Базы данных и SQL

Семинар 5.

Вопросы?

Вопросы?

Вопросы?

План на сегодня:

- ➔ Викторина
- ➔ Оконные функции
- ➔ Перерыв
- ➔ Представления
- ➔ Домашнее задание

Викторина

Для создания представления, используется команда:

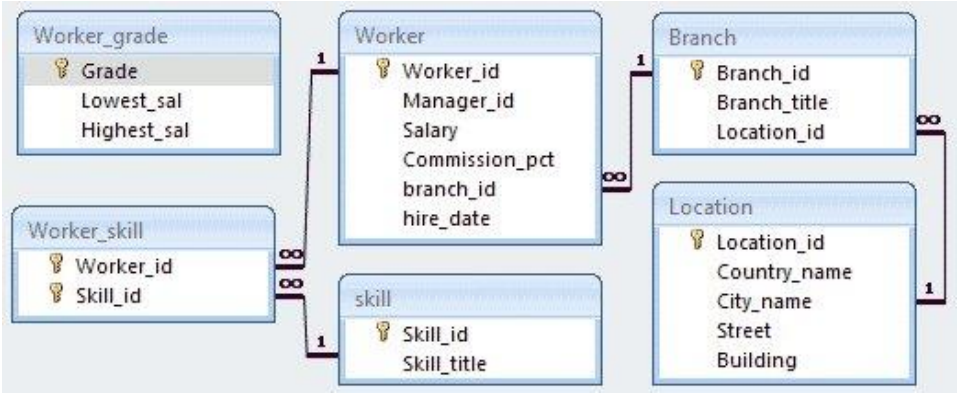
1. CREATE VIRTUAL TABLE
2. CREATE VIEW
3. ALTER VIEW

Для создания представления, используется команда:

1. CREATE VIRTUAL TABLE
2. CREATE VIEW
3. ALTER VIEW

Для создания представления, в которое должны попасть только имена сотрудников, работающих в отделе Research, используется запрос:

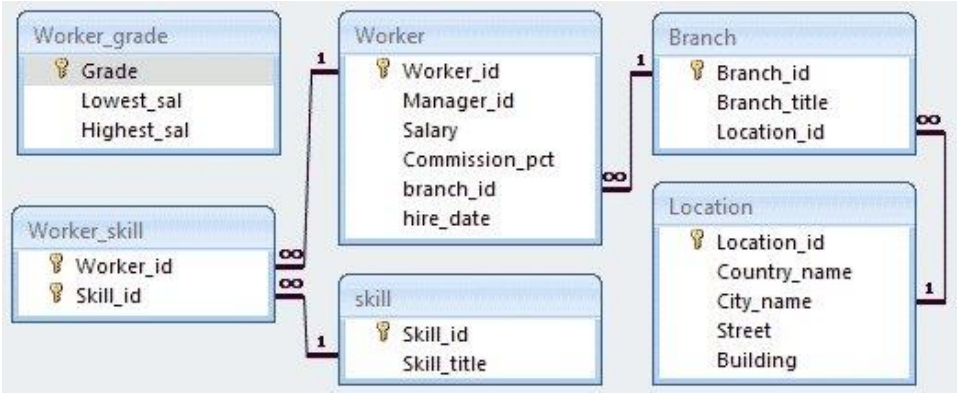
CREATE _____
SELECT Worker_name FROM Worker w, Branch b
WHERE w.Branch_id = b.Branch_id AND Branch_title LIKE ‘Research’



1. VIEW AS
2. view1 AS
3. VIEW view1
4. VIEW view1 AS SUBQUERY
5. VIEW view1 AS

Для создания представления, в которое должны попасть только имена сотрудников, работающих в отделе Research, используется запрос:

```
CREATE _______
SELECT Worker_name FROM Worker w, Branch b
                    WHERE w.Branch_id = b.Branch_id AND Branch_title LIKE
'Research'
```

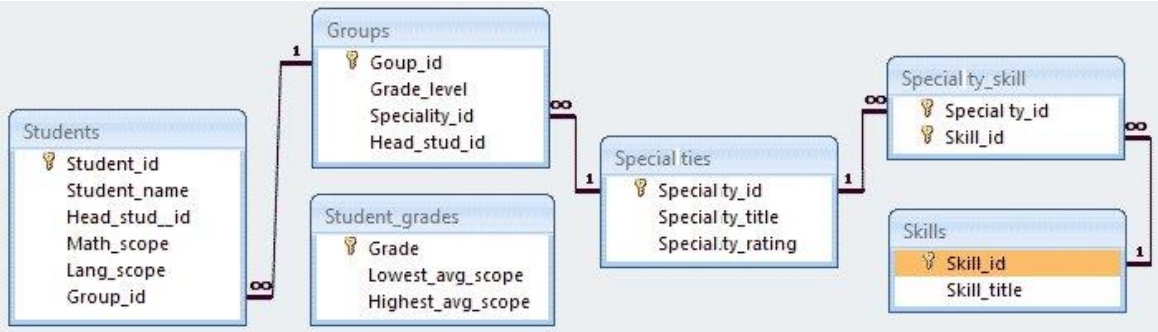


1. VIEW AS
2. view1 AS
3. VIEW view1
4. VIEW view1 AS SUBQUERY
5. VIEW view1 AS

Для создания представления, в которое должны попасть только имена студентов второго курса, используется запрос:

CREATE VIEW view 1

AS.....

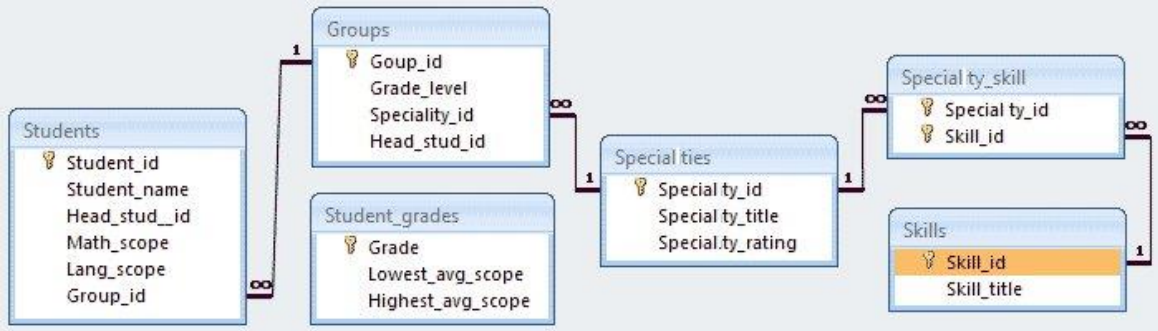


1. (SELECT Student_name FROM Students JOIN Groups ON Students.Group_id = Groups.Group_id)
   WITH CHECK OPTION Grade_level = 2
2. SELECT Student_name FROM Students, Groups WHERE Students.Group_id = Groups.Group_id AND Grade_level = 2
3. (SELECT Student_name FROM Students JOIN Groups ON Students.Group_id = Groups.Group_id AND Grade_level = 2)
4. WITH CHECK OPTION Grade_level=2 (SELECT Student_name FROM Students JOIN Groups ON Students.Group_id = Groups.Group_id)

Для создания представления, в которое должны попасть только имена студентов второго курса, используется запрос:

CREATE VIEW view 1

AS.....



1. (SELECT Student_name FROM Students JOIN Groups ON Students.Group_id = Groups.Group_id) WITH CHECK OPTION Grade_level = 2
2. SELECT Student_name FROM Students, Groups WHERE Students.Group_id = Groups.Group_id AND Grade_level = 2
3. (SELECT Student_name FROM Students JOIN Groups ON Students.Group_id = Groups.Group_id AND Grade_level = 2)
4. WITH CHECK OPTION Grade_level=2 (SELECT Student_name FROM Students JOIN Groups ON Students.Group_id = Groups.Group_id)

В чем заключается главное отличие оконных функций от функций агрегации с группировкой?

1. При использовании агрегирующих функций предложение GROUP BY сокращает количество строк в запросе с помощью их группировки, а при использовании оконных функций количество строк в запросе не уменьшается по сравнении с исходной таблицей.
2. Никакого различия нет
3. При использовании агрегирующих функций предложение GROUP BY не сокращает количество строк в запросе с помощью их группировки, а при использовании оконных функций количество строк в запросе не уменьшается по сравнении с исходной таблицей.

В чем заключается главное отличие оконных функций от функций агрегации с группировкой?

1. При использовании агрегирующих функций предложение GROUP BY сокращает количество строк в запросе с помощью их группировки, а при использовании оконных функций количество строк в запросе не уменьшается по сравнении с исходной таблицей.
2. Никакого различия нет
3. При использовании агрегирующих функций предложение GROUP BY не сокращает количество строк в запросе с помощью их группировки, а при использовании оконных функций количество строк в запросе не уменьшается по сравнении с исходной таблицей.

Что не позволяют делать оконные функции:

1. Ранжирование
2. Редактирование
3. Смещение
4. Агрегация

Что не позволяют делать оконные функции:

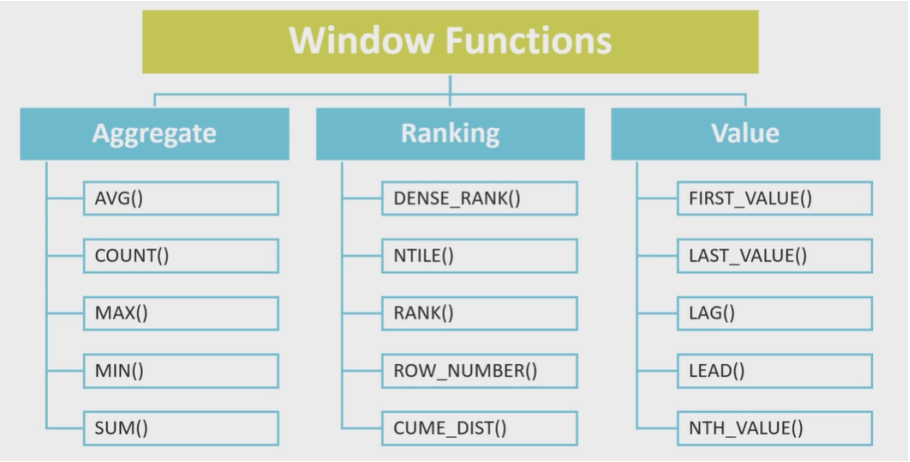
1. Ранжирование
2. Редактирование
3. Смещение
4. Агрегация

Оконные функции

Обычный запрос

Запрос с оконной функцией

SELECT
Название функции (столбец для вычислений)
OVER (
PARTITION BY столбец для группировки
ORDER BY столбец для сортировки
ROWS или RANGE выражение для ограничения строк в пределах группы
)



Примеры.

1. Вывести список всех сотрудников и указать место в рейтинге по зарплатам

```
SELECT DENSE_RANK() OVER (ORDER BY salary DESC) AS rank_salary,
CONCAT(firstname, ' ', lastname), post, salary FROM staff
ORDER BY rank_salary;
```

2. Вывести список всех сотрудников и указать место в рейтинге по зарплатам, но по каждой должности

```
SELECT
DENSE_RANK() OVER (PARTITION BY post ORDER BY salary DESC) AS rank_salary,
CONCAT(firstname, ' ', lastname), post, salary
FROM staff ORDER BY post, rank_salary;
```

3. Найти самых высокооплачиваемых сотрудников по каждой должности

```
SELECT rank_salary, staff, post, salary FROM
(SELECT
DENSE_RANK() OVER (PARTITION BY post ORDER BY salary DESC) AS rank_salary,
CONCAT(firstname, ' ', lastname) AS staff, post, salary FROM staff) AS list
WHERE rank_salary=1
ORDER BY salary DESC;
```

Примеры.

4. Вывести список всех сотрудников, отсортировав по зарплатам в порядке убывания и указать на сколько процентов ЗП меньше, чем у сотрудника со следующей (по значению) зарплатой

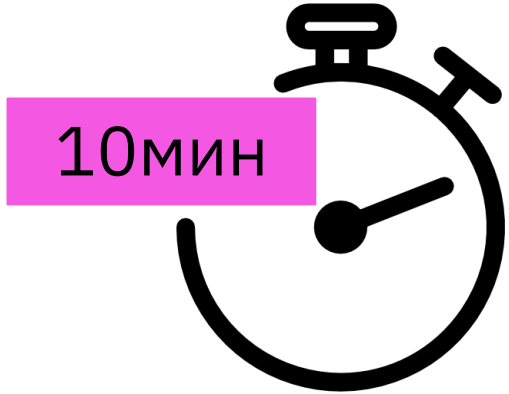
```
SELECT
 id,  CONCAT(firstname, ' ', lastname) AS staff, post, salary,
 LEAD(salary, 1, 0) OVER(ORDER BY salary DESC) AS last_sarary,
 ROUND((salary-LEAD(salary, 1, 0) OVER(ORDER BY salary DESC))*100/salary)
        AS diff_percent
FROM staff;
```

Примеры.

5. Вывести всех сотрудников, отсортировав по зарплатам в рамках каждой должности и рассчитать:
-- общую сумму зарплат для каждой должности
-- процентное соотношение каждой зарплаты от общей суммы по должности
-- среднюю зарплату по каждой должности
-- процентное соотношение каждой зарплаты к средней зарплате по должности
Вывести список всех сотрудников и указать место в рейтинге по зарплатам, но по каждой должности

```
SELECT
 id, CONCAT(firstname, ' ', lastname) AS staff, post, salary,
 SUM(salary) OVER w AS sum_salary,
 ROUND(salary*100/SUM(salary) OVER w) AS percent_sum,
 AVG(salary) OVER w AS avg_salary,
 ROUND(salary*100/AVG(salary) OVER w) AS percent_avg
FROM staff
WINDOW w AS (PARTITION BY post);
```

Задача 1. Получить с помощью оконных функции:



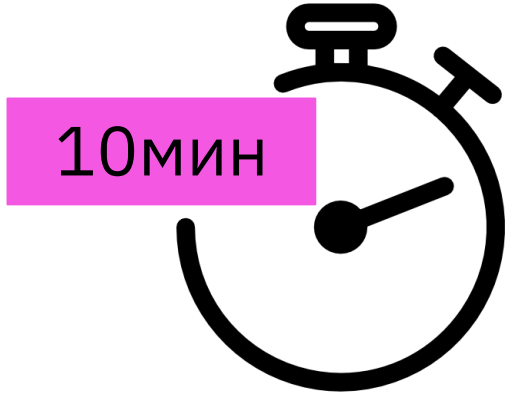
- **средний балл ученика**
- **наименьшую оценку ученика**
- **наибольшую оценку ученика**
- **сумму всех оценок ученика**
- **количество всех оценок ученика**

Решение 1. Получить с помощью оконных функции:

- средний балл ученика
- наименьшую оценку ученика
- наибольшую оценку ученика
- сумму всех оценок
- количество всех оценок

```
SELECT
 name, quartal, subject, grade,
 AVG(grade) OVER w AS avg_grade,
 MIN(grade) OVER w AS min_grade,
 MAX(grade) OVER w AS max_grade,
 SUM(grade) OVER w AS sum_grade,
 COUNT(grade) OVER w AS count_grade
FROM academic_record
WINDOW w AS (PARTITION BY name);
```

Задача 2. Получить информацию об оценках Пети по физике по четвертям:



- **текущая успеваемость**
- **оценка в следующей четверти**
- **оценка в предыдущей четверти**

Решение 2. Получить информацию об оценках Пети по физике по четвертям:

- **текущая успеваемость**
- **оценка в следующей четверти**
- **оценка в предыдущей четверти**

```
SELECT
name, quartal, subject,
grade,
LAG(grade) OVER w AS prev_grade,
LEAD (grade) OVER w AS last_grade
FROM academic_record
WHERE name = 'Петя' AND subject =
'физика'
WINDOW w AS (ORDER BY  quartal);
```

Ваши вопросы?

Перерыв

Временная таблица (TEMPORARY TABLE)

```
CREATE TEMPORARY TABLE new_tbl
SELECT * FROM orig_tbl LIMIT 0;
```

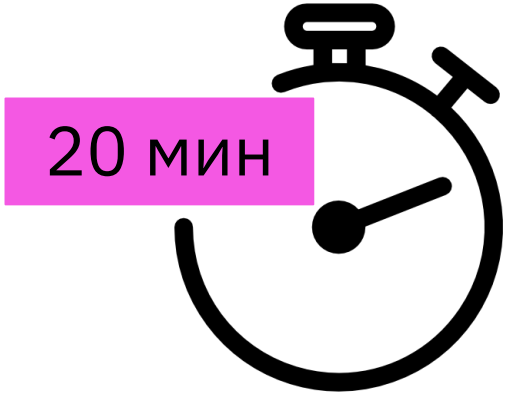
Общее табличное выражение (WITH)

```
WITH
  cte1 AS (SELECT a, b FROM table1),
  cte2 AS (SELECT c, d FROM table2)
SELECT b, d FROM cte1 JOIN cte2
WHERE cte1.a = cte2.c;
```

Представление (VIEW)

```
CREATE OR REPLACE VIEW v_tbl
SELECT * FROM orig_tbl LIMIT 0;
```

Задача 3. Для базы lesson_4 решите :



1. создайте представление, в котором будут выводится все сообщения, в которых принимал участие пользователь с id = 1;

2. найдите друзей у друзей пользователя с id = 1 и поместите выборку в представление;
   (решение задачи с помощью CTE)

3. найдите друзей у друзей пользователя с id = 1.
   (решение задачи с помощью представления “друзья”)

Решение 3. Для базы lesson_4 решите :

1. создайте представление, в котором будут выводится все сообщения, в которых принимал участие пользователь с id = 1;

```
CREATE OR REPLACE VIEW v_messages_user AS
SELECT id, body FROM lesson_4.messages
WHERE from_user_id = 1 -- от пользователя
OR to_user_id = 1; -- к пользователю
```

Решение 3. Для базы lesson_4 решите :

2. найдите друзей у друзей пользователя с id = 1 и поместите выборку в представление; *(решение задачи с помощью CTE)*

```
CREATE OR REPLACE VIEW v_friends_friends AS
WITH friends AS (
    SELECT initiator_user_id AS id
    FROM lesson_4.friend_requests WHERE status = 'approved' AND target_user_id = 1
    UNION
    SELECT target_user_id AS id
    FROM lesson_4.friend_requests WHERE status = 'approved' AND initiator_user_id = 1
)
SELECT fr.initiator_user_id AS friend_id
FROM friends f JOIN lesson_4.friend_requests fr ON fr.target_user_id = f.id
WHERE fr.initiator_user_id != 1  AND fr.status = 'approved'
UNION
SELECT fr.target_user_id
FROM  friends f JOIN lesson_4.friend_requests fr ON fr.initiator_user_id = f.id
WHERE fr.target_user_id != 1  AND status = 'approved';

SELECT friend_id FROM v_friends_friends;
```

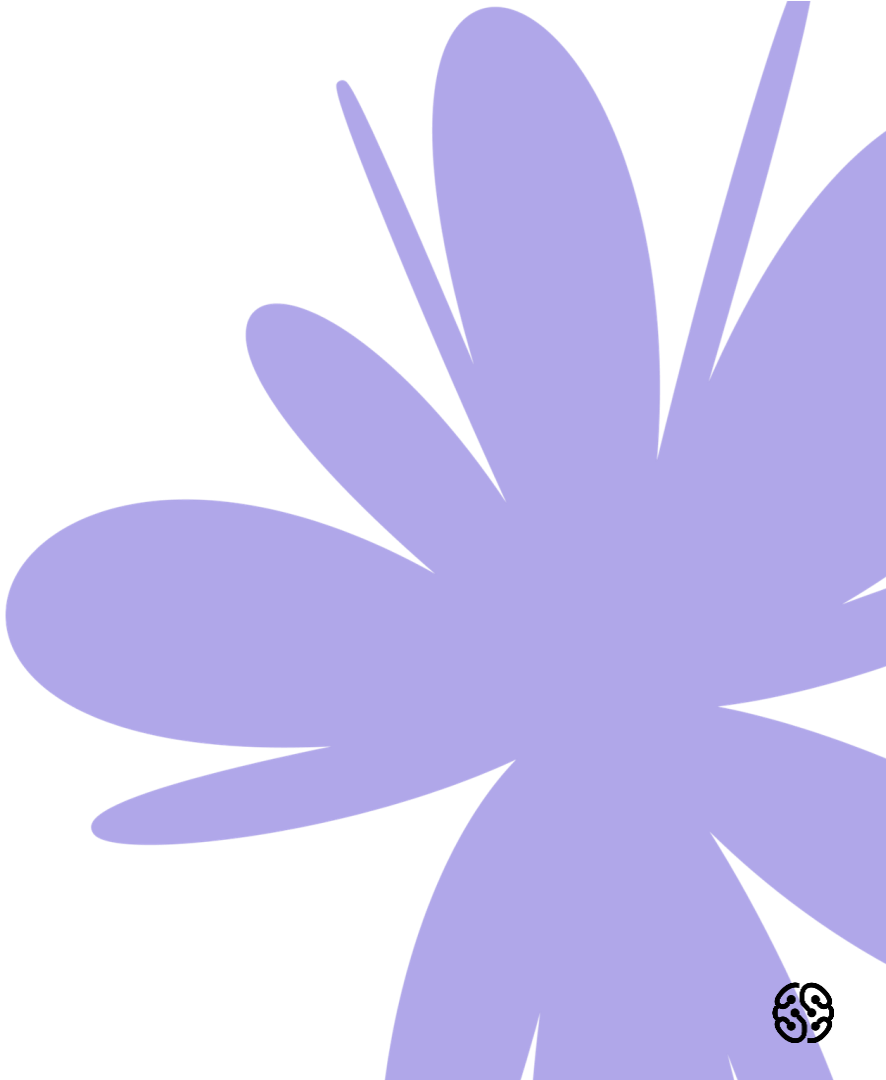
Решение 3. Для базы lesson_4 решите :

3. найдите друзей у друзей пользователя с id = 1.

(решение задачи с помощью представления “друзья”)

```
SELECT fr.initiator_user_id AS friend_id
FROM v_friends f
JOIN lesson_4.friend_requests fr ON fr.target_user_id = f.friend_id
WHERE fr.initiator_user_id != 1 AND f.user_id=1  AND fr.status =
'approved'
UNION
SELECT fr.target_user_id
FROM  v_friends f
JOIN  lesson_4.friend_requests fr ON fr.initiator_user_id = f.friend_id
WHERE fr.target_user_id != 1  AND f.user_id=1 AND  status = 'approved';
```

Ваши вопросы?



Домашнее задание

Для решения задач используйте базу данных lesson_4
(скрипт создания, прикреплен к 4 семинару).

1. Создайте представление, в которое попадет информация о пользователях (имя, фамилия, город и пол), которые не старше 20 лет.

2. Найдите кол-во, отправленных сообщений каждым пользователем и выведите ранжированный список пользователей, указав имя и фамилию пользователя, количество отправленных сообщений и место в рейтинге (первое место у пользователя с максимальным количеством сообщений) . *(используйте DENSE_RANK)*

3. Выберите все сообщения, отсортируйте сообщения по возрастанию даты отправления (created_at) и найдите разницу дат отправления между соседними сообщениями, получившегося списка. *(используйте LEAD или LAG)*

Рефлексия

Был урок полезен вам?

Узнали вы что-то новое?

Что было сложно?

Спасибо за внимание