Minecraft Item Database Documentation

By: Jakob Surgeon

Class: Database Programming (CPSC 2100)

Professor: Dr. Matthew Preston

Date: 2023-10-19

Dataset Selection

For my dataset I chose data on all the items in Minecraft Java Edition (up until 1.19) with associated properties of each item. I am basing this database off of the dataset from an open-source project called the Minecraft Item Property Encyclopedia (GitHub Repository) with some personal modifications to add some other useful item information. This dataset is well suited for a database as opposed to a flat or semi-structured file since there are many items (over 1000) and many attributes that do not apply to each item. In its base state you can easily look through the table that is the dataset and see many instances of null values where certain item properties do not apply. In addition, there are some properties that could be reduced by adding lookup tables for them if they fall in a certain category making less properties (attributes) overall. By transforming this dataset to a database, I can make searching for information for any given item or mix of properties more streamlined and less prone to anomalies. Lastly, this dataset is also a good choice for me personally as someone who has been playing Minecraft since I got my first Kindle Fire 10 years ago or so.

Problem Statement

The best-selling video game of all time, Minecraft, at its core is an open-world, 3D, block game with endless possibilities and things to do. Over time Minecraft has received consistent updates to not just patch the game but add new content which introduces many new items. As a result of this it is difficult for novice players to learn about the items of the game and for veteran players to keep up with new ones. This is a big issue, as to play most of the game modes (especially survival) in Minecraft, item management is a key aspect. If a player does not know what different items do and what their purpose is they are unable to play the game. A database is an excellent solution for this issue as it will allow players to easily search up and find information on items by using the item’s display name or ID. In addition, by focusing specifically on items’ game properties Minecraft map developers could find items based off their properties to use for their map’s needs. For example, if a Minecraft map developer was making a hunger games map and wanted to make sure the weapons spread throughout were balanced, they could sort items based off of if they are a weapon, what their attack speed is, and what their attack damage is to get a digestible list of the weapons in the game to figure out how they should be spread. Lastly, as updates add, remove, or change items in the game the database could be easily updated to match the current items in the game.

Requirements Outline

I. **What It Can Do**

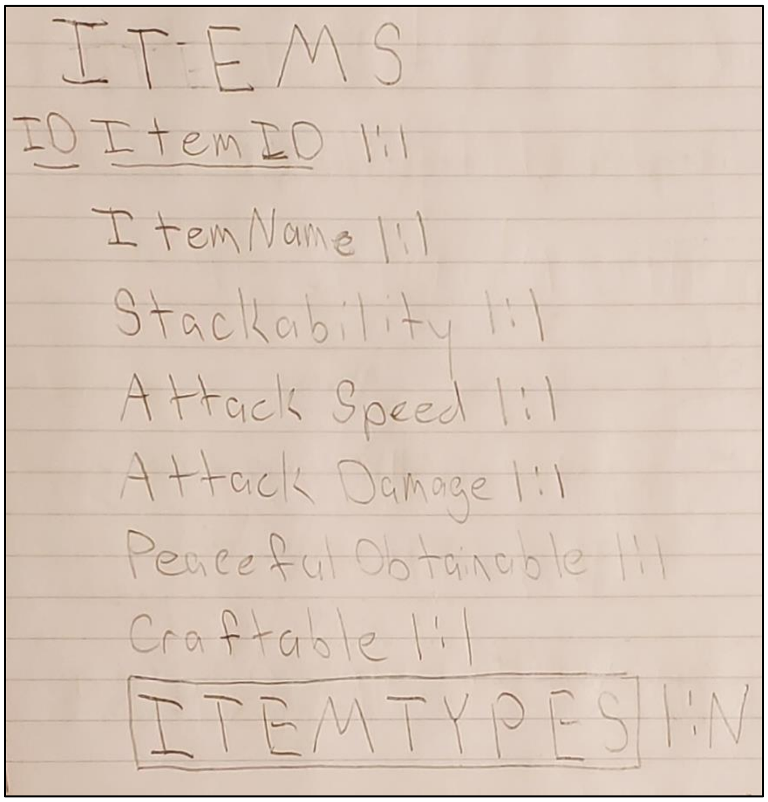
- Users can find a specific item and its associated properties by searching for that item with its display name or item ID.
    - If nothing is found when searching an error message will be displayed.
- Users can find a list of items that fall into a specific item type, property values, or both.
    - They will be able to organize these lists by a given attribute in ascending or descending order.
    - They will be able to select a specific item from this list to look at more closely.
- Users can customize their view by showing or not showing specific properties of whatever items they are looking for.
    - If the item(s) do(es) not hold specific properties those properties will not be included as viewable to the user.
    - The item display name attribute will always be displayed.
- Users can view a full list of all items with the properties that apply to all items.
    - This essentially will be a view of the “items” table.
- Users can prompt for a description of the different properties (attributes) to give context on what different properties of any given item mean.
    - The information for this will not be stored in the database itself but in the front end.
- Users can find information on item properties (attributes) that are not stored in the database but calculated in the front end (i.e. damage per second)

II. **What It Cannot Do**

- The search function of the database will be picky and will not account for spelling errors but will not be case sensitive.
- Users will not be able to update the database themselves to avoid incorrect data from user error and to protect against SQL injection.
    - Instead, the database will be updated (if updated at all) as Minecraft updates are rolled out by the support team (me)
- Even though crafting is an essential part of Minecraft items crafting recipes will not be included as part of the database as it introduces too many complications.
    - A property (attribute) of if the item can be crafted will be available though.

- Users will not be able to search up items by their numerical item ID nor is it included in this database since numerical IDs were discontinued in 1.8 and do not apply to any items added afterwards.

Semantic Data Model


ITEMS

ID ItemID 1:1

ItemName 1:1

Stackability 1:1

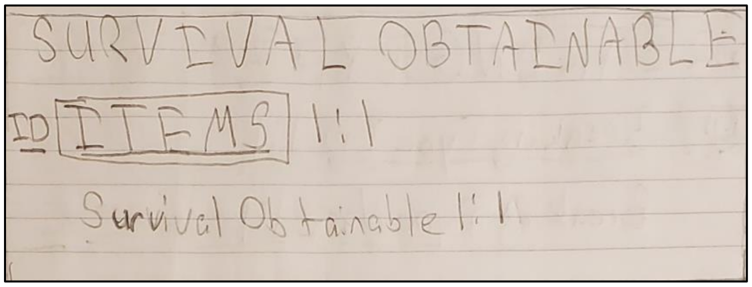
Attack Speed 1:1

Attack Damage 1:1

Peaceful Obtainable 1:1

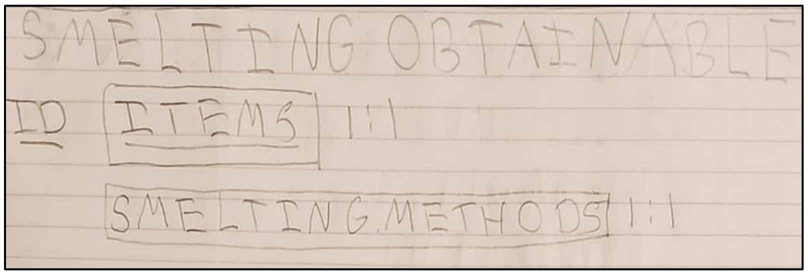
Craftable 1:1

$\boxed{\text{ITEMTYPES}}$ 1:N


SURVIVAL OBTAINABLE

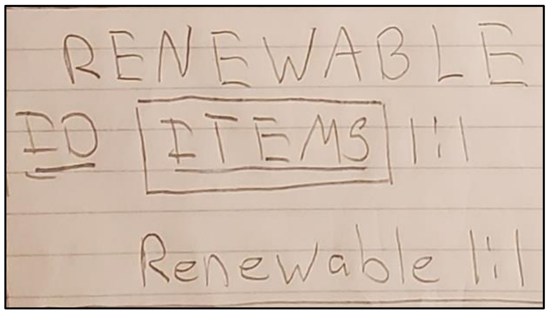
ID $\boxed{\text{ITEMS}}$ 1:1

Survival Obtainable 1:1


SMELTING OBTAINABLE

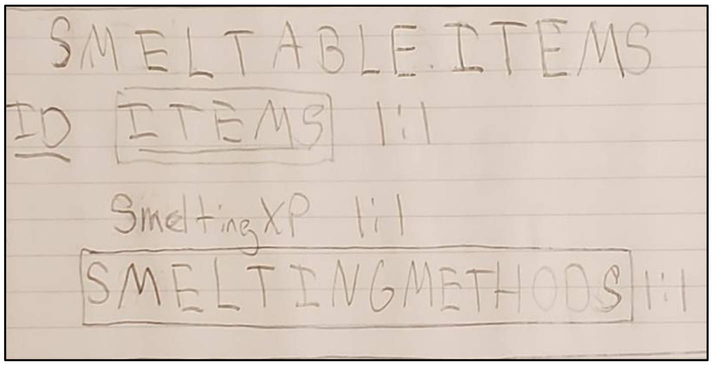
ID $\boxed{\text{ITEMS}}$ 1:1

$\boxed{\text{SMELTINGMETHODS}}$ 1:1


RENEWABLE

ID $\boxed{\text{ITEMS}}$ 1:1

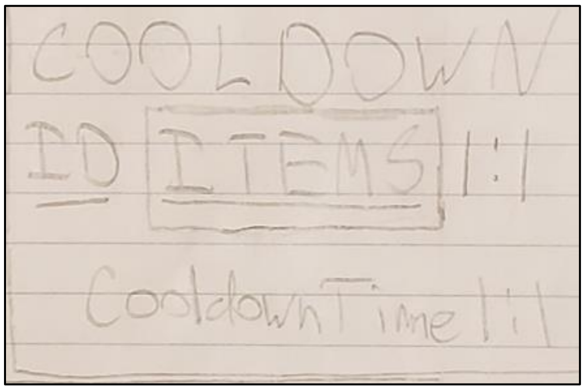
Renewable 1:1


SMELTABLE ITEMS

ID $\boxed{\text{ITEMS}}$ 1:1

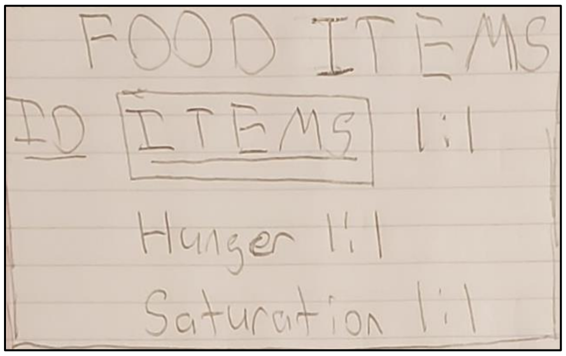
SmeltingXP 1:1

$\boxed{\text{SMELTINGMETHODS}}$ 1:1


COOLDOWN

ID $\boxed{\text{ITEMS}}$ 1:1

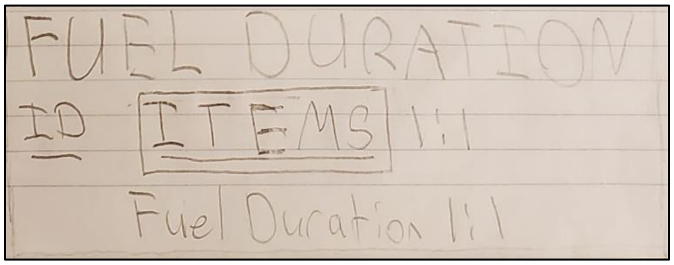
CooldownTime 1:1


FOOD ITEMS

ID $\boxed{\text{ITEMS}}$ 1:1

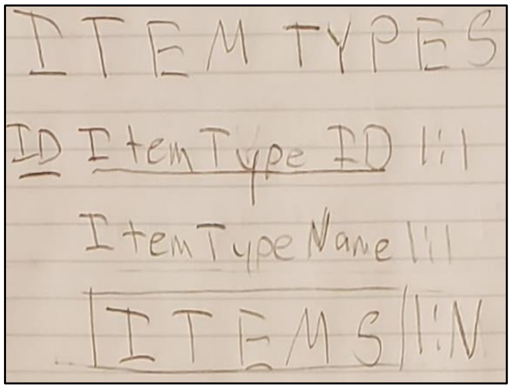
Hunger 1:1

Saturation 1:1


FUEL DURATION

ID $\boxed{\text{ITEMS}}$ 1:1

Fuel Duration 1:1

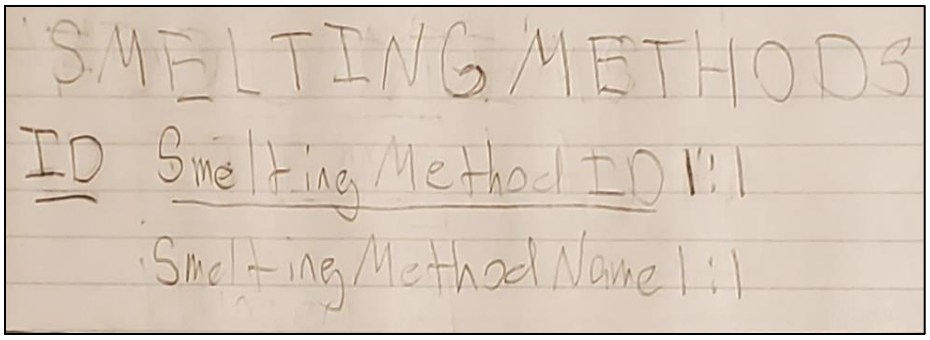


ITEM TYPES

$\underline{\text{ID}}$ $\underline{\text{ItemType ID}}$ 1:1

ItemTypeName 1:1

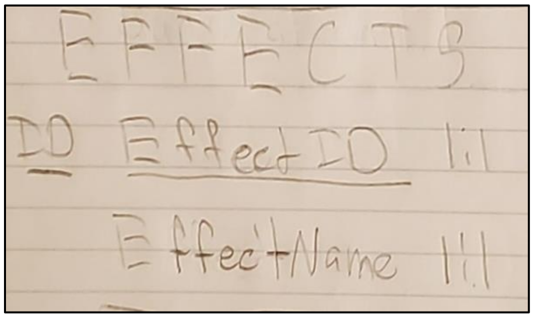
$\boxed{\text{ITEMS}}$ 1:N



SMELTING METHODS

$\underline{\text{ID}}$ $\underline{\text{Smelting Method ID}}$ 1:1

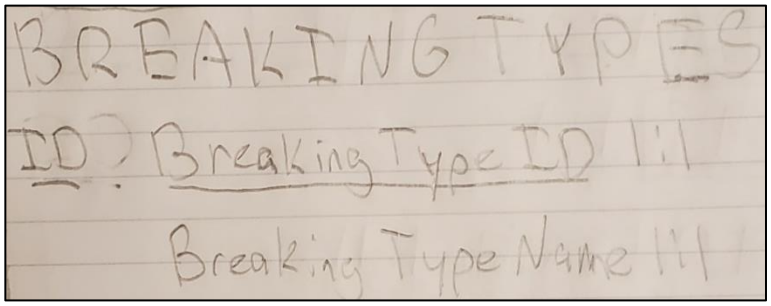
Smelting Method Name 1:1



EFFECTS

$\underline{\text{ID}}$ $\underline{\text{Effect ID}}$ 1:1

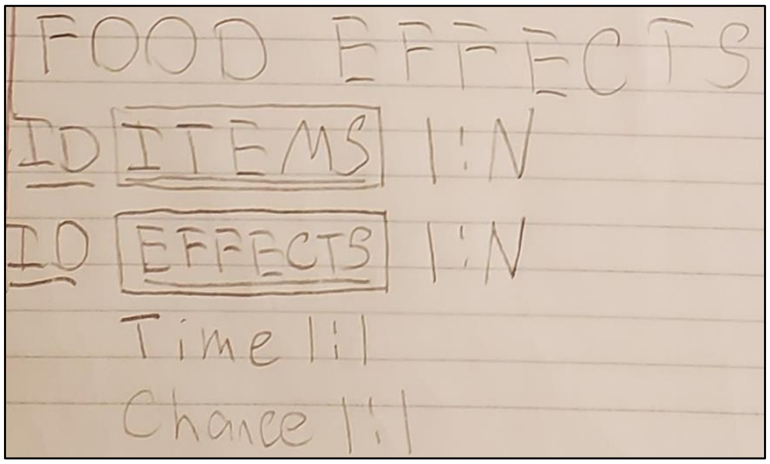
EffectName 1:1



BREAKING TYPES

$\underline{\text{ID}}$ $\underline{\text{Breaking Type ID}}$ 1:1

Breaking Type Name 1:1



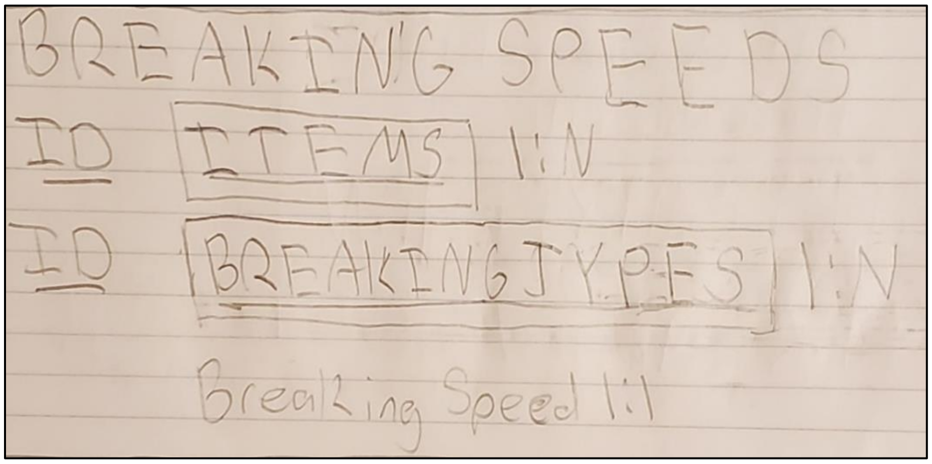
FOOD EFFECTS

$\underline{\text{ID}}$ $\boxed{\underline{\text{ITEMS}}}$ 1:N

$\underline{\text{ID}}$ $\boxed{\underline{\text{EFFECTS}}}$ 1:N

Time 1:1

Chance 1:1



BREAKING SPEEDS

$\underline{\text{ID}}$ $\boxed{\underline{\text{ITEMS}}}$ 1:N

$\underline{\text{ID}}$ $\boxed{\underline{\text{BREAKINGTYPES}}}$ 1:N

Breaking Speed 1:1

Data Dictionary

Key

PRI = Primary Key

FOR = Foreign Key

items Table

Attribute	Data Type	Null	Key	Description
item_id	varchar(100)	NO	PRI	Official game item ID used as primary key
item_name	varchar(100)	NO		The display name of the item
stackibility	smallint	NO		Amount of the item that can be stacked in one item slot
attack_speed	double	NO		Speed at which the item can attack entities
attack_damage	double	NO		Amount of attack damage per strike with item
peaceful_obtainable	bool	NO		Value if the item is obtainable at peaceful difficulty in the survival game mode or not
craftable	bool	NO		Value if the item can be crafted or not

renewable table

Attribute	Data Type	Null	Key	Description
item_id	varchar(100)	NO	PRI, FOR	Foreign key linking to items table and primary key since the tables are 1 to 1
renewable	bool	NO		Value if the item can be farmed, crafted, or traded or not

survival_obtainable Table

Attribute	Data Type	Null	Key	Description
item_id	varchar(100)	NO	PRI, FOR	Foreign key linking to items table and primary key since the tables are 1 to 1
survival_obtainable	bool	NO		Value if the item is obtainable in the survival game mode or not

smelting_obtainable Table

Attribute	Data Type	Null	Key	Description
item_id	varchar(100)	NO	PRI, FOR	Foreign key linking to items table and primary key since the tables are 1 to 1
smelting_method_id	smallint	NO	FOR	Foreign key linking to the smelting_methods table. If the item can be obtained through smelting its ID will be in this table related with how it can be smelted.

cooldown Table

Attribute	**Data Type**	**Null**	**Key**	**Description**
item_id	varchar(100)	NO	PRI, FOR	Foreign key linking to items table and primary key since the tables are 1 to 1
cooldown	double	NO		Number of seconds an item takes to cooldown after being used (applications vary)

fuel_duration Table

Attribute	**Data Type**	**Null**	**Key**	**Description**
item_id	varchar(100)	NO	PRI, FOR	Foreign key linking to items table and primary key since the tables are 1 to 1
fuel_duration	smallint	NO		How many seconds an item can be used as a fuel source in the standard furnace

food_items Table

Attribute	**Data Type**	**Null**	**Key**	**Description**
item_id	varchar(100)	NO	PRI, FOR	Foreign key linking to items table and primary key since the tables are 1 to 1
hunger	smallint	NO		Number of hunger points an item recovers after being consumed
saturation	double	NO		Number of saturation points an item recovers after being consumed (Saturation points set the amount of time before the hunger bar decreases)

smeltable_items Table

Attribute	**Data Type**	**Null**	**Key**	**Description**
item_id	varchar(100)	NO	PRI, FOR	Foreign key linking to items table and primary key since the tables are 1 to 1
smelting_xp	double	NO		Number of experience points received after smelting an item
smelting_method_id	varchar(50)	NO	FOR	Foreign key linking to the smelting_methods table. If the item can be smelted its ID will be in this table related with how it can be smelted.

item_types Table

Attribute	**Data Type**	**Null**	**Key**	**Description**
item_type_id	serial	NO	PRI	Auto incrementing primary key that identifies the item type
item_type_name	varchar(50)	NO		Name of the item type

smelting_methods Table

Attribute	Data Type	Null	Key	Description
smelting_method_id	serial	NO	PRI	Auto incrementing primary key that identifies the smelting method
smelting_method_name	varchar(50)	NO		Name of the smelting method (Smelting is cooking or obtaining refined goods by putting them in a furnace)

effects Table

Attribute	Data Type	Null	Key	Description
effect_id	serial	NO	PRI	Auto incrementing primary key that identifies the effect
effect_name	varchar(50)	NO		Name of the effect (effects in Minecraft are conditions that affect an entity in a good or bad way)

breaking_types Table

Attribute	Data Type	Null	Key	Description
breaking_type_id	serial	NO	PRI	Auto incrementing primary key that identifies the breaking type
breaking_type_name	varchar(25)	NO		Name of the breaking type (breaking type being the type of block being broken)

items_item_types Table

Attribute	Data Type	Null	Key	Description
item_id	varchar(100)	NO	PRI, FOR	Foreign key linking to items table and primary key since this table represents the many to many relationship between items and item_types
item_type_id	smallint	NO	PRI, FOR	Foreign key linking to item_types table and primary key since this table represents the many to many relationship between items and item_types

food_effects Table

Attribute	Data Type	Null	Key	Description
item_id	varchar(100)	NO	PRI, FOR	Foreign key linking to items table and primary key since this table represents the many to many relationship between items and effects
effect_id	serial	NO	PRI, FOR	Foreign key linking to effects table and primary key since this table represents the many to many relationship between items and effects
time	smallint	NO		Number of seconds the effect is applied when consumed
chance	double	NO		The chance of the effect being applied in decimal format (0.8 = 80% etc.)

breaking_speeds Table

Attribute	Data Type	Null	Key	Description
item_id	varchar(100)	NO	PRI, FOR	Foreign key linking to items table and primary key since this table represents the many to many relationship between items and breaking_types
breaking_type_id	smallint	NO	PRI, FOR	Foreign key linking to breaking_types table and primary key since this table represents the many to many relationship between items and breaking_types
breaking_speed	double	NO		Tool speed number that affects the player digging speed

Normalization Application

items Table

This table is the table that is the most like the starting dataset and because of that it really is the table that all the normalization stemmed from. To solve some many to many relationships that resulted in multiple values in one attribute therefore failing 1NF I created several lookup and association tables that work with the items table. These tables include item_types, items_item_types, effects, food_effects, breaking_types, and breaking_speeds. To get rid of dependencies that failed 3NF I made a few tables that have a 1 to 1 relationship with items. These tables include survival_obtainable and renewable. The rest of the tables incorporated were primarily focused on reducing the number of null values since there were quite a few attributes that only applied to a small number of the items.

item_types, effects, and breaking_types Tables

These tables are lookup tables that were created to help solve the many to many relationships that made the items table fail 1NF. By having these lookup tables, the many types of items, effects, and breaking types can be separated from the items table so multiple values do not have to be in one attribute failing 1NF. These tables are also simple two attribute tables that have a unique primary key that follow 1NF, so they follow 3NF on their own.

items_item_types, food_effects, and breaking_speeds Tables

These tables are association tables that were created to help solve the many to many relationships that made the items table fail 1NF. By having these association tables, the many to many relationships can be properly represented without having to force multiple values into one attribute therefore passing 1NF. These tables also either do not have any non-primary key attributes to cause dependencies or just do not have them anyways so they also follow 3NF on their own.

survival_obtainable and renewable Tables

These tables have a 1 to 1 relationship with items to prevent dependencies caused by these tables' attributes being in one table. If an item is obtainable in peaceful it is also obtainable in survival and if an item can be crafted it also is obtainable in survival. If an item is renewable it is also obtainable in survival and if an item is can be crafted it also is renewable. By separating these attributes to their own separate tables, the attributes can remain while preventing dependencies that fail 3NF.

smelting_obtainable, cooldown, fuel_duration, food_items, and smeltable_items Tables

These tables all have a 1 to 1 relationship with items not to prevent normalization failures but to reduce the number of null values in the database. Nonetheless, normalization was still considered when making these and these tables are mostly simple two attribute tables that have a unique primary key that follow 1NF, so they follow 3NF on their own.

smelting_methods Table

This table is useful as a lookup table since smelting methods are used in both the smelting_obtainable and smeltable_items. This table is a simple two attribute table that has a unique primary key that follows 1NF, so it follows 3NF on its own.