Пояснительная записка по БД (Portfolio Risk)

Основано на фактическом коде и H2 базе проекта.

1. Назначение и пользователи

Система позволяет вести портфели, маржинальные сделки и спотовые операции. Пользователь авторизуется (`AuthController`), выбирает портфель (`PortfolioController`), открывает/закрывает сделки (`TradeController` , `TradeSellController`), вносит/выводит кэш и дивиденды (`SpotTransactionController`) и смотрит отчеты прибыли (`TradeRepository`).

2. Функциональные требования (ссылки на код)

- Пользователи: регистрация/логин, поиск по username/email — `controller/AuthController` , `service/UserService` , `repository/UserRepository` .
- Портфели: создание, выборка, фильтр по типу, деактивация — `controller/PortfolioController` , `service/PortfolioService` , `repository/PortfolioRepository` .
- Маржинальные сделки: открытие, массовый импорт, список — `controller/TradeController` (`/buy` , `/bulk-import` , `GET /trades`), логика в `service/TradeService` .
- Закрытия FIFO: `controller/TradeSellController` (`/trades/sell/fifo`), расчёты в `TradeService.fifoClose` .
- События финансирования (ставка/погашение/залог) — `model/FinancingEvent` , `FinancingEventRepository` .
- Спотовые операции: депозит/вывод/покупка/продажа/дивиденды, статистика и позиции — `SpotTransactionController` , `SpotTransactionRepository` .
- Отчеты: прибыль по тикерам и месяцам — `TradeRepository` (`findSymbolProfits` , `findMonthlyProfits`); позиции/кэш — `SpotTransactionController.getPortfolio` .
- Цены: `PriceProxyController` , `PriceService` .

3. Нефункциональные требования

- Доступ только к своим данным: контроллеры берут пользователя из `SecurityContextHolder` и фильтруют по `portfolio.user` .
- Целостность: PK/FK/UNIQUE/ENUM/NOT NULL в `documentation/tradedb_schema_only.sql` ; подтверждается аннотациями JPA в `model/*.java` .
- Транзакции: Spring Data/JPA по умолчанию оборачивает записи; массовые операции идут через сервисы.
- Производительность: H2; индексы на PK/FK и явные индексы по портфелю/символу.
- Логирование SQL: настроено в `backend/src/main/resources/application.properties` .

4. ER-сущности и связи

- `User` — 1:N `Portfolio` .
- `Portfolio` — 1:N `Trade` , 1:N `SpotTransaction` .
- `Trade` — 1:N `TradeClosure` , 1:N `FinancingEvent` .
- `TradeClosure` — закрытие части сделки.
- `FinancingEvent` — изменения ставки/погашения/залог.

- `SpotTransaction` — спотовые операции (кэш и бумаги). Диаграмма: `documentation/er_diagram.svg` (исходник `er_diagram.mmd`).

5. Зависимости и ограничения

- `users`: username → email,password,first_name,last_name,enabled; email → username,password,...; id — PK.
- `portfolios`: id → user_id,name,type,currency,is_active,...; (user_id,name) уникально.
- `trades`: id → portfolio_id,symbol,entry_price,quantity,entry_date,margin_amount,...; trade_id →> financing_events; trade_id →> trade_closures.
- `trade_closures`: id → trade_id, closed_quantity, exit_price, exit_date.
- `financing_events`: id → trade_id, event_date, event_type, rate, amount_change.
- `spot_transactions`: id → portfolio_id,ticker,transaction_type,price,quantity,amount,trade_date.
  Текстовые ограничения: сделки/события/закрытия принадлежат ровно одному родителю; количество и цены > 0; типы соответствуют ENUM; имя портфеля уникально в рамках пользователя; BUY/WITHDRAW делают amount отрицательным.

6. Нормализация

Исходная денорма (журнал) делится на: `users`, `portfolios`, `trades` + (`financing_events`, `trade_closures`), `spot_transactions`. Многозначные зависимости вынесены, атрибуты зависят только от ключей своих таблиц — 3НФ/BCNF.

7. Пример аномалии

Хранение истории ставок в `TRADES` по строке на событие дублирует поля сделки и удаляет сделку при удалении события. Разделение на `TRADES` + `FINANCING_EVENTS` убирает аномалии вставки/обновления/удаления.

8. SQL DDL (факт)

Файл: `documentation/tradedb_schema_only.sql`. Кратко:

```
CREATE TABLE users (..., username UNIQUE, email UNIQUE, enabled BOOLEAN NOT NULL);
CREATE TABLE portfolios (..., user_id REFERENCES users ON DELETE CASCADE,
portfolio_type CHECK (...), UNIQUE (user_id, name));
CREATE TABLE trades (..., portfolio_id REFERENCES portfolios ON DELETE CASCADE,
quantity > 0, entry_price > 0, margin_amount > 0, financing_rate_type CHECK (...));
CREATE TABLE trade_closures (..., trade_id REFERENCES trades ON DELETE CASCADE,
closed_quantity > 0, exit_price > 0);
CREATE TABLE financing_events (..., trade_id REFERENCES trades ON DELETE CASCADE,
event_type CHECK (...));
CREATE TABLE spot_transactions (..., portfolio_id REFERENCES portfolios ON DELETE
CASCADE, transaction_type CHECK (...));
CREATE INDEX idx_trades_portfolio ON trades(portfolio_id);
CREATE INDEX idx_trades_symbol ON trades(symbol);
CREATE INDEX idx_financing_trade ON financing_events(trade_id, event_date);
CREATE INDEX idx_trade_closure_trade ON trade_closures(trade_id);
CREATE INDEX idx_spot_portfolio ON spot_transactions(portfolio_id);
CREATE INDEX idx_spot_ticker ON spot_transactions(ticker);
```

9. SQL DML (примеры)

```sql
-- Пользователь
INSERT INTO users (username, email, password, first_name, last_name)
VALUES ('risk.officer', 'risk@example.com', '{bcrypt}', 'Ivan', 'Petrov');

-- Портфель
INSERT INTO portfolios (user_id, name, portfolio_type, currency, description)
VALUES (1, 'Margin US', 'MARGIN', 'USD', 'US equity margin book');

-- Сделка
INSERT INTO trades (portfolio_id, symbol, entry_price, quantity, entry_date,
                    margin_amount, leverage, borrowed_amount, collateral_amount,
                    maintenance_margin, financing_rate_type, financing_currency)
VALUES (1, 'AAPL', 180.25, 100, DATE '2024-03-01',
        8.50, 2.5, 18000, 7200, 25.0, 'FIXED', 'USD');

-- Смена ставки
INSERT INTO financing_events (trade_id, event_date, event_type, rate, notes)
VALUES (1, DATE '2024-03-15', 'RATE_CHANGE', 7.90, 'note');

-- Частичное закрытие
INSERT INTO trade_closures (trade_id, closed_quantity, exit_price, exit_date, notes)
VALUES (1, 40, 195.10, DATE '2024-04-05', 'partial exit');

-- Спотовые операции
INSERT INTO spot_transactions (portfolio_id, company, ticker, transaction_type,
price, quantity, amount, trade_date, note)
VALUES (1, 'Cash', 'USD', 'DEPOSIT', 1, 5000, 5000, DATE '2024-03-01', 'funding');
INSERT INTO spot_transactions (portfolio_id, company, ticker, transaction_type,
price, quantity, amount, trade_date, note)
VALUES (1, 'Microsoft Corp', 'MSFT', 'BUY', 370.5, 10, -3705, DATE '2024-03-02',
'Lot 1');

-- Прибыль по тикерам
SELECT t.symbol, SUM((t.exit_price - t.entry_price) * t.quantity) AS profit
FROM trades t JOIN portfolios p ON p.id = t.portfolio_id
WHERE t.exit_date IS NOT NULL AND p.user_id = 1
GROUP BY t.symbol ORDER BY profit DESC;

-- Месячная прибыль
SELECT FORMATDATETIME(t.exit_date, 'yyyy-MM') AS month,
       SUM((t.exit_price - t.entry_price) * t.quantity) AS profit
FROM trades t JOIN portfolios p ON p.id = t.portfolio_id
WHERE t.exit_date IS NOT NULL AND p.user_id = 1
GROUP BY FORMATDATETIME(t.exit_date, 'yyyy-MM')
ORDER BY month;

-- Позиции и кэш
SELECT ticker,
```

```
       SUM(CASE WHEN transaction_type='BUY' THEN quantity
                WHEN transaction_type='SELL' THEN -quantity ELSE 0 END) AS
position_qty,
       SUM(amount) AS cash_flow
FROM spot_transactions
WHERE portfolio_id = 1
GROUP BY ticker;

-- Исправление знаков
UPDATE spot_transactions
SET amount = -ABS(amount)
WHERE transaction_type IN ('WITHDRAW','BUY') AND amount > 0;
```

10. Транзакции (пример)

```
BEGIN;
  INSERT INTO trades (...) VALUES (...);           -- trade_id = currval(...)
  INSERT INTO financing_events (trade_id, event_date, event_type, rate)
    VALUES (currval('trades_id_seq'), DATE '2024-03-15', 'RATE_CHANGE', 7.9);
  INSERT INTO trade_closures (trade_id, closed_quantity, exit_price, exit_date)
    VALUES (currval('trades_id_seq'), 50, 195.1, DATE '2024-04-05');
COMMIT;

BEGIN;
  INSERT INTO spot_transactions (...) VALUES (...);
  INSERT INTO spot_transactions (...) VALUES (...);
COMMIT;
```

11. Нефункциональные детали и риски

- Безопасность: JWT, секрет в `application.properties` (`jwt.secret`), фильтрация по пользователю.
- Производительность: H2; для отчётов нужны индексы по exit_date и symbol (в H2 есть индексы на FK).
- Бэкапы: файлы `backend/data/*backup*.mv.db` — служебные, не включать.

12. Как использовать на защите

- Требования и связи — по разделам 1–4 с указанием контроллеров/моделей.
- Схема — `tradedb_schema_only.sql` или `er_diagram.svg` / `er_diagram.mmd` .
- Аномалия — раздел 7.
- Нормализация — раздел 6.
- Запросы — разделы 8–10 или `db_homework.md` .

13. Мини-набор запросов (коротко)

```
INSERT INTO portfolios (user_id, name, portfolio_type, currency) VALUES (1, 'Margin
US', 'MARGIN', 'USD');
INSERT INTO trades (portfolio_id, symbol, entry_price, quantity, entry_date,
```

```
margin_amount) VALUES (1, 'AAPL', 180.25, 100, CURRENT_DATE, 8.50);
INSERT INTO financing_events (trade_id, event_date, event_type, rate) VALUES (1,
CURRENT_DATE, 'RATE_CHANGE', 7.9);
INSERT INTO trade_closures (trade_id, closed_quantity, exit_price, exit_date) VALUES
(1, 40, 195.10, CURRENT_DATE);
INSERT INTO spot_transactions (portfolio_id, company, ticker, transaction_type,
price, quantity, amount, trade_date)
VALUES (1, 'Microsoft Corp', 'MSFT', 'BUY', 370.5, 10, -3705, CURRENT_DATE);
SELECT t.symbol, SUM((t.exit_price - t.entry_price) * t.quantity) AS profit
FROM trades t JOIN portfolios p ON p.id = t.portfolio_id
WHERE t.exit_date IS NOT NULL AND p.user_id = 1
GROUP BY t.symbol ORDER BY profit DESC;
SELECT FORMATDATETIME(t.exit_date, 'yyyy-MM') AS month,
       SUM((t.exit_price - t.entry_price) * t.quantity) AS profit
FROM trades t JOIN portfolios p ON p.id = t.portfolio_id
WHERE t.exit_date IS NOT NULL AND p.user_id = 1
GROUP BY FORMATDATETIME(t.exit_date, 'yyyy-MM')
ORDER BY month;
```

14. Что сдавать и ссылки

- Пояснительная записка: `documentation/db_explanatory_note.md`.
- Полный набор DML/DDL/транзакций: `documentation/db_homework.md`.
- Фактическая схема без данных: `documentation/tradedb_schema_only.sql`.
- Демо-дамп с тестовыми данными: `documentation/tradedb_schema_sample_dump.sql`.
- ER-схема: `documentation/er_diagram.svg` (исходник `er_diagram.mmd`).
- Репозиторий/ветка: `https://github.com/KJkloun/Portfolio_rick_BD` (ветка `main`).