**Использование escape-последовательностей SQL**

Драйвер Microsoft JDBC для SQL Server поддерживает использование escape-последовательностей SQL, описанных в определении API JDBC. Escape-последовательности используются в инструкции SQL для передачи драйверу сведений о том, что экранированные части SQL-строки должны обрабатываться иначе. При обработке драйвером JDBC части escape-последовательности SQL-строки выполняется преобразование этой части строки в SQL-код, распознаваемый SQL Server.

Существует пять типов escape-последовательностей, требуемых API JDBC, и все они поддерживаются драйвером JDBC:

* Литералы-шаблоны LIKE
* Обработка функций
* Литералы даты и времени
* Вызовы хранимых процедур
* Внешние объединения
* Escape-синтаксис LIMIT *(типа TOP)*

Драйвер JDBC использует следующий синтаксис escape-последовательности:

{keyword ...parameters...}

**Примечание**

Для драйвера JDBC всегда включена обработка escape-последовательностей SQL.

В следующих разделах описывается пять типов escape-последовательностей и способы их поддержки драйвером JDBC.

**Литералы-шаблоны LIKE**

Драйвер JDBC поддерживает синтаксис {escape 'escape character'} для использования шаблонов предложения LIKE в качестве литералов. Например, в следующем коде возвращаются значения col3, для которых значение col2 начинается с символа подчеркивания (и не используется как шаблон).

Java

ResultSet rst = stmt.executeQuery("SELECT col3 FROM test1 WHERE col2

LIKE '\\\_%' {escape '\\'}");

**Примечание**

Escape-последовательность должна быть в конце инструкции SQL. При наличии в командной строке нескольких инструкций SQL escape-последовательность должна быть в конце каждой соответствующей инструкции SQL.

**Обработка функций**

Драйвер JDBC поддерживает использование escape-последовательностей функций в инструкциях SQL со следующим синтаксисом:

SQL

{fn functionName}

где functionName представляет собой функцию, поддерживаемую драйвером JDBC. Пример:

SQL

SELECT {fn UCASE(Name)} FROM Employee

В следующей таблице перечислены различные функции, поддерживаемые драйвером JDBC при использовании escape-последовательностей функций:

| **Строковые функции** | **Числовые функции** | **Функции Datetime** | **Системные функции** |
| --- | --- | --- | --- |
| ASCII  CHAR  CONCAT  DIFFERENCE  INSERT  LCASE  LEFT  LENGTH  LOCATE  LTRIM  REPEAT  REPLACE  RIGHT  RTRIM  SOUNDEX  SPACE  SUBSTRING  UCASE | ABS  ACOS  ASIN  ATAN  ATAN2  CEILING  COS  COT  DEGREES  EXP  FLOOR  LOG  LOG10  MOD  PI  POWER  RADIANS  RAND  ROUND  SIGN  SIN  SQRT  TAN  TRUNCATE | CURDATE  CURTIME  DAYNAME  DAYOFMONTH  DAYOFWEEK  DAYOFYEAR  EXTRACT  HOUR  MINUTE  MONTH  MONTHNAME  NOW  QUARTER  SECOND  TIMESTAMPADD  TIMESTAMPDIFF  WEEK  YEAR | DATABASE  IFNULL  Пользователь |

**Примечание**

Использование функции, не поддерживаемой базой данных, приведет к возникновению ошибки.

**Литералы даты и времени**

Ниже приводится синтаксис escape-последовательности для литералов даты, времени и отметок времени:

SQL

{literal-type 'value'}

где literal-type является одним из следующих:

| **Тип литерала** | **Описание** | **Формат значения** |
| --- | --- | --- |
| d | Дата | гггг-мм-дд |
| t | Time | чч:мм:сс [1] |
| ts | TimeStamp | гггг-мм-дд чч:мм:сс[.f...] |

Пример:

SQL

UPDATE Orders SET OpenDate={d '2005-01-31'}

WHERE OrderID=1025

**Вызовы хранимых процедур**

Драйвер JDBC поддерживает синтаксис escape-последовательностей {? = call proc\_name(?,...)} и {call proc\_name(?,...)} для вызовов хранимых процедур, в зависимости от того, требуется ли обработка возвращаемого параметра.

Процедура представляет собой исполняемый объект, который хранится в базе данных. Обычно процедурой является одна или несколько заранее скомпилированных инструкций SQL. Ниже приводится синтаксис escape-последовательности вызова хранимой процедуры:

SQLКопировать

{[?=]call procedure-name[([parameter][,[parameter]]...)]}

где procedure-name указывает имя хранимой процедуры, а parameter указывает параметр хранимой процедуры.

Дополнительные сведения об использовании call escape-последовательности с хранимыми процедурами см. в разделе [Использование инструкций с хранимыми процедурами](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/using-statements-with-stored-procedures?view=sql-server-ver15).

**Внешние объединения**

Драйвер JDBC поддерживает синтаксис левого, правого и полного внешнего соединения SQL92. Ниже приводится синтаксис escape-последовательности для внешних соединений:

SQL

{oj outer-join}

где внешнее соединение:

SQL

table-reference {LEFT | RIGHT | FULL} OUTER JOIN

{table-reference | outer-join} ON search-condition

где table-reference является именем таблицы, а search-condition представляет собой условия соединения, которые необходимо использовать для таблиц.

Пример:

SQL

SELECT Customers.CustID, Customers.Name, Orders.OrderID, Orders.Status

FROM {oj Customers LEFT OUTER JOIN

Orders ON Customers.CustID=Orders.CustID}

WHERE Orders.Status='OPEN'

Драйвер JDBC поддерживает следующие escape-последовательности внешнего соединения:

* Левые внешние соединения
* Правые внешние соединения
* Полные внешние соединения
* Вложенные внешние соединения

**Escape-синтаксис LIMIT**

**Примечание**

Escape-синтаксис LIMIT поддерживается только драйвером Microsoft JDBC Driver 4.2 (и более поздних версий) для SQL Server при использовании JDBC 4.1 или более поздней версии.

Escape-синтаксис LIMIT выглядит следующим образом:

SQL

LIMIT <rows> [OFFSET <row offset>]

Escape-синтаксис состоит из двух частей: <*строки*> — это обязательная часть, которая указывает число возвращаемых строк. OFFSET и <*смещение строки*> — это необязательные компоненты, указывающие число строк, которые необходимо пропустить перед началом получения строк. Драйвер JDBC поддерживает только обязательную часть, преобразуя запрос для использования TOP вместо LIMIT. SQL Server не поддерживает предложение LIMIT. **Драйвер JDBC не поддерживает необязательную часть <смещение строки>. При ее использовании драйвер вызовет исключение**.