Федеральное государственное бюджетное образовательное учреждение высшего

образования

«Чувашский государственный университет им. И.Н. Ульянова»

Кафедра вычислительной техники

Лабораторная работа № 4

SQL-запросы

Вариант: Инвестор

Выполнил: Назаров Ярослав

студент группы ИВТ-42-23

Проверил: старший преподаватель   
Марков А.В.

Чебоксары, 2025

## Таблица запросов

|  |  |  |  |
| --- | --- | --- | --- |
| № | Имя запроса | Текст запроса | Запрос на |
| 1 | q\_Топ5ДорогихАктивов | Вывести 5 самых дорогих активов по цене за единицу | SELECT name, ticker, unit\_price  FROM investor\_asset  ORDER BY unit\_price DESC  LIMIT 5; |
| 2 | q\_10ПоследнихТранзакций | Вывести 10 последних транзакций | SELECT \*  FROM investor\_transaction  ORDER BY transaction\_datetime DESC  LIMIT 10; |
| 3 | q\_ТранзакцииЗаПериод | Вывести транзакции в указанном интервале дат | SELECT \*  FROM investor\_transaction  WHERE transaction\_datetime  BETWEEN $1 AND $2; |
| 4 | q\_СчетаВИСилиБрокерские | Вывести счета, которые являются "ИИС" или "Брокерский" | SELECT name, account\_type, status  FROM investor\_account  WHERE account\_type IN ('iis', 'broker'); |
| 5 | q\_АктивныеСчета | Вывести все активные счета инвесторов | SELECT a.name, a.account\_type,  a.open\_date, i.name as investor\_name  FROM investor\_account a  JOIN investor\_investor i ON a.investor\_id = i.id  WHERE a.status = 'active'; |
| 6 | q\_ИнвесторыБезСчетов | Вывести инвесторов, у которых нет открытых счетов | SELECT name  FROM investor\_investor i  WHERE NOT EXISTS (  SELECT 1  FROM investor\_account a  WHERE a.investor\_id = i.id  ); |
| 7 | q\_СчетаПоФамилии | Вывести все счета инвестора по введенной фамилии | SELECT [i.name](https://i.name/) as investor\_name,  [a.name](https://a.name/) as account\_number,  a.account\_type FROM investor\_investor i JOIN investor\_account a ON [i.id](https://i.id/) = a.investor\_id WHERE [i.name](https://i.name/) ILIKE '%' |
| 8 | q\_ИнвесторыРожденныеВ90х | Вывести инвесторов, родившихся в 1990-х годах | SELECT name, birth\_date  FROM investor\_investor  WHERE EXTRACT(YEAR FROM birth\_date)  BETWEEN 1990 AND 1999; |
| 9 | q\_БрокерыБолее5счетов | Вывести брокеров, у которых открыто более 5 счетов | SELECT b.name, COUNT(a.id) as account\_count  FROM investor\_broker b  JOIN investor\_account a ON b.id = a.broker\_id  GROUP BY b.id, b.name  HAVING COUNT(a.id) > 5; |
| 10 | q\_КолвоСчетовПоТипам | Вывести количество счетов каждого типа | SELECT account\_type, COUNT(\*) as count\_accounts  FROM investor\_account  GROUP BY account\_type; |
| 11 | q\_ПолнаяИнформацияТранзакции | Вывести полную информацию о транзакциях | SELECT i.name as investor\_name,  b.name as broker\_name,  a.name as account\_number,  ast.name as asset\_name,  t.operation\_type, t.amount,  t.transaction\_datetime  FROM investor\_transaction t  JOIN investor\_account a ON t.account\_id = a.id  JOIN investor\_investor i ON a.investor\_id = i.id  JOIN investor\_broker b ON a.broker\_id = b.id  LEFT JOIN investor\_asset ast ON t.asset\_id = ast.id  ORDER BY t.transaction\_datetime DESC; |
| 12 | q\_ИнвесторыСАктивнымиСчетами | Вывести инвесторов с активными счетами | SELECT name  FROM investor\_investor i  WHERE EXISTS (  SELECT 1  FROM investor\_account a  WHERE a.investor\_id = i.id  AND a.status = 'active'  ); |
| 13 | q\_ИнвесторыСПочтойGmail | Вывести инвесторов с почтой на [gmail.com](https://gmail.com/) | SELECT name, email  FROM investor\_investor  WHERE email LIKE '%@gmail.com'; |
| 14 | q\_ИнвесторыСТойЖеДатойРождения | Вывести пары инвесторов с одинаковой датой рождения | SELECT i1.name as investor1,  i2.name as investor2,  i1.birth\_date  FROM investor\_investor i1  JOIN investor\_investor i2 ON  i1.birth\_date = i2.birth\_date  AND i1.id < i2.id; |
| 15 | q\_ВсеБрокерыИСчета | Вывести всех брокеров и количество их счетов | SELECT b.name as broker\_name,  COUNT(a.id) as account\_count  FROM investor\_broker b  LEFT JOIN investor\_account a ON b.id = a.broker\_id  GROUP BY b.id, b.name  ORDER BY account\_count DESC; |
| 16 | q\_ДобавитьИнвестора | Добавить новую запись в таблицу инвесторов | INSERT INTO investor\_investor  (name, birth\_date, phone, email)  VALUES ('Иванов Иван Иванович',  '1985-10-15',  '+7 (123) 456-78-90',  'ivanov.ii@email.com'); |
| 17 | q\_ОбновитьСтатусСчета | Обновить статус счёта на "Закрыт" | UPDATE investor\_account  SET status = 'closed'  WHERE name = '40817810000000001234'; |
| 18 | q\_СписокВсехКонтактов | Вывести единый список инвесторов и брокеров | SELECT name, 'investor' as type  FROM investor\_investor  UNION ALL  SELECT name, 'broker' as type  FROM investor\_broker  ORDER BY type, name; |
| 19 | q\_СоздатьТаблицуАрхив | Создать таблицу архивных транзакций | CREATE TABLE archive\_transactions\_2024 AS  SELECT \*  FROM investor\_transaction  WHERE EXTRACT(YEAR FROM transaction\_datetime) = 2024; |
| 20 | q\_УдалитьТранзакции | Удалить транзакции с суммой менее 100 | DELETE FROM investor\_transaction  WHERE amount < 100; |
| 21 | q\_СводкаПоТипамОпераций | Сводка операций по типам для инвесторов | SELECT i.name as investor\_name,  SUM(CASE WHEN t.operation\_type='buy' THEN t.amount ELSE 0 END) as total\_buy,  SUM(CASE WHEN t.operation\_type='sell' THEN t.amount ELSE 0 END) as total\_sell  FROM investor\_investor i  LEFT JOIN investor\_account a ON i.id = a.investor\_id  LEFT JOIN investor\_transaction t ON a.id = t.account\_id  GROUP BY i.id, i.name; |

**Таблица запросов**

|  |  |  |
| --- | --- | --- |
| **№** | **Имя запроса** | **Код на (Odoo)** |
| 1 | q\_Топ5ДорогихАктивов | @api.model  def get\_top\_expensive\_assets(self):  return self.env['investor.asset'].search(  [], order='unit\_price desc', limit=5  ) |
| 2 | q\_10ПоследнихТранзакций | @api.model  def get\_recent\_transactions(self):  return self.env['investor.transaction'].search(  [], limit=10, order='transaction\_datetime desc'  ) |
| 3 | q\_ТранзакцииЗаПериод | @api.model  def get\_transactions\_by\_date\_range(self, start\_date, end\_date):  return self.env['investor.transaction'].search([  ('transaction\_datetime', '>=', start\_date),  ('transaction\_datetime', '<=', end\_date)  ]) |
| 4 | q\_СчетаВИСилиБрокерские | @api.model  def get\_iis\_broker\_accounts(self):  return self.env['investor.account'].search([  ('account\_type', 'in', ['iis', 'broker'])  ]) |
| 5 | q\_АктивныеСчета | @api.model  def get\_active\_accounts(self):  return self.env['investor.account'].search([  ('status', '=', 'active')  ]) |
| 6 | q\_ИнвесторыБезСчетов | @api.model  def get\_investors\_without\_accounts(self):  all\_investors = self.env['investor.investor'].search([])  investors\_with\_accounts = self.env['investor.account'].search([]).mapped('investor\_id')  return all\_investors - investors\_with\_accounts |
| 7 | q\_СчетаПоФамилии |  |
| 8 | q\_ИнвесторыРожденныеВ90х | @api.model  def get\_investors\_born\_in\_90s(self):  return self.env['investor.investor'].search([  ('birth\_date', '>=', '1990-01-01'),  ('birth\_date', '<=', '1999-12-31')  ]) |
| 9 | q\_БрокерыБолее5счетов | @api.model  def get\_brokers\_with\_many\_accounts(self):  brokers = self.env['investor.broker'].search([])  result = []  for broker in brokers:  account\_count = self.env['investor.account'].search\_count([  ('broker\_id', '=', broker.id)  ])  if account\_count > 5:  result.append(broker)  return result |
| 10 | q\_КолвоСчетовПоТипам | @api.model  def get\_accounts\_by\_type(self):  accounts = self.env['investor.account'].read\_group(  [], ['account\_type'], ['account\_type']  )  return accounts |
| 11 | q\_ПолнаяИнформацияТранзакции | @api.model  def get\_transaction\_details(self):  return self.env['investor.transaction'].search\_read(  [], order='transaction\_datetime desc'  ) |
| 12 | q\_ИнвесторыСАктивнымиСчетами | @api.model  def get\_investors\_with\_active\_accounts(self):  return self.env['investor.investor'].search([  ('account\_ids.status', '=', 'active')  ]) |
| 13 | q\_ИнвесторыСПочтойGmail | @api.model  def get\_gmail\_investors(self):  return self.env['investor.investor'].search([  ('email', 'ilike', '@gmail.com')  ]) |
| 14 | q\_ИнвесторыСТойЖеДатойРождения | @api.model  def get\_investors\_same\_birthdate(self):  investors = self.env['investor.investor'].search([])  result = []  processed = set()  for i1 in investors:  for i2 in investors:  if (i1.id != i2.id and  i1.birth\_date == i2.birth\_date and  (i2.id, i1.id) not in processed):  result.append((i1, i2))  processed.add((i1.id, i2.id))  return result |
| 15 | q\_ВсеБрокерыИСчета | @api.model  def get\_brokers\_accounts\_count(self):  brokers = self.env['investor.broker'].search([])  result = []  for broker in brokers:  account\_count = len(broker.account\_ids)  result.append({  'broker': broker.name,  'account\_count': account\_count  })  return result |
| 16 | q\_ДобавитьИнвестора | @api.model  def create\_investor(self, vals):  return self.env['investor.investor'].create(vals)  # Использование:  # self.create\_investor({  # 'name': 'Иванов Иван Иванович',  # 'birth\_date': '1985-10-15',  # 'phone': '+7 (123) 456-78-90',  # 'email': 'ivanov.ii@email.com'  # }) |
| 17 | q\_ОбновитьСтатусСчета | @api.model  def close\_account(self, account\_number):  account = self.env['investor.account'].search([  ('name', '=', account\_number)  ])  if account:  account.status = 'closed' |
| 18 | q\_СписокВсехКонтактов | @api.model  def get\_all\_contacts(self):  investors = self.env['investor.investor'].search([])  brokers = self.env['investor.broker'].search([])  return {  'investors': investors.mapped('name'),  'brokers': brokers.mapped('name')  } |
| 19 | q\_СоздатьТаблицуАрхив | @api.model  def archive\_2024\_transactions(self):  # В Odoo обычно используют отдельную модель для архива  transactions\_2024 = self.env['investor.transaction'].search([  ('transaction\_datetime', '>=', '2024-01-01'),  ('transaction\_datetime', '<=', '2024-12-31')  ])  # Архивирование через создание записей в архивной модели  # или пометку существующих записей |
| 20 | q\_УдалитьТранзакции | @api.model  def delete\_small\_transactions(self):  small\_transactions = self.env['investor.transaction'].search([  ('amount', '<', 100)  ])  small\_transactions.unlink() |
| 21 | q\_СводкаПоТипамОпераций | @api.model  def get\_operations\_summary(self):  investors = self.env['investor.investor'].search([])  summary = []  for investor in investors:  accounts = investor.account\_ids  transactions = self.env['investor.transaction'].search([  ('account\_id', 'in', accounts.ids)  ])  buy\_total = sum(t.amount for t in transactions if t.operation\_type == 'buy')  sell\_total = sum(t.amount for t in transactions if t.operation\_type == 'sell')  summary.append({  'investor': investor.name,  'buy\_total': buy\_total,  'sell\_total': sell\_total  })  return summary |

## Виды запросов

|  |  |  |
| --- | --- | --- |
| № | Вид запроса | Шифр(ы) |
| 1 | **Запросы с выражениями** (Between, In, Like, статистические функции) | 3, 4, 5, 7, 8, 13 |
| 2 | **Использование логических операторов** (И, ИЛИ) | 4, 5, 9, 12, 14 |
| 3 | **Запрос с вычисляемым полем** | 9, 10, 15, 18, 21 |
| 4 | **Запрос с предикатом LIMIT** | 1, 2 |
| 5 | **Запрос с внутренним соединением по нескольким полям** | 5, 11, 14 |
| 6 |  |  |
| 7 | **Запрос с использованием внешнего соединения (LEFT JOIN)** | 15, 21 |
| 9 | **Запрос с self-соединением (самообъединение)** | 14 |
| 10 | **Подзапросы** (вложенный, коррелированный) | 6, 12 |
| 11 | **Запрос с оператором UNION** | 18 |
| 12 | **Запрос с предикатом EXISTS** | 6, 12 |
| 13 |  |  |
| 14 | **Перекрестный запрос** | 21 |
| 15 | **Запросы с параметрами** | 3, 7 |
| 16 | **Запросы на группирование данных** (итоговые значения по группам) | 9, 10, 15, 21 |
| 17 | **Запросы на удаление записи** | 20 |
| 18 | **Запросы на обновление записи** | 17 |
| 19 | **Запросы на добавление записей** | 16 |
| 20 | **Запросы на создание таблицы** | 19 |