которого задается <масштабом>, а точность определена разработчиком реализации и может быть равна или больше значения, заданного <точностью>.

8) INTEGER задает тип данных точного числа, точность которого определена разработчиком реализации, а значение масштаба принято равным 0.

9) SMALLINT задает тип данных точного числа с нулевым масштабом, точность которого, определенная разработчиком реализации, не превышает точности, определенной разработчиком реализации для типа INTEGER.

10) FLOAT задает тип данных приближенного числа, точность двоичной формы которого равна или больше значения, заданного в <точности>.

11) REAL задает тип данных приближенного числа, точность которого определена разработчиком реализации.

12) DOUBLE PRECISION задает тип данных приближенного числа, точность которого, определенная разработчиком реализации, превышает точность, определенную разработчиком реализации для REAL.

Общие правила

Не имеется.

5.6 <спецификация значения> и <спецификация цели>

Функция

Задает одно или более значений, параметров или переменных.

Формат

```
<спецификация значения> : : =
                    <спецификация параметра>
                   |<спецификация переменной>
                   |<литерал>
                   |USER
<спецификация цели> : : =
                    <спецификация параметра>
                   |<спецификация переменной>
<спецификация параметра> : : =
                    <имя параметра> [<параметр-признак>
<параметр-признак> : : =
                        [INDICATOR] <имя параметра>
<спецификация переменной> : : =
    <имя встроенной переменной> [<переменная-признак>]
```

<переменная-признак> ::=
[INDICATOR] <имя вложенной переменной>

Правила синтаксиса

1) <спецификация значения> задает такие значения, которые не выбираются из таблицы.

2) <спецификация параметра> идентифицирует параметр или же параметр с параметром-признаком. Параметр-признак относится к точному числовому типу данных с нулевой дробной частью. Особенности <точного числового типа> применительно к параметрам-признакам определяются разработчиком реализации.

3) <спецификация переменной> идентифицирует переменную, принадлежащую включающей программе или такую же переменную вместе с переменной-признаком. Переменная-признак должна относиться к тому типу данных, который определен разработчиком реализации для параметра-признаков.

4) <спецификация цели> задает параметр или переменную, которым можно присвоить значение.

5) <спецификация параметра> должна включаться в <модуль> <спецификация переменной> должна включаться во <встроенный SQL-оператор>.

6) USER относится к строчному типу данных, длина строки определяется разработчиком реализации.

Общие правила

1) Если <спецификация параметра> включает в себя <параметр-признак>, значение которой является отрицательной величиной, то <спецификация параметра> задает неопределенное значение. В противном случае <спецификация параметра> задает значение, равное значению параметра, указанного в <имени параметра>.

2) Если <спецификация переменной> включает в себя <переменную-признак>, значение которой является отрицательной величиной, то <спецификация переменной> задает неопределенное значение. В противном случае <спецификация переменной> задает значение, равное значению переменной, указанного в <имени встроенной переменной>.

3) <литерал> задает значение, выраженное этим литералом.

4) Словом USER задается значение, равное значению <идентификатора полномочий>, указанного в качестве <идентификатора полномочий модуля> в <модуле>, содержащем <SQL-оператор>, в результате выполнения которого вычисляется значение <спецификации переменной> USER.

5.7 <спецификация столбца>

Функция

Указывает именованный столбец.

Формат

<спецификация столбца> : : =
[<префикс>] <имя столбца>

<префикс> : : =
<имя таблицы> | <соотнесенное имя>

Правила синтаксиса

1) <спецификация столбца> указывает именованный столбец. Смысл такого указания зависит от контекста.

2) <имя столбца> в <спецификации столбца> будем обозначать буквой *C*.

3) Возможны следующие варианты:

a) если <спецификация столбца> включает в себя <префикс>, то <спецификация столбца> должна применяться в тех контекстах, где однократно или многократно используется <имя таблицы> или же <соотнесенное имя>, заданное <префиксом>. Если такое <имя таблицы> или <соотнесенное имя> используется многократно, то <спецификация столбца> будет отнесена к тому из них, которое используется в наиболее локальном контексте. Таблица, обозначенная <именем таблицы>, должна включать в себя столбец, имеющий <имя столбца> *C*;

b) если <спецификация столбца> не включает в себя <префикс>, то она должна включаться в те контексты, где однократно или многократно используются имена <имя таблицы> или <соотнесенное имя>. Положим, что понятие «возможны префиксы» означает такие имена <имя таблицы> и <соотнесенное имя>, которые относятся к таблице, включающей в себя столбец с <именем столбца> *C*. При этом в качестве неявно заданного <имени таблицы> или же <соотнесенного имени> может использоваться только один из возможных префиксов — тот, который входит в наиболее локальный контекст.

Примечание — Области применения («контекст») <имени таблицы> или <соотнесенного имени> указаны в 5.20 <спецификатор отображения>, в 6.2< определение таблицы>, в 8.5 <оператор удаления: поиск>, в 8.11 <оператор корректировки: по положению>, в 8.12 <оператор корректировки: поиск>.

4) Если <спецификация столбца> входит в <выражение над таблицами> *T*, а явно или неявно заданный <префикс>, относящийся к <спецификации столбца>, употребляется в контексте какого-либо <SQL-оператора> или же <выражения над таблицами>, куда входит <выражение над таблицами> *T*, то <спецификация столбца> представляет собой «внешнюю ссылку» на таблицу, заданную данным <префиксом>.

5) Таблицу, к которой относится явно или неявно заданный <префикс> *R*, обозначим буквой *T*. Типом данных <спецификации столбца> является тип данных столбца *C* таблицы *T*.

Общие правила

1) «*C*» или «*R.C*» указывает столбец *C* в соответствующей строке таблицы *T*.

5.8 <спецификация функции набора>

Функция

Задает значение, которое является результатом функции от аргумента.

Формат

<спецификация функции набора> :: =
COUNT (*) | <функция набора различимых объектов> | <функция мультинабора>

<функция набора различимых объектов> :: =
{AVG | MAX | MIN | SUM | COUNT} (DISTINCT <спецификация столбца>)

<функция мультинабора> :: =
{AVG | MAX | MIN | SUM} ([ALL] <выражение значения>)

Правила синтаксиса

1) Аргументом функции COUNT (*) и источником аргументов для <функции набора различимых объектов> и <функции мультинабора> является таблица или группа в сгруппированной таблице (см. 5.19 <табличное значение>, 5.24 <подзапрос>, 5.25 <спецификация запроса>).

2) Аргумент или источник аргументов для <спецификации функции набора> обозначим буквой *R*.

3) <спецификация столбца> в <функции набора различимых объектов> и каждая <спецификация столбца> в <выражении над значениями > в <функции мультинабора> должна однозначным образом указывать столбец в *R* и не должна быть ссылкой на столбец, полученный в результате выполнения функции, заданной <спецификацией функции набора>.

4) <выражение значения> в <функции мультинабора> должно включать в себя <спецификацию столбца>, указывающую столбец в *R*, и не должно включать <спецификацию функции набора>. Если <спецификация столбца> является внешней ссылкой, то <выражение значения> не должно включать в себя каких-либо операций.

Примечание — Понятие «внешняя ссылка» определено в 5.7 <спецификация столбца>.

5) Если <спецификация функции набора> содержит <спецификацию столбца>, которая является внешней ссылкой, то такая <спецификация функции набора> должна входить в <подзапрос> в составе <спецификатора выборки групп>.

Примечание — Понятие «внешняя ссылка» определено в 5.7 <спецификация столбца>.

6) Тип данных, к которому относятся значения, полученные в результате вычисления <спецификации столбца> или <выражения значения>, обозначим буквой *T*.

7) Если задана функция COUNT, то результатом <спецификации функции набора> будет значение, относящееся к точному числовому типу данных с точностью, определяемой разработчиком реализации, и нулевой дробной частью.

8) Если задана функция MAX или MIN, то результат будет относиться к типу данных *T*.

9) Если задана функция SUM или AVG, то:

a) *T* не должен быть строкой символов;

b) если задана функция SUM, а *T* — точный числовой тип, то результат будет относиться к точному числовому типу данных, точность и дробная часть которого определяются разработчиком реализации;

c) если задана функция AVG, а *T* — точный числовой тип, то результат будет относиться к точному числовому типу данных, точность и дробная часть которого определяются разработчиком реализации;

d) если *T* — приближенный числовой тип, то результат будет относиться к приближенному числовому типу с точностью, определяемой разработчиком реализации.

Общие правила

1) Аргументом <функции набора различимых объектов> является набор значений. Этот набор получается в результате исключения всех неопределенных значений и всех избыточных дубликатных значений из столбца в *R*, заданного <спецификацией столбца>.

2) Аргументом <функции мультинабора> является мультинабор. Этот набор получается в результате исключения всех неопределенных значений из набора, полученного посредством применения <выражения значения> для каждой строки в *R*. Смысл <функции мультинабора> не зависит от того, задано ключевое слово ALL или нет.

3) Аргумент <функции набора различимых объектов> или <функции мультинабора> обозначим буквой *S*.

4) Возможны следующие варианты:

a) если задана <функция набора различимых объектов> COUNT, то результатом будет мощность *S*;

b) если задана функция COUNT (*), то результатом будет мощность *R*;

c) если заданы функции AVG, MAX, MIN или SUM, а *S* — пустой набор, то результатом будет нулевое значение;

d) если задана функция MAX или MIN, то результатом будет, соответственно, максимальное или минимальное значение в *S*. Эти значения определяются с помощью правил сравнения, описанных в 5.11 <предикат сравнения>;

e) если задана функция SUM, то результатом будет сумма значений, составляющих *S*. Сумма не должна выходить за пределы диапазона, обусловленного типом данных результата;

f) если задана функция AVG, то результатом будет среднее от значений, составляющих *S*. Сумма не должна выходить за пределы диапазона, обусловленного типом данных результата.

5.9 <выражение значения>

Функция

Задает значение.

Формат

```
<выражение значения> : : =
                <терм>
              | <выражение значения> + <терм>
              | <выражение значения> — <терм>
<терм> : : =
             <коэффициент>
           | <терм>  * <коэффициент>
           | <терм> / <коэффициент>
<коэффициент> : : =
```

[+|—] <первичное>

<первичное> :: =
<спецификация значения>
|<спецификация столбца>
|<спецификация функции набора>
|(<выражение значения>)

Правила синтаксиса

1) Если <выражение значения> включает в себя <функцию набора различимых объектов>, то оно не должно включать каких-либо бинарных операций.

2) Первый <символ> <лексемы>, следующей за унарной операцией, не должен быть знаком «плюс» или «минус».

3) Если <первичное> относится к строковому типу данных, то <выражение значения> не должно включать в себя никаких операций. Результат будет относиться к строковому типу данных.

4) Если оба операнда относятся к точному числовому типу данных, то результат будет относиться к точному числовому типу, точность и масштаб которого определяются следующим образом:

a) обозначим масштаб первого и второго операндов как *s1* и *s2* соответственно;

b) точность результата операций сложения и вычитания определяется разработчиком реализации, а масштаб определяется как max (*s1*, *s2*);

c) точность результата операции умножения определяется разработчиком реализации, а масштаб определяется как *s1*+*s2*;

d) точность и дробная часть результата операции деления определяется разработчиком реализации.

5) Если хотя бы один из операндов, участвующих в операции, относится к приближенному числовому типу данных, то результат будет относиться к приближенному числовому типу. Точность результата определяется разработчиком реализации.

Общие правила

1) Если значение любого из <первичных> является неопределенным, то результат <выражения значения> будет неопределенным.

2) Если операции не заданы, то результатом <выражения значения> будет значение заданного <первичного>.

3) Если <выражение значения> применяется к строке таблицы, то указание столбца означает указание значения элемента данного столбца в данной строке.

4) Унарные арифметические операции + и — представляют собой унарные плюс и минус соответственно. Унарный плюс не изменяет того операнда, к которому он относится. Унарный минус рассматривается как знак операнда.

5) Бинарные арифметические операции +, —, * и / означают, соответственно, сложение, вычитание, умножение и деление. Делитель не должен быть нулем.

6) Если результат арифметической операции относится к точному числовому типу данных, то возможны следующие варианты:

a) если данная операция не является делением, то математический результат операции может быть представлен точным числом, где точность и дробная часть обусловлены типом данных результата;

b) если данная операция является делением, то математический результат операции представляется приближенным числом, точность и дробная часть которого обусловлены типом данных результата, при этом потеря хотя бы одного старшего значащего разряда является недопустимой.

7) Первыми вычисляются выражения, заключенные в скобки, а если скобки отсутствуют, то первыми выполняются вычисления, соответствующие унарным операциям, затем умножение и деление, а затем сложение и вычитание, операции одной и той же очередности выполняются слева направо.

5.10 **<предикат>**

Функция

Задает условие, в результате вычисления которого может быть получено логическое значение «истина», «ложь» или «неизвестное».

Формат

```
<предикат> : : =
                <предикат сравнения>
              | <предикат интервала>
              | <предикат принадлежности>
              | <предикат подобия>
              | <предикат нуля>
              | <квантифицированный предикат>
              | <предикат существования>
```

Правила синтаксиса

Не имеется.

Общие правила

1) Результатом <предиката> является результат его применения к заданной строке таблицы.

5.11 <**предикат сравнения**>

Функция

Задает сравнение двух значений.

Формат

<предикат сравнения> : : =
<выражение значения>
<операция сравнения> {<выражение значения>
|<подзапрос>}
<операция сравнения> : : =
=|<>|<|>|<=|>=

Правила синтаксиса

1) Типы данных первого и второго <выражения значения> или же <подзапроса> должны быть сравнимы между собой.

Общие правила

1) Результат первого <выражения значения> будем обозначать буквой *x*, а результат <подзапроса> или же второго <выражения значения> — буквой *y*. Мощность результата <подзапроса> не должна превышать 1.

2) Если *x* и *y* представляют собой неопределенные значения или же результатом <подзапроса> является пустой набор, то результатом выражения «*x*<операция сравнения>*y*» будет неизвестное значение.

3) Если *x* и *y* являются определенными значениями, то выражение «*x*<операция сравнения>*y*» может принимать значения «истина» или «ложь»:

«*x*=*y*» истинно тогда и только тогда, когда *x* равно *y*;

«*x*<>*y*» истинно тогда и только тогда, когда *x* не равно *y*;

«*x*<*y*» истинно тогда и только тогда, когда *x* меньше *y*;

«*x*>*y*» истинно тогда и только тогда, когда *x* больше *y*;

«*x*<=*y*» истинно тогда и только тогда, когда *x* не больше *y*;

«*x*>=*y*» истинно тогда и только тогда, когда *x* не меньше *y*;

4) Числовые типы сравниваются как алгебраические величины.

5) Сравнение двух символьных строк — это сравнение <символ>ов, занимающих позиции с одним и тем же порядковым номером. Если символьные строки имеют неодинаковую длину, то для сравнения используется рабочая копия более короткой строки, дополненная с правой стороны <пробел>ами таким образом, чтобы ее длина совпадала с длиной более длинной строки.

6) Две символьные строки равны между собой, если равны между собой все <символ>ы, занимающие позиции с одним и тем же порядковым номером. Если две строки не равны, то их

отношение определяется путем сравнения между собой первой слева пары несовпадающих <символ>ов. Это сравнение производится в соответствии со схемой упорядочения, определенной разработчиком реализации.

7) Хотя результатом выражения «$x=y$», когда x и y являются неопределенными значениями, будет неизвестное значение, в контекстах GROUP BY, OPDER BY и DISTINGT два неопределенных значения считаются тождественными друг другу или же дубликатными.

5.12 **<предикат интервала>**

Функция

Задает сравнение с граничными значениями интервала.

Формат

<предикат интервала> : : =
<выражение значения>
[NOT] BETWEEN<выражение значения>
AND<выражение значения>

Правила синтаксиса

1) Типы данных всех трех <выражений значения> должны быть сравнимы между собой.

Общие правила

1) Обозначим буквами x, y и z, соответственно, результаты первого, второго и третьего <выражения значения>.

2) Результат предиката «x BETWEEN y AND z» совпадает с результатом выражения «$x>=y$ and $x<=z$».

3) Результат предиката «x NOT BETWEEN y AND z» совпадает с результатом выражения «NOT x BETWEEN y AND z».

5.13 **<предикат принадлежности>**

Функция

Задает квантифицированное сравнение.

Формат

<предикат принадлежности> : : =
<выражение значения> [NOT] IN {<подзапрос>
| (<список принадлежащих значений>)}
<список принадлежащих значений> : : =
<спецификация значения> {, <спецификация значения>}...

Правила синтаксиса

1) Типы данных первого <выражения значения> и <подзапроса> или же первого <выражения значения> и всех специфи-

каций <спецификация значения> из <списка принадлежащих значений> должны быть сравнимы между собой.

Общие правила

1) Результат <выражения значения> обозначим *x*. Результат <подзапроса>, как и в <квантифицированном предикате>, обозначим *S*. Так же будем обозначать значения, заданные в <списке принадлежащих значений>, которые рассматриваются как элементы строк единственного столбца таблицы-столбца.

2) Результат предиката «*x* IN *S*» равен результату «*x*=ANY *S*». Результат предиката «*x* NOT IN *S*» равен результату «NOT *x* IN *S*».

5.14 <предикат подобия>

Функция

Задает сравнение с шаблоном.

Формат

<предикат подобия> : : =
<спецификация столбца> [NOT] LIKE <шаблон>
[ESCAPE <спецсимвол>]
<шаблон> : : = <спецификация значения>
<спецсимвол> : : = <спецификация значения>

Правила синтаксиса

1) <спецификация столбца> должна относиться к столбцу строкового типа.

2) <шаблон> *должен относиться к* строковому типу данных.

3) <спецсимвол> должен представлять собой строку, включающую в себя один символ.

Общие правила

1) Обозначим значение, указанное с помощью <спецификации столбца>, буквой *x*, а результат <спецификации значения>, с помощью которой задан <шаблон>, — буквой *y*.

2) Возможны следующие варианты:

а) если задан <спецсимвол>, то:

— обозначим результат <спецификации значения>, задающей <спецсимвол>, буквой *z*;

— необходимо иметь строку *y*, которая состоит из подстрок, включающих в себя либо 1, либо 2 символа; подстрока длиной в 1 символ не может быть спецсимволом *z*, а в каждой подстроке длиной 2 символа первым должен быть спецсимвол *z*, а вторым — либо спецсимвол *z*, либо символ «подчеркивание», либо символ «процент». При таком формате строки *y* каждая из двух символьных

подстрок обозначает однократное появление второго символа подстроки. Односимвольная строка, представляющая собой символ «подчеркивание», — это признак произвольного символа. Односимвольная подстрока, представляющая собой символ «процент», — это признак произвольной строки символов. Все односимвольные подстроки, которые не являются символом «подчеркивание» или же символом «процент», обозначают те символы, которые составляют эти подстроки;

b) если <спецсимвол> не задан, то каждый символ «подчеркивание» в составе строки *y* является признаком произвольного символа, каждый символ «процент» — признаком произвольной строки символов, а каждый символ строки *y*, отличный от символов «подчеркивание» и «процент», обозначает сам себя.

3) Строка символов *y* представляет собой последовательность, включающую в себя минимальное количество признаков подстрок, при этом каждый <символ> строки *y* может входить в состав только одного признака подстроки. Признак подстроки — это признак произвольного символа или признак произвольной строки символов, или любая последовательность <символ>ов, отличных от признака произвольного символа или признака произвольной строки символов.

4) Результатом предиката «*x* LIKE *y*» будет неизвестное значение, если *x* или *y* — неопределенные значения. Если *x* и *y* являются определенными значениями, то результатом «*x* LIKE *y*» будут либо значение «истина», либо значение «ложь».

5) Результатом предиката «*x* LIKE *y*» будет значение «истина», если *x* можно разделить на подстроки следующим образом:

a) подстрока в составе *x* представляет собой последовательность, состоящую из нуля или более соседних <символ>ов из *x*, причем каждый <символ> из *x* входит в состав только одной подстроки;

b) если *i*-й признак подстроки в *y* является признаком произвольного символа, то *i*-я подстрока в *x* включает в себя один любой символ;

c) если *i*-й признак подстроки в *y* является признаком произвольной строки символов, то *i*-я подстрока в *x* включает в себя любую последовательность из нуля или более <символ>ов;

d) если *i*-й признак подстроки в *y* не является ни признаком произвольного символа, ни признаком произвольной строки символов, то *i*-я подстрока в *x* должна совпадать

с признаком подстроки и иметь ту же длину, что и данный признак подстроки;

е) количество подстрок в *x* должно быть равно количеству признаков подстрок в *y*.

6) результат предиката «*x* NOT LIKE *y*» равен результату «NOT (*x* LIKE *y*)».

5.15 **<предикат нуля>**

Функция

Задает проверку на неопределенное значение.

Формат

```
<предикат нуля> ::=
        <спецификация столбца> I [NOT] NULL
```

Правила синтаксиса

Не имеется.

Общие правила

1) Обозначим значение, указанное с помощью <спецификации столбца> буквой *x*.

2) Результатом предиката «*x* IS NULL» может быть либо значение «истина», либо значение «ложь».

3) Результатом «*x* IS NULL» будет «истина» тогда и только тогда, когда *x* является неопределенным значением.

4) Результат предиката «*x* IS NOT NULL» равен результату «NOT (*x* IS NULL)».

5.16 **<квантифицированный предикат>**

Функция

Задает квантифицированное сравнение.

Формат

```
<квантифицированный предикат> ::=
        <выражение значения>
        <операция сравнения> <квантор>
        <подзапрос>
<квантор> ::=
        <все> | <некоторые>
<все> ::= ALL
<некоторые> ::= SOME | ANY
```

Правила синтаксиса

1) Типы данных <выражения значения> и <подзапроса> должны быть сравнимы между собой.

Общие правила

1) Результат <выражения значения> обозначим буквой *x*, а результат <подзапроса> — буквой *S*.

2) Результатом предиката «*x* <операция сравнения> <квантор> *S*» будет результат неявно заданного <предиката сравнения> «*x* <операция сравнения> *S*», примененного для каждого значения из *S*.

Возможны следующие варианты:

a) если *S* — пустой набор или же результатом неявно заданного <предиката сравнения> для каждого значения *s* из *S* является «истина», то результатом предиката «*x* <операция сравнения> <все> *S*» будет значение «истина»;

b) если хотя бы для одного значения *s* из *S* результатом неявно заданного <предиката сравнения> будет «ложь», то результатом предиката «*x* <операция сравнения> <все> *S*» будет значение «ложь»;

c) если хотя бы для одного значения *s* из *S* результатом неявно заданного <предиката сравнения> будет «истина», то результатом предиката «*x* <операция сравнения> <некоторые> *S*» будет значение «истина»;

d) если *S* — пустой набор, или же результатом неявно заданного <предиката сравнения> для каждого значения *s* из *S* является «ложь», то результатом предиката «*x* <операция сравнения> <некоторые> *S*» будет значение «ложь»;

e) если результатом предиката «*x* <операция сравнения> <квантор> *S*» не является ни значение «истина», ни значение «ложь», то его результатом будет неизвестное значение.

5.17 <предикат существования>

Функция

Задает проверку на пустой набор.

Формат

<предикат существования> :: =
EXISTS <подзапрос>

Правила синтаксиса

Не имеется.

Общие правила

1) Результат <подзапроса> обозначим буквой *S*.

2) Результатом предиката «EXISTS *S*» может быть либо значение «истина», либо значение «ложь».

3) Результатом предиката «EXISTS *S*» будет значение «истина» тогда и только тогда, когда *S* — непустой набор.

5.18 **<условие поиска>**

Функция

Задает условие, которое может принимать значения «истина», «ложь» или «неизвестное» в зависимости от результата выполнения логических операций над заданными условиями.

Формат

```
<условие поиска> ::=
                    < логический терм>
                  |<условие поиска> OR <логический терм>
<логический терм> ::=
        <логический коэффициент>
      |<логический терм> AND <логический коэффициент>
<логический коэффициент> ::=
                    [NOT] <логическое первичное>
<логическое первичное> ::=
                    <предикат> | (<условие поиска>)
```

Правила синтаксиса

1) <Спецификация столбца> или <выражение значения>, заданное в <условие поиска>, непосредственно содержится в этом <условии поиска>, если <спецификация столбца> или <выражение значения> не указано в <спецификации функции набора> или в <подзапросе> этого <условия поиска>.

Общие правила

1) Результатом <условия поиска> будет результат применения логических операций к тем условиям, которые получены при применении заданного <предиката> к данной строке таблицы или же к данной группе сгруппированной таблицы. Если же логические операции не заданы, то результатом <условия поиска> будет результат заданного <предиката>.

2) Результатом NOT («истина») является значение «ложь», результатом NOT («ложь») является значение «истина», результатом NOT («неизвестное») является неизвестное значение. Операции AND и OR определены в приведенных ниже истинностных таблицах.

AND	истина	ложь	неизвестное
истина	истина	ложь	неизвестное
ложь	ложь	ложь	ложь
неизвестное	неизвестное	ложь	неизвестное

OR	истина	ложь	неизвестное
истина	истина	истина	истина
ложь	истина	ложь	неизвестное
неизвестное	истина	неизвестное	неизвестное

3) Первыми вычисляются те выражения, которые заключены в скобки, а если скобки отсутствуют, то первой выполняется операция NOT, затем операция AND, затем операция OR, операции одной и той же очередности выполняются слева направо.

4) Если <условие поиска> применяется к строке таблицы, то каждая <спецификация столбца>, непосредственным образом заданная в <условии поиска>, указывает на значение в данной строке таблицы, принадлежащее заданному столбцу.

5.19 **<табличное выражение>**

Функция

Задает таблицу или же сгруппированную таблицу.

Формат

<табличное выражение> :: =
<спецификатор отображения>
[<спецификатор выборки>]
[<спецификатор группировки>]
[<спецификатор выборки групп>]

Правила синтаксиса

1) Если таблица, указанная в <спецификаторе отображения>, является сгруппированным представлением, то <табличное выражение> не должно включать в себя <спецификатор выборки>, <спецификатор группировки> или <спецификатор выборки групп>.

Общие правила

1) Если в <табличном выражении> отсутствуют необязательные спецификаторы, то заданная им таблица представляет собой результат <спецификатора отображения>. В противном случае каждый заданный спецификатор применяется к результату предыдущего спецификатора, и заданная выражением таблица представляет собой результат последнего из заданных спецификаторов. Таблица, полученная в качестве результата <табличного выражения> такова, что *i*-й столбец производной таблицы наследует описание *i*-го столбца таблицы, заданной <спецификатором отображения>.

5.20 <спецификатор отображения>

Функция

Задает таблицу, которая получается из одной или более именованных таблиц.

Формат

<спецификатор отображения> : : =
FROM <ссылка на таблицу> [{,<ссылка на таблицу>}...]
<ссылка на таблицу> : : =
<имя таблицы> | <соотнесенное имя>

Правила синтаксиса

1) <имя таблицы>, заданное в <ссылке на таблицу>, будет открыто в содержащем эту <ссылку на таблицу> <спецификаторе отображения> тогда и только тогда, когда в этой <ссылке на таблицу> не будет задано <соотнесенное имя>.

2) <имя таблицы>, открытое в <спецификаторе отображения>, должно отличаться от всех остальных имен <имя таблицы>, открытых в данном <спецификаторе отображения>.

3) <соотнесенное имя>, заданное в <ссылке на таблицу>, должно отличаться от всех остальных имен <соотнесенное имя>, заданных в содержащем данную <ссылку на таблицу> <спецификаторе отображения>, кроме того оно не должно совпадать с <идентификатором таблицы> какого-либо <имени таблицы>, открытого в данном <спецификаторе отображения>.

4) Контекстом <соотнесенного имени> и открытого <имени таблицы>, заданных в <спецификаторе отображения>, является самый внутренний <подзапрос>, <спецификация запроса> или <оператор выборки>, которые содержат <табличное выражение>, включающее в себя <спецификатор отображения> <спецификатор отображения> является контекстом заданного в нем <имени таблицы> тогда и только тогда, когда это <имя таблицы> открыто в данном <спецификаторе отображения>.

5) Если таблица, которую идентифицирует <имя таблицы>, является сгруппированным представлением, то <спецификатор отображения> должен включать в себя только одну <ссылку на таблицу>.

6) Возможны следующие варианты:

a) если <спецификатор отображения> содержит только одно <имя таблицы>, то описание результата <спецификатора отображения> совпадает с описанием таблицы, которую идентифицирует данное <имя таблицы>;

b) если <спецификатор отображения> содержит более одного <имени таблицы>, то описание результата <спе-

цификатора отображения> будет представлять собой конкатенацию описаний тех таблиц, имена которых заданы в <спецификаторе отображения>. Порядок конкатенации описаний определяется очередностью включения имен <имя таблицы> в <спецификатор отображения>.

Общие правила

1) Когда в <ссылке на таблицу> задается <соотнесенное имя> или же открытое <имя таблицы>, то это <соотнесенное имя> или <имя таблицы> определяется тем самым в качестве указателя таблицы, идентифицируемой <именем таблицы> из данной <ссылки на таблицу>.

2) Возможны следующие варианты:

a) если <спецификатор отображения> содержит только одно <имя таблицы>, то результатом <спецификатора отображения> будет таблица, идентифицируемая данным <именем таблицы>;

b) если <спецификатор отображения> содержит более одного <имени таблицы>, то результатом <спецификатора отображения> будет расширенное Декартово произведение идентифицируемых этими именами таблиц. Расширенное Декартово произведение R представляет собой мультинабор, включающий в себя все строки r, которые являются результатом конкатенации строк всех идентифицированных таблиц, причем порядок конкатенации определяется очередностью идентификации этих таблиц. Мощность мультинабора R представляет собой произведение мощностей идентифицированных таблиц. Порядковый номер столбца в R равен $r+s$, где r — порядковый номер данного столбца в именованной таблице T, из которых он взят, а s равно сумме числа столбцов во всех таблицах, идентифицированных в <спецификаторе отображения> перед T.

5.21 **<спецификатор выборки>**

Функция

Задает таблицу, полученную путем применения <условия поиска> к результату предшествующего <спецификатора отображения>.

Формат

<спецификатор выборки> : : =
WHERE <условие поиска>

Правила синтаксиса

1) Описание результата предшествующего <спецификатора отображения> обозначим буквой *T*. Каждая <спецификация столбца>, непосредственным образом заданная в <условии поиска>, должна однозначным образом указывать столбец в *T* или же являться внешней ссылкой.

Примечание — Понятие «внешняя ссылка» определено в 5.7 <спецификация столбца>.

2) <Выражение значения>, непосредственным образом заданное в <условии поиска>, не должно включать в себя ссылок на столбец, которые являются результатом функции.

3) Если <выражение значения>, непосредственным образом заданное в <условии поиска>, представляет собой <спецификацию функции набора>, то <спецификатор выборки> должен входить в <спецификатор выборки групп>, а <спецификация столбца> в <спецификации функции набора> должна являться внешней ссылкой.

Примечание — Понятие «внешняя ссылка» определено в 5.7 <спецификация столбца>.

Общие правила

1) Обозначим результат <спецификатора отображения> буквой *R*.

2) <условие поиска> применяется к каждой строке из *R*. Результатом <спецификатора выборки> является таблица, составленная из тех строк *R*, для которых <условие поиска> принимает значение «истина».

3) Все <подзапрос>ы из <условия поиска> выполняются для каждой строки из *R*, а полученные результаты используются при применении к данной строке *R* <условия поиска>. Если какой-либо из выполненных <подзапрос>ов содержит внешнюю ссылку на столбец *R*, то это будет ссылка на значение элемента данного столбца в данной строке *R*.

Примечание — Понятие «внешняя ссылка» определено в 5.7 <спецификация столбца>.

5.22 <спецификатор группировки>

Функция

Задает сгруппированную таблицу, полученную путем применения <спецификатора группировки> к результату предшествующего спецификатора.

Формат

<спецификатор группировки> :: =

GROUP BY <спецификация столбца>
[{, <спецификация столбца>}...]

Правила синтаксиса

1) Описание результата предшествующего <спецификатора отображения> или же <спецификации выборки> обозначим буквой *T*.

2) Каждая <спецификация столбца> в <спецификаторе группировки> должна однозначным образом указывать столбец в *T*. Столбец, заданный в <спецификаторе группировки>, является группирующим столбцом.

Общие правила

1) Результат предшествующего <спецификатора отображения> или же <спецификатора выборки> обозначим буквой *R*.

2) В результате применения <спецификатора группировки> *R* разбивается на набор групп. Набор включает в себя минимальное количество групп таких, что для каждого группирующего столбца во всех группах, состоящих более чем из одной строки, соблюдается условие, что все значения группирующего столбца должны быть идентичны.

3) Каждая строка данной группы содержит одно и то же значение данного группирующего столбца. Если к группе применяется <условие поиска> или же <выражение значения>, то ссылка на группирующий столбец является ссылкой на это значение.

5.23 **<спецификатор выборки групп>**

Функция

Задает ограничение для сгруппированной таблицы, полученной в результате применения предшествующего <спецификатора группировки> или же <спецификатора отображения>, которое состоит в том, что исключаются группы, не удовлетворяющие <условию поиска>.

Формат

<спецификатор выборки групп> : : =
HAVING <условие поиска>

Правила синтаксиса

1) Описание результата предшествующего <спецификатора отображения>, <спецификатора выборки> или же <спецификатора группировки> обозначим буквой *T*. Каждая <спецификация столбца>, непосредственным образом заданная в <условии поиска>, должна однозначным образом указывать группирующий столбец из *T* или же являться внешней ссылкой.

Примечание — Понятие «внешняя ссылка» определено в 5.7 <спецификация столбца>.

2) Каждая <спецификация столбца>, которая входит в <подзапрос> в <условии поиска> и указывает столбец в *T*, должна указывать в *T* группирующий столбец, в противном случае она должна задаваться посредством <спецификации функции набора>.

Общие правила

1) Результат предшествующего <спецификатора отображения>, <спецификатора выборки> или же <спецификатора группировки> обозначим буквой *R*. Если этот спецификатор не является <спецификатором группировки>, то *R* включает в себя единственную группу и не имеет группирующего столбца.

2) <условие поиска> применяется к каждой группе из *R*. Результатом <спецификатора выборки групп> будет сгруппированная таблица, в которую войдут те группы из *R*, для которых результатом <условия поиска> будет значение «истина».

3) Когда <условие поиска> применяется к данной группе из *R*, то эта группа является аргументом или же источником аргументов для всех спецификаций <спецификация функции набора>, которые непосредственным образом входят в <условие поиска>, если только <спецификация столбца> в <спецификации функции набора> не является внешней ссылкой.

Примечание — Понятие «внешняя ссылка» определено в 5.7 <спецификация столбца>.

4) Все <подзапрос>ы из <условия поиска> выполняются для каждой группы из *R*, а полученные результаты используются при применении к данной группе *R* <условия поиска>. Если какой-либо из выполненных <подзапрос>ов содержит внешнюю ссылку на столбец *R*, то это будет ссылка на значения данного столбца в данной группе *R*.

Примечание — Понятие «внешняя ссылка» определено в 5.7 <спецификация столбца>.

5.24 **<подзапрос>**

Функция

Задает мультинабор значений, полученных как результат <табличного выражения>

Формат

<подзапрос> : : =
(SELECT [ALL | DISTINCT] <спецификация результата>
<табличное выражение>)

<спецификация результата> :: =
<выражение значения>
| *

Правила синтаксиса

1) Применимые привилегии для каждого <имени таблицы>, которое содержится в <табличном выражении>, должны включать SELECT.

Примечание — Понятие «применимые <привилегии>» в отношении <имени таблицы> определено в 6.10 <определение привилегий>.

2) Возможны следующие варианты:

a) если <спецификация результата> «*» задана в <подзапросе>, который входит в любой <предикат>, кроме <предиката существования>, то результатом <выражения над таблицами> будет таблица, состоящая из одного столбца, а <спецификация результата> будет эквивалентна <выражению над значениями>, представляющему собой <спецификацию столбца>, которая указывает единственный столбец <табличного выражения>;

b) если <спецификация результата> «*» задана в <подзапросе>, который входит в <предикат существования>, то <спецификация результата> эквивалентна произвольному <выражению значения>, которое не включает в себя <спецификации функции набора> и допустимо для использования в <подзапросе>.

3) Значения <подзапроса> относятся к тому типу данных, к которому относится явным или неявным образом заданное <выражение значения>.

4) Результат <табличного выражения> обозначим буквой *R*.

5) Каждая <спецификация столбца> в <выражении значения> должна однозначным образом указывать столбец в *R*.

6) Если *R* является сгруппированным представлением, то <спецификация результата> не должна включать в себя <спецификацию функции набора>.

7) Если *R* является сгруппированной таблицей, то каждая <спецификация столбца> в <выражении значения> должна указывать группирующий столбец, в противном случае она должна задаваться посредством <спецификации функции набора>. Если *R* не является сгруппированной таблицей, а <выражение значения> включает в себя <спецификацию функции набора>, то каждая <спецификация столбца> в <выражении значения> должна быть задана посредством <спецификации функции набора>.

8) <ключевое слово> DISTINCT не должно задаваться в <подзапросе> более одного раза, за исключением случаев, когда оно задается в <подзапрос>ах, вложенных в данный <подзапрос>.

9) Если <подзапрос> задан в составе <предиката сравнения>, то <табличное выражение> не должно включать в себя ни <спецификатор группировки>, ни <спецификатор выборки групп>, кроме того оно не должно идентифицировать сгруппированное представление.

Общие правила

1) Если *R* не является сгруппированной таблицей, а <выражение значения> включает в себя <спецификацию функции набора>, то *R* представляет собой аргумент или же источник аргументов для каждой <спецификации функции набора> из <выражения значения>, а результатом <подзапроса> будет значение, заданное <выражением значения>.

2) Если *R* не является сгруппированной таблицей, а <выражение значения> не включает в себя <спецификацию функции набора>, то <выражение значения> применяется к каждой строке *R*, и в результате получается мультинабор, состоящий из *n* значений, где *n* — мощность *R*. Если <ключевое слово> DISTINCT не задано, то результатом <подзапроса> будет мультинабор. Если же <ключевое слово> DISTINCT задано, то результатом <подзапроса> будет набор значений, полученный из мультинабора путем исключения всех избыточных дубликатных значений.

3) Если *R* является сгруппированной таблицей, то <выражение значения> применяется к каждой строке *R*, в результате чего получается мультинабор, состоящий из *n* значений, где *n* — количество групп в *R*. Когда <выражение значения> применяется к данной группе из *R*, то эта группа представляет собой аргумент или источник аргументов для каждой <спецификации функции набора> из <выражения значения>. Если <ключевое слово> DISTINCT не задано, то результатом <подзапроса> будет мультинабор. Если же <ключевое слово> DISTINCT задано, то результатом <подзапроса> будет набор значений, полученный из мультинабора путем исключения всех избыточных дубликатных значений.

5.25 <спецификация запроса>

Функция

Задает таблицу, полученную из результата <табличного выражения>.

Формат

<спецификация запроса> : : =
SELECT [ALL|DISTINCT] <список выборки>
<табличное выражение>

<список выборки> : : =
<выражение значения> [{, <выражение значения>}...]
| *

Правила синтаксиса

1) Применимые привилегии для каждого <имени таблицы>, которое содержится в <табличном выражении>, должны включать SELECT.

Примечание — Понятие «применимые привилегии» в отношении <имени таблицы> определено в 6.10 <определение привилегий>.

2) Результат <табличного выражения> обозначим буквой *R*.

3) Число столбцов в таблице, заданной с помощью <спецификации запроса>, равно мощности <списка выборки>.

4) Если задан <список выборки> « *», то это эквивалентно последовательности <выражений значения>, где каждое <выражение значения> представляет собой <спецификацию столбца>, которая указывает столбец в *R*, причем каждый столбец в *R* может быть указан только один раз. Столбцы должны быть указаны в порядке возрастания их порядковых номеров в *R*.

5) Каждая <спецификация столбца> в каждом <выражении значения> должна однозначным образом указывать столбец из *R*. <ключевое слово> DISTINCT не должно задаваться в <спецификации запроса> более одного раза, за исключением случаев, когда оно задается в <подзапрос>ах, входящих в эту <спецификацию запроса>.

6) Если *R* является сгруппированным представлением, то <список выборки> не должен включать в себя <спецификацию функции набора>.

7) Если *R* является сгруппированной таблицей, то каждая <спецификация столбца> в <выражении значения> должна указывать группирующий столбец, в противном случае она должна задаваться посредством <спецификации функции набора>. Если *R* не является сгруппированной таблицей, а каждое <выражение значения> включает в себя <спецификацию функции набора>, то каждая <спецификация столбца> в каждом <выражении значения> должна быть задана посредством <спецификации функции набора>.

8) Для каждого столбца таблицы, которая является результатом <спецификации запроса>, тип данных, длина. точность и

дробная часть те же, что и для <выражения значения>, посредством которого был получен столбец.

9) Если i-е <выражение значения> из <списка выборки> включает в себя только <спецификацию столбца>, то i-й столбец результата будет именованным столбцом, <имя столбца> которого взято из <спецификатора столбца>. В противном случае i-й столбец будет неименованным столбцом.

10) На столбец результирующей таблицы <спецификации запроса> накладывается требование содержать только определенные значения тогда и только тогда, когда данный столбец является именованным столбцом, на который наложено требование содержать только определенные значения.

11) Обновление <спецификации запроса> допускается тогда и только тогда, когда соблюдаются следующие условия:

a) <ключевое слово> DISTINCT не задано:

b) каждое <выражение значения> из <списка выборки> включает в себя только <спецификацию столбца>, и ни одна <спецификация столбца> не встречается более одного раза;

c) <спецификатор отображения> из <табличного выражения> задает только одну <ссылку на таблицу>, причем эта <ссылка на таблицу> указывает на базовую или же на производную таблицу, для которых допустимо обновление;

d) <спецификатор выборки> из <табличного выражения> не включает в себя <подзапрос>;

e) <табличное выражение> не включает в себя ни <спецификатор группировки>, ни <спецификатор выборки групп>.

Общие правила

1) Если R не является сгруппированной таблицей, а <список выборки> включает в себя <спецификатор функции набора>, то R представляет собой аргумент или же источник аргументов для каждой <спецификации функции набора> из <спецификации запроса>, а результатом <спецификации запроса> будет таблица, состоящая из одной строки. i-м значением в строке будет значение, заданное i-м <выражением значения>.

2) Если R не является сгруппированной таблицей, а <список выборки> не включает в себя <спецификацию функции набора>, то каждое <выражение значения> применяется к каждой строке R, и в результате получается таблица, состоящая из m строк, где m -- мощность R. i-й столбец таблицы будет состоять из зна-

чений, полученных посредством применения i-го <выражения значения>.

Если <ключевое слово> DISTINCT не задано, то результатом <спецификации запроса> будет такая таблица. Если же <ключевое слово> DISTINCT задано, то результатом <спецификации запроса> будет таблица, полученная из описанной путем исключения всех избыточных дубликатных строк.

3) Если R является сгруппированной таблицей, количество групп в которой равно нулю, то результатом <спецификации запроса> будет пустая таблица.

4) Если R является сгруппированной таблицей, которая состоит из одной или более групп, то каждое <выражение значения> применяется к каждой группе из R, в результате чего получается таблица, состоящая из m строк, где m — количество групп в R. i-й столбец таблицы будет состоять из значений, полученных посредством применения i-го <выражения значения>. Когда <выражение значения> применяется к данной группе из R, то эта группа становится аргументом или же источником аргументов для каждой <спецификации функции набора> из <выражения значения>. Если <ключевое слово> DISTINCT не задано, то результатом <спецификации запроса> будет такая таблица. Если же <ключевое слово> DISTINCT задано, то результатом <спецификации запроса> будет таблица, полученная из описанной путем исключения всех избыточных дубликатных строк.

5) Строка считается дубликатной по отношению к другой строке тогда и только тогда, когда значения в каждой паре элементов с одним и тем же порядковым номером будут идентичными.

6 ЯЗЫК ОПРЕДЕЛЕНИЯ СХЕМЫ

6.1 <схема>

Функция

Определяет <схему>

Формат

```
<схема> :: =
        CREATE SCHEMA <спецификатор полномочий
        схемы>
        [<элемент схемы> ...]
<спецификатор полномочий схемы> :: =
        AUTORIZATION <идентификатор полномочий
        схемы>
```

<идентификатор полномочий схемы> ::=
<идентификатор полномочий>

<элемент схемы> ::=
<определение таблицы>
|<определение представления>
|<определение привилегий>

Правила синтаксиса

1) В одной и той же среде <идентификатор полномочий схемы> одной <схемы> должен отличаться от <идентификатора полномочий схемы> любой другой <схемы>. Понятие среды определяется разработчиком реализации.

Общие правила

Не имеется.

6.2 **<определение таблицы>**

Функция

Определяет базовую таблицу.

Формат

<определение таблицы> ::=
CREATE TABLE <имя таблицы>
(<элемент таблицы>[{, <элемент таблицы>}...])

<элемент таблицы> ::=
<определение столбца>
|<определение ограничений для таблиц>

Правила синтаксиса

1) Если <имя таблицы> включает в себя <идентификатор полномочий>, то этот <идентификатор полномочий> должен совпадать с <идентификатором полномочий схемы> той <схемы>, в которой определена таблица.

2) В пределах <схемы> <имя таблицы> не должно совпадать с <именем таблицы> из любого другого <определения таблицы> или <определения представления>.

3) <определение таблицы> должно включать в себя по крайней мере одно <определение столбца>.

4) Контекстом <имени таблицы> является <определение таблицы>.

5) Описание таблицы, заданное посредством <определения таблицы>, включает в себя <имя таблицы> и описания столбцов, каждое из которых задается посредством <определения столбца>. *i*-е <определение столбца> представляет собой описание *i*-го столбца.

Общие правила

1) <определение таблицы> определяет базовую таблицу.

6.3 **<определение столбца>**

Функция

Определяет столбец таблицы.

Формат

<определение столбца> : : =
<имя столбца> <тип данных>
[<спецификатор умолчания>]
[<ограничение для столбца>]

<ограничение для столбца> : : =
NOT NULL [<спецификация уникальности>]
|<спецификация ссылок>
|CHECK (<условие поиска>)

Правила синтаксиса

1) В пределах <определения таблицы> <имя столбца> не должно совпадать с <именем столбца> из любого другого <определения столбца>.

2) *i*-й столбец таблицы описывается *i*-м <определением столбца> в <определении таблицы>. Имя столбца и тип его данных задаются, соответственно, <именем столбца> и <типом данных>.

3) <имя столбца> в <определении столбца> обозначим буквой *C*.

4) Если задано NOT NULL, то таким образом будет неявно задано следующее <определение проверки ограничений>:

CHECK (C IS NOT NULL)

5) Если NOT NULL не задано, а также не задан <спецификатор умолчания>, то неявно заданным <спецификатором умолчания> будет DEFAULT NULL.

6) Если задана <спецификация уникальности>, то таким образом будет неявно задано следующее <определение ограничения уникальности>:

<спецификация уникальности> (*C*)

Примечание — Определение <спецификации уникальности> дано в 6.6 <определение ограничения уникальности>.

7) Если задана <спецификация ссылок>, то таким образом будет неявно задано следующее <определение ограничения на ссылки>:

FOREIGN KEY (C) <спецификация ссылок>

Примечание — Определение <спецификации ссылок> дано в 6.7 <определение ограничения на ссылки>.

8) Если задано CHECK, то каждая <спецификация столбца> в <условии поиска> должна указывать столбец *C*, и таким образом будет неявно задано следующее <определение проверки ограничения>:

CHECK (<условие поиска>)

9) Описание столбца, определяемое <определением столбца>, включает в себя имя, которое задается <именем столбца>, и тип данных, который задается <типом данных>.

Общие правила

Не имеется.

6.4 **<спецификатор умолчания>**

Функция

Устанавливает умолчания, принятые для <определения столбца>.

Формат

<спецификатор умолчания> : : =
DEFAULT {<литерал> | USER | NULL}

Правила синтаксиса

1) Обусловленным <типом данных> для <спецификатора умолчания> является <тип данных> того <определения столбца>, в котором содержится этот <спецификатор умолчания>.

2) Возможны следующие варианты:

a) если задан <литерал>:

— если обусловленный <тип данных> является строковым типом, то <литерал> должен быть <строковым литералом>. Длина <строкового литерала> не должна превышать <длину> обусловленного <типа данных>;

— если обусловленный <тип данных> является точным числовым типом, то <литерал> должен быть <точным числовым литералом>, при этом обусловленный <тип данных> должен давать возможность представить значение, заданное <точным числовым литералом>, без потери значащих разрядов;

— если обусловленный <тип данных> является приближенным числовым типом, то <литерал> должен быть <приближенным числовым литералом> или же <точным числовым литералом>;

b) если задано USER, то обусловленный <тип данных> является строковым типом, <длина> обусловленного

<типа данных> должна быть больше или равна 18;

c) если задано NULL, то <определение столбца>, в котором содержится <спецификатор умолчания>, не должно включать в себя NOT NULL.

Общие правила

1) Когда в таблицу, заданную <определением таблицы>, производится вставка строки, то инициализация столбца, заданного <определением столбца>, будет выполняться следующим образом:

a) если в <определении столбца> <спецификатор умолчания> не задан или же явным или неявным образом задан <спецификатор умолчания> вида NULL, то столбец инициализируется как содержащий неопределенное значение;

b) если в <определении столбца> задан <спецификатор умолчания>, который содержит <литерал>, то возможны следующие варианты:

— если в <определении столбца> <тип данных> задает точный числовой или же приближенный числовой тип, то столбец инициализируется как содержащий числовое значение <литерала>;

— если в <определении столбца> <тип данных> задает строку символов, длина которой равна длине <литерала>, то столбец инициализируется как содержащий значение <литерала>;

— если в <определении столбца> <тип данных> задает строку символов, длина которой превышает длину <литерала>, то столбец инициализируется как содержащий значение <литерала>, дополненное справа необходимым количеством пробелов до той длины, которая задается <типом данных>;

c) если в <определении столбца> задан <спецификатор умолчания> вида USER, то столбец инициализируется как содержащий значение, задаваемое <спецификацией значения> USER, дополненное справа необходимым количеством пробелов до той длины, которая соответствует <типу данных> из <определения столбца>.

6.5 <определение ограничений для таблиц>

Функция

Задает ограничение целостности.

Формат

<определение ограничений для таблиц> :: =
<определение ограничения уникальности>
| <определение ограничения на ссылки>
| <определение проверки ограничений>

Правила синтаксиса

Не имеется.

Общие правила

1) После выполнения каждого <SQL-оператора> контролируется соблюдение требований, заданных <определением ограничений для таблиц>.

6.6 **<определение ограничения уникальности>**

Функция

Задает ограничение уникальности для таблицы.

Формат

<определение ограничения уникальности> :: =
<спецификация уникальности> (<список уникальных столбцов>)
<спецификация уникальности> :: =
UNIQUE | PRIMARY KEY
<список уникальных столбцов> :: =
<имя столбца> [{, <имя столбца>}...]

Правила синтаксиса

1) Таблицу, для которой задаются ограничения, обозначим буквой *T*.

2) Каждое <имя столбца> из <списка уникальных столбцов> должно идентифицировать столбец в *T*, причем один и тот же столбец не должен идентифицироваться более одного раза.

3) В <определении столбца> для каждого <имени столбца> из <списка уникальных столбцов> должно быть задано NOT NULL.

4) В <определении таблицы> не должно задаваться как в явной, так и в неявной форме более одного <определения ограничения уникальности>, содержащего спецификацию PRIMARY KEY.

Общие правила

1) Столбец, имя которого идентифицировано посредством <имени столбца>, внесенного в <список уникальных столбцов>, будем называть «обозначенным столбцом».

2) Ограничение, которое накладывается на *T*, состоит в том, что *T* не должна содержать дубликатных строк, причем дубликатность строк определяется только по тем элементам, которые относятся к обозначенным столбцам. Две строки считаются дубликатными в том случае, когда значение элемента каждого обозначенного столбца в одной строке равно значению соответствующего элемента во второй строке. После выполнения каждого SQL-оператора контролируется соблюдение данного требования.

6.7 <определение ограничения на ссылки>

Функция

Задает ограничения на ссылки.

Формат

```
<определение ограничения на ссылки> ::=
        FOREIGN KEY (<обращающиеся столбцы>)
        <спецификация ссылок>
<спецификация ссылок> ::=
    REFERENCES <таблица и столбцы, к которым
    обращаются>
<обращающиеся столбцы> ::=
        <перечень столбцов ссылок>
<таблица и столбцы, к которым обращаются> ::=
        <имя таблицы> [(<перечень столбцов ссылок>)]
<перечень столбцов ссылок> ::=
        <имя столбца> [{, <имя столбца>}...]
```

Правила синтаксиса

1) Пусть «обращающаяся таблица» означает содержательную таблицу. Пусть «таблица, к которой обрашаются» означает таблицу, идентифицированную <именем таблицы> в <таблице и столбцах, к которым обращаются>. Пусть «обращающиеся столбцы» означают столбцы, идентифицированные <перечнем столбцов ссылок> в <обращающихся столбцах>.

2) Возможны следующие варианты:

   а) Если <таблица и столбцы, к которым обращаются> задает <имя столбца> или <перечень столбцов ссылок>, то это <имя столбца> или <перечень столбцов ссылок> должны быть идентичными <списку уникальных столбцов> в <определении ограничения уникальности> таблицы, к которой обращаются. Пусть «столбцы, к которым обращаются», обозначают столбцы, идентифицированные этим <именем столбца> или <перечнем столбца ссылок>.

b) Если <таблица и столбец, к которым обращаются> не задают <имя столбца> или <перечень столбцов ссылок>, то <определение таблицы> для таблицы, к которой обращаются, будет содержать <определение ограничения уникальности>, которое задает PRIMARY KEY. Пусть «столбцы, к которым обращаются» обозначают столбцы, идентифицированные <перечнем уникальных столбцов> в этом <определении ограничения уникальности>.

3) Применимые <привилегии> для <имени таблицы> должны включать REFERENCES для каждого столбца, к которому обращаются.

Примечание — «Применимые <привилегии>» для <имени таблицы> определены в 6.10 «<определение привилегий>».

4) Таблица, к которой обращаются, должна быть базовой.

5) Каждый обращающийся столбец должен идентифицировать столбец обращающейся таблицы, и одно и то же имя столбца не должно идентифицироваться более одного раза. Каждый столбец, к которому обращаются, должен идентифицировать столбец таблицы, к которой обращаются, и одно и то же имя столбца не должно идентифицироваться более одного раза.

6) <Обращающиеся столбцы> должны содержать то же самое количество имен столбцов, что и <таблица и столбцы, к которым обращаются>. i-й столбец, идентифицированный в <обращающихся столбцах>, соответствует i-му столбцу, идентифицированному в <таблице и столбцах, к которым обращаются>. Тип данных каждого обращающегося столбца должен быть таким же, как и тип данных соответствующего столбца, к которому обращаются.

Общие правила

1) Обращающаяся таблица и таблица, к которой обращаются, удовлетворяют <определению ограничения на ссылки>, если и только если для каждой строки в обращающейся таблице либо:

a) все обращающиеся столбцы в строке содержат ненулевые значения, и в таблице, к которой обращаются, существует строка такая, что для каждого обращающегося столбца значение обращающегося столбца равно значению соответствующего столбца, к которому обращаются; или

b) какой-то обращающийся столбец в строке содержит нулевое значение.

6.8 **<определение ограничения проверки>**

Функция

Задает условия для таблицы.

Формат

<определение ограничения проверки> : : =
CHECK (<условие поиска>)

Правила синтаксиса

1) <Условие поиска> не должно содержать <подзапрос>, <спецификацию функций набора> или <спецификацию цели>.

2) Каждая <спецификация столбца> в <условии поиска> должна ссылаться на столбец, определенный в содержащем <определении таблицы>.

Общие правила

1) База данных не удовлетворяет <определению ограничения проверки> тогда и только тогда, когда таблица, определенная содержащим <определением таблицы>, содержит строку, для которой <условие поиска> ложно.

6.9 **<определение представления>**

Функция

Определение представляемой таблицы.

Формат

<определение представления> : : =
CREATE VIEW <имя таблицы>
[(<перечень столбцов
представления>)]
AS <спецификации запроса>
[WITH CHECK OPTION]

<перечень столбцов представления> : : =
<имя столбца> [{, <имя столбца> . . .}]

Правила синтаксиса

1) Если <имя таблицы> содержит <идентификатор полномочий>, то этот <идентификатор полномочий> должен быть таким же, как и <идентификатор полномочий схемы> в содержащей <схеме>.

2) <Имя таблицы> должно отличаться от <имени таблицы> любого другого <определения представления> или <определения таблицы> в содержащей <схеме>.

3) Если <спецификация запроса> является корректируемой, то представляемая таблица является корректируемой таблицей. В противном случае это только считываемая таблица.

4) Если любые два столбца в таблице, заданной с помощью <спецификации запроса>, имеют одинаковое <имя столбца>,

или если любой столбец этой таблицы является непоименованным столбцом, то необходимо задать <перечень столбцов представления>.

5) Одно и то же <имя столбца> не должно задаваться более чем один раз в <перечне столбцов представления>.

6) Количество <имен столбцов> в <перечне столбцов представления> должно быть таким же, как и порядок таблицы, заданной с помощью <спецификации запроса>.

7) Описание таблицы, определенной <определением представления>, включает в себя имя <имя таблицы> и описания столбцов в таблице, заданной <спецификацией запроса>. Если задан <перечень столбцов представления>, то имя i-го столбца является i-м <именем столбца> в этом <перечне столбцов представления>.

8) Если <спецификация запроса> содержит <спецификатор группировки> или <спецификатор выборки групп>, который не содержится в <подзапросе>, то представляемая таблица, определенная <определением представления>, является группированным представлением.

9) Если задано WITH CHECK OPTION, то представляемая таблица должна быть корректируемой.

Общие правила

1) <Определение представления> определяет представляемую таблицу. Представляемая таблица V является таблицей, которая получится при выполнении <спецификации запроса>. Материализуется ли представляемая таблица, определяется разработчиком.

2) Если V корректируема, то пусть T обозначает таблицу, идентифицированную <именем таблицы>, заданным <спецификатором отображения> в <спецификации запроса>. Для каждой строки в V существует соответствующая строка в T, из которой получается эта строка V. Для каждого столбца в V существует соответствующий столбец в T, из которого можно вывести столбец таблицы V. Вставка строки в V является вставкой соответствующей строки в T. Удаление строки из V является удалением соответствующей строки из T. Корректирование столбца строки в V является корректированием соответствующей строки в T.

3) Возможны следующие варианты:

а) Если задано WITH CHECK OPTION, и <спецификация запроса> задает <спецификатор выборки>, то <оператор вставки>, <оператор корректировки: по положению> или <оператор корректировки: поиск> для данного представления не должны приводить к образова-

нию строки, для которой этот <спецификатор выборки> является ложным.

b) Если WITH CHECK OPTION не задано, то <определение представления> не должно ограничивать данные, которые могут быть вставлены в корректируемую представляемую таблицу.

Примечание — Общие правила 2 в 8.7, «<оператор вставки>», 5 в 8.11, «<оператор корректировки: по положению>» и 4 в 8.12, «<оператор корректировки: поиск>».

6.10 <определение привилегий>

Функция

Определяет привилегии.

Формат

```
<определение привилегий> ::=
    GRANT <привилегии> ON <имя таблицы>
TO <получатель привилегий> [{, <получатель
привилегий>}...]
    {WITH GRANT OPTION]
<привилегии> ::=
                ALL PRIVILEGES
              |<операция> [{, <операция>}...]
<операция> ::=
     SELECT|INSERT|DELETE
    |UPDATE [(<перечень столбцов предоставления привиле-
    гий>)]
    |REFERENCES [(<перечень столбцов предоставления при-
    вилегий>)]
<перечень столбцов предоставления привилегий> ::=
                    <имя столбца> [{, <имя столбца>}...]
<получатель привилегий> ::=
          PUBLIC | <идентификатор полномочий>
```

Правила синтаксиса

1) Пусть *T* обозначает таблицу, идентифицированную <именем таблицы>. <Привилегии> задают одну или более привилегий для *T*.

2) UPDATE (<перечень столбцов предоставления привилегий>) задает привилегию UPDATE для каждого столбца таблицы *T*, заданного в <перечне столбцов предоставления привилегий>. Каждое <имя столбца> в <перечне столбцов предоставления привилегий>должно идентифицировать столбец в *T*. Если <перечень столбцов предоставления привилегий> опущен, то UPDATE задает привилегии UPDATE на всех столбцах *T*.

3) REFERENCES (<перечень столбцов предоставления привилегий>) определяют привилегию REFERENCES для каждого столбца таблицы *T*, идентифицированного в <перечне столбцов предоставления привилегий>. Каждое <имя столбца> в <перечне столбцов предоставления> должно идентифицировать столбец в *T*. Если <перечень столбцов предоставления привилегий> опущен, то REFERENCES задает привилегию REFERENCES для всех столбцов таблицы *T*.

4) Применимые <привилегии> для ссылки на <имя таблицы> определяются следующим образом:

a) Возможны следующие варианты:

— Если существование <имени таблицы> содержится в <схеме>, то пусть применимый <идентификатор полномочий> будет <идентификатором полномочий>, заданным в качестве <идентификатора полномочий схемы> в <схеме>.

— Если существование <имени таблицы> содержится в <модуле>, то пусть применимый <идентификатор полномочий> будет <идентификатором полномочий>, заданным в виде <модульного идентификатора полномочий> в <модуле>.

b) В случае:

— Если применимый <идентификатор полномочий> тот же, что и <идентификатор полномочий>, явно или неявно заданный в <имени таблицы>, то:
Возможны следующие варианты:

1. Если *T* — базовая таблица, то применимыми <привилегиями> являются INSERT, SELECT, UPDATE, DELETE, и REFERENCES, и эти <привилегии> предоставляются.

2. Если *T* — представляемая таблица, которая не является корректируемой, то применимыми <привилегиями> есть SELECT, и эта привилегия может быть предоставлена, если и только если применимые привилегии SELECT по всем <именам таблиц>, содержащихся в <спецификации запроса>, являются предоставимыми.

3. Если *T* является представляемой таблицей, которая может корректироваться, то применимые <привилегии> на *T* являются применимыми <привилегиями>, за исключением REFERENCES на <имя таблицы>

T2, заданное в <спецификаторе отображения> <спецификации запроса>. Привилегия может быть предоставлена *T* тогда и только тогда, когда она может быть предоставлена *T2*.

— Если применимый <идентификатор полномочий> не такой же, как <идентификатор полномочий>, явно или неявно заданный в <имени таблицы>, то применимое <определение привилегий> состоит из всех <определений привилегий>, чье <имя таблицы> такое же, как и данное <имя таблицы>, и чьи <получатели привилегий> либо содержат применимый <идентификатор полномочий>, либо содержат PUBLIC, и применимые <привилегии> состоят из всех <привилегий>, заданных в применимых <определениях привилегий>. Привилегия предоставима тогда и только тогда, когда она задана в <привилегиях> некоторого применимого <определения привилегий>, которое задает WITH GRANT OPTION и применимый <идентификатор полномочий>.

5) ALL эквивалентно перечню <операций>, который включает в себя все применимые <привилегии> для <имени таблицы>.

6) Применимые <привилегии> для <имени таблицы> <определения привилегий> должны включать в себя <привилегии>, заданные в <определении привилегий>.

Общие правила

Нет.

7 МОДУЛЬНЫЙ ЯЗЫК

7.1 <модуль>

Функция

Определяет модуль.

Формат

```
<модуль> ::=
            <спецификатор имени модуля>
            <спецификатор языка>
            <спецификатор полномочий модуля>
           [<объявить курсор>...]
            <процедура>...
<спецификатор языка> ::=
            LANGUAGE {COBOL | FORTRAN | PASCAL | PL1}
<спецификатор> ::=
            AUTHORIZATION <идентификатор полномочий
            модуля>
```

<идентификатор полномочий модуля> :: =
<идентификатор полномочий>

Правила синтаксиса

1) Для каждого <объявления курсора> в <модуле> должна быть точно одна <процедура> в этом <модуле>, которая содержит <оператор открытия>, который задает <имя курсора>, объявленное в <объявлении курсора>.

2) <Модуль> должен быть связан с прикладной программой во время его выполнения. Прикладная программа должна быть связана не более чем с одним <модулем>.

Общие правила

1) Если <спецификатор языка> в <модуле> задает COBOL (соответственно FORTRAN, PASCAL, PL1) и если фактор, выполняющий вызов <процедуры> в этом <модуле>, не является стандартной программой COBOL (соответственно стандартным FORTRAN, PASCAL, PL1), то результаты оказываются неопределенными.

2) После последнего раза, когда фактор языка программирования осуществляет вызов <процедуры> в <модуле>, неявно выполняется <оператор блокировки> или <оператор возврата>. Выбор того, какой из этих <операторов SQL> выполнять, определяется разработчиком. Если произошла неустранимая ошибка, то DBMS должен выполнять <оператор возврата>.

7.2 <спецификатор имени модуля>

Функция

Именование <модуля>.

Формат

<спецификатор имени модуля> :: =
MODULE [<имя модуля>]

Правила синтаксиса

1) <Имя модуля> должно отличаться от <имени модуля> любого другого <модуля> в той же среде. Понятие среды определяет разработчик.

Общие правила

1) <Спецификатор имени модуля> определяет произвольный <идентификатор>, который будет <именем модуля>, обозначающим содержащий <модуль> в данной среде.

7.3 <процедура>

Функция

Определяет процедуру и оператор SQL.

Формат

<процедура> :: =
PROCEDURE <имя процедуры> <обявление параметра> ...;
<оператор SQL>;
<объявление параметра> :: =
<имя параметра> <тип данных>
| <параметр SQLCODE>
<параметр SQLCODE> :: =
SQLCODE
<оператор SQL> :: =
<оператор закрытия>
<оператор блокировки>
<оператор удаления: по положению>
<оператор удаления: поиск>
<оператор выборки>
<оператор вставки>
<оператор открытия>
<оператор отката>
<оператор выбора>
<оператор коррекции: по положению>
<оператор коррекции: поиск>

Правила синтаксиса

1) <Имя процедуры> должно отличаться от <имени процедуры> любой другой <процедуры> в содержащем <модуле>.

2) <Имя параметра> каждого <объявления параметра> в <процедуре> должно отличаться от <имени параметра> любого другого <объявления параметра> в этой <процедуре>.

3) Любое <имя параметра>, содержащееся в <операторе SQL> <процедуры>, должно быть задано в <объявлении параметра> в этой <процедуре>.

4) Если <имя столбца> в <операторе SQL> идентично <имени параметра> в <объявлении параметра> <процедуры>, содержащей <оператор SQL>, то <спецификация столбца>, которая содержит <имя столбца>, должно содержать <префикс>.

5) Достоверный вызов <процедуры> должен предоставить n параметров, где n — число <объявлений параметра> в <процедуре>.

6) <Процедура> должна содержать точно один <параметр SQLCODE>. Параметр, соответствующий <параметру SQLCODE>, называется параметром SQLCODE.

7) Подлежащим <спецификатора языка> в <процедуре> является <спецификатор языка> содержащего <модуля>.

8) Возможны следующие варианты:

a) Если подлежащее <спецификатор языка> задает COBOL, то:

— Типом параметра SQLCODE должен быть использован шаблон КОБОЛа COMPUTATIONAL S9(PC), где PC — задаваемая разработчиком точность, которая больше или равна 4.

— Любой <тип данных> в <объявлении параметра> должен задавать либо CHARACTER, либо NUMERIC.

— Если *i*-е <объявление параметра> задает <тип данных>, который является CHARACTER (*L*) для некоторой <длины> *L*, то тип *i*-го параметра должен быть буквенно-цифровым КОБОЛа с длиной *L*.

— Если *i*-е <объявление параметра> задает <тип данных>, который является NUMERIC (*P*, *S*) для некоторой <точности> и <масштаба> *P* и *S*, то тип *i*-го параметра должен быть с использованием КОБОЛа DISPLAY SIGN LEADING SEPARATE с последующим PICTURE.

Возможны следующие варианты:

1. Если $S=P$, то за PICTURE с «*S*» следует «*V*», за которым следует *P* «9».

2. Если $P>S>0$, то за PICTURE с «*S*» следует $P-S$ «9», за ним «*V*», и *S* «9».

3. Если $S=0$, то за PICTURE с «*S*» следует *P* «9» и за ним следует «*V*».

b) Если подлежащее <спецификатора языка> задает FORTRAN, то:

— Тип параметра SQLCODE должен быть INTEGER ФОРТРАНа.

— Любой <тип данных> в <объявлении параметра> должен задавать либо CHARACTER, INTEGER, REAL или DOUBLE PRECISION.

— Если *i*-е <объявление параметра> задает <тип данных>, который является CHARACTER (*L*) для некоторой <длины> *L*, то типом *i*-го параметра должен быть CHARACTER ФОРТРАНа с длиной *L*.

— Если *i*-е <объявление параметра> задает <тип данных>, который является INTEGER, REAL или DOUBLE PRECISION, то типом *i*-го параметра должны быть соответственно тип ФОРТРАНа INTEGER, REAL или DOUBLE PRECISION.

c) Если подлежащее <спецификатора языка> задает PASCAL, то:

— Типом параметра SQLCODE должен быть INTEGER языка ПАСКАЛЬ.
— Любой <тип данных> в <объявлении параметра> должен задавать CHARACTER INTEGER или REAL.
— Если *i*-е <объявление параметра> задает <тип данных>, который является CHARACTER (*L*) для некоторой <длины> *L*, то типом *i*-го параметра должна быть строка ПАСКАЛя с длиной *L*.
— Если *i*-е <объявление параметра> задает <тип данных>, который является INTEGER или REAL, то типом *i*-го параметра должны быть соответственно в ПАСКАЛе INTEGER или REAL.

d) Если подлежащее <спецификатора языка> задает PL1, то:

— Тип параметра SQLCODE должен быть тип ПЛ/1 FIXED BINARY (PP), где PP — задаваемая разработчиком точность, которая превышает или равна 15.
— Любой <тип данных> в <объявлении параметра> должен задавать либо CHARACTER, DECIMAL, либо FLOAT.
— Если *i*-е <объявление параметра> задает <тип данных>, который является CHARACTER (*L*) для некоторой <длины> *L*, то типом *i*-го параметра должен быть тип ПЛ/1 CHARACTER с длиной *L*.
— Если *i*-е <объявление параметра> задает <тип данных>, являющийся DECIMAL (*P*, *S*) для некоторой <точности> и <масштаба> *P* и *S*, то типом *i*-го параметра должен быть тип ПЛ/1 FIXED REAL DECIMAL (*P*, *S*).
— Если *i*-е <объявление параметра> задает <тип данных>, являющийся FLOAT (*P*), для некоторой <точности> *P*, то типом *i*-го параметра должен быть тип ПЛ/1 FLOAT REAL BINARY (*P*).

Общие правила

1) <Процедура> обозначает процедуру, которую можно вызвать с помощью установленного разработчиком средства.

2) Когда <процедура> вызывается средством языка программирования:

a) Если нет активных транзакций для этого средства, то производится эффективная инициализация транзакции и ее связывание с этим вызовом и с последующими вызовами с помощью данного средства любых других <про-

цедур> в содержащем их <модуле> до тех пор, пока посредник не закончит эту транзакцию.

b) <оператор SQL> <процедуры> выполняется.

3) Возможны следующие варианты:

a) Если *S* успешно выполняется, то возможны следующие варианты:

— Если *S* — <оператор выборки>, для которого следующей строки не существует, то параметр SQLCODE полагают равным 100.

— Если *S* — <оператор вставки>, для которого нет кандидатуры строки, то параметр SQLCODE полагают равным 100.

— Если *S* — <оператор выборки>, результатом которого была пустая таблица, то параметр SQLCODE полагают равным 100.

— Если *S* — <оператор корректировки: поиск> или <оператор удаления: поиск>, для которых не было выходной строки для корректировки или удаления, то параметр SQLCODE полагают равным 100.

— В прочих случаях параметр SQLCODE устанавливается равным 0.

b) Если *S* не выполнено успешно, то:

— Все изменения, произведенные над базой данных при выполнении *S*, исключаются.

— Параметр SQLCODE приобретает отрицательное числовое значение по определению разработчика.

8 ЯЗЫК МАНИПУЛИРОВАНИЯ ДАННЫМИ

8.1 <оператор закрытия>

Функция

Закрытие курсора.

Формат

<оператор закрытия> : : =
CLOSE <имя курсора>

Правила синтаксиса

1) Содержащий <модуль> должен содержать <объявление курсора> CR, для которого <имя курсора> такое же, как и <имя курсора> в <операторе закрытия>.

Общие правила

1) Курсор CR должен быть в открытом состоянии.

2) Курсор CR помещается в закрытом состоянии, и копия <спецификации курсора> CR уничтожается.

8.2 **<оператор блокировки>**

Функция

Окончание текущей транзакции фиксацией изменений.

Формат

```
<оператор блокировки> : : =
        COMMIT WORK
```

Правила синтаксиса

Нет.

Общие правила

1) Текущая транзакция заканчивается.

2) Любые курсоры, которые были открыты текущей транзакцией, закрываются.

3) Любые изменения в базе данных, которые были произведены текущей транзакцией, фиксируются.

8.3 **<объявление курсора>**

Функция

Определение курсора.

Формат

```
<объявление курсора> : : =
                DECLARE <имя курсора> CURSOR
                FOR <спецификация курсора>
<спецификация курсора> : : =
<выражение запроса> [<спецификатор порядка>]
<выражение запроса> : : =
        <элемент запроса>
      | <выражение запроса> UNION [ALL] <элемент
      запроса>
<элемент запроса> : : =
        <спецификация запроса> | (<выражение запроса>)
<спецификатор порядка> : : =
        ORDER BY <спецификация упорядочения>
                        [{, <спецификация упорядочения>}...]
<спецификация упорядочения> : : =
  {<беззнаковое целое> | <спецификация столбца>}
  [ASC | DESC]
```

Правила синтаксиса

1) <Имя курсора> не должно быть идентичным <имени кур-

сора>, заданному в любом другом <объявлении курсора> в одном и том же <модуле>.

2) Любое <имя параметра>, содержащееся в <спецификации курсора>, должно быть определено в <объявлении параметра> в <процедуре> в содержащем <модуле>, который содержит <оператор открытия>, задающий <имя курсора>.

Примечание — Смотри Правила синтаксиса в 7.1, «<модуль>».

3) Пусть *T* обозначает таблицу, заданную <спецификацией курсора>.

4) Возможны следующие варианты:

a) Если задан ORDER BY, то *T* является только считываемой таблицей с заданным порядком упорядочения.

b) Если не задано ни ORDER BY, ни UNION, и <спецификация запроса> корректируема, то *T* является корректируемой таблицей.

c) В других случаях *T* является только считываемой таблицей.

5) Возможны следующие варианты:

a) Если UNION не задан, то описание *T* является описанием <спецификации запроса>.

b) Если задан UNION, то для каждого заданного UNION пусть *T1* и *T2* обозначают таблицы, заданные <выражением запроса> и <элементом запроса>. <Перечни выбора> для спецификаций *T1* и *T2* должны состоять из « * » или <спецификаций столбцов>. За исключением имен столбцов, описания *T1* и *T2* должны быть идентичными. Все столбцы результата являются непоименованными. За исключением <имен столбцов>, описание результата такое же, как и описания *T1* и *T2*.

6) Если задан ORDER BY, то каждая <спецификация упорядочения> в <спецификаторе порядка> должна идентифицировать столбец *T*. Возможны следующие варианты:

a) Если <спецификация упорядочения> содержит <спецификацию столбца>, то <спецификация упорядочения> идентифицирует столбец таблицы *T* с именем, заданным этой <спецификацией столбца>.

b) Если <спецификация упорядочения> содержит <целое без знака>, то <целое без знака> должно быть больше 0, но не больше степени *R*. <Спецификация упорядочения> идентифицирует столбец *T* с положением по порядку, заданным <целым без знака>.

Общие правила

1) Возможны следующие варианты:
   a) Если *T* является корректируемой таблицей, то курсор связан с поименованной таблицей, идентифицированной <именем таблицы> в <спецификаторе отображения> Пусть *B* обозначает эту поименованную таблицу. Для каждой строки в *T* существует соответствующая строка в *B*, из которой выводится строка в *T*. Когда курсор располагается на строке в *T*, то курсор также располагается на соответствующей строке в *B*.
   b) Во всех других случаях курсор не связан с поименованной таблицей.

2) Возможны следующие варианты:
   a) Если не задан UNION, то *T* является результатом заданной <спецификации запроса>.
   b) Если UNION задан, то для каждого UNION, который задан, пусть *T1* и *T2* будут результатом <выражения запроса> и <элемента запроса>. Результат UNION эффективно выводится следующим образом:
   — задается результат в пустую таблицу;
   — каждая строка *T1* и каждая строка *T2* вставляются в этот результат;
   — если не задано ALL, то исключаются любые избыточные дублированные строки из результата.

3) Возможны следующие варианты:
   a) Если не задан ORDER BY, то:
   — упорядочение строк в *T* задается разработчиком;
   — упорядочение строк в *T* может изменяться между транзакциями;
   — упорядочение строк в *T*, когда курсор открыт, будет таким же в следующий момент после открытия курсора в этой же транзакции, при условии, что:

     1 Не было принудительных изменений в любой таблице, опрашиваемой в <спецификации курсора>.

     2 Упорядоченный набор значений параметров, предоставленный для <процедуры>, содержащей <оператор открытия>, одинаков в обоих случаях. Если эти условия не соблюдаются, то упорядочение строк в *T* может отличаться в последовательных открытиях курсора в рамках одной и той же транзакции.

   b) Если задан ORDER BY, то *T* имеет упорядоченный порядок:
   — Упорядоченный порядок является последовательностью

упорядоченных групп. Упорядоченная группа представляет собой последовательность строк, в которых все значения упорядоченного столбца идентичны. Кроме того, упорядоченная группа может быть последовательностью упорядоченных групп.

— Количество элементов в последовательности и положение по порядку каждой упорядоченной группы определяется значениями крайнего значащего упорядоченного столбца. Количество элементов в последовательности является минимальным количеством упорядоченных групп, так что для каждой упорядоченной группы из более чем одной строки все значения этого упорядоченного столбца идентичны.

— Если упорядоченный порядок основывается на дополнительных упорядоченных столбцах, то каждая упорядоченная группа из более чем одной строки является последовательностью упорядоченных групп. Количество элементов в каждой последовательности и положение по порядку каждой упорядоченной группы в каждой последовательности определяется значениями следующего наиболее *значащего упорядоченного столбца.* Количество элементов в каждой последовательности является минимальным количеством упорядоченных групп, так что для каждой упорядоченной группы с более чем одной строкой все значения этого упорядоченного столбца идентичны.

— Предыдущий параграф касается по очереди каждого дополнительного упорядоченного столбца. Если упорядоченная группа состоит из множественных строк и не является последовательностью упорядоченных групп, то порядок строк в этой упорядоченной группе является неопределенным.

— Пусть *C* будет упорядоченным столбцом и *S* обозначает последовательность, определяемую значениями *C*.

— Упорядоченное направление связано с каждым упорядоченным столбцом. Если направление *C* восходящее, то первая упорядоченная группа *S* содержит наинизшее значение *C*, а каждая последующая упорядоченная группа содержит значение *C*, большее, чем значение *C* в предшествующей упорядоченной группе. Если направление нисходящее, то первая упорядоченная группа содержит наивысшее значение *C*, а каждая последующая упорядоченная группа содержит значение *C*, меньшее, чем значение *C* в предшествующей упорядоченной группе.

— Упорядочение определяется правилами сравнения, определенными в 5.11 «<предикат сравнения>». Порядок нулевого значения по отношению к ненулевым значениям определяется разработчиком, но он должен быть либо больше, либо меньше, чем все ненулевые значения.
— <Спецификация упорядочения> задает упорядоченный столбец и направление. Упорядоченный столбец является столбцом, который опрашивается <целым без знака> или <спецификацией столбца>. <Целое без знака> *i* опрашивает *i*-й столбец *T*. <Спецификация столбца> опрашивает поименованный столбец.
— Если DESC задано в <спецификации упорядочения>, то направление упорядоченного столбца, заданного этой <спецификацией упорядочения>, нисходящее. Если задано ASC, или не задано ни ASC, ни DESC, то направление упорядоченного столбца восходящее.
— Последовательность <спецификация упорядочения> определяет относительную значимость упорядоченных столбцов. Упорядоченный столбец, заданный первой <спецификацией упорядочения>, является наиболее значимым упорядоченным столбцом, и каждый последовательно задаваемый упорядоченный столбец является менее значимым, чем ранее заданный упорядоченный столбец.

8.4 <оператор удаления: по положению>

Функция

Удаление строки из таблицы.

Формат

<оператор удаления: по положению> :: =
DELETE FROM <имя таблицы>
WHERE CURRENT OF <имя курсора>

Правила синтаксиса

1) Применимые <привилегии> для <имени таблицы> должны включать в себя DELETE.

Примечание — «Применимые <привилегии>» для <имени таблицы> определены в 6.10, «<определение привилегий>».

2) Содержащий <модуль> должен содержать <объявление курсора> CR, чье <имя курсора> такое же, как и <имя курсора> в <операторе удаления: по положению>.

3) Таблица, обозначенная CR, не должна быть только считываемой таблицей.

4) Пусть *T* обозначает таблицу, идентифицированную <именем таблицы>. *T* должна быть таблицей, идентифицированной в первом <спецификаторе отображения> в <спецификации курсора> CR.

Общие правила

1) Курсор CR должен располагаться на строке.

2) Строка, из которой выводится текущая строка CR, удаляется.

8.5 <оператор удаления: поиск>

Функция

Удаление строки из таблицы.

Формат

<оператор удаления: поиск> : : =
DELETE FROM <имя таблицы>
WHERE <условие поиска>

Правила синтаксиса

1) Применимые <привилегии> для <имени таблицы> должны включать в себя DELETE.

Примечание — «Применимые <привилегии>» для <имени таблицы> определены в 6.10 «<определение привилегий>».

2) Пусть *T* обозначает таблицу, идентифицированную <именем таблицы>. *T* не должна быть только считываемой таблицей или таблицей, которая идентифицированна в <спецификаторе отображения> любого <подзапроса>, содержащегося в <условии поиска>.

3) Контекстом <имени таблицы> является весь <оператор удаления: поиск>.

Общие правила

1) Возможны следующие варианты:

a) Если <условие поиска> не задано, то удаляются все строки таблицы *T*.

b) Если <условие поиска> задано, то оно применяется к каждой строке *T* с <именем таблицы>, связанным с этой строкой, и все строки, для которых результатом <условия поиска> является истина, удаляются. Каждый <подзапрос> в <условиях поиска> эффективно выполняется для каждой строки таблицы *T*, а результаты используются в применении <условий поиска> к данной строке *T*. Если любой выполняемый <подзапрос> содержит внешнюю ссылку к столбцу таблицы *T*, то ссылка делается на значение этого столбца в данной строке *T*.

Примечание — «Внешняя ссылка» определена в 5.7 «<спецификация столбца>».

8.6 <оператор выборки>

Функция

Установка курсора на следующей строке таблицы и отыскание значений из этой строки.

Формат

<оператор выборки> :: =
FETCH <имя курсора> INTO <перечень цели выборки>
<перечень цели выборки> :: =
<спецификация цели> [{, <спецификация цели>}...]

Правила синтаксиса

1) Содержащий <модуль> должен содержать <объявление курсора> CR, чье <имя курсора> такое же, как и <имя курсора> в <операторе выборки>. Пусть *T* будет таблицей, определенной <спецификацией курсора> CR.

2) Число <спецификаций цели> в <перечне цели выборки> должно быть таким же, как и порядок таблицы *T*.

3) Возможны следующие варианты:

a) Если тип данных цели, заданной *i*-й <спецификацией цели> в <перечне цели выборки>, является строкой символов, то тип данных в *i*-м столбце таблицы *T* должен быть строкой символов.

b) Если тип данных цели, заданной *i*-й <спецификацией цели> в <перечне цели выборки>, является точным числом, то тип данных в *i*-м столбце таблицы *T* должен быть точным числом.

c) Если тип данных цели, заданной *i*-й <спецификацией цели> в <перечне цели выборки>, является приближенным числом, то тип данных в *i*-м столбце таблицы *T* должен быть приближенным или точным числом.

Общие правила

1) Курсор CR должен быть в открытом состоянии.

2) Если таблица, заданная курсором CR, пуста или если CR располагается на последней строке или после нее, то CR располагается после последней строки, значение 100 приписывается параметру SQLCODE, и значения не присваиваются целям, идентифицированным <перечнем цели выборки>.

3) Если CR располагается перед строкой, то CR располагается на этой строке, и значения присваиваются их соответствующим целям.

4) Если положение CR находится на *r*, где *r* — строка, отличающаяся от последней строки, то CR располагается на строке непосредственно после *r*, и значения в строке непосредственно после *r* присваиваются своим соответствующим целям.

5) Присвоение значений целям в <перечне целей выборки>, отличающимся от параметра SQLCODE, определяется разработчиком. Параметр SQLCODE получает значение последним.

6) Если происходит ошибка во время присвоения значения цели, то параметр SQLCODE получает установленное разработчиком отрицательное число, и значения целей, отличных от параметра SQLCODE, определяются разработчиком.

7) Пусть *V* будет целью, и пусть *v* обозначает соответствующее значение в текущей строке CR.

8) Если *v* — нулевое значение, то нужно задать признак для *V*, и этот признак устанавливается в —1. Если *v* — ненулевое значение, и *V* имеет признак, то возможны следующие варианты:

a) Если тип данных *V* является строкой символов длиной *L*, а длина *M* для *v* больше, чем *L*, то признак устанавливается на *M*.

b) В других случаях признак устанавливается равным 0.

c) Цель, идентифицированная *i*-й <спецификацией цели> <перечня целей выборки>, соответствует *i*-му значению текущей строки CR.

10) Возможны следующие варианты:

a) Если тип данных *V* является строкой символов, а длина *v* равна длине *V*, то значение *V* устанавливается для *v*.

b) Если тип данных в *V* является строкой символов длиной *L*, а длина *v* больше, чем *L*, то значение *V* устанавливается на первые *L* символов в *v*.

c) Если тип данных *V* является строкой символов длиной *L*, а длина *M* для *v* меньше, чем *L*, то первые *M* символов в *V* устанавливаются для *v*, а последние *L*—*M* символов *V* приобретают символ пробела.

d) Если тип данных *V* является точным числом, то должно быть представление значения *v* в тип данных *V*, при котором не теряются никакие первые значащие разряды, и значение *V* устанавливается на это представление.

e) Если тип данных в *V* является приближенным числом, то значение *V* устанавливается на приближенное значение *v*.

8.7 <оператор вставки>

Функция

Создание новых строк в таблице.

Формат

<оператор вставки> : : =
INSERT INTO <имя таблицы> [(<перечень столбцов вставки>)]
{VALUES (<перечень значений вставки>)
|<спецификация запроса>}
<перечень столбцов вставки> : : =
<имя столбца> [{, <имя столбца>}...]
<перечень значений вставки> : : =
<вставить значение> [{, <вставить значение>}...]
<вставить значение> : : =
<спецификация значения> | NULL

Правила синтаксиса

1) Применимые <привилегии> для <имени таблицы> должны включать в себя INSERT.

Примечание — «Применимые <привилегии>» для <имени таблицы> определены в 6.10 «<определение привилегий>».

2) Пусть *T* обозначает таблицу, идентифицированную <именем таблицы>. *T* не должна быть только считываемой таблицей или таблицей, которая идентифицируется в <спецификаторе отображения> <спецификации запроса> или любом <подзапросе>, содержащемся в <спецификации запроса>.

3) Каждое <имя столбца> в <перечне столбцов вставки> должно идентифицировать столбец *T*, и этот же столбец не должен идентифицироваться более одного раза. Пропуск <перечня столбцов вставки> является неявной спецификацией <перечня столбцов вставки>, которая идентифицирует все столбцы *T* в восходящей последовательности их расположения по порядку в пределах *T*.

4) Столбец, идентифицированный <перечнем столбцов вставки>, является предметным столбцом.

5) Возможны следующие варианты:

a) Если задан <перечень значений вставки>, то количество <значений вставки> должно быть равным количеству <имен столбцов> в <перечне столбцов вставки>. Пусть *i*-й пункт <оператора вставки> относится к *i*-й <спецификации значения> в этом <перечне значений вставки>.

b) Если задана <спецификация запроса>, то порядок таблицы, заданный этой <спецификацией запроса>, должен быть равен количеству <имен столбцов> в <перечне столбцов вставки>. Пусть *i*-й элемент <операто-

ра вставки> относится к *i*-му столбцу таблицы, заданной <спецификацией запроса>.

6) Если *i*-й элемент <оператора вставки> не является <значением вставки> NULL, то возможны следующие варианты:

a) Если тип данных столбца таблицы *T*, обозначенного *i*-м <именем столбца>, является строкой символов длиной *L*, то тип данных *i*-го элемента <оператора вставки> должен быть строкой символов длиной меньше или равной *L*.

b) Если тип данных столбца таблицы *T*, обозначенного *i*-м <именем столбца>, является точным числом, то тип данных *i*-го элемента <оператора вставки> должен быть точным числом.

c) Если тип данных столбца таблицы *T*, обозначенного *i*-м <именем столбца>, является приближенным числом, то тип данных *i*-го элемента <оператора вставки> должен быть приближенным или точным числом.

Общие правила

1) Строка вставляется по следующим этапам:

a) Строка-кандидат эффективно создается так, как это описано в Общих правилах в 6.4 «<спецификатор умолчания>». Если *T* является базовой таблицей *B*, то строка-кандидат включает в себя каждый столбец из *B*. Если *T* является представляемой таблицей, то строка-кандидат включает в себя каждый столбец базовой таблицы *B*, из которой выведена таблица *T*.

b) Для каждого предметного столбца в строке-кандидате значение заменяется на вставляемое значение.

c) Строка-кандидат вставляется в *B*.

2) Если *T* является представляемой таблицей, определенной <определением представления>, которое задает WITH CHECK OPTION, то если <спецификация запроса>, содержащаяся в <определении представления>, задает <спецификатор выборки>, не содержащийся в <подзапросе>, то <условие поиска> этого <спецификатора выборки> должно быть истинным для строки-кандидата.

3) Если задан <перечень значений вставки>, то возможны следующие варианты:

a) Если *i*-е <значение вставки> из <перечня значений вставки> является <спецификацией значения>, то значение столбца в строке-кандидате, соответствующее *i*-му предметному столбцу, является значением этой <спецификации значения>.

b) Если i-е <значение вставки> <перечня значений вставки> является NULL, то значение столбца строки-кандидата, соответствующего i-му предметному столбцу, является нулевым значением.

4) Если <спецификация запроса> задана, то пусть R будет результатом <спецификации запроса>. Если R пустой, то параметру SQLCODE приписывается значение 100, и никакая строка не вставляется. Число созданных строк-кандидатов равно мощности R. Вставляемые значения строки-кандидата являются значениями в одной строке R, а значения в одной строке R являются вставляемыми значениями одной строки-кандидата.

5) Пусть V обозначает строку из R или последовательность значений, заданных <перечнем значений вставки>. i-е значение V является вставляемым значением предметного столбца, идентифицированного i-м <именем столбца> в <перечне значений вставки>.

6) Пусть C обозначает предметный столбец, а v — ненулевое вставляемое значение в C.

7) Возможны следующие варианты:

a) Если тип данных C — это строка символов, а длина v равная длине C, то значение C устанавливается в v.

b) Если тип данных C — строка символов длиной L, а длина M для v меньшая, чем L, то первые M символов в C устанавливаются в v, а последние L—M символов в C приобретают вид пробелов.

c) Если типом данных C является точное число, то должно быть представление значения v в типе данных в C, которое не теряет никаких первых значащих разрядов, и значение C устанавливается по этому представлению.

d) Если тип данных C является приближенным числом, то значение C устанавливается в приближенное значение v.

8.8 <оператор открытия>

Функция

Открытие курсора.

Формат

<оператор открытия> : : =
OPEN <имя курсора>

Правила синтаксиса

1) Содержащий <модуль> должен содержать <объявление курсора> CR, чье <имя курсора> является таким же, как и

<имя курсора> в <операторе открытия>.

Общие правила

1) Курсор CR должен быть в закрытом состоянии.

2) Пусть *S* обозначает <спецификацию курсора> для курсора CR.

3) Курсор CR открывается следующими операциями:

a) Эффективно проводится копирование *S*, в котором каждая <спецификация цели> заменяется значением цели, которое его идентифицирует.

b) Если *S* задает только считываемую таблицу, то эта таблица эффективно создается, как задано копированием *S*.

c) Курсор CR помещается в открытое состояние, и его положение находится перед первой строкой таблицы.

8.9 <оператор отката>

Функция

Окончание текущей транзакции аннулированием изменений.

Формат

<оператор возврата> : : =
ROLLBACK WORK

Правила синтаксиса

Нет.

Общие правила

1) Любые изменения базы данных, сделанные текущей транзакцией, аннулируются.

2) Любые курсоры, которые были открыты текущей транзакцией, закрываются.

3) Текущая транзакция заканчивается.

8.10 <оператор выбора>

Функция

Отыскание значений из заданной строки таблицы.

Формат

<оператор выбора> : : =
SELECT [ALL | DISTINCT] <перечень выбора>
INTO <перечень цели выбора>
<выражение таблицы>

<перечень цели выбора> : : =
<спецификация цели> [{ <спецификация цели>}...]

Правила синтаксиса

1) Применимые <привилегии> для каждого <имени таблицы>, содержащегося в <выражении таблицы>, должны включать в себя SELECT.

Примечание — «Применимые <привилегии>» для <имени таблицы> определены в 6.10 «<определение привилегий>».

2) Выражение таблицы не должно включать <спецификатор группировки> или <спецификатор выборки групп> и не должно идентифицировать групповое представление.

3) Количество элементов в <перечне выборки> должно быть таким же, как и число элементов в <перечне целей выбора>.

4) Возможны следующие варианты:

a) Если тип данных цели, обозначенной *i*-й <спецификацией цели> в <перечне целей выбора>, является строкой символов, то тип данных *i*-го <выражения значения> в <перечне выборки> должен принадлежать к строке символов.

b) Если тип данных цели, заданной *i*-й <спецификацией цели> в <перечне целей выборки>, относится к точным числам, то типом данных для *i*-го <выражения значения> в <перечне выборки> должно быть точное число.

c) Если типом данных цели, заданной *i*-й <спецификацией цели> в <перечне целей выборки>, является приближенное число, то типом данных *i*-го <выражения значения> в <перечне выборки> должно быть приближенное число или точное число.

5) Пусть *S* будет <спецификацией запроса>, чьи <перечень выборки> и <выражение таблицы> заданы в <операторе выбора>, и он задает ALL или DISTINCT, если это задано в <операторе выбора>. *S* должно быть справедливой <спецификацией запроса>.

Общие правила

1) Пусть *R* обозначает результат <спецификации запроса> *S*.

2) Мощность *R* не должна быть больше единицы. Если *R* пустая, то параметр SQLCODE приобретает значение 100, и значения не присваиваются целям, идентифицированным <перечнем целей выборки>.

3) Если *R* не пустая, то значения в строке *R* присваиваются их соответствующим целям.

4) Присвоение значений целям в <перечне целей выборки>, отличных от параметра SQLCODE, определяется разработчиком.

Параметр SQLCODE приобретает последнее значение.

5) Если во время присвоения значений целям происходит ошибка, то параметр SQLCODE приобретает отрицательное число, величина которого определяется разработчиком, и значения целей, отличных от параметра SQLCODE, задаются разработчиком.

6) Цель, идентифицированная *i*-й <спецификацией цели> в <перечне целей спецификации>, соответствует *i*-му значению в строке *R*.

7) Пусть *V* будет идентифицированной целью, а *v* обозначает ее соответствующее значение в строке *R*.

8) Если *v* — нулевое значение, то для *V* следует задать признак, и установить этот признак в —1. Если *v* является ненулевым значением, а *V* имеет признак, то возможны следующие варианты:

a) Если типом данных *V* является строка символов длиной *L*, и длина *M* больше, чем *L*, то признак устанавливается в *M*.

b) В других случаях признак устанавливается на 0.

9) Возможны также следующие варианты:

a) Если типом данных *V* является строка символов, а длина *v* эквивалентна длине *V*, то значение *V* устанавливается в *v*.

b) Если типом данных *V* является строка символов длиной *L*, а длина *v* больше, чем *L*, то значение *V* устанавливается на первые *L* символов в *v*.

c) Если типом данных *V* является строка символов длиной *L*, и длина *M* в *v* меньшая, чем *L*, то первые *M* символов в *V* устанавливаются для *v*, а последние *L*—*M* символов в *V* приобретают характер пробелов.

d) Если типом данных *V* является точное число, то должно быть представление значения *v* в типе данных *V*, при котором не теряется ни один первый значащий разряд, и значение *V* устанавливается по этому представлению.

e) Если типом данных в *V* является приближенное число, то значение *V* устанавливается в приближенное значение *v*.

8.11 <оператор корректировки: по положению>

Функция

Корректировка строки таблицы.

Формат

<оператор корректировки: по положению> : : =
UPDATE <имя таблицы>

SET <спецификатор набора: по положению>
[{, <спецификатор набора: по положению>}...]
WHERE CURRENT OF <имя курсора>

<спецификатор набора: по положению> :: =
<предметный столбец: по положению> =
{<выражение значения> | NULL}

<предметный столбец: по положению> :: = <имя столбца>

Правила синтаксиса

1) Применимые <привилегии> для <имени таблицы> должны включать в себя UPDATE для каждого <предметного столбца: по положению>.

Примечание — «Применимые <привилегии>» для <имени таблицы> определены в 6.10 «<определение привилегий>».

2) Содержащий <модуль> должен содержать <объявление курсора> CR, чье <имя курсора> такое же, как и <имя курсора> в <операторе корректировки: по положению>.

3) Таблица, обозначенная CR, не должна быть только считываемой таблицей.

4) Пусть *T* обозначает таблицу, идентифицированную <именем таблицы>. *T* должна быть таблицей, идентифицированной в первом <спецификаторе отображения> в <спецификации курсора> CR.

5) <Выражение значения> в <спецификаторе набора: по положению> не должно включать в себя <спецификацию функций набора>.

6) Каждое <имя столбца>, заданное как <предметный столбец: по положению>, должно идентифицировать столбец в *T*. То же <предметный столбец: по положению> не должно появляться более чем один раз в <операторе корректировки: по положению>.

7) Контекстом <имени таблицы> является весь <оператор коррекции: по положению>.

8) Для каждого <спецификатора набора: по положению> возможны следующие варианты:

a) Если задано NULL, то столбец, обозначенный как <предметный столбец: по положению>, должен допускать нули.

b) Если типом данных в столбце, обозначенном как <предметный столбец: по положению>, является строка символов длиной *L*, то типом данных <выражения значения> должна быть строка символов длиной менее или равной длине *L*.

c) Если типом данных столбца, обозначенного как <предметный столбец: по положению>, является точное число, то типом данных <выражения значения> должно быть точное число.

d) Если типом данных столбца, обозначенного как <предметный столбец: по положению>, является приближенное число, то типом данных в <выражении значения> должно быть приближенное или точное число.

Общие правила

1) Курсор CR должен быть установлен на строку.

2) Предметная строка — это та строка, из которой выводится текущая строка CR.

3) Предметная строка корректируется так, как это задается каждым <спецификатором набора: по положению>. <Спецификатор набора: по положению> задает предметный столбец и значение корректировки в этом столбце. Предметный столбец — это столбец, идентифицированный как <предметный столбец: по положению> в <спецификаторе набора: по положению>. Значение для корректировки — это нулевое значение или значение, заданное <выражением значения>. Если <выражение значения> содержит ссылку на столбец таблицы *T*, то ссылка эта делается на значение этого столбца в предметной строке перед тем, как корректируется любое значение предметной строки.

4) Предметная строка корректируется следующим образом:

a) Создается строка-кандидат, которая является копией предметной строки.

b) Для каждого <спецификатора набора: по положению> значение заданного предметного столбца в строке-кандидате заменяется заданным скорректированным значением.

c) Предметная строка заменяется строкой-кандидатом.

5) Если *T* — представляемая таблица, определяемая <определением представления>, которое задает «WITH CHECK OPTION», то если <спецификация запроса>, содержащаяся в <определении представления>, задает <спецификатор выборки>, не содержащийся в <подзапросе>, то <условие поиска> этого <спецификатора выборки> должно быть истинным для строки-кандидата.

6) Пусть *C* обозначает предметный столбец, а *v* — ненулевое корректируемое значение в *C*.

7) Возможны следующие варианты:

a) Если типом данных *C* является строка символов, а длина равна длине *C*, то значение *C* устанавливается в *v*.

b) Если типом данных *C* является строка символов длиной *L*, и длина *M* в *v* меньше, чем длина *L*, то первые *M* символов в *C* устанавливаются в *v*, а последние символы *L*—*M* в *C* приобретают смысл пробелов.

c) Если типом данных в *C* является точное число, то должно быть представление значения *v* в типе данных *C*, которое не теряет ни одного первого значащего разряда, а значение *C* устанавливается в это представление.

d) Если типом данных в *C* является приближенное число, то значение *C* устанавливается в приближенное значение *v*.

8.12 **<оператор корректировки: поиск>**

Функция

Корректировка строк в таблице.

Формат

```
<оператор корректировки: поиск> : : =
    UPDATE <имя таблицы>
    SET <спецификатор набора: поиск>
        [{, <спецификатор набора: поиск>}...]
    [WHERE <условие поиска>]
<спецификатор набора: поиск> : : =
    <предметный столбец: поиск> =
                        {<выражение значения> | NULL}
<предметный столбец: поиск> : : = <имя столбца>
```

Правила синтаксиса

1) Применимые <привилегии> для <имени таблицы> должны включать в себя UPDATE для каждого <предметного столбца: поиск>.

Примечание — «Применимые <привилегии>» для <имени таблицы> определены в 6.10 «<определение привилегий>».

2) Пусть *T* обозначает таблицу, идентифицированную <именем таблицы>. *T* не должна быть только считываемой таблицей или таблицей, которая идентифицируется в спецификаторе отображения любого <подзапроса>, который содержится в <условии поиска>.

3) <Выражение значения> в <спецификаторе набора: поиск> не должно включать в себя <спецификацию набора функций>.

4) Каждое <имя столбца>, заданное как <предметный столбец: поиск>, должно идентифицировать столбец таблицы *T*. Тот же <предметный столбец: поиск> не должен появиться более чем один раз в <операторе корректировки: поиск>.

5) Контекстом <имени таблицы> является весь <оператор корректировки: поиск>.

6) Для каждого <спецификатора набора: поиск> возможны следующие варианты:

a) Если задан NULL, то столбец, обозначенный как <предметный столбец: поиск>, должен разрешать нулевые значения.

b) Если типом данных в столбце, обозначенном как <предметный столбец: поиск>, является строка символов длиной L, то типом данных <выражения значения> должна быть строка символов длиной меньше или равной L.

c) Если типом данных в столбце, обозначенным как <предметный столбец: поиск>, является приближенное число, то типом данных в <выражении значения> должно быть приближенное или точное число.

Общие правила

1) Возможны следующие варианты:

a) Если <условие поиска> не задано, то все строки таблицы T являются предметными строками.

b) Если <условие поиска> задано, то оно применяется для каждой строки в таблице T с <именем таблицы>, связанным с этой строкой, и предметными строками являются те строки, для которых результат <условия поиска> — истина. Каждый <подзапрос> в <условии поиска> эффективно выполняется для каждой строки в таблице T, а результаты используются применительно к <условию поиска> к данной строке таблицы T. Если какой-либо <подзапрос> содержит внешнюю ссылку на столбец таблицы T, то ссылка делается на значение этого столбца в данной строке таблицы T.

Примечание — «Внешняя ссылка» определена в 5.7 «<спецификация столбца>».

2) Каждая предметная строка корректируется в соответствии с каждым <спецификатором набора: поиск>. <Спецификатор набора: поиск> задает предметный столбец и корректируемое значение этого столбца. Предметным столбцом является столбец, определяемый <предметным столбцом: поиск>. Корректируемое значение является нулевым значением или значением, задаваемым <выражением значения>. Если <выражение значения> содержит ссылку на столбец таблицы T, то ссылка делается на значение этого столбца в предметной строке перед тем, как корректировать любое значение предметной строки.

3) Предметная строка корректируется следующим образом:
   a) Создается строка-кандидат, которая является копией предметной строки.
   b) Для каждого <спецификатора набора: поиск> значение заданного предметного столбца в строке-кандидате заменяется заданным корректируемым значением.
   c) Предметная строка заменяется строкой-кандидатом.

4) Если *T* является представляемой таблицей, определяемой <определением представления>, которое задает WITH CHECK OPTION, то если <спецификация запроса>, содержащаяся в <определении представления>, задает <спецификатор выборки>, не содержащийся в <подзапросе>, то <условие поиска> этого <спецификатора выборки> будет истиной для строки-кандидата.

5) Пусть *C* обозначает предметный столбец, а *v* — ненулевое корректируемое значение в *C*.

6) Возможны следующие варианты:
   a) Если типом данных *C* является строка символов, а длина *v* равна длине *C*, то значение *C* устанавливается в *v*.
   b) Если типом данных *C* является строка символов длиной *L*, а длина *M* в *v* меньше, чем *L*, то первые *M* символов в *C* устанавливаются в *v*, а последние *L*—*M* символов *C* приобретают вид пробелов.
   c) Если типом данных *C* является точное целое, то должно быть представление значения в *v* в типе данных *C*, которое не упускает никаких первых значащих разрядов, и значение *C* устанавливается в это представление.
   d) Если типом данных является приближенное число, то значение *C* устанавливается в приближенное значение *v*.

9 УРОВНИ

Настоящий стандарт задает два уровня и отдельное средство расширения целостности.

Средство расширения целостности предоставляет следующие синтаксические конструкции наряду с их соответствующими Правилами синтаксиса и Общими правилами:

1) <спецификатор умолчания>;

2) все варианты <ограничения столбца>, отличные от NOT NULL и NOT NULL UNIQUE;

3) все варианты <определения ограничения таблицы>, за исключением UNIQUE (<перечень уникальности столбцов>);

4) операция REFERENCES (<перечень столбцов предоставления привилегий>).

Уровень 2 — это полный SQL язык баз данных, за исключением средства расширения целостности. Уровень 1 является подуровнем для Уровня 2, который удовлетворяет следующим дополнительным правилам.

1) 4.16 «Транзакции»:

a) Первое предложение в параграфе 1 заменяется на:

— Транзакция является последовательностью операций, включая операции над базами данных, и рассматривается как целое в отношении восстановления.

b) Параграф 2 удаляется.

c) Второе предложение в параграфе 3 удаляется.

2) 5.3 «<лексема>»:

Идентификатор должен состоять из не более чем 12 символов.

3) 5.4 «<имена>»:

<Имя таблицы> не должно содержать <идентификатор полномочий>.

4) 5.6 «<спецификация значения> и <спецификация цели>»

a) <спецификация значения> не должна содержать USER.

b) <спецификация параметра> не должна задавать <параметр-признак>.

c) <спецификация переменной> не должна задавать <переменную-признак>.

5) 5.7 «<спецификация столбца>»:

К правилу синтаксиса 4 добавляется следующее:

<Спецификация столбца> не должна быть внешней ссылкой.

6) 5.8 «<спецификация функции набора>», 5.24 «<подзапрос>», и 5.25 «<спецификация запроса>»:

<Функция мультинабора>, <подзапрос> и <спецификация запроса> не должны содержать ALL.

Примечание — В Уровне 1 сохранение дубликатов задается опусканием DISTINCT.

7) 5.8 «<спецификация функции набора>»:

<Функция набора различимых объектов> не должна содержать AVG, MAX, MIN или SUM.

8) 5.11 «<предикат сравнения>»:

<Операция сравнения> не должна содержать «< >».

Примечание — В Уровне 1 сравнение в форме «A< >B» выражается с помощью эквивалентного «NOT A=B».

9) 5.14 «<предикат подобия>»:

a) <Предикат подобия> не должен задавать ESCAPE <спецсимвол>.

b) <Предикат подобия> не должен задавать NOT.

Примечание — В Уровне 1 <предикат подобия>, содержащий NOT, будет выражаться с помощью эквивалентного <условия поиска> для «NOT <предикат подобия>».

10) 5.17 «<предикат существования>»:

<Предикат> не должен задавать <предикат существования>.

11) 5.22 «<спецификатор группировки>»:

К Общему правилу 2 добавляется следующее предложение:

«Группирование строк, в которых значение одного или более группирующих столбцов является нулем, задается разработчиком».

12) 5.25 «<спецификация запроса>»:

Правило синтаксиса 11 заменяется следующим:

«Разработчик определяет, является ли <спецификатор запроса> корректируемым или только считываемым».

13) 6.1 «<схема>»:

<Схема> не должна задаваться. Реализация уровня 1 должна предоставлять некоторый механизм для связывания <идентификатора полномочий> и <определения таблицы>, <определения представления> или <определения привилегий>.

14) 6.2 «<определение таблиц>»:

<Определение таблиц> не должно содержать <определение ограничения уникальности>. Реализация уровня 1 должна предоставлять некоторый механизм для задания ограничения уникальности для таблицы.

15) 6.3 «<определение столбца>»:

a) <Тип данных> в <определении столбца> не должен содержать REAL, DOUBLE PRECISION или NUMERIC.

b) <Определение столбца> должно задавать NOT NULL.

c) <Определение столбца> не должно задавать UNIQUE.

16) 6.9 «<определение представления>»:

<Определение представления> не должно содержать WITH CHECK OPTION.

17) 6.10 «<определение привилегий>»:

<Определение привилегий> не должно содержать WITH GRANT OPTION.

18) 7.3 «<процедура>»:

a) Правило синтаксиса 8, вариант (а) заменяется следующим:

Любой <тип данных> в <объявлении параметра> должен задавать CHARACTER.

b) В Общих правилах 3, вариант (а) каждое появление числа «100» заменяется выражением «положительное число, величина которого задается разработчиком».

c) В Общих правилах 3, вариант (б) (1) заменяется следующим:

Разработчик определяет, уничтожаются ли изменения, внесенные в базу данных при выполнении *S*.

19) 8.3 «<объявление курсора>»:

a) <Спецификатор упорядочения> не должен содержать <беззнаковое целое>.

b) <Спецификатор упорядочения> не должен содержать ASC.

Примечание — В Уровне 1 восходящий порядок задается опусканием DESC

c) <Выражение запроса> не должно содержать UNION.

Примечание — В Уровне 1 функция объединения не присутствует.

20) 8.7 «<оператор вставки>»:

<Оператор вставки> не должен содержать <спецификацию запроса>.

21) 8.11 «<оператор коррекции: по положению>» и 8.4 «<оператор удаления: по положению>»:

<Оператор SQL> не должен задавать <оператор корректировки: по положению> или <оператор удаления: по положению>.

ПРИЛОЖЕНИЕ А. <ВСТРОЕННАЯ ОБЩАЯ ПРОГРАММА SQL>

(Это приложение не является составной частью всего стандарта)

Функция

Задает встроенную прикладную программу SQL.

Формат

```
<встроенная общая программа SQL> ::=
            <встроенная программа SQL КОБОЛ>
            <встроенная программа SQL ФОРТРАН>
            <встроенная программа SQL ПАСКАЛЬ>
            <встроенная программа SQL ПЛ/1>
<встроенный оператор SQL> ::=
             <префикс SQL>
                 {<объявление курсора>
                 |<встроенное объявление исключений>
                 |<оператор SQL>
              <терминатор SQL>
<префикс SQL> ::=
          EXEC SQL
<терминатор SQL> ::=
          END — EXEC |;
<секция объявления встроенного SQL> ::=
      <объявление начала встроенного SQL>
    [<определение главной переменной>...]
     <объявление окончания встроенного SQL>
<объявление начала встроенного SQL> ::=
  <префикс SQL> BEGIN DECLARE SECTION
 [<терминатор SQL>]
<объявление окончания встроенного SQL> ::=
  <префикс SQL> END DECLARE SECTION
  <терминатор SQL>
<определение главной переменной> ::=
          <определение переменной КОБОЛ>
        |<определение переменной ФОРТРАН>
        |<определение переменной ПАСКАЛЬ>
        |<определение переменной ПЛ/1>
<имя встроенной переменной> ::=
          |<главный идентификатор>
<главный идентификатор> ::=
           <главный идентификатор КОБОЛ>
          |<главный идентификатор ФОРТРАН>
           |<главный идентификатор ПАСКАЛЬ>
           |<главный идентификатор ПЛ/1>
```

Правила синтаксиса

1) <Главная программа встроенного SQL> — это прикладная программа, которая состоит из текста языка программирования и текста SQL. Текст языка программирования должен соответствовать требованиям специфичного стандартного языка программирования. Текст SQL должен состоять из одного или более <операторов встроенного SQL> и, произвольно, одной или более <секций объявления встроенного SQL>.

2) <Оператор встроенного SQL>, <объявление начала встроенного SQL> или <объявление окончания встроенного SQL> содержащиеся во <встроенной программе SQL КОБОЛ>, должны содержать <признак конца SQL>, т. е. END—EXEC. <Оператор встроенного SQL>; <объявление начала встроенного SQL> или <объявление окончания встроенного SQL>, содержащиеся во <встроенной программе SQL ФОРТРАН>, не должны содержать <терминатор SQL>. <Оператор встроенного SQL>, <объявление начала встроенного SQL> или <объявление окончания встроенного SQL>, содержащиеся во <встроенной программе SQL ПЛ/1>, должны содержать <терминатор SQL>, т. е. полудвоеточие. Во <встроенной программе SQL ПАСКАЛЬ> <объявление начала встроенного SQL> должно содержать <терминатор SQL>, т. е. полудвоеточие; <объявление окончания встроенного SQL> или <оператор встроенного SQL>, за которым непосредственно следует <объявление начала встроенного SQL> или <оператор встроенного SQL>, должно содержать <терминатор SQL>, т. е. точка с запятой; в других случаях <объявление окончания встроенного SQL> или <оператор встроенного SQL> не должны содержать <терминатор SQL>, а должны заканчиваться в соответствии с правилами для операторов Паскаля.

3) <Префикс SQL>, <объявление начала встроенного SQL> или <объявление окончания встроенного SQL>, должны задаваться в пределах одной строки без комментариев. В остальных случаях правила продолжения строк и лексем от одной строки к другой и для помещения комментариев те же, что и для языка программирования содержащей его <главной программы встроенного SQL>.

4) <Объявление курсора>, содержащееся в <главной программе встроенного SQL>, должно предшествовать в тексте этой <главной программы встроенного SQL> любому <оператору SQL>, который ссылается на <имя курсора> в <объявлении курсора>.

5) Любой <главный идентификатор>, присутствующий в <операторе встроенного SQL> в <главной программе встроен

ного SQL>, должен определяться в точно одном <определении главной переменной>, содержащейся в этой <главной программе встроенного SQL>. Это <определение главной переменной> должно появляться в тексте <главной программы встроенного SQL> перед любым <оператором встроенного SQL>, который ссылается на <главный идентификатор >. <Определение главной переменной> должно быть таким, чтобы ссылка главного языка на <главный идентификатор> была справедливой при каждом <операторе встроенного SQL>, который содержит <главный идентификатор>.

6) <Определение главной переменной> определяет тип данных главного языка для <главного идентификатора>. Для каждого такого типа данных главного языка эквивалентный тип данных SQL задан в Приложении C, «<программа встроенного SQL КОБОЛ>», Приложении D «<программа встроенного SQL ФОРТРАН>», Приложении E «<программа встроенного SQL Паскаль>» и Приложении F «<программа встроенного SQL ПЛ/1>».

7) Что касается <главной программы встроенного SQL>H, то существует заключенный в SQL <модуль>*M*, выводимый из *H* следующим образом:

a) *M* содержит <спецификатор имени модуля>, чье <имя модуля> либо задается разработчиком, либо опускается.

b) *M* содержит <спецификатор языка>, который задает либо КОБОЛ, ФОРТРАН, Паскаль, либо ПЛ/1, где *H* соответственно <программа встроенного SQL КОБОЛ>, <программа встроенного SQL ФОРТРАН>, <программа встроенного SQL Паскаль> или <программа встроенного SQL ПЛ/1>.

c) *M* содержит <спецификатор полномочий модуля>, для которого <идентификатор полномочий модуля> определяется разработчиком.

d) Для каждого <объявления курсора> EC, содержащегося в *H*, *M* содержит одно <объявление курсора> PC и одну <процедуру> PS, которая содержит <оператор открытия>, ссылающийся к PC. <Имя процедуры> PS определяется разработчиком. PC является копией EC, в котором каждое отдельное <имя встроенной переменной> заменено на отдельное определенное разработчиком <имя параметра>. PS содержит <объявление параметра> для каждого <имени параметра>, содержащегося в PC, и <объявление параметра>, которое за-

дает SQLCODE. Порядок <объявлений параметра> в PS определяется разработчиком. <Объявление параметра>, которое соответствует данному отдельному <имени встроенной переменной> *V*, которое появляется в EC, задает отдельное <имя параметра>, с которым была произведена замена *V*, и <тип данных> SQL, который эквивалентен типу данных главного языка в *V*.

е) *M* содержит <процедуру>, соответствующую каждому <оператору SQL>, содержащемуся в *H*. <Процедура> PS из *M*, соответствующая <оператору SQL> ES из *H*, определяется следующим образом:

Возможны такие варианты:

— ES не является <оператором открытия>:

1 <Имя процедуры> PS определяется разработчиком.

2 <Оператор SQL> PS является копией ES, в котором каждое отдельное <имя встроенной переменной> согласованно заменено на отдельное <имя параметра>, определенное разработчиком.

3 PS содержит <объявление параметра> для каждого отдельного <имени параметра>, определенного разработчиком, которое содержится в <операторе SQL> PS, и <объявление параметра>, которое задает SQLCODE. Порядок <объявлений параметра> задается разработчиком. <Объявление параметра>, соответствующее данному отдельному <имени встроенной переменной> *V*, существующему в ES, задает отдельное <имя параметра>, с которым был произведен обмен с *V*, и <тип данных> SQL, эквивалентный типу данных главного языка в *V*.

4. Разработчик определяет и то, что одна <процедура> *M* может соответствовать более чем одному <оператору SQL> в *H*.

— ES является <оператором открытия>:

1 Пусть EC будет <объявлением курсора> в *H*, к которому ссылается ES.

2 PS является <процедурой> в *M*, которая содержит <оператор открытия>, который ссылается к <открытию курсора> в *M* согласно EC.

8) Что касается <главной программы встроенного SQL>, *H* то существует применяемый стандарт, согласующий главную программу *P*, выведенную из *H*, следующим образом:

a) Каждое <объявление начала встроенного SQL> и каждое <объявление окончания встроенного SQL> удалены.

b) Каждый <оператор встроенного SQL>, содержащий <объявление курсора> или <объявление встроенного исключения>, удален.

c) Каждый <оператор встроенного SQL>, содержащий <оператор SQL>, заменен на оператор главного языка, который осуществляет следующие операции:

— Процедура главного языка или подпрограмма CALL <процедуры> заключенного <модуля> *M* в *H*, который соответствует <оператору SQL>. Если <оператор SQL> не является <оператором открытия>, то аргументы CALL включают в себя каждый отдельный <главный идентификатор>, содержащийся в <операторе SQL>, наряду с переменной SQL. Если <оператор SQL> является <оператором открытия>, то аргументы CALL включают в себя каждый отдельный <главный идентификатор>, содержащийся в соответствующем <объявлении курсора> в *H* наряду с переменной SQL. Порядок аргументов в CALL соответствует порядку соответствующего <объявления параметра> в соответствующей <процедуре>.

Примечание — «Переменная SQLCODE» сокращается как «SQLCOD» в <программе встроенного SQL ФОРТРАН>. См. Правило синтаксиса 6 в Приложении D, «<программа встроенного SQL ФОРТРАН>».

— Операция исключения, как задано в Приложении B, «<объявление исключений>».

9) Выведение содержащейся программы *P* и содержащегося <модуля> *M* в <главной программе встроенного SQL> *H* эффективно осуществляет обработку команд манипулирования текстом программы главного языка, таких как включение или копирование текста.

10) При <главной программе встроенного SQL> *H* с содержащимся <модулем> *M* и содержащейся программой *P*, определенной выше:

a) Содержащийся в *H* <модуль> *M* должен быть стандартным <модулем> SQL, как задано форматами и Правилами синтаксиса данного стандарта.

b) Если *H* является <программой встроенного SQL КОБОЛ>, то содержащаяся программа *P* должна быть

стандартной соответствующей программой КОБОЛ. Если *H* — <программа встроенного SQL ФОРТРАН>, то содержащаяся программа *P* должна быть стандартной соответствующей программой ФОРТРАН. Если *H* является <программой встроенного SQL Паскаль>, то содержащаяся в ней программа *P* должна быть стандартной соответствующей программой Паскаль. Если *H* является <программой встроенного SQL ПЛ/1>, то содержащаяся в ней программа *P* должна быть стандартной соответствующей программой ПЛ/1.

Общие правила

1) Интерпретация <главной программы встроенного SQL> *H* по определению эквивалентна интерпретации содержащейся в *H* программы *P* и содержащегося в *H* <модуля> *M*.

ПРИЛОЖЕНИЕ В <ОБЪЯВЛЕНИЕ ВСТРОЕННОЙ ОСОБОЙ СИТУАЦИИ>

(Это приложение не является составной частью всего стандарта)

Функция

Задает действие, которое следует предпринять, когда <оператор SQL> приводит к особой ситуации.

Формат

<объявление встроенной особой ситуации> : : =
WHENEVER <условие> <реакция на особую ситуацию>

<условие> : : =
SQLERROR | NOT FOUND

<реакция на особую ситуацию> : : =
CONTINUE | <переход>

<переход> : : =
{GOTO | GO TO} <цель>

<цель> : : = :<главный идентификатор> | <целое без знака>

Правила синтаксиса

1) Возможны следующие варианты:

a) Если <объявление встроенной особой ситуации> содержится в <программе встроенного SQL КОБОЛ>, то <цель> <перехода> должна задавать <главный идентификатор>, т. е. имя раздела или неуточненное имя параграфа.

b) Если <объявление встроенной особой ситуации> содержится в <программе встроенного SQL ФОРТРАН>, то <цель> <перехода> должна быть <целым без знака>, т. е. меткой оператора, которая появляется в том же блоке программы, что и <переход>.

c) Если <объявление встроенной особой ситуации> содержится в <программе встроенного SQL Паскаль>, то <цель> <перехода> должна быть <целым без знака>, т. е. меткой.

d) Если <объявление встроенной особой ситуации> содержится в <программе встроенного SQL ПЛ/1>, то <цель> <перехода> должна задавать <главный идентификатор>, т. е. постоянную метку или переменную метку.

2) <Объявление встроенной особой ситуации>, содержащееся в <главной программе встроенного SQL>, применяется к <оператору SQL>, содержащемуся в этой <главной программе встроенного SQL>, тогда и только тогда, когда <оператор SQL>

появляется после <объявления встроенной особой ситуации> в последовательности текста <главной программы встроенного SQL>, и не существует других <объявлений встроенной особой ситуации>, которые задают это же <условие>, между <объявлением встроенной особой ситуации> и <оператором SQL> в текстовой последовательности <главной программы встроенного SQL>.

3) Если <объявление встроенной особой ситуации> задает <переход>, то <главный идентификатор> или <целое без знака> в <переходе> должны быть такими, чтобы оператор GO TO главного языка, задающий этот <главный идентификатор> или <целое без знака>, был справедливым при каждом <операторе SQL>, к которому применяется <объявление встроенной особой ситуации>.

Общие правила

1) Непосредственно после выполнения <оператора SQL> в <главной программе встроенного SQL>: возможны следующие варианты:

a) Если значение переменной SQLCODE (SQLCOD) равно +100, а <главная программа встроенного SQL> содержит <объявление встроенной особой ситуации>, которое применяется к <оператору SQL> и чьим условием является NOT FOUND, а также чьей <реакцией на особую ситуацию> является <переход>, то выполняется оператор GO TO главного языка, задавая <главный идентификатор> или <целое без знака> в <переходе>.

b) Если значение переменной SQLCODE (SQLCOD) является отрицательным числом, а <главная программа встроенного SQL> содержит <объявление встроенной особой ситуации>, которое применяется к <оператору SQL>, чьим <условием> является SQLERROR, а <реакцией на особую ситуацию> — <переход>, то выполняется оператор GO TO главного языка, задавая <главный идентификатор> или <целое без знака> для <перехода>.

c) Если <главная программа встроенного SQL> не содержит <объявления встроенной особой ситуации>, которая применяется к <оператору SQL> или если она содержит <объявление встроенной особой ситуации>, которое применяется к <оператору SQL> и дает CONTINUE, то для <оператора SQL> не выполняются никакие дальнейшие операции.

ПРИЛОЖЕНИЕ С. <ПРОГРАММА ВСТРОЕННОГО SQL КОБОЛ>

(Это приложение не является составной частью всего стандарта)

Функция

Задает модуль SQL, встроенный в программу КОБОЛ.

Формат

```
<программа встроенного SQL КОБОЛ> :: =
        См. правила синтаксиса.
<определение переменной КОБОЛ> :: =
        {01|77} <главный идентификатор КОБОЛ>
                        <спецификация типа КОБОЛ>
        [<символ> ...].
<главный идентификатор КОБОЛ> :: =
        См. Правило синтаксиса 3.
<спецификация типа КОБОЛ> :: =
          <тип символа КОБОЛ>
        |<тип числа КОБОЛ>
        |<тип целое КОБОЛ>
<тип символа КОБОЛ> :: =
        PIC[TURE][IS]X(<длина>)
<тип числа КОБОЛ> :: =
        PIC[TURE][IS]
        S{<девятки> [V<девятки>]
         |<девятки> V
       |V<девятки>}
        [USAGE [IS]] DISPLAY SIGN LEADING SEPARATE
<тип целое КОБОЛ> :: =
        PIC[TURE][IS]
        S <девятки>
        [USAGE [IS]] COMP[UTATIONAL]
<девятки> :: = {9[(<целое без знака>)]}.
```

Правила синтаксиса

1) <Программа встроенного SQL КОБОЛ> является прикладной программой, которая состоит из текста КОБОЛ и текста SQL. Текст КОБОЛ должен соответствовать стандартному языку КОБОЛ. Текст SQL должен состоять из одного или более <операторов встроенного SQL> и произвольно из одного или более разделов <объявления встроенного SQL>.

2) <Оператор встроенного SQL> в <программе встроенного SQL КОБОЛ> может быть задан везде, где может быть задан оператор КОБОЛ в Процедурном разделе <программы встроенного SQL КОБОЛ>. Если оператор КОБОЛ может быть непос-

редственно предварен именем параграфа, то можно непосредственно предварить <оператор встроенного SQL> именем параграфа.

3) <Главный идентификатор КОБОЛ> — это любое истинное имя переменной КОБОЛ. <Главный идентификатор КОБОЛ> должен содержаться в <программе встроенного SQL КОБОЛ>.

4) <Определение переменной КОБОЛ> является ограниченной формой ввода описания данных КОБОЛ, которая определяет главную переменную.

a) <Определение переменной КОБОЛ> должно быть истинным вводом описания данных в Раздел данных программы, выведенный из <программы встроенного *SQL КОБОЛ*>.

b) Произвольная последовательность <символов> в <определении переменной КОБОЛ> может задавать спецификатор VALUE. Разработчик определяет, могут ли быть заданы другие спецификаторы. Последовательность <символов> должна быть такой, чтобы <определение переменной КОБОЛ> было истинным вводом описания данных КОБОЛ.

c) <Тип символа КОБОЛ> описывает переменную строки символов. Эквивалентным типом данных в SQL является CHARACTER той же длины.

d) <Тип числа КОБОЛ> описывает точную численную переменную. Эквивалентным типом данных в SQL является NUMERIC той же точности и масштаба.

e) <Тип целого КОБОЛ> описывает точную численную переменную. Эквивалентным типом данных в SQL является INTEGER.

Примечание — Этот тип данных сохраняется только дря SQLCODE; см. Правило синтаксиса 5.

5) <Программа встроенного SQL КОБОЛ> должна содержать переменную с именем SQLCODE, определяемую типом данных с использованием COMPUTATIONAL шаблона S9 (PC), где PC — определяемая разработчиком точность, заданная для параметров SQLCODE в 7.3 «<процедура>».

Общие правила

См. приложение А «<главная программа встроенного SQL>».

ПРИЛОЖЕНИЕ D <ПРОГРАММА ВСТРОЕННОГО SQL ФОРТРАН>

(Это приложение не является составной частью данного стандарта)

Функция

Задает модуль SQL, встроенный в программу ФОРТРАН.

Формат

<программа встроенного SQL ФОРТРАН> :: =
    См. Правила синтаксиса.
<определение переменной ФОРТРАН> :: =
    <спецификация типа ФОРТРАН>
    <главный идентификатор ФОРТРАН>
    [{, <главный идентификатор ФОРТРАН>}...]
<главный идентификатор ФОРТРАН> :: =
    См. Правило синтаксиса 4.
<спецификация типа ФОРТРАН> :: =
    CHARACTER [*<длина>]
  | INTEGER
  | REAL
  | DOUBLE PRECISION

Правила синтаксиса

1) <Программа встроенного SQL ФОРТРАН> является прикладной программой, которая состоит из текста ФОРТРАН и текста SQL. Текст ФОРТРАН должен соответствовать стандартному языку ФОРТРАН. Текст SQL должен соответствовать одному или более <операторам встроенного SQL> и произвольно одному или более <разделам объявления встроенного SQL>.

2) <Оператор встроенного SQL> может быть задан тогда, когда можно задать исполнительный оператор ФОРТРАН. <Оператор встроенного SQL>, предшествующий какому-либо исполнительному оператору ФОРТРАН в содержащей его <программе встроенного SQL ФОРТРАН>, не должен иметь номер оператора ФОРТРАН. В других случаях, если оператор ФОРТРАН мог бы иметь номер оператора, то <оператор встроенного SQL> может иметь номер оператора.

3) Для <операторов встроенного SQL> большое значение имеют пробелы. Правила для <разделителей> в <операторе встроенного SQL> такие же, как приведены в 5.3 «<лексема>».

4) <Главный идентификатор ФОРТРАН> — это любое справедливое имя переменной языка ФОРТРАН. <Главный идентификатор ФОРТРАН> должен содержаться в <программе встроенного SQL ФОРТРАН>.

5) <Определение переменной ФОРТРАН> является ограниченной формой оператора типа ФОРТРАН, который задает главную переменную.

a) <Определение переменной ФОРТРАН> должно быть достоверным оператором типа ФОРТРАН в программе, выведенной из <программы встроенного SQL ФОРТРАН>.

b) CHARACTER описывает переменную строки символов. Эквивалентным типом данных SQL является CHARACTER такой же длины.

c) ЦЕЛОЕ описывает целую численную переменную. Эквивалентным типом данных SQL является INTEGER.

d) REAL описывает приближенную числовую переменную. Эквивалентным типом данных в SQL является REAL.

e) DOUBLE PRECISION описывает приближенную числовую переменную. Эквивалентным типом данных SQL является DOUBLE PRECISION.

6) <Программа встроенного SQL ФОРТРАН> должна содержать переменный поименованный SQLCOD, определенный с помощью типа данных INTEGER. В <программе встроенного SQL ФОРТРАН> SQLCOD должен использоваться в качестве сокращения для SQLCODE.

Общие правила

См. Приложение А, «<главная программа встроенного SQL>.

ПРИЛОЖЕНИЕ Е. <ПРОГРАММА ВСТРОЕННОГО SQL Паскаль>

(Это приложение не является составной частью данного стандарта)

Функция

Задает модуль SQL, встроенный в программу Паскаль.

Формат

<программа встроенного SQL Паскаль> : : =
    См. Правила синтаксиса.

<определение переменной Паскаля > : : =
    <главный идентификатор Паскаля>
    [{, <главный идентификатор Паскаля>}...]:
    <спецификация типа Паскаля>

<главный идентификатор Паскаля> : : =
    См. Правило синтаксиса 3.

<спецификация типа Паскаля> : : =
    PACKED ARRAY <левая скобка> I <длина>
        <правая скобка> OF CHAR
  | INTEGER
  | REAL

<левая скобка> : : = [

<правая скобка> : : =]

Правила синтаксиса

1) <Программа встроенного SQL Паскаль> является прикладной программой, которая состоит из текста на языке Паскаль и текста на языке SQL. Текст Паскаль должен соответствовать стандартному языку Паскаль. Текст SQL должен состоять из одного или более <операторов встроенного SQL> и произвольно одного или более <разделов объявления встроенного SQL>.

2) <Оператор встроенного SQL> может быть задан там, где может быть задан оператор Паскаля. <Оператор встроенного SQL> может предваряться меткой Паскаля.

3) <Главный идентификатор Паскаля> — это любой достоверный идентификатор-переменная Паскаля. <Главный идентификатор Паскаля> должен содержаться в <программе встроенного SQL Паскаль>.

4) <Определение переменной Паскаля> определяет главную переменную.

a) <Определение переменной Паскаля> должно быть достоверным объявлением переменной Паскаля в программе, выведенной из <программы встроенного SQL Паскаль>.

b) PACKED ARRAY [1..<длина>] OF CHAR описывает переменную строки символов. Эквивалентным типом данных в SQL является CHARACTER такой же длины.

c) INTEGER описывает точную числовую переменную. Эквивалентным типом данных в SQL является INTEGER.

d) REAL описывает приближенную числовую переменную. Эквивалентным типом данных в SQL является REAL.

5) <Программа встроенного SQL Паскаль> должна содержать переменный поименованный SQLCODE, определенный с помощью типа данных INTEGER.

Общие правила

См. Приложение А, «<главная программа встроенного SQL>».

ПРИЛОЖЕНИЕ F. <ПРОГРАММА ВСТРОЕННОГО SQL ПЛ/1>

(Это Приложение не является составной частью данного стандарта)

Функция

Задает модуль SQL, встроенный в программу ПЛ/1.

Формат

```
<программа встроенного SQL ПЛ/1> :: =
      См. Правила синтаксиса.
<определение переменной ПЛ/1> :: =
      {DCL | DECLARE}
      {<главный идентификатор ПЛ/1>
       |<главный идентификатор ПЛ/1>
        [{, <главный идентификатор ПЛ/1>}...]}
        <спецификация типа ПЛ/1>
       [<символ> ...]
<главный идентификатор ПЛ/1> :: =
      См. Правило синтаксиса 3.
<спецификация типа PL/1> :: =
        CHAR[ACTER] <длина>
      | {DEC[IMAL] FIXED | FIXED DEC[IMAL]}
                        [точность> [, <масштаб>]
      | {BIN[ARY] FIXED | FIXED BIN[ARY]}
      [(<точность>)]
      | {BIN[ARY] FLOAT | FLOAT BIN[ARY]}
      (<точность>)
```

Правила синтаксиса

1) <Программа встроенного SQL ПЛ/1> является прикладной программой, которая состоит из текста ПЛ/1 и текста SQL. Текст ПЛ/1 должен соответствовать стандарту ПЛ/1. Текст SQL должен состоять из одного или более <операторов встроенного SQL> и произвольно одного или более <разделов объявления встроенного SQL>.

2) <Оператор встроенного SQL> может быть задан тогда, когда может быть задан оператор ПЛ/1 в процедурном блоке. Если оператор ПЛ/1 может включать префиксную метку, то <оператор встроенного SQL> может непосредственно предваряться префиксной меткой.

3) <Главный идентификатор ПЛ/1> является любым достоверным идентификатором переменной ПЛ/1. <Главный идентификатор ПЛ/1> должен содержаться в <программе встроенного SQL ПЛ/1>.

4) <Определение переменной ПЛ/1> определяет одну или более главных переменных.

a) <Определение переменной ПЛ/1> должно быть достоверным объявлением данных в ПЛ/1 в программе, выводимой из <программы встроенного SQL ПЛ/1>.

b) <Определение переменной ПЛ/1> должно задавать скалярную переменную, а не матрицу или структуру.

c) Произвольная строка <символов> в <определении переменной ПЛ/1> может задавать спецификатор INITIAL. Разработчик определяет, могут ли задаваться другие спецификаторы. Последовательность <символов> должна быть такой, чтобы <определение переменной ПЛ/1> было достоверным оператором ПЛ/1 DECLARE.

d) CHAR[ACTER] описывает переменную строку символов. Эквивалентным типом данных в SQL является CHARACTER той же длины.

e) FIXED DEC[IMAL] описывает точную числовую переменную. <Масштаб> не должна быть больше, чем <точность>. Эквивалентным типом данных SQL является DECIMAL с той же <точностью> и <масштабом>.

f) FLOAT BIN[ARY] описывает приближенную числовую переменную. Эквивалентным типом данных является FLOAT с той же <точностью>.

g) FIXED BIN[ARY] описывает точную числовую переменную. Эквивалентным типом данных SQL является INTEGER.

Примечание — Этот тип данных сохраняется только для SQLCODE; см. правило синтаксиса 5.

5) <Программа встроенного SQL ПЛ/1> должна содержать переменный поименованный SQLCODE, определенный с помощью типа данных FIXED BINARY (PP), где PP — определяемая разработчиком <точность>, заданная для параметров SQLCODE в 7.3, «<процедура>».

Общие правила

См. Приложение А, «<главная программа встроенного SQL>».

УКАЗАТЕЛЬ

1 — операция;
2 — все;
3 — функция полного набора;
4 — приближенный числовой литерал;
5 — приближенный числовой тип;
6 — идентификатор полномочий;
7 — предикат интервала;
8 — логический коэффициент;
9 — логическое первичное;
10 — логический терм;
11 — символ;
12 — представление символов;
13 — литерал строки символов;
14 — тип строки символов;
15 — определение ограничения проверки;
16 — оператор открытия;
17 — тип символа КОБОЛ;
18 — главный идентификатор КОБОЛ;
19 — тип целого КОБОЛ;
20 — числовой тип КОБОЛ;
21 — спецификация типа КОБОЛ;
22 — определение переменной КОБОЛ;
23 — ограничение столбца;
24 — определение столбца;
25 — имя столбца;
26 — спецификация столбца;
27 — комментарий;
28 — начало комментария;
29 — оператор блокировки;
30 — операция сравнения;
31 — предикат сравнения;
32 — условие;
33 — имя корреляции;
34 — имя курсора;
35 — спецификация курсора;
36 — тип данных;
37 — объявление курсора;
38 — спецификатор умолчания;
39 — оператор удаления: по положению;
40 — оператор удаления: поиск;
41 — лексема — ограничитель;
42 — цифра;

43 — функция набора различимых объектов;
44 — объявление встроенной особой ситуации;
45 — объявление начала встроенного SQL;
46 — программа встроенного SQL КОБОЛ;
47 — объявление окончания встроенного SQL;
48 — программа встроенного SQL ФОРТРАН;
49 — главная программа встроенного SQL;
50 — программа встроенного SQL Паскаль;
51 — программа встроенного SQL ПЛ/1;
52 — оператор встроенного SQL;
53 — имя встроенной переменной;
54 — спецсимвол;
55 — литерал целого;
56 — тип целого;
57 — операция особой ситуации;
58 — предикат существования;
59 — порядок;
60 — коэффициент;
61 — оператор выборки;
62 — перечень целей выборки;
63 — главный идентификатор ФОРТРАН;
64 — спецификация типа ФОРТРАН;
65 — определение переменной ФОРТРАН;
66 — спецификатор отображения;
67 — переход;
68 — перечень предоставления столбцам привилегий:
69 — получатель привилегий;
70 — спецификатор группировки;
71 — спецификатор выборки групп;
72 — главный идентификатор;
73 — определение главной переменной;
74 — идентификатор;
75 — предикат принадлежности;
76 — перечень значений принадлежности;
77 — параметр — признак;
78 — переменная — признак;
79 — перечень столбцов вставки;
80 — оператор вставки;
81 — вставить значение;
82 — перечень значений вставки;
83 — ключевое слово;
84 — спецификатор языка;
85 — левая скобка;
86 — длина;

87 — буква;
88 — буква или цифра;
89 — предикат подобия;
90 — литерал;
91 — строчная буква;
92 — мантисса;
93— модуль;
94 — спецификатор полномочий модуля;
95 — идентификатор полномочий модуля;
96 — имя модуля;
97 — спецификатор имени модуля;
98 — новая строка;
99 — девятки;
100 — лексема, не являющаяся ограничителем;
101 — символ, отличный от кавычек;
102 — предикат нуля;
103 — числовой литерал;
104 — предметный столбец: по положению;
105 — предметный столбец: поиск;
106 — оператор открытия;
107 — спецификатор порядка;
108 — объявление параметра;
109 — имя параметра;
110 — спецификация параметра;
111 — главный идентификатор Паскаля;
112 — спецификация типа Паскаль;
113 — определение переменной Паскаля;
114 — структура;
115 — главный идентификатор ПЛ/1;
116 — спецификация типа ПЛ/1;
117 — определение переменной ПЛ/1;
118 — точность;
119 — предикат;
120 — первичный;
121 — определение привилегий;
122 — привилегии;
123 — процедура;
124 — имя процедуры;
125 — префикс;
126 — квантифицированный предикат;
127 — квантор;
128 — выражение запроса;
129 — спецификация запроса;
130 — терм запроса;

131 — кавычки;
132 — перечень столбцов ссылок;
133 — таблицы и столбцы, к которым обращаются;
134 — спецификация ссылок;
135 — обращающиеся столбцы;
136 — определение ограничений на ссылки;
137 — спецификация результата;
138 — правая скобка;
139 — оператор возврата;
140 — масштаб;
141 — схема;
142 — спецификатор полномочий схемы;
143 — идентификатор полномочий схемы;
144 — элемент схемы;
145 — условие поиска;
146 — перечень выбора;
147 — оператор выбора;
148 — перечень целей выбора;
149 — разделитель;
150 — спецификатор набора: по положению;
151 — спецификатор набора: поиск;
152 — спецификация функции набора;
153 — целое со знаком;
154 — некоторый;
155 — спецификация упорядочения;
156 — пробел;
157 — спецсимвол;
158 — префикс SQL;
159 — оператор SQL;
160 — терминатор SQL;
161 — параметр SQLCODE;
162 — подзапрос;
163 — определение ограничений таблицы;
164 — определение таблицы;
165 — элемент таблицы;
166 — табличное выражение;
167 — идентификатор таблицы;
168 — имя таблицы;
169 — табличная ссылка;
170 — цель;
171 — спецификация цели;
172 — терм;
173 — лексема;
174 — подчеркивание;

175 — перечень столбцов уникальности;
176 — определение ограничения уникальности;
177 — спецификация уникальности;
178 — целое без знака;
179 — оператор корректировки: по положению;
180 — оператор корректировки: поиск;
181 — прописная буква;
182 — выражение значения;
183 — спецификация значения;
184 — спецификация переменной;
185 — перечень столбцов представления;
186 — определение представления;
187 — спецификатор выборки;
188 — раздел объявления встроенного SQL.

УДК 681.3.04:006.354

П 85

Ключевые слова: обработка данных, взаимосвязь открытых систем, базы данных, языки программирования.

gost.e-ecolog.ru