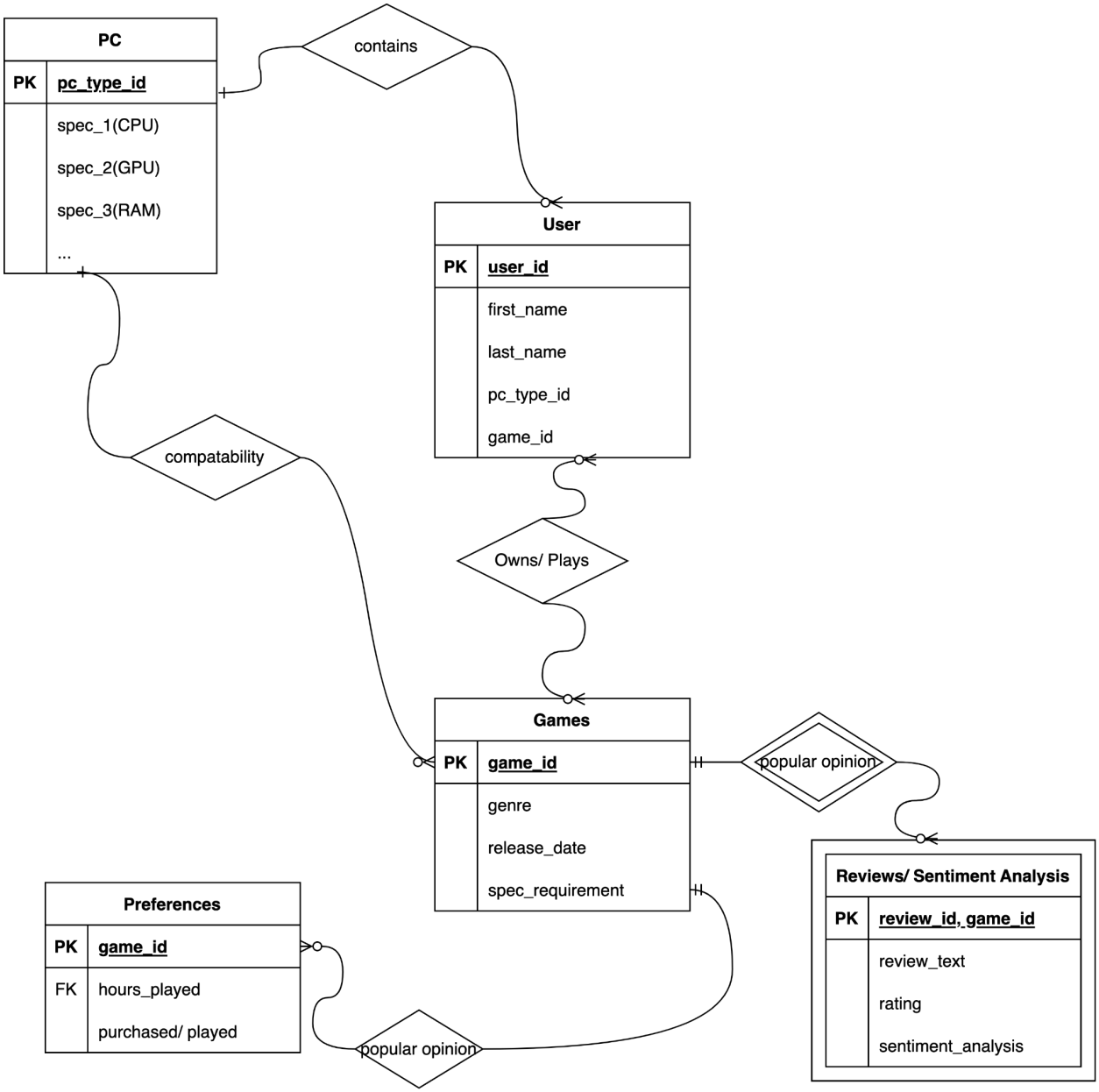

Assumptions:

The user and game entity are the key to our web application GameFinder. The user creates an optional many to one relation with the PC database. This allows users to check if their computer has the correct specs to play a game of their choice. The PC specs are also there to check compatibility with the Games themselves. This way users are able to choose games of interest that work well on their device. Similarly, the Games table holds the list of all the games that are on Steam's platform. From here, users in an optional many to optional many are able to browse and select games of interest to them. The Reviews

table is a populated table with many reviews of the top games. This is a weak entity of the Games table as the reviews are reliant on the game itself. Alternatively, the Preferences table shows popular opinion games. We plan to highlight games with high hours played and compare that against game specs to create a comprehensive view for the users to play around with.

Similarly, the Reviews table is a weak entity of the User. This mandatory one to optional many allows the user to write reviews about games that they have played. The reviews table in turn is directly related to the user table. Continuing on, the User to Contact entity highlights the final weak entity relationship. This optional one to optional many highlights that the user can choose to contact the admin (us) if they choose about an issue/ game related content as few or as many times as they want. Lastly, the User to Games has a optional many to optional many. This is because the user can look through games of interest to them, including games they might own. Similarly, the games themselves are not tied to any specific User. The games can be owned by multiple users and vice versa.

The review database will have pre-added reviews related to steam games from a separate database, with options for users and admins to add/ delete users. Similarly the games entity has thousands of pre-loaded games available on Steam. Lastly, the Preferences entity also contains built in purchase models of previous games and user preferences.

Translated Model/ Relational Schema:

```
PCs(pc_type_id: INT [PK],
spec_1(CPU): INT,
spec_2(GPU): INT,
spec_3(RAM)
)

Contacts(email: VARCHAR(255) [PK],
       user_id: INT [FK to Users.user_id],
       message: VARCHAR(255)
)

Users(user_id: INT [PK],
first_name: VARCHAR(255),
last_name: VARCHAR(255),
pc_type_id: INT,
```

```
game_id: INT
)

Games(game_id: INT [PK],
 genre: VARCHAR(255),
release_date: DATE,
spec_requirement: VARCHAR(255)
)

Preferences(preference_id: INT [PK],
user_id: INT [FK to Users.user_id],
genre: VARCHAR(255),
price_min: INT,
price_max: INT
)

Reviews(review_id: INT [PK],
user_id: INT [FK to Users.user_id],
review_text: VARCHAR(255),
game_id: INT, rating: FLOAT(2)
)

Owns(game_id: VARCHAR(255) [PK],
user_id: INT [PK]
)
```