Санкт-Петербургский национально исследовательский университет

информационных технологий, механики и оптики

Факультет программной инженерии и компьютерной техники



**Курсовая работа по дисциплине**

**«Информационные системы и базы данных»**

Этап 3.

Выполнили:

Студенты группы P33131

Осипов Василий

Ибадуллаев Алибаба

Преподаватель: Николаев В. В.

Санкт-Петербург

2022

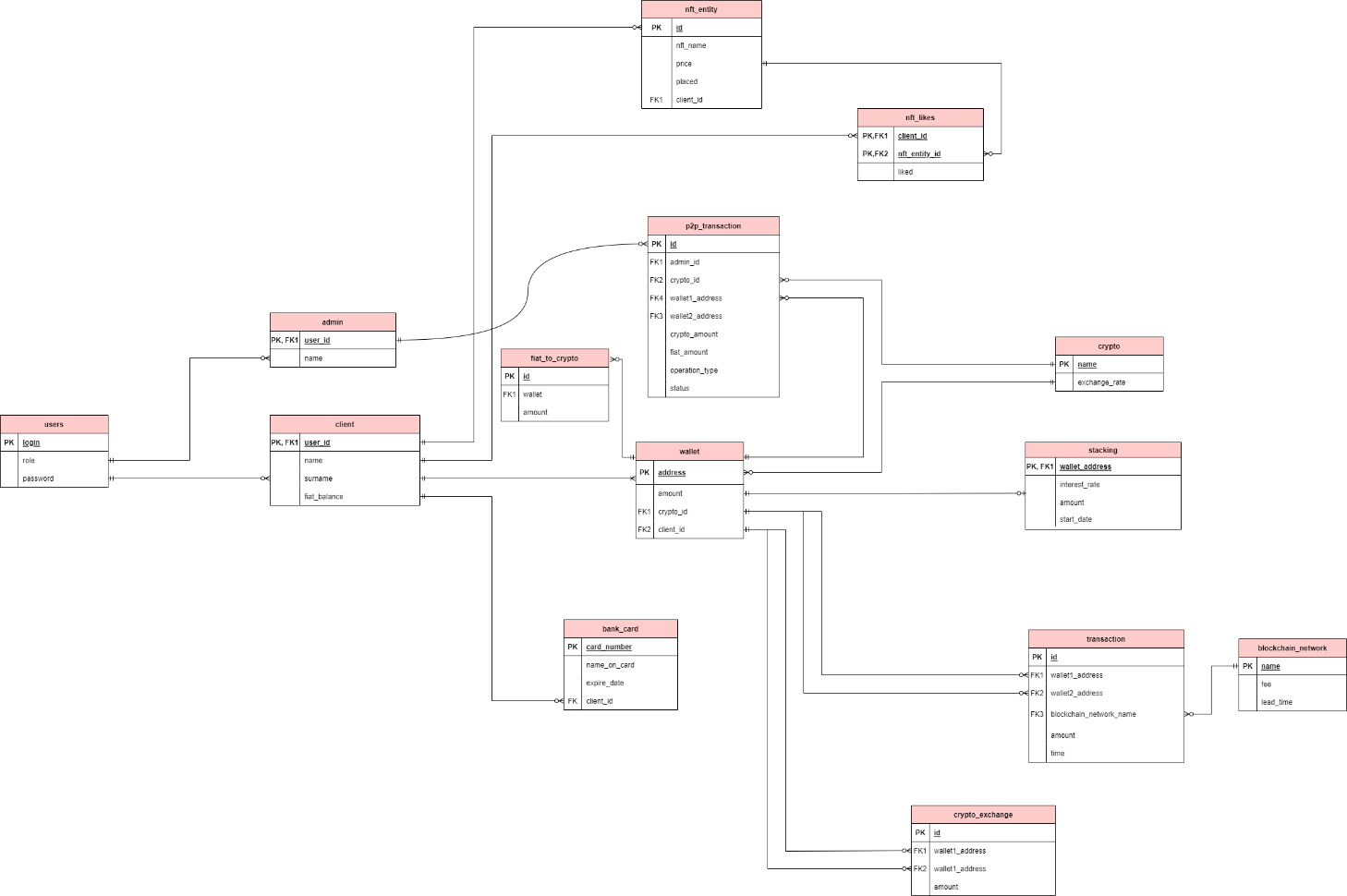
**Описание предметной области.**

В качестве предметной области выбран криптовалютный сервис, представляющий собой облегченную версию всеми знакомых бирж, в котором будет всё необходимое для рядового пользователя, который окажется перед изобилующим функционалом существующих аналогов, в которых всё так сложно и непонятно. С ростом популярности цифрового актива, людей, как правило, интересует два главных вопроса: *КАК купить* и *ГДЕ хранить* заветное цифровое золото? Тут в игру вступает предоставленная нашей системой возможность покупки криптовалюты с фиатного счёта, который легко пополняется с банковской карты. Причем пользователь может выбирать, какую валюту купить из широкого спектра поддерживаемых монет, а также, при необходимости, купить её у другого человека, посредством P2P торговли.

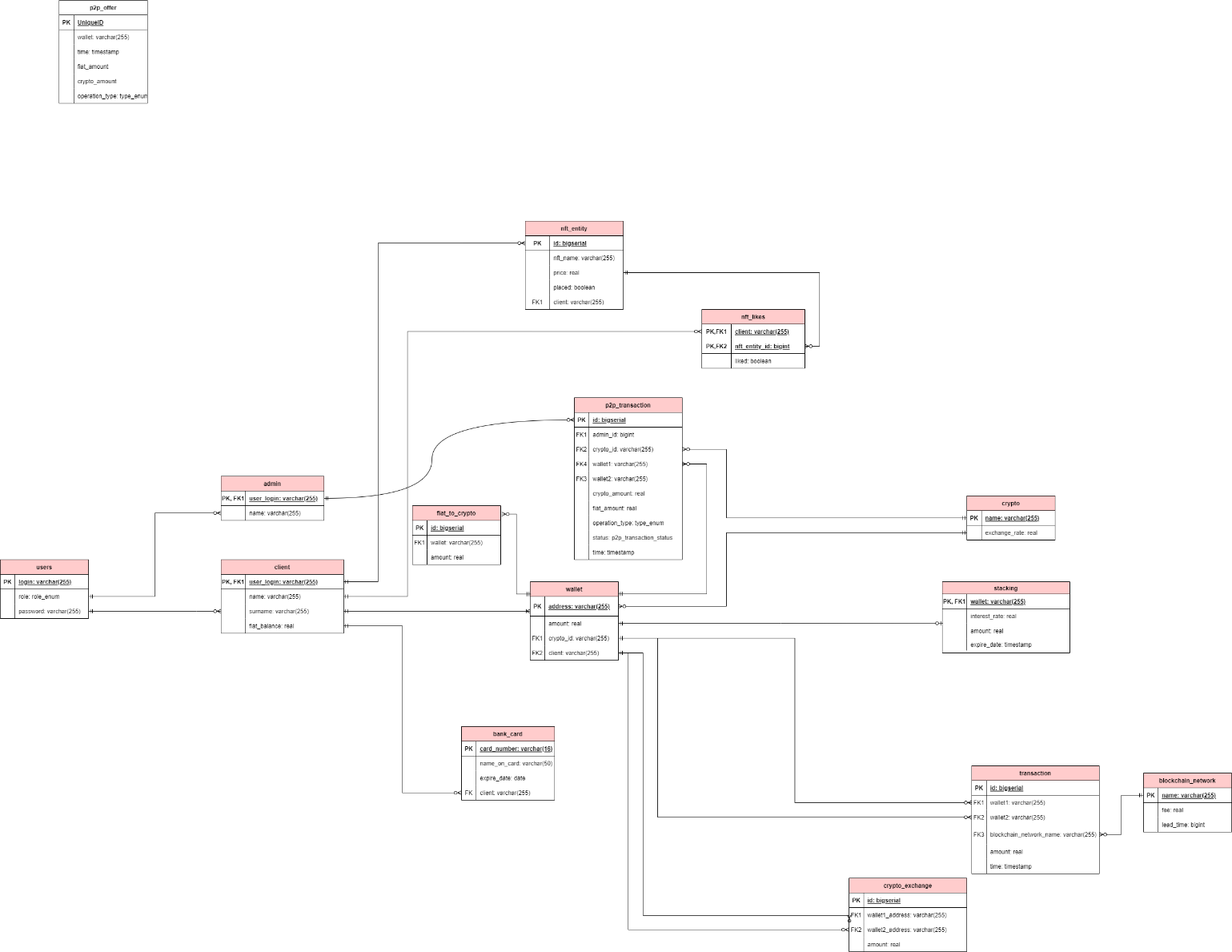
Став счастливым обладателем своих первых токенов, пользователь, почувствовавший себя частью мирового технологического прогресса, захочет воочию лицезреть краеугольный камень крипто индустрии - *транзакции*: переводы криптовалют с широким выбором блокчейн-сетей для удобства пользователей. Возможность получать криптовалюту на свой счёт или отправлять её друзьям на другие кошельки внутри выбранной сети теперь у юзера под рукой. А для предприимчивых пользователей под другой рукой будет возможность торговли криптовалютой на бирже. Трейдинг и спот-торговля перестанут выйдут из области фантастики для наших пользователей в область их влияния.

Чего не хватает нашему дорогому пользователю, уже постигшему азы блокчейн-технологий? Правильно, более глубокого погружения в мир WEB 3.0. Именно поэтому наш сервис будет обладать своей собственной торговой площадкой в сфере NFT. Хранить разные предметы из P2E, M2E секторов в одном месте и загружать свои собственные на обозрение общественности станет легко и интересно.

**Инфологическая модель.**



**Даталогическая модель.**



**Создание необходимых объектов базы данных.**

[*https://github.com/AliIbadullayev/Data-Base/blob/main/course\_work/stage3/tables.sql*](https://github.com/AliIbadullayev/Data-Base/blob/main/course_work/stage3/tables.sql)

**Заполнение созданных таблиц.**

[*https://github.com/AliIbadullayev/Data-Base/tree/main/course\_work/stage3/tables*](https://github.com/AliIbadullayev/Data-Base/tree/main/course_work/stage3/tables)

**Триггеры.**

*См. с описанием здесь:* [*https://github.com/AliIbadullayev/Data-Base/blob/main/course\_work/stage3/triggers.sql*](https://github.com/AliIbadullayev/Data-Base/blob/main/course_work/stage3/triggers.sql)

**Функции и процедуры.**

*См. с описанием здесь:* [*https://github.com/AliIbadullayev/Data-Base/blob/main/course\_work/stage3/function.sql*](https://github.com/AliIbadullayev/Data-Base/blob/main/course_work/stage3/function.sql)

**Скрипты для удаления и очистки таблиц.**

[*https://github.com/AliIbadullayev/Data-Base/blob/main/course\_work/stage3/drop.sql*](https://github.com/AliIbadullayev/Data-Base/blob/main/course_work/stage3/drop.sql)

[*https://github.com/AliIbadullayev/Data-Base/blob/main/course\_work/stage3/truncate.sql*](https://github.com/AliIbadullayev/Data-Base/blob/main/course_work/stage3/truncate.sql)

**Анализ использования созданной базы данных.**

Наиболее часто используемая информация хранится в таблицах “client”, “wallet”, “nft\_entity”. Дело в том, что ключевыми операциями в нашей предметной области будут транзакции, которые напрямую и косвенно обращаются к вышеупомянутым сущностям. Оценивание nft на маркетплейсе - ещё один ключевой бизнес-процесс, требующий большого числа запросов. Поэтому вставка новых записей чаще всего происходит в таблицы “transaction”, “p2p\_transaction”, “nft\_likes”, из которых поступают частые запросы к следующим таблицам:

1)     wallet – основной объект, к которому обращаются транзакции, как обычный перевод криптовалюты, так и p2p-транзакции, так и обмен внутри своих кошельков. Соответственно, постоянно происходит обращение и поиск кошелька с нужным адресом. В основном используется оператор “=” для поиска и операторы “<” “>” для сравнения баланса.

2)     client – не менее важный объект, на котором завязана реализация БД. Почти все действия совершаются клиентом, поэтому к этой таблице приходит много запросов по поиску нужного клиента, например обращение по внешнему ключу из wallet, для того чтобы узнать fiat balance для совершения p2p-транзакции. Так же внешний ключ на клиента содержится в таблице nft-likes, и нам нужна будет информация о том, что и какие nft оценил.

3)     nft-entity – объект, часто подвергающийся изменениям за счёт особенности системы оценивания на маркетплейсе. Когда какой-то клиент ставит оценку, то происходит обращение к этой таблице и меняется стоимость nft-сущности. Клиентов много, каждый ставит много оценок, поэтому и запросов будет немало.

**Индексы.**

*См. скрипт:* [*https://github.com/AliIbadullayev/Data-Base/blob/main/course\_work/stage3/index.sql*](https://github.com/AliIbadullayev/Data-Base/blob/main/course_work/stage3/index.sql)

**get\_client\_from\_wallet\_index**– hash индекс созданный для объекта “wallet” по атрибуту "client". Необходим для ускорения поиска принадлежности кошелька клиенту. Для реализации данного индекса лучше всего подходит hash index, т. к. запросы, которые используются у нас в функциях и процедурах используют только операнд ‘=’, а hash index как известно лучший для данного вида операций.

**likes\_index** - btree индекс, созданный для объекта “nft\_likes” по атрибутам "clent", "nft\_entity\_id". Необходим для ускорения поиска оценки конкретной nft конкретным пользователем.  Для реализации данного индекса лучше всего подходит hash index, т. к. запросы, которые используются у нас в функциях и процедурах используют только операнд ‘=’, а hash index как известно лучший для данного вида операций. Но из-за ограничения hash индекса на создание индексации по нескольким атрибутам мы решили использовать btree index. Конечно, в таком случае можно было создать два hash индекса на каждый из столбцов, но нам кажется это проиграет по времени одному btree индексу, т. к. потребуется еще память для хранения еще одного hash индекса, а также при обновлении/удалении индексируемых столбцов будут меняться значения в обоих индексах.

**nft\_entity\_index** - hash индекс, созданный для объекта “nft\_entity” по атрибуту "placed". Необходим для выяснения факта наличия nft на маркетплейсе. Для реализации данного индекса лучше всего подходит hash index, т. к. запросы, которые используются у нас в функциях и процедурах используют только операнд ‘=’, а hash index как известно лучший для данного вида операций.

**transaction\_check\_balance\_index** - btree индекс, созданный для объекта "transaction" по атрибутам "wallet1", "amount". Необходим для ускорения проверок перед совершением транзакции. При совершении транзакции каждый раз происходит сравнение баланса первого кошелька (отправителя) с суммой транзакции. Для этого используются операторы “<” , ”>”, а как известно btree индекс лучший для столбцов, участвующих в сравнении.

**p2p\_check\_index** - btree индекс, созданный для объекта "p2p\_transaction" по атрибутам "wallet1", "wallet2","crypto\_amount", "fiat\_amount". Необходим для ускорения проверок перед совершением p2p - транзакции. Аналогично с индексом для обычной транзакции, в p2p постоянно происходят операции сравнения балансов, но почему мы применили этот индекс сразу по четырём полям? Потому что в этом виде транзакций проверяются сразу два баланса, и отправителя, и получателя. Причем виды этих балансов разные, один фиатный, другой криптовалютный. Для операций сравнения лучше всего подходит btree индекс.