Тест на знание SQL

Что такое реляционные базы данных:

- ○ База данных, в которой информация хранится в виде двумерных таблиц, связанных между собой
- ○ База данных, в которой одна ни с чем не связанная таблица
- ○ Любая база данных - реляционная
- ○ Совокупность данных, не связанных между собой

Как выглядит запрос, для вывода ВСЕХ значений из таблицы Orders:

- select ALL from Orders;
- select % from Orders;
- select * from Orders;
- select *.Orders from Orders;

Какие данные мы получим из этого запроса?
select id, date, customer_name from Orders;

- ○ Неотсортированные номера и даты всех заказов с именами заказчиков
- ○ Никакие, запрос составлен неверно
- ○ Номера и даты всех заказов с именами заказчиков, отсортированные по первой колонке
- ○ Номера и даты всех заказов с именами заказчиков, отсортированные по всем колонкам, содержащим слово Order

Есть ли ошибка в запросе?
select id, date, customer_name from Orders where customer_name = Mike;

- ○ Запрос составлен правильно
- ○ Mike необходимо записать в кавычках 'Mike'
- ○ Нужно убрать лишние поля из запроса
- ○ Строчку с where поменять местами с from

Что покажет следующий запрос:
select * from Orders where date between '2017-01-01' and '2017-12-31'

- ○ Все данные по заказам, совершенным за 2017 год, за исключением 01 января 2017 года
- ○ Все данные по заказам, совершенным за 2017 год, за исключением 31 декабря 2017 года
- ○ Все данные по заказам, совершенным за 2017 год
- ○ Ничего, запрос составлен неверно

Что не так с этим запросом
select id, date from Orders where seller_id = NULL;

- ○ Все верно, запрос покажет все заказы, продавцы которых не проставлены
- ○ NULL нужно взять в кавычки
- ○ Сравнение с NULL можно проводить только с оператором IS
- ○ Сравнение с NULL можно проводить только с оператором ON

Порядок выполнения операторов AND и OR следующий:

- ○ Сначала выполняется AND, а затем OR
- ○ Сначала выполняется OR, а затем AND
- ○ Порядок выполнения операторов AND и OR зависит от того, какой операторов стоит первым
- ○ Операторы AND и OR выполняются одновременно

Что покажет следующий запрос: select DISTINCT seller_id order by seller_id from Orders;

- Уникальные ID продавцов, отсортированные по возрастанию
- Уникальные ID продавцов, отсортированные по убыванию
- Ничего, запрос составлен неверно, ORDER BY всегда ставится в конце запроса
- Неотсортированные никак уникальные ID продавцов

Что делает спецсимвол '_' в паре с оператором LIKE: select * from Orders where customer_name like 'mik_';

- найдет все имена, которые начинаются на mik и состоят из 4 символов
- найдет все имена, которые начинаются на mik, вне зависимости от того, из какого количества символов они состоят
- найдет данные, где имя равно mik
- запрос составлен неверно, в паре с оператором like не используются спецсимволы

Выберите корректный пример использования функции CONCAT:

- select concat = index and city from Orders;
- select concat IN (`index`, `city`) from Orders;
- select concat(`index`," ", `city`) from Orders;
- нет правильного примера

Что покажет следующий запрос:
select concat(`index`," ", `city`) AS delivery_address from Orders;

- ничего, запрос составлен неверно
- покажет уникальные значения индексов и адресов из таблицы Orders
- соединит поля с индексом и адресом из таблицы Orders и покажет их с псевдонимом delivery_address
- соединит поля с индексом и адресом из таблицы Orders, но покажет их без псевдонима

Выберите правильный пример использования функции округления ROUND

- select id, price * discount AS total price from Orders ROUND (2);
- select id, price * discount ROUND (2) AS total price from Orders;
- select id, ROUND (price * discount, 2) AS total price from Orders;
- нет правильного примера

Что покажет следующий запрос: select id from Orders where year (date) > 2018;

- номера заказов, сделанных до 2018 года
- номера заказов, сделанных в 2018 году
- уникальные номера заказов
- номера заказов, сделанных после 2018 года

Для чего используется LIMIT: select * from Orders limit 10;

- ○ необходим, чтобы показать все заказы, содержащие цифру 10
- ○ необходим, чтобы показать первых 10 записей в запросе
- ○ необходим, чтобы показать рандомные 10 записей в запрос
- ○ не существует такого оператора

Что такое агрегирующие функции:

- ○ функции, которые фильтруют значения
- ○ функции, которые сортируют значения
- ○ функции, которые работают с набором данных, превращая их в одно итоговое значение
- ○ функции, которые суммируют все значения

Выберите пример правильно составленного запроса с использованием агрегирующей функции SUM:

- select sum(price) from Orders;
- select sum(price), customer_name from Orders;
- select * from Orders where price=sum();
- select sum() from Orders group by price desc;

Возможно ли использование одновременно двух агрегирующих функций: select min(price), max(price) from Orders;

- да, но данный запрос составлен неверно, надо так: select * from Orders where price IN (min, max);
- да, в результате мы получим минимальную и максимальную стоимости
- да, в результате мы получим стоимости, отсортированные от минимальной к максимальной
- нет, две функции использовать одновременно нельзя

Выберите корректно составленный запрос с функцией GROUP BY:

- select count(*) from Orders GROUP seller_id;
- select seller_id, count(*) from Orders GROUP seller_id;
- select seller_id, count(*) from Orders GROUP BY seller_id;
- select count(*) from Orders GROUP ON seller_id;

Что покажет следующий запрос: select seller_id, count(*) from Orders GROUP BY seller_id HAVING seller_id IN (2,4,6);

- количество заказов сгруппированное по продавцам 2, 4 и 6
- количество продавцов, у которых 2, 4 или 6 товаров
- ничего, запрос составлен неверно, HAVING указывается до группировки
- ничего, запрос составлен неверно, для указания условия должно быть использовано WHERE

Выберите пример корректно написанного запроса с использованием подзапроса, который выводит информацию о заказе с самой дорогой стоимостью:

- select * from Orders where price = (select big(price) from Orders)
- select * from Orders where price = max
- select count(*) from Orders
- select * from Orders where price = (select max(price) from Orders)

Что такое JOIN:

- ○ операция объединения
- ○ операция группировки
- ○ операция суммирования
- ○ операция создания

Какого из перечисленных ниже видов JOIN на самом деле не существует:

- LEFT JOIN - который выведет все записи первой таблицы, а для ненайденных пар из правой таблицы проставит значение NULL
- RIGHT JOIN - который выведет все записи второй таблицы, а на место недостающей информации из первой таблицы проставит NULL
- INNER JOIN - который показывает только те записи, для которых нашлись пары
- TRUE JOIN - который выведет все верные значения

Выберите корректный пример составленного запроса с использованием JOIN. Данный запрос выведет нам данные ID заказа, имя заказчика и продавца:

- select Orders.id, Orders.customer_name, Sellers.id from Orders LEFT JOIN ON Sellers AND Orders.seller_id = Sellers.id;
- select id AND customer_name AND seller_id from Orders LEFT JOIN Sellers ON seller_id = id;
- select Orders.id, Orders.customer_name, Sellers.id from Orders LEFT JOIN Sellers ON Orders.seller_id = Sellers.id;
- select Orders.id, Orders.customer_name, Sellers.id from Orders JOIN Sellers WHEN Orders.seller_id = Sellers.id;

Выберите правильный пример запроса с использованием UNION:

- select id, city from Orders order by id union select id, city from Sellers order by city;
- select id, city, seller_id from Orders and select city, id from Sellers order by id;
- select id, city from Orders union select id, city from Sellers order by id;
- Все запросы верные

Какого строкового типа данных нет в SQL:

- VARCHAR
- STRING
- CHAR
- TEXT

Чем отличается CHAR и VARCHAR?

- ○ Это одно и то же
- ○ VARCHAR не существует
- ○ CHAR - это тип данных, а VARCHAR - подтип
- ○ CHAR дополняет строку пробелами до максимальной длины, а VARCHAR тратит лишнюю память на хранение значения длины строки

Как получить значение текущего года в SQL?

- select now();
- select year();
- select year(now());
- select year from Date;

Как правильно добавить строку в таблицу? Какой запрос верный?

- INSERT INTO `SimpleTable` (`some_text`) VALUES ("my text");
- INSERT INTO `SimpleTable` SET `some_text`="my text";
- SET INTO `SimpleTable` VALUE `some_text`="my text";
- UPDATE INTO `SimpleTable` SET `some_text`="my text";

Какие поля из таблицы обязательно перечислять в INSERT для вставки данных?

- Конечно все
- Только те, у которых нет DEFAULT значения
- Те, у которых нет DEFAULT значения и которые не имеют атрибут auto_increment
- Все поля имеют негласное DEFAULT значения, обязательных полей в SQL нет

Как сделать несколько записей в таблицу за один запрос?

- ○ Использовать MULTI INSERT INTO вместо INSERT INTO
- ○ Использовать подзапрос
- ○ Перечислить через запятую все наборы значений после VALUES
- ○ Никак, расходимся по домам

Зачем существует команда UPDATE, если можно сначала удалить запись, а потом добавить новую, исправленную.

- Именно так и делаю, UPDATE не использую
- Так меньше нагрузки на базу, ведь команда одна, а не две
- Потому что в записи могут быть автоматически проставляемые поля, такие как auto_increment или timestamp, которые собьются при внесении записи заново
- Как раз удалять записи в SQL нельзя, вместо этого используется UPDATE с NULL-значениями для всех полей

В каких командах можно использовать LIMIT?

- ○ Только Select
- ○ Select и Insert
- ○ Select, Update, Delete
- ○ Select, Insert, Delete, Update

Как можно заранее узнать, какие записи будут удалены при выполнении DELETE?

- Зачем заранее, просто вызвать его и посмотреть какие записи пропали
- Заменить DELETE на SELECT *, ведь в остальном синтаксис DELETE похож на синтаксис простого SELECT
- Сделать DELETE с LIMIT 1, одну запись не жалко
- SQL создан для хранения данных, их нельзя удалять

Какой командой можно создать новую таблицу?

- CREATE TABLE
- MAKE TABLE
- SET TABLE
- Создавать таблицы можно только через интерфейс СУБД, специальной SQL команды для этого нет

Можно ли поменять тип данных поля в уже существующей таблице?

- ○ Да, при помощи команды ALTER
- ○ Да, достаточно сделать INSERT с новым типом данных
- ○ Нет, только пересоздать таблицу
- ○ Тип бывает только у таблицы, а не у поля таблицы

Спасибо за внимание!