**Задача 1.** Описание работы запроса:

1. Из таблицы Sales выбираются столбцы Id и YEAR(creationdate).

2. "YEAR(creationdate)" - это выражение, которое вычисляет год из даты создания ("creationdate") продажи. Функция YEAR() возвращает только год из указанной даты.

3. Псевдоним SaleDate задается для столбца YEAR(creationdate), чтобы в результате запроса это поле называлось SaleDate.

4. В условии WHERE указано, что SaleDate (год создания) должен быть больше или равен 2021.

5. Результат сортируется по возрастанию SaleDate.

Таким образом, запрос выберет ID записей и год их создания (под псевдонимом SaleDate) для тех, которые были созданы в 2021 году и позже. Результат будет отсортирован по возрастанию SaleDate.

**Задача 2.** Вывод ФИО пациента, данных последнего активного полиса:

SELECT

m.fullname,

p.series,

p.number,

p.issuedate,

p.enddate

FROM medcards m

JOIN polises p ON m.mcid = p.mcid

WHERE p.enddate > CURRENT\_DATE

ORDER BY p.issuedate DESC

LIMIT 1

Здесь используется JOIN для объединения таблиц medcards и polises. В условии WHERE ограничиваем поиск активных полисов, у которых enddate больше текущей даты. Сортируем по убыванию даты выдачи полиса и ограничиваем 1 строкой, чтобы получить самый последний полис.

**Задача 3.**

SELECT

CASE

WHEN REGEXP\_INSTR(your\_string, '[0-9][^a-zA-Z]|[0-9]$') > 0 THEN

REGEXP\_INSTR(your\_string, '[0-9][^a-zA-Z]|[0-9]$')

ELSE NULL

END AS index\_of\_first\_number

FROM your\_table;

**Задача 4.**

SELECT fullname

FROM medcards

ORDER BY LENGTH(fullname) DESC

LIMIT 1;

В обоих запросах мы выбираем столбец "fullname" из таблицы "medcards".

С помощью функции "LENGTH" мы определяем длину имени каждого пациента.

Затем, используя "ORDER BY LENGTH(fullname) DESC", мы сортируем записи по убыванию длины имени, чтобы самое длинное имя было первым.

**Задача 5**

DELETE FROM medcards m1

JOIN medcards m2

ON m1.fullname = m2.fullname

AND m1.birthdate = m2.birthdate

AND m1.phone = m2.phone

WHERE m1.creationdate > m2.creationdate;

В запросе используется синтаксис соединения двух экземпляров таблицы "medcards" с помощью JOIN по комбинации ФИО, даты рождения и номера телефона (fullname, birthdate, phone).

В условии "WHERE" выбираются только те записи (m1), для которых существуют другие записи (m2) с тем же ФИО, датой рождения и номером телефона, но с более поздней датой создания (m1.creationdate > m2.creationdate). Это означает, что будут удалены записи, которые имеют более позднюю дату создания и являются дубликатами.

В результате, запрос удалит все продублированные медкарты, оставив только записи с наименьшей датой создания для каждой комбинации ФИО, даты рождения и номера телефона.

**Задача 6**

UPDATE Services AS s

JOIN #TempTable AS t ON s.code = t.code

SET s.basecost = t.basecost,

s.purchasePrice = t.purchasePrice;

1. В запросе мы используем оператор UPDATE для обновления данных в таблице Services.

2. Мы создаем псевдонимы для таблиц Services (s) и временной таблицы #TempTable (t), чтобы обращаться к ним с помощью кратких имен в запросе.

3. В операторе JOIN мы связываем таблицу Services с временной таблицей #TempTable по полю "code", чтобы найти совпадающие услуги.

4. С помощью SET мы обновляем поля "basecost" и "purchasePrice" в таблице Services значениями из соответствующих полей во временной таблице #TempTable.

Таким образом, после выполнения этого запроса базовая стоимость и цена закупки в таблице Services будут обновлены на значения из временной таблицы #TempTable для соответствующих услуг, у которых есть совпадение по коду услуги.

**Задача 7**

Вывод суммы продаж с накоплением:

SELECT

SaleDate,

SUM(SaleSum) OVER (ORDER BY SaleDate) AS TotalSales

FROM Sales

Используем оконную функцию SUM() OVER для подсчета накопленной суммы по мере сортировки по дате.

**Задача 8**

SELECT

fullname,

birthdate

FROM medcards

WHERE

(MONTH(birthdate) = 2 AND DAY(birthdate) BETWEEN 10 AND 28)

OR (

MONTH(birthdate) = 2

AND DAY(birthdate) = 29

AND (

(YEAR(GETDATE()) % 4 = 0 AND YEAR(GETDATE()) % 100 != 0)

OR YEAR(GETDATE()) % 400 = 0

)

)

1. Нам нужно отфильтровать записи в таблице medcards по дате рождения, которая попадает в заданный промежуток с 10 по 28 февраля.

2. Для этого мы проверяем, что месяц рождения (MONTH(birthdate)) равен 2, то есть февраль.

3. Затем для обычных дней рождения (с 10 по 28 число) мы используем условие, что день рождения (DAY(birthdate)) находится между 10 и 28 с помощью оператора BETWEEN.

4. Остается учесть 29 февраль для високосных годов. Для этого мы добавляем еще одно условие OR, где проверяем, что день рождения - 29, а год является високосным.

5. Чтобы определить високосный год, используем правило: год високосный, если он делится на 4, но не делится на 100, либо делится на 400. Это мы проверяем через остаток от деления и функцию GETDATE(), чтобы учесть текущий год.

6. В результате получаем записи, которые удовлетворяют либо первому условию для обычных дней рождения, либо второму условию для 29 февраля високосных годов.

**Задача 9**

SELECT

A.account AS AccountNumber,

M.fullname AS PatientName,

STRING\_AGG(S.service\_name, ', ') AS Services

FROM

Accounts AS A

JOIN Medcards AS M ON A.mcid = M.mcid

JOIN Accounts\_contents AS AC ON A.account = AC.account

JOIN Services AS S ON AC.service = S.service\_id

WHERE

A.account = (

SELECT MAX(account)

FROM Accounts

)

GROUP BY

A.account,

M.fullname

1. Нам нужно получить максимальный (последний) номер счета из таблицы Accounts. Это можно сделать с помощью вложенного запроса, который выберет максимальное значение столбца account.

2. Затем необходимо узнать ФИО пациента, которому принадлежит этот счет. Для этого используем JOIN таблицы Accounts с таблицей Medcards по столбцу-ключу mcid.

3. Далее нужно получить список услуг, начисленных на данный счет. Делаем еще один JOIN - на этот раз таблицы Accounts\_contents, чтобы получить ID услуг.

4. И последний JOIN - с таблицей Services по ID услуги, чтобы узнать названия этих услуг.

5. В итоге, чтобы сконкатенировать названия услуг в одну строку через запятую, используем агрегатную функцию STRING\_AGG.

6. И в конце группируем результат по номеру счета и ФИО, чтобы получить нужный вывод для каждого счета.

Таким образом, с помощью вложенного запроса, нескольких JOIN и агрегирования строится запрос, который выводит требуемые данные по последнему счету пациента.

**Задача 10**

SELECT

PatientFullname,

YEAR(SaleDate) as Year,

MAX(Summa) as MaxSumma,

MIN(Summa) as MinSumma

FROM Sales

WHERE

YEAR(SaleDate) IN (2019, 2020)

GROUP BY

PatientFullname,

YEAR(SaleDate)