
# ATLANTISS CONTENT FIXERS GUIDE

by Atlantiss Team

# Contents

# Database

*~ by bandysc*

## What is MySQL? (Skip it if you're already familiar with SQL)

MySQL is a relational database management system. In our case it is the storage for all data about objects (npc, gameobjects, items), loot (what to drop from which object), basic scripts (called SmartScripts), conditions (e.g. spell should be casted only on specific target), creature texts (what npc says), gossip menu (when you talk to the npcs) and many more.

The database of WoW cores is really expanded, however do not be scared. You do not have to know everything by heart, there is no need. A lot of information about database is in the TrinityCore wiki:

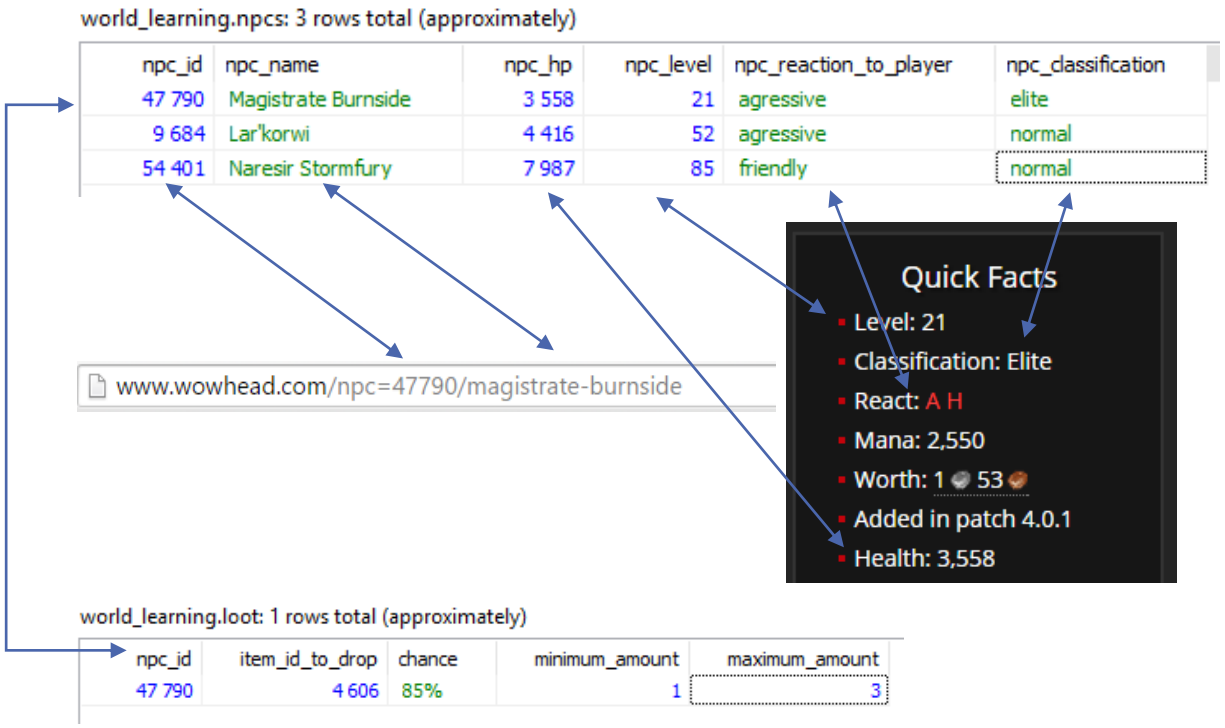
https://trinitycore.atlassian.net/wiki/display/tc/world

Save the link in bookmarks, you'll visit it often ☺

## Quick SQL guide

SQL is a programming language designed for managing data held in a relational database management system, such as MySQL. That means there are also other database systems that uses same SQL language. They are, more or less, compatible with each other. So: SQL is a language that we use to communicate with the database systems. MySQL is one of them. SQL language is really powerful, however for our needs, we will only use small fractions of it.

How is data stored? Let's take a look:



> **Note**: This is not actual TrinityCore structure, it is just a simplified example to get to know how database works.

MySQL stores DATABASES, in the picture you can see a database called "WORLD_LEARNING". Database consists of TABLES – in the picture you can see two tables: NPCS and LOOT.

> Dot: dot is used to divide database and table: [DATABASE_NAME] . [TABLE], e.g.:
>
> Database name → world.creatures ← table name

Since it's a table, it consists of COLUMNS and ROWS. The columns provide the structure according to which the rows are composed. In example above, table NPCS consist of columns: `npc_id`, `npc_name`, `npc_hp`, `npc_level`, `npc_reaction_to_player`, `npc_classification`. Names are already self-explainable so reading the tables is really simple. ROW represents a single, implicitly structured data item in a table, e.g. npc entry.

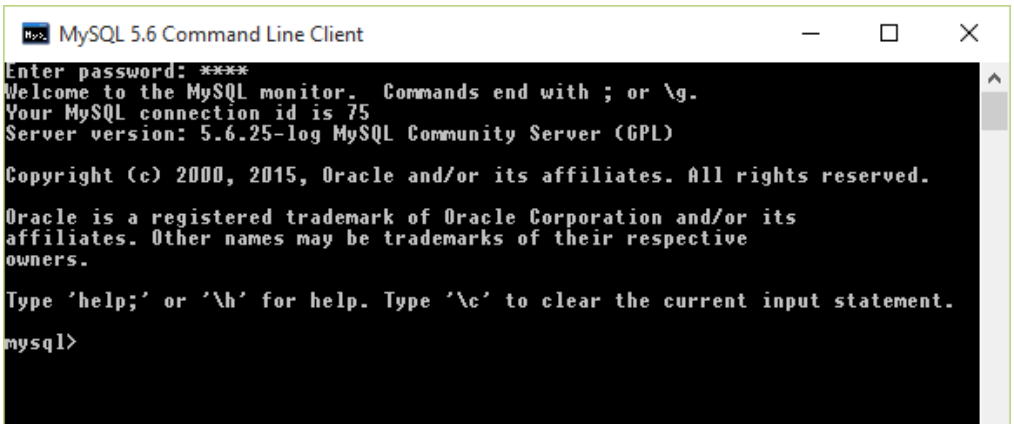
Since you already know the most important terms, namely: database, table, column and row, we can now read more info. When you kill npc 47790 you have 85% chances to loot 1 to 3 items 4606. Which one is npc 47790? We have to find the row with this id in the `npcs table`. It is of course NPC Magistrate Burnside. There is a RELATION between `npcs` and `loot tables`. That's why we call it RELATIONAL DATABASE SYSTEM. We do not put NPC name into the `loot` table, NPC name is only in `npcs table`. It still can be easily found, but at least the data is not DUPLICATED.

Let's say some lazy person decided to put npc name in table `loot`, so that no one have to switch between npc and `loot tables` all the time? It might seem as a good idea (it saves few seconds), but when an npc name is changed, then we'd have to change the name of this NPC in all tables, but if we forget about at least one table, we'll have problems! Not to mention the fact that all data weight some bytes. One row is nothing, but with thousands of rows we would waste many megabytes of precious hard disk storage.

## How to edit tables?

Finally we come to SQL language. As I have already mentioned, SQL is a really extended language, however for our purposes we only need few of its features. Database client applications are really powerful, bur for better understanding it's better to learn without any GUI client.

Run MySQL `Command Line Client` (it is installed by default with MySQL server)




**Picture 1 Doesn't look pretty :)**

Once you run the console, type the password and hit enter.

> Now we will switch to actual TrinityCore database, make sure you have already loaded it.

Let's switch to our main database – `world`.

```
mysql> use world;
```

Hit enter now. Console should output:

```
Database changed
```

Congratulations! You've just learned your first SQL command

## Selecting data

Now, let's `select` some data from database. Instead of theory, just type:

```
mysql> SELECT entry, name FROM creature_template LIMIT 5;
```

The output should be following:

```
mysql> SELECT entry, name FROM creature_template LIMIT 5;
+-------+--------------------------------+
| entry | name                           |
+-------+--------------------------------+
|     1 | Waypoint (Only GM can see it)  |
|     3 | Flesh Eater                    |
|     6 | Kobold Vermin                  |
|    30 | Forest Spider                  |
|    36 | Harvest Golem                  |
+-------+--------------------------------+
5 rows in set (0.00 sec)
```

This looks pretty simple, we SELECTED COLUMNS `entry` and `name` from TABLE `creature_template`. LIMITED to 5 ROWS.

> While this part of the guide is general SQL syntax, "`LIMIT`" is custom MySQL syntax

How do we know columns names? You can check TC wiki. Or use a special command to list all columns:

```
mysql> SHOW COLUMNS FROM creature_template_addon;
```

```
mysql> SHOW COLUMNS FROM creature_template_addon;
+---------+----------------------+------+-----+---------+-------+
| Field   | Type                 | Null | Key | Default | Extra |
+---------+----------------------+------+-----+---------+-------+
| entry   | mediumint(8) unsigned | NO   | PRI | 0       |       |
| path_id | int(10) unsigned     | NO   |     | 0       |       |
| mount   | mediumint(8) unsigned | NO   |     | 0       |       |
| bytes1  | int(10) unsigned     | NO   |     | 0       |       |
| bytes2  | int(10) unsigned     | NO   |     | 0       |       |
| emote   | mediumint(8) unsigned | NO   |     | 0       |       |
| auras   | text                 | YES  |     | NULL    |       |
+---------+----------------------+------+-----+---------+-------+
7 rows in set (0.01 sec)
```

> `SHOW COLUMNS FROM` is one more custom mysql command. I promise it is the last custom command ☺

As you can see we have not only column names (in this example: entry, path_id, mount, bytes1, bytes2, emote, auras) but also the type of column (text, int – integer number), can be null (no/yes) – whether the field in row has to have value. If Null = No, it cannot be null and HAS to have value, if null = Yes, it can be null and does not have to have a value, e.g. while inserting a new row.

Instead of specifying column names, you can use * (it means SELECT *= everything FROM…). It will output all columns:

```
mysql> SELECT * FROM npc_spellclick_spells LIMIT 1;
```

```
mysql> SELECT * FROM npc_spellclick_spells LIMIT 1;
+-----------+----------+------------+-----------+
| npc_entry | spell_id | cast_flags | user_type |
+-----------+----------+------------+-----------+
|     24418 |    43768 |          3 |         0 |
+-----------+----------+------------+-----------+
1 row in set (0.00 sec)
```

## Limiting rows

I know what you're asking for: how on earth do I select a specific row. As an example we'll use NPC Magistrate Burnside. Here we come to the WHERE clause. Again, instead of theory, just type:

```
mysql> SELECT entry, name, minlevel, maxlevel FROM
creature_template WHERE entry = 47790;
```

```
mysql> SELECT entry, name, minlevel, maxlevel FROM creature_template WHERE entry = 47790;
+-------+--------------------+----------+----------+
| entry | name               | minlevel | maxlevel |
+-------+--------------------+----------+----------+
| 47790 | Magistrate Burnside |       21 |       21 |
+-------+--------------------+----------+----------+
1 row in set (0.02 sec)
```

```
WHERE [column_name] [operator_type] [value]
```

As you can see, in this example we SELECT rows WHERE entry is equal to 47790.

| OPERATOR | DESCRIPTION |
|---|---|
| = | Checks if the values of two operands are equal |
| != | Checks if the values of two operands are NOT equal |
| > | Checks if the value of left operand is greater than the value of right operand |
| < | Checks if the value of left operand is less than the value of right operand |
| >= | Checks if the value of left operand is greater than or equal to the value of right operand |
| <= | Checks if the value of left operand is less than or equal to the value of right operand |

http://www.tutorialspoint.com/sql/sql-operators.htm

We can use multiple conditions in one query with two operators:

| LOGICAL OPERATOR | DESCRIPTION |
|---|---|
| **AND** | The row will be shown only if ALL conditions joined with AND are true |
| **OR** | The row will be shown if ANY of the conditions joined with OR is true |

You can also use parenthesis, which work just like in math.

```
mysql> SELECT entry FROM creature_template WHERE entry = 47790
OR entry > 1000 AND entry < 1010;
```

Those are not all operators, let's take a look at the following. They are also very helpful:

| OPRATOR | DESCRIPTION |
|---|---|
| **BETWEEN a AND b** | Used to search for values that are within a set of values, given the minimum value and the maximum value. |
| **IN (a, b, c…)** | Used to compare a value to a list of literal values that have been specified. |
| **NOT** | Reverses the meaning of the logical operator with which it is used. This is a negate operator. |
| **IS NULL** | Used to compare a value with a NULL value. |

http://www.tutorialspoint.com/sql/sql-operators.htm

Examples:

```
mysql> SELECT entry FROM creature_template WHERE minlevel
BETWEEN 20 AND 30;

mysql> SELECT entry FROM creature_template WHERE entry IN
(1001, 1007, 1008);

mysql> SELECT entry FROM creature_template WHERE entry NOT IN
(1001, 1007, 1008) LIMIT 5;
```

## Inserting rows

Time to insert some rows! Type:

```
mysql> INSERT INTO npc_spellclick_spells
(npc_entry, spell_id, cast_flags) VALUES
(47790, 46598, 0);
```

This insert has no sense for quests, it is only for learning purposes. We will get rid of it in a minute

We've just inserted these following rows into the npc_spellclick_spells table:

| Column | Value |
|---|---|
| **npc_entry** | 47790 |
| **spell_id** | 46598 |
| **cast_flags** | 0 |

Try to execute the query again:

```
mysql> INSERT INTO npc_spellclick_spells (npc_entry, spell_id, cast_flags) VALUES (47790, 46598, 0);
ERROR 1062 (23000): Duplicate entry '47790-46598' for key 'PRIMARY'
```

Luckily MySQL won't let you do this. You can have only one row for key-pair NPC_ENTRY <-> SPELL_ID in `npc_spellclick_spells`. The same error will show if you try to insert NPC which is already in table `creature_templates`.

If you remember from previous chapter, `table npc_spellclick_spells` has 4 columns, but we have only specified values for three of them. It is possible, because column `user_type` has default value 0:

```
mysql> SHOW COLUMNS FROM npc_spellclick_spells;
+------------+----------------------+------+-----+---------+-------+
| Field      | Type                 | Null | Key | Default | Extra |
+------------+----------------------+------+-----+---------+-------+
| npc_entry  | int(10) unsigned     | NO   | PRI | NULL    |       |
| spell_id   | int(10) unsigned     | NO   | PRI | NULL    |       |
| cast_flags | tinyint(3) unsigned  | NO   |     | NULL    |       |
| user_type  | smallint(5) unsigned | NO   |     | 0       |       |
+------------+----------------------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

If you try to omit cast_flags, error would appear:

```
mysql> INSERT INTO npc_spellclick_spells (npc_entry, spell_id) VALUES (47790, 46598);
ERROR 1364 (HY000): Field 'cast_flags' doesn't have a default value
```

You can also omit column if it can be NULL.

You can also insert multiple rows at once:

```
mysql> INSERT INTO npc_spellclick_spells
(npc_entry, spell_id, cast_flags) VALUES
(47790, 46598, 0),   <-- comma
(47791, 46598, 0),   <-- comma
(47792, 46598, 0);   <-- semicolon
```

## Changing rows

Since you already know how to limit returned data (LIMIT, WHERE), there is nothing simpler than modifying already inserted data. Of course, we could select a row first, then delete it and insert again with other values - and we'll do that often. However, sometimes it's pointless to remove whole row only to change one field. Type:

```
mysql> UPDATE npc_spellclick_spells SET
cast_flags = 1, user_type = 2
WHERE npc_entry = 47790 AND spell_id = 46598;
```

We use the same WHERE syntax as in SELECT queries, then there is no need to explain that. It should be self-explainable. We change `cast_flags` value to `1` and `user_type` values to `2` in `npc_spellclick_spells` table where `npc_entry` equals `47790` and `spell_id` equals `46598`.

## Removing rows

One more thing before we actually begin. Let's remove previously inserted rows:

```
mysql> DELETE FROM npc_spellclick_spells WHERE npc_entry IN
(47790, 47791, 47792) AND spell_id = 46598;
```

We use the same WHERE syntax as in UPDATE, SELECT queries, then there is no need to explain that.

You might ask: do we have to specify `spell_id` value? In this example – no. The result would be the same as `WHERE npc_entry IN (47790, 47791, 47792);` However, if there were already some rows for creature 47790 (with other spells) and we had removed everything from table `npc_spellclick_spells` narrowing only to `npc_entry`, we would break some quests.

> Always make sure that you're updating/deleting only the rows that you want to update/delete. Nothing more, nothing less!

## Summary

We've learned few basic SQL statements, let's revise them. Value in square parentheses means that this part is optional.

```
SELECT column1, column2… FROM table_name [WHERE condition] [LIMIT amount];

INSERT INTO table_name (column1, column2…) VALUES (row1_value1, row1_value2),
(row2_value1, row2_value2);

UPDATE table_name SET column1 = value1, column2 = value2… [WHERE condition]
[LIMIT amount];

DELETE FROM table_name [WHERE conditions] [LIMIT amount];
```

> Remember: every SQL command ends with semicolon (;)

```
LIMIT and SHOW COLUMNS is MySQL specific syntax

ALWAYS CHECK FEW TIMES WHETHER YOU ARE DELETING PROPER ROWS!
```

## A bit of... bits and binary math.

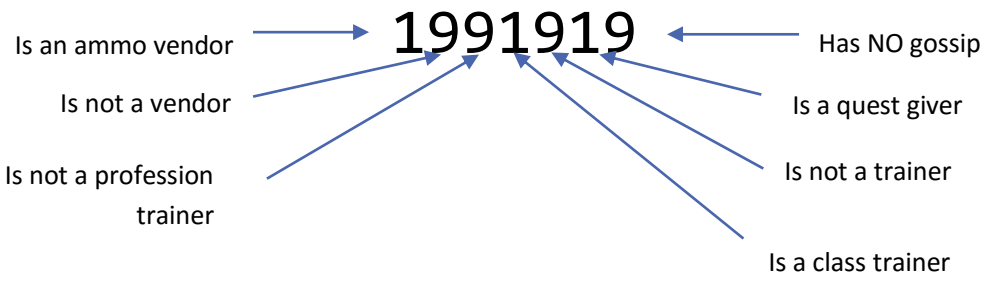
(you can skip it if you know what bitmasks are)

In WoW we have to store a lot of boolean values (true/false, on/off). NPC can be an innkeeper, a merchant, a class trainer. Then, we could store NPC type in column npc_type and define:

- If npc_type is equals to 0, npc is normal mob
- If npc_type is equals to 1, npc is an innkeeper
- If npc_type is equals to 2, npc is a merchant
- If npc_type is equals to 3, npc is a class trainer

It might pull it off, but there is a problem. NPC can be **AT ONCE** an innkeeper and a class trainer. Then the idea is useless, does it mean we need tons of columns such as: `is_innkeeper`, `is_class_trainer`, `is_merchant`? Well, It'd do the trick, but in result we'd have too many columns, and there are already a lot of them.

Let's think how we can store boolean values in a number. What if the rightmost digit showed if npc is an innkeeper, next digit would show if npc is a merchant and so on? Let's say, a digit 9 means npc IS NOT of specified type, digit 1 means npc IS of specified type:




How many true/false values can we save in such system? We have to check what is the highest number that computer is capable of storing. Well, what highest number can computer store easily and efficiently. We can say that the highest number is 4 294 967 295. It consists of 10 digits. It might seem enough, but usually it's not.

This way of saving data saves some columns, but is not really efficient. We are wasting a lot of digits: 0, 2, 3, 4, 5, 6, 7, 8. For saving boolean values we need only two digits and in decimal numeral system we have ten digits. Numeral system with two digits will be fine for use. And what is the most common numeral system with two digits? Of course **BINARY NUMERAL SYSTEM**.

> At this point I assume you know already what binary numeral system is, how to read binary numbers and how to convert them to decimal and vice versa. If you don't have any knowledge of it, read about it first. It is really necessary for WoW scripting:
>
> http://www.math.grin.edu/~rebelsky/Courses/152/97F/Readings/student-binary
>
> https://en.wikipedia.org/wiki/Binary_number

How much is 4 294 967 295 in binary?

$4\ 294\ 967\ 295_{(10)} = 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111_{(2)}$

32 digits (it takes 32 bits = 4 bytes, that's why it is called unit32 – it is unsigned (doesn't contain data about sign of the number) integer number in 32 bits, over three times more than in decimal system and still perfect for our purposes, since we only need two digits.

Then if npc has type $1001010_{(2)}$, it means it is a quest giver, a class trainer and ammo vendor. You might say: the only difference is that we use 0 instead of 9. That's right, but also it is not a number in binary system.

$1001010_{(2)} = 74_{(10)}$

And it is equal to 74 in decimal system. Now you can see we have still a lot of place to store more types.

## How to check if npc is of type?

It must be the question that you're asking yourself right now. If we'd store types in separate columns, finding quest givers would be really simple:

```
SELECT * FROM creature_template WHERE is_questgiver = true;
```

We cannot compare npc_type to 2 (2, because $10_{(2)} = 2_{(10)}$), because npc with type 74 wouldn't be returned, while he should be, etc.

## Bit-operators

Here we come to bit operators. **BIT OPERATOR AND (&)** compares two same bits and returns 1 if both are 1, else it returns 0. Example:

| a | b | a & b |
|---|---|---|
| **1** | 1 | 1 |
| **1** | 0 | 0 |
| **0** | 1 | 0 |
| **0** | 0 | 0 |

```
   1001010(2)
&  0000010(2)
= --------
   0000010(2) = 2(10)
```

What if npc hadn't any quest giver flag and we bit-anded it with $10_{(2)}$?

```
   1111000(2)
&  0000010(2)
= --------
   0000000(2) = 0(10)
```

We either get 2 or 0. Two means that the npc is a quest giver, zero means that he's not. That way, we can easily find quest givers:

```
SELECT * FROM creature_template WHERE npcflag & 2 > 0;
```

## Adding type flag to npc

Next thing: how to update npc type? Yes, we might simply execute:

| a | b | a \| b |
|---|---|---|
| **1** | 1 | 1 |
| **1** | 0 | 1 |
| **0** | 1 | 1 |
| **0** | 0 | 0 |

```
UPDATE creature_template SET npcflag = 2
WHERE entry = 36694;
```

However, npc 36694 already had some npc type flags. We would have to check npc's old type every time we add a new type. There is a better way: **BIT-OR OPERATOR (|).** If both bits are 0, bit-or returns 0, else it returns 1, example:

```
   1111000(2)
|  0000010(2)
= --------
   1111010(2)
```

What if an npc already is a quest giver?

```
   1111010(2)
|  0000010(2)
= --------
   1111010(2)
```

He will be a quest giver – nothing will change.

```
UPDATE creature_template SET npcflag = npcflag | 2 WHERE entry
= 36694;
```

## Removing type flag from NPC

| a | b | Result |
|---|---|---|
| **1** | 1 | 0 |
| **1** | 0 | 1 |
| **0** | 1 | 0 |
| **0** | 0 | 0 |

Another thing that we need is to removing a flag. We would need an operator that for bit 1 and 0 gives 1, 0 and for 0 gives 0, 1 and for 1 gives 0. Also for 0 and 1, it has to return 0, or else the flag wouldn't be removed, but toggled.

However, such operator does not exists… because it is not needed. We can use two different operators to remove flag.

| a | ~a |
|---|---|
| **1** | 0 |
| **0** | 1 |

**INVERSE OPERATOR (NEGATE; ~, !)** is a bit different operator, because it takes only one argument. It works pretty simple: for 0 it returns 1 and for 1 it returns 0.

If we bit-and inversed value, we can remove a flag (let's remove quest giver flag). For starters, inverse flag "quest giver"

```
  ~0000010(2)
= --------
   1111101(2)
```

```
   1001010(2)
&  1111101(2)
= --------
   1001000(2)
```

In result, the one bit has been set to 0. Bingo! Even if you execute the query twice, the flag will always be removed.

```
UPDATE creature_template SET npcflag = npcflag & ~2 WHERE entry
= 36694;
```

## Summary

In order to set a flag, use bit-or operator:

```
SET column_name = column_name | [value]
```

To remove a flag, bit-and inversed value:

```
SET column_name = column_name & ~ [value]
```

To check if npc has flag, use bit-and operator:

```
WHERE column_name & [value] > 0
```

# Most important TrinityCore tables

TrinityCore consists of many tables, but you do not have to check/get to know every single one of them. To be honest, you will not even visit many of them. The best source to get to know the tables is TC wiki, here I will show you the most important tables in the world database.

## creature_template

I would say `creature_template` along with `creature` are the most important tables in world database. As long as the client contains the data about map, terrain and all spawns they are sent directly while playing – NPC data are NOT stored in the client, they are sent from the server as you move. `Creature_template` stores definition of all NPCs. Each NPC is described by unique number – `entry`. Most fields represent data directly sent to client, then entries can be automatically added from SNIFFS. While, most likely NPC has already proper name and model, sometimes you have to correct its level (`minlevel`, `maxlevel`), add flags (`unit_flags, unit_flags2, npcflags`), add gossip menu (`gossip_menu_id`), change speed (`speed_walk, speed_run`) or modify it as a vehicle (`VehicleId`, will take about it later). All columns mentioned here are sent directly to the client, one column is a bit different - `flags_extra` are additional custom flags added by core community. That means they will never be found in sniffs.

## creature

While table `creature_template` contains definition about a group of npcs, `creature` table contains data about specific spawn. Each spawn has global unique identifier – `GUID`. Each creature has its own guid. Actually, in whole game world each object (npc/item/gameobject, etc.) has its own guid. Does it mean that when once a specific GUID is used in `creature` table, it cannot be use in table `gameobject`? No, actually, in database we store only part of GUID. Whole GUID consists of object type and "lower part" – object number. Though this is core task to manage them. For us – guid in table unambiguously identifies object (creature/gameobject/item).

Of course table creature contains position of the npc (`map, position_x, position_y, position_z, orientation`),

Position shown in game IS NOT actual object position!

spawn time (`spawntimesecs`) and spawn distance (`spawndist`). It can also override some data from creature_template (e.g. `npcflag, unit_flags, dynamicflags`).

## gameobject_template

`Gameobject_template` and `gameobject` work pretty much in the same way as creature counterpart. You might be surprised that `gameobject_template` contains a lot of unnamed columns (data1, data2, data3…). It is because, for every different gameobject types, the data have different interpretation. Check TC wiki to find out what do those fields mean for specific gameobject type.

## creature_text

Another table you will visit all the time while scripting quests, instances and raids is creature_text. Pretty self-explainable. All NPCs dialogues are in this table. One dialogue line is identified by its `group id`. Though, for one `group_id` you can add an unlimited amount of lines (with different `ids`). The core will choose randomly one line from all within same `group_id`.

## smart_scripts

You will visit smart_scripts more often than creature_text. But not directly. Smart Scripts are simple, but quite powerful (compared to ease of creating) tool to script things. The same thing can be done with C++ scripts in core, but sometimes there is no need for that. If npc should cast spell during a fight – Smart AI is perfect. If after a quest completion npc should portray a scene with dialogues – SAI is enough. Each SAI row consists of, what to do (action), when to do (event) and who should do it (target). Writing scripts by hand is not efficient, that's why there are few tools to make scripting easier. In the guide we will use Visual SAI Studio.

# Scripting

Finally we can have some fun with scripting and fixing. There is no quick way to learn this. You have to script, try, script, try and try one more time. Trying and checking is best way to learn TC. In this guide we will show you some examples of broken stuff and a way to fix it. Of course there are many ways to fix most of the following stuff, but here we'll show and describe our idea, but still - it doesn't mean that it is the only proper one.

## Quest drop

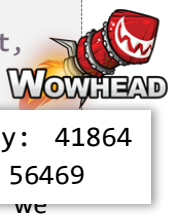
*By Raknar*

**DESCRIPTION:** The quest does not work, because the item doesn't drop.

**FIX:** Add quest item to loot table

```
DELETE FROM creature_loot_template WHERE entry = 41864 and item = 56469;

INSERT INTO creature_loot_template (entry, item, Chance, QuestRequired, MinCount,
MaxCount) VALUES
(41864, 56469, 96, 1, 1, 1);
```




You must be asking: "why are we deleting a row from creature_loot_template if we are adding one?"

ALWAYS BEFORE INSERTING ROW, DELETE IT FIRST!

SQLs might be executed many times and they shouldn't break anything. That's why we are deleting first (even it doesn't exist) and then insert new one.

How do we know what particular columns mean? From the wiki. In this case, the column names are self-explaining though and I think there is no need to explain them.

# The Abandoned Crypt, Ambushed!

*By Bandysc*

**DESCRIPTION:** The quest The Abandoned Crypt does work, but the npc (guard) is missing. The quest Ambushed! Cannot be completed, because there are no npcs to kill. The difficulty in scripting the quest is phasing – after turning the quest The Abandoned Crypt we are phased and see the npcs.

**FIX:** Add missing npcs and make proper phasing.

We could add the npcs by hand but with sniffs, the parser can generate spawns automatically! Also – the question is: what phase should be used? The answer is also in the sniff!

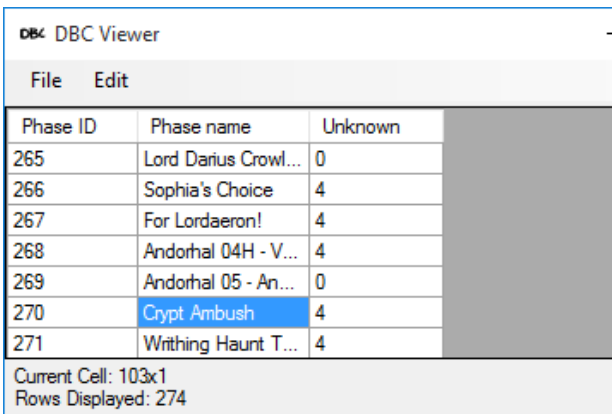
First missing npc is Foulmane (1847). TrinityCore Packet Parser has successfully parsed the spawn of this mob:

```
DELETE FROM creature WHERE id = 1847;
SET @CGUID := (SELECT MAX(guid)+1 FROM creature);
INSERT INTO creature (guid, id, map, spawnMask, phaseMask, phaseId, position_x,
position_y, position_z, orientation, spawntimesecs, spawndist, MovementType)
VALUES
(@CGUID, 1847, 0, 1, 1, 0, 1060.536, -1899.833, 30.90833, 5.061455, 120, 0, 0);
```

What's next? After we turn the quest The Abandoned Crypt (27170) we are phased. What phase is it? The hint is in the sniffs:

```
ServerToClient: SMSG_PHASE_SHIFT_CHANGE (0x137C) Length: 42 ConnIdx: 0 Time:
12/27/2015 01:26:07.719 Number: 43898
Client: Full: 0x08338000000000000000000000B746CF Player/0 R3296/S0 Map: 0 Low:
12011215
PhaseShiftFlags: 16
PhaseShiftCount: 1
PersonalGUID: Full: 0x0
[0] PhaseFlags: 0
[0] Id: 270
PreloadMapIDsCount: 4
[0] PreloadMapID: 1066 (1066)
[1] PreloadMapID: 1190 (1190)
UiWorldMapAreaIDSwap: 0
VisibleMapIDsCount: 4
[0] VisibleMapID: 992 (992)
[1] VisibleMapID: 683 (683)
```

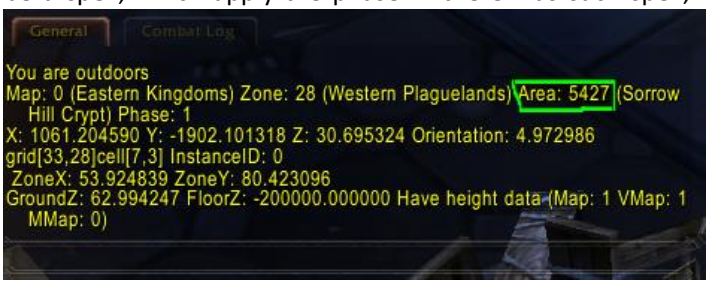
It is the phase 270. To confirm this, we could check how the phase was named in DBC.




**RECOMMENDED TOOL:**
**NAME:** DBC Viewer
**AUTHOR:** tomrus88
**DOWNLOAD:** github.com

Client data is stored in dbc database. A lot of interesting data can be found there.

"Phase 270 – Crypt Ambush". No doubt this is the correct one☺. How do we apply the phase? There might be a spell, which apply the phase. If there was such spell, we would add it to the `spell_area` table.




However if we analyze each spell in sniff – none spell is for the phase. Therefore we use `phase_area` table, where we simply force phase to specific area.

How do we know what area we are in? There is handy command `.gps`.

```
DELETE FROM phase_area where phaseId = 270 and areaid = 5427;
INSERT INTO phase_area (AreaId, PhaseId, Comment) VALUES
(5427, 270, 'Crypt Ambush after quest The Abanadoned Crypt');
```

I guess this need no explanation. However, you must be asking yourself: it can't work now. The game doesn't know the phase should be applied only when player has finished quest The Abandoned Crate. That's Right! We have to add a condition (`CONDITION_SOURCE_TYPE_PHASE = 26`) so that the phase 270 will be applied only with quest The Abandoned Crate rewarded (`CONDITION_QUESTREWARDED = 8`).

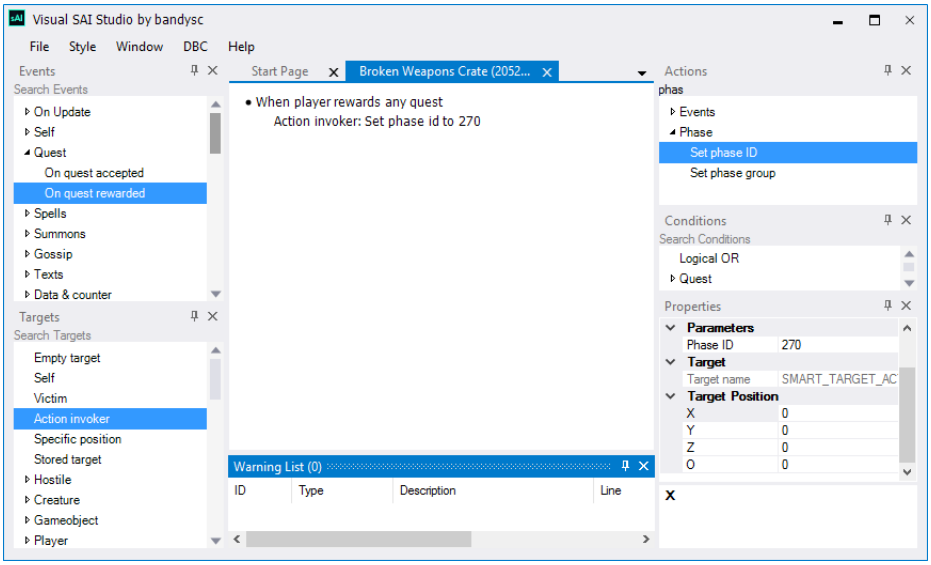
```
DELETE FROM conditions WHERE SourceTypeOrReferenceId=26 AND SourceEntry=270;
INSERT INTO conditions (SourceTypeOrReferenceId, SourceEntry,
ConditionTypeOrReference, ConditionValue1, Comment) VALUES
(26, 270,8,27170,"Apply phase 270 only if quest The Abandoned Crate is rewarded");
```

Now we just need to add missing npcs - `Skeletal Sorcerer (1784)` and `Skeletal Flayer (1783).` Once again TrinityCore Packet Parses has done good job. We just need to set phase 270. We should also set random movement (`MovementType = 1`) with small `spawndist`.

```
DELETE FROM creature WHERE id IN (1784, 1783);
SET @CGUID := (SELECT MAX(guid)+1 FROM creature);
INSERT INTO creature (guid, id, map, phaseId, position_x, position_y, position_z,
orientation, spawntimesecs, spawndist, MovementType) VALUES
(@CGUID+0, 1783, 0, 270, 1115.802, -1891.251, 61.19556, 5.524169, 120, 5, 1),
(@CGUID+1, 1784, 0, 270, 1088.996, -1864.045, 62.80103, 0.06643827, 120, 5, 1),
(@CGUID+2, 1784, 0, 270, 1045.695, -1908.965, 30.82499, 2.453376, 120, 5, 1),
(@CGUID+3, 1783, 0, 270, 1075.717, -1909.538, 48.06567, 1.980794, 120, 5, 1),
(@CGUID+4, 1784, 0, 270, 1044.053, -1910.494, 48.07217, 1.993965, 120, 5, 1),
(@CGUID+5, 1784, 0, 270, 1045.991, -1943.7, 39.16716, 3.179086, 120, 5, 1),
(@CGUID+6, 1784, 0, 270, 1058.261, -1930.749, 39.45127, 2.985977, 120, 5, 1),
(@CGUID+7, 1784, 0, 270, 1045.575, -1897.862, 49.9626, 1.364547, 120, 5, 1),
(@CGUID+8, 1783, 0, 270, 1076.561, -1933.695, 38.68034, 5.657713, 120, 5, 1),
(@CGUID+9, 1783, 0, 270, 1127.605, -1872.129, 62.54224, 5.609202, 120, 5, 1),
(@CGUID+10, 1784, 0, 270, 1073.505, -1903.743, 48.06619, 4.372558, 120, 5, 1),
(@CGUID+11, 1784, 0, 270, 1078.652, -1883.786, 63.50883, 0.3303253, 120, 5, 1),
(@CGUID+12, 1784, 0, 270, 1077.337, -1918.378, 63.13568, 1.060374, 120, 5, 1),
(@CGUID+13, 1784, 0, 270, 1097.204, -1896.626, 60.87663, 2.277877, 120, 5, 1),
(@CGUID+14, 1784, 0, 270, 1079.752, -1943.462, 39.16718, 3.182107, 120, 5, 1),
(@CGUID+15, 1783, 0, 270, 1074.003, -1904.089, 30.82499, 2.174419, 120, 5, 1),
(@CGUID+16, 1784, 0, 270, 1060.87, -1912.286, 51.18807, 5.066514, 120, 5, 1),
```

```
(@CGUID+17, 1783, 0, 270, 1042.91, -1930.071, 39.16716, 6.254864, 120, 5, 1);
```

Let's check what we have done in game. We turn the quest in and… nothing happens. Sadly it is TrinityCore bug. At our core it works fine (or maybe on TC it needs another step? If anyone knows let me know). But if only you go out of the crypt and enter it again – the npcs will appear – the phase will be applied (the TC bug is: after you turn the quest in, the phases are not updated. Phases are only updated when entering a new zone). The workaround is SAI script:




```
-- Gameobject Broken Weapons Crate 205258 SAI
SET @ENTRY := 205258;
UPDATE gameobject_template SET AIName="SmartGameObjectAI" WHERE entry= @ENTRY;
DELETE FROM smart_scripts WHERE entryorguid=@ENTRY AND source_type=1;
INSERT INTO smart_scripts (entryorguid, source_type, id, link, event_type,
event_phase_mask, event_chance, event_flags, event_param1, event_param2,
event_param3, event_param4, action_type, action_param1, action_param2,
action_param3, action_param4, action_param5, action_param6, target_type,
target_param1, target_param2, target_param3, target_x, target_y, target_z,
target_o, comment) VALUES
(@ENTRY, 1, 0, 0, 20, 0, 100, 0, 0, 0, 0, 0, 44, 270, 1, 0, 0, 0, 0, 7, 0, 0, 0,
0, 0, 0, 0, "When player rewards any quest - Action invoker: Set phase id to 270
// ");
```

But it is an ugly workaround and the proper fix should be in core so that phases update after changing quest state.

With this workaround this works, but all creates disappear. It makes sense. Those gameobjects were in phase 0 and now we are in phase 270. We have to readd those gameobjects `205257, 205258, 205259`.

How can we find the IDs quickly? Use command `.gobject near 20`.

```
SET @GGUID := (SELECT MAX(guid)+1 FROM gameobject);
DELETE FROM gameobject WHERE id in (205257, 205258, 205259) and phaseid = 270;
INSERT INTO gameobject (guid, id, map, PhaseId, position_x, position_y,
position_z, orientation, spawntimesecs, animprogress, state) VALUES
(@GGUID + 0, 205259, 0, 270, 1067.57, -1907.65, 30.695, 4.95674, 120, 255, 1),
(@GGUID + 1, 205259, 0, 270, 1068.99, -1909.92, 30.695, 2.65289, 120, 255, 1),
(@GGUID + 2, 205257, 0, 270, 1067.8, -1903.96, 30.695, 4.08407, 120, 255, 1),
(@GGUID + 3, 205259, 0, 270, 1069.4, -1898.94, 30.825, 4.95674, 120, 255, 1),
```

```
(@GGUID + 4, 205257, 0, 270, 1064.06, -1899.78, 30.825, 4.08407, 120, 255, 1),
(@GGUID + 5, 205257, 0, 270, 1060.91, -1904.83, 30.695, 0, 120, 255, 1),
(@GGUID + 6, 205257, 0, 270, 1057.48, -1909.67, 30.695, 4.08407, 120, 255, 1),
(@GGUID + 7, 205259, 0, 270, 1057.2, -1904.23, 30.695, 4.39823, 120, 255, 1),
(@GGUID + 8, 205257, 0, 270, 1063.7, -1913.92, 30.825, 0, 120, 255, 1),
(@GGUID + 9, 205257, 0, 270, 1057.48, -1899.68, 30.825, 0, 120, 255, 1),
(@GGUID + 10, 205257, 0, 270, 1058.71, -1906.3, 30.695, 4.08407, 120, 255, 1),
(@GGUID + 11, 205258, 0, 270, 1060.9, -1894.8, 31.7765, 5.0091, 120, 255, 1),
(@GGUID + 12, 205259, 0, 270, 1059.66, -1908.35, 30.695, 4.95674, 120, 255, 1),
(@GGUID + 13, 205257, 0, 270, 1053.4, -1914.14, 30.825, 0.820303, 120, 255, 1),
(@GGUID + 14, 205257, 0, 270, 1047.99, -1903.93, 30.825, 0, 120, 255, 1),
(@GGUID + 15, 205257, 0, 270, 1054.28, -1903.9, 30.695, 0.85521, 120, 255, 1);
```

The last bug is no loot in npcs 1784 and 1783, but this is trivial and I am sure there is no need to show how to fix it ☺.

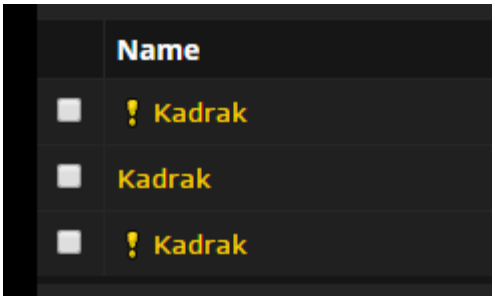
Great! Now those two quests are blizzlike!

# Kadrak phasing

*By Bandysc*

**DESCRIPTION:** NPC Kadrak is missing in Ashenvale.

**FIX:** We have to add the npc and give him proper phasing.




Again, for this issue sniff will be very helpful. Let's find the npc ID with wowhead.com.

It turns out, there are 3 npcs named Kadrak! If you visit their wowhead page, you will see Kadrak 8582 is sitting in The Mor'shan Rampart, Kadrak 33837 is in Splintertree Post. Kadrak 34006 has shown its position all the way from Mor'shan Rampart to Splitertree Post. If you played the Ashenvale in the retail, you probably know in one quest he drives us on the kodo, then since it is a quest npc, we will not add him.

With the sniff we can easily find precise position of both Kadraks:

```
[3] UpdateType: CreateObject1
[3] GUID: Full: 0xF130218600000B0C Type: Creature Entry: 8582 Low: 2828
[3] Position: X: 1229.058 Y: -2231.313 Z: 91.79168
[3] Orientation: 1.584056
[3] UNIT_FIELD_BYTES_1: 65536/9.18355E-41
[3] UNIT_FIELD_BYTES_2: 257/3.601337E-43
```

```
[12] UpdateType: CreateObject1
[12] GUID: Full: 0xF130842D00409373 Type: Creature Entry: 33837 Low: 4232051
[12] Position: X: 2284.792 Y: -2544.701 Z: 98.41223
[12] Orientation: 3.351032
[12] UNIT_FIELD_BYTES_1: 65536/9.18355E-41
[12] UNIT_FIELD_BYTES_2: 257/3.601337E-43
```

Adding them is trivial:

```
DELETE FROM creature WHERE id IN (8582, 33837);
SET @CGUID := (SELECT MAX(guid)+1 FROM creature);
INSERT INTO creature (guid, id, map, phaseId, position_x, position_y, position_z,
orientation, spawntimesecs, spawndist, MovementType) VALUES
(@CGUID+0, 8582, 1, 0, 1229.058, -2231.313, 91.79168, 1.584056, 120, 0, 0),
(@CGUID+1, 33837, 1, 0, 2284.792, -2544.701, 98.41223, 3.351032, 120, 0, 0);
```

Now we have to phase them. How? The answer is in the sniff. If you played retail, you should be recognize this type of phasing – it is invisibility auras. Then, let's check what auras does Kadrak wear.

```
ServerToClient: SMSG_AURA_UPDATE (0x1B8D) Length: 30 ConnIdx: 2 Time: 04/28/2014
20:11:42.748 Number: 3613
```

```
[0] Effect Mask: 1
[0] Flags: NoCaster, Positive (3)
[0] Spell ID: 65052 (65052)
[0] Caster Level: 31
[0] Charges: 0
[0] Slot: 0
GUID2: Full: 0xF130218600000B0C Type: Creature Entry: 8582 Low: 2828
```

```
ServerToClient: SMSG_AURA_UPDATE (0x1B8D) Length: 31 ConnIdx: 2 Time: 04/28/2014
20:41:58.880 Number: 26149
[0] Effect Mask: 1
[0] Flags: NoCaster, Positive (3)
[0] Spell ID: 65052 (65052)
[0] Caster Level: 31
[0] Charges: 0
[0] Slot: 0
GUID2: Full: 0xF130842D00409373 Type: Creature Entry: 33837 Low: 4232051
```

Let's check SpellWork. `65052` is a spell `Quest Invisibility 2` with aura `SPELL_AURA_MOD_INVISIBILITY` then it must be right spell. It is endless aura, we could make SAI for those mobs which would apply the aura, but there is better way – table `creature_template_addon` and column `auras`.

```
DELETE FROM creature_template_addon WHERE entry IN (8582, 33837);
INSERT INTO creature_template_addon (entry, bytes1, bytes2, auras) VALUES
(33837, 65536, 257, '65052'),
(8582, 65536, 257, '65052');
```

Fields bytes1 and bytes2 also comes directly from sniff. It is always good to add them since they bring small visual details.

At the moment both Kadraks are invisible. We have to apply invis detect aura on the player so that he will see them. The spell is in the sniffs, but we can find the spell in the other way. Usually Invisibility Detect spells are close to the Invis spell. So is the spell in this case: `65053 See Quest Invisibility 2`. We have to apply the aura on the player in `The Mor'shan Rampart` until player has quest `To The Rescue! (13712)` unfinished or not taken yet. Later player should receive aura in Splintertree Post. This is the knowledge we get from observation while playing ☺.

Again, firstly let's check area id using command `.gps`. For Mor'shan Rampart it is 1703 and for Splitertree Post it is 431.

```
DELETE FROM spell_area WHERE spell = 65053 AND area = 1703;
DELETE FROM spell_area WHERE spell = 65053 AND area = 431;
INSERT INTO spell_area (spell, area, quest_start, quest_start_status, quest_end,
quest_end_status, autocast) VALUES
```

```
(65053, 1703, 0, 0, 13712, 9, 1), -- shows Kadrak if player hasn't completed quest
13712 yet
(65053, 431, 13712, 66, 0, 0, 1); -- Shows Kadrak if player has quest 13712
rewarded or completed
```

We apply aura 65053 in area 1703 as long as (= quest_end) quest 13712 is in state QUEST_STATUS_NONE (1) or in state QUEST_STATUS_INCOMPLETE (8) = 1 | 8 = 9. Similarly we apply aura in area 431 since player has quest To The Rescue! QUEST_STATUS_COMPLETE (2) or QUEST_STATUS_REWARDED (64) = 2 | 64 = 66. Of course it should be autocasted then we set field `autocast` to 1 (true).

The last thing connected with Kadrak is a small detail – Kadrak 8582 should be moving. Let's fix it.

```
ServerToClient: SMSG_ON_MONSTER_MOVE (0x12D8) Length: 65 ConnIdx: 2 Time:
04/28/2014 20:11:49.378 Number: 3815
Float30: 0
Int28: 528514068
Float34: 0
Float2C: 0
Int40: 1048576
[0] Spline Waypoint: X: 1237.152 Y: -2251.238 Z: 92.10088
Int50: 718
Owner GUID: Full: 0xF13021860041DC6C Type: Creature Entry: 8582 Low: 4316268
Guid2: 0x0
```

If we use command `.go xyz 1237.152 -2251.238 92.10088` and `.go xyz 1229.058 -2231.313 91.79168` we will see those are exactly two points Kadrak moves between. The convention says we set waypoint ID as NPC entry multiplied by 10.

```
DELETE FROM waypoint_data WHERE id = 85820;
INSERT INTO waypoint_data (id, point, position_x, position_y, position_z) VALUES
(85820, 1, 1237.152, -2251.238, 92.10088),
(85820, 2, 1229.058, -2231.313, 91.79168);
```

And we set `path_id` in creature_template_addon.

```
UPDATE creature_template_addon SET path_id = 85820 WHERE entry = 8582;
```

We can't forget to set `MovementType` in creature to 2 so that core will know creature should use path.

```
UPDATE creature SET MovementType = 2 WHERE id = 8582;
```

# Ashes to Ashes

*By Bandysc*

**DESCRIPTION:** The quest doesn't work. Tent doesn't give kill credit.

**FIX:** Make tents give quest kill credit and disappear.

This is such simple quest we even do not need sniffs. Let's start our investigation what to do. In the quest we have to use object Krom'gar Flame Thrower on the tents. Wowhead gives big hint - the description of the item is a link to the spell used by the item. It turns out, when you click on the item, spell Flamethower is casted. Let's check what the spell do in the spellwork:

```
Effect 0: Id 3 (SPELL_EFFECT_DUMMY)
BasePoints = 0 (0)
Targets (46, 0) (TARGET_DEST_NEARBY_ENTRY, NO_TARGET)
```

```
Effect 1: Id 6 (SPELL_EFFECT_APPLY_AURA)
BasePoints = 0 (0)
Targets (1, 0) (TARGET_UNIT_CASTER, NO_TARGET)
Aura Id 23 (SPELL_AURA_PERIODIC_TRIGGER_SPELL), value = 0, misc = 0 (0), miscB =
0, periodic = 8000
ScalingMultiplier = 0, RandomPointsScalingMultiplier = 0,
ComboPointsScalingMultiplier = 0
Radius (Id 29) 6,00
   Trigger spell (78140) Incinerate. Chance = 0
```

It has two effects, the first one is dummy effect and has to be scripted manually. The second one is aura SPELL_AURA_PERIODIC_TRIGGER_SPELL - and this aura is already scripted in core - it means we do nothing and spell 78140 will trigger.

Then let's check what spell 78140 does.

```
Effect 0: Id 86 (SPELL_EFFECT_ACTIVATE_OBJECT)
BasePoints = 0 (0)
Targets (22, 51) (TARGET_SRC_CASTER, TARGET_GAMEOBJECT_SRC_AREA)
EffectMiscValueA = 12
Max Radius (Id 13) 10,00
```

It has one effect SPELL_EFFECT_ACTIVATE_OBJECT. This one is also scripted in core - then all objects (TARGET_GAMEOBJECT_SRC_AREA) in 10 yards (Max Radius (Id 13) 10,00) will be activated.

It means we just need simple Smart Script for the tents. How to check tents ID? Stand next to the tent and use command `.gobject near`. It turns out the tent has entry 203431. Let's write the script for the entry then. It it not that obvious, but SPELL_EFFECT_ACTIVATE_OBJECT changes object loot state to GO_ACTIVATED. And we have special event for this - "`Gameobject -> On loot state changed`". In properties of this event select GO_ACTIVATED. What should happen? The object should be deactivated. Then let's add an action "`Gameobject -> Set loot state`" with state "GO_JUST_DEACTIVATED" with

target SELF. Even though we do not kill anyone in the quest, all credits are "kill credits". We have built-in action "`Quest -> Give monster kill credit`".

How do we know what entry should we use? Let's check in quest_template table:

```
SELECT requirednpcorgo1 FROM quest_template WHERE id = 26010;
```

The output is 41936. Actually it is real mob entry - called "Northwatch Tent Quest Credit" though there is no spawn in the world :) Of course we set target of the action to ACTION_INVOKER.

Finally, our SAI looks like:

```
-- Gameobject Northwatch Tent 203431 SAI
SET @ENTRY := 203431;
UPDATE gameobject_template SET AIName="SmartGameObjectAI" WHERE entry= @ENTRY;
DELETE FROM smart_scripts WHERE entryorguid=@ENTRY AND source_type=1;
INSERT INTO smart_scripts (entryorguid, source_type, id, link, event_type,
event_phase_mask, event_chance, event_flags, event_param1, event_param2,
event_param3, event_param4, action_type, action_param1, action_param2,
action_param3, action_param4, action_param5, action_param6, target_type,
target_param1, target_param2, target_param3, target_x, target_y, target_z,
target_o, comment) VALUES
(@ENTRY, 1, 0, 1, 70, 0, 100, 0, 2, 0, 0, 0, 99, 3, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, "On loot state changed to GO_ACTIVATED - Self: Set loot state
GO_JUST_DEACTIVATED (only gameobjects) // "),
(@ENTRY, 1, 1, 0, 61, 0, 100, 0, 0, 0, 0, 0, 33, 41936, 0, 0, 0, 0, 0, 7, 0, 0, 0,
0, 0, 0, 0, " Linked - Action invoker: Give kill credit Northwatch Tent Quest
Credit (41936) // ");
```

Let's check if it works. The kill credit is given, but it turns out the tent doesn't disappear. Let's check gameobject_template table for the object.

```
SELECT flags FROM gameobject_template WHERE entry = 203431;
```

The object has flags 36 = 32 | 4 = GO_FLAG_NODESPAWN | GO_FLAG_INTERACT_COND. And our problem is solved. We want it to despawn - let's remove its 32 flag.

```
UPDATE gameobject_template SET flags = flags &~ 32 WHERE entry = 203431;
```

The quest now works blizzlike. However, we should add one more thing - the condition. At the moment spell 78140 will activate any gameobject in range! As long as the player use it in the tent area nothing bad would happen. But players can do many strange things - it may turn out some gameobject is activated even though it shouldn't be.

`Condition CONDITION_SOURCE_TYPE_SPELL_IMPLICIT_TARGET = 13 and CONDITION_OBJECT_ENTRY_GUID = 31` will force the target to tent only.

```
DELETE FROM conditions WHERE SourceTypeOrReferenceId=13 AND SourceGroup=1 AND
SourceEntry=78140;
INSERT INTO conditions
(SourceTypeOrReferenceId,SourceGroup,SourceEntry,ConditionTypeOrReference,Conditio
nTarget,ConditionValue1,ConditionValue2,Comment) VALUES
(13,1,78140,31,0,5,203431,"Implict target for spell 78140 to gobject 203431");
```

Done!

# Glorious Harvest

*By Bandysc*

**DESCRIPTION:** The quest doesn't work. Eggs are spawned, but they are not on the bears. They cannot be collected.

**FIX:** Make eggs collectible and make them ride bears.

As we can see in bare TC eggs are spawned, but they are not on the bears. Therefore we need to delete those spawns:

```
DELETE FROM creature WHERE id = 47203;
```

How to make eggs ride bears? It works the same as in quest Astranaar Bound. The bears are "Vehicles". To make creature a vehicle we need to find vehicle ID. It can be found in the sniffs in packet UPDATE_OBJECT:

```
[2] UpdateType: CreateObject1
[2] Object Guid: Full: 0x200F2400002E1900000021000009FAED Vehicle/0 R969/S33 Map:
0 Entry: 47204 Low: 654061[2] Object Type: Unit (3)
.
.
.
[2] RecID: 1251
```

The SQL is simple:

```
UPDATE creature_template SET vehicleId=1251 WHERE entry=47204;
```

In the previous quest, player casted spell after choosing a gossip option, here eggs should be spawned along with bear spawns. How to make them spawn instantly? Here comes the table vehicle_template_accessory, where we can place what npcs should ride on a vehicle after spawns.

```
DELETE FROM vehicle_template_accessory where entry=  47204;
INSERT INTO vehicle_template_accessory (entry, accessory_entry, seat_id,
description, summontype) VALUES
(47204, 47203, 0, 'Creeper Egg', 7),
(47204, 47203, 1, 'Creeper Egg', 7),
(47204, 47203, 2, 'Creeper Egg', 7),
(47204, 47203, 3, 'Creeper Egg', 7),
(47204, 47203, 4, 'Creeper Egg', 7),
(47204, 47203, 5, 'Creeper Egg', 7),
(47204, 47203, 6, 'Creeper Egg', 7),
(47204, 47203, 7, 'Creeper Egg', 7);
```

How do we know what seat_id should we use? For sniffs of course! For each bear I could find 8 eggs with:

```
[3] Transport Guid: Full: 0x200F2400002E1900000021000009FAED Vehicle/0 R969/S33
Map: 0 Entry: 47204 Low: 654061
[3] Transport Position: X: -0.9546 Y: 0.3 Z: 2.5791 O: 3.141595
[3] Transport Seat: XXX <---- the numbers from 0 to 7
```

However this is not enough. We have to tell core what spell should be casted to enter the vehicle. For some reason the table `npc_spellclick_spells` is used to do so. It makes no sense, but someone has written the core in such way :) What spell is casted? Again - here comes the sniff.

```
ServerToClient: SMSG_AURA_UPDATE (0x10AF) Length: 71 ConnIdx: 0 Time: 01/05/2016
01:28:13.953 Number: 5367
UpdateAll: True
UnitGUID: Full: 0x1C0F2400002E18C0000021000089FAED Creature/0 R969/S33 Map: 0
Entry: 47203 Low: 9042669
AurasCount: 1
[0] Slot: 0
[0] HasAura: True
[0] SpellID: 87978 (87978)
[0] SpellXSpellVisualID: 0
[0] Flags: NoCaster, Positive (3)
[0] ActiveFlags: 2
[0] CastLevel: 1
[0] Applications: 0
[0] Int56 Count: 0
[0] Effect Count: 0
[0] HasCastUnit: False
[0] HasDuration: False
[0] HasRemaining: False
```

I have checked the spell 87978 in spell work and it is spell Creeper Egg with following effects:

```
Effect 0: Id 6 (SPELL_EFFECT_APPLY_AURA)
Targets (25, 0) (TARGET_UNIT_TARGET_ANY, NO_TARGET)
Aura Id 236 (SPELL_AURA_CONTROL_VEHICLE), value = 0, misc = 0 (0), miscB = 0,
periodic = 0
ScalingMultiplier = 0, RandomPointsScalingMultiplier = 0,
ComboPointsScalingMultiplier = 0
```

```
Effect 1: Id 6 (SPELL_EFFECT_APPLY_AURA)
Targets (1, 0) (TARGET_UNIT_CASTER, NO_TARGET)
Aura Id 4 (SPELL_AURA_DUMMY), value = 0, misc = 0 (SPELLMOD_DAMAGE), miscB = 0,
periodic = 0
ScalingMultiplier = 0, RandomPointsScalingMultiplier = 0,
ComboPointsScalingMultiplier = 0
```

It has aura "Control vehicle" then it must be right spell.

```
DELETE FROM npc_spellclick_spells WHERE npc_entry = 47204;
INSERT INTO npc_spellclick_spells (npc_entry, spell_id, cast_flags) VALUES
(47204, 87978, 0);
```

The next part is to make eggs collectible. Now it is task for real spell_click spell. It is WoW feature - if npc has npc_flag 16777216, it means when player click on it, a spell is casted.

```
[19] UpdateType: CreateObject1
[19] Object Guid: Full: 0x1C0F2400002E18C00000240000097FEC Creature/0 R969/S36
Map: 0 Entry: 47203 Low: 622572
[19] Object Type: Unit (3)
.
.
.
[19] UNIT_NPC_FLAGS: 16777216
```

```
UPDATE creature_template SET npcflag=16777216 WHERE entry = 47203;
```

What spell should be casted? Let's search for packet CMSG_SPELL_CLICK

```
ClientToServer: CMSG_SPELL_CLICK (0x150A) Length: 14 ConnIdx: 2 Time: 01/05/2016
01:28:21.672 Number: 5573
SpellClickUnitGUID: Full: 0x1C0F2400002E18C0000020000189FB93 Creature/0 R969/S32
Map: 0 Entry: 47203 Low: 25820051
TryAutoDismount: False
```

It doesn't contain info about spell, but when we search for nearest packet with GUID 0x1C0F2400002E18C0000020000189FB93, it turns out the spell is 88695

```
ServerToClient: SMSG_SPELL_START (0x1077) Length: 110 ConnIdx: 0 Time: 01/05/2016
01:28:22.125 Number: 5588
(Cast) CasterGUID: Full: 0x1C0F2400002E18C0000020000189FB93 Creature/0 R969/S32
Map: 0 Entry: 47203 Low: 25820051
(Cast) CasterUnit: Full: 0x1C0F2400002E18C0000020000189FB93 Creature/0 R969/S32
Map: 0 Entry: 47203 Low: 25820051
(Cast) CastID: 0
(Cast) SpellID: 88695 (88695)
(Cast) SpellXSpellVisualID: 34767
(Cast) CastFlags: 10
(Cast) CastTime: 0
(Cast) HitTargetsCount: 0
(Cast) MissTargetsCount: 0
```

```
(Cast) MissStatusCount: 0
(Cast) (Target) Flags: Unit (2)
(Cast) (Target) HasSrcLocation: False
(Cast) (Target) HasDstLocation: False
(Cast) (Target) HasOrientation: False
(Cast) (Target) Unit: Full: 0x080F2400000000000000000A76ED7 Player/0 R969/S0
Map: 0 Low: 10972887
```

The cast_flags is 3, because the one who cast the spell is player (check the wiki for more info).

```
DELETE FROM npc_spellclick_spells WHERE npc_entry = 47203;
INSERT INTO npc_spellclick_spells (npc_entry, spell_id, cast_flags) VALUES
(47203, 88695, 3);
```

Let's try it. You should be able to collect the eggs, but it turns out the eggs doesn't disappear. Of course it makes sense - we haven't said the core that the egg should disappear. Simple SAI will do the trick:

```
 -- Creature Creeper Egg 47203 SAI
SET @ENTRY := 47203;
UPDATE creature_template SET AIName="SmartAI" WHERE entry= @ENTRY;
DELETE FROM smart_scripts WHERE entryorguid=@ENTRY AND source_type=0;
INSERT INTO smart_scripts (entryorguid, source_type, id, link, event_type,
event_phase_mask, event_chance, event_flags, event_param1, event_param2,
event_param3, event_param4, action_type, action_param1, action_param2,
action_param3, action_param4, action_param5, action_param6, target_type,
target_param1, target_param2, target_param3, target_x, target_y, target_z,
target_o, comment) VALUES
(@ENTRY, 0, 0, 1, 73, 0, 100, 0, 0, 0, 0, 0, 83, 16777216, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, "On spellclick - Self: Remove npc flags UNIT_NPC_FLAG_SPELLCLICK //
"),
(@ENTRY, 0, 1, 0, 61, 0, 100, 0, 0, 0, 0, 0, 41, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, " Linked - Self: Despawn instantly // ");
```

The quest is almost ready. Small detail is: we can collect eggs even without quest `Glorious Harvest` (88695). We need one condition for this one with sourcetype: `SOURCE_TYPE_SPELL_CLICK_EVENT` (18) and condition `CONDITION_QUESTTAKEN` (9).

```
DELETE FROM conditions WHERE SourceTypeOrReferenceId=18 AND SourceGroup=47203 AND
SourceEntry=88695;
INSERT INTO conditions
(SourceTypeOrReferenceId,SourceGroup,SourceEntry,ConditionTypeOrReference,Conditio
nValue1,Comment) VALUES
(18,47203,88695,9,28114,"Let player spell click npc 47203 only if quest 28114 is
active");
```

Great! It should work now. However, after setting vehicleId a small bug appeared. We can attack the bear, but he doesn't want to attack us. It turns out vehicles by default doesn't attack. But if we set AIName to "SmartAI", he will attack again.

```
UPDATE creature_template SET AIName="SmartAI" WHERE entry = 47203;
```

The last small visual detail is: on retail, bears start with HP between 210-360. To fix it we have to disable health regeneration:

```
UPDATE creature_template SET regenhealth=0 WHERE entry=47204;
```

And update curhealth in creature table. We will use built-in Random function to MySQL.

```
UPDATE creature SET curhealth=(FLOOR(RAND() * 140) + 210) WHERE id = 47204;
```

Voila!

# Astranaar Bound

*By Bandysc*

**DESCRIPTION:** The quest can be completed but is not scripted at all. We should be able to talk to quest giver to ride to Sentinel Luara at Astranaar and avoid Horde army.

**FIX:** Add gossip menu and write a script to it.

It is almost impossible to script the quest 100% blizzlike without any sniffs.

As you can see in the sniff logs, once player accepts the quest, gossip menu appears. Once he choose "Sentinel, I would like to ride one of your nightsabers to Astranaar", spell 63020 is casted, then spell 63022 is casted, finally spell 63021. NPC 33452 is summoned and he starts running.

In this sniff there were a lot of spells, how do I know that those are the spells that we need? I've checked every spell.

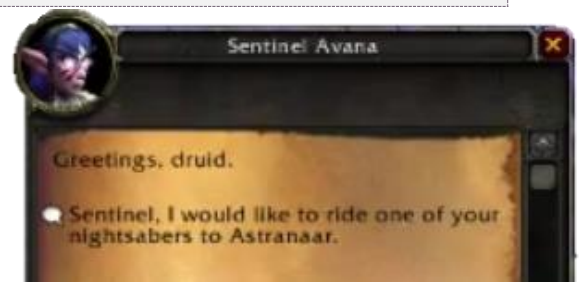
| Spell ID | Name | Effect | Description |
|---|---|---|---|
| **63020** | Astranaar Bound: Character Force Cast From Gossip | EFFECT_FORCE_CAST | Makes target cast spell 63022 |
| **63022** | Astranaar Bound: Summon Avana's Nightsaber | EFFECT_SUMMON | Summons NPC with entry 33452 |
| **63021** | Riding Nightsaber to Astranaar | EFFECT_APPLY_AURA | Gives aura AURA_CONTROL_VEHICLE – caster can ride on target |

For each WoW version parsed sniff output differs a bit. Currently, for newest WoD version, parser shows exactly who is the caster and who is the target. Since the sniff comes from older WoW version when parser was not yet that advanced, I had to play a little guessing game.

Before we script the NPC we have to add a gossip option. According to sniff:

```
[0] Required money: 0
[0] Text: Sentinel, I would like to ride one of your nightsabers to Astranaar.
[0] Index: 0
[0] Icon: Chat (0)
[0] Box Text:
[0] Box: false
Text Id: 14347
Menu Id: 10339
```

Each gossip menu can contain few gossip menu options. It this case, we have only one. Each gossip menu has its own ID (here: 10339 - it should be taken from sniffs). The text shown in the gossip also has its own ID (here: 14347). It means different gossip menus can have same text ID. Let's add a row to the `gossip_menu` table.




```
DELETE FROM gossip_menu WHERE entry = 10339 AND text_id = 14347;
INSERT INTO gossip_menu (entry, text_id) VALUES
(10339, 14347);
```

Of course we need remove the gossip menu we insert first, so that SQL can be executed many times and still work (if we didn't remove the row first, on the second time MySQL would output an error).

Where does text_id come from? Text_id comes from table npc_text. As you can check yourself, TrinityCore already contains text_id 14347 in table npc_text. Probably some other npc uses it. Then there is no need to remove and insert it again. However, if there would've been such need, SQL would look like this:

npc_text:

| | |
|---|---|
| ID | Text_id from sniff |
| Text0_0 | Male English text |
| Text0_1 | Female English text |
| Prob0 | Probability |
| VerifiedBuild | Build of sniff |

```
DELETE FROM npc_text WHERE ID = 14347;
INSERT INTO npc_text (ID, text0_0, text0_1, prob0, VerifiedBuild) VALUES (14347,
'', 'Greetings, $c.', 1, 15595);
```

The npc_text can be also found in sniff as long as it has been properly done. In this case, author of sniff didn't clear his cache, in result it wasn't sent to the client = it couldn't be sniffed.

Time to add a gossip menu_option:

```
DELETE FROM gossip_menu_option WHERE menu_id = 10339;
INSERT INTO gossip_menu_option (menu_id, id, option_icon, option_text, option_id,
npc_option_npcflag) VALUES
(10339, 1, 0, "Sentinel, I would like to ride one of your nightsabers to
Astranaar.", 1, 1);
```

There is no any magic in this. We have added gossip menu 10339, but in game, the NPC still doesn't have the menu. Why? We haven't assigned the menu to the npc, yet ☺. Creature_template has column gossip_menu_id.

```
UPDATE creature_template SET gossip_menu_id = 10339 WHERE entry = 33445;
```

Now you can run the game and check if NPC has any gossip menu. Finally the time has come to write the script.

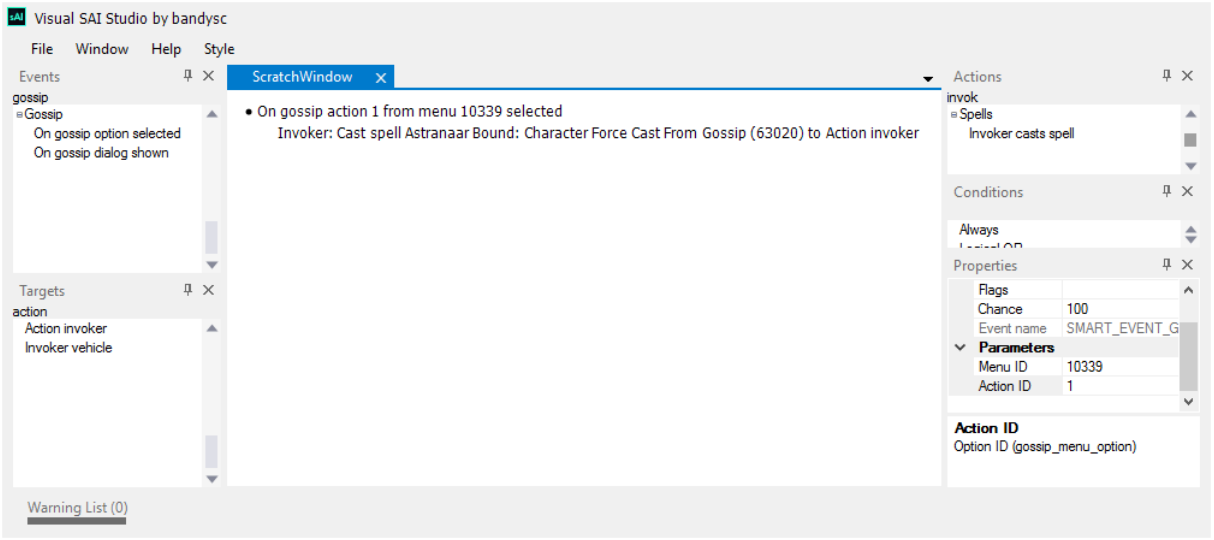
Let's think what should happen: when player selects gossip option #1, NPC should cast spell 63020 on the player. Ok, now open Visual SAI Studio.

1. Start a new script for NPC 33445.
2. Drag event "On gossip menu option selected"
3. Select the event and change menu ID to 10339, action ID to 1.
4. Drag action "Invoker cast"
5. Select the action and change spell ID to 63020.
6. Find target "Action invoker" and drag it over to our action.

**RECOMMENDED TOOL:**
**NAME:** Visual SAI Studio
**AUTHOR:** bandysc
**DOWNLOAD:** here

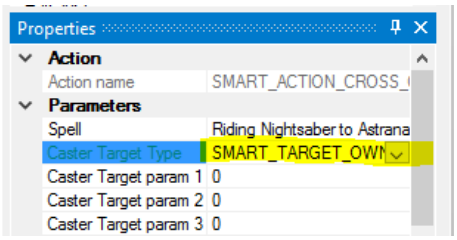
Only tool required to write any script in SAI. Everything is well explained and you only have to drag and drop proper event, action and target.

Visual SAI Studio should look like:




And generated SAI should look like this:

```
SET @ENTRY := 33445;
UPDATE creature_template SET AIName="SmartAI" WHERE entry= 0;
DELETE FROM smart_scripts WHERE entryorguid=@ENTRY AND source_type=0;
INSERT INTO smart_scripts (entryorguid, source_type, id, link, event_type,
event_phase_mask, event_chance, event_flags, event_param1, event_param2,
event_param3, event_param4, action_type, action_param1, action_param2,
action_param3, action_param4, action_param5, action_param6, target_type,
target_param1, target_param2, target_param3, target_x, target_y, target_z,
target_o, comment) VALUES
(@ENTRY, 0, 0, 0, 62, 0, 100, 0, 10339, 1, 0, 0, 85, 63020, 0, 0, 0, 0, 0, 7, 0,
0, 0, 0, 0, 0, 0, "On gossip action 1 from menu 10339 selected - Invoker: Cast
spell Astranaar Bound: Character Force Cast From Gossip (63020) to Action
invoker");
```




Simple and self-explaining. We are also in need of a script for summoned unit Avana's Nightsaber ID 33452. When the NPC is summoned, player should cast spell 63021 on the Nightsaber. For this we need event SMART_EVENT_JUST_SUMMONED, action SMART_ACTION_CROSS_CAST and target TARGET_SELF. This action let us specify who should cast the spell. In this case spell should be casted by SMART_TARGET_OWNER_OR_SUMMONER. Now, select the action and select the target from action parameters.

What should happen next? Once player begins his adventure riding Nightsaber, the mount should start running. There is no need to add an additional, same event. One event can have multiple actions. Drag action SMART_ACTION_WP_START. Smart AI uses paths from table waypoints. We'll add path in the second. Path is identified by ID, which is the same as NPC entry. Set action parameters according to the table on the left.

| Parameters | |
|---|---|
| Run/Walk | Run |
| Path ID | 33452 |
| Repeat | False |
| Quest | Unknown Quest (0) |
| Despawn time | 1000 |
| React State | Passive |

Table waypoints are really simple. We just have to specify NPC `entry`, `point ID` (starting from 1) and `position_x, position_y, position_z`. In this case, whole path is sent directly to client in one packet. Thanks to this, preparing the SQL will be really simple. Let me skip the whole sql query, it will be quite long so I will give you an example how it should start.

Path (small portion) in sniff:

```
[0] Waypoint: X: 3289.608 Y: 159.8615 Z: 9.196117
[1] Waypoint: X: 3282.608 Y: 167.3615 Z: 9.946117
[2] Waypoint: X: 3266.608 Y: 170.6115 Z: 10.19612
[3] Waypoint: X: 3255.358 Y: 169.8615 Z: 9.196117
[4] Waypoint: X: 3237.608 Y: 167.3615 Z: 8.696117
...
```

And our SQL query:

```
DELETE FROM waypoints WHERE entry = 33452;
INSERT INTO waypoints (entry, pointid, position_x, position_y, position_z) VALUES
(33452, 1, 3289.608, 159.8615, 9.196117),
(33452, 2, 3282.608, 167.3615, 9.946117),
(33452, 3, 3266.608, 170.6115, 10.19612),
(33452, 4, 3255.358, 169.8615, 9.196117),
(33452, 5, 3237.608, 167.3615, 8.696117),
...
```

Whole SQL for NPC 33452 looks like this. I have left only values to save some space ☺

```
(@ENTRY, 0, 0, 1, 54, 0, 100, 0, 0, 0, 0, 0, 86, 63021, 0, 23, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, "On just summoned - SMART_TARGET_OWNER_OR_SUMMONER: Cast spell
Riding Nightsaber to Astranaar (63021) at Self"),
(@ENTRY, 0, 1, 0, 61, 0, 100, 0, 0, 0, 0, 0, 53, 1, 33452, 0, 0, 1000, 0, 0, 0, 0,
0, 0, 0, 0, 0, " Linked - Self: Start path #33452, run, do not repeat, Passive");
```

It might seem like the quest is finally scripted, but it is not finished yet. In the quest, we are riding an NPC, however, creatures are not “rideable” by default. In order to make them “rideable”, we have to change them to **VEHICLE**. It is really simple – we just have to set a field `VehicleID` to proper ID. How to find the proper ID? We have to take a look at the sniff. Vehicle ID is sent to client when creature is created – in packet `SMSG_UPDATE_OBJECT`. Remember the name! We will often look for this packet because it contains a lot of useful data for `creature_template` and `creature_template_addon` table.

Most important things for the npc from the sniff.

```
[0] GUID: Full: 0xF15082AC00648184 Type: Vehicle Entry: 33452 Low: 6586756
[0] Fly Speed: 7
[0] Run Back Speed: 2.5
[0] Run Speed: 18
[0] Walk Speed: 2.5
[0] Vehicle Id: 364
```

```
[0] UNIT_FIELD_FLAGS: 33288/4.664642E-41
[0] UNIT_FIELD_FACTIONTEMPLATE: 4/5.605194E-45
```

Note: in newer Packet Parser Vehicle ID can be found as RecId

Sniff shows us directly that the NPCs faction is 4 (UNIT_FIELD_FACTIONTEMPLATE), unit flags are 33288 (UNIT_FIELD_FLAGS), vehicle ID is 364 (Vehicle Id). Let's update NPC 33452

```
UPDATE creature_template SET vehicleid = 364, unit_flags=unit_flags|33288,
speed_run=2.57 WHERE entry = 33452;
```

"Hold on a second, why speed_run is set to 2.57?" – is what you ask yourself right now. Good question!

speed_run in database is result of division of Run Speed by 7. speed_walk is result of division of Walk Speed by 2.5.

Finally, now you can give this script a try. It should work almost splendidly by now. Well, almost… There is yet, one small issue. Gossip menu option is always visible, even if you haven't accept the quest. It should be available only if you have taken the quest (and have it active) or you have already completed the quest.

To limit the gossip menu (and not only) we need a condition. Conditions can also be stored in database – in table conditions (makes sense). We have to specify what we want to limit (SourceTypeOrReferenceId) and how we want to limit (ConditionTypeOrReference).

We want to limit visibility of gossip menu – check TC wiki:

```
CONDITION_SOURCE_TYPE_GOSSIP_MENU_OPTION = 15,
```

How do we want to limit it? It should be visible when player has the quest in quest log (note: this quest is marked as completed as soon as it is taken) or has the quest marked as rewarded. Again, take a look at wiki:

```
CONDITION_QUESTREWARDED = 8

CONDITION_QUEST_COMPLETE = 28
```

SourceGroup, SourceEntry and ConditionTarget change its meaning depending on source type. For source TYPE_GOSSIP_MENU_OPTION:

| | |
|---|---|
| **SourceGroup** | Gossip menu entry (gossip_menu_option.menu_id) |
| **SourceEntry** | Gossip menu option id (gossip_menu_option.id) |
| **SourceId** | Unused (always 0) |
| **ConditionTarget** | 0 - Player for which gossip text is shown<br>1 - Object providing gossip |

ConditionValue1, ConditionValue2, ConditionValue3 also depend on condition. For CONDITION_QUEST

| | |
|---|---|
| **ConditionValue1** | Quest ID |
| **ConditionValue2** | Unused (always 0) |
| **ConditionValue3** | Unused (always 0) |

Let's write SQL then!

| Column | Row 1 | Row 2 |
|---|---|---|
| **SourceTypeOrReferenceId** | 15 | 15 |
| **SourceGroup** | 10339 | 10339 |
| **SourceEntry** | 1 | 1 |
| **SourceId** | 0 | 0 |
| **ConditionTypeOrReference** | 8 | 28 |
| **ConditionTarget** | 0 | 0 |
| **ConditionValue1** | 13646 | 13646 |
| **ConditionValue2** | 0 | 0 |
| **ConditionValue3** | 0 | 0 |

This, however, will not work. Why you may ask. Because the whole condition for particular source is met if all rows/entries for the same source type are met. The gossip would be visible if player has at once quest 13646 in his quest log and is rewarded. This of course can never happen. Instead of logical AND (quest is in quest log AND is rewarded) we need logical OR (quest is in quest log OR is rewarded). In order to do so, there is one column that we need – `ElseGroup`.

> All entries belonging to the same condition (same SourceType, SourceGroup and SourceEntry) that share the same number in ElseGroup, define one group. The entire condition is met when any of its groups is met.
>
> http://collab.kpsn.org/display/tc/conditions#conditions-ElseGroup

Then, one row has to have ElseGroup 0, second any other number. Of course you can have multiple conditions in single group. Therefore, the final SQL looks like this:

```
DELETE FROM conditions WHERE SourceTypeOrReferenceId=15 AND SourceGroup=10339 AND
SourceEntry=1;
INSERT INTO conditions (SourceTypeOrReferenceId, SourceGroup, SourceEntry,
ElseGroup, ConditionTypeOrReference, ConditionTarget, ConditionValue1, Comment)
VALUES
(15, 10339, 1, 0, 28, 0, 13646, "Show gossip option if player has quest Astrannar
Bound completed"),
(15, 10339, 1, 1, 8, 0, 13646, "Show gossip option if player has quest Astrannar
Bound rewarded");
```

Quest is now 100% blizzlike!

# Extras

*By Bandysc*

## Sniffing

Retail core and wow database are Blizzard's top secret data. What we want to achieve are scripts and spawns as similar to retail as possible. Even though official database is not revealed, scripting is not entirely blind guess. Somehow server has to send some data to client and we can catch the data. Of course, there is no magic. Sniffs are not recipe how to script content, but direct orders for client (e.g. "move NPC #1233 to position X: 1232.3, Y: 1829.1 Z: 120", "let NPC #421 cast spell 4123" - in form understandable for the computer). Still, those data is very valuable as it makes scripting content better and more accurate (we do not have to guess waypoints - they are in sniffs, we do not have to rewrite all texts - they are in sniffs, etc.).

To sniff we need special application – sniffer. The best one is Whiff by Zedron. Just run retail wow, then run sniffer. Every data (excluding login info) sent to/from server is now saved into a file.

> Do not sniff private servers. It is pointless.
>
> Usually it is good to clear cache before sniffing
>
> Usually you should start sniffer before going into the world.

Of course server doesn't save messages readable by humans. Server/client sends bare numbers, which have to be decoded next. We need another application which could "translate" sniffs into human readable text. It is task for Packet Parser by Trinitycore.

## Phasing

Though phasing first appeared in Wrath of the Lich King, in Cataclysm phasing is especially often used. Actually, all revamped zones are more or less phased though it is nothing compared to the new Cata zones.

## Phases

Generally phasing is done either with phase IDs or phase masks depending on core implementation. TrinityCore 4.3.4 (and 6.x) uses phase IDs, SkyFire 4.0.6 (and Atlantiss) uses phase masks. What is the difference between those two implementation? Of course you can use multiple masks at once, but you are limited to few phases (31 phases??), with phase IDs you have almost unlimited amount of phases, but NPC can be at one phase ID at once. Does Blizzard use IDs or masks? We do not know. In phase spells we can guess value A is phase mask and value B is phase ID. Most spells have both value A and value B. On the other hand some spells have no longer Value A – only phase ID. Then Blizzard might have changed their mind on phase storage.

Each world object (npc, gameobject, player, etc.) has its own phase. You can see only those NPCs who are in your phase. Phases are used when a lot of NPC should change depending in quest. Quick example: Gilneas. After you turn first quest Lockdown! All previous creatures disappear and worgens with humans fighting appear.

Since Cataclysm phase can change not only NPCs but also map (terrain):

How to set creature phase? Nothing more simple. You just have to set `PhaseMask` or `PhaseId` in table creature. How to detect NPC from phase? We need to be in same phase. Usually player phase is applied with spells. It you guess you are phased, it is always best to check spells in sniff (and check their effect in Spell Work). Then we can add a row to `spell_area`. However, not all phases are made with spells. Probably it depends on Blizzard developer preference. You will not find any phase spells in Kezan. It of course makes things harder – you have to decide what phase you should apply (spells usually have its phase given). To apply phase without spell use table `phase_area`.

## Invisibility auras

This type is designed for single npc phasing – when only one creature is phased (e.g. depending on quest) and the rest stay intact. This is achieved really simple. NPC has (usually) permanent invisibility aura. Player cannot see invisible npc unless he has other aura – invisibility detect. Each invis and invis detect spell has its ID. `Spell Generic Quest Invisibility 1 (49414)` has invisibility ID 7 (Misc B Value), if you have any spell with aura `SPELL_AURA_MOD_INVISIBILITY_DETECT` and same misc B value – you will see the npc. However, you shouldn't use any spell. It is always best to use invis spells from sniff. You will always find them – in packet `AURA_UPDATE`. In general – npc has permanent invisibility aura taken from sniff usually set in `creature_tenplate_addon`. Player invisibility detect aura can be set in table spell_area.

## Personal phases

Last type of phasing is not supported by any open source wow core – it is kind of "personal" phase – npcs you see are your own personal spawns. It is used by some quests – e.g. Basic Botany (Plants vs Zombies chain) in Hillsbard Foothills. Only you can see zombies and plants you place.