Minecraft Database

Ein Python und SQLite Projekt von Elias Cuin und Florian Strobl

Zeitraum: Sept. & Dez. 2022

Lehrer: Herr Raake

Inhaltsverzeichnis

[Beschreibung des Projektes 2](#_Toc123721638)

[ER-Modell 3](#_Toc123721639)

[Relationen-Modell 4](#_Toc123721640)

[Integritätsbedingungen 5](#_Toc123721641)

[SQLite CREATE Befehle für alle Tables 8](#_Toc123721642)

[Beispiel Testdaten 11](#_Toc123721643)

[SQLite INSERT INTO Befehle für alle Tables mit Testdaten 14](#_Toc123721644)

[Codestruktur mit allen Funktionssignaturen 15](#_Toc123721645)

[Erklärung zu der Anzeige und der Einpflege Funktion 16](#_Toc123721646)

[SQLite Aufgaben mit Lösungen 17](#_Toc123721647)

[Link zu einem Repl und Anhang mit Sourcecode 18](#_Toc123721648)

# Beschreibung des Projektes

Ein Minecraftserver besitzt eine Welt, in der viele verschiedene Spieler miteinander spielen können. Sie verändern die Blöcke in der Welt und interagieren mit den vielen Tieren im Spiel. Jeder Spieler kann sich genau aussuchen auf welchen Server, und damit verbunden in welcher Welt er spielen möchte.

(Entity, Relation, Attribut)

"Serverworlds" haben eine ID ("server id"), um sich mit ihnen verbinden zu können und einen Namen ("name") und Logo ("icon"), damit auch wir Menschen sie unterscheiden können.

In diesen "Serverworlds" spielen ("plays") verschiedene Spieler ("Player"). Darüber hinaus sind diese bevölkert ("populated by") durch verschiedene Entities ("m\_Entities"), also Dinge wie Tiere, und ist ("has") voller "Chunks", die selbst wiederum aus Blöcken ("Blocks") bestehen ("build of"). Diese "Chunks" speichern immer deren absolute Position ("chunk position"), um später zu wissen, wo sie sich in der Welt ("Serverworld") befinden.

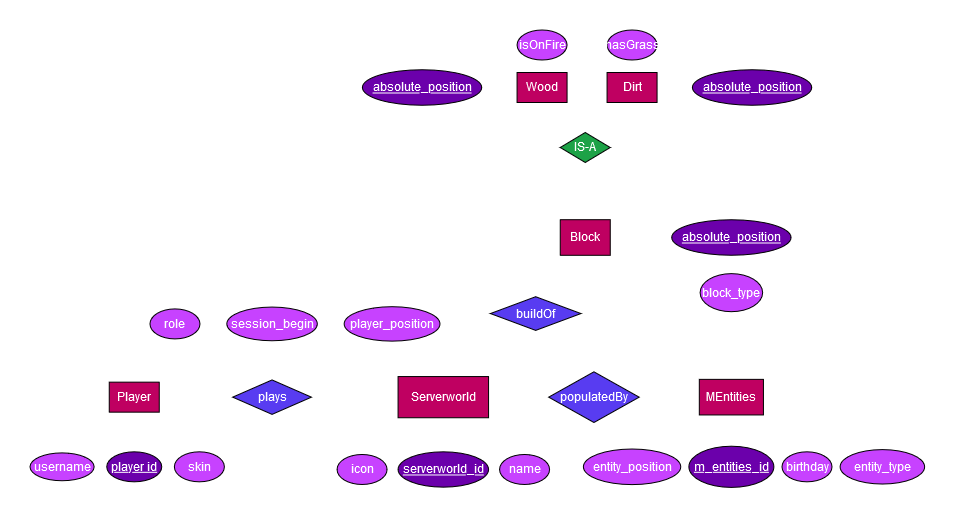
Jeder "Player" hat einen Namen ("username"), ein Avatar ("skin"), sowie eine eindeutige ID ("player id").

Das Spielen ("plays") in einer "Serverworld" muss die aktuelle absolute Position ("player position") des Spielers ("Player") speichern, wobei noch der Rang ("role") und die Uhrzeit des Spielbeginnes ("session begin") erfasst werden.

Die Enitites ("m\_Entities") der Welten haben jeweils eine eindeutige ID ("entity id"), ein Typ ("entity type"), wie beispielsweise "Kuh", eine absolute Position ("entity position") und zuletzt dessen Erstellungszeitpunkt ("birthday") gespeichert.

Die Blöcke ("Blocks") in den "Chunks" haben ihre relative Position ("relative position") gespeichert und darüber hinaus noch aus was sie bestehen ("block type"). Blöcke ("Blocks") können aus Holz ("Wood") bestehen, welche brennen können ("isOnFire") oder sie sind aus Erde ("Dirt"), welche wiederum Gras auf ihrer Oberfläche wachsen lassen können ("hasGras").

# ER-Modell



# Relationen-Modell

Serverworld(serverworld\_id, name, icon)

Player(player\_id, username, skin)

MEntities(m\_entities\_id, entity\_position, birthday, entity\_type)

Block(absolute\_position, block\_type)

Wood(^absolute\_position, isOnFire)

absolute\_position verweist auf den Primärschlüssel von Block

Dirt(^absolute\_position, hasGrass)

absolute\_position verweist auf den Primärschlüssel von Block

plays(^player\_id, ^serverworld\_id, session\_begin, player\_position, role)

player\_id verweist auf den Primärschlüssel von Player

serverworld\_id verweist auf den Primärschlüssel von Serverworld

populatedBy(^m\_entities\_id, ^serverworld\_id)

m\_entities\_id verweist auf den Primärschlüssel von MEntities

serverworld\_id verweist auf den Primärschlüssel von Serverworld

buildOf(^absolute\_position, ^serverworld\_id)

absolute\_position verweist auf den Primärschlüssel von Block

serverworld\_id verweist auf den Primärschlüssel von Serverworld

# Integritätsbedingungen

Serverworld

serverworld\_id: bigint, unique

name: string

icon: string or null

Player

player\_id: bigint, unique

username: string

skin: string

MEntities

m\_entities\_id: bigint, unique

entity\_position: string

birthday: integer

entity\_type: integer

CONSTRAINT:

0 <= entity\_type < 10

Block

absolute\_position: string, unique

block\_type: integer

Wood

^absolute\_position: string, unique reference to Block

ON DELETE cascade

ON UPDATE cascade

isOnFire: boolean, default 0

Dirt

^absolute\_position: string, unique reference to Block

ON DELETE cascade

ON UPDATE cascade

hasGrass: boolean, default 0

plays

^player\_id: bigint, unique in combination with serverworld\_id, references Player

ON DELETE cascade

ON UPDATE cascade

^serverworld\_id: bigint, unique in combination with player\_id, references Serverworld

ON DELETE cascade

ON UPDATE cascade

session\_begin: integer

player\_position: string

role: string

populatedBy

^m\_entities\_id: bigint, unique in combination with serverworld\_id, references MEntities

ON DELETE cascade

ON UPDATE cascade

^serverworld\_id: bigint, unique in combination with m\_entities\_id, references Serverworld

ON DELETE cascade

ON UPDATE cascade

buildOf

^absolute\_position: string, unique in combination with serverworld\_id, references Block

ON DELETE cascade

ON UPDATE cascade

^serverworld\_id: bigint, unique in combination with absolute\_position, references Serverworld

ON DELETE cascade

ON UPDATE cascade

*Notes:*

“boolean” steht für einen Integer der entweder 0 oder 1 sein kann.

„cascade“ steht, wie in SQLite, für “übernehme diese Aktion“.

Ein Teil der Integritätsbedingungen werden von SQLite3 übernommen, ein anderer Teil wird durch den Python code umgesetzt.

# SQLite CREATE Befehle für alle Tables

Text

Description automatically generated

CREATE TABLE IF NOT EXISTS Serverworld (

serverworld\_id bigint PRIMARY KEY,

name TEXT NOT NULL,

icon TEXT

);

CREATE TABLE IF NOT EXISTS Player (

player\_id bigint PRIMARY KEY,

username TEXT NOT NULL,

skin TEXT NOT NULL

);

CREATE TABLE IF NOT EXISTS MEntities (

m\_entities\_id bigint PRIMARY KEY,

entity\_position TEXT NOT NULL,

birthday INTEGER NOT NULL,

entity\_type INTEGER NOT NULL

);

CREATE TABLE IF NOT EXISTS Block (

absolute\_position TEXT PRIMARY KEY,

block\_type INTEGER NOT NULL

);

CREATE TABLE IF NOT EXISTS Wood (

absolute\_position TEXT NOT NULL UNIQUE,

isOnFire INTEGER DEFAULT 0,

PRIMARY KEY(absolute\_position),

FOREIGN KEY(absolute\_position) REFERENCES Block(absolute\_position)

on UPDATE cascade

on DELETE cascade

);

CREATE TABLE IF NOT EXISTS Dirt (

absolute\_position TEXT NOT NULL UNIQUE,

hasGrass INTEGER DEFAULT 0,

PRIMARY KEY(absolute\_position),

FOREIGN KEY(absolute\_position) REFERENCES Block(absolute\_position)

on UPDATE cascade

on DELETE cascade

);

CREATE TABLE IF NOT EXISTS plays (

player\_id bigint NOT NULL,

serverworld\_id bigint NOT NULL,

session\_begin INTEGER NOT NULL,

player\_position TEXT NOT NULL,

role TEXT NOT NULL,

PRIMARY KEY(player\_id, serverworld\_id),

FOREIGN KEY(player\_id) REFERENCES Player(player\_id)

on UPDATE cascade

on DELETE cascade,

FOREIGN KEY(serverworld\_id) REFERENCES Serverworld(serverworld\_id)

on UPDATE cascade

on DELETE cascade

);

CREATE TABLE IF NOT EXISTS populatedBy (

m\_entities\_id bigint NOT NULL,

serverworld\_id bigint NOT NULL,

PRIMARY KEY(m\_entities\_id, serverworld\_id),

FOREIGN KEY(m\_entities\_id) REFERENCES MEntities(m\_entities\_id)

on UPDATE cascade

on DELETE cascade,

FOREIGN KEY(serverworld\_id) REFERENCES Serverworld(serverworld\_id)

on UPDATE cascade

on DELETE cascade

);

CREATE TABLE IF NOT EXISTS buildOf (

absolute\_position TEXT NOT NULL,

serverworld\_id bigint NOT NULL,

PRIMARY KEY(absolute\_position, serverworld\_id),

FOREIGN KEY(absolute\_position) REFERENCES Block(absolute\_position)

on UPDATE cascade

on DELETE cascade,

FOREIGN KEY(serverworld\_id) REFERENCES Serverworld(serverworld\_id)

on UPDATE cascade

on DELETE cascade

);

# Beispiel Testdaten

Serverworld:

|  |  |  |
| --- | --- | --- |
| serverworld\_id | name | icon |
| 4265536020256728 | Imperial Japan | https://mc-icons/6ab4c42f87984158820dd285bba7a4e7 |
| 4335715097468013 | The Lion`s Pride | https://mc-icons/aca94a74dcaf4257882dc52bbd2b4856 |
| 1976886614033885 | Mongol Empire | https://mc-icons/814575a13d9342739481f25a188c8c5a |
| 2825036397637892 | Frosty Fortress | https://mc-icons/00e8486d5d3b42799c5c421ceb14ba64 |
| 1945457973125363 | Neo Tokyo | https://mc-icons/0c693bff8ace4b739e7d6405cad87ff0 |

Player:

|  |  |  |
| --- | --- | --- |
| player\_id | Username | skin |
| 469244117836675 | Terminator | https://mc-skin/eccc8472b01a47e095d54c78eaf15e0d |
| 2010747666924997 | Shooter | https://mc-skin/29a7cbe4a5ba44638f8935f9f01d53f2 |
| 567010368764946 | Sick Rebellious Names | https://mc-skin/dbf7f8a5be5448ed885f1024b3b2a273 |
| 2084141176415482 | Dropkick | null |
| 923845784918491 | Bowler | https://mc-skin/142c396991ba43e087c6eb0eb1432771 |

MEntities:

|  |  |  |  |
| --- | --- | --- | --- |
| m\_entities\_id | entity\_position | Birthday | entity\_type |
| 1249116255128559 | (5784561, 22617947, 4422489) | 1680627754 | 3 |
| 1477335842422153 | (4513773, 3364412, 6775063) | 1670022822 | 5 |
| 309176951353911 | (22950435, 7109287, 17234290) | 1679859647 | 3 |
| 3618547799519888 | (3763394, 8301312, 13256823) | 1666251670 | 2 |
| 1072600461443545 | (25415578, 9483589, 29479241) | 1672899056 | 7 |

Block:

|  |  |
| --- | --- |
| absolute\_position | block\_type |
| (21730538, 7257256, 12166861) | 6 |
| (29681001, 22411625, 5323664) | 5 |
| (5643271, 7661161, 21081451) | 2 |
| (2213165, 20874322, 16722080) | 1 |
| (6771785, 2416044, 20519029) | 7 |

Wood:

|  |  |
| --- | --- |
| absolute\_position | isOnFire |
| (5643271, 7661161, 21081451) | 1 |
| (6771785, 2416044, 20519029) | 1 |
| (2213165, 20874322, 16722080) | 0 |
| (21730538, 7257256, 12166861) | 1 |
| (29681001, 22411625, 5323664) | 1 |

Dirt:

|  |  |
| --- | --- |
| absolute\_position | hasGrass |
| (5643271, 7661161, 21081451) | 1 |
| (2213165, 20874322, 16722080) | 1 |
| (6771785, 2416044, 20519029) | 1 |
| (29681001, 22411625, 5323664) | 0 |
| (21730538, 7257256, 12166861) | 0 |

plays:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| player\_id | serverworld\_id | session\_begin | player\_position | role |
| 2084141176415482 | 2825036397637892 | 1666602795 | (5958751, 19326618, 13565303) | Moderator |
| 923845784918491 | 4335715097468013 | 1674296095 | (20308392, 7050927, 1940677) | Player |
| 469244117836675 | 4265536020256728 | 1682641393 | (15059814, 13646139, 28813591) | Admin |
| 469244117836675 | 4335715097468013 | 1676081651 | (28917752, 21778988, 19790312) | Moderator |
| 2010747666924997 | 4335715097468013 | 1677866115 | (16220625, 1713882, 26215383) | Admin |

populatedBy:

|  |  |
| --- | --- |
| m\_entities\_id | serverworld\_id |
| 1249116255128559 | 4335715097468013 |
| 1249116255128559 | 1976886614033885 |
| 3618547799519888 | 4335715097468013 |
| 1477335842422153 | 1976886614033885 |
| 1477335842422153 | 2825036397637892 |

buildOf:

|  |  |
| --- | --- |
| absolute\_position | serverworld\_id |
| (5643271, 7661161, 21081451) | 1945457973125363 |
| (29681001, 22411625, 5323664) | 4265536020256728 |
| (29681001, 22411625, 5323664) | 1945457973125363 |
| (21730538, 7257256, 12166861) | 4265536020256728 |
| (5643271, 7661161, 21081451) | 4265536020256728 |

# SQLite INSERT INTO Befehle für alle Tables mit Testdaten

INSERT INTO Serverworld (serverworld\_id, name, icon) VALUES (x, 'y', 'z')

# Codestruktur mit allen Funktionssignaturen

# Erklärung zu der Anzeige und der Einpflege Funktion

# SQLite Aufgaben mit Lösungen

# Link zu einem Repl und Anhang mit Sourcecode