SQL

SQL является информационно-логическим языком, предназначенным для описания, изменения и извлечения данных, хранимых в реляционной БД.

Каждое предложение SQL – это либо запрос данных из базы, либо обращение к БД, которое приводит к изменению данных в базе.

Типы запросов:

1. Запросы на создание или изменение БД;
2. Запросы на получение данных;
3. Запросы на удаление данных.

Обращение к СУБД

Запросы делятся на:

- оперирующие таблицы;
- оперирующие с отдельными записями.

Преимущества:

1. Независимость от конкретной СУБД;
2. Наличие стандартов;
3. Декларативность (с помощью SQL программист описывает только то, какие данные нужно извлечь или модифицировать).

Недостатки:

1. Сложность (запросы сложные и громоздкие);
2. Отступление от стандартов (многие компании вносят изменения в язык, что способствует модификации языка, которые отступают от общепринятого стандарта).
3. Сложность работы с иерархическими структурами.

Библиотека классов представляет собой коллекцию многократно используемых типов, которые надежно интегрируются с общеязыковой средой выполнения.

После оператора ***select*** пишутся названия тех столбцов, которые необходимо вывести в запросе.

Select фамилия, имя

После оператора from указываются таблицы, через запятую, из которых необходимо выводить столбцы.

Select фамилия, имя
from студент

Оператор ***where*** – оператор условия, после него записываются условия, которому должны удовлетворятся выводимые данные.

Select фамилия
from студент
where Отчество = «Александрович»

Для сортировки выводимых данных существует оператор *ORDER by*, который записывается самой последней строкой в запросе.

Select *
from студент
ORDER by имя

По умолчанию сортировка идет по возрастанию (*ASC*), для сортировки по убыванию используется оператор (*Desc*).

Предикаты – логические: **and, or, not, (и, или, не).**

Также в операторе условия используются скобки, для указания порядка выполнения.

- сравнение:
  >, <, >=, <=, =, ><.

Select фамилия, имя
From студент
where (дата_рождения > 2000) and (имя = «Дмитрий»)

Оператор between используется для выбора диапазона значений.

Select фамилия
from студент
where дата_рождения between 1999 and 2000

Выведем значения, в которых дата рождения в диапазоне от 1999 до 2000.

Select фамилия
from студент
where дата_рождения not between 1999 and 2000

Выводим значения, в которых дата рождения в диапазоне до 1999 и от 2000.

Оператор distinct исключает повторы из результатов запроса.

Select distinct имя
from студент

Предикат – in используется для указания вхождения искомого значения в диапазон (множества значений).

Select фамилия
from студенты
where группа in («П-16», «П-17»)

В данном запросе идет выборка студентов из групп П-16 и П-17.

Переименование столбцов.

Select дата_рождения as возраст
from студент

переименовывает выводимый столбец в указанное значение (возраст).

Оператор **Like**.

Select name
from student
where name Like 'Татьяна'

выводятся значения столбца name = 'Татьяна'
% - любое количество символов. (* MS Access)
_ - 1 символ. (? MS Access)
[a – z] – диапазон символов.
name Like 'a %' – все значения, начинающиеся с буквы а.
name Like 'a_' – все значения, состоящие из двух букв, первая из которых а.
name Like '[a – z]%' – выводит все записи, начинающиеся с маленькой буквы.

Получение итоговых значений(агрегатные функции):

Select min (цена)

from телефон

получение минимального значения цены.

Select max (цена)
from телефон

получение максимального значения цены.

Select avg (цена)
from телефон

получение среднего значения цены.

Select sum (price)
from pc

получение суммы всех значений.

Select count (имя)
from студент

вычисление количества значений.

Вместе со всеми операторами получения итоговых значений, может использоваться оператор GROUP by, который записывается после операторов условия для группировки выводимых значений по дополнительному параметру и имеет вид:

Select имя, count (имя)
from студент
GROUP by имя

Использование в запросе нескольких источников записей

Select *
from студент, группа

Выводятся все столбцы из таблицы

Select *
from студент, группа
where студент.гр=группа.id

Выводятся только те записи из таблицы студент и группа, у которых поле с id группы совпадает.

Для выведения данных из нескольких таблиц, после служебного слова from, указываем необходимые таблицы, через запятую.

Для согласованности выводимых данных, одно из полей таблиц необходимо приравнять.

Явные операции соединения (inner join, right join, left join).

Select *
from студент *inner join*
группа *on* студент.гр=группа.id

Будут выводиться только те записи из таблиц студент и группа, у которых поле с id группы совпадает.

Select*
from студент *left join*
группа *on* студент.гр=группа.id

Left join – это внешнее соединение, означает, что помимо строк, для которых выполняется условие равенства, выведены будут все остальные строки из первой таблицы (левой), при этом отсутствующие записи из правой таблицы, будут заменены null значением.

Select*
From студент *right join*
группа *on* студент.гр=группа.id

Соединение right join обратно соединению left join.

Оператор having.

Если в запросе применяется группировка (group by), то для фильтрации групп по значениям, применяется оператор having.

Select имя, avg (оценка)
From студент
Group by имя
Having avg (оценка) < 4

Одновременное использование нескольких запросов.

Для объединения запросов используется служебное слово **Union**. По умолчанию результат выводится без повторов, чтобы включить в результат повторы используется **Union all**

Select id, фамилия
From студент
Union
Select id, фамилия
From сотрудник

В данном запросе выводится id, фамилии студентов и сотрудников.

Пересечение и разность.

Intersect – пересечение.

Except – разность.

Select фамилия
From студент
Intersect
Select фамилия
From сотрудник

Данный запрос выводит только те фамилии, которые одновременно есть и в таблице студент и в таблице сотрудник.

Select фамилия
From студент
Except
Select фамилия
From сотрудник

Данный запрос выводит только те записи, которые отсутствуют либо в таблице студент, либо в таблице сотрудник.

Операторы объединения выполняются в следующем порядке:

1. **Intersect**
2. **Union, Except.**

Подзапросы:
Использование подзапроса внутри **From**

Select *
From (select *
From студент
Where id>3) as студент2

Where студент2.id<10

Использование подзапроса внутри **Where**

```
Select *
From студент
Where id > (select avg(id)
            From студент)
```

Оператор **Exists**:

Принимает значение истина, если подзапрос содержит любое количество строк, иначе его значение – ложь.

```
Select фамилия
From студент
Where group_st = 1 and exists (select фамилия
                               From студент
                               Where group_st = 2)
```

Данный подзапрос выводит только тех студентов, однофамильцы которых учатся П-17 и УК-17.

Ключевое слово **ANY**, если хотя бы одно (любое) значение из подзапроса удовлетворяет условию, то возвращается истина.

Ключевое слово **ALL**, если все значения из подзапроса удовлетворяет условию, то возвращается истина.

```
Select фамилия
From студент
Where имя = ANY (select имя
From студент)
```

Операторы модификации данных

Оператор **Insert**.

Вставляет новые записи в таблице, при этом значения столбцов могут представлять собой как константы, так и результат выполнения подзапроса.

```
Insert into студент
Values (27, 'Иванов', 'Иван', 'Иванович', 'м', 01.01.2000)
```

Вставляет в таблицу студент id, Фамилию, Имя и Отчество нового студента.

Вставка в одном запросе несколько записей.

Insert into product (фамилия, имя, отчество)
Values (‘Шаповалова’, ‘Анастасия’, ‘Валериевна’),
(‘Горочев’, ‘Владислав’, ‘Игоревич’),
(‘Одинцов’, ‘Тихон’, ‘Владимирович’)

Оператор **Update**:

Меняет уже имеющиеся данные в таблице.

Update студент
Set id = 3
Where пол = ‘ж’

Для записи из таблицы студент, id которых = 3, значение пола меняется на «ж».

Оператор **Delete**.

Удаляет строки из таблиц.

Delete from студент
Where id > 25

Удаляет из таблицы студент все записи, у которых id > 25.