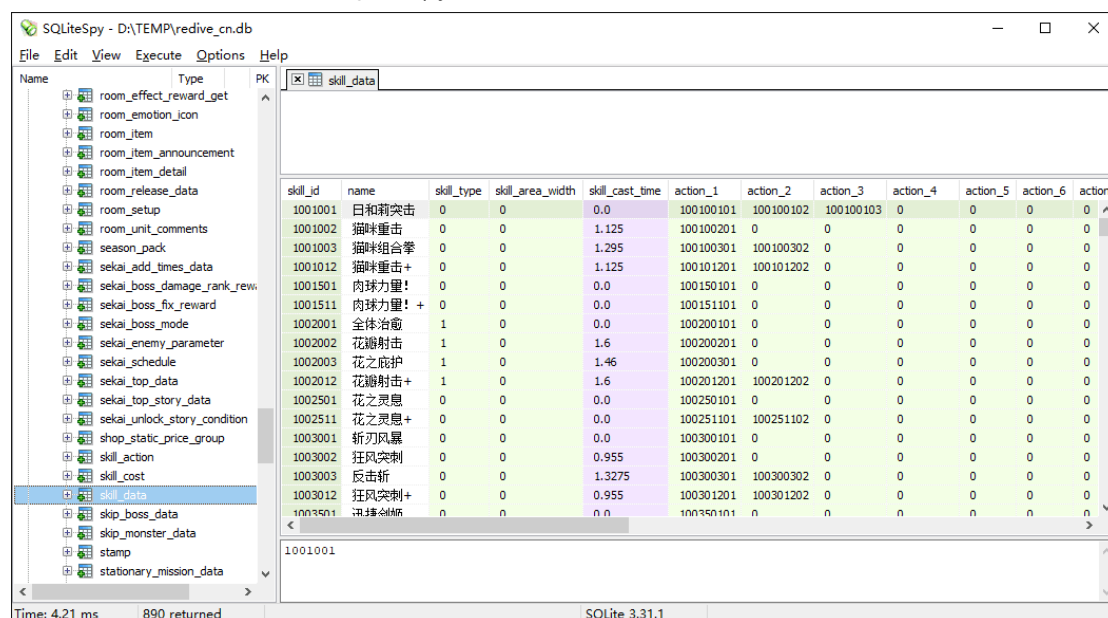


## 公主连结时间轴相关数据分析 ver1.2

各位大佬大家好呀，这里是 [zzpong](#)。熟练使用各种语言的大家可能更习惯于从代码角度分析数据的运行方式，但身为一个在 NS 游戏里拆包折腾了大半年的人来说，我更习惯于单从游戏数据的规律分析内容。毕竟并非所有程序都有人性化的拆包结果，如果不能在线调试，那反汇编伪代码的读取更是难于登天。所以在这先提供大家一些我自己从数据端研究出的规律（虽然会在数据间交叉验证，但确信度要低于直接阅读代码所获得的规律），权当抛砖引玉。文中一切数据来源皆出自“干炸里脊资源站”，网址在[这儿](#)。

**想要获取数据**，有两种方式——读取 database 文件与下载 github 上的 sql 文件。

读取 database 文件请使用这个[网址](#)，并下载对应服务器 database。请注意，下载的 redive\_cn.db.br 文件为 brotli 压缩后的文件，需要解包才可以使用，解包网站在[这儿](#)。在解包好以后，你就可以使用“SQLiteSpy”读取 db 文件了，这是群大佬 **mozzit** 所提供的截图：



skill_id	name	skill_type	skill_area_width	skill_cast_time	action_1	action_2	action_3	action_4	action_5	action_6	action_7
1001001	日和莉突击	0	0	0.0	100100101	100100102	100100103	0	0	0	0
1001002	猫咪重击	0	0	1.125	100100201	0	0	0	0	0	0
1001003	猫咪组合拳	0	0	1.295	100100301	100100302	0	0	0	0	0
1001012	猫咪重击+	0	0	1.125	100101201	100101202	0	0	0	0	0
1001501	肉球力量!	0	0	0.0	100150101	0	0	0	0	0	0
1001511	肉球力量! +	0	0	0.0	100151101	0	0	0	0	0	0
1002001	全体治愈	1	0	0.0	100200101	0	0	0	0	0	0
1002002	花瓣射击	1	0	1.6	100200201	0	0	0	0	0	0
1002003	花之庇护	1	0	1.46	100200301	0	0	0	0	0	0
1002012	花瓣射击+	1	0	1.6	100201201	100201202	0	0	0	0	0
1002501	花之灵息	0	0	0.0	100250101	0	0	0	0	0	0
1002511	花之灵息+	0	0	0.0	100251101	100251102	0	0	0	0	0
1003001	斩刃风暴	0	0	0.0	100300101	0	0	0	0	0	0
1003002	狂风突刺	0	0	0.955	100300201	0	0	0	0	0	0
1003003	反击斩	0	0	1.3275	100300301	100300302	0	0	0	0	0
1003012	狂风突刺+	0	0	0.955	100301201	100301202	0	0	0	0	0
1003501	连续剑斩	0	0	0.0	100350101	0	0	0	0	0	0

或者你也可以使用附件里青はやし所提供解压后的 redive\_cn.db 文件。

在上述[网址](#)的另一个位置，则是角色预制的 prefab 文件 PuriCone unit prefab dump。什么是 prefab 文件可以参考[这儿](#)。该文件为每个角色的相关战斗信息，包括普通攻击、ub、技能等等，之后将详细介绍，其中 UNIT\_100101 的数字代表角色 id（如果猜错的话（应该不可能不是），请进入文件搜索“ActionId”，这个对应角色技能，如此一来你就可以知道文件数据是哪个角色的了）。当然，你也可以直接看添加了角色名字的“trans.json”文件，文件在附件中，感谢[我爬了](#)所提供资源。

直接下载 github 上的 sql 文件则是另一种获取数据方法。优点是下载就能用，推荐用 VS Code 打开；缺点就是文件数据为日文版。这个链接可以通过“干炸里脊站”的[订阅数据库更新](#)进入，或者直接打开这个[网址](#)获取。

**粗浅的数据分析**如下：以下数据仅是根据各数据文件及游戏中角色的实际反映所进行的猜测，并不一定准确，请仅作为参考辅助。因为作者是个铁憨憨，不想做 br 解压缩，所以就直接去 github 上下载 sql 数据了，根本没注意到群里有解压好的文件。请各位尽量阅读

redive\_cn.db 以获得更好的用户体验。如果实在想读取 sql，推荐超好用的 VS Code，一键加载文件夹，文件间切换秒读取，真的是



下面将分别对摸轴比较的文件进行说明：

#### -----角色 ID-----

**actual\_unit\_background.sql** 角色及其背景，可能用于九宫格展示。

```
/*unit_id*/100231,
```

角色 id（唯一性，涉及数据调用），

```
/*unit_name*/"草野 優衣",
```

角色名

```
/*bg_id*/510220,
```

角色背景（比如九宫格角色背后的 xx 公会、xx 森林，所以 id 才会有大量重复）

```
/*face_type*/2
```

face type

#### -----角色基础与普通攻击-----

**unit\_data.sql** 角色基础信息相关，以莫妮卡为例：

```
/*unit_id*/105301,
```

莫妮卡 id（比 **actual\_unit\_background.sql** 更精准，**推荐以后从这儿查找角色 id**）。

```
/*unit_name*/"モニカ", /*kana*/"もにか",
```

角色平、片假名。

```
/*prefab_id*/105301,
```

预制动作 id，可用于**查询 prefabs.zip**，或者直接用现成的 **trans.json（附件）**。

```
/*is_limited*/0,
```

是否为限定角色。

```
/*rarity*/3,
```

稀有度/初始星数。

```
/*motion_type*/4,
```

行为模式，可能与工会小屋有关。

```
/*se_type*/4,
```

sound effect, 声音效果。

```
/*move_speed*/450,
```

角色移动速度。

```
/*search_area_width*/410,
```

角色索敌范围。

```
/*atk_type*/1,
```

角色攻击类型, 1 近战, 2 远战, 0 预留。

```
/*normal_atk_cast_time*/2.24,
```

普通攻击等待时间。

```
/*cutin_1*/105301, /*cutin_2*/0, /*cutin1_star6*/105301, /*cutin2_star6*/0,
```

切入动画。

```
/*guild_id*/12,
```

所属工会 id。

```
/*exskill_display*/0,
```

是否显示 exskill。

```
/*only_disp_owned*/0,
```

是否仅在拥有后显示在 box 中。如为 1, 则“未获得角色”中不会显示该角色。

对了, 上面的文件不仅可以让你查看角色信息, 还可以让你看到将要实装的角色信息 (没错, 这就是好多游戏有人拆包在官方前告诉玩家新活动的方法), 以及剧情角色信息。如何获取呢? 请搜索:

```
/*search_area_width*/0,
```

并查看其对应

```
/*move_speed*/
```

如果非零, 比如

```
/*move_speed*/600,
```

则其为“剧情角色”, 比如:

```
/*unit_id*/900102, /*unit_name*/"ヒヨリ", /*kana*/"ひより
```

```
", /*prefab_id*/900102, /*is_limited*/0, /*rarity*/1, /*motion_type*/0,
```

```
/*se_type*/1, /*move_speed*/600, /*search_area_width*/675, /*atk_type*/
```

```
1, /*normal_atk_cast_time*/50.0,
```

游戏开始剧情里的远距离攻击日和 (虽然没打拳就倒了。也从侧面表明角色没有站位数据, 只有“索敌范围”);

如果为零, 比如:

```
/*move_speed*/0,
```

则其为预装角色信息, 比如:

```
/*unit_id*/110201, /*unit_name*/"ミサキ (サマー)", /*kana*/"みさき
```

```
", /*prefab_id*/110201, /*is_limited*/0, /*rarity*/1, /*motion_type*/7,
```

```
/*se_type*/7, /*move_speed*/0, /*search_area_width*/0, /*atk_type*/0,
```

```
/*normal_atk_cast_time*/0.0,
```

泳装大眼,

```
/*unit_id*/111801, /*unit_name*/"ペコリーヌ (ニューイヤー)", /*kana*/"ペ
```

```
こりーぬ
```

```

", /*prefab_id*/111801, /*is_limited*/0, /*rarity*/0, /*motion_type*/5,
/*se_type*/0, /*move_speed*/0, /*search_area_width*/0, /*atk_type*/0,
/*normal_atk_cast_time*/0.0,

```

新年佩可（赶紧实装吧），

```

/*unit_name*/"ラビリスタ", /*kana*/"らびりすた
", /*prefab_id*/106801, /*is_limited*/1, /*rarity*/1, /*motion_type*/0,
/*se_type*/0, /*move_speed*/0, /*search_area_width*/0, /*atk_type*/0,
/*normal_atk_cast_time*/0.0,

```

七冠之一的晶。

-----角色技能相关-----

skill\_cost.sql 为避免好奇在这解释一下，仅表示“升级 1 技能等级需要花费多少 mana”。

skill\_data.sql 角色技能相关参数，每个角色对应四个技能，但“技能+（专属武器或五星带来的新技能）”与“技能”间 id 独立，互不影响。

```

/*skill_id*/1009012,

```

角色技能 id。

其中 1 为标志位：“1”表示玩家角色技能，“2”表示敌方炮灰角色技能，“3”表示 boss 角色技能（活动、会战等），“4”表示 **召唤物技能**（感谢 L1nv3GA），“9”表示…测试样本以及咲恋、哈哈剑 ub 与由加利充电技能（测试）。之后的 009 表示技能组计数（十进制），比如 009 表示咲恋（假设），010 表示哈哈剑（假设）。最后的 012 表示技能组中技能，其中 001 表示一号技能（ub），002 表示二号，003 表示三号，012 表示二号的技能升级版（专武解锁）。请注意，没有 004 号技能，RANK7 解锁的那个被动技能序号为 501，升级版为 511（五星解锁）。另外，**白雪人**也从分析方法名的角度出发，给出了相同的结论：

<pre> public enum UnitType {     // Token: 0x040082E9 RID: 33513     [FieldOffset(Offset = "0x0")]     NONE,     // Token: 0x040082EA RID: 33514     [FieldOffset(Offset = "0x0")]     PERSON,     // Token: 0x040082EB RID: 33515     [FieldOffset(Offset = "0x0")]     MONSTER,     // Token: 0x040082EC RID: 33516     [FieldOffset(Offset = "0x0")]     BOSS, </pre>	<pre> // Token: 0x040082ED RID: 33517 [FieldOffset(Offset = "0x0")] SUMMON_PERSON, // Token: 0x040082EE RID: 33518 [FieldOffset(Offset = "0x0")] SUMMON_MONSTER, // Token: 0x040082EF RID: 33519 [FieldOffset(Offset = "0x0")] GUEST = 9, // Token: 0x040082F0 RID: 33520 [FieldOffset(Offset = "0x0")] SKILL_EFFECT </pre>
--	---

```

/*name*/"羅刹涅槃・無式+"

```

角色名，不知道为啥这个代码黑框这么大，可能是因为中二吧。

```

/*skill_type*/1,

```

技能属性个数，与 skill\_action.sql 挂钩，对应 action\_id 下该技能有几行数据，仅有 0、1 两种可能。0 表示该技能仅单属性，比如狗拳，四个技能分别是伤害、伤害、加伤害、加伤害，因此四技能全 0；1 表示该技能有多属性，比如中二，二技能包括伤害与魔防降低，因此该技能 type 为 1。请注意，对于含有强化技的技能，即使本体技能为 0，也会随着强化技为 1 而强制设置为 1，然后软件作者会在普通技能的对应 action\_2 设置空值“”，俗称摸鱼。

```
/*skill_area_width*/0,
```

技能释放范围，0 表示单体，其他则表示范围。

```
/*skill_cast_time*/1.58,
```

进场后需要多长时间才能释放该技能，这儿为中二的二技能。

```
/*action_1*/100901201, /*action_2*/100901202, /*action_3*/0, /*action_4*/0, /*action_5*/0, /*action_6*/0, /*action_7*/0,
```

对应角色释放该技能的具体属性 id，其中数字对应 skill\_action.sql 的编号（也可以在 prefabs.zip（干炸里脊站下载）或 trans.json（附件）的对应角色里找到）。上面截图为中二升级版二技能的 action。

```
/*depend_action_1*/0, /*depend_action_2*/103000201, /*depend_action_3*/103000201, /*depend_action_4*/0, /*depend_action_5*/0, /*depend_action_6*/0, /*depend_action_7*/0
```

对应不同情况下会触发的内容，这儿是扇子妮依的“旋风击”，因为存在“击破敌人回复 TP”的情况，所以该内容就会有对应 action id。如果没有选择枝，则全为 0。

```
/*description*/"目の前の敵1キャラに魔法中ダメージを与え、さらに魔法防御力を小ダウンさせる。",
```

技能描述，可以看到中二升级版二技能是有两个属性的。

```
/*icon_type*/2002
```

图标标记，每位都有其含义：物理/魔法、恢复/妨碍等（其实该参数已经直接对应技能图标了，不研究每一位的含义也行，除非你想自建角色）。但因为跟时间轴无关，请自行研究。

**skill\_action.sql** 角色技能详情，与 skill\_data.sql 密切相关，下面将接续中二的冒险之旅。

```
/*action_id*/100901201,
```

对应上文 skill\_id，形式为“skill\_id + num”，其中 num 按照该技能所拥有的属性依次产生（“伤害”+“敌魔防降低”则分为 100901201 与 100901202），十进制表示。

```
/*class_id*/1,
```

表示是玩家角色，对应 skill\_id 的第一个标志位，详情请参看 skill\_id。

```
/*action_type*/1,
```

表示技能属性，其中 90 表示魔法攻击力上升，1 表示（魔法）伤害，10 表示敌魔防降低，16 表示 tp 瞬间回复，48 表示 tp 缓慢回复等等，其余请自行查找。

```
/*action_detail_1*/2, /*action_detail_2*/0, /*action_detail_3*/0,
```

action\_type 对应的 detail，伤害有多高，降防有多狠等，请自行研究。

```
/*action_value_1*/20.0, /*action_value_2*/20.0, /*action_value_3*/1.2, /*action_value_4*/0.0, /*action_value_5*/0.0, /*action_value_6*/0.0, /*action_value_7*/0.0,
```

对应 action\_detail，与时间轴无关。

```
/*target_assignment*/1, /*target_area*/1, /*target_range*/-1, /*target_type*/3, /*target_number*/0, /*target_count*/1,
```

目标相关参数，与时间轴无关。感兴趣请自行选择角色并比对技能效果与此处数字，以推测其含义。

```
/*description*/"敵単体に{0}の魔法ダメージ",
```

技能的其中一部分描述，非完整技能。

```
/*level_up_disp*/"魔法ダメージ+{0}"
```

升级技能后对应该部分升级效果，非完整技能。



## -----角色攻击循环-----

**unit\_attack\_pattern.sql** 角色技能循环表，这里以莫妮卡为例（为什么上面中二这儿莫妮卡，是不是有猫腻？不是的，只是因为楼主后来全都用莫妮卡测试的……）

通过 **unit\_data.sql** 或 **actual\_unit\_background.sql** 我们知道莫妮卡的 unit\_id 为 105331:

```
/*unit_id*/105301,
```

**actual\_unit\_background.sql** 的 3 哪去了？普遍被删掉了……

```
/*pattern_id*/10530101,
```

该动作循环 id，程序调用时使用。

```
/*loop_start*/4, /*loop_end*/6, /*atk_pattern_1*/1001, /*atk_pattern_2*/  
/1, /*atk_pattern_3*/1002, /*atk_pattern_4*/1, /*atk_pattern_5*/1, /*at  
k_pattern_6*/1002,
```

后面全是 0，就不粘贴了。这个的含义很明显：loop\_start 与 loop\_end 表示“从哪里开始循环”。当角色进场时，会优先使用“1001”，即 1 技能（不是 ub，是那个 buff 技），随后的“1”表示“普通攻击”，然后“1002”表示“锁定射击”。接下来已经到了 loop\_start 的标记位 atk\_pattern\_4，这时候就会循环“1-1-1002”，即“普攻-普攻-锁定射击”。

**unit\_skill\_data.sql** 角色技能列表。

---

在你完全了解上面内容后，还差一步就可以了解数据的全貌了，下面请打开 **prefabs.zip（干炸里脊站下载）** 或 **trans.json（附件）**。既然叫做 prefabs，就表明该文件是角色做出行为后的一系列效果动画，包括“攻击/技能/Buff/Debuff”等效果以及“屏幕晃动/特写”等动画特效。有些小伙伴可能已经猜到了，**这游戏角色动作和伤害是完全分开的，根本没有什么即时演算**。那有些小伙伴会说，不对，我看那远程的魔弹打到别人身上才跳数字的啊。这些都是假象，是由一个叫做 ExecTimeForPrefab.data.time 参数所造成的假象。魔弹飞出去是播片，伤害则是在 ExecTimeForPrefab.data.time 固定显示的，给你一种“打上去”的错觉（其实老游戏都是这么做的）。

下面依旧以莫妮卡为例，但因为篇幅原因，请直接查看“莫妮卡.json”原文，这儿将会以行数来表示（请尽量安装 VS Code 打开该文件）。

先看普攻吧，这儿第 12 行，普攻有“AttackDetail”，这是只有普攻才有的参数，“visible”表示该属性不可见，其对应的 AttackDetail.ExecTimeForPrefab.data.time 这可能是伤害计算的时间。但我更建议你去查看第 29 行的“Attack”，因为它与之后的 skill 形式是对应的，更为关键可靠。这时我们来到第 30 行，IsPrincessForm，是否为公主形态，以及下一行公主形态要播什么片，反正不重要就是了。请查看接下来的 ActionParametersOnPrefab，这是普通攻击的**全部信息**：

35 行 Visible 为 1 表示该属性可见（实际上就是游戏中跳伤害数字）；

36 行 ActionType 为 0 表示“伤害”（可以从附件（感谢白雪人提供）中查找，或者查看上文的 **skill\_action.sql** ActionType 来自行分析（后面有 action 描述））；

37 行表示该 action 会产生的各种效果，没错，**skill 分割成 action，action 有分割成一个个可见或者不可见的特效**，这就是本游戏战斗的运作机理之一。

40 行表示该伤害数字“可见”；

44 行**表示播片后**（就是喊口号以后。如果普攻，那就从默认动作开始，默认动作就是角色装备界面的那个动作，或者你看几帧动作不变就是了）**经过几秒显示（Visible=1）该伤害**；

45、46 行表示显示的那个字的特效；

47 行表示显示的字符大小，DuangDuang 的；

57 行表示攻击动画；

63 行表示击退动画；

**73、74、75、76、92 都是该攻击带来的效果与 buff 的特效，但莫妮卡普攻没啥特效，有兴趣可以看其技能的对应位置，各种 buff 的特效之类的都在；**

100 行表示强制禁止攻击，后面的内容看名字应该都能明白了；

但请注意，113-115 行**并不是动画播放时间，并不是动画播放时间，并不是动画播放时间。**

-----  
带着上面的知识我们来复习下 UB 吧，还是算了，UB 的特效太多了（万圣节炸弹人有 1000 行，全是可见与不可见的特效。包括那个延时炸弹也是特效，并不是独立个体（日社员工の慵懒）），咱们看小技能（仅挑重点）“羽翼的加护”（好像是这么叫的）：

469 行，这儿是“羽翼的加护”第一个 action 的类型，10 表示 buff/debuff；

477 行，这儿是“羽翼的加护”第一个 action 产生效果的时间，在这之前用 ub 打断，buff 就么得了。如果入乡随俗一点，这个可以叫做“前摇”与“后摇”的分割线……

485 行，这儿是“羽翼的加护”第一个 action 的 id，也是我为啥判断是第一个 Action 的原因；

508 行。**为什么要特地指出这个呢？因为这个虽然是空的，但其他一些 action 的子特效是存在内容的，但标识 visible 为 0，即不可见。比如可可萝的冲刺无敌，又比如其他一堆你觉得技能没描述但却有这个效果有种“卧槽真实物理碰撞引擎”的错觉，问题就在这儿了；**

至于该技能剩下的内容，自己看啦，跟上面讲的差不多，稍微变通下就好了。

-----  
你要问 buff 的持续时间在哪里？稍微有点怀疑 skill\_action 的/\*action\_detail\_1\*/，但涉及函数调用，没有证据。不过跟行为模式没太大关系，可以往后放放；

你要问动画时间在哪里？这个真没有……要么查看 spine 数据，要么跟上面的 buff 时间一起，录视频数帧数吧……

-----  
**以上所有内容均仅依据游戏本身以及所获得的数据文件推测而来，并未涉及或阅读任何代码部分，请酌情参考。最后祝大家游戏愉快~**

-----  
**具体角色逻辑请移步另一篇文章，PCR-Timeline-Logic-Analysis ver1.1。**