Hausarbeit im Rahmen des Berufsbegleitenden Vorbereitungsdienstes

Stärkung der Modellierungs- und Implementierungskompetenz für relationale Datenbanken durch Entwicklung eines browserbasierten Spiels anhand vorgegebener Komponenten im Fach Informatik der elften Klasse eines Gymnasiums

Beginn der Unterrichtseinheit:	30.04.2024
Umfang der Unterrichtseinheit:	8 Stunden
Stellung der Lehrprobe in der Unterrichtseinheit:	8. Stunde

Name:	Lukas Convent	Kurs:	Stufe 11, Grundkurs Inf-11-1
Schule:	Ernst-Barlach-Gymnasium Schönberg	Fach:	Informatik
Schulleiter:	Torsten Barth	Datum:	31.05.2024
Mentor:	Torsten Barth	Uhrzeit:	5. Stunde, 11:40-12:25 Uhr
Studienleiter:	Christian Pohla	Raum:	K10
Fachleiter:	Tino Hempel		

Inhaltsverzeichnis

1 Bedingungsanalyse

(Zur Anonymisierung entfernt)

1.1 Infrastruktur

Der Unterricht findet im Computerkabinett in Raum K10 statt. Da die Unterrichtseinheit überwiegend aus Projektarbeit besteht, ist es wichtig, dass sich die S zusammensetzen können, um ungestört ihre Ideen diskutieren und am Rechner umsetzen zu können. Aufgrund der gegebenen kleinen Gruppengröße eignet sich der Raum K10 dafür.

Für die Implementierung können die S entweder zu zweit an einem Rechner arbeiten oder aber einzeln an Rechnern arbeiten und sich per IServ-System vernetzen. Als benötigte Software wird der *DB-Browser for SQLite* [4] vorinstalliert zur Verfügung gestellt sowie die *Eskuel-Suite* [5] per Webbrowser (mehr dazu später). Standardwerkzeuge wie Zeichenprogramme (siehe Differenzierung) sind ebenfalls vorhanden und die Nutzung weiterer Werkzeuge per Webbrowser (z.B. ChatGPT) ist möglich.

1.2 Vorkenntnisse

Die Unterrichtseinheit schließt unmittelbar an die Unterrichtsreihe „Grundlagen von Relationalen Datenbanken“ an. Die Vorkenntnisse der S lassen sich wie folgt aufschlüsseln.

- **Fachlich**. Die S haben das *relationale Modell* als natürliche Verallgemeinerung einer Tabellenkalkulation wie Excel kennengelernt. Sie sind in der Lage, sowohl grundlegende als auch fortgeschrittene *SQL-Abfragen* an eine solche Datenbank zu formulieren. Diese Inhalte waren Gegenstand der bereits geschriebenen Klausur, welche mit gemischten Ergebnissen abschlossen wurde.
- **Organisatorisch**. Die S haben die Kompetenzen, sich in Gruppen zu organisieren und kooperativ zu arbeiten. Das agile Vorgehen (siehe Sachanalyse) lässt sich gut in bereits erprobte und durch die S in vielen Situationen unter Beweis gestellte kooperative Prozesse integrieren.
- **Projektanforderung**. Die Projektanforderung ist die Entwicklung eines *browserbasierten Escape-Room-Spiels für relationale Datenbanken*. Obwohl ausführlich formuliert, ist die Aufgabenstellung am eindrücklichsten zu erfassen, wenn man bereits mit mindestens einem möglichen aus der Aufgabenstellung rsultierenden Spiel konfrontiert wurde. Die S haben das Spiel „SQL Island“ [7] gespielt, teilweise sogar noch zur Vorbereitung auf die Klausur. Der Spieler versetzt sich hierbei in die Lage eines auf einer Insel gestrandeten Schiffbrüchigen, der nur mit Hilfe korrekt formulierter SQL-Abfragen den Weg von der Insel findet. Das Spiel war der Anlass für die Entwicklung der vorliegenden Unterrichtseinheit. Durch die Konfrontation mit SQL Island haben die S eine gute Vorstellung davon, wie ein mögliches Produkt aussehen *könnte*.

Während die S die fachlichen Grundlagen für den Umgang mit dem relationalen Modell besitzen (und ggf. weiter verbessern), werden sie nun in dieser Unterrichtseinheit das erste Mal damit konfrontiert, ein umfangreicheres relationales Modell *selbst zu entwerfen* und es gleichzeitig zu *erproben*.

1.3 Schülergruppe und Schüler-Lehrer-Verhältnis

(Zur Anonymisierung entfernt)

Fast alle S lassen sich grundsätzlich begeistern. Da der überwiegende Teil dieser Unterrichtseinheit in Projektarbeit stattfindet, haben die S viel Freiraum mit weniger durch den L vorgegebener

Struktur als sonst üblich. Die zentrale Konsequenz ist, im Voraus dafür zu sorgen, *dass die Eigenmotivation der S hoch ist*, damit auch trotz vergleichweise wenig „externer Taktung“ Arbeitsfortschritt erzielt wird. In welchen Aspekten die konkret vorliegende Variante der Projektarbeit die Motivation der S beeinflussen soll, wird in der methodischen Analyse erläutert.

2 Sachanalyse

In der Unterrichtseinheit geht es darum, ein auf **relationalen Datenbanken** basierendes browserbasiertes **Spiel** zu modellieren und zu implementieren, unter Verwendung eines systematischen **Softwareentwicklungsprozesses**. Die Unterrichtseinheit kann entlang dreier fachlicher Aspekte analysiert werden.

2.1 Relationales Modell

Als **Datenbankmanagementsystem (DBMS)** wird ein Informatiksystem bezeichnet, welches große Datenmengen effizient 1. *auffinden*, 2. *hinzufügen*, 3. *verändern* und 4. *löschen* kann. Der zugrundeliegende Datenbestand eines solchen Systems wird als **Datenbank** bezeichnet [6].

Relationale Datenbanken strukturieren Daten mittels *Tabellen*. Anders als eine Tabellenkalkulation jedoch enthält eine relationale Datenbank in der Regel mehrere Tabellen, die inhaltlich *stark miteinander verknüpft* sein können. Das zugrunde liegende Modell kann mathematisch definiert werden und wird als **Relationales Modell** bezeichnet. Der mathematische Begriff der „Relation“ (basierend auf der Mengenlehre) ist in der Praxis synonym zum Begriff „Tabelle“. Desweiteren wird der mathematische Begriff „Attribut“ in der Praxis als „Spalte“ und der Begriff „Tupel“ als „Zeile“ bezeichnet.

2.1.1 Relationenschema

Als **Relationenschema** wird die Struktur der Tabellen bezeichnet. Bsp.:

fahrlehrer(kuerzel, vorname, nachname, telefonnr)
fahrschueler(nr, vorname, nachname, gebdatum, theorie_bestanden, anz_fahrstunden, *fl_kuerzel*)

Das Schema spezifiziert hier zwei Tabellen, nämlich die Tabelle `fahrlehrer` mit den Spalten `kuerzel`, `vorname`... sowie die Tabelle `fahrschueler` mit den Spalten `nr`, `vorname`... Manchmal werden im Schema noch Datentypen für die einzelnen Spalten angegeben, z.B. „Ganzzahl“ für die Spalte `nr`. Hier ein konkretes Beispiel für eine Datenbank, welche das Schema implementiert:

kuerzel	vorname	nachname	telefonnr
Mu	Maria	Musterfrau	987-654-3210
Mus	Max	Mustermann	123-456-7890
Pet	Peter	Petersen	555-123-4567
Schm	Anna	Schmidt	111-222-3333

nr	vorname	nachname	gebdatum	theorie_bestanden	anz_fahrstunden	fl_kuerzel
4	Marie	Hoffmann	2006-04-04	0	20	Schm
6	Marie	Hermes	2001-12-02	0	17	Mus
12	Kai	Schulz	2005-11-11	1	10	Mus
13	Lara	Hoffmann	2001-12-12	0	15	Mu
14	Jan	Koch	2004-01-01	1	12	Pet

Die Unterstreichung von `kuerzel` und `nr` bedeutet, dass es sich diesen Spalten jeweils um den Primärschlüssel der Tabelle `fahrlehrer` bzw. `fahrschueler` handelt. Ein **Primärschlüssel** ist eine Menge von Spalten, deren Werte eine Zeile eindeutig identifiziert. Im Beispiel identifiziert die Spalte `kuerzel` eindeutig alle Fahrlehrer und ist somit ein valider Primärschlüssel.

Die Kursivschreibung von `fl_kuerzel` bedeutet, dass es sich bei dieser Spalte um einen Fremdschlüssel handelt. Ein **Fremdschlüssel** ist eine Menge von Spalten, deren Werte auf den Primärschlüssel einer anderen („fremden“) Tabelle verweisen. Im Beispiel verweist die Spalte `fl_kuerzel` in der Tabelle `fahrschueler` auf den Primärschlüssel `kuerzel` der Tabelle `fahrlehrer`. Somit ist also ersichtlich, dass die Schülerin „Marie Hoffmann“ bei „Anna Schmidt“ Fahrunterricht erhält.

2.1.2 Anfragesprache SQL

Gegeben eine relationale Datenbank, stellt sich nun die Frage der Benutzerschnittstelle: Wie kann nun z.B. abgefragt werden, wie viele Schüler mit bereits bestandener Theorieprüfung von der Fahrlehrin Lara Hoffmann Fahrunterricht erhalten?

Während ein Endnutzer (z.B. der Chef der Fahrschule) vermutlich eine grafische Benutzeroberfläche mit Eingabefeld erhält, wird vom DBMS eine sehr nützliche und etablierte Schnittstelle zur Verfügung gestellt, auf dessen Basis dann z.B. solche grafische Benutzeroberflächen errichtet werden können: Mit der **Structured Query Language (SQL)** können im Wesentlichen vier Arten von Anfragen an die Datenbank gestellt werden. Eine **SQL-Anfrage** kann grob als englischer Satz im Imperativ interpretiert werden, der je nach Art der Anfrage mit einem der folgenden Verben beginnt:

1. `SELECT ...`: Zeilen *auffinden*
2. `INSERT ...`: Zeilen *hinzuzufügen*
3. `UPDATE ...`: Zeilen *verändern*
4. `DELETE ...`: Zeilen *löschen*

```
SELECT   vorname, nachname, anz_fahrstunden
FROM     fahrschueler
WHERE    theorie_bestanden = 1
ORDER BY anz_fahrstunden
```

vorname	nachname	anz_fahrstunden
Kai	Schulz	10
Jan	Koch	12

Die wesentlichen weiteren Operationen sind **Selektion** (Filtern mit `WHERE`), **Projektion** (Spalten auswählen) und **Sortierung** (Zeilen sortieren). Zudem können Tabellen zusammengeführt werden, durch die Operationen **Schnitt, Vereinigung, Differenz** (Tabellen vertikal zusammenführen) und **Kartesisches Produkt / Verbände** (Tabellen horizontal zusammenführen).

2.2 Gamification

Unter dem Begriff „Gamification“ versteht man die Nutzung eines spielerischen Ansatzes zur Motivation eines ansonsten nicht unbedingt mit „Spiel“ assoziierten Prozesses. Im Bereich der Didaktik wird Gamification auf Lernprozesse angewandt [9].

Die Ausgestaltung geschieht über verschiedene Elemente, z.B. über Narrative, über Punktesysteme, Levelsysteme, Wettbewerb („leaderboard“), Abzeichen („badges“), soziale Interaktionen und Interaktivität. Die grundsätzlich starke Ausprägung des Spieltriebs im Menschen ist offenbar. Diesen Spieltrieb nun auf zunächst scheinbar spielferne Kontexte zu lenken, bietet enorme Chancen.

Als Beispiel wird an dieser Stelle die Inspiration für die vorliegende Unterrichtseinheit genannt, nämlich das Spiel „SQL Island“ [7]. Das Genre des Spiels kann genauer als *Adventure* [10] und noch genauer als *Escape Room Game* klassifiziert werden. Ganz zentral bei einem *Adventure* ist das Narrativ aus der Ich-Perspektive. Im Fall von „SQL Island“ identifiziert sich der Spieler z.B. mit einem auf einer Insel gestrandeten Schiffbrüchigen. Ein *Escape Room Game* konkretisiert das Narrativ noch zu dem Spielziel, einer ungünstigen Situation zu entkommen, z.B. den Weg von einer Insel zu finden. Im Fall von „SQL Island“ gibt es ein Levelsystem als Fortschrittsindikator. Eine sehr interaktive Spielweise wird erreicht, indem Eingaben mit (Fehler-)Hinweisen oder Erfolgsmeldungen beantwortet werden.

2.3 Softwareentwicklungsprozess

Die Entwicklung eines Informatiksystems besteht aus verschiedenen aufeinander folgenden Phasen, hier nur verkürzt dargestellt.

1. **Anforderungsanalyse**. Im Dialog mit dem Kunden werden die Anforderungen an das System erfasst. Häufig ist die Vorstellung des Kunden in dieser Phase noch nicht klar und schlüssig, so dass in dieser Phase auch über die genaue Zielstellung beraten wird.

2. **Modellierung und Entwurf**. Ausgehend von präzise definierten Anforderungen werden die für die Datenverarbeitung *relevanten Aspekte* identifiziert und strukturiert. Bezogen auf das vorliegende Projekt beinhaltet dies den Entwurf eines Relationenschemas. Typischerweise geht dem Entwurf des Relationenschemas noch ein konzeptioneller Entwurf mittels eines abstrakteren Entity-Relationship-Modell voraus. Da dieses Modell jedoch im Unterricht noch nicht behandelt wurde, wird es hier nicht weiter betrachtet.

3. **Implementierung**. Der Entwurf wird umgesetzt und mündet in einem Software-Artefakt. Bezogen auf das vorliegende Projekt entsteht eine Spiel-Datei, die sowohl Datenbank als auch Spielszenen, Lösungen etc. enthält.

4. **Testen und Wartung**. Das Artefakt wird auf Fehler überprüft und ggf. verbessert. Bezogen auf das vorliegende Projekt bedeutet dies, dass das Spiel getestet wird und ggf. noch Anpassungen vorgenommen werden.

Die klassische Literatur geht von einem linearen Durchlaufen der vorgestellten Phasen aus, auch als **Wasserfallmodell** [1] bezeichnet. Die Praxis hat diese Theorie jedoch widerlegt, da sich oftmals gezeigt hat, dass die Phasen nicht strikt voneinander trennbar sind und dass die Anforderungen sich während der Entwicklung ändern können.

Ein praxisbewährterer Ansatz sind **agile Vorgehensmodelle**, welches in verschiedenen Ausprägungen existiert, und deren Grundideen sich in dem „Manifest für Agile Softwareentwicklung“ [2] wiederfinden. Alle agilen Vorgehensmodelle haben gemein, dass die Entwicklung in kurze Iterationen unterteilt wird, meist *sprints* genannt, in denen jeweils lauffähige Produkte entstehen. Diese Zwischenprodukte werden dann immer wieder dem Kunden präsentiert, um Feedback zu erhalten. Das Feedback wird dann in die nächste Iteration einbezogen.

3 Didaktische Analyse

3.1 Curriculare Legitimation

Die curriculare Legitimation des Unterrichtsthemas wird im Folgenden mit Bezug auf den Informatik-Rahmenplan für die Qualifkationsphase der gymnasialen Oberstufe in Mecklenburg-Vorpommern [12] aufgeschlüsselt.

3.1.1 Inhaltsbereiche

Das Thema „Relationale Datenbanksysteme“ ist mit empfohlenen 30 Stunden prominent im Rahmenplan verankert, aufgeteilt in zwei Abschnitte „Datenbanken abfragen“ und „Datenbanken entwickeln“. Die vorliegende Unterrichtseinheit erfordert den Umgang mit fast allen grundlegenden Konzepten relationaler Datenbanken, insbesondere dem relationalen Modell und der Anfragesprache SQL. Zudem werden im Entwicklungsprozess verschiedene Phasen der Datenbankentwicklung durchlaufen (*logische Phase*, *physische Phase*). Nicht einbezogen wird die *konzeptionelle Phase* mittels ER-Modell, für welche die fachlichen Grundlagen erst im weiteren Verlauf des Semesters erarbeitet werden.

Das Thema „komplexe Aufgabenstellungen informatisch lösen“ legitimiert die vorliegende Unterrichtseinheit hinsichtlich des zu durchlaufenden Entwicklungsprozesses. Dieser Entwicklungsprozess muss als komplex bezeichnet werden, da er sämtliche Phasen der Softwareentwicklung umfasst, welche dann nach agilem Vorgehensmodell durchlaufen werden.

3.1.2 Kompetenzen

Im Rahmenplan werden informatische Kompetenzen entlang von fünf Prozessbereichen erfasst. Der folgende Prozessbereich wird in der Unterrichtseinheit besonders gefördert: **Modellieren und Implementieren**.

- *Modellieren.* Anhand einer selbst gewählten Spielidee muss ein **relationales Modell** entwickelt werden, d.h., eine „Abstraktion zu einem bestimmten Zweck“ [12], nämlich zum Zweck der Datenverarbeitung mittels eines relationalen DBMS. Auf eine vorgeschaltete konzeptionelle Modellierung mit der Darstellungsform „ER-Modell“ wird verzichtet, da die Darstellungsform noch nicht behandelt wurde. Dies ist insofern trotzdem hinnehmbar, als die Komplexität des Projekts auch ohne ER-Modell beherrschbar ist.
- *Implementieren.* Die Implementierung erfolgt durch verschiedene Komponenten: Zum einen muss die Datenbank implementiert werden. Zum anderen müssen die Szenen und die dazugehörigen SQL-Abfragen formuliert werden. Mit Hilfe des SQL-Spieleeditors müssen diese Daten zusammengeführt werden und zu einer Spieldatei kompiliert werden, die mit der SQL-Spielekonsole geladen werden kann, „wodurch das Ergebnis des Modellierens erlebbar wird“ [12]. Abschließend wird das Produkt getestet und ggf. verbessert, was einer „anschließenden Reflexion“ [12] genüge tut.

3.2 Exemplarität

Ich orientiere mich am von Wolfgang Klafki beschriebenen Kriterium: „Am potentiellen Thema müssen sich allgemeinere Zusammenhänge, Beziehungen, Gesetzmäßigkeiten, Strukturen, Widersprüche, Handlungsmöglichkeiten erarbeiten lassen.“ [11]

Die Inhalte der vorliegenden Unterrichtseinheit sind exemplarisch für den Entwicklungsprozess eines *domänenspezifischen Softwareprodukts*. Softwareprodukte lassen sich grob hinsichtlich der Dimension Anpassung-vs-Eigenentwicklung ordnen, mit den folgenden Extrema:

- **Softwareanpassung**. Das Softwareprodukt besteht lediglich aus der *Konfiguration* eines bereits bestehenden Softwarekontexts, der auch als *modifiable off-the-shelf (MOTS)* bezeichnet wird. Erst die Konfiguration erschließt die Software für die angewandte Domäne. Ein Beispiel ist die *Domäne* „Schulwebsite“, für welche ein Content-Management-Systems (CMS) wie Wordpress als *Softwarekontext* konfiguriert werden kann.
- **Eigenentwicklung**. Das Softwareprodukt löst ein so spezifisches Problem, dass auf kein bereits bestehendes Softwareprodukt aufgebaut werden kann. Ein Beispiel hierfür ist die Entwicklung eines Autopiloten für ein neuartiges selbstfahrendes Fahrzeug.

Das in dieser Unterrichtseinheit entstehende Produkt fällt eher in die Kategorie „Softwareanpassung“. Der zugrundeliegende *Softwarekontext* sind der SQL-Spieleeditor und die SQL-Spielekonsole [5]. Dies ermöglicht den Fokus auf die Modellierung und Implementierung der *Domäne* (z.B. „Fischmuss-im-Ozean-nach-Hause-finden“), welche frei von den S gewählt werden kann.

Alle Entwicklungsprozesse in der Informatik lassen sich letztlich hinsichtlich dieser Dimension klassifizieren: Gegeben eine bestehende Softwarelandschaft, wird auf möglichst viele bestehende Komponenten zurückgegriffen, um die Entwicklung zu beschleunigen und nicht „das Rad neu zu erfinden“. Die *Domäne* hingegen ist in der Regel so spezifisch, dass eine auf das Problem zugeschnitte Modellierung und Implementierung unumgänglich sind.

3.3 Gegenwartsbedeutung

Sowohl der in dieser Unterrichtseinheit durchlaufene Entwicklungsprozess im Allgemeinen als auch die Modellierung und Implementierung von Datenbanksystemen im Speziellen haben eine fundamentale Gegenwartsbedeutung.

Die systematische (statt ad-hoc) Herangehensweise an die Produktentwicklung kann auf jegliche Ingenieurstätigkeit verallgemeinert werden. Ziel dieser Systematik ist es, der Komplexität des zu lösenden Problems habhaft zu werden und vermeidbare Fehlerquellen bereits im Vorhinein auszuschließen. In vielen Bereichen ist dies nicht nur aus Effizienzgründen notwendig, sondern auch zur Qualitätssicherung, wenn gar Menschenleben gefährdet sind (z.B. Autopilot oder Medizingeräte).

Die Bedeutung von Daten für unsere moderne Wissensgesellschaft ist fundamental, da sie wertvolle Informationen tragen. Durch bahnbrechende Technologien wie das Internet, moderne Sensortechnik etc. stehen uns so viele Daten wie nie zuvor zur Verfügung. Ihre Nutzbarmachung wird erst gewährleistet durch Systeme, welche die Daten in ihrer *Struktur* erfassen und abfragbar machen. Genau diese Strukturerfassung wird durch die Modellierung und Implementierung von Datenbanksystemen ermöglicht.

3.4 Zukunftsbedeutung

Es scheint unausweichlich, dass unsere Wissensgesellschaft auch in Zukunft Technologie einsetzen wird, um Daten zur Informationsbeschaffung zu erschließen.

Während die exakte zukünftige Ausprägung solcher Technologien unklar ist, so kann man das relationale Modell als eine unserer bisher einfachsten und besten Theorien für viele Datenbankanwendungen ansehen. Die Beschäftigung mit unseren bislang besten Theorien gibt erst Anlass zu neuen Problemstellungen (z.B. war die Problemstellung der Skalierbarkeit von verteilten sozialen Netzwerken vor 30 Jahren nicht einmal formuliert, geschweige denn gelöst), ermöglicht deren Erforschung und bietet dann erst die Möglichkeit, sie neuen (eventuell überlegenen) Theorien gegenüberzustellen.

Die Zukunftsbedeutung von kooperativen systematischen Softwareentwicklungsprozessen ist ebenfalls offenbar. Es ist davon auszugehen und es gilt optimistisch zu sein, dass unsere Gesellschaft auch weiterhin vor Lösungsversuchen komplexer Problemstellungen nicht zurückschreckt. Die Umsetzung jeglicher technologischer Lösungen erfordert immer die Zusammenarbeit vieler Menschen,

und ein solcher gemeinsamer Prozess muss immer wieder neu erprobt, gelernt und verbessert werden.

3.5 Reduktion

Um den S die Inhalte der vorliegenden Stunde zugänglich zu machen, wurde das von den S zu erarbeitende Sachgebiet massiv reduziert. Ich gehe im Folgenden nicht nur auf die Reduktion der Sachgebiete, sondern auch auf die Reduktion innerhalb der durchgeführten Methodik ein.

- **Einschränkung des Spielkonzepts**. Ein auf relationalen Datenbanken basierendes Escape-Room-Spiel kann sehr verschieden ausgestaltet werden, und der Raum für Ideen ist gar nicht erfassbar. An dieser Stelle wurde die Aufgabenstellung *stark eingeschränkt*, und zwar so, dass das entstehende Spiel im Wesentlichen die gleichen strukturellen Eigenschaften wie das SQL Island-Spiel [7] haben soll.

  Diese Eigenschaften umfassen z.B. eine Spielführung, die *linear* ist (Aneinanderreihung von Inhaltsszenen) und *deterministisch* (Endspielzustand der Datenbank ist vorgegeben und nicht vom Spieler beeinflussbar). Desweiteren sind für den Kern der Spiellogik, nämlich das „Lösen“ von Inhaltsszenen, nur zwei Szenenvarianten möglich: 1. Eingabe einer `SELECT`-Abfrage; 2. Eingabe einer Manipulations-Abfrage.

- **Softwareanpassung statt Eigenentwicklung**. Wie bereits in Abschnitt 3.3 besprochen, ist das entstehende Softwareprodukt eher eine Softwareanpassung als eine Eigenentwicklung. Diese Bezeichnung soll keineswegs die zu erwartende Schöpfungshöhe klein reden. Im Gegenteil:

  Eine Programmierung der Spiellogik wird von den S nicht gefordert, da sie vom Thema der Unterrichtseinheit nicht nur ablenken, sonder erhebliche (!) Entwicklungskapazitäten binden würde. Ohne die Infrastruktur der SQL-Spielekonsole wäre die vorliegende Unterrichtseinheit unmöglich durchführbar in einem Grundkurs. Die vorhandenen Kapazitäten können also voll auf das Ziel der Unterrichtseinheit fokussiert werden: Die Modellierung und Implementierung der Datenbank und der Inhaltsszenen.

- **Erzeugung der Spieldatei**. Das entstehende Produkt ist eine XML-Datei, welche sowohl die Datenbank, als auch die Inhaltsszenen enthält. Das Dateiformat XML wurde gewählt, da es...

  1. *manuell* von den S mit dem Texteditor *geschrieben* und *gelesen* werden kann.
  2. *maschinell* von der SQL-Spielekonsole *gelesen* werden kann.
  3. *maschinell* vom SQL-Spieleeditor *geschrieben* werden kann.

  Möglichkeit 2 stellt das Szenario für den Endkunden dar: Der Spieler lädt die Datei in die SQL-Spielekonsole und spielt. Möglichkeit 3 wurde in den ersten Überlegungen zu dieser Unterrichtseinheit noch nicht in Betracht gezogen. Der ursprüngliche Plan war, dass die S das Spiel mit dem Texteditor in XML *manuell* eintippen (Möglichkeit 1). In einem professionellen Entwicklerkontext würden Softwareentwickler/innen aus Effizienzgründen auch genau dies für einen ersten Prototyp machen. Die Reduktion, die an dieser Stelle vorgenommen wurde, hat eine *maßgebliche Auswirkung* auf die gesamte Unterrichtseinheit:

  Während die Beschäftigung mit dem Dateiformat XML *für sich genommen* interessant ist und perfekt curricular legitimiert werden könnte (Inhaltsbereich „Formale Sprachen“), so lenkt sie doch an dieser Stelle vom Unterrichtsgegenstand der relationalen Datenbanken ab. Die Implementierung des Spiels soll so *unmittelbar* wie möglich geschehen, was am besten über eine maßgeschneiderte Benutzerschnittstelle erfolgt, welche ich den S mit dem SQL-Spieleeditor bereitstellen kann [5].

3.6 Lernziele

3.6.1 Unterrichtseinheit

Grobziel: Die S sind in der Lage, ein auf *relationalen Datenbanken* aufbauendes browserbasiertes Spiel zu **modellieren** und zu **implementieren**, unter Verwendung eines **systematischen Softwareentwicklungsprozesses**.

Feinziele:

1. Die S können ein auf relationalen Datenbanken basierendes Softwareprodukt **modellieren**, mündend in einem *Relationenschema* sowie sowie einer *Abfolge von Inhaltsszenen für ein Escape-Room-Spiel*.
2. Die S können ein auf relationalen Datenbanken basierendes Softwareprodukt **implementieren**, inklusive seiner **Integration** in einen bestehenden Softwarekontext (SQL-Spielekonsole) und seiner **Testung**.
3. Die S können einen **Softwareentwicklungsprozess** systematisch und ggf. kooperativ durchführen, unter Anwendung eines *agilen Vorgehensmodells*.

3.6.2 Stunde 1: Entwicklung eines groben Spielkonzepts

Grobziel: Die S kennen die Anforderungen an ein browserbasiertes und auf dem relationalen Modell basierendes Spiel, können die SQL-Spielekonsole und den SQL-Spieleeditor bedienen und können ein grobes Konzept für eine dazugehörige Spielwelt inklusive Handlungsstrang entwerfen.

Feinziele:

1. Die S können das Konzept „Gamification“ konzeptionell auf relationale Datenbanken und die Abfragesprache SQL anwenden und Anforderungen an ein solches Spiel diskutieren.
2. Die S kennen die technischen Möglichkeiten, die Spiele als *Spieldateien* in der SQL-Spielekonsole zu *spielen* und im SQL-Spieleeditor zu *editieren*.
3. Die S können eine eigene grobe Spielwelt mit dazugehörigem Handlungsstrang gemäß der zuvor erschlossenen Anforderungen entwickeln.

3.6.3 Stunden 2+3: Modellierung und Implementierung einer Datenbank

Grobziel: Die S können eine grob konzipierte Spielwelt auf ihre benötigten Datenverarbeitungsaspekte hin analysieren, diese Aspekte als Relationenschema modellieren, und mit zum Handlungsstrang passenden Beispieldaten in einer Datenbank implementieren.

Feinziele:

1. Die S können eine grob konzipierte Spielwelt analysieren und diese mit Tabellen und Spalten in einem Relationenschema modellieren.
2. Die S können Ideen für einen Handlungsstrang formulieren, diesen in Beziehung zu dem sich entwickelnden Relationenschema setzen und geeignete Beispieldaten generieren.
3. Die S können das Relationenschema als SQLite-Datenbank implementieren und diese mit Beispieldaten füllen.

3.6.4 Stunden 4+5: Modellierung eines Handlungsstrangs mit SQL-Abfragen

Grobziel: Die S können einen durch eine Spielwelt führenden Handlungsstrang als Inhaltsszenen mit Aufgaben modellieren, die mit zum Relationenschema passenden SQL-Abfragen lösbar sind, sowie die SQL-Abfragen in einer bereits implementierten Datenbank testen.

Feinziele:

1. Die S können einen gegebenen Handlungsstrang in Inhaltsszenen unterteilen und hinsichtlich der für den Spielfortschritt relevanten Informationsbedürfnisse analysieren.
2. Die S können die erschlossenen Informationsbedürfnisse präzise als sich auf das Relationenschema beziehende SQL-Abfragen formulieren.
3. Die S können formulierte SQL-Abfragen im SQL-Browser auf einer zuvor implementierten Datenbank testen.

3.6.5 Stunden 6+7: Implementierung eines Spiels

Grobziel: Die S können verschiedene zu einem Spiel gehörige Komponenten mit dem SQL-Editor zu einer Spieldatei kompilieren, das entstandene Spiel im Gesamtzusammenhang mit der SQL-Spielekonsole testen und gegebenenfalls einzelne Komponenten verbessern.

Feinziele:

1. Die S sind vertraut mit allen technischen Möglichkeiten, welche die SQL-Spielekonsole und der SQL-Spieleeditor bieten, und können diese nutzen, um aus gegebenen Spielkomponenten eine Spieldatei zu kompilieren.
2. Die S können ein selbst erstelltes Softwareprodukt im Gesamtzusammenhang testen und mit ihren Erwartungen abgleichen.
3. Die S können Probleme im entstandenen Produkt identifizieren und gegebenenfalls beheben.

3.6.6 Stunde 8: Präsentation und Bewertung eines Spiels

Grobziel: Die S können ein selbst entwickeltes Spielprodukt präsentieren und Einblick in den Entwicklungsprozess geben, fremde Projekte hinsichtlich gegebener Projektanforderungen testen und bewerten sowie gegenseitige Rückmeldungen formulieren.

Feinziele:

1. Die S können ein selbst entwickeltes Spielprodukt präsentieren und Herausforderungen und Erlebnisse während des Entwicklungsprozesses reflektieren.
2. Die S können ein fremdes Spielprodukt hinsichtlich gegebener Projektanforderungen testen und bewerten.
3. Die S können gegenseitige Rückmeldungen auf einem dazu vorbereiten Feedbackbogen formulieren, inklusive eventueller Verbesserungsvorschläge.

3.7 Struktur der Unterrichtseinheit

Die Unterrichtseinheit enthält einen sehr hohen Anteil an Gruppenarbeit mit viel eigenständiger Arbeit. Das hohe Maß an eigenständigem Arbeiten ist notwendig, da innerhalb der Gruppen die Entwicklungsphasen durch den agilen Ansatz ineinander verwoben sind und ein synchrones Vorgehen erschweren. Jede Stunde bzw. Doppelstunde ist immer mit einem an die Projekte gekoppelten Ziel versehen, so dass der Entwicklungsfortschritt von den Gruppen grob selbst und auch durch den L eingeschätzt werden kann.

Die Struktur der Unterrichtseinheit orientiert sich ganz grob an dem in der Sachanalyse erläuterten Softwareentwicklungsprozess. Während in professionellen Entwicklungskontexten auf jede der fünf genannten Phasen ganze Wochen verwendet werden und eine Trennung der Phasen (auch bei agiler Vorgehensweise) expliziter hervortritt, scheint eine zu strike Trennung der Phasen für die vorliegende Unterrichtseinheit künstlich und unpraktikabel. Diese Unpraktikabilität rührt einmal daher, dass die verschiedenen Gruppen nicht komplett synchron arbeiten. Desweiteren ist die Gruppengröße und die Projektkomplexität so gewählt, dass schnell zwischen den Phasen gewechselt werden kann. In der Konsequenz lässt sich jeder kleinste Projektfortschritt zwar eindeutig einer Phase

zuordnen, es wird jedoch auf eine feste Taktung verzichtet (z.B. „in dieser Doppelstunde führt ihr bitte *unbedingt nur* die Implementierungsphase durch“).

Stattdessen wählen die S mitunter selbst, wann sie zwischen Phasen übergehen. Notwendigerweise wird den S viel Zeit in eigenständiger Verantwortung übertragen. Der L nimmt hierbei eher die Rolle des *Problemhelfers* statt des *Anleiters* an. Dies macht es um so notwendiger, den selbstständigen Arbeitsphasen einen Rahmen zu geben, um Orientierung zu geben. In jeder Einzelstunde bzw. Doppelstunde wird deshalb immer

- ...zu Beginn ein konkretes Ziel für die S formuliert. Es kann dabei passieren, dass eine Gruppe am Ende das Ziel nicht erreicht oder aber über das Ziel hinausschießt. Beides ist erwartbar und dies wird auch so von Lehrerseite signalisiert. Der Zweck ist vornehmlich, Bewusstsein bei den S dafür zu schaffen, wo im Entwicklungsprozess sie sich befinden und ob eventuell Rückstände aufgeholt werden müssen.

- ...zum Ende hin ein kurzes Feedback gegeben, manchmal im Plenum, meistens aber individuell von Lehrerseite zur Gruppe hin. Der L lässt sich den Projektfortschritt zeigen, stellt Fragen zum weiteren Vorgehen und gibt Hinweise, wo er Probleme sieht. Bei vielen Modellierungsentscheidungen gibt es mehere gute Lösungen, fast immer auch eine *im Gesamtkontext betrachtet* beste. Der L tritt an dieser Stelle vollkommen in beratender Funktion auf und es ist auch in Ordnung, wenn die S sich nicht für die aus Lehrersicht beste Lösung (sondern die z.B. nur zweitbeste) entscheiden. Die S entscheiden selbst, wie sie mit den Hinweise zu ihrem Produkt umgehen.

Auch über das abschließende Feedback hinaus tritt der L ständig in beratender Funktion auf. Dies umfasst auch die technische Unterstützung, z.B. bei der Bedienung des SQL-Spieleeditors oder der SQL-Spielekonsole, sowie die Erklärung von fachlichen Inhalten. S mit Wissenslücken erhalten hier die Chance, bei dieser ganz konkreten Anwendung des geforderten Wissens durch eine Erklärung neue Einsichten zu erhalten. Es darf erwartet werden, dass nicht nur die Modellierungs- und Implementierungskompetenz der S gestärkt wird, sondern auch eventuelle fachliche Wissenslücken bei einzelnen S geschlossen werden.

3.7.1 Einordnung der Lehrprobenstunde

Stunde	Thema	Grobziel
1. 30.04.2024	Entwicklung eines groben Spielkonzepts	Die S kennen die Anforderungen an ein browserbasiertes und auf dem relationalen Modell basierendes Spiel, können die SQL-Spielekonsole und den SQL-Spieleeditor bedienen und können ein grobes Konzept für eine dazugehörige Spielwelt inklusive Handlungsstrang entwerfen.
2. + 3. 03.05.2024	Modellierung und Implementierung einer Datenbank	Die S können eine grob konzipierte Spielwelt auf ihre benötigten Datenverarbeitungsaspekte hin analysieren, diese Aspekte als Relationenschema modellieren, und mit zum Handlungsstrang passenden Beispieldaten in einer Datenbank implementieren.
4. + 5. 14.05.2024	Modellierung eines Handlungsstrangs mit SQL-Abfragen	Die S können einen durch eine Spielwelt führenden Handlungsstrang als Inhaltsszenen mit Aufgaben modellieren, die mit zum Relationenschema passenden SQL-Abfragen lösbar sind, sowie die SQL-Abfragen in einer bereits implementierten Datenbank testen.
6. + 7. 24.05.2024	Implementierung eines Spiels	Die S können verschiedene zu einem Spiel gehörige Komponenten mit dem SQL-Editor zu einer Spieldatei kompilieren, das entstandene Spiel im Gesamtzusammenhang mit der SQL-Spielekonsole testen und gegebenenfalls einzelne Komponenten verbessern.
28.05.2024	Abgabeschluss	Die S kompilieren das gesamte Produkt zu einer einzelnen Spieldatei und reichen es per IServ ein.
8. 31.05.2024	Präsentation und Bewertung eines Spiels	Die S können ein selbst entwickeltes Spielprodukt präsentieren und Einblick in den Entwicklungsprozess geben, fremde Projekte hinsichtlich gegebener Projektanforderungen testen und bewerten sowie gegenseitige Rückmeldungen formulieren.

4 Methodische Analyse

Die zentrale Methode der Unterrichtseinheit ist die (ggf. kooperative) Projektarbeit, welche auf die vorliegende Problemstellung zugeschnitten ist. Ich erläutere zunächst die konkrete Ausgestaltung dieser Projektarbeit und legitimiere die Methode anschließend hinsichtlich ihrer Passung zu den Lernzielen.

4.1 Projektarbeit: Ausgestaltung

Die S durchlaufen einen Softwareentwicklungsprozess als Gruppe von maximal vier Personen, mit Ausnahme auch alleine (mehr dazu weiter unten). Alle Arbeit zielt auf ein in hohem Maß selbst gewähltes und gestaltetes Endprodukt ab.

Das Endprodukt besteht aus *einer einzigen Datei*, die alle Arbeitsergebnisse umfasst. Diese Datei wird als *Spieldatei* bezeichnet und enthält den *Spielnamen*, einen *Teaser* (eine Art Klappentext), die *Datenbank* bestehend aus *Relationenschema* und *Beispieldaten* sowie die *Inhaltsszenen inklusive SQL-Abfragen*.

Die Entscheidung für eine Bündelung aller Arbeitsergebnisse in einer Datei ist bewusst so getroffen worden, da eine einzelne Datei die wohl handhabbarste und einfachste Form ist, um Daten zu speichern und zu übertragen. Zudem ist die Versionierung besonders einfach. Während des Entwicklungsprozesses entstehen immer weiter fortschreitende Versionen des Spiels. Eine Datei eignet sich besonders gut zur Versionierung. Auf mächtigere Versionierungssysteme wie sie im professionellen Entwicklungsprozess unabdingbar sind, kann verzichtet werden. Stattdessen kann eine Gruppe z.B. zu Beginn der nächsten Stunde die aktuelle Version kopieren, um den alten Entwicklungsstand nicht zu verlieren und im Zweifel die Entwicklung stellenweise zu revidieren.
Die Organisation und Ausgestaltung der Projektarbeit innerhalb einer Gruppe beinhaltet viele Aspekte.

- Interne Arbeitsteilung
- Arbeitsweise: Alleine an mehreren Rechnern oder zusammen an einem Rechner?
- Kommunikation innerhalb der Gruppe per E-Mail, IServ-Gruppenordner, Whatsapp, ...
- Arbeit zusätzlich zuhause oder nicht?
- Verwendung von Werkzeugen wie einem privaten ChatGPT-Account etc.
- Erfassung von Zwischenergebnissen handschriftlich oder digital?

Das Ausprobieren dieser verschiedenen Möglichkeiten darf frei gewählt und ausprobiert werden. Es werden vom L keine Vorgaben bezüglich dieser Aspekte gemacht, sondern nur Empfehlungen gegeben. Vielmehr wird zwischendurch und auch zum Schluss die Möglichkeit gegeben, gewählte Organisations- und Arbeitstechniken zu reflektieren und Erfahrungsberichte auszutauschen.

4.2 Projektarbeit: Legitimation

Die Passung der konkreten Ausgestaltung dieser Methode zeigt sich hinsichtlich der im Folgenden erläuterten Aspekte. Die ersten beiden Punkte schätze ich aufgrund ihres Motivationspotentials als besonders zielführend ein.

- **Kreativprodukt.** Das Produkt ist zwar von seiner Form her relativ eingeengt, von seinem Thema her jedoch unbeschränkt. Denkbar sind sowohl klassische Spielwelten wie „Schuldaten-müssen-organisiert-werden“ als auch fantastische wie „Fisch-muss-im-Ozean-nach-Hause-finden“. Die Verwirklichung eigener Ideen soll Anlass zu Motivation geben.
- **Sinnvolles Endprodukt.** Das Produkt kann nach Abschluss der Unterrichtseinheit sinnvoll verwendet werden. Die Erfahrung hat gezeigt, dass das Spiel SQL Island sehr gut angenommen wurde, was ein Potential für Konkurrenzprodukte aufzeigt. Die Spieldatei kann einen

Markt finden z.B. unter den S des nachfolgenden Informatik-Jahrgangs zur spielerischen Klausurvorbereitung.

Der zunächst naheliegende, zeitsparendere und vor allem aber weit weniger effektvolle Ansatz wäre gewesen, das Spiel bloß auf konzeptioneller Ebene entwickeln zu lassen. Durch die gegebene Infrastruktur (SQL-Spielekonsole und SQL-Spieleeditor) ist die Implementierung jedoch ohne Zeitverlust für das vorliegende zu entwickelnde Projekt möglich. Eine Projektarbeit mit spielbarem Endprodukt und potentiellem Absatzmarkt darf als besonders schlüssig und sinnhaftig wahrgenommen werden.

- **Technologieoffenheit**. Die genutzte Methode lädt dazu ein, verschiedene Technologien einzusetzen. Während das Projektziel vorgegeben ist, sind die Mittel zur Zielerreichung frei wählbar. Ich verspreche mir folgende Vorteile.
  - **Natürliche Arbeitsweise.** Es entspricht unserer menschlichen Natur, die besten Werkzeuge zur Lösung unserer Probleme verwenden zu wollen. Technikverbote werden im Schulalltag zwar manchmal als nötig empfunden, z.B. das Verbot eines Taschenrechners zu Prüfungszwecken. Wenn solch ein Verbot aber nicht gerade als spielerisch wahrgenommen wird (z.B. soll es Menschen geben, denen Kopfrechnen Spaß bereitet), so widerspricht ein Technikverbot *grundsätzlich* unserer Natur und sollte deshalb nur mit Bedacht eingesetzt werden.
  - **Bessere Produkte.** Die besten Produkte entstehen unter Verwendung der besten Werkzeuge. Dass ein mit ChatGPT geschriebener Text schlechter sein *kann* als ohne (und es häufig ist!), zeigt nur, dass der Umgang mit dem Werkzeug als eigene Kompetenz verstanden werden muss. Die Hinzunahme mächtiger Werkzeuge kann per se ein Produkt nicht schlechter machen; ihre falsche Benutzung natürlich schon. Während eine Technologie wie ChatGPT unbedingt auch in seiner grundsätzlichen Funktionsweise im Informatikunterricht thematisiert werden sollte (ein Werkzeug kann erst evaluiert werden, wenn man es auf einer hinreichend tiefen Ebene verstanden hat), so muss vor allem aber auch der Umgang mit ihr erprobt und erlernt werden.

  Natürlich wird der Umgang mit ChatGPT von den S bereits auch ohne die Schule (oder aber sogar: indirekt durch in der Schule gestellte Aufgaben) erlernt, so wie alle Menschen mit Internetzugang dies gerade lernen. Ich habe selbst feststellen dürfen, dass ChatGPT für erstaunlich spezielle Domänen, z.B. Programmierung, eine **enorme** (!) Hilfe darstellt, und man muss davon ausgehen, dass das Potential dieser Werkzeugklasse – sogennannter *Large Language Models (LLMs)* – noch lange nicht erschlossen ist. Während ChatGPT zunächst als universelles mächtiges Werkzeug erkannt ist, so ist es noch unzureichend auf unsere verschiedenen Problemwelten angewandt. Die Anwendung und Erforschung von ChatGPT *auf spezifische Domänen* entfaltet erst ihr Potential. Im Rahmen dieser Unterrichtseinheit kann ChatGPT z.B. für die Erstellung von Relationenschemata, zur Generierung von Beispieldaten, zur Analyse von Fehlern in SQL-Abfragen und vielem mehr genutzt werden. Das Potential ist noch gar nicht richtig erfasst, und diese Unterrichtseinheit fordert die S dazu auf – mit nur begrenzten Tipps von Lehrerseite, der den S nur wenige Schritte voraus ist – das Potential so weit wie möglich zu erforschen und zu nutzen.

- **Realbezug**. Die genutzte Methode orientiert sich an der *berufsmäßigen Softwareentwicklung* nach einer agilen Vorgehensweise. Ein Softwareprodukt wird in der Regel kooperativ entwickelt, unterteilt in sogenannte *sprints*, regelmäßig getestet und dem Kunden (hier: Lehrer und Mitschüler) vorgestellt. Die Lebenswirklichkeit der professionellen projektgetriebenen Arbeitswelt an den mitunter etwas abgekapselten „Kosmos Schule“ anzudocken, kann grundsätzlich andere (Zukunfts-)Perspektiven eröffnen.

- **Spezialisierung**. Die genutzte Methode lädt zu *Arbeitsteilung* und daraus resultierender *Spezialisierung* ein. Dies ist zunächst begrüßenswert, da der Fokus auf selbstverantwortete und vor allem selbstgewählte (!) Teilziele das Potential zu besonderer Identifikation mit dem eigenen Wirken bergen. Zum Beispiel könnte es sich ergeben, dass sich ein S auf das Schreiben

kreativer Szenentexte fokussiert, während zeitgleich ein anderer S die Implementierung der Datenbank im SQLite-Browser übernimmt. Das fokussierte und selbst als sinnvoll erachtete Arbeiten abseits der *gewöhnlichen* Arbeitsweise bietet zudem eine Chance für kreative und *außergewöhnliche* Leistungen.

Die Gefahr, dass einzelne Gruppenmitglieder darüber den Blick auf das Projekt im Großen verlieren, schätze ich als gering ein und deshalb in der Abwägung als hinnehmbaren Nachteil: Alle Spielkomponenten sind so sehr miteinander verwoben, dass der Gesamtüberblick sich als dauerhaft notwendig herausstellen wird. Z.B. müssen die Inhaltsszenen passgenau zum relationalen Modell formuliert werden, weshalb der losgelöste Blick auf nur eine einzelne Komponente nicht ausreichen wird.

- **Präsentation und Publikation**. Die genutzte Methode schließt mit einer Präsentation in der letzten Unterrichtsstunde ab. Ein nutzbares Produkt muss immer auch vermarktet werden. Die Präsentation bietet die Möglichkeit, ein real existierendes eigenes Produkt anzupreisen. Der Aspekt der Vermarktung bietet eine weitere Möglichkeit der Reflektion: Es könnte z.B. passieren, dass erst die Konfrontation mit dem Markt bewusst macht, dass man an der Marktnachfrage vorbei entwickelt hat. Je nach sich entwickelnder Dynamik könnte sich sogar ein natürlicher Wettbewerb um das beste Produkt ergeben.

  Auch eine Alternative zu kommerzieller Innovation könnte sich aufzeigen. Für digitale Inhalte wie z.B. Software gibt es eine sehr besondere Form der Vermarktung, die unter „freie Software“ bekannt ist. Mit „frei“ ist hierbei gemeint, dass Einsicht, Verwendung und sogar Kommerzialisierung einer Software ohne Lizenzgebühren möglich ist. Das berühmteste Beispiel hierfür ist das Linux-Betriebssystem, welches auch die Grundlage jedes Android-Smartphones bildet. In der Bildungswelt werden solche als frei lizenzierte Lerninhalte als *open educational resources* (OER) [3] bezeichnet.

  Als Entwickler der SQL-Spielekonsole [5], welche selbst freie Software ist, werde ich die S auf die Möglichkeit hinweisen, ihre Spieldatei unter einer freien Lizenz zu veröffentlichen. Dies könnte z.B. dazu führen, dass die Spieldatei von anderen Schulen genutzt wird, um deren eigenen Informatikunterricht zu bereichern. Auch eine Übersetzung in andere Sprachen ist möglich, was das Produkt sogar einem potentiell weltweiten Markt eröffnet.

4.3 Sozialform

Die vorherrschende Sozialform ist Gruppenarbeit. Die Zuordnung zu Gruppen von maximal vier Personen in der ersten Stunde erfolgt auf freiwilliger Basis, um möglichst gut funktionierende Kommunikation zu ermöglichen. Für den Fall, dass S alleine arbeiten möchten, reagiere ich zunächst mit der Ermunterung, sie doch in eine Gruppe zu integrieren. Wer auf Einzelarbeit beharrt, und auch glaubhaft versichern kann, eine passende Projektidee zu haben, darf dies tun.

Ich habe mich für diesen Zuordnungsmechanismus entschieden, um die Eigenmotivation möglichst hoch zu halten. Es ist bekannt, dass viele außergewöhnliche Leistungen in Einzelarbeit entstehen können (z.B. in der Literatur) und es ist anzunehmen, dass eine Pflichtzuordnung zu Gruppen häufig sogar das Potential haben, Werke zu verhindern (z.B. würden sich vermutlich die meisten Schriftsteller dagegen sperren). Frei zusammengestellte Arbeitsgruppen können hingegen – bei gegebener Motivation und Zielanstrengung – sehr effektiv zusammenarbeiten.

4.4 Differenzierung

Aufgrund der heterogenen Leistungsstände der S müssen Problemstellungen auf verschiedenen Niveaus angeboten werden. Im Folgenden werden drei Aspekte erläutert, welche dies leisten, wobei der erste die aus meiner Sicht tragende und maßgebliche Säule der Differenzierung ist.

- **Natürliche Differenzierung**. Die Wahl der Spielidee und die Ausgestaltung der Spielwelt sind frei. Die S können sich also ein Projekt aussuchen, welches ihrem Leistungsstand entspricht. Ein einfaches Spiel könnte z.B. nur zwei Tabellen und wenige Inhaltsszenen enthalten. Durch die agile Vorgehensweise muss die Planung nicht zwingend zu Projektbeginn feststehen. Dies kann somit auch eine Dynamik ermöglichen, bei der sich eine Gruppe z.B. in der vorletzten Stunde dazu entscheiden kann, ihre Inhaltsszenen noch komplexer zu gestalten als sie es sich ursprünglich zugetraut hat. Auch dieses mögliche (spontane) Umentscheiden ist Ausdruck natürlicher Differenzierung.

- **Erforschung des XML-Formats der Spieldateien**. Wie in Abschnitt 3.5 erläutert, wurde das ursprünglich vorgesehene manuelle Eintippen der Spieldatei in XML durch eine benutzerfreundliche Eingabe mittels SQL-Spieleeditor ersetzt. Die nun vorliegende Situation bietet eine gute Möglichkeit, die S mit folgendem in der Informatik universellen Phänomen zu konfrontieren: Daten (hier: Spieldateien im XML-Dateiformat) können verschieden *interpretiert* werden. Sie können hier z.B. als...

  1. linearer **Text** (darstellbar z.B. per Texteditor)
  2. beschrifteter **Baum** (darstellbar z.B. per Webbrowser)
  3. dynamisches **Spiel** (ausführbar per SQL-Spielekonsole)
  4. statische **Spielvorlage** (darstellbar über SQL-Spieleeditor)

  ...interpretiert werden. Ein fortgeschrittener Arbeitsauftrag an leistungsstarke S könnte z.B. sein: „Öffne die XML-Spieldatei doch mal mit einem Texteditor. Vergleiche den Text mit dem in der SQL-Spielekonsole geladenen Spiel. Inwiefern handelt es sich um zwei Sichtweisen auf letztlich dasselbe?“

- **Erweiterung um Bild-Szenen**. Das zu entstehende Spiel ist zunächst als sogenanntes *Text-Adventure* konzipiert. Die neueste Version der SQL-Spielekonsole bietet nun aber auch die Möglichkeit, eine neue Variante von Inhaltsszenen zu verwenden, nämlich die Bild-Szene. Diese Erweiterung ist noch nicht im SQL-Spieleeditor integriert (aus Zeitgründen noch nicht fertiggestellt). Fortgeschrittene S – und dabei insbesondere solche, die vor technischen Details nicht zurückschrecken oder sich sogar für solche interessieren – haben die Möglichkeit, Bild-Szenen einzufügen mittels manueller Bearbeitung der XML-Spieldatei im Texteditor. Diese fortgeschrittene Aufgabenstellung kann als sehr motivierend für die S erwartet werden, da die Form des Spiels grundlegend auf eine neue Ebene gehoben wird und der Gestaltungsspielraum sowohl durch manuell erstellte aber auch durch KI-generierte Bilder stark erweitert.

Verlaufsplanung: Stunde 1

Zeit	Verlaufsform	Lehrertätigkeit	Schülertätigkeit	Sozialform
12:25 *(3')*	Lehrerimpuls, 1	• „Heute startet ihr ein 8-stündiges Projekt. Worum geht es? Ihr werdet ein Konkurrenzprodukt zu *SQL Island* konzipieren und implementieren.“ • „Ich stelle euch dazu die sogenannte SQL-Spielekonsole zur Verfügung. Man kann mit ihr eine Spieledatei laden, ganz ähnlich, wie ein Gameboy ein Spiel laden kann.“ • „Wie soll das funktionieren? Ladet bitte jetzt die Adresse https://eskuel.de in eurem Webbrowser und wählt *SQL-Spielekonsole* aus. • „Klickt nun auf *Laden* und ladet das Spiel *island*. Wie ihr seht, gibt es auch noch weitere Spiele. Probiert euch einfach mal durch. Ihr werdet sehen, dass das *island*-Spiel genau dem orginalen *SQL Island*-Spiel entspricht, welches ihr bereits kennt. Der Unterschied ist nur, dass es jetzt über die SQL-Spielekonsole ausgeführt wird. Ich lasse euch jetzt 5 Minuten spielen und helfe bei der Bedienung, sofern ihr Fragen habt.“	S öffnen die SQL-Spielekonsole.	PU
12:28 *(7')*	Erarbeitung	• L geht rum und beantwortet Fragen.	S entdecken die SQL-Spielekonsole, indem sie das Spiel *island* laden und spielen.	EA
12:35 *(3')*	Lehrerimpuls, 2	• „Nun möchte ich euch das Gegenstück zur SQL-Spielekonsole zeigen. Es handelt sich um den SQL-Spieleeditor. Die beiden Apps ergänzen sich perfekt, denn mit dem SQL-Spieleeditor kann man genau jene Spieldateien erstellen, die danach mit der SQL-Spielekonsole geladen werden können.“ • „Ihr seht schon, worauf es hinausläuft: Ihr werdet eine solche Spieldatei als Endprodukt eures Projekts erstellen.“ • „Klickt nun im Hauptmenü von https://eskuel.de auf *SQL-Spieleditor*. Ladet die gleiche Datei wie eben, *island*.“ • „Ihr erhaltet nun eine neue Sicht auf das Spiel, nämlich eine Editor-Sicht. Editiert das Spiel, und probiert folgende Dinge: Löscht ein paar Szenen. Fügt neue Szenen hinzu. Ändert die Reihenfolge der Szenen. Editiert ein paar Szenen. Was genau lässt sich alles einstellen? Ändert den Namen des Spiels. Abschließend: Klickt auf das *Speichern*-Symbol. Euch wird angeboten, die Spieldatei abzuspeichern. Speichert sie in eurem	S öffnen den SQL-Spieleeditor.	PU

Zeit	Verlaufsform	Lehrertätigkeit	Schülertätigkeit	Sozialform
		Informatik-Ordner ab. Ladet das editierte Spiel anschließend wieder in der SQL-Spielekonsole. Ich gehe rum und helfe bei Fragen.“		
12:38 *(7')*	Erarbeitung	• L geht rum und beantwortet Fragen.	S entdecken den SQL-Spieleeditor, indem sie das Spiel *island* laden, editieren und abschließend speichern.	EA
12:45 *(5')*	Organisatorisches	• „In der nächsten Doppelstunde werden wir uns den SQL-Spieleeditor noch einmal genauer anschauen. Ihr wisst nun aber, was grob auf euch zukommt.“ • „Überlegt euch bitte, in welcher Gruppenkonstellation ihr das Projekt durchführen wollt. Die maximale Größe ist 4. Die minimale Größe ist 1. Ich empfehle euch, zu zweit oder zu dritt zu arbeiten.“ • L moderiert die Gruppenbildung. L stellt sicher, dass alle S zufrieden mit der sich ergebenden Konstellation sind. Falls es Unmut gibt (z.B. ein S keinen Anschluss findet, dies aber möchte), wird moderierend eingegriffen und eine Gruppe vermittelt.	S finden sich in Gruppen zusammen.	PU+GA

12:50 *(1')*	Lehrerimpuls	• „Überlegt euch bitte nun schon einmal grob folgende Aspekte eures Spiels:“
		  ◦ Wie sieht die Spielwelt aus?
		  ◦ Gibt es einen Protagonisten? Gibt es andere Charaktere im Spiel?
		  ◦ Gibt es einen linearen Handlungsstrang oder sind die Szenen eher unabhängig voneinander?
		  ◦ Was könnten interessante Aufgaben sein, die der Spieler lösen muss?
		• „Ich habe euch alle Informationen und Projektanforderungen auf dieser Webseite aufgeschrieben: https://mainlab.convnet.de/organisatorisches/sql-spiel/. Ihr braucht es euch heute noch nicht im Detail durchzulesen; den Link habt ihr soeben per E-Mail erhalten. “
		• „Ich gehe rum und helfe bei Fragen bzgl. Umsetzbarkeit. In 10 Minuten stellt ihr euch gegenseitig eure Ideen vor.“
			S entwickeln Ideen.	PU

12:51 *(10')*	Erarbeitungsphase	• L hält sich zurück.	S entwickeln Spielwelt mit Ideen für groben Handlungsstrang.	GA

Zeit	Verlaufsform	Lehrertätigkeit	Schülertätigkeit	Sozialform
13:01 *(9')*	Austauschphase	• „Bitte stellt den anderen Gruppen eure Spielidee vor.“ • L stellt Nachfragen, falls es sich anbietet.	S stellen vor, beantworten Fragen von Mitschülern und L, falls es welche gibt.	PU
13:10	Schluss			

Sozialformen: PU - Plenumsunterricht, GA - Gruppenarbeit, TA - Tandemarbeit, EA - Einzelarbeit

Verlaufsplanung: Stunden 2+3

Zeit	Verlaufsform	Lehrertätigkeit	Schülertätigkeit	Sozialform
11:40 *(5')*	Lehrerimpuls	• „Wie ihr von *SQL Island* wisst, muss die Spielwelt im Relationalen Modell modelliert werden; es muss also geeignete Tabellen und Spalten geben, was zusammengefasst als Relationenschema bezeichnet wird.“ • „Das heutige Ziel sollte sein, dass ihr eure Spielwelt als Relationenschema modelliert habe.“ • „Doch Vorsicht: Habt bereits jetzt einen Handlungsstrang im Hinterkopf. Die Geschichte, welche eure Figuren erleben, muss mit den modellierten Tabellen verwoben werden. Schließlich sollen nachher vom Spieler Aufgaben gelöst werden, welche diesen Tabellen mit SQL-Abfragen Informationen entlocken.“ • „Es ist vollkommen okay, wenn ihr später euer Relationenschema noch einmal überarbeitet. In der Fachprache spricht man auch von einem *agilen Vorgehen*. Je besser ihr vorausschauen könnt, welche Handlungen und Aufgaben von euren Tabellen unterstützt werden müssen, desto besser natürlich. Eine später Anpassung in einer sogenannten weiteren Iteration eures Entwicklungsprozesses ist aber vollkommen in Ordnung.“ • „Denkt euch zudem direkt passend zum Relationenschema auch Beispieldaten aus. Implementiert das Relationenschema und die Beispieldaten im SQLlite-Browser.“ • „Ihr habt dafür nun einige Zeit, die ihr aber auch benötigen werdet. Ich werde umhergehen und euch zwischendurch mal über die Schulter gucken. Ihr dürft mich jederzeit zu euch rufen. Erst 20 Minuten vor Stundenende gehen wir zur nächsten Phase über.“	S hören zu	PU
11:45 *(50')* inkl. Pause	Erarbeitungsphase	• L geht umher, berät und beantwort Fragen	S modellieren ihre Spielwelt als Relationenschema und implementieren es im SQLlite-Browser	GA
12:35 *(5')*	Lehrerimpuls	• „Wie ich gesehen habe, haben alle ein Relationenschema erstellt. Ich möchte euch nun eine technische Sache zeigen, nämlich, wie ihr die Datenbank aus	S hören zu und vollziehen nach.	PU

Zeit	Verlaufsform	Lehrertätigkeit	Schülertätigkeit	Sozialform
		dem SQLite-Browser in das Spiel laden könnt.“ • „Der SQLite-Browser bietet euch an, eure Datenbank als **.db*-Datei abzuspeichern. Tut dies bitte auch zu Sicherungszwecken. Doch wie gelangt die Datenbank nun in die Spieldatei? Ich zeige euch nun den ersten Schritt dazu. “ • „Der SQLite-Browser bietet euch an, eure Datenbank auch als **.sql*-Datei abzuspeichern. Was verbirgt sich dahinter? Wir führen diesen Speichervorgang einfach mal zusammen durch und schauen in die gespeicherte *.sql-Datei mit einem Texteditor hinein.“ • „Wie ihr seht, handelt es sich um lauter SQL-Befehle, welche die Tabellen anlegen und Beispieldaten einfügen. Es sind genau die Tabellen und Daten, die ihr manuell vorhin eingegeben habt.“ • „Geht nun zu https://eskuel.de in eurem Webbrowser und ladet diese **.sql*-Datei. Wie ihr seht, wird das Relationenschema erkannt und der SQL-Browser bietet euch nun im Wesentlichen eine Bedienmöglichkeit: Ihr könnt hier SQL-Abfragen an die geladene Datenbank absezten.“ • „Wir starten einfach mal mit einer sehr einfachen Abfrage: `SELECT * FROM` … an dieser Stelle können wir nicht mehr gemeinsam fortfahren. Eure Tabellen sind ja alle unterschiedlich. Ich gebe hier `SELECT * FROM dorf` ein, und ihr verfahrt bitte entsprechend.“ • „Ich gebe euch nun einige Zeit, eure Datenbank zu laden und einige Abfragen mit dem SQL-Browser zu testen. Probiert unbedingt eine Abfrage aus und überlegt schon einmal, welche davon man wie in Inhaltsszenen übernehmen kann.“		
12:40 *(15')*	Erarbeitungsphase	• L geht umher, berät und beantwort Fragen, insbesondere auch fachliche Fragen zur Abfragesprache SQL	S laden ihre Datenbank in den SQL-Browser und denken sich zu ihrer Spielidee passende Abfragen aus, um diese anschließend auszutesten.	GA
12:55 *(15')*	Lehrerimpuls und Feedbackrunde	• „Bevor ihr in der nächsten Doppelstunde damit beginnt, ganz konkrete Inhaltsszenen zu formulieren und zu implementieren, möchte ich euch noch eine schnelle Feedbackrunde machen lassen.“	S präsentieren ihre Arbeitsergebnisse.	PU

Zeit	Verlaufsform	Lehrertätigkeit	Schülertätigkeit	Sozialform
		• „Jede Gruppe beschreibt bitte kurz, welche Tabellen sie verwenden wollen. Bitte beschreibt auch, ob und falls ja wie diese Tabellen miteinander verknüpft sind. Beschreibt darüber hinaus eine Abfrage, welche eurer Datenbank nützliche Informationen entlockt.“ • Falls keine Fragen aus dem Plenum kommen, stellt der L eine weitergehende Frage hinsichtlich der späteren Verwendung der Tabellen.		
13:10	Schluss			

Sozialformen: PU - Plenumsunterricht, GA - Gruppenarbeit, TA - Tandemarbeit, EA - Einzelarbeit

Verlaufsplanung: Stunden 4+5

Zeit	Verlaufsform	Lehrertätigkeit	Schülertätigkeit	Sozialform
11:40 *(10')*	Lehrerimpuls, 1	• „Heute soll es an die Modellierung eines Handlungsstrangs für eure bereits in ein Relationenschema gegossene Spielwelt gehen. Wie ihr wisst, besteht ein fertiges Spiel bloß aus einer Abfolge von Szenen. Es gibt drei Szenenvarianten: 1) Text-Szenen, bei denen man einfach nur auf *Weiter* klickt, 2) Select-Szenen, bei denen der Spieler eine `SELECT ...`-Abfrage eingibt und… 3) Manipulate-Szenen.“ • „Hat jemand bereits herausgefunden, was es mit den Manipulate-Szenen auf sich hat?“ • Falls nicht durch einen S erklärt: „In Manipulate-Szenen kann der Spieler die Spielwelt verändern. Das kann z.B. das Hinzufügen eines neuen Spielcharakters oder das Löschen eines Spielcharakters sein. Um solch eine Szene zu erstellen, muss einmal die Musterlösung angegeben werden, und ebenso eine sogenannte Prüfabfrage. Die Prüfabfrage wird automatisch vom Spiel durchgeführt, um herauszufinden, ob die geforderte Änderung tatsächlich vom Spieler veranlasst wurde.“ • Erklärung von Manipulate-Szenen anhand eines Beispiels • „Öffnet bitte nun den SQL-Browser auf https://eskuel.de in eurem Webbrowser. Ladet eure Datenbank.“ • „Ihr habt nun sehr viel Zeit, um den Handlungsstrang zu modellieren. Am Ende der Stunde solltet ihr mindestens 8 Select-Szenen ausgedacht haben. Wer möchte, kann auch Manipulate-Szenen einfügen, dies ist aber optional. Die Implementierung im Spiel steht jetzt noch nicht im Vordergrund. Notiert euch die Szenen sowie die dazu passenden SQL-Abfragen digital oder handschriftlich. Testet jede Abfrage. Falls die Tabellenstruktur nicht zu euren Ideen passt, fragt am besten vorher kurz nach, bevor ihr das Relationenschema umgestaltet.“	S öffnen den SQL-Browser und laden ihre Datenbank.	PU
11:50 *(55')* inkl. Pause	Erarbeitungsphase	• L geht durch die Reihen und hilft bei der Modellierung des Handlungsstrangs; klärt Fragen.	S modellieren den Handlungsstrang.	GA
12:45 *(5')*	Lehrerimpuls	• „Ich habe viele gute Ideen gesehen. Bestimmt wollen einige von euch nun schon einmal ausprobieren, ob sie ihr eigenes Spiel laden können. Macht doch bitte einmal Folgendes: Ladet den SQL-Editor auf https://eskuel.de in eurem Webbrowser“	S hören zu und laden den SQL-Spieleeditor.	PU

Zeit	Verlaufsform	Lehrertätigkeit	Schülertätigkeit	Sozialform
		• „Öffnet ein bestehendes Spiel, egal welches. Löscht alle Szenen bis auf eine. Editiert diese, so dass sie der ersten Szene eures Spiels entspricht. Ladet eure Datenbank. Speichert das Spiel ab als *.xml-Spieldatei.“ • „Ladet nun diese Datei in der SQL-Spielekonsole auf https://eskuel.de in eurem Webbrowser.“ • „Überzeugt euch damit davon, dass euer Spiel *prinzipiell* implementierbar ist. Den Großteil der Implementierung führen wir in der nächsten Doppelstunde durch.“ • „Ich gebe euch nun 10 Minuten Zeit, den Mechanismus zur Spieleerstellung zu erproben. “		
12:50 *(5')*	Erarbeitungsphase	• L geht durch die Reihen und hilft; klärt Fragen.	S überzeugen sich davon, dass ihr Spiel *prinzipiell* implementierbar ist.	GA
12:55 *(15')*	Feedbackrunde	• „Bitte stellt knapp vor, in welche Richtung sich euer Spiel ausgestaltet hat. Beschreibt, ob ihr Änderungen vorgenommen habt. Holt Tipps aus dem Plenum ein, was noch verbessert werden könnte.“ • „Alle anderen überlegen bitte derweil, ob das Spiel *spielbar* wirkt für sie. Stellt ggf. Fragen.“	S stellen den neuesten Entwicklungsstand ihrer Spiele vor.	PU
13:10	Schluss			

Sozialformen: PU - Plenumsunterricht, GA - Gruppenarbeit, TA - Tandemarbeit, EA - Einzelarbeit

Verlaufsplanung: Stunden 6+7

Zeit	Verlaufsform	Lehrertätigkeit	Schülertätigkeit	Sozialform
11:40 *(5')*	Lehrerimpuls	• „Heute sollt ihr das Spiel komplett fertig implementieren. Wir verzichten heute auf eine Feedbackrunde, da alle Gruppen die Zeit vollkommen benötigen.“ • „Falls ihr doch schon fertig sein solltet, so gibt es noch ein Bonuselement: Es gibt neuerdings die Möglichkeit, Bildszenen in das Spiel einzufügen. Wie das genau geht, dazu habe ich euch ein Tutorial verfasst, welches ihr unter https://mainlab.convnet.de/organisatorisches/sql-spiel/ im letzten Abschnitt findet.“ • „Ich werde die ganze Doppelstunde über ansprechbar sein und bei technischen oder auch fachlichen sowie inhaltlichen Fragen zur Seite stehen. Bitte schaut auch noch einmal in die genaue Aufgabenbeschreibung unter https://mainlab.convnet.de/organisatorisches/sql-spiel/, so dass ihr keine Komponenten vergesst. Frohes Schaffen!“		PU
11:45 *(80')*	Erarbeitungsphase	• Betreuung der Gruppen • Beantwortung von Fragen • Hilfestellung bei technischen Problemen • Hilfestellung bei fachlichen Problemen	S implementieren ihr Spiel.	GA
13:05 *(5')*	Organisatorisches	• „Bitte ladet eure Datei in der IServ-Aufgabe hoch. Falls eine Gruppe noch nicht fertig geworden ist, so ist das kein Problem; ich lasse wie besprochen die Abgabe bis Dienstag, 28.05. geöffnet.“	Gruppen laden ihre Abgabe hoch.	PU
13:10	Schluss			

Sozialformen: PU - Plenumsunterricht, GA - Gruppenarbeit, TA - Tandemarbeit, EA - Einzelarbeit

Verlaufsplanung: Stunde 8

Zeit	Verlaufsform	Lehrertätigkeit	Schülertätigkeit	Sozialform
11:40 *(5')*	Einstieg	• Begrüßung, Vorstellung des Besuchs • „Heute ist Release Day! Ihr habt mir eure Spiele geschickt. Ich habe sie hochgeladen auf eine Website.“ • „Wir führen die heutige Präsentationsstunde wie folgt durch.“ 1. Kurze Spielvorstellung und Resümieren des Entwicklungsprozesses 2. Spielen der jeweils anderen Spiele 3. Feedback verfassen und den anderen Gruppen zukommen lassen		PU
11:45 *(15')*	Präsentationen	• L hört zu, stellt nur im Ausnahmefall Fragen	S präsentieren, stellen evtl. Fragen	PU
12:00 *(2')*	Lehrerimpuls	• „Jetzt geht es ans Spielen. Startet pro Gruppe auf drei Rechnern den Browser und ladet die Seite https://eskuel.de/inf-11-1/. Wählt euer *eigenes* Spiel aus; ihr werdet direkt in die SQL-Spielekonsole weitergeleitet“. • L teilt Feedbackbögen aus. • „Probiert nun die verschiedenen Spiele aus. Nachdem ihr ein Spiel gelöst habt (oder aber frustriert stecken geblieben seid) füllt bitte einen Feedbackbogen aus. Ich sammele alle Feedbackbögen am Ende der Stunde und lasse sie euch dann heute Abend digital zukommen.“ • L moderiert Zuordnung; nur, falls nötig. (Die genaue Zuordnung ist unerheblich) • „Viel Spaß beim Spielen!“		EA+TA
12:02 *(21')*	Erarbeitungsphase	• Lehrer beobachtet Spielerlebnisse der S (hat bereits alle Spiele vorab gespielt)	S spielen, testen, füllen Feedbackbögen aus	EA+TA
12:23 *(2')*	Verabschiedung	• „Ich hoffe, ihr habt die Früchte eurer Arbeit genießen können. Wer Lust hat, kann sein Spiel als freie Software zur Verfügung stellen. Ich würde das Spiel dann direkt auf die Startseite https://eskuel.de/ stellen und somit können es auch Schüler/innen an anderen Schulen spielen, oder aber der nächste Informatik-Jahrgang am EBG im neuen Schuljahr.“ • Verabschiedung		PU

Zeit	Verlaufsform	Lehrertätigkeit	Schülertätigkeit	Sozialform
12:25	Schluss			

Sozialformen: PU - Plenumsunterricht, GA - Gruppenarbeit, TA - Tandemarbeit, EA - Einzelarbeit

Projektanforderungen und Hinweise

Hier folgt eine Übersicht über die Projektarbeit.

Zeitplan

Datum	
Di., 30.4.	**1.**: Gruppen bilden + Projektidee
Fr., 3.5.	**2.+3.**: Tabellen entwerfen und Story-Line beginnen
Di., 14.5.	**4.+5.**: Story-Line entwerfen + erste SQL-Abfragen formulieren (dabei eventuell Tabellen anpassen)
Fr., 24.5.	**6.+7.**: Implementierung
Di., 28.5.	**(Abgabeschluss)**
Fr., 31.5.	**8.**: Präsentation, Spielerunde + Feedback

Inhaltliche Konzeption

Konzipiere und erstelle ein Spiel, welches zum Lernen der Datenbankabfragesprache *SQL* verwendet werden kann. Hier eine genauere Projektbeschreibung.

- ***Spielwelt***

  Spielt der Protagonist im Weltall, unter Wasser, ...? Wer interagiert mit wem?

  Entwirf ein relationales Modell (Tabllenstruktur) für die Spielwelt. Alle für das Spiel benötigten Daten müssen irgendwo in den Tabellen abgelegt werden können.

- ***Story-Line***

  Dies ist das Herz des Spiels. Überlege dir einen Weg, wie du didaktisch geschickt die Sprache SQL einführst. Natürlich sollten die ganz einfachen Abfragen zu Beginn eingeführt werden. Das Spiel soll auch ohne Profiwissen spielbar sein.

  Deine Story sollte als eine Abfolge von Szenen darstellbar sein, wobei jede Szene einem von drei Typen entspricht:

  1. **Inhalts-Szene**. Diese Art von Szene kann aus Dialog, Erzählung bestehen. Der Spieler kann anschließend auf „Weiter“ klicken, um zur nächsten Szene zu gelangen.
  2. **Select-Szene**. Nach einem Text wird der Spieler aufgefordert, eine SQL-Abfrage abzusetzen. Beachte, dass die einzugebende SQL-Abfrage zu Beginn auch bereits

im Text vorgegeben sein darf, um dem Spieler etwas zu demonstrieren (z.B. „Tippe jetzt `SELECT * FROM bewohner` ein.“). Es kann aber auch komplett dem Spieler überlassen werden, die passende SQL-Abfrage zu formulieren. Nur, wenn die SQL-Abfrage das passende Ergebnis bzw. den gewünschten Effekt hat, gelangt der Spieler zur nächsten Szene.

3. **Manipulate-Szene**. Es wird eine `INSERT...` oder `UPDATE...` oder `DELETE...`-Anfrage erwartet. Zusätzlich muss nun noch eine `SELECT`-Abfrage angegeben werden, welche Aufschluss darüber gibt, ob die Manipulation korrekt durchgeführt wurde. Wenn z.B. ein Gegenstand gelöscht werden soll, muss der Spieler eine `DELETE`-Anfrage formulieren. Zur Prüfung prüft eine `SELECT`-Abfrage (Teil deines Spiels), welche Gegenstände noch da sind (z.B. `SELECT * FROM gegenstaende`). Das hier zurückgelieferte Ergebnis gibt Aufschluss darüber, ob der Spieler korrekt gelöscht hat.

Deine Story sollte **10 bis 15 Aufgaben** enthalten, in denen der Spieler selbstständig eine SQL-Abfrage formulieren soll. Welche Aspekte von SQL vorgestellt werden, bleibt dir überlassen. Einzige Bedingung ist, dass in irgendeiner Form **Joins** und **Aggregation** vorkommen sollen, in jedoch beliebigem Umfang.

Inspiration

Die folgenden SQL-Games gibt es bereits. Sie können als Anregungsquelle dienen. Euer Spiel wird später im Browser laufen, ganz ähnlich zu „SQL Jungle“.

- SQL Island: Das Original!
- SQL-Spielekonsole mit vorgeladenen Spielen *baufirma*, *jack-frost*, *ozean*, *island*, *sql-jungle*
- Kommissar Smiths Abenteuer (jedoch sehr textlastig)

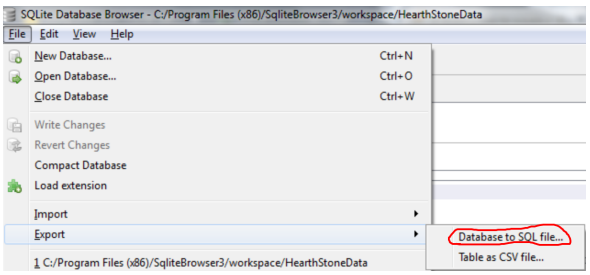
Weitere Lernmaterialien / Nachschlagehilfen:

- SQL-Browser
- Lernmaterialien zum Datenbanken & SQL

Werkzeuge

Die folgenden Werkzeuge sollten zum Einsatz kommen.

- **SQLiteBrowser** zum Erstellen der Datenbank, Anlegen der Tabellen, Einfügen von Daten, Testen von SQL-Abfragen etc. Abschließend kann die Datenbank als `*.sql`-Datei exportiert werden.



- **SQL-Spieleeditor** zum Erstellen der Spieldatei. Die damit erzeugte Spieldatei kann dann von der SQL-Spielekonsole eingelesen werden.
- **ChatGPT** oder vergleichbares LLM (Large Language Model). Unterstützung beim Formulieren von SQL, Erzeugen von Daten, Formulieren der Story, ...
- Erweiterung: Grafiken für die Szenen erzeugen mit z.B. **DALL·E**, Midjourney, Paint, ...

Abgabe

Die Abgabe besteht aus zwei Teilen.

1. **Spieldatei**: Einreichung bis **Dienstag, 28.05.2024**, per IServ

   Diese `*.xml`-Datei enthält alle Spielinformationen. Falls ihr Grafiken erzeugt habt, gebt ihr diese separat dazu auch noch ab.

2. **Release-Präsentation**: Release Day ist der **Freitag, 31.05.2024**

   Stellt als Gruppe euer Spiel kurz 3-5 Minuten vor: Welche Spielwelt habt ihr euch überlegt, was waren Schwierigkeiten bei der Umsetzung? Nicht zuuu viel verraten, das Spiel nur kurz anteasern.

Weitere Erklärungen zu manipulierenden SQL-Abfragen

- Erklärungen zum **Hinzufügen**, **Ändern** und **Löschen** von Datensätzen mittels SQL:
  - `INSERT INTO`
  - `UPDATE`
  - `DELETE FROM`

Neues Feature Bild-Szenen (Nur Prototyp!)

Wie funktioniert's? Folge diesem Mini-Tutorial:

1. Probiere das Feature aus, indem du diese Spieldatei `island.xml` herunterlädst und dann in die Spielkonsole hochlädst.
2. Die Spieldatei `island.xml` ist im XML-Format gegeben, deshalb auch die Dateinamensendung `.xml`. Öffne die Datei mit einem Texteditor, z.B. mit Visual Studio Code. Du siehst nun den XML-Code. XML folgt dem gleichen Prinzip wie HTML.
3. Betrachte nun die verschiedenen Szenentypen. Es gibt `text-scene`, `select-scene`, `manipulate-scene` und... seit dieser Version eben auch eine `image-scene`. Was wird in das Tag `<image-scene>` eingetragen? Ein Bild! Das Bild muss als `*.png`-Datei vorliegen. Und jetzt stellt sich noch die Frage: Wie kommt die PNG-Datei in die XML-

Datei rein?

4. Folgender Trick wird verwendet: Es gibt für PNG-Dateien eine sogenannte Base64-Kodierung. Das bedeutet, dass die Bilddaten in einen einzigen langen String übersetzt wird. Und später dann wieder zurück in das Bild. Es reicht also, wenn du im XML-Dokument diesen String angibst.

5. Um deine Datei in einen Base64-String umzuwandeln, mache folgendes. Nimm dein Bild, z.B.  und übersetze es hier in einen Base64-String. Dies ergibt einen kryptisch anmutenden langen String, der mit `iVBORw0KGgoAAAANSUhEUgAAAB8AAAAfCAI...` beginnt. Hier ist der komplette String: Link.

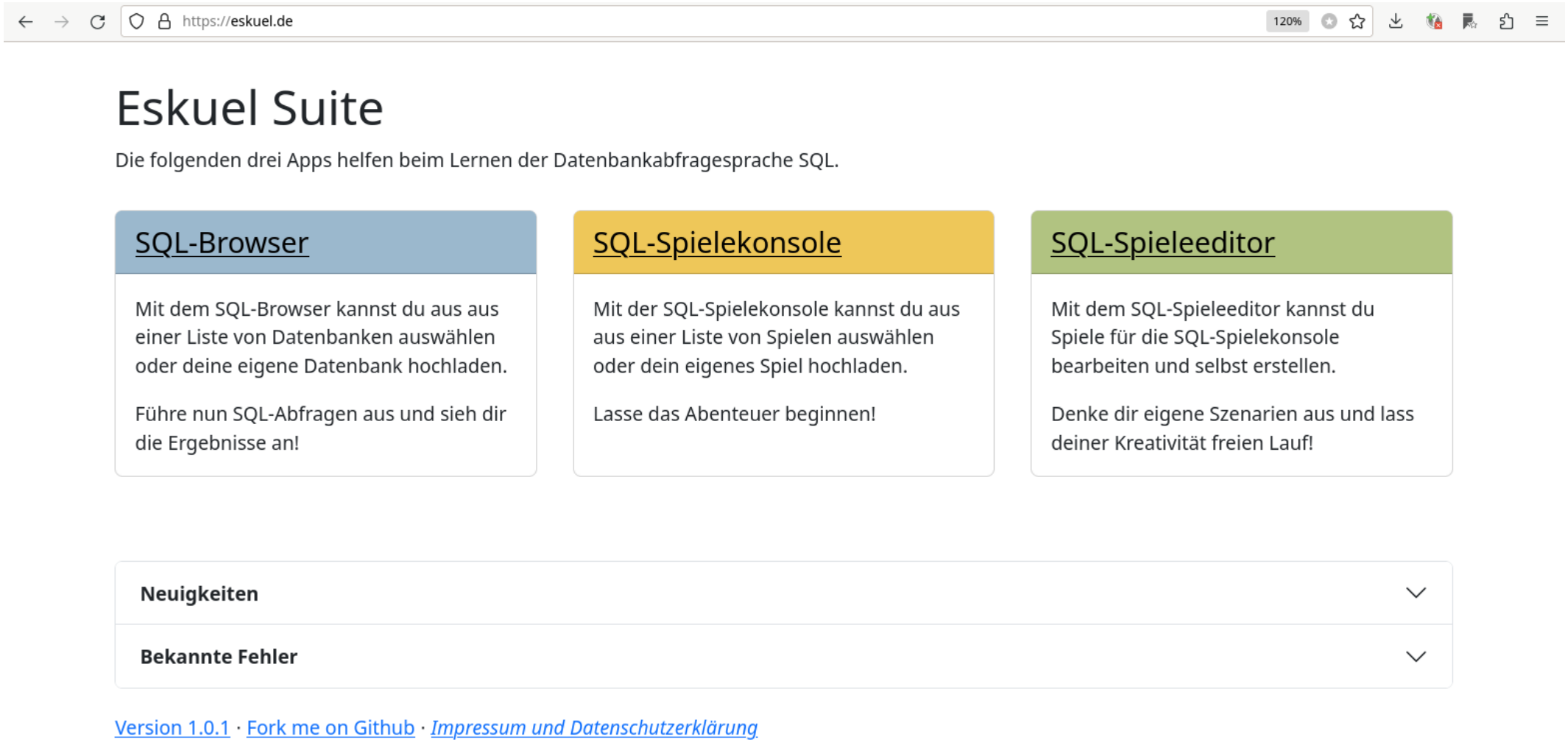
6. Genau diesen String schreibst du nun in das Tag `<image-scene>` in deiner XML-Datei. Das Bild wird dann in der Spielkonsole angezeigt.

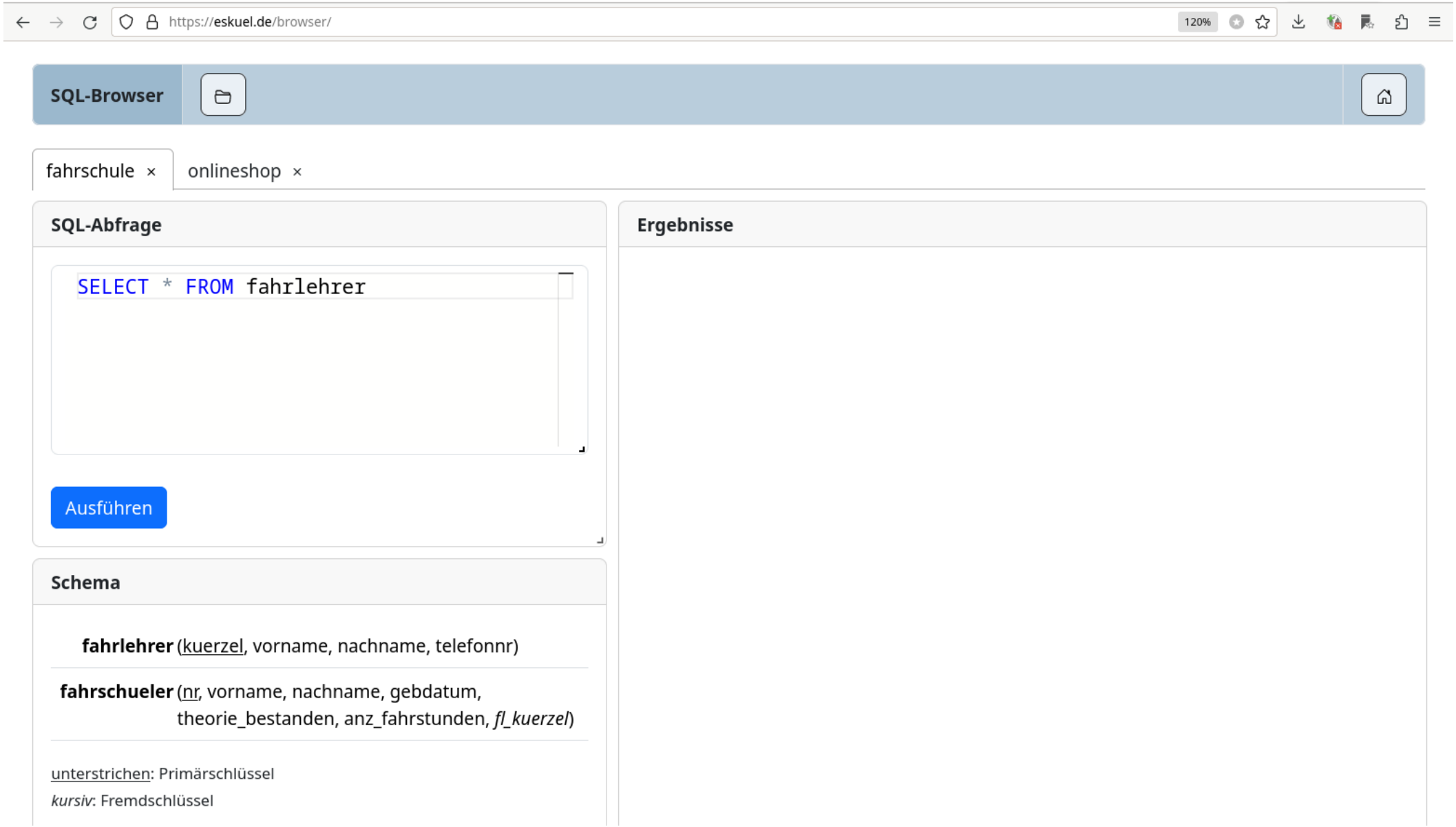
Image-Szenen können leider momentan nur über den XML-Code mit einem Texteditor hinzugefügt werden. Die Unterstützung im SQL Spieleeditor folgt, sobald ich wieder Zeit habe😅

Eskuel-Suite

Die Eskuel-Suite besteht aus den drei Apps SQL-Browser, SQL-Spielekonsole und SQL-Spieleeditor und ist online verfügbar unter

https://eskuel.de/




https://eskuel.de/browser/
120%
SQL-Browser
fahrschule ×
onlineshop ×
SQL-Abfrage
SELECT * FROM fahrlehrer
Ausführen
Ergebnisse
Schema
fahrlehrer (kuerzel, vorname, nachname, telefonnr)
fahrschueler (nr, vorname, nachname, gebdatum, theorie_bestanden, anz_fahrstunden, fl_kuerzel)
unterstrichen: Primärschlüssel
kursiv: Fremdschlüssel

SQL-Spielekonsole

SQL Island

Neustart

Der Klassiker! (Aus Urheberrechtsgründen sind nur die ersten Szenen enthalten)

Szene 7 / 15

„Probier doch mal die folgende Abfrage aus: SELECT * FROM dorf“

Weiter

SQL-Abfrage

```
SELECT * FROM bewohner
```

Ergebnisse

Noch nicht gelöst ×

```
SELECT * FROM bewohner
```

bewohnernr	name	dorfnr	geschlecht	beruf	gold	status
1	Paul Backmann	1	m	Baecker	850	friedlich
2	Ernst Peng	3	m	Waffenschmied	280	friedlich
3	Rita Ochse	1	w	Baecker	350	friedlich
4	Carl Ochse	1	m	Kaufmann	250	friedlich
5	Dirty Dieter	3	m	Schmied	650	boese
6	Gerd Schlachter	2	m	Metzger	4850	boese

https://eskuel.de/game-editor/ 120%

SQL-Spieleeditor

island ×

Name:	SQL Island
Copyright:	© Johannes Schildgen, TU Kaiserslautern, https://sql-island.informatik.uni-kl.de

Teaser: Der Klassiker! (Aus Urheberrechtsgründen sind nur die ersten Szenen enthalten)

Image ×

(Hier kommt das Bild hin -- noch nicht implementiert)

Select ×

„Probier doch mal die folgende Abfrage aus: SELECT * FROM dorf“

SQL-Lösung

```
SELECT * FROM dorf
```

Select ×

„Zeige mir die Liste der Bewohner.“

SQL-Lösung

```
SELECT * FROM bewohner
```

Datenbank

Datenbank erfolgreich geladen

gegenstand (gegenstand, besitzer)

dorf (dorfnr, name, haeuptling)

bewohner (bewohnernr, name, dorfnr, geschlecht, beruf, gold, status)

unterstrichen: Primärschlüssel
kursiv: Fremdschlüssel

Feedbackbogen

Spiel: _____ Name des Testers: _____

Was war gut?	Was kann verbessert werden?	Weitere Tipps/Hinweise:

Feedbackbogen

Spiel: _____ Name des Testers: _____

Was war gut?	Was kann verbessert werden?	Weitere Tipps/Hinweise: