Ivan Myrotiuk

Task2

1. Вывести список всех рубрик 1 уровня (не имеющих родительских рубрик)

Before:

SELECT \* FROM rubric



Query:

SELECT \* FROM rubric WHERE parent\_rubric IS NULL



1. Вывести список рубрик, упорядоченных по убыванию количества новостей в них (при одинаковом количестве сортировка по имени рубрики)

Before:

SELECT r.name AS rubric, n.name AS header, n.news FROM rubric AS r

JOIN news AS n

ON r.id = n.id\_rubric



Как можем наблюдать к рубрике opera, classical, jaz, hot jaz, operetta относиться несколько новостей

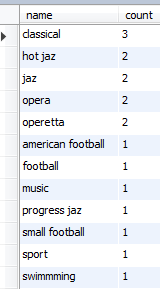
Query:

SELECT rubric.name, COUNT(news.id) AS count FROM rubric

JOIN news ON rubric.id = news.id\_rubric

GROUP BY rubric.name

ORDER BY count(news.id) desc, rubric.name



1. Найти рубрики с наибольшим количеством новостей

SELECT r.name AS rubric, n.name AS header, n.news FROM rubric AS r

JOIN news AS n

ON r.id = n.id\_rubric



Как можем наблюдать только к classical рубрике относиться само больше новостей

Query:

Есть два решения этой задачи :

1)

SELECT rubric.name, COUNT(news.id) AS count

FROM rubric JOIN news ON rubric.id = news.id\_rubric

GROUP BY rubric.name

HAVING COUNT(news.id) = (SELECT MAX(tmp\_table.count) FROM( SELECT rubric.name, COUNT(news.id) AS count

FROM rubric JOIN news ON rubric.id = news.id\_rubric

GROUP BY rubric.name) AS tmp\_table)

2)

SELECT rubric.name, COUNT(news.id) AS count

FROM rubric JOIN news ON rubric.id = news.id\_rubric

GROUP BY rubric.name

ORDER BY count DESC

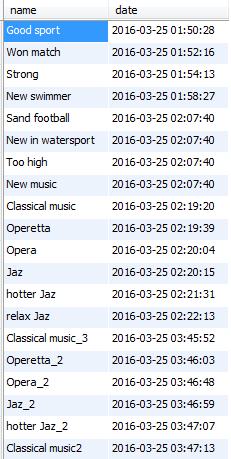
LIMIT 1



1. Вывести 10 самых «свежих» (по дате) новостей.

Before:

SELECT name, date FROM news

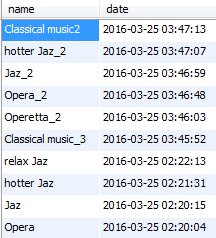


Query:

SELECT name, date FROM news

ORDER BY date DESC

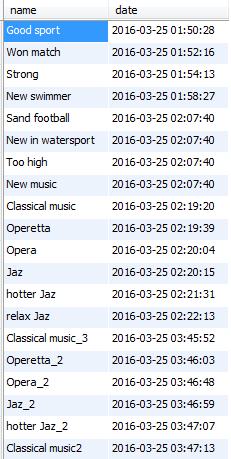
LIMIT 10



1. Для новостей эмулировать пейджинг (вывести «свежие» новости в интервале n по m)

Before:

SELECT name, date FROM news

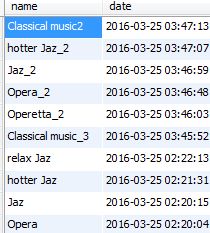


Теперь выведем для наглядности первые 10 свежих новостей

SELECT name, date FROM news

ORDER BY date DESC

LIMIT 10

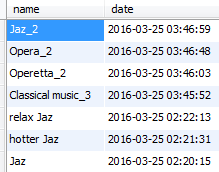


Теперь выведем свежие новости в интервале от 2 до (и следующие) 7:

SELECT name, date FROM news

ORDER BY date DESC

LIMIT 2, 7



1. Подсчитать количество обзоров, не связанных ни с одной новостью.

Выведем новости, которые связаны с обзорами

Before:

SELECT news.name AS news\_name, review.name AS review\_name FROM news

JOIN ref\_news\_review ON ref\_news\_review.id\_news = news.id

JOIN review ON ref\_news\_review.id\_view = review.id



Теперь выведем обзоры, которые не связаны ни с одной новостью

SELECT review.name AS review\_name FROM review

WHERE review.id NOT IN (SELECT id\_view from ref\_news\_review)



Но если мы хотим просто вывести количество, то:

SELECT count(review.name) AS review\_name FROM review

WHERE review.id NOT IN (SELECT id\_view from ref\_news\_review)



1. Вывести заголовки всех новостей, связанных с двумя и более обзорами.

Выведем все новости для наглядности и связанные с ними обзоры

SELECT news.name AS news\_header, review.name AS review\_header FROM news

JOIN ref\_news\_review ON ref\_news\_review.id\_news = news.id

JOIN review ON ref\_news\_review.id\_view = review.id



Теперь выведем новости, которые связаны с 2 и более раз

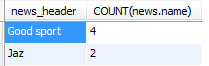
SELECT news.name AS news\_header, COUNT(news.name) FROM news

JOIN ref\_news\_review ON ref\_news\_review.id\_news = news.id

JOIN review ON ref\_news\_review.id\_view = review.id

GROUP BY news\_header

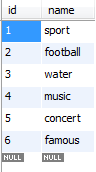
HAVING COUNT(news\_header) >= 2



1. Найти теги (новостей), не связанные с новостями, и новости, не имеющие тегов.

Выведем все теги:

SELECT \* FROM news\_reviews.tag\_news;



Выведем теперь какие теги соответствуют новостям:

SELECT tag\_news.name AS tag, news.name AS header\_news FROM tag\_news

JOIN news\_by\_tag ON tag\_news.id = news\_by\_tag.id\_tag

JOIN news ON news\_by\_tag.id\_news = news.id



Выведем теги которые не связанные с новостями:

SELECT tag\_news.name AS tag\_news\_name FROM tag\_news

WHERE tag\_news.id NOT IN (SELECT news\_by\_tag.id\_tag FROM news\_by\_tag)



Продемонстрируем все новости, которые у нас

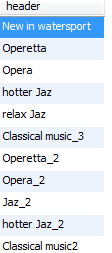
SELECT \* FROM news\_reviews.news; (result of query look next page)



Теперь можем показать какие новости не имеют тегов:

SELECT news.name AS header FROM news

WHERE news.id NOT IN (SELECT news\_by\_tag.id\_news FROM news\_by\_tag)



1. Выбрать новости рубрики А (А – название рубрики) авторов с активностью на сайте выше среднего (активность - количество новостей данного автора).

Самого начала выберем все рубрики и новости с авторами:

SELECT news.name AS header\_news, rubric.name AS rubric\_header, author.name AS author\_name FROM news

JOIN rubric ON news.id\_rubric = rubric.id

JOIN author ON news.id\_author = author.id

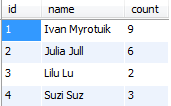


Выберем сколько каждый автор написал новостей

SELECT author.id, author.name, COUNT(news.id\_author) as count FROM author

JOIN news ON author.id = news.id\_author

GROUP BY author.id



Выберем среднее среди этих количеств

SELECT AVG(count\_per\_person.count) AS average FROM (SELECT author.id, author.name, COUNT(news.id\_author) AS count FROM author

JOIN news ON author.id = news.id\_author

GROUP BY author.id) AS count\_per\_person



Теперь выберем новости авторов, которые написали выше среднего

SELECT author.id, author.name, COUNT(news.id\_author) AS count FROM author

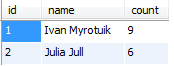
JOIN news ON author.id = news.id\_author

GROUP BY author.id

having count > (SELECT AVG(count\_per\_person.count) AS average FROM (SELECT author.id, author.name, COUNT(news.id\_author) AS count FROM author

JOIN news ON author.id = news.id\_author

GROUP BY author.id) AS count\_per\_person)



Теперь выведем новости популярных авторов из рубрики classical

SELECT news.name AS header\_news FROM news

WHERE news.id\_rubric = (SELECT rubric.id FROM rubric WHERE rubric.name = "classical") AND

news.id\_author IN (SELECT main\_author.authorID FROM (SELECT author.id as authorID, author.name, COUNT(news.id\_author) AS count FROM author

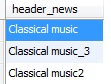
JOIN news ON author.id = news.id\_author

GROUP BY author.id

having count > (SELECT AVG(count\_per\_person.count) AS average FROM (SELECT author.id, author.name, COUNT(news.id\_author) AS count FROM author

JOIN news ON author.id = news.id\_author

GROUP BY author.id) AS count\_per\_person) ) as main\_author)



Добавим для проверки запись к Lilu Lu author

insert into news(id\_author, id\_rubric, name, news, video\_link) values(3, 8, "Classical music Lilu", "Lilu for music..","www.news.com/music");

Проделаем все опять:

SELECT news.name AS header\_news, rubric.name AS rubric\_header, author.name AS author\_name FROM news

JOIN rubric ON news.id\_rubric = rubric.id

JOIN author ON news.id\_author = author.id



SELECT author.id, author.name, COUNT(news.id\_author) as count FROM author

JOIN news ON author.id = news.id\_author

GROUP BY author.id



SELECT author.id, author.name, COUNT(news.id\_author) AS count FROM author

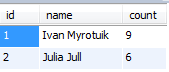
JOIN news ON author.id = news.id\_author

GROUP BY author.id

having count > (SELECT AVG(count\_per\_person.count) AS average FROM (SELECT author.id, author.name, COUNT(news.id\_author) AS count FROM author

JOIN news ON author.id = news.id\_author

GROUP BY author.id) AS count\_per\_person)



SELECT news.name AS header\_news FROM news

WHERE news.id\_rubric = (SELECT rubric.id FROM rubric WHERE rubric.name = "classical") AND

news.id\_author IN (SELECT main\_author.authorID FROM (SELECT author.id as authorID, author.name, COUNT(news.id\_author) AS count FROM author

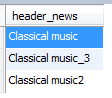
JOIN news ON author.id = news.id\_author

GROUP BY author.id

having count > (SELECT AVG(count\_per\_person.count) AS average FROM (SELECT author.id, author.name, COUNT(news.id\_author) AS count FROM author

JOIN news ON author.id = news.id\_author

GROUP BY author.id) AS count\_per\_person) ) as main\_author)



Как можем наблюдать выбрались только новости популярных авторов

А новости от Lilu не выбрались так как количество новостей у нее меньше чем средние количество всех новостей от всех авторов.

1. Перенести все обзоры из некоторой указанной подрубрики с названием A в некоторую указанную подрубрику B.