Экзаменационный проект по дисциплине Базы данных

студента гр. **M3437**

Комарова Андрея Игоревича

по теме

Коллекция игрока в Hearthstone

Описание проекта

Коллекция игрока в Hearthstone - есть игроки, колоды, классы (Воин, Маг..), карты, герои или скины (для мага, например - Джайна, Медив), рубашки для карт.

У игрока есть коллекция карт, рубашек и скинов. Игрок может создавать колоды для конкретного класса, добавляя туда карты из коллекции. Также игрок может из своей коллекции выбирать любимый скин (героя) для каждого класса и любимую рубашку.

Построение отношений

В результате предварительного проектирования были выделены следующие отношения:

- Player - игрок / пользователь. Username у двух пользователей может быть одинаковым, потому что пользователь уникально идентифицируется по id, и в играх от Blizzard для этого есть BattleTag <UserName>#<Id> .
- Deck - колода игрока, слабая сущность, каждая колода обязательно относится только к одному из классов и только к одному игроку, но при этом у одного игрока может быть несколько колод для одного класса. Number - номер колоды в списке колод игрока.
- Class - класс в игре (Воин, Маг..), каждый класс обязательно имеет хотя бы один скин (героя по умолчанию).
- CardClass - описывает класс карты в игре, если карта не нейтральная.
- CardType - тип карты в игре (Заклинание, Существо..).
- Rarity - редкость карты в игре (Легендарная, Обычная..).
- Card - карта в игре, описывается множеством атрибутов, характерных для карты в Hearthstone.
- CollectionCard - карта в коллекции игрока, слабая сущность. Отражает количество конкретной карты у конкретного игрока. Уникально идентифицируется парой CardId, PlayerId. Хотелось иметь ограничение на уровне БД, чтобы игрок мог добавлять в колоду карту только из своей коллекции.
- CardInDeck - описывает количество карты из коллекции игрока (CollectionCard) в колоде игрока.
- CardBack - рубашка для карт в игре.
- PlayerCardBack - описывает наличие рубашки для карт в коллекции игрока.
- Hero - герой/скин в игре. Каждый герой обязательно привязан только к одному классу.
- CollectionHero - слабая сущность, описывает наличие героя (скина на класс) в коллекции игрока. Хотелось на уровне БД иметь ограничение,

чтобы игрок мог выбрать в качестве основного героя для класса героя только из своей коллекции.

- FavoriteHero - описывает, какой герой из коллекции игрока выбран основным для класса этого игрока.

Отношение Player

Атрибуты:

- Id
- Username
- Password
- Email
- FavCardBackId
- Email

Функциональные зависимости:

- Id -> Username, Password, Email, FavCardBackId
- Email -> Id

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (Id) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. все неключевые атрибуты функционально зависят непосредственно (нетранзитивно) от ключа целиком.

Все ключи (и Id, и Email) простые, поэтому по первой теореме Дейта-Фейгина отношение находится в 5НФ.

Отношение Class

Атрибуты:

- Id
- Name
- HeroId

Функциональные зависимости:

- Id -> Name, HeroId
- Name -> Id
- HeroId -> Id

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (Id) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. все неключевые атрибуты функционально зависят непосредственно (нетранзитивно) от ключа целиком.

Все ключи (и Id, и Name, и HeroId) простые, поэтому по теореме Дейта-Фейгина 1 отношение находится в 5НФ.

Отношение CardClass

Атрибуты:

- CardId
- ClassId

Функциональные зависимости:

- CardId -> ClassId

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (CardId) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. все неключевые атрибуты функционально зависят непосредственно (нетранзитивно) от ключа целиком.

Единственный ключ (CardId) простой, поэтому по первой теореме Дейта-Фейгина отношение находится в 5НФ.

Отношение CardType

Атрибуты:

- Id
- Name

Функциональные зависимости:

- Id -> Name
- Name -> Id

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (Id) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. все неключевые атрибуты функционально зависят непосредственно (нетранзитивно) от ключа целиком.

Все ключи (и Id, и Name) простые, поэтому по первой теореме Дейта-Фейгина отношение находится в 5НФ.

Отношение Rarity

Атрибуты:

- Id
- Name

Функциональные зависимости:

- Id -> Name
- Name -> Id

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (Id) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. все неключевые атрибуты функционально зависят непосредственно (нетранзитивно) от ключа целиком.

Все ключи (и Id, и Name) простые, поэтому по первой теореме Дейта-Фейгина отношение находится в 5НФ.

Отношение Card

Атрибуты:

- Id
- Neutral
- Name
- Type
- Cost
- Rarity
- Elite
- Attack
- Health
- Description

Функциональные зависимости:

- Id -> Name, Neutral, Type, Cost, Rarity, Elite, Attack, Health, Description
- Name -> Id

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (Id) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. все неключевые атрибуты функционально зависят непосредственно (нетранзитивно) от ключа целиком.

Все ключи (и Id, и Name) простые, поэтому по первой теореме Дейта-Фейгина отношение находится в 5НФ.

Отношение CollectionCard

Атрибуты:

- Quantity
- CardId
- PlayerId

Функциональные зависимости:

- CardId, PlayerId -> Quantity

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (CardId, PlayerId) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. все неключевые атрибуты функционально зависят непосредственно (нетранзитивно) от ключа целиком.

Отношение находится в НФБК (CardId, PlayerId - надключ).

Пусть R - отношение. Докажем, что у нас нет никакой МЗ X->>Y|Z или что у нас для любых X, Y, Z не может получиться корректной декомпозиции $\pi_{X,Y}(R)$

$\bowtie \pi_{X,Z}(R)$. Мы **всегда** можем взять отношение, в котором все атрибуты из множества X равны, а из Y и Z - нет. И, сделав декомпозицию $\pi_{X,Y}(R)$, $\pi_{X,Z}(R)$ и затем natural join $\pi_{X,Y}(R) \bowtie \pi_{X,Z}(R)$ у нас получится отношение, которое будет содержать новые элементы (всевозможные пары Y, Z). Таким образом у нас нет ни одной корректной декомпозиции ⇔ нет МЗ => отношение находится в 4НФ.

Ранее были рассмотрены все МЗ, то есть смысла рассматривать ЗС *{X_1, X_2, .., X_n} с n<3 нет. Пусть X - это CardId, Y - PlayerId, Z - Quantity. Рассмотрим ЗС размера 3. $\pi_{X,Y}(R) \bowtie \pi_{X,Z}(R) \bowtie \pi_{Y,Z}(R)$. Это множественное разбиение будет корректно, при условиях, что:

- У игрока p1 есть карта c1
- У игрока p1 есть количество карты (какой-то) q1
- В количестве q1 представлена карта c1 (в коллекции какого-то игрока)

будет выполнено условие, что игрок p1 имеет карту c1 в количестве q1, что не верно. Поэтому это множественное разбиение не корректно, следовательно у нас нет ЗС (смысла рассматривать оставшиеся ЗС для n>3 нет, у нас всего 3 атрибута). Поэтому отношение находится в 5НФ.

Отношение Deck

Атрибуты:

- Number
- PlayerId
- Name
- ClassId

Функциональные зависимости:

- Number, PlayerId -> Name, ClassId

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (Number, PlayerId) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. все неключевые атрибуты функционально зависят непосредственно (нетранзитивно) от ключа целиком.

Отношение находится в НФБК (Number, PlayerId - надключ).

Пусть R - отношение. Докажем, что у нас нет никакой МЗ X->>Y|Z или что у нас для любых X, Y, Z не может получиться корректной декомпозиции $\pi_{X,Y}(R) \bowtie \pi_{X,Z}(R)$. Мы **всегда** можем взять отношение, в котором все атрибуты из множества X равны, а из Y и Z - нет. И, сделав декомпозицию $\pi_{X,Y}(R)$, $\pi_{X,Z}(R)$ и затем natural join $\pi_{X,Y}(R) \bowtie \pi_{X,Z}(R)$ у нас получится отношение, которое будет содержать новые элементы (всевозможные пары Y, Z). Таким образом у нас нет ни одной корректной декомпозиции ⇔ нет МЗ => отношение находится в 4НФ.

В отношении нет ни одной корректной множественной декомпозиции по аналогии с CollectionCard. Если мы будем проверять каждое, то столкнемся с невыполнением кольцевых ограничений => 5НФ.

Отношение CardInDeck

Атрибуты:

- DeckNo
- CardId
- PlayerId
- Amount

Функциональные зависимости:

- DeckNo, CardId, PlayerId -> Amount

Отношение находится в 1НФ, т.к. нет повторяющихся групп. DeckPlayerId и CardPlayerId - оба являются id игрока, но не повторяющейся группой, т.к. имеют разное назначение (DeckPlayerId - id игрока, в колоду которого мы добавляем карту, CardPlayerId - id игрока, карту из коллекции которого мы добавляем в колоду). Есть ключ (Id) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. все неключевые атрибуты функционально зависят непосредственно (нетранзитивно) от ключа целиком.

Отношение находится в НФБК (DeckNo, CardId, PlayerId - надключ).

Пусть R - отношение. Докажем, что у нас нет никакой МЗ X->>Y|Z или что у нас для любых X, Y, Z не может получиться корректной декомпозиции $\pi_{X,Y}(R) \bowtie \pi_{X,Z}(R)$. Мы **всегда** можем взять отношение, в котором все атрибуты из множества X равны, а из Y и Z - нет. И, сделав декомпозицию $\pi_{X,Y}(R)$, $\pi_{X,Z}(R)$ и затем natural join $\pi_{X,Y}(R) \bowtie \pi_{X,Z}(R)$ у нас получится отношение, которое будет содержать новые элементы (всевозможные пары Y, Z). Таким образом у нас нет ни одной корректной декомпозиции ⇔ нет МЗ => отношение находится в 4НФ.

В отношении нет ни одной корректной множественной декомпозиции по аналогии с CollectionCard. Если мы будем проверять каждое, то столкнемся с невыполнением кольцевых ограничений => 5НФ.

Если, какая-то корректная множественная декомпозиция все таки есть, то после декомпозиции мы потеряем ограничение, что в колоде конкретного игрока может лежать карта только из коллекции этого же игрока, поэтому в данном случае декомпозировать - это не то, что нам хочется. Так что в этом случае после нормализации следовала бы денормализация к тому виду, который есть сейчас.

Отношение CardBack

Атрибуты:

- Id

- Name
- Description

Функциональные зависимости:

- Id -> Name, Description
- Name -> Id
- Description -> Id

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (Id) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. все неключевые атрибуты функционально зависят непосредственно (нетранзитивно) от ключа целиком.

Все ключи (и Id, и Name, и Description) простые, поэтому по первой теореме Дейта-Фейгина отношение находится в 5НФ.

Отношение PlayerCardBack

Атрибуты:

- PlayerId
- CardBackId

Функциональные зависимости:

- Нет нетривиальных ФЗ

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (PlayerId, CardBackId) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. нет неключевых атрибутов.

Отношение находится в НФБК (нет нетривиальных ФЗ).

Пусть X - множество атрибутов PlayerId, Y - множество атрибутов CardBackId. R - отношение. У нас всего 2 атрибута, и R не является декартовым произведением, так как это бы подразумевало, что для каждого игрока мы всегда храним все возможные рубашки для карт в в игре, что семантически неверно, т.к. у разных игроков могут быть разные рубашки для карт в коллекции. Иными словами двум разным x1, x2 из X будут соответствовать разные y1, y2 из Y. Таким образом у нас нет МЗ $\varnothing$ ->>X|Y, следовательно, учитывая все сказанное ранее, отношение находится в 4НФ.

В отношении нет нетривиальных ЗС, т.к. у нас только 2 атрибута и одно множество X_i из любой ЗС $*\{X_1..X_n\}$ будет обязательно целым отношением, следовательно отношение находится в 5НФ.

Отношение Hero

Атрибуты:

- Id
- Name

- ClassId
- Description

Функциональные зависимости:

- Id -> Name, ClassId, Description
- Name -> Id
- Description -> Id

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (Id) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. все неключевые атрибуты функционально зависят непосредственно (нетранзитивно) от ключа целиком.

Все ключи (и Id, и Name, и Description) простые, поэтому по первой теореме Дейта-Фейгина отношение находится в 5НФ.

Отношение CollectionHero

Атрибуты:

- PlayerId
- HeroId

Функциональные зависимости:

- Нет нетривиальных ФЗ

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (PlayerId, HeroId) и все атрибуты атомарны.

Отношение находится во 2НФ и в 3НФ, т.к. нет неключевых атрибутов.

Отношение находится в НФБК (нет нетривиальных ФЗ).

Отношение находится в 4НФ, доказательство аналогично доказательству для PlayerCardBack (с точностью до переименования HeroId на CardBackId).

Отношение находится в 5НФ. Доказательство аналогично доказательству для PlayerCardBack.

Отношение FavoriteHero

Атрибуты:

- ClassId
- PlayerId
- FavHeroId

Функциональные зависимости:

- ClassId, PlayerId -> FavHeroId

Отношение находится в 1НФ, т.к. нет повторяющихся групп, есть ключ (ClassId, PlayerId) и все атрибуты атомарны.

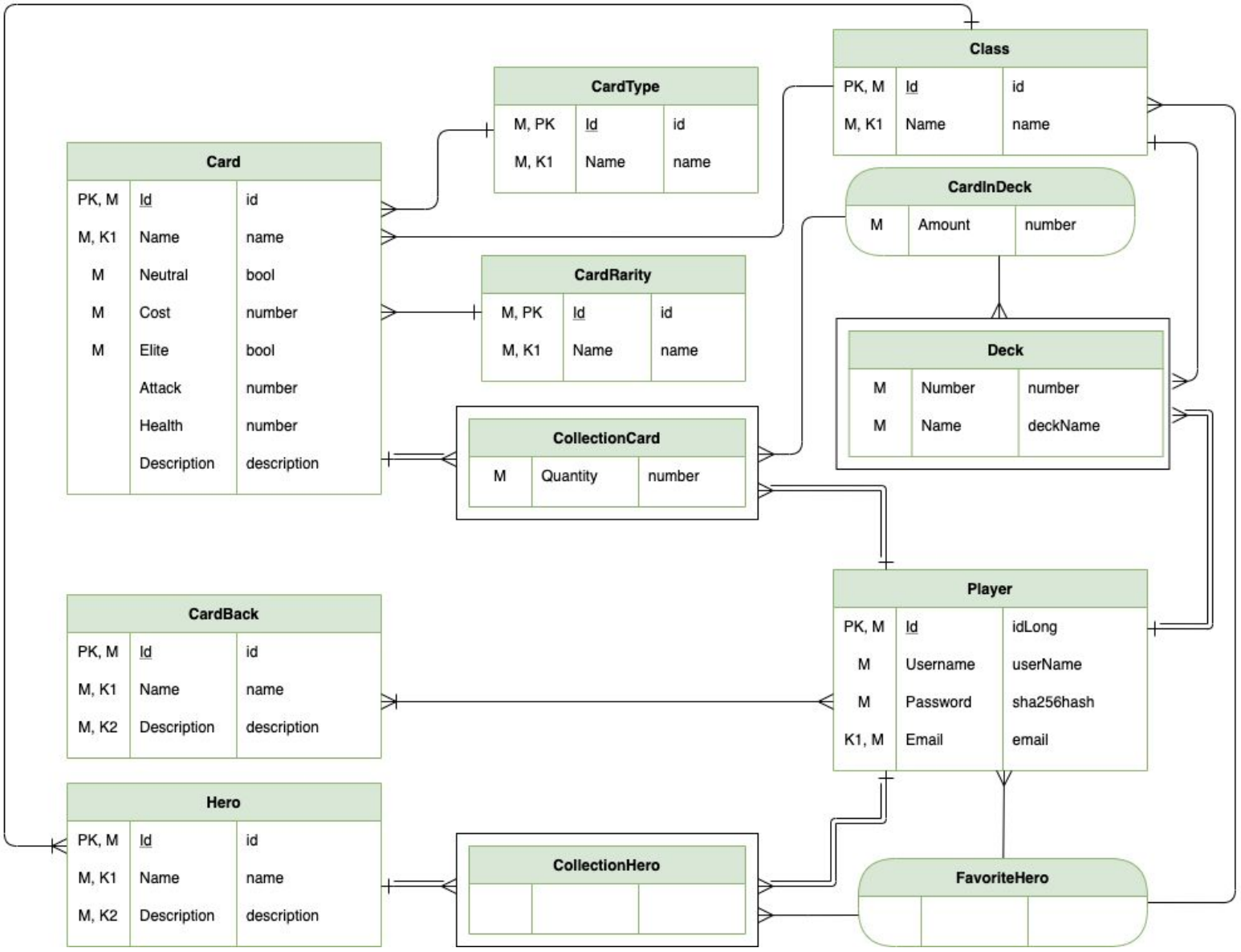
Отношение находится во 2НФ и в 3НФ, т.к. все неключевые атрибуты функционально зависят непосредственно (нетранзитивно) от ключа целиком.

Отношение находится в НФБК (ClassId, PlayerId - надключ).

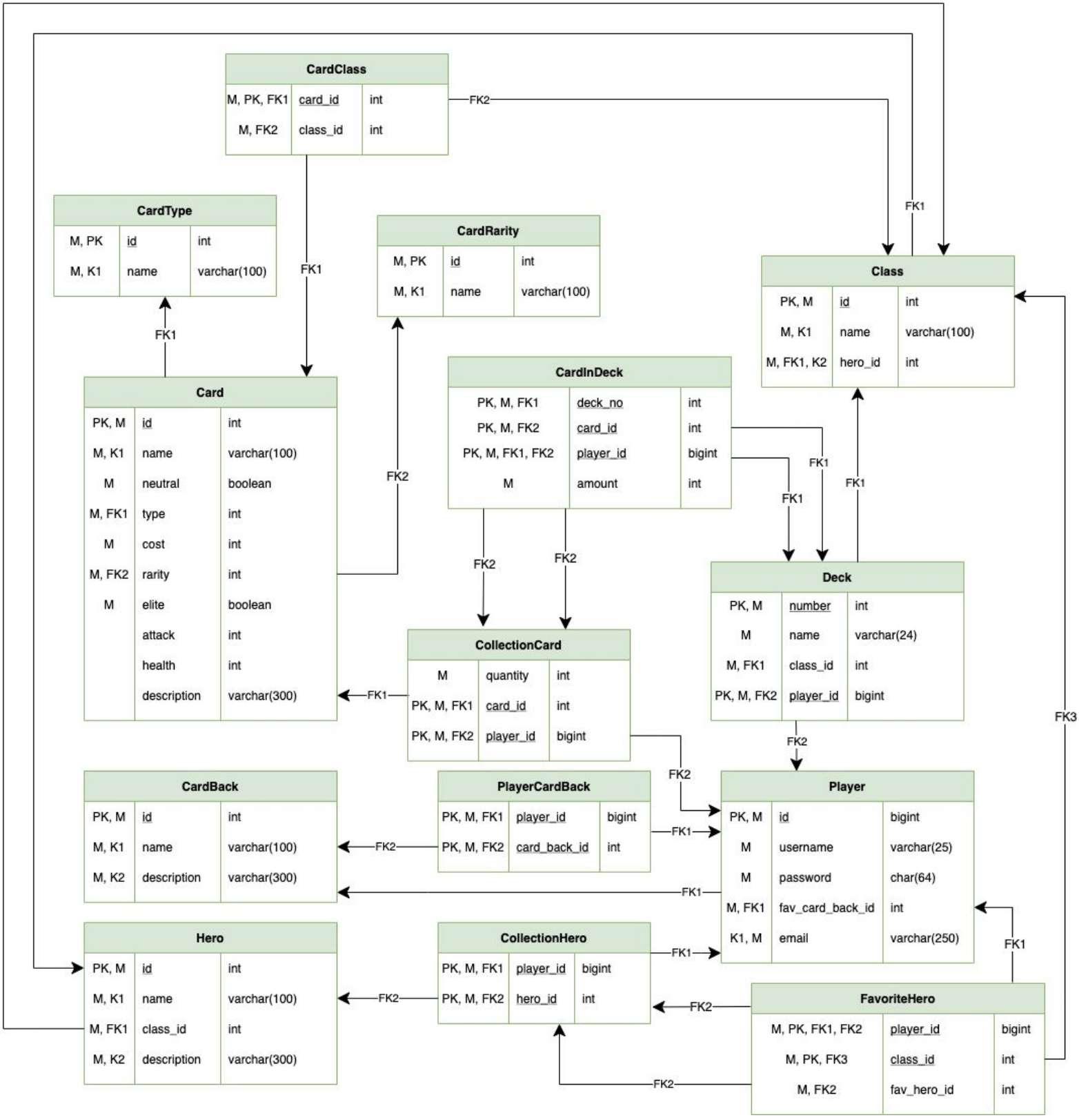
Пусть R - отношение. Докажем, что у нас нет никакой МЗ X->>Y|Z или что у нас для любых X, Y, Z не может получиться корректной декомпозиции $\pi_{X,Y}(R) \bowtie \pi_{X,Z}(R)$. Мы **всегда** можем взять отношение, в котором все атрибуты из множества X равны, а из Y и Z - нет. И, сделав декомпозицию $\pi_{X,Y}(R)$, $\pi_{X,Z}(R)$ и затем natural join $\pi_{X,Y}(R) \bowtie \pi_{X,Z}(R)$ у нас получится отношение, которое будет содержать новые элементы (всевозможные пары Y, Z). Таким образом у нас нет ни одной корректной декомпозиции ⇔ нет МЗ => отношение находится в 4НФ.

Отношение находится в 5НФ, доказательство аналогично доказательству для CollectionCard.

Модель сущность-связь


Class
PK, M Id id
M, K1 Name name
CardType
M, PK Id id
M, K1 Name name
Card
PK, M Id id
M, K1 Name name
M Neutral bool
M Cost number
M Elite bool
Attack number
Health number
Description description
CardInDeck
M Amount number
CardRarity
M, PK Id id
M, K1 Name name
Deck
M Number number
M Name deckName
CollectionCard
M Quantity number
Player
PK, M Id idLong
M Username userName
M Password sha256hash
K1, M Email email
CardBack
PK, M Id id
M, K1 Name name
M, K2 Description description
Hero
PK, M Id id
M, K1 Name name
M, K2 Description description
CollectionHero
FavoriteHero

Физическая модель



При построении физической модели использовалось следующее отображение доменов в типы:

Домен	Тип
`id`	`integer`
`idLong`	`bigint`
`number`	`integer`
`userName`	`varchar(25)`
`sha256hash`	`char(64)`
`email`	`varchar(250)`
`name`	`varchar(100)`
`deckName`	`varchar(24)`
`description`	`varchar(300)`
`bool`	`boolean`

Определения таблиц

Для реализации проекта использовалась СУБД PostgreSQL 13.0. Определения таблиц и их индексов приведено в файле `ddl.sql`.

Для таблиц Player и Class один foreign key определяется отдельно от таблиц, т.к. нельзя указать внешний ключ на таблицу, которая еще не определена.

Также в начале файла определены 3 домена: для атрибутов значения которых не могут меньше нуля, для количества карт в колоде игрока и для количества карт в коллекции игрока.

Тестовые данные

Скрипт для добавления тестовых данных приведен в файле `data.sql`.

Запросы на получение данных

В рамках проекта были реализованы следующие запросы:

- getDecksByPlayerIddAndClassId — получить все колоды игрока для определенного класса (номер, название)
- getDecksByPlayerIdAndDeckName - получить все колоды игрока с определенным именем (все атрибуты колод)
- getHeroesByClassId - получить имена и описания всех героев для определенного класса
- getCardsByType - получить имена и описания всех карт определенного типа
- getCardsByRarity - получить имена и описания всех карт определенной редкости
- EliteCards - Получить имена и описания всех элитных карт
- getCardsByAttack - получить имена и описания всех карт с определенным значением атаки
- getCardsByHealth - получить имена и описания всех карт с определенным значением здоровья
- getCardsByHealthAndAttack - получить имена и описания всех карт с определенными здоровьем и атакой
- getCardsByCost - получить имена и описания всех карт определенной стоимости
- NeutralCards - получить имена и описания всех нейтральных карт
- getCardsByClass - получить имена и описания всех карт для определенного класса
- getCardsByPlayerId - получить имена и описания всех карт игрока
- getCardBacksByPlayerId - получить названия всех рубашек для карт игрока
- getHeroesByPlayerId - получить имена всех героев в коллекции игрока
- getUsersByUsername - получить всех игроков с определенным ником (все атрибуты игрока)
- getUsersFavHeroes - получить список героев, выбранных у игрока в качестве основных для всех классов (ник игрока, имя класса, имя героя)
- getDeckCards - получить все карты из колоды игрока (имя колоды, имя карты, количество карты в колоде)
- getDeckSize - посчитать количество карт в колоде

Запросы на получение данных и вспомогательные представления приведены в файле `selects.sql`.

Запросы на изменение данных

В рамках проекта были реализованы следующие запросы:

- deleteCardFromCollection - распылить все экземпляры карты у игрока
- deleteOneCardFromCollection - распылить только один экземпляр карты у игрока
- renameDeck - изменить имя колоды
- checkFavCardBack - проверить, лежит ли любимая рубашка для карт игрока в коллекции игрока. checkFavCardBackTrigger - соответствующий триггер
- editCardBack - изменить любимую рубашку
- editFavHero - изменить основного героя для класса
- deleteOneCardFromCollection - удалить один экземпляр карты из колоды
- checkCardClassInDeck - проверить, соответствует ли карта в колоде классу колоды. checkCardClassInDeckTrigger - соответствующий триггер
- checkFavHeroClass - проверить, есть ли новый герой коллекции игрока и соответствует ли он своему классу. checkFavHeroClassTrigger - соответствующий триггер

Запросы на изменение данных, хранимые процедуры и триггеры приведены в файле `updates.sql`.