

# Язык *SQL* для работы с базами данных

В настоящее время язык *SQL* является общепринятым стандартом для работы с реляционными базами данными.

# Из истории языка *SQL*

- Предшественником этого языка является язык *SEQUEL*, который был использован в качестве инструментария запросов в экспериментальной системе **SYSTEM R**. Разработка этой системы была выполнена исследовательским центром компании *IBM* в Йорктаун-Хейтс (**Сан-Хосе, Калифорния**) в период с **1975** по **1979** годы.

Реализация языка **SQL** в среде коммерческой СУБД впервые была осуществлена в **1979** году компанией *Relation Software Inc.*, на базе которой впоследствии была создана известная корпорация **Oracle Cor**

- Первая стандартизация языка **SQL** была осуществлена в **1986** году американским национальным институтом стандартов – *American National Standards Institute, ANSI* и в **1987** году Международной организацией по стандартизации – *International Organization for Standardization, ISO*. Кроме стандарта **SQL-86**, известны следующие стандарты: **SQL-89 (SQL 1)**, **SQL-92 (SQL 2)** и **SQL-99 (SQL 3)**.

- В **SQL 1** стандарт был определен только на элементы ядра языка.
- Основным отличием диалектов языка **SQL** является несовпадение в определении таких элементов, как: **коды ошибок, типы данных, системные таблицы, последовательность сравнения, динамический SQL, интерактивный SQL** и др.
- Наиболее существенные отличия связаны с различными **типами данных**, применяемыми в коммерческих СУБД.

- Большинство из указанных элементов были стандартизованы в **SQL-92**. Кроме того, в **SQL 2** формально определены такие операции, как *полное, левое и правое внешние соединения*, спецификации для **динамического SQL** и **встроенного SQL** для ряда включающих языков программирования *C, Cobol, PL/1* и др.
- Основное отличие стандарта **SQL 3** – это определение *объектных свойств* базы данных, **темпоральных** баз данных, средств для управления **мультимедийными данными**.

# SQL – непроцедурный язык программирования

- Основными элементами языка являются: **ключевые слова**, **предложения** и **команды**.
- **Ключевое слово** – это отдельный элемент SQL: *SELECT, FROM, WHERE, HAVING, ALL, ANY, AVG, MIN, MAX, COUNT, SUM, BETWEEN, AND, CHAR, DECIMAL, FLOAT, INTEGER, CHECK, CREATE, INSERT, INTO, DELETE, UPDATE, DISTINCT, EXISTS, FOREIGN, GRANT, GROUP, IN, IS, KEY, LIKE, NOT, NULL, NUMERIC, ORDER, PRIMARY, PUBLIC, REFERENCES, SMALLINT, TABLE, TO, UNION, UNIQUE, USER, VALUES, VIEW, WITH, INDEX* и др.
- **Предложение** – это часть команды SQL, содержащая определенные ключевые слова. Например: *SELECT fio, kaf; FROM podr* и др.
- **Команда** – это комбинация из двух и более предложений.
  - Например,
  - *SELECT \**
  - *FROM podr;*
- Альтернативное наименование команды – это **SQL – оператор**.

Для удобства чтения и редактирования команд **SQL** рекомендуется использовать следующие **правила**:

- **ключевые слова нельзя** сокращать и переносить с одной строки на другую;
- **предложения** формируются на отдельных строках;
- **команды** могут занимать одну или несколько строк;
- **команды не различают** регистры символов. Однако **целесообразно** для **ключевых слов** применять **заглавные буквы**, а для **имен таблиц, столбцов, переменных** – **строчные**.

# Назначение команд **SQL**

• Название команд	Назначение и классификация команды
• <b>SELECT</b>	Выбор данных из базы данных
• <b>INSERT</b> <b>DML).</b>	Команды языка манипулирования данными ( <i>Data Manipulation Language – DML</i> ). Соответственно включение, изменение и удаление данных
• <b>UPDATE</b>	
• <b>DELETE</b>	
• <b>CREATE</b>	Команды языка определения данных ( <i>Data Definition Language – DDL</i> ). Создание, изменение и удаление структур данных
• <b>ALTER</b>	
• <b>DROP</b>	
• <b>RENAME</b>	Управление изменениями, производимыми командами <i>DML</i>
• <b>TRUNCATE</b>	
• <b>COMMIT</b>	
• <b>ROLLBACK</b>	
• <b>SAVEPOINT</b>	
• <b>GRANT</b>	Команды языка управления данными ( <i>Data Control Language – DCL</i> ). Предоставление и изъятие прав доступа к базе данных и элементам ее
• <b>REVOKE</b> объектов	

# Описание таблиц и связей между ними

При описании таблиц необходимо указать:

- имя таблицы;
- имена атрибутов/столбцов;
- тип и размер данных для каждого атрибута;
- первичный ключ;
- внешний и родительский ключи;
- ограничения для атрибутов;
- ограничения целостности объектов;
- наличие статуса *NULL* для значений атрибута;
- значение по умолчанию для атрибута;
- уникальность значений атрибута.

# В *Oracle SQL* применяются следующие правила задания имен:

- **имена таблиц и столбцов** могут иметь длину от **1 до 30** СИМВОЛОВ;
- **имена** должны содержать только символы **A-Z, a-z, 0-9, –** (символ подчеркивания), **\$** и **#**;
- **имена не должны** совпадать с именами других объектов, принадлежащих тому же пользователю сервера *Oracle*, а также совпадать со словами, зарезервированными сервером *Oracle*.

Рекомендуется использовать описательные имена для таблиц и атрибутов. При совпадении имен атрибутов в разных таблицах при выполнении команд с этими таблицами необходимо применять полные имена атрибутов. Полное имя содержит имя таблицы и имя атрибута, разделенные точкой:

- **<имя таблицы>.<имя атрибута> .**

# Основными типами данных, поддерживаемыми современными промышленными СУБД, являются:

- символьные данные фиксированной длины – *CHAR (n)*;
- символьные данные переменной длины – *VARCHAR (n)*;
- целые числа – *INTEGER*;
- укороченные целые числа – *SMALLINT*;
- масштабируемые десятичные числа – *DEC (p, s)*;
- числа с плавающей запятой – *FLOAT (p)*;
- числа с плавающей запятой низкой точности – *REAL*;
- числа с плавающей запятой высокой точности – *DOUBLE* ;
- календарная дата – *DATE*;
- время – *TIME*;
- временной интервал – *INTERVAL*;
- денежная величина – *MONEY*;
- логические данные – *LOGICAL*;
- длинный текст – *LONG VARCHAR*;
- поток текстов – *RAW*.

Упрощенный синтаксис команды **SQL**, используемой для создания **базовой структуры таблицы**, имеет следующий вид:

- **CREATE TABLE** <имя таблицы>
- (<имя атрибута 1> <тип [(размер)]> [**DEFAULT** <значение>],
- <имя атрибута 2> <тип [(размер)]> [**DEFAULT** <значение>],
- .....
- .....
- <имя атрибута *n*> <тип [(размер)]> [**DEFAULT** <значение> ] );

Элементы, заключенные в квадратные скобки, являются необязательными.

- Опция *DEFAULT* задает значение по умолчанию, которое должно соответствовать типу данных атрибута. В качестве значения по умолчанию может быть использован литерал, выражение или такая функция SQL, как *SYSDATE*.  
Использование этой опции позволяет исключить запись неопределенного значения в соответствующий столбец таблицы.

Пример. Для пояснения особенностей команды **CREATE TABLE** рассматривается ее применение

для создания таблицы *sotr*,

- содержащей атрибуты, характеризующие деятельность сотрудников университета: *cod* – код сотрудника; *fio* – фамилия, имя, отчество; *dol* – должность; *st* – ученая степень; *sv* – ученое звание; *unag* – учебная нагрузка; *oklad* – должностной оклад; *vosr* – возраст; *pod* – подразделение; *adress* – адрес

регистрации сотрудника

<i>cod</i>	<i>fio</i>	<i>dol</i>	<i>st</i>	<i>sv</i>	<i>unag</i>	<i>oklad</i>	<i>vosr</i>	<i>pod</i>	<i>adress</i>
------------	------------	------------	-----------	-----------	-------------	--------------	-------------	------------	---------------

- Учитывая, что большинство сотрудников зарегистрированы в городе, в котором расположен университет, то при описании атрибута *adress* целесообразно использовать опцию **DEFAULT**:
- ***adress VARCHAR (20) DEFAULT 'НОВОЧЕРКАССК'***.

---

При создании таблиц используются ограничения, проверка которых производится **автоматически СУБД** при реализации каждой из команд манипулирования данными – **включения нового кортежа, изменения элементов данных кортежа**. Ограничения задаются либо на уровне атрибута, либо на уровне таблицы.

- **Ограничение на уровне атрибута имеет следующий синтаксис:**
- <имя атрибута > <тип [(размер)]>  
[**CONSTRAINT** <имя ограничения>] <тип ограничения>.
- **Ограничение на уровне таблицы представляется следующим образом :**
- <имя атрибута 1> <тип [(размер)]>, ...,
- [**CONSTRAINT** <имя ограничения>] <тип ограничения>
- (<имя атрибута 1>, <имя атрибута 2>).

При создании таблицы могут быть заданы следующие ограничения:

- ***NOT NULL*** – указывает на недопустимость неопределенных значений. Атрибуты, не содержащие ограничения ***NOT NULL***, могут иметь неопределенные значения. Это ограничение можно задавать только на **уровне атрибута**:

***fio VARCHAR (15) CONSTRAINT sotr\_fio NOT NULL.***

---

При использовании описания в виде:

***fio VARCHAR (15) NOT NULL***

имя такому ограничению будет присвоено СУБД **автоматически**;

- ***PRIMARY KEY*** – на атрибут или группу атрибутов, образующих первичный ключ. Такое ограничение является единственным для таблицы и обеспечивает контроль уникальности значений атрибутов или комбинации атрибутов, а также контроль за отсутствием неопределенных значений в любой части составного первичного ключа:

***cod NUMBER (4) PRIMARY KEY;***

- ***UNIQUE*** – на атрибут или группу атрибутов, образующих уникальный ключ (***Unique KEY***). В соответствии с этим ограничением никакие две строки таблицы не должны содержать одинаковые значения в указанном атрибуте или группе атрибутов. В случае отсутствия ограничения ***NOT NULL*** у атрибутов, указанных в ограничении ***UNIQUE***, эти атрибуты могут содержать произвольное количество неопределенных значений, т.к. неопределенные значения различаются между собой.
- Ограничения ***UNIQUE*** на уровне атрибута имеет следующий синтаксис:

**<имя атрибута 1> <тип [(размер)]> [CONSTRAINT <имя ограничения>] *UNIQUE*;**

- **CHECK** – ограничение, с помощью которого контролируется выполнение бизнес-правил для каждой строки таблицы. Ограничения **CHECK** устанавливаются для: **диапазона изменения значений атрибута;**  
**принадлежности значений определенному множеству; соответствия шаблону и т.д.**
- Для одного атрибута может быть задано произвольное количество ограничений **CHECK:**

```
col VARCHAR2 (25) CHECK (col IN ('АССИСТЕНТ', 'СТАРШИЙ ПРЕПОДАВАТЕЛЬ', 'ДОЦЕНТ', 'ПРОФЕССОР'))).
```

Для приведенного выше примера команда для создания таблицы *sotr* имеет следующий вид

- ***CREATE TABLE sotr (***  
***cod NUMBER (4) PRIMARY KEY NOT NULL,***  
***fio VARCHAR2 (35) UNIQUE NOT NULL,***  
***dol VARCHAR2 (25) CONSTRAINT sotr\_dol CHECK (dol IN ('АССИСТЕНТ',***  
***'СТАРШИЙ ПРЕПОДАВАТЕЛЬ', 'ДОЦЕНТ', 'ПРОФЕССОР')),***  
***st VARCHAR 2(3) CHECK (st IN ('КТН', 'ДТН', 'НЕТ')),***  
***sv VARCHAR 2(9) CHECK (sv IN ('ДОЦЕНТ', 'ПРОФЕССОР', 'НЕТ')),***  
***unag NUMBER (4),***  
***oklad NUMBER (5,2) CONSTRAINT sotr\_okl CHECK (oklad BETWEEN 7200***  
***AND 25000),***  
***vosr NUMBER (2),***  
***pod VARCHAR 2(25),***  
***adress VARCHAR 2(20) DEFAULT 'НОВОЧЕРКАССК'***  
***);***

С ограничениями можно выполнять следующие операции манипулирования: добавление и удаление; включение и выключение.

Необходимо отметить, что недопустимой является операция изменения ограничений!!!!

**Добавление** ограничения производится с помощью следующей команды:

- ***ALTER TABLE*** <имя таблицы>
- ***ADD [CONSTRAINT*** <имя ограничения> <тип ограничения> (<условие>);

***ALTER TABLE sotr***

***ADD CONSTRAINT sotr\_vosr CHECK (vosr <= 70);***

**Удаление** ограничения производится с помощью следующей команды:

- ***ALTER TABLE*** <имя таблицы> ***DROP CONSTRAINT*** <имя ограничения>;
- ***ALTER TABLE sotr DROP CONSTRAINT sotr\_okl;***

**Отключение ограничения** производится командой *ALTER TABLE* с предложением *DISABLE*:

- *ALTER TABLE* <имя таблицы>  
*DISABLE CONSTRAINT* <имя ограничения>  
[*CASCADE*];

## **Включение ограничения**

осуществляется командой *ALTER TABLE* с предложением *ENABLE*

- *ALTER TABLE* <имя таблицы>  
*ENABLE CONSTRAINT* <имя ограничения>;

# Операции манипулирования со структурой таблиц

- добавление нового атрибута столбца;
- модификация параметров атрибута;
- удаление столбца;
- удаление таблицы;
- переименование таблицы;
- удаление всех строк таблицы;
- добавление комментария к таблице

# Добавление нового атрибута

- ***ALTER TABLE*** <имя таблицы>

***ADD*** <имя атрибута> <тип данных>  
[(размер)] [***DEFAULT*** <значение>];

- ***ALTER TABLE sotr***

***ADD (sv VARCHAR (25) DEFAULT 'ДОЦЕНТ');***

# Модификация параметров атрибута

- ***ALTER TABLE*** <имя таблицы>  
***MODIFY*** <имя атрибута> <тип  
данных>[(размер)][***DEFAULT*** <значение>];
- ***ALTER TABLE sotr MODIFY (dol CHAR (15));***

# Удаление атрибута из таблицы

- ***ALTER TABLE*** <имя таблицы> ***DROP COLUMN*** <имя атрибута>;
- С помощью этой команды нельзя удалить последний столбец/атрибут таблицы. При наличии большого количества строк в таблице удаление столбца приводит к существенным временным затратам. В этом случае имеется возможность с помощью опции *SET UNUSED* пометить как неиспользуемые, а впоследствии эти столбцы удалить с помощью опции *DROP UNUSED COLUMNS*, т.е.
- ***ALTER TABLE*** <имя таблицы>  
***SET UNUSED COLUMN*** <имя атрибута>;
- ***ALTER TABLE*** <имя таблицы>  
***DROP UNUSED COLUMNS***;

Удаление таблицы ***DROP TABLE*** <имя  
таблицы>;

*Переименование таблицы*  
**RENAME** <имя 1> **TO** <имя 2>;

**RENAME** *sotr* **TO** *sotr\_uni*.

*Удаление всех строк таблицы*

**TRUNCATE TABLE** <имя таблицы>;

*Добавление комментариев к таблице*

**COMMENT ON TABLE** <имя таблицы> **IS** '<текст>;

*Добавление комментариев к столбцу*

**COMMENT ON COLUMN** <имя таблицы>. <имя  
атрибута>

**IS** '<текст>;