Оглавление

[**Часть 1** 3](#_Toc211862278)

[***1. Выборка из одной таблицы.*** 3](#_Toc211862279)

[1.1 Выбрать из произвольной таблицы данные и отсортировать их по двум произвольным имеющимся в таблице признакам (разные направления сортировки). 3](#_Toc211862280)

[1.2 Выбрать из произвольной таблицы те записи, которые удовлетворяют условию отбора (where). Привести 2–3 запроса. 3](#_Toc211862281)

[1.3 Привести примеры 2–3 запросов с использованием агрегатных функций (count, max, sum и др.) с группировкой и без группировки. 4](#_Toc211862282)

[1.4 Привести примеры подведения подытога с использованием GROUP BY [ALL] [CUBE | ROLLUP] (2–3 запроса). В ROLLUP и CUBE использовать не менее 2-х столбцов. 5](#_Toc211862283)

[1.5 Выбрать из таблиц информацию об объектах, в названиях которых нет заданной последовательности букв (LIKE). 7](#_Toc211862284)

[***2. Выборка из нескольких таблиц.*** 8](#_Toc211862285)

[2.1 Вывести информацию подчиненной (дочерней) таблицы, заменяя коды (значения внешних ключей) соответствующими символьными значениями из родительских таблиц. Привести 2–3 запроса с использованием классического подхода соединения таблиц (where). 8](#_Toc211862286)

[2.2. Реализовать запросы пункта 2.1 через внутреннее соединение inner join. 11](#_Toc211862287)

[2.3. Левое внешнее соединение left join. Привести 2–3 запроса. 14](#_Toc211862288)

[2.4. Правое внешнее соединение right join. Привести 2–3 запроса. 16](#_Toc211862289)

[2.5. Привести примеры 2–3 запросов с использованием агрегатных функций и группировки. 18](#_Toc211862290)

[2.6. Привести примеры 2–3 запросов с использованием группировки и условия отбора групп (Having). …………………………………………………………………………………………………………………19](#_Toc211862291)

[2.7. Привести примеры 3–4 вложенных (соотнесенных, c использованием IN, EXISTS) запросов. …………………………………………………………………………………………………………………20](#_Toc211862292)

[***3. Представления.*** 21](#_Toc211862293)

[3.1 На основе любых запросов из п. 2 создать два представления (VIEW). 21](#_Toc211862294)

[3.2 Привести примеры использования общетабличных выражений (СТЕ) (2–3 запроса). 23](#_Toc211862295)

[***4. Функции ранжирования.*** 26](#_Toc211862296)

[4.1 Привести примеры 3–4 запросов с использованием ROW\_NUMBER, RANK, DENSE\_RANK (c PARTITION BY и без). 26](#_Toc211862297)

[***5. Объединение, пересечение, разность.*** 30](#_Toc211862298)

[5.1 Привести примеры 3–4 запросов с использованием UNION / UNION ALL, EXCEPT, INTERSECT. Данные в одном из запросов отсортируйте по произвольному признаку. 30](#_Toc211862299)

[***6. Использование CASE, PIVOT и UNPIVOT*** 33](#_Toc211862300)

[6.1 Привести примеры получения сводных (итоговых) таблиц с использованием CASE. 33](#_Toc211862301)

[6.2 Привести примеры получения сводных (итоговых) таблиц с использованием PIVOT и UNPIVOT. ………….……………………………………………………………………………………………………35](#_Toc211862302)

[**Часть 2** 39](#_Toc211862303)

[***1. Найти номера, не занятые и не забронированные на данный момент:*** 39](#_Toc211862304)

[***2. Найти клиентов, выехавших из номера раньше срока, указанного в договоре*** 40](#_Toc211862305)

[***3. Найти номера, которые ни разу не сдавались с начала текущего года:*** 42](#_Toc211862306)

[***4. Найти клиентов, забронировавших номер, но так и не въехавших в него:*** 43](#_Toc211862307)

[***5. Среди всех клиентов, наиболее часто пользующихся услугами гостиницы, найти клиентов с максимальным сроком проживания:*** 44](#_Toc211862308)

# **Часть 1**

Цель: изучить конструкции языка SQL для манипулирования данными в СУБД MSSQL.

Задания и краткое описание работы:

## ***1. Выборка из одной таблицы.***

### 1.1 Выбрать из произвольной таблицы данные и отсортировать их по двум произвольным имеющимся в таблице признакам (разные направления сортировки).

Сортировка по Статусу в порядке возрастания, по Предварительной\_стоимости в порядке убывания:

SELECT \*

FROM Бронирование

ORDER BY Статус ASC, Предварительная\_стоимость DESC;

Изображение выглядит как текст, снимок экрана, число, Параллельный

Содержимое, созданное искусственным интеллектом, может быть неверным.

### 1.2 Выбрать из произвольной таблицы те записи, которые удовлетворяют условию отбора (where). Привести 2–3 запроса.

1. Таблица Клиент — выбрать всех женщин:

SELECT \*

FROM Клиент

WHERE Пол = 'Ж';

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Таблица Номер — выбрать все номера дороже 4000 рублей за сутки:

SELECT \*

FROM Номер

WHERE Цена\_сутки > 4000;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Таблица Бронирование — выбрать все активные (созданные) брони, начиная с сентября 2025:

SELECT \*

FROM Бронирование

WHERE Статус = 'создана'

AND Дата\_предположительного\_заселения >= '2025-09-01';

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

### 1.3 Привести примеры 2–3 запросов с использованием агрегатных функций (count, max, sum и др.) с группировкой и без группировки.

Без группировки

1. Посчитать количество клиентов в базе:

SELECT COUNT(\*) AS Количество\_клиентов

FROM Клиент;



1. Посчитать общую сумму оплат по всем заселениям:

SELECT SUM(Сумма) AS Общая\_сумма\_оплат

FROM Оплата;



1. Найти максимальную цену за сутки среди всех номеров:

SELECT MAX(Цена\_сутки) AS Максимальная\_цена\_номера

FROM Номер;



С группировкой

1. Посчитать количество бронирований по каждому статусу:

SELECT Статус, COUNT(\*) AS Количество

FROM Бронирование

GROUP BY Статус;

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Сумма оплат по каждому способу оплаты:

SELECT Способ\_оплаты, SUM(Сумма) AS Общая\_сумма

FROM Оплата

GROUP BY Способ\_оплаты;

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Найти среднюю стоимость номера по каждой комфортности:

SELECT Комфортность, AVG(Цена\_сутки) AS Средняя\_цена

FROM Номер

GROUP BY Комфортность;

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

### 1.4 Привести примеры подведения подытога с использованием GROUP BY [ALL] [CUBE | ROLLUP] (2–3 запроса). В ROLLUP и CUBE использовать не менее 2-х столбцов.

1. GROUP BY ALL

Подсчёт количества клиентов по полу:

SELECT Пол, COUNT(\*) AS Количество

FROM Клиент

GROUP BY ALL Пол;

Изображение выглядит как снимок экрана, текст, Шрифт, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. ROLLUP

Количество бронирований по клиентам и статусу:  
SELECT Клиент\_ID, Статус, COUNT(\*) AS Количество

FROM Бронирование

GROUP BY ROLLUP (Клиент\_ID, Статус);

Изображение выглядит как текст, снимок экрана, число, Параллельный

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. CUBE

Сумма оплат по способу оплаты и клиенту:

SELECT Заселение\_ID, Способ\_оплаты, SUM(Сумма) AS Общая\_сумма

FROM Оплата

GROUP BY CUBE (Заселение\_ID, Способ\_оплаты);

Изображение выглядит как текст, снимок экрана, число, Параллельный

Содержимое, созданное искусственным интеллектом, может быть неверным.

### 1.5 Выбрать из таблиц информацию об объектах, в названиях которых нет заданной последовательности букв (LIKE).

1. Услуги, в названии которых нет буквы "а":

SELECT \*

FROM Услуга

WHERE Название NOT LIKE '%а%';

Изображение выглядит как текст, Шрифт, снимок экрана, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Клиенты, у которых ФИО не содержит подстроку "ов":

SELECT \*

FROM Клиент

WHERE ФИО NOT LIKE '%ов%';

Изображение выглядит как текст, снимок экрана, линия, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

## ***2. Выборка из нескольких таблиц.***

### 2.1 Вывести информацию подчиненной (дочерней) таблицы, заменяя коды (значения внешних ключей) соответствующими символьными значениями из родительских таблиц. Привести 2–3 запроса с использованием классического подхода соединения таблиц (where).

1. Показать все бронирования с ФИО клиента (вместо Клиент\_ID):

SELECT

Бронирование.ID AS Бронь\_ID,

Клиент.ФИО,

Бронирование.Статус,

Бронирование.Дата\_предположительного\_заселения,

Бронирование.Предварительная\_стоимость

FROM

Бронирование, Клиент

WHERE

Бронирование.Клиент\_ID = Клиент.ID;

Изображение выглядит как текст, число, снимок экрана, Параллельный

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Показать все заселения с указанием ФИО клиента и номера комнаты:

SELECT

Заселение.ID AS Заселение\_ID,

Клиент.ФИО,

Номер.ID AS Номер\_ID,

Номер.Комфортность,

Заселение.Дата\_заселения,

Заселение.Итоговая\_стоимость

FROM

Заселение, Клиент, Заселение\_Номер, Номер

WHERE

Заселение.Клиент\_ID = Клиент.ID

AND Заселение\_Номер.Заселение\_ID = Заселение.ID

AND Заселение\_Номер.Номер\_ID = Номер.ID;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Показать все оплаты с указанием клиента и способа оплаты:

SELECT

Оплата.ID AS Оплата\_ID,

Клиент.ФИО,

Оплата.Дата\_оплаты,

Оплата.Сумма,

Оплата.Способ\_оплаты

FROM

Оплата, Заселение, Клиент

WHERE

Оплата.Заселение\_ID = Заселение.ID

AND Заселение.Клиент\_ID = Клиент.ID;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

### 2.2. Реализовать запросы пункта 2.1 через внутреннее соединение inner join.

1. Показать все бронирования с ФИО клиента (вместо Клиент\_ID):

SELECT

Бронирование.ID AS Бронь\_ID,

Клиент.ФИО,

Бронирование.Статус,

Бронирование.Дата\_предположительного\_заселения,

Бронирование.Предварительная\_стоимость

FROM

Бронирование

INNER JOIN

Клиент ON Бронирование.Клиент\_ID = Клиент.ID;

Изображение выглядит как текст, число, снимок экрана, Параллельный

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Показать все заселения с указанием ФИО клиента и номера комнаты:

SELECT

Заселение.ID AS Заселение\_ID,

Клиент.ФИО,

Номер.ID AS Номер\_ID,

Номер.Комфортность,

Заселение.Дата\_заселения,

Заселение.Итоговая\_стоимость

FROM

Заселение

INNER JOIN

Клиент ON Заселение.Клиент\_ID = Клиент.ID

INNER JOIN

Заселение\_Номер ON Заселение\_Номер.Заселение\_ID = Заселение.ID

INNER JOIN

Номер ON Заселение\_Номер.Номер\_ID = Номер.ID;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Показать все оплаты с указанием клиента и способа оплаты:

SELECT

Оплата.ID AS Оплата\_ID,

Клиент.ФИО,

Оплата.Дата\_оплаты,

Оплата.Сумма,

Оплата.Способ\_оплаты

FROM

Оплата

INNER JOIN

Заселение ON Оплата.Заселение\_ID = Заселение.ID

INNER JOIN

Клиент ON Заселение.Клиент\_ID = Клиент.ID;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

### 2.3. Левое внешнее соединение left join. Привести 2–3 запроса.

1. Все клиенты и их бронирования (включая тех, у кого нет брони):

SELECT

Клиент.ФИО,

Бронирование.ID AS Бронь\_ID,

Бронирование.Статус,

Бронирование.Дата\_предположительного\_заселения

FROM

Клиент

LEFT JOIN

Бронирование ON Клиент.ID = Бронирование.Клиент\_ID

ORDER BY

Клиент.ФИО;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Все заселения и их оплаты (включая незавершённые без оплаты):

SELECT

Заселение.ID AS Заселение\_ID,

Клиент.ФИО,

Заселение.Дата\_заселения,

Оплата.Дата\_оплаты,

Оплата.Сумма

FROM

Заселение

LEFT JOIN

Оплата ON Заселение.ID = Оплата.Заселение\_ID

INNER JOIN

Клиент ON Заселение.Клиент\_ID = Клиент.ID

ORDER BY

Заселение.ID;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

### 2.4. Правое внешнее соединение right join. Привести 2–3 запроса.

1. Все оплаты и связанные с ними заселения (даже если по заселению не указаны данные клиента):

SELECT

Оплата.ID AS Оплата\_ID,

Оплата.Дата\_оплаты,

Оплата.Сумма,

Заселение.ID AS Заселение\_ID,

Заселение.Дата\_заселения,

Заселение.Фактическая\_дата\_выезда

FROM

Заселение

RIGHT JOIN

Оплата ON Заселение.ID = Оплата.Заселение\_ID

ORDER BY

Оплата.Дата\_оплаты;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Все услуги и заселения, в которых они использовались (включая услуги, которые никто не заказывал):

SELECT

Услуга.Название,

Услуга.Цена,

Заселение.ID AS Заселение\_ID,

Заселение.Дата\_заселения

FROM

Заселение\_Услуга

RIGHT JOIN

Услуга ON Заселение\_Услуга.Услуга\_ID = Услуга.ID

RIGHT JOIN

Заселение ON Заселение\_Услуга.Заселение\_ID = Заселение.ID

ORDER BY

Услуга.Название;

Изображение выглядит как текст, снимок экрана, число, меню

Содержимое, созданное искусственным интеллектом, может быть неверным.

### 2.5. Привести примеры 2–3 запросов с использованием агрегатных функций и группировки.

1. Средняя стоимость суток по категориям номеров:

SELECT

Комфортность,

AVG(Цена\_сутки) AS Средняя\_цена

FROM Номер

GROUP BY Комфортность;

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Подсчёт количества бронирований у каждого клиента:

SELECT

Клиент\_ID,

COUNT(\*) AS Количество\_бронирований

FROM

Бронирование

GROUP BY

Клиент\_ID;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

### 2.6. Привести примеры 2–3 запросов с использованием группировки и условия отбора групп (Having).

1. Услуги, которые были заказаны более одного раза:

SELECT

Услуга\_ID,

COUNT(\*) AS Количество\_заказов

FROM

Заселение\_Услуга

GROUP BY

Услуга\_ID

HAVING

COUNT(\*) > 1;

Изображение выглядит как текст, снимок экрана, линия, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Номера, которые заселяли больше одного раза:

SELECT

Номер\_ID,

COUNT(\*) AS Количество\_заселений

FROM

Заселение\_Номер

GROUP BY

Номер\_ID

HAVING

COUNT(\*) > 1;

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

### 2.7. Привести примеры 3–4 вложенных (соотнесенных, c использованием IN, EXISTS) запросов.

1. Клиенты, которые пользовались хотя бы одной услугой:

SELECT DISTINCT ФИО

FROM Клиент

WHERE ID IN (

SELECT Заселение.Клиент\_ID

FROM Заселение

WHERE ID IN (

SELECT Заселение\_ID

FROM Заселение\_Услуга

)

);

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Номера, в которых жили клиенты, оплатившие проживание картой:

SELECT DISTINCT n.\*

FROM Номер n

WHERE EXISTS (

SELECT 1

FROM Заселение\_Номер zn

JOIN Оплата o ON zn.Заселение\_ID = o.Заселение\_ID

WHERE zn.Номер\_ID = n.ID AND o.Способ\_оплаты = 'Карта'

);

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Клиенты, забронировавшие номер, но не заселившиеся:

SELECT DISTINCT k.ФИО

FROM Клиент k

JOIN Бронирование b ON k.ID = b.Клиент\_ID

WHERE b.Статус <> 'отменена'

AND NOT EXISTS (

SELECT 1

FROM Заселение z

WHERE z.Клиент\_ID = k.ID

);



## ***3. Представления.***

### 3.1 На основе любых запросов из п. 2 создать два представления (VIEW).

1. Все бронирования с ФИО клиента:

CREATE VIEW Бронирования\_С\_Клиентами AS

SELECT

Бронирование.ID AS Бронь\_ID,

Клиент.ФИО,

Бронирование.Статус,

Бронирование.Дата\_предположительного\_заселения,

Бронирование.Предварительная\_стоимость

FROM

Бронирование

INNER JOIN

Клиент ON Бронирование.Клиент\_ID = Клиент.ID;

SELECT \* FROM Бронирования\_С\_Клиентами;

Изображение выглядит как текст, снимок экрана, число, Параллельный

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Клиенты, пользовавшиеся услугами:

CREATE VIEW Клиенты\_С\_Услугами AS

SELECT DISTINCT

Клиент.ФИО

FROM

Клиент

WHERE

Клиент.ID IN (

SELECT Заселение.Клиент\_ID

FROM Заселение

WHERE Заселение.ID IN (

SELECT Заселение\_ID

FROM Заселение\_Услуга

)

);

SELECT \* FROM Клиенты\_С\_Услугами;

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

### 3.2 Привести примеры использования общетабличных выражений (СТЕ) (2–3 запроса).

1. Подсчитать количество бронирований для каждого клиента:

WITH КоличествоБронирований AS (

SELECT Клиент\_ID, COUNT(\*) AS Кол\_во\_бронирований

FROM Бронирование

GROUP BY Клиент\_ID

)

SELECT

Клиент.ФИО,

КоличествоБронирований.Кол\_во\_бронирований

FROM

Клиент

JOIN

КоличествоБронирований ON Клиент.ID = КоличествоБронирований.Клиент\_ID

ORDER BY

КоличествоБронирований.Кол\_во\_бронирований DESC;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Определить общую сумму оплат по каждому клиенту:

WITH СуммаОплат AS (

SELECT

Заселение.Клиент\_ID,

SUM(Оплата.Сумма) AS Общая\_сумма

FROM

Оплата

JOIN

Заселение ON Оплата.Заселение\_ID = Заселение.ID

GROUP BY

Заселение.Клиент\_ID

)

SELECT

Клиент.ФИО,

СуммаОплат.Общая\_сумма

FROM

Клиент

JOIN

СуммаОплат ON Клиент.ID = СуммаОплат.Клиент\_ID

ORDER BY

Общая\_сумма DESC;

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Найти номера, чья цена выше средней:

WITH СредняяЦена AS (

SELECT AVG(Цена\_сутки) AS Средняя

FROM Номер

)

SELECT

ID,

Комфортность,

Цена\_сутки

FROM

Номер, СредняяЦена

WHERE

Номер.Цена\_сутки > СредняяЦена.Средняя;

Изображение выглядит как текст, Шрифт, снимок экрана, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

## ***4. Функции ранжирования.***

### 4.1 Привести примеры 3–4 запросов с использованием ROW\_NUMBER, RANK, DENSE\_RANK (c PARTITION BY и без).

1. Нумерация заселений по дате заселения:

SELECT

ROW\_NUMBER() OVER (ORDER BY Дата\_заселения) AS Номер\_заселения,

ID AS Заселение\_ID,

Клиент\_ID,

Дата\_заселения,

Фактическая\_дата\_выезда,

Итоговая\_стоимость

FROM

Заселение;

Изображение выглядит как текст, снимок экрана, число, Параллельный

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Нумерация бронирований внутри каждого клиента:

SELECT

Клиент\_ID,

ROW\_NUMBER() OVER (PARTITION BY Клиент\_ID ORDER BY Дата\_создания) AS Номер\_брони,

ID AS Бронь\_ID,

Статус,

Дата\_создания,

Предварительная\_стоимость

FROM

Бронирование;

Изображение выглядит как текст, число, снимок экрана, Параллельный

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Рейтинг номеров по цене за сутки:

SELECT

ID AS Номер\_ID,

Комфортность,

Цена\_сутки,

RANK() OVER (ORDER BY Цена\_сутки DESC) AS Рейтинг\_цены

FROM

Номер;

Изображение выглядит как текст, число, снимок экрана, меню

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Рейтинг номеров по цене внутри каждой категории комфорта:

SELECT

Комфортность,

ID AS Номер\_ID,

Цена\_сутки,

DENSE\_RANK() OVER (PARTITION BY Комфортность ORDER BY Цена\_сутки DESC) AS Ранг\_в\_категории

FROM

Номер

ORDER BY

Комфортность, Ранг\_в\_категории;

Изображение выглядит как текст, число, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

## ***5. Объединение, пересечение, разность.***

### 5.1 Привести примеры 3–4 запросов с использованием UNION / UNION ALL, EXCEPT, INTERSECT. Данные в одном из запросов отсортируйте по произвольному признаку.

1. Список уникальных клиентов, которые либо делали бронирование, либо были заселены:

SELECT Клиент\_ID

FROM Бронирование

UNION

SELECT Клиент\_ID

FROM Заселение;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Все клиенты с бронированием и заселением, включая дубликаты:

SELECT Клиент\_ID, 'Бронирование' AS Тип

FROM Бронирование

UNION ALL

SELECT Клиент\_ID, 'Заселение' AS Тип

FROM Заселение;

Изображение выглядит как текст, число, Шрифт, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Найти клиентов, которые и бронировали номера, и были заселены:

SELECT Клиент\_ID

FROM Бронирование

INTERSECT

SELECT Клиент\_ID

FROM Заселение;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Клиенты, которые бронировали, но не заселились:

SELECT ID, ФИО, Паспортные\_данные, Пол

FROM Клиент

WHERE ID IN (

SELECT Клиент\_ID

FROM Бронирование

WHERE Статус <> 'отменена'

EXCEPT

SELECT Клиент\_ID

FROM Заселение

);



## ***6. Использование CASE, PIVOT и UNPIVOT***

### 6.1 Привести примеры получения сводных (итоговых) таблиц с использованием CASE.

1. Итоги по оплатам - сравнение способов оплаты:

SELECT

Способ\_оплаты,

COUNT(\*) AS Количество\_оплат,

SUM(Сумма) AS Общая\_сумма,

SUM(CASE WHEN Сумма >= 10000 THEN 1 ELSE 0 END) AS Оплаты\_свыше\_10000,

SUM(CASE WHEN Сумма < 10000 THEN 1 ELSE 0 END) AS Оплаты\_до\_10000

FROM

Оплата

GROUP BY

Способ\_оплаты

ORDER BY

Общая\_сумма DESC;

Изображение выглядит как снимок экрана, текст, линия, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Анализ услуг - как часто используются платные и бесплатные:

SELECT

CASE

WHEN Цена = 0 THEN 'Бесплатные услуги'

WHEN Цена BETWEEN 1 AND 1000 THEN 'Недорогие услуги'

ELSE 'Премиум услуги'

END AS Категория\_услуги,

COUNT(\*) AS Количество\_услуг

FROM

Услуга

GROUP BY

CASE

WHEN Цена = 0 THEN 'Бесплатные услуги'

WHEN Цена BETWEEN 1 AND 1000 THEN 'Недорогие услуги'

ELSE 'Премиум услуги'

END

ORDER BY

Количество\_услуг DESC;

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

### 6.2 Привести примеры получения сводных (итоговых) таблиц с использованием PIVOT и UNPIVOT.

PIVOT

1. Сколько каждый клиент потратил на разные услуги:

SELECT \*

FROM

(

SELECT

c.ФИО,

u.Название AS Услуга,

u.Цена

FROM Заселение\_Услуга zu

JOIN Заселение z ON zu.Заселение\_ID = z.ID

JOIN Клиент c ON z.Клиент\_ID = c.ID

JOIN Услуга u ON zu.Услуга\_ID = u.ID

) AS src

PIVOT

(

SUM(Цена)

FOR Услуга IN ([Завтрак], [Ужин], [Трансфер], [Сауна], [Бассейн], [Парковка], [Wi-Fi], [Мини-бар], [Спортзал], [Прачечная])

) AS pvt;

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Сколько заселений было в каждом типе номера:

SELECT \*

FROM

(

SELECT

n.Комфортность,

z.ID AS Заселение\_ID

FROM Заселение\_Номер zn

JOIN Заселение z ON zn.Заселение\_ID = z.ID

JOIN Номер n ON zn.Номер\_ID = n.ID

) AS src

PIVOT

(

COUNT(Заселение\_ID)

FOR Комфортность IN ([Стандарт], [Комфорт], [Комфорт+], [Люкс], [Апартаменты])

) AS pvt;



UNPIVOT

1. Сводная таблица стоимости бронирования и заселения:

SELECT

ID AS Заселение\_ID,

Тип\_стоимости,

Сумма

FROM

(

SELECT

ID,

Предварительная\_стоимость,

Итоговая\_стоимость

FROM Заселение

) AS src

UNPIVOT

(

Сумма FOR Тип\_стоимости IN (Предварительная\_стоимость, Итоговая\_стоимость)

) AS unpvt;

Изображение выглядит как текст, снимок экрана, число, Параллельный

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. Анализ числовых характеристик номеров:

SELECT ID, Характеристика, Значение

FROM (

SELECT

ID,

CAST(Вместимость AS DECIMAL(10,2)) AS Вместимость,

CAST(Этаж AS DECIMAL(10,2)) AS Этаж,

Цена\_сутки

FROM Номер

) AS src

UNPIVOT (

Значение FOR Характеристика IN (Вместимость, Этаж, Цена\_сутки)

) AS unpvt;

Изображение выглядит как текст, число, Шрифт, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным. Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

# **Часть 2**

## ***1. Найти номера, не занятые и не забронированные на данный момент:***

SELECT n.ID, n.Вместимость, n.Этаж, n.Комфортность, n.Цена\_сутки

FROM Номер n

WHERE n.ID NOT IN (

SELECT nb.Номер\_ID

FROM Номер\_Бронирование nb

JOIN Бронирование b ON nb.Бронь\_ID = b.ID

WHERE b.Статус = 'создана'

)

AND n.ID NOT IN (

SELECT zn.Номер\_ID

FROM Заселение\_Номер zn

JOIN Заселение z ON zn.Заселение\_ID = z.ID

WHERE z.Фактическая\_дата\_выезда IS NULL

);

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

## ***2. Найти клиентов, выехавших из номера раньше срока, указанного в договоре***

WITH Плановые\_Даты AS (

SELECT

b.ID AS Бронь\_ID,

nb.Номер\_ID,

b.Клиент\_ID,

b.Дата\_предположительного\_заселения,

n.Цена\_сутки,

b.Предварительная\_стоимость,

DATEADD(DAY, b.Предварительная\_стоимость / n.Цена\_сутки, b.Дата\_предположительного\_заселения) AS Плановая\_дата\_выезда

FROM Бронирование b

JOIN Номер\_Бронирование nb ON b.ID = nb.Бронь\_ID

JOIN Номер n ON nb.Номер\_ID = n.ID

),

Фактические\_Заселения AS (

SELECT

z.ID AS Заселение\_ID,

zn.Номер\_ID,

z.Клиент\_ID,

z.Дата\_заселения,

z.Фактическая\_дата\_выезда

FROM Заселение z

JOIN Заселение\_Номер zn ON z.ID = zn.Заселение\_ID

WHERE z.Дата\_заселения IS NOT NULL

AND z.Фактическая\_дата\_выезда IS NOT NULL

)

SELECT

k.ФИО,

f.Клиент\_ID,

f.Номер\_ID,

f.Дата\_заселения,

f.Фактическая\_дата\_выезда,

p.Плановая\_дата\_выезда,

DATEDIFF(DAY, f.Фактическая\_дата\_выезда, p.Плановая\_дата\_выезда) AS На\_сколько\_раньше\_выехал

FROM Фактические\_Заселения f

JOIN Плановые\_Даты p

ON f.Номер\_ID = p.Номер\_ID

AND f.Клиент\_ID = p.Клиент\_ID

AND f.Дата\_заселения >= p.Дата\_предположительного\_заселения

JOIN Клиент k ON k.ID = f.Клиент\_ID

WHERE f.Фактическая\_дата\_выезда < p.Плановая\_дата\_выезда

ORDER BY k.ФИО, f.Дата\_заселения;



## ***3. Найти номера, которые ни разу не сдавались с начала текущего года:***

SELECT n.ID AS Номер\_ID,

n.Комфортность,

n.Вместимость,

n.Этаж,

n.Цена\_сутки

FROM Номер n

WHERE

NOT EXISTS (

SELECT 1

FROM Заселение\_Номер zn

JOIN Заселение z ON z.ID = zn.Заселение\_ID

WHERE zn.Номер\_ID = n.ID

AND z.Дата\_заселения >= '2025-01-01'

)

AND NOT EXISTS (

SELECT 1

FROM Номер\_Бронирование nb

JOIN Бронирование b ON nb.Бронь\_ID = b.ID

WHERE nb.Номер\_ID = n.ID

AND b.Статус IN ('создана', 'завершена')

);

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

## ***4. Найти клиентов, забронировавших номер, но так и не въехавших в него:***

SELECT DISTINCT

k.ID AS Клиент\_ID,

k.ФИО,

b.ID AS Бронь\_ID,

b.Дата\_предположительного\_заселения,

b.Статус

FROM Бронирование b

JOIN Клиент k ON b.Клиент\_ID = k.ID

LEFT JOIN Заселение z ON z.Клиент\_ID = k.ID

LEFT JOIN Заселение\_Номер zn ON zn.Заселение\_ID = z.ID

LEFT JOIN Номер\_Бронирование nb ON nb.Номер\_ID = zn.Номер\_ID AND nb.Бронь\_ID = b.ID

WHERE

-- бронь создана или завершена (не отменена)

b.Статус IN ('создана', 'завершена')

-- но по этой броне нет фактического заселения

AND nb.Бронь\_ID IS NULL

ORDER BY k.ФИО;



## ***5. Среди всех клиентов, наиболее часто пользующихся услугами гостиницы, найти клиентов с максимальным сроком проживания:***

WITH Клиенты\_Частота AS (

SELECT

z.Клиент\_ID,

COUNT(\*) AS Количество\_заселений

FROM Заселение z

WHERE z.Дата\_заселения IS NOT NULL

GROUP BY z.Клиент\_ID

),

Максимальная\_Частота AS (

SELECT MAX(Количество\_заселений) AS Макс\_число

FROM Клиенты\_Частота

),

Клиенты\_МаксЧастота AS (

SELECT c.Клиент\_ID

FROM Клиенты\_Частота c

JOIN Максимальная\_Частота m ON c.Количество\_заселений = m.Макс\_число

),

Длительность\_Проживания AS (

SELECT

z.Клиент\_ID,

DATEDIFF(DAY, z.Дата\_заселения, z.Фактическая\_дата\_выезда) AS Срок\_проживания

FROM Заселение z

WHERE z.Дата\_заселения IS NOT NULL AND z.Фактическая\_дата\_выезда IS NOT NULL

),

Клиенты\_Срок AS (

SELECT

k.ID AS Клиент\_ID,

k.ФИО,

MAX(d.Срок\_проживания) AS Максимальный\_срок

FROM Клиент k

JOIN Длительность\_Проживания d ON k.ID = d.Клиент\_ID

WHERE k.ID IN (SELECT Клиент\_ID FROM Клиенты\_МаксЧастота)

GROUP BY k.ID, k.ФИО

)

SELECT \*

FROM Клиенты\_Срок

WHERE Максимальный\_срок = (SELECT MAX(Максимальный\_срок) FROM Клиенты\_Срок);

