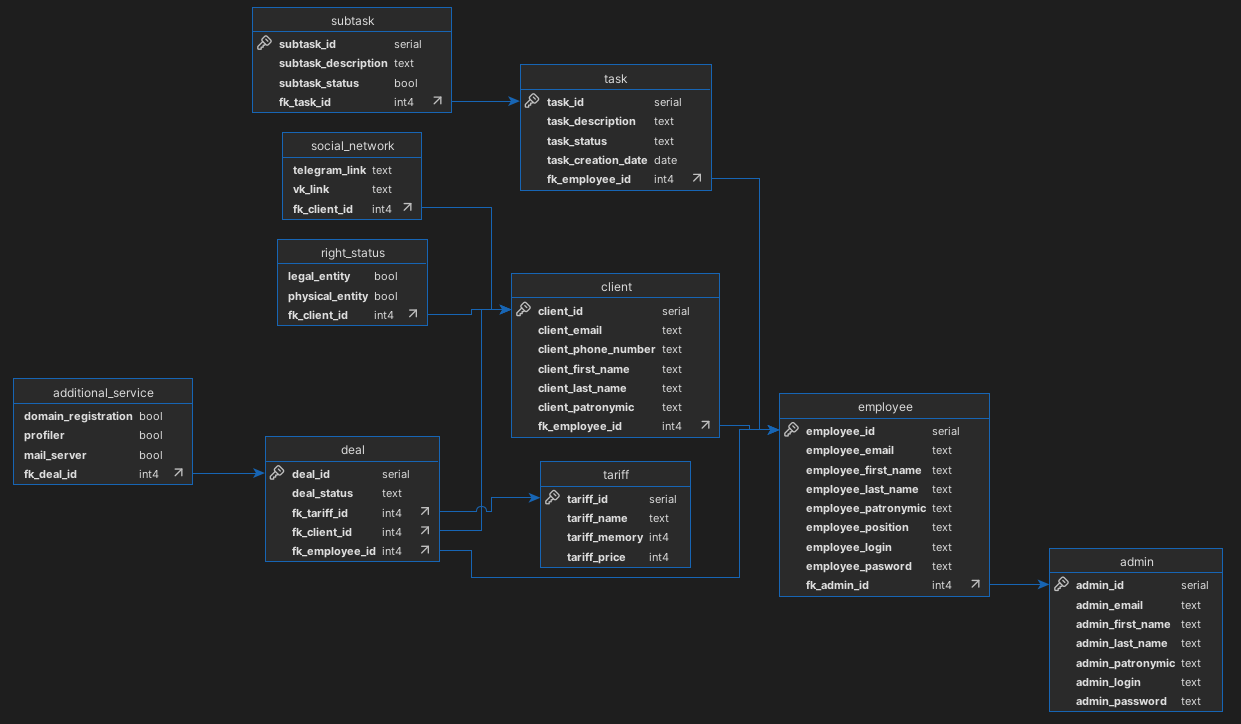
|  |  |  |
| --- | --- | --- |
| Лабораторная работа №4  Реализация SQL-запросов базы данных | Ф.И.О. | Мельников В. Д. |
| Группа | ИВТ-264 |
| Преподаватель | Аль-Мерри Гаис Мохаммед Салех |
| Дата сдачи |  |

**Скриншот реляционной схемы БД**



Функциональные требования:

– Оформление сделки с клиентом

– Сбор информации о клиентах и сделках с ними

– Отслеживание задач пользователями

– Сбор ключевых показателей

**Запросы**

**1) Запросы, указанные в функциональных требованиях (в заголовке указать, что за требование)**

1 Оформление сделки с клиентом

INSERT INTO deal (fk\_client\_id, fk\_tariff\_id, deal\_status, deal\_date)

VALUES (5, 5, 'active', NOW());

2 Получение списка клиентов и сделок с ними

SELECT client.client\_id, client.client\_email, deal.deal\_id, deal.deal\_status, deal.deal\_date

FROM client

LEFT JOIN deal ON client.client\_id = deal.fk\_client\_id

ORDER BY client.client\_id;

3 Получение списка всех задач сотрудников

SELECT task\_id, task\_name, task\_status, task\_deadline, fk\_employee\_id

FROM task

ORDER BY task\_deadline ASC;

**2) UPDATE в разных таблицах, с WHERE**

4 Обновление значения столбца subtask\_status на true в строке со значением столбца subtask\_id равным 2 для таблицы subtask

UPDATE subtask SET

subtask\_status = true

WHERE subtask\_id = 2;

5 Обновление значения столбца task\_status на 'Завершено' в строке со значением столбца task\_id = 1 для таблицы task

UPDATE task

SET task\_status = 'Завершено'

WHERE task\_id = 1;

6 Обновление значения столбца tariff\_name на ‘Минимальный’ в строке со значением столбца tariff\_id = 1 для таблицы tariff

UPDATE tariff

SET tariff\_name = 'Минимальный'

WHERE tariff\_id = 1;

7 Обновление значения столбца telegram\_link на 'https://vk.com/dmitry\_kuznetsov' в строке со значением столбца fk\_client\_id = 3 для таблицы social\_network

UPDATE social\_network

SET telegram\_link = 'https://vk.com/dmitry\_kuznetsov'

WHERE fk\_client\_id = 3;

8 Обновление значения столбца legal\_entity и physical\_entity на false и true соответственно, в строке со значением столбца fk\_client\_id = 1 для таблицы right\_status

UPDATE right\_status

SET legal\_entity = false, physical\_entity = true

WHERE fk\_client\_id = 1;

**3) DELETE в разных таблицах, с WHERE**

9-13 Предварительное добавление данных для удаления в таблицы

INSERT INTO admin

VALUES (999, 'test@test.com', 'test', 'test', 'test', 'test', 'test');

INSERT INTO employee

VALUES (999, 'test@test.com', 'test', 'test', 'test', 'test', 'test', 'test', 999);

INSERT INTO client

VALUES (11, 'test@test.com', 'test', 'test', 'test', 'test', 999);

INSERT INTO right\_status

VALUES (true, true, 11);

INSERT INTO social\_network

VALUES ('test', 'test', 11);

14 Удаление строки со значением fk\_client\_id = 11 из таблицы social\_network

DELETE FROM social\_network WHERE fk\_client\_id = 11;

15 Удаление строки со значением fk\_client\_id = 11 из таблицы right\_status

DELETE FROM right\_status WHERE fk\_client\_id = 11;

16 Удаление строки со значением client\_id = 11 из таблицы client

DELETE FROM client WHERE client\_id = 11;

17 Удаление строки со значением employee\_id = 999 из таблицы employee

DELETE FROM employee WHERE employee\_id = 999;

18 Удаление строки со значением admin\_id = 999 из таблицы admin

DELETE FROM admin WHERE admin\_id = 999;

**4) SELECT, DISTINCT, WHERE, AND/OR/NOT, IN, BETWEEN, различная работа с датами и числами, преобразование данных, IS NULL, AS для таблиц и столбцов и др. в различных вариациях**

19 Выбирает fk\_client\_id клиентов, у которых заполнены оба поля: vk\_link и telegram\_link.

SELECT fk\_client\_id AS client\_id\_with\_vk\_and\_tg

FROM social\_network

WHERE vk\_link IS NOT NULL AND telegram\_link IS NOT NULL;

20 Выбирает fk\_client\_id клиентов, у которых есть vk\_link (но telegram\_link может быть пустым).

SELECT fk\_client\_id AS fk\_client\_id\_without\_vk

FROM social\_network

WHERE vk\_link IS NOT NULL;

21 Выбирает fk\_client\_id клиентов, у которых есть telegram\_link (но vk\_link может быть пустым).

SELECT fk\_client\_id AS fk\_client\_id\_without\_tg

FROM social\_network

WHERE telegram\_link IS NOT NULL;

22 Выбирает fk\_client\_id клиентов, у которых legal\_entity = true**.**

SELECT fk\_client\_id AS client\_is\_legal\_entity

FROM right\_status

WHERE legal\_entity = true;

23 Выбирает fk\_client\_id клиентов, у которых physical\_entity = true

SELECT fk\_client\_id AS client\_is\_physical\_entity

FROM right\_status

WHERE physical\_entity = true;

24 Выбирает описания задач (task\_description), которые находятся в статусе "В процессе" или "Ожидание".

SELECT task\_description

FROM task

WHERE task\_status = 'В процессе' OR task\_status = 'Ожидание';

25 Выбирает все поля задач, созданных с 9 марта 2025

SELECT \*

FROM task

WHERE task\_creation\_date BETWEEN '2025-03-9' AND '2100-03-14';

26 Выбирает уникальные fk\_employee\_id сотрудников, которые привязаны к клиентам.

SELECT DISTINCT fk\_employee\_id

AS id\_of\_employee\_with\_clients

FROM client;

27 Выбирает уникальные fk\_employee\_id сотрудников, которые привязаны к клиентам.

SELECT DISTINCT fk\_employee\_id

AS id\_of\_employee\_with\_clients

FROM client;

28 Выбирает уникальные fk\_task\_id задач, у которых есть подзадачи.

SELECT DISTINCT fk\_task\_id

AS id\_of\_tasks\_that\_have\_subtasks

FROM subtask;

29 Выбирает описания подзадач (subtask\_description) и их родительские задачи (fk\_task\_id), где subtask\_status = true (выполнены).

SELECT subtask\_description AS completed\_subtasks, fk\_task\_id

FROM subtask

WHERE subtask\_status = true;

30 Аналогично, но для subtask\_status = false (не выполнены).

SELECT subtask\_description AS not\_completed\_subtasks, fk\_task\_id

FROM subtask

WHERE subtask\_status = false;

31 Тарифы по цене (500–2000)

SELECT tariff\_name, tariff\_memory

FROM tariff

WHERE tariff\_price BETWEEN 500 AND 2000;

32 Тарифы с памятью >100

SELECT tariff\_name, tariff\_price

FROM tariff

WHERE tariff\_memory > 100;

33 Выбирает тарифы с памятью <100 и ценой <500

SELECT tariff\_name, tariff\_price

FROM tariff

WHERE tariff\_memory < 100 AND tariff\_price < 500;

**5) LIKE и другая работа со строками**

34 Клиенты с фамилией заканчивающейся на ОВ

SELECT \*

FROM client

WHERE client\_last\_name LIKE '%ов';

35 Клиенты с фамилией заканчивающейся на ОВА

SELECT \*

FROM client

WHERE client\_last\_name LIKE '%ова';

36 Клиенты со второй буквой А в имени

SELECT \*

FROM employee

WHERE employee\_first\_name LIKE '\_а%';

37 Администраторы с отчеством, заканчивающимся на ИЧ

SELECT \*

FROM admin

WHERE admin\_patronymic LIKE '%ич';

38 Администраторы с отчеством, заканчивающимся на НА

SELECT \*

FROM admin

WHERE admin\_patronymic LIKE '%на';

**6) SELECT INTO или INSERT SELECT, что поддерживается СУБД (2-3 шт.). Для использования запроса INSERT SELECT вначале можно создать новую тестовую таблицу или несколько, в которые будут скопированы данные из существующих таблиц с помощью данного запроса. Код создания таблиц также приложить в лабораторную работу**

39 Создание таблицы с выполненными подзадачами

CREATE TABLE completed\_subtasks (

subtask\_id INT UNIQUE,

sabstask\_description TEXT

);

40 Вставка значений в созданную выше таблицу

INSERT INTO completed\_subtasks

SELECT subtask\_id, subtask\_description FROM subtask

WHERE subtask\_status = true;

41 Соаздание таблицы с клиентами, являющимися юридич. лицами

CREATE TABLE clients\_is\_legal\_entity (

client\_id INT UNIQUE,

client\_first\_name TEXT,

client\_last\_name TEXT,

client\_patronymic TEXT

);

42 Вставка значений в созданную выше таблицу

INSERT INTO clients\_is\_legal\_entity

SELECT client\_id, client\_first\_name, client\_last\_name, client\_patronymic

FROM client

JOIN right\_status ON client.client\_id = right\_status.fk\_client\_id

AND

right\_status.legal\_entity = true

**7) JOIN: INNER, OUTER (LEFT, RIGHT, FULL), CROSS, NATURAL, разных, в различных вариациях, несколько запросов с более, чем одним JOIN.**

43 -- INNER JOIN (Список сделок с информацией о клиентах)

SELECT deal.deal\_id, deal.deal\_status, client.client\_first\_name, client.client\_last\_name

FROM deal

INNER JOIN client ON deal.fk\_client\_id = client.client\_id;

44 -- INNER JOIN (Список задач с ответственными сотрудниками)

SELECT task.task\_id, task.task\_description, employee.employee\_first\_name, employee.employee\_last\_name

FROM task

INNER JOIN employee ON task.fk\_employee\_id = employee.employee\_id;

45 -- LEFT JOIN (Все клиенты и их сделки, включая тех, у кого сделок нет)

SELECT client.client\_id, client.client\_first\_name, client.client\_last\_name, deal.deal\_id, deal.deal\_status

FROM client

LEFT JOIN deal ON client.client\_id = deal.fk\_client\_id;

46 -- RIGHT JOIN (Все сделки и их клиенты, включая те сделки, у которых не указан клиент)

SELECT client.client\_id, client.client\_first\_name, client.client\_last\_name, deal.deal\_id, deal.deal\_status

FROM client

RIGHT JOIN deal ON client.client\_id = deal.fk\_client\_id;

47 -- FULL OUTER JOIN (Все клиенты и все сделки, даже если они не связаны)

SELECT client.client\_id, client.client\_first\_name, client.client\_last\_name, deal.deal\_id, deal.deal\_status

FROM client

FULL OUTER JOIN deal ON client.client\_id = deal.fk\_client\_id;

48 -- CROSS JOIN (Комбинация всех сотрудников и всех тарифов)

SELECT employee.employee\_first\_name, employee.employee\_last\_name, tariff.tariff\_name, tariff.tariff\_price

FROM employee

CROSS JOIN tariff;

49 -- NATURAL JOIN (Клиенты и их социальные сети, если у них есть связанные записи)

SELECT \*

FROM client

NATURAL JOIN social\_network;

50 -- INNER JOIN с условием (Клиенты с активными сделками)

SELECT client.client\_first\_name, client.client\_last\_name, deal.deal\_status

FROM client

INNER JOIN deal ON client.client\_id = deal.fk\_client\_id

WHERE deal.deal\_status = 'Активна';

51 -- LEFT JOIN с фильтрацией (Клиенты без сделок)

SELECT client.client\_first\_name, client.client\_last\_name

FROM client

LEFT JOIN deal ON client.client\_id = deal.fk\_client\_id

WHERE deal.deal\_id IS NULL;

52 -- RIGHT JOIN с дополнительными услугами по сделкам

SELECT deal.deal\_id, deal.deal\_status, additional\_service.domain\_registration, additional\_service.mail\_server

FROM deal

RIGHT JOIN additional\_service ON deal.deal\_id = additional\_service.fk\_deal\_id;

53 -- FULL OUTER JOIN с фильтрацией (Клиенты или сделки, у которых нет пары)

SELECT client.client\_id, client.client\_first\_name, deal.deal\_id, deal.deal\_status

FROM client

FULL OUTER JOIN deal ON client.client\_id = deal.fk\_client\_id

WHERE client.client\_id IS NULL OR deal.deal\_id IS NULL;

54 -- CROSS JOIN (Комбинация всех сотрудников и всех клиентов)

SELECT employee.employee\_first\_name, employee.employee\_last\_name, client.client\_first\_name, client.client\_last\_name

FROM employee

CROSS JOIN client;

55 --INNER JOIN (Задачи и их подзадачи)

SELECT task.task\_description, subtask.subtask\_description, subtask.subtask\_status

FROM task

INNER JOIN subtask ON task.task\_id = subtask.fk\_task\_id;

56 -- LEFT JOIN (Сотрудники и их администраторы)

SELECT employee.employee\_first\_name, employee.employee\_last\_name, admin.admin\_first\_name, admin.admin\_last\_name

FROM employee

LEFT JOIN admin ON employee.fk\_admin\_id = admin.admin\_id;

57 --RIGHT JOIN (Тарифы и их сделки, включая сделки без тарифа)

SELECT tariff.tariff\_name, tariff.tariff\_price, deal.deal\_id, deal.deal\_status

FROM tariff

RIGHT JOIN deal ON tariff.tariff\_id = deal.fk\_tariff\_id;

**8) GROUP BY (некоторые с HAVING), с LIMIT, ORDER BY (ASC|DESC) вместе с COUNT, MAX, MIN, SUM, AVG в различных вариациях, можно по отдельности**

58 Кол-во сделок по каждому из тарифов

SELECT COUNT(deal\_id), tariff\_name

FROM deal

LEFT JOIN tariff

ON deal.fk\_tariff\_id = tariff.tariff\_id

GROUP BY tariff.tariff\_name;

59 Кол-во клиентов по каждому сотруднику

SELECT fk\_employee\_id,

COUNT(client\_id) AS client\_count

FROM client

GROUP BY fk\_employee\_id

ORDER BY client\_count DESC;

60 Вывод 3-х самых дорогих тарифов

SELECT tariff\_name, tariff\_price, tariff\_memory

FROM tariff

ORDER BY tariff\_price DESC

LIMIT 3;

61 Вывод 3 тарифом с наименьшим количеством памяти

SELECT tariff\_name, tariff\_price

FROM tariff

ORDER BY tariff\_memory ASC

LIMIT 3;

62 Топ 5 клиентов с наибольшим количеством сделок

SELECT fk\_client\_id, COUNT(deal\_id) AS total\_deals

FROM deal

GROUP BY fk\_client\_id

ORDER BY total\_deals DESC

LIMIT 5;

63 Вывести среднюю стоимость всех тарифов

SELECT AVG(tariff\_price) AS avg\_price

FROM tariff;

64 Вывод суммарного дохода по каждому из тарифов

SELECT t.tariff\_name, SUM(t.tariff\_price) AS total\_revenue

FROM deal d

JOIN tariff t ON d.fk\_tariff\_id = t.tariff\_id

GROUP BY t.tariff\_name

ORDER BY total\_revenue DESC;

65 Отображение сотрудников с более чем 1 задачей

SELECT fk\_employee\_id, COUNT(task\_id) AS task\_count

FROM task

GROUP BY fk\_employee\_id

HAVING COUNT(task\_id) > 1

ORDER BY task\_count DESC;

66 Подсчет кол-ва клиентов зарегистрированных в соц. сетях

SELECT

(SELECT COUNT(\*) FROM social\_network WHERE telegram\_link IS NOT NULL) AS telegram\_users,

(SELECT COUNT(\*) FROM social\_network WHERE vk\_link IS NOT NULL) AS vk\_users;

67 Кол-во доп услуг по каждой сделке

SELECT fk\_deal\_id,

SUM(domain\_registration::int + profiler::int + mail\_server::int) AS t otal\_additional\_services

FROM additional\_service

GROUP BY fk\_deal\_id

ORDER BY total\_additional\_services DESC;

68 Кол-во выполненных подзадач для каждой задачи

SELECT fk\_task\_id, COUNT(subtask\_id) AS completed\_subtasks

FROM subtask

WHERE subtask\_status = TRUE

GROUP BY fk\_task\_id

ORDER BY fk\_task\_id ASC;

69 Вывод минимальной стоимости тарифа

SELECT MIN(tariff\_price) AS min\_tariff\_price

FROM tariff;

70 Вывод максимальной стоимости тарифа

SELECT MAX(tariff\_price) AS max\_tariff\_price

FROM tariff;

**9) UNION, EXCEPT, INTERSECT**

80 Создание объединенного списка клиентов и работников с их почтой, имененм и фамилией

SELECT client\_email AS email, client\_first\_name AS first\_name, client\_last\_name AS last\_name, 'Client' AS role

FROM client

UNION

SELECT employee\_email, employee\_first\_name, employee\_last\_name, 'Employee'

FROM employee

ORDER BY role;

81-82 Вывод списка клиентов, не имеющих сделки

INSERT INTO client VALUES (11, 'test@test.com', 'test', 'test', 'test', 'test');

SELECT client\_id, client\_email

FROM client

EXCEPT

SELECT DISTINCT fk\_client\_id, client\_email

FROM deal

JOIN client ON deal.fk\_client\_id = client.client\_id;

83 Вывод id клиентов, являющихся юр. лицом и имеющих ссылку на ВК

SELECT fk\_client\_id AS client\_id

FROM right\_status

WHERE legal\_entity = TRUE

INTERSECT

SELECT fk\_client\_id

FROM social\_network

WHERE vk\_link IS NOT NULL;

**10) Вложенные SELECT с GROUP BY, ALL, ANY, EXISTS**

84 Вывод клиентов, имеющих сделок больше, чем среднее значение

SELECT client\_id, client\_email,

(SELECT COUNT(deal\_id) FROM deal WHERE deal.fk\_client\_id = client.client\_id) AS deal\_count

FROM client

WHERE (SELECT COUNT(deal\_id) FROM deal WHERE deal.fk\_client\_id = client.client\_id) >

(SELECT AVG(deal\_count) FROM

(SELECT fk\_client\_id, COUNT(deal\_id) AS deal\_count FROM deal GROUP BY fk\_client\_id) AS avg\_deals);

85 Вывод сотрудников, имеющих хотя бы одну завершенную задачу

SELECT employee\_id, employee\_email

FROM employee

WHERE EXISTS (SELECT 1 FROM task WHERE task.fk\_employee\_id = employee.employee\_id AND task.task\_status = 'Завершено');

SELECT client\_id, client\_email

FROM client

WHERE client\_id = ALL (SELECT fk\_client\_id FROM deal WHERE deal\_status = 'completed');

86 Клиенты с хотя бы одной сделкой

SELECT client\_id, client\_email

FROM client

WHERE client\_id = ANY (SELECT fk\_client\_id FROM deal);

**11) GROUP\_CONCAT (string\_agg)**

87 Вывод email адресов всех клиентов через запятую

SELECT STRING\_AGG(client\_email, ', ') AS all\_emails

FROM client;

88 Вывод всех названий тарифов через |

SELECT STRING\_AGG(tariff\_name, ' | ') AS available\_tariffs

FROM tariff;

89 Вывод всех задач сотрудников через запятую

SELECT fk\_employee\_id, STRING\_AGG(task\_description, ', ') AS assigned\_tasks

FROM task

GROUP BY fk\_employee\_id;

**12) Запросы с WITH**

90 Вывод тарифы, цена которых выше средней.

WITH avg\_price AS (

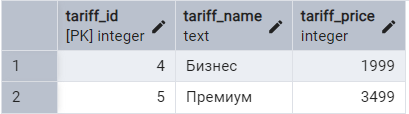
SELECT AVG(tariff\_price) AS avg\_tariff FROM tariff

)

SELECT tariff\_id, tariff\_name, tariff\_price

FROM tariff, avg\_price

WHERE tariff\_price > avg\_tariff;



91 Вывод информация о клиентах и количества завершенных с ними сделках

WITH CompletedDeals AS (

SELECT fk\_client\_id, COUNT(\*) AS total\_completed

FROM deal

WHERE deal\_status = 'Завершена'

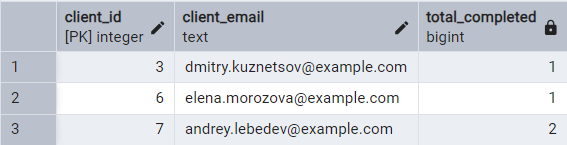
GROUP BY fk\_client\_id

)

SELECT client.client\_id, client.client\_email, total\_completed

FROM client

JOIN CompletedDeals ON client.client\_id = CompletedDeals.fk\_client\_id;



92. Вывод информации о сотрудниках и количестве их задач

WITH TaskCounts AS (

SELECT fk\_employee\_id, COUNT(\*) AS task\_total

FROM task

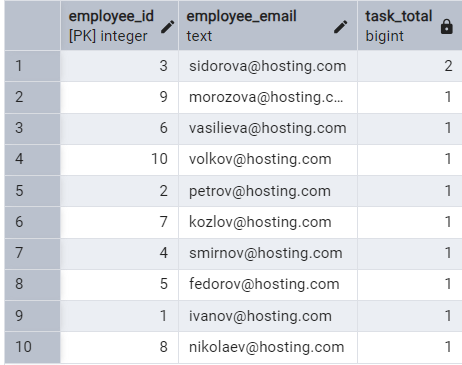
GROUP BY fk\_employee\_id

)

SELECT employee.employee\_id, employee.employee\_email, task\_total

FROM employee

JOIN TaskCounts ON employee.employee\_id = TaskCounts.fk\_employee\_id;



**13) Запросы со строковыми функциями СУБД, с функциями работы с датами временем (форматированием дат), с арифметическими функциями**

93 Вывод email клиентов в верхнем регистре

SELECT client\_id, UPPER(client\_email) AS email\_upper

FROM client;

94 Вывод текущей даты и времени

SELECT CURRENT\_TIMESTAMP AS now;

95 Объединение имени и фамилии клиента в одну строку

SELECT client\_id, CONCAT(client\_first\_name, ' ', client\_last\_name) AS full\_name

FROM client;

96 Вывод дня недели создания каждой из задач

SELECT task\_id, TO\_CHAR(task\_creation\_date, 'Day') AS task\_day

FROM task;

**14) Сложные запросы, входящие в большинство групп выше, т.е. SELECT ... JOIN ... JOIN ... WHERE ... GROUP BY ... ORDER BY ... LIMIT ...; (5-7 шт. +), можно написать больше вместо простых**

97 Вывод клиентов, имеющих сделок больше, чем среднее значение

SELECT client\_id, client\_email,

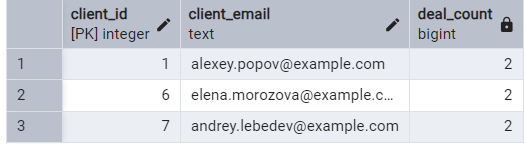
(SELECT COUNT(deal\_id) FROM deal WHERE deal.fk\_client\_id = client.client\_id) AS deal\_count

FROM client

WHERE (SELECT COUNT(deal\_id) FROM deal WHERE deal.fk\_client\_id = client.client\_id) >

(SELECT AVG(deal\_count) FROM

(SELECT fk\_client\_id, COUNT(deal\_id) AS deal\_count FROM deal GROUP BY fk\_client\_id) AS avg\_deals);

****

98. 5 самых активных сотрудников по количеству задач

SELECT e.employee\_id,

e.employee\_email,

COUNT(t.task\_id) AS total\_tasks

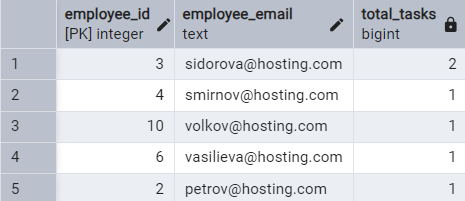
FROM employee e

JOIN task t ON e.employee\_id = t.fk\_employee\_id

GROUP BY e.employee\_id

ORDER BY total\_tasks DESC

LIMIT 5;



99. Средняя сумма сделок по каждому клиенту

SELECT c.client\_id,

c.client\_email,

ROUND(AVG(t.tariff\_price), 2) AS avg\_spent

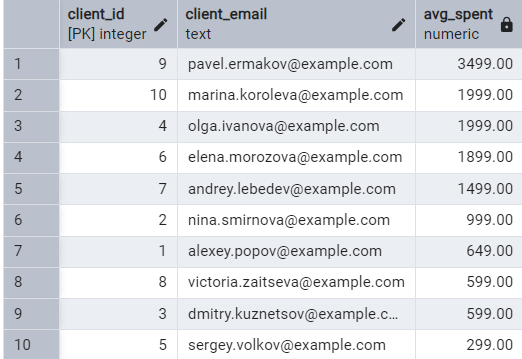
FROM client c

JOIN deal d ON c.client\_id = d.fk\_client\_id

JOIN tariff t ON d.fk\_tariff\_id = t.tariff\_id

GROUP BY c.client\_id

ORDER BY avg\_spent DESC;



100 Доход с продаж (завершенных сделок)

SELECT t.tariff\_id,

t.tariff\_name,

SUM(t.tariff\_price) AS total\_revenue

FROM deal d

JOIN tariff t ON d.fk\_tariff\_id = t.tariff\_id

WHERE d.deal\_status = 'Завершена'

GROUP BY t.tariff\_id

ORDER BY total\_revenue DESC;

