**Group 7 - RDBMS Designed for Beginners about Hongkai: Star Rail**

**1.Background**

Our topic is a relational database designed for beginners to learn about the world-famous game Hongkai: Star Rail, which was awarded as the best mobile game at 10 am (UTC+8) on Dec 8. It is the first Chinese game to win this award. It is a turn-based game, so the up-to-date static data is important during the gameplay.

The initial motivation for choosing this topic was the data bank(**Figure 1**) inside the game, which collects the information we have encountered in different terms. It looks like a guidebook. However, the information in the data bank is insufficient, as many of the terms do not help the players understand how everything should be developed and lots of basic data is not shown there. Only a small part of it is useful. Therefore, we need to identify the data we really need to build a clear structure of the scattered data. And that is the combat-related data, because the whole game is based on developing characters and defeating enemies.



Figure 1. Data Bank

**2.Conceptual design(a lookup table for nouns in the game is shown in part 7.)**

When we start to get familiar with this game, the panel data appears most frequently, which are static or can be modified outside the combat. We only consider this kind of data in this project. The whole ER diagram is shown in **Figure 2**.

There are many words which are hard to understand, and we will explain them by entity.

There are two entities with sub-types, shown in **Figure 3**, but the respective sub-types can still exist in the same table and because they are disjoint, we only need one more attribute to distinguish them. Here the word limit means that this character or light cone only appeared during a specific period.

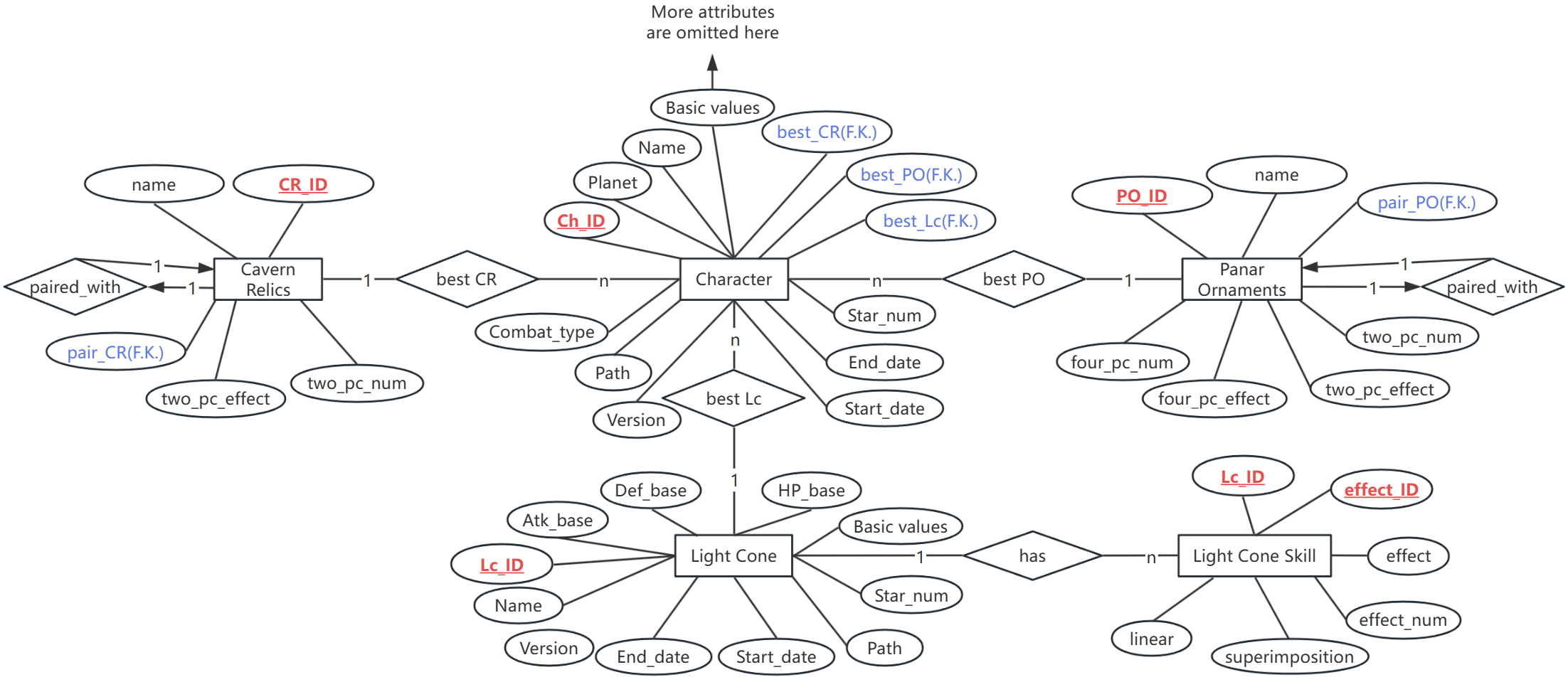


Figure 2. ER

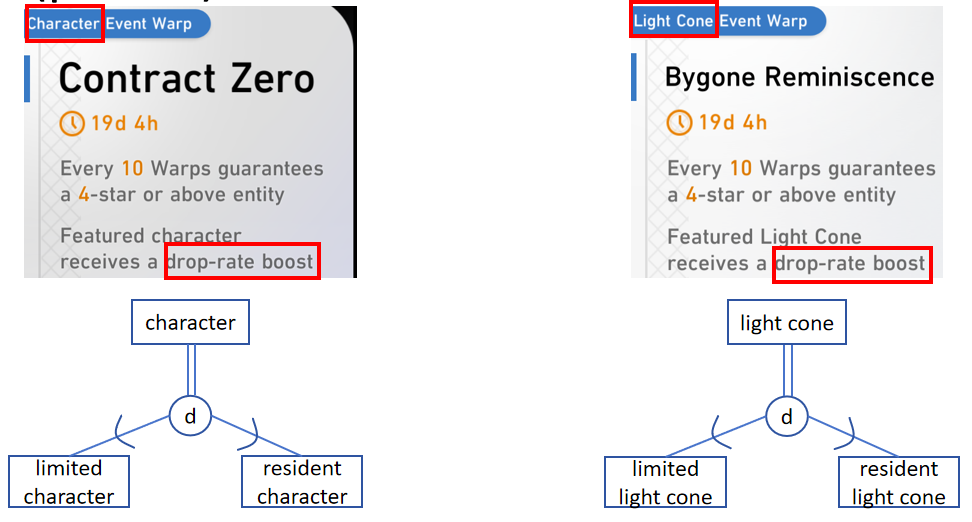


Figure 3. EER

**3.Logical design**

Here we present the main features of a certain character in **Figure 4**. Please note that there are three foreign keys of a character, which denote some matches with their suitable weapons and equipment. These matches are based on the experience of the players along with precise calculation and they are highly reliable. Since all other attributes can be uniquely determined by the primary key, it strictly follows BCNF.

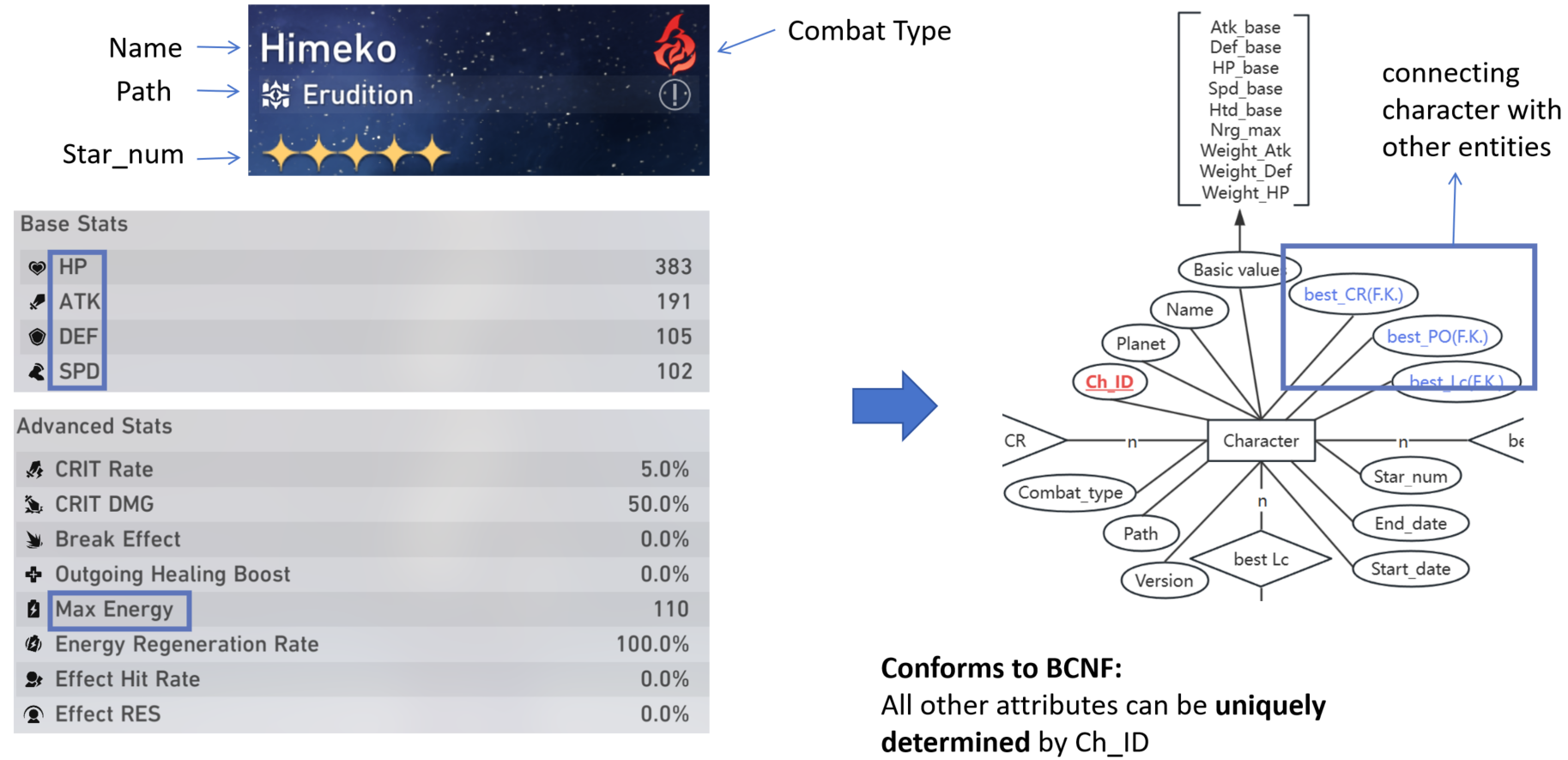


Figure 4. Character

Light cone can be interpreted as weapon. Many of its basic features are similar to that of characters. However, when it comes to describing the skills, we encountered one problem that the ID of one light cone cannot determine other attributes(**Figure 5**) because there are more than one effect of a light cone.

Here we broke them down in two steps, considering different effect and splitting direction and number.

**Step 1**: split each of the effect;

**Step 2**: split the number and direction of one effect.

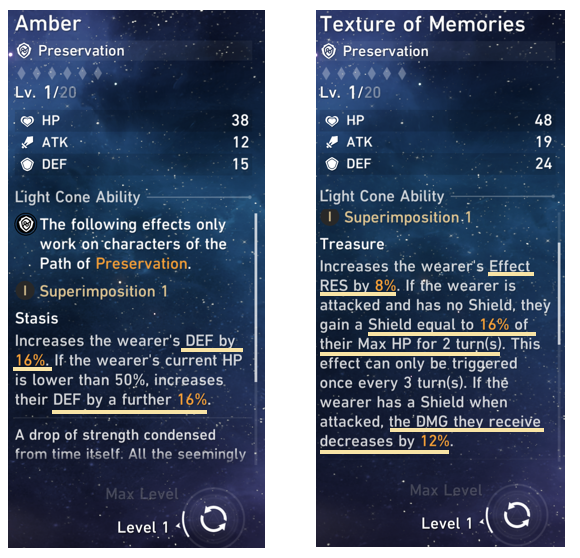


Figure 5. Light cones may have different number effects

The processing method is shown in **Figure 6**. Since the basic attributes of light cones are relatively independent, we treat them as a separate entity and put the skills to another. There is no doubt that it can be done within only **one** entity but we prefer to have a better view of the features so we choose this form. The superimposition in the lower left corner means the enhancement level of the light cone and determines the number of each effect so we listed the number by level.

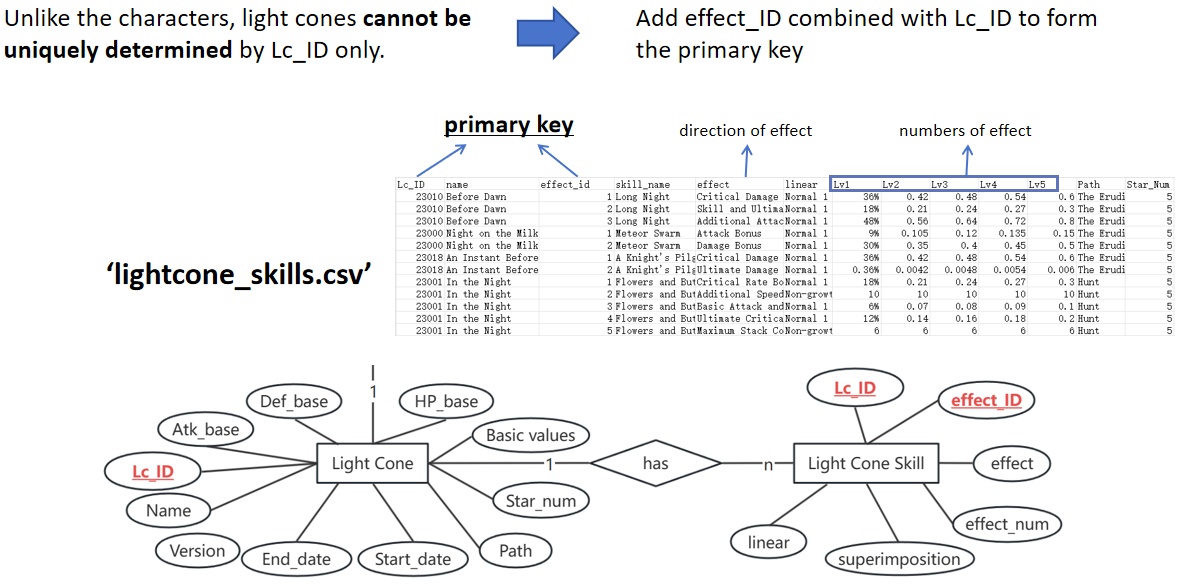


Figure 6. Processing method of light cones

The relics and ornaments can be interpreted as equipment. A set of relics contains 4 pieces and a set of ornaments contains 2 pieces. We found that most of the effect of the 4-pc relics are difficult to encode because they consist of some if-else judgement so it should be implemented together with many more complicated issues like time series so we only include the 2-pc effect of relics(**Figure 7**). In terms of ornaments, there are always two effects brought by one set so we divide them into 4 columns(**Figure 7**), similar to the break down idea of light cone.

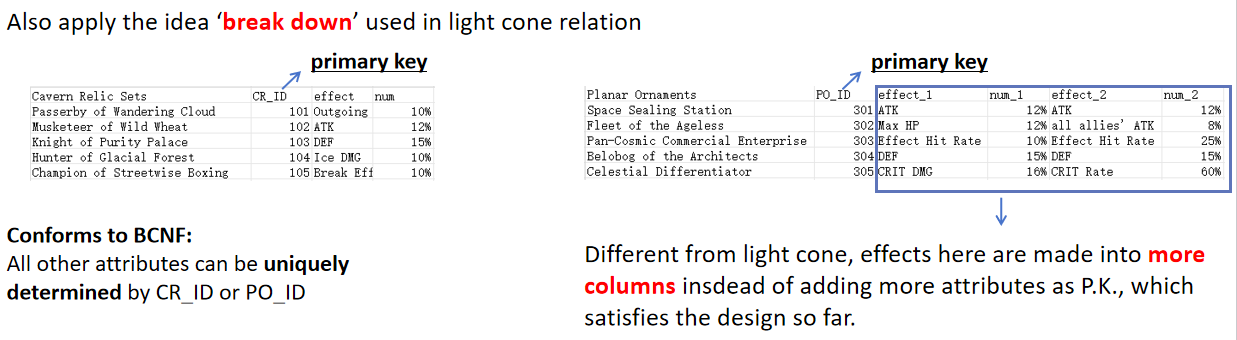


Figure 7. Relics & Ornaments

**4.Data and implementation**

Let's now discuss the implementation of establishing the database. We'll start with data collection. We collected data using web crawlers, specifically from the official website <starrailstation.com>. The website provides game-related data. As an example, consider this screenshot of a data page for a game character on the website. Our objective was to extract the relevant data within the red box(Figure 8). For web scraping, we utilized the Selenium and BeautifulSoup (bs4) modules.

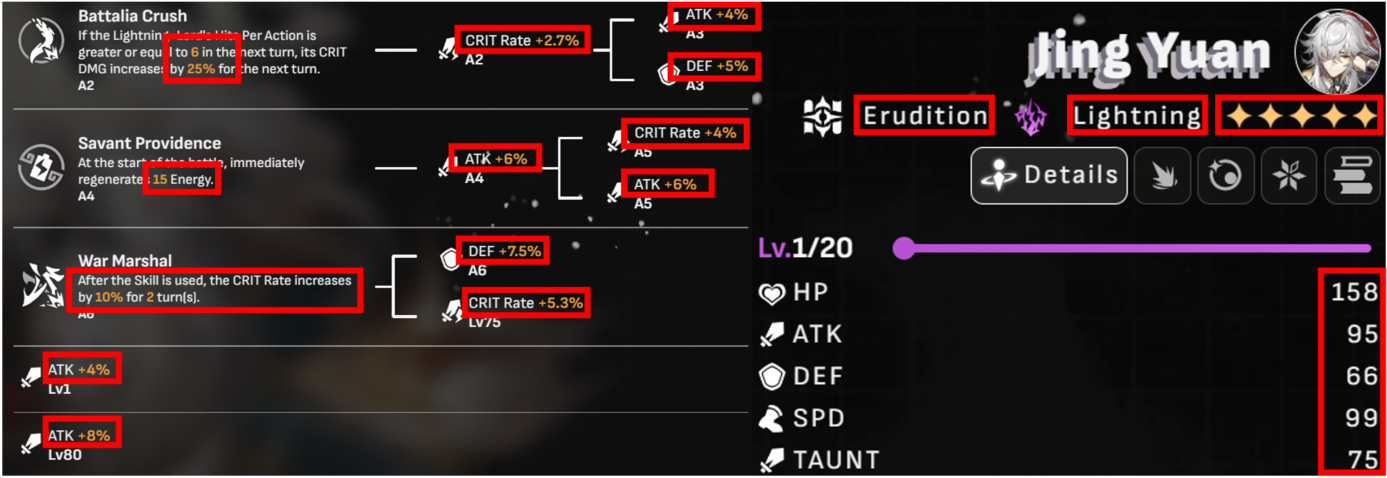


Figure 8. The source website of all the data

Based on the ER diagram, we crawled and collected data from five categories, storing each category as a CSV file. Next, we import the data. All operations were performed using Workbench. First, we created a database named "railway" and then imported the CSV files one by one using the "Table Data Import Wizard" feature. During the import process, we specify the data types for each column.

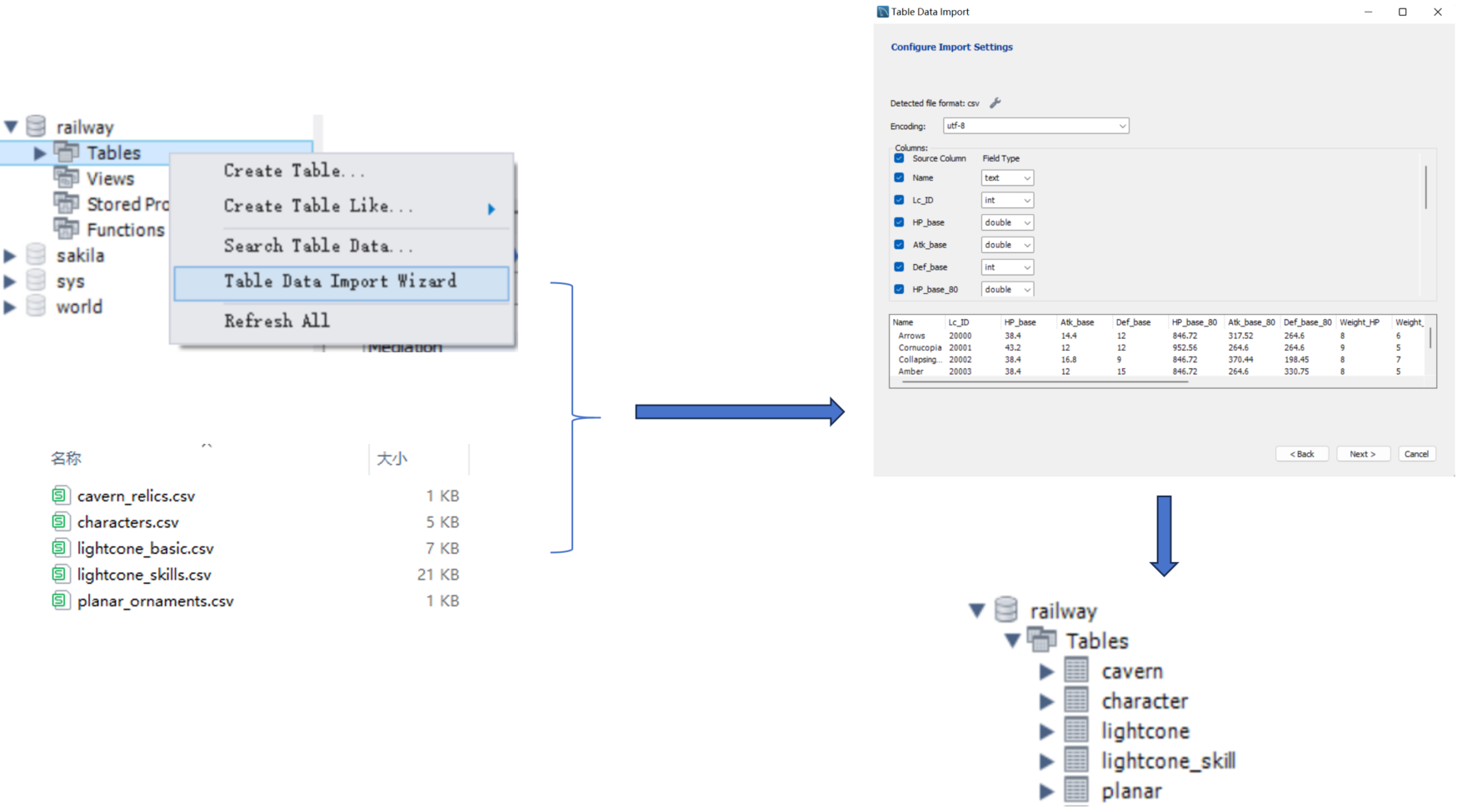


Figure 9. Data import

Next, we applied constraints to each table, including primary key and foreign key constraints(**Figure 10**). Adding constraints was not always a straightforward process; we encountered errors where primary keys or foreign keys couldn't be set. This was usually due to missing data or formatting errors. In such cases, we reviewed the data, performed data cleansing, and attempted again. We repeated this process until the constraints were successfully set.

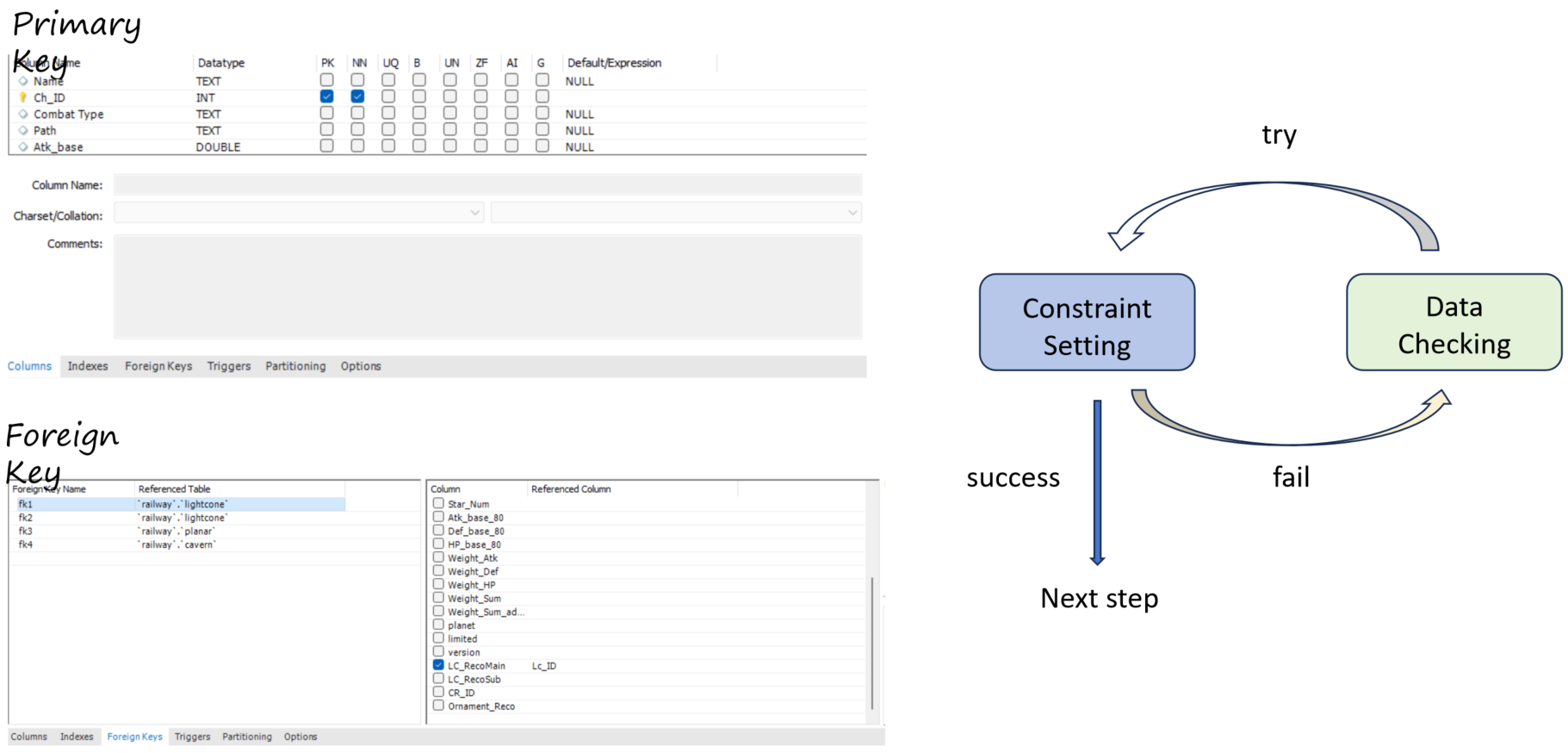


Figure 10. Constraints

At this point, the database establishment is considered complete. Here is the EER diagram generated by Workbench(**Figure 11**). It provides a clear view of the connections between the tables, with the constraints correctly set.

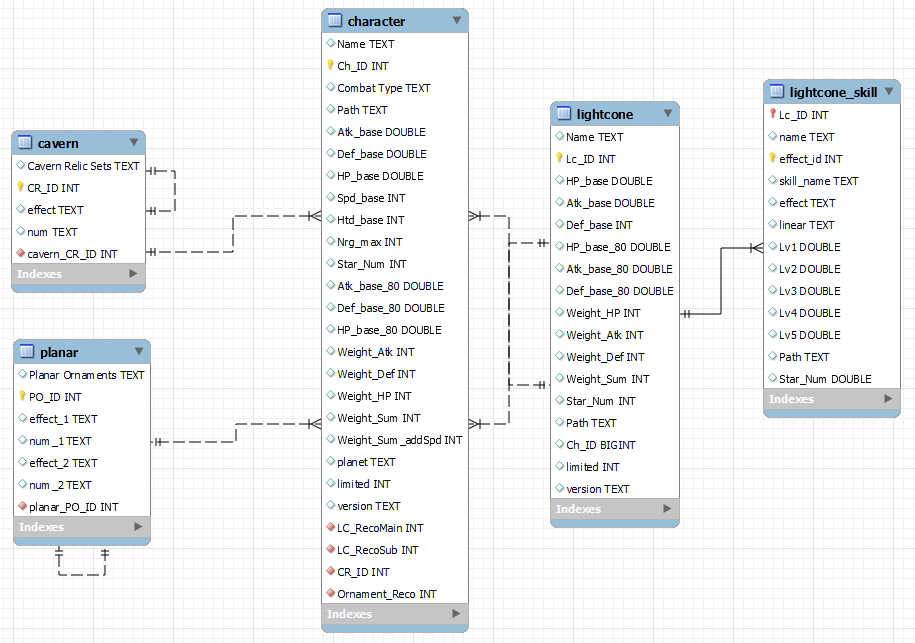


Figure 11. EER generated by workbench

**5.Potential queries**

Finally, let's discuss some practical use cases of the database. We have selected three examples(**Figure 12**) that new players may encounter in actual gameplay scenarios: "How to improve critical damage?", "Which suits can be obtained together?", and "What's the recommended match for a character?" By performing queries, users can easily get answers. We want to use these examples to demonstrate how our database can better assist new players in getting started and enjoying the game.

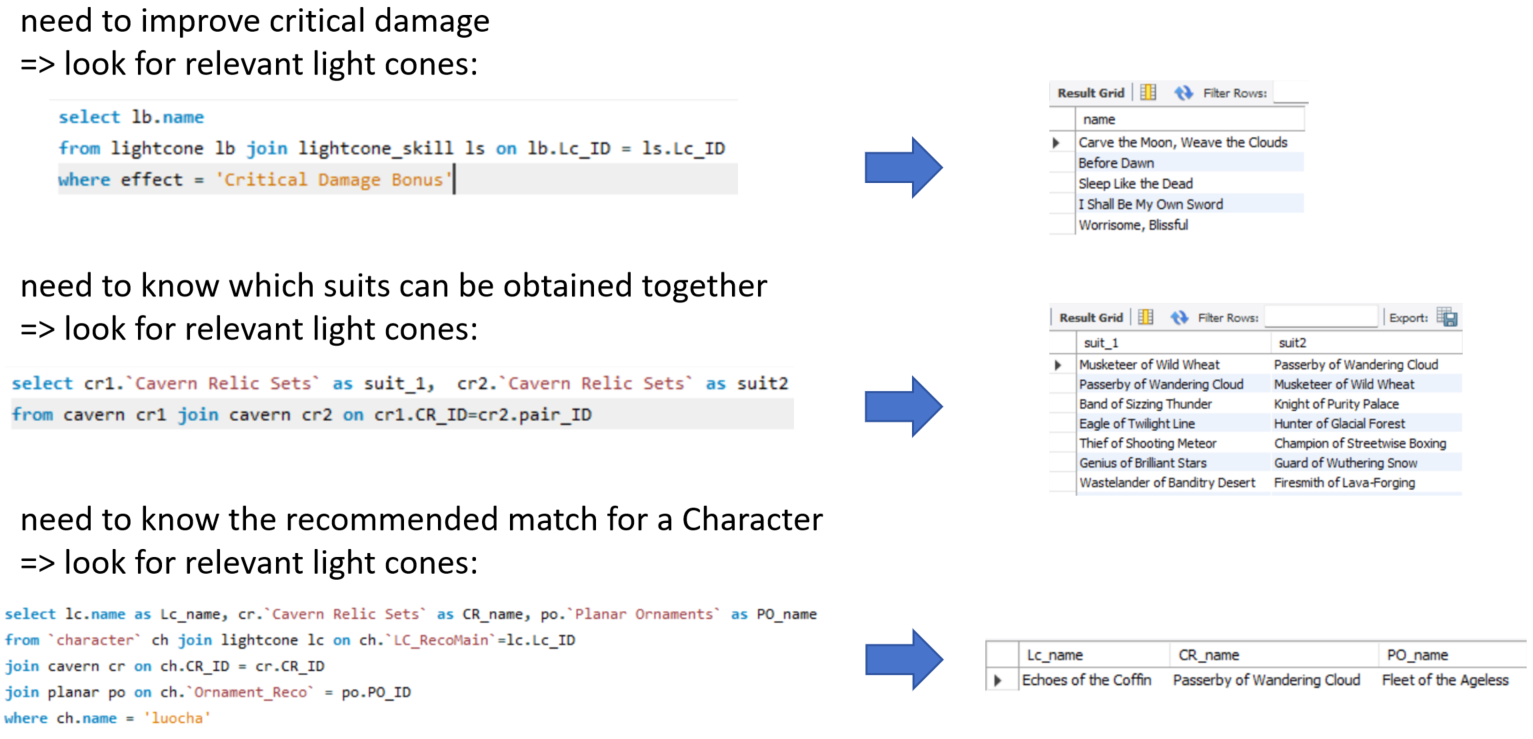


Figure 12. practical queries

**7.Lookup table for game-specific nouns**

|  |  |
| --- | --- |
| **Nouns in game** | **Interpretation** |
| Light cone | Weapon, one character can only equip with one light cone |
| Relics | Equipment set, one character can equip at most 4 pieces |
| Ornaments | Equipment set, one character can equip at most 2 pieces |
| Combat Type | The element of a character, such as ice, fire, etc. |
| Path | Describing the aptitude of a character |
| Star\_num | Indicating how precious a character or light cone is |

**8.Division of cooperation**

|  |  |  |
| --- | --- | --- |
| **Student name** | **Student number** | **work** |
| Yidong Lyu | 222041019 | Ligical & conceptual design, presentation |
| Yang Yang | 222041011 | Implementation, presentation |
| Youyi Wei | 222041018 | Data pre-processing and formatting |
| Jiale Yu | 222041021 | Logical design, implementation |
| Hongxuan Li | 222041047 | Data crawling, data pre-processing |
| Xiaoyu Song | 222041046 | Conceptual design, DB integrity checking |