



SEISMIK

Dokumentation

Daniel Schwarz

18.11.2021



Inhaltsverzeichnis

1 Vorwort	6
2 Aufgabenstellung	7
2.1 Anforderungen	7
3 Grobstruktur	8
3.1 Übersicht	8
3.2 Produktnamen	8
3.3 Logo	8
3.4 Funktionen	8
4 Feinstruktur	9
4.1 Referenz (Instagram)	9
4.1.1 Users	10
4.1.2 Posts	11
4.1.3 Tags (und Following)	12
4.2 Relevante Objekte	12
4.2.1 Users	12
4.2.2 Photos	14
4.2.3 Tags	16
4.3 Beziehungen	18
4.3.1 posten	18
4.3.2 liken	18
4.3.3 comments	19
4.3.4 photo_tags	19
4.3.5 follows	20
4.4 Typische Operationen	20
4.5 weitere relevante Informationen	20
4.5.1 Datenmengen	21
4.5.2 Gesetzliche Bestimmungen	21
4.5.3 Sicherung der Datenbank	21
4.5.4 Zeichensatz	21
5 Entity Relationship-Modell	22
5.1 Gesamtdarstellung	22
5.2 Detaildarstellung	23
5.2.1 USERS	23
5.2.2 PHOTOS	23



5.2.3 TAGS	23
6 Relationenmodell	24
7 Normalisierung	25
8 Datenbankobjekte	26
8.1 Tabellen	26
8.2 Dummy-Datensätze	26
9 Datenmanipulation	27
9.1 Foto posten	27
9.2 Foto löschen	27
9.3 Liken	27
9.4 Unliken	27
9.5 Folgen	28
9.6 Unfollow	28
10 Datenabfrage	29
10.1 Foto des Users anzeigen	29
10.2 Kommentare eines Fotos	29
10.3 Top 5 liked Fotos	29
11 Menugestaltung	30
11.1 Ordner- und Datei-Struktur	30
11.2 Menu-Übersicht	31
11.2.1 Hauptmenu	32
11.2.2 Untermenü: Datenbankstruktur	33
11.2.3 Script: Benutzerauswahl / -wechsel	34
11.2.4 Untermenü: Lesen spezifischer Datensätze	35
11.2.5 Untermenü: ändern von Datensätzen	36
12 Quellenverzeichnis	38
12.1 Abbildung 1-4	38
12.2 Logo	38



Tabellenverzeichnis

1	Users.id	13
2	Users.username	13
3	Users.passwd	14
4	Users.created_at	14
5	Photos.id	15
6	Photos.image_url	15
7	Photos.created_at	16
8	Tags.id	16
9	Tags.tag_name	17
10	Tags.created_at	17
11	Beziehung posten	18
12	Beziehung liken	18
13	Beziehung comments	19
14	Beziehung photo_tags	19
15	Beziehung follows	20
16	Hauptmenu	32
17	Untermenü - Datenbankstruktur	33
18	Untermenü - Datensätze Lesen	35
19	Untermenü - Datensätze ändern	36



Abbildungsverzeichnis

1	Instagram - Übersicht	9
2	Instagram - Useraccount	10
3	Instagram - Post eines Fotos	11
4	Instagram - Following und Hashtags	12
5	Gesamtübersicht SEISMIK	22
6	SEISMIK - Entität USERS	23
7	SEISMIK - Entität PHOTOS	23
8	SEISMIK - Entität TAGS	23
9	SEISMIK - Relationenmodell	24
10	Ordner-/Dateistruktur	30
11	Menugestaltung	31
12	Hauptmenu	32
13	Menu Datenbankstruktur	33
14	Benutzerauswahl / -wechsel	34
15	Datensätze Lesen	35
16	Datensätze ändern	36



1 Vorwort

Diese Dokumentation wurde für mein Projekt bei Cimdata mit OracleSQL erstellt.

Aufgabe ist es, eine Datenbank anzulegen und mit Hilfe von SQL-Scripten via dem cl-tool sqlplus/sqlcl ein kleines “Programm” zu erstellen. Zusätzlich ist eine Dokumentation sowie Präsentation gewünscht.

Die SQL-Scripte sollen:

- eine Datenbank anlegen
- die erforderlichen Tabellen erstellen
- Dummy Daten in die Tabellen einfügen
- eine Menustruktur zur wiederholten Ausführung anbieten
- man soll folgendes im Menu auswählen können
 - Tabellen leeren/erstellen
 - Datensätze löschen
 - Datensätze anlegen
 - Datenabfragen realisieren

Ich habe hierbei mit Gnu/Linux - Fedora Silverblue¹ - gearbeitet. Um den Oracle SQL Server aufzusetzen benutze ich Podman - ein Container-Tool, verwendet um den Server in einem Container aufzusetzen. Das Image ist von gvenzl/oracle-xe

Listing 1: Podman container cmd

```
1 podman run -d -p 1521:1521 --name oracle-18c -v  
2 <path/to/your/initScripts>:/container-entrypoint-initdb.d:Z  
3 -e ORACLE_PASSWORD="yourPWD" -v oracle-volume:/opt/oracle/oradata  
4 gvenzl/oracle-xe:18.4.0-full
```

Zur Dokumentation habe ich Markdown benutzt und dieses mit dem Template von Eisvogel via Pandoc in eine PDF-Datei gerendert.

Listing 2: Eisvogel cl-Befehl

```
1 pandoc documentation.md -o documentation.pdf --from markdown --template  
eisvogel -V lof -V lot -V lang=de --shift-heading-level-by=-1 --toc  
--number-sections --listings
```

¹ Immutable OS

2 Aufgabenstellung

Entwickeln Sie eine Datenbank zu einem von Ihnen gewählten Thema mit den Schwerpunkten: Konzeptueller und physischer Entwurf der Datenbank.

2.1 Anforderungen

Innerhalb der Projektwoche (Montag bis Donnerstag) soll eine Aufgabe oder abgegrenzte Teilaufgabe bearbeitet werden. Hierfür müssen folgende Projekt-Schwerpunkte umgesetzt werden:

- Konzeptueller Entwurf der Datenbank, deren Ergebnis in der Dokumentation festgehalten wird,
- Physischer Entwurf der Datenbank, deren Ergebnis durch Skripte realisiert wird und
- Präsentation des Projektes

Die Bearbeitung des Projektes kann als Einzel- oder Gruppenarbeit, bestehend aus 2 Teilnehmern erfolgen. Der Dozent ist prinzipiell der Gruppenleiter, dessen Eingreifen in die Geschicke der Gruppe nur in Ausnahmefällen passiert, um beispielsweise das Projekt voranzubringen.

Mit Beginn des Projektes muss sich jeder Studierende entschieden haben, ob das Projekt als Einzel- oder Gruppenarbeit durchgeführt wird. Ein nachträgliches Ändern des Teams ist während der Projektphase nicht möglich.



3 Grobstruktur

3.1 Übersicht

Der Kunde wünscht, eine europäische Konkurrenz-Entität für Instagram zu etablieren. Hierfür soll verstärkt auf Anonymität, No-Tracking und Datenschutz für zukünftige Benutzer gesetzt werden. Zudem soll es Werbefrei gestaltet werden. User können Geld verdienen in dem Sie bestimmte Kriterien erfüllen.

Das Produkt soll in einer ersten Version als Prototyp dienen, um notwendige Schnittstellen, zu bereits vorhanden Backendsystemen des Kunden auszubauen und anpassen zu können. Später wird das Projekt dann im Umfang ausgebaut und auf verschiedenen Frontendapps verteilt. Angedacht sind Mobile-apps, eine Web-app als SaaS sowie eine Desktop-app für die gängigen Betriebssysteme.

3.2 Produktnamen

Für das Projekt habe ich den Namen “**SEISMIK**” gewählt. In der Überzeugung das wir mit diesem Produkt den Markt der Social-Media-Apps “erschüttern” werden.

3.3 Logo

Das Logo und die Corporate Identity habe ich in Abstimmung mit dem Auftraggeber von einem Grafik-Designer entwickeln lassen und lizenziert.

3.4 Funktionen

Benutzer sollen sich Registrieren können und anschließend ein- bzw. ausloggen. Sie können dann Posts erstellen in deren Inhalt es sich - in der ersten Ausbaustufe - vor allem um Bilder handelt. Die Benutzer sollen Posts von anderen Benutzern liken und kommentieren können. Benutzer können anderen Benutzern folgen.

4 Feinstruktur

4.1 Referenz (Instagram)

Instagram ist ein soziales Netzwerk, ähnlich Twitter oder Facebook. Der Fokus liegt hier vor allem auf Video- und Foto-Sharing.

Mit mehr als 20 Mio. Nutzern allein in Deutschland und 150 Mio. Weltweit zählt Instagram zu einem der größeren Vertreter der Social-Media-Apps.

Man kann es sich wie eine Art Online-Fotoalbum vorstellen, in dem man Einblicke in sein Leben geben kann. Es ist auf allen gängigen Plattformen erhältlich und wird auch gern von Unternehmen zur Reichweitensteigerung genutzt.

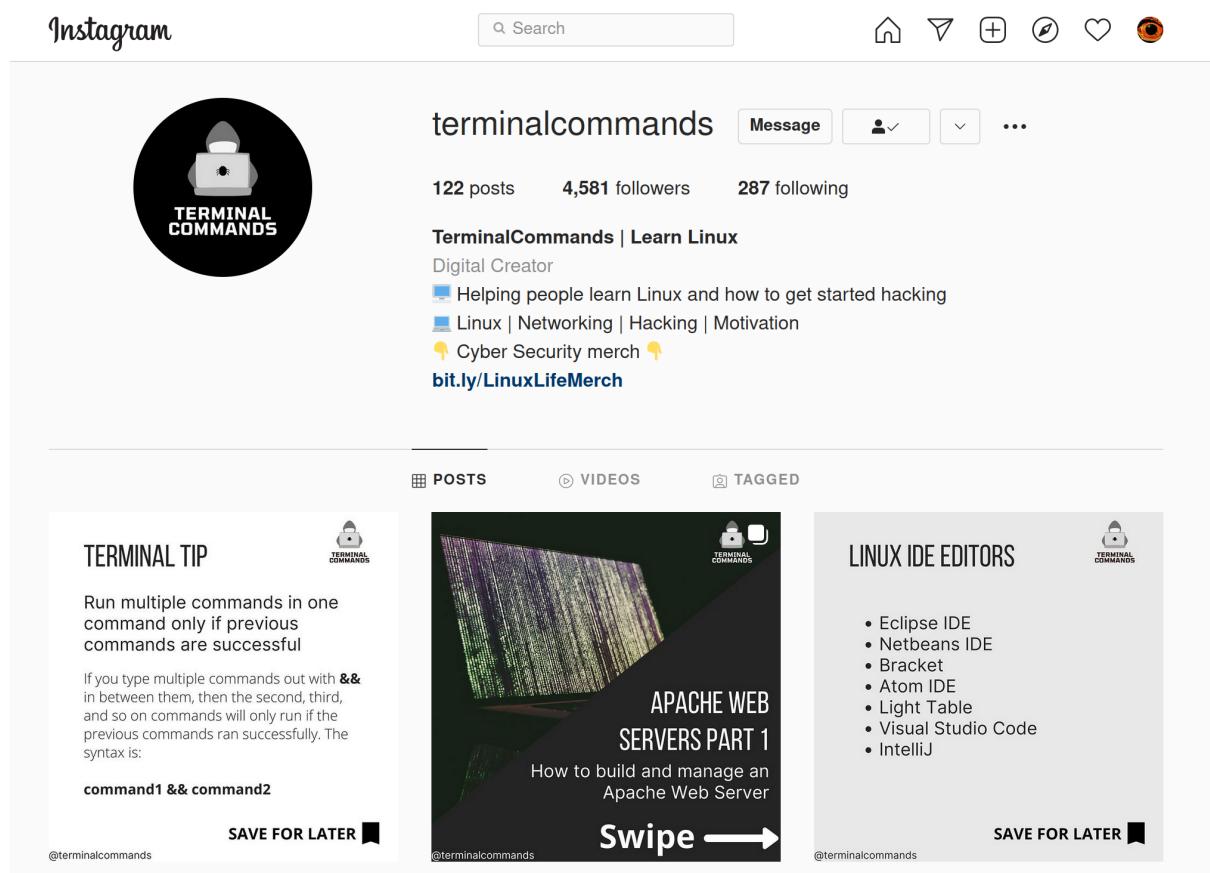


Abbildung 1: Instagram - Übersicht

4.1.1 Users

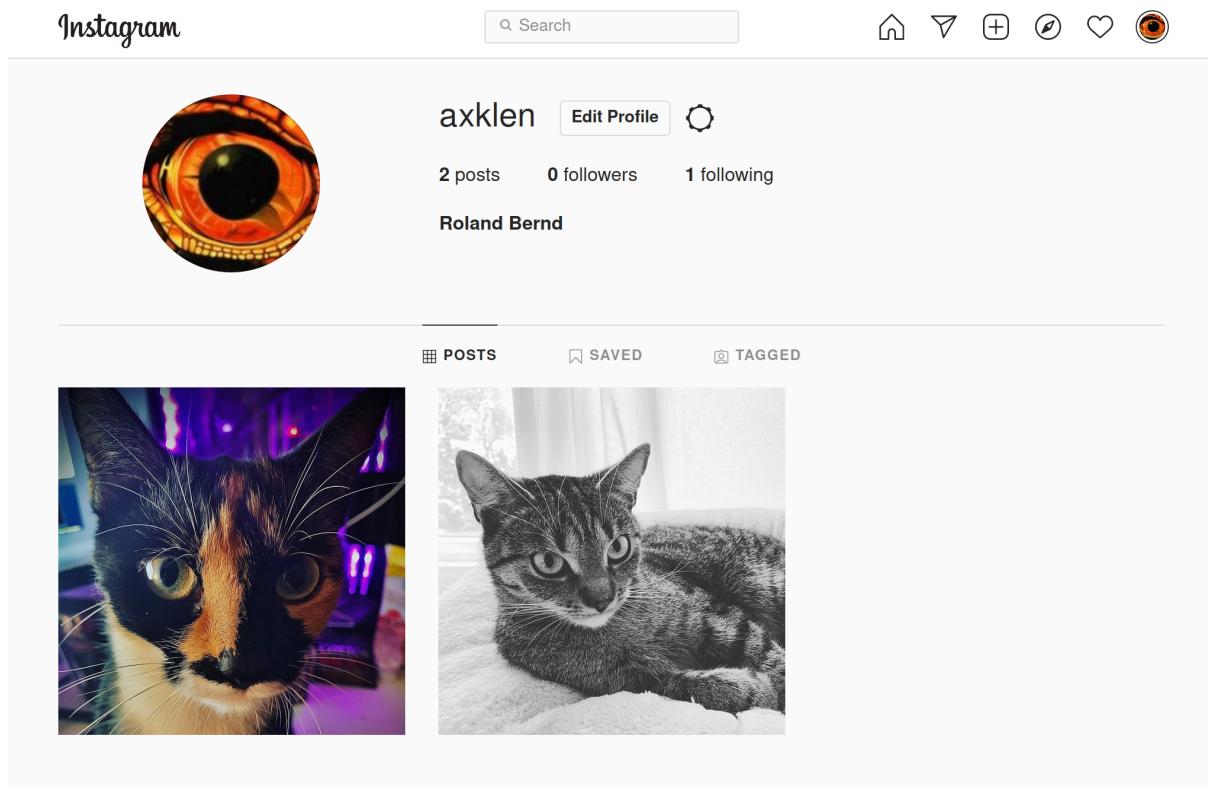


Abbildung 2: Instagram - Useraccount

Benutzer laden Bilder hoch, die sie durch Tags (Hashtags) kategorisieren können. Diese Bilder können in Ihrer Größe verändert werden und man kann verschiedene Farbfilter auf sie Anwenden.

Funktionsübersicht

- Hochladen von Bildern aus dem Album oder direkte Verknüpfung mit der Kamera des Smartphones oder Tablets
- Hochladen oder Drehen von Videos
- Geotag-Funktion, über die eine eigene Fotokarte eines Nutzers angelegt werden kann
- Austausch über Fotos mit anderen Usern
- Like-Funktion in Form eines Herzes
- Kommentar-Funktion, mit der Möglichkeit Hashtags zu verwenden

4.1.2 Posts

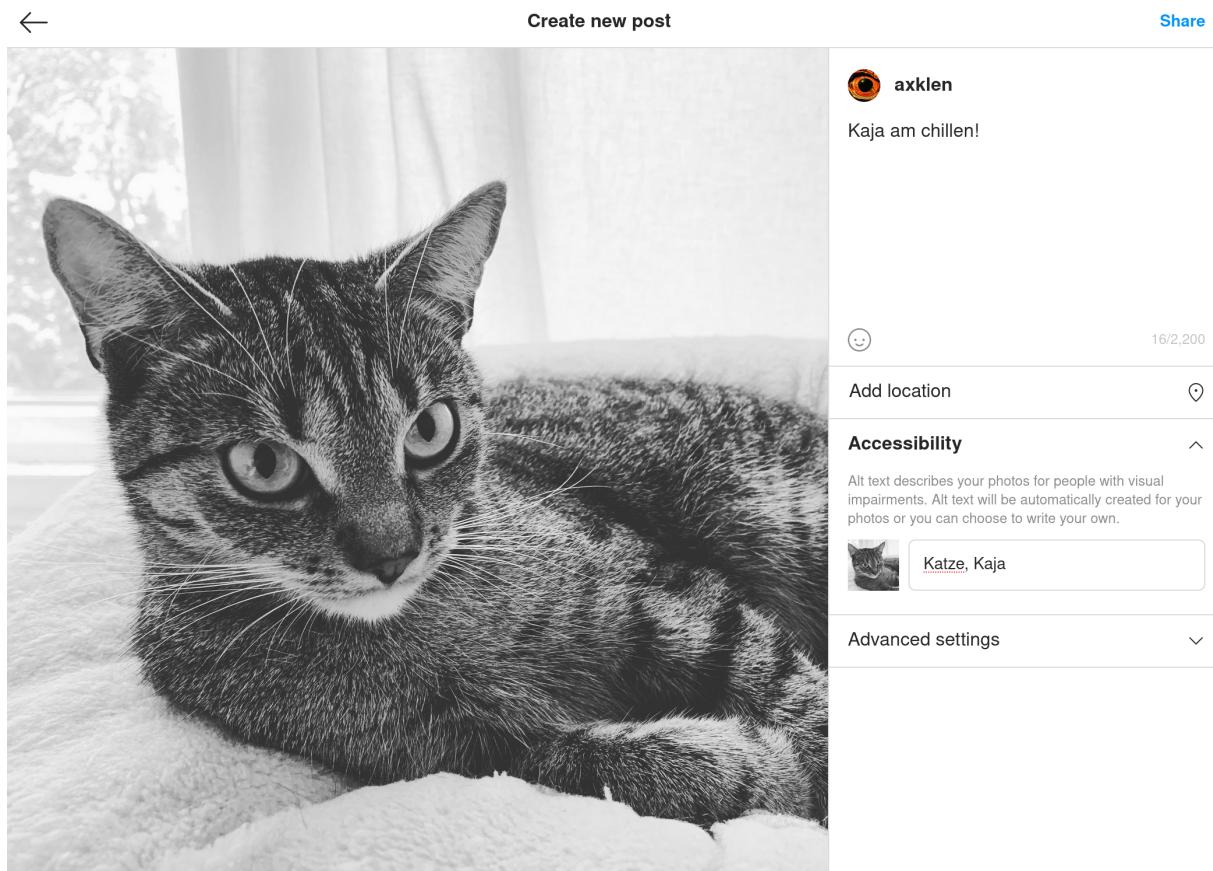


Abbildung 3: Instagram - Post eines Fotos

4.1.3 Tags (und Following)

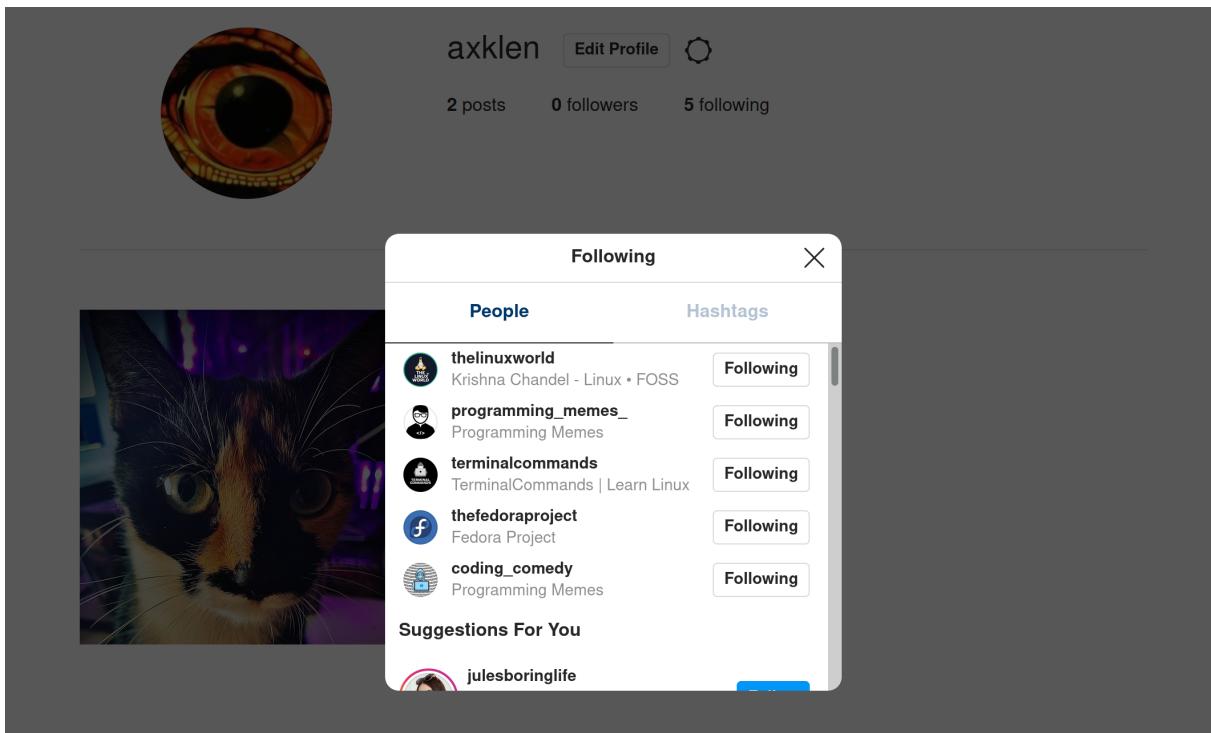


Abbildung 4: Instagram - Following und Hashtags

4.2 Relevante Objekte

Aus der Analyse von Instagram ergeben sich die nachfolgend beschriebenen relevanten Objekte für unser Projekt SEISMIK.

4.2.1 Users

Das sind die Benutzeraccounts zur Identifizierung eines Benutzers.

- Benutzername ist unique und nicht leer (min. 4 Zeichen)
- Passwort ist nicht leer (min. 8 Zeichen)
- beim erstellen des Benutzers wird ein Timestamp erstellt

**Tabelle 1:** Users.id

id	Eindeutige ID zur internen Identifizierung der User
Wertebereich	Ganze Zahlen
Werte-Einschränkung	Nein
Wertevergabe	System / Auto_Increment
Default-Wert	Nein
NULL-Wert-erlaubt	Nein
Duplikate erlaubt	Nein
Primärschlüssel	Ja
Sonst	automatisch vom System verwaltet über: <code>GENERATED ALWAYS AS IDENTITY</code>
Beispielwerte	1, 2, 3, ...

Tabelle 2: Users.username

username	Eindeutiger vom Benutzer gewählter Identifier, Unique und min. 4 Zeichen
Wertebereich	Text
Werte-Einschränkung	min. 4 Zeichen
Wertevergabe	Benutzereingabe
Default-Wert	Nein
NULL-Wert-erlaubt	Nein
Duplikate erlaubt	Nein
Primärschlüssel	Nein
Sonst	-
Beispielwerte	'Roland', '11KidShredder', 'user.dummy'

Tabelle 3: Users.passwd

passwd	vom Benutzer gewähltes Passwort (gehashed in SHA2), min. 8 Zeichen
Wertebereich	Text
Werte-Einschränkung	min. 8 Zeichen
Wertevergabe	Benutzereingabe
Default-Wert	Nein
NULL-Wert-erlaubt	Nein
Duplikate erlaubt	Ja
Primärschlüssel	Nein
Sonst	-
Beispielwerte	‘12345678’, ‘12fdf2e123’, ‘sdf-?. .ddd’

Tabelle 4: Users.created_at

created_at	vom System automatisch erstellter Timestamp bei Account-Erstellung
Wertebereich	Datum & Zeit
Werte- Einschränkung	Nein
Wertevergabe	System, bei Erstellung des Accounts
Default-Wert	Ja, sysdate
NULL-Wert-erlaubt	Ja
Duplikate erlaubt	Ja
Primärschlüssel	Nein
Sonst	-
Beispielwerte	‘11.11.2021 23:59:59’ ...

4.2.2 Photos

Dies sind die Photos die von Usern gepostet werden

- Das Foto wird vorerst als url gespeichert
- beim erstellen des Fotos wird ein Timestamp erstellt

Tabelle 5: Photos.id

id	Eindeutige ID zur internen Identifizierung der Fotos
Wertebereich	Ganze Zahlen
Werte-Einschränkung	Nein
Wertevergabe	System / Auto_Increment
Default-Wert	Nein
NULL-Wert-erlaubt	Nein
Duplikate erlaubt	Nein
Primärschlüssel	Ja
Sonst	automatisch vom System verwaltet über: GENERATED ALWAYS AS IDENTITY
Beispielwerte	1, 2, 3, ...

Tabelle 6: Photos.image_url

image_url	Verlinkt zum geposteten Foto eines Benutzers, später ersetzt durch blob
Wertebereich	Text
Werte- Einschränkung	Nein
Wertevergabe	Benutzereingabe
Default-Wert	Nein
NULL-Wert-erlaubt	Nein
Duplikate erlaubt	Ja
Primärschlüssel	Nein
Sonst	-
Beispielwerte	'img/katze.jpg' ...

Tabelle 7: Photos.created_at

created_at	vom System automatisch erstellter Timestamp beim upload des Fotos
Wertebereich	Datum & Zeit
Werte-Einschränkung	Nein
Wertevergabe	System, bei Erstellung des Accounts
Default-Wert	Ja, sysdate
NULL-Wert-erlaubt	Ja
Duplikate erlaubt	Ja
Primärschlüssel	Nein
Sonst	-
Beispielwerte	'11.11.2021 23:59:59' ...

4.2.3 Tags

Jedes Foto kann verschiedenen Tags zugeordnet werden.

- Tags sind Einzigartig
- beim erstellen des Tags wird ein Timestamp erstellt

Tabelle 8: Tags.id

id	Eindeutige ID zur internen Identifizierung der Fotos
Wertebereich	Ganze Zahlen
Werte-Einschränkung	Nein
Wertevergabe	System / Auto_Increment
Default-Wert	Nein
NULL-Wert-erlaubt	Nein
Duplikate erlaubt	Nein
Primärschlüssel	Ja
Sonst	automatisch vom System verwaltet über: GENERATED ALWAYS AS IDENTITY
Beispielwerte	1, 2, 3, ...

Tabelle 9: Tags.tag_name

tag_name	Bezeichner für die verschiedenen Tags der Fotos, vorerst keine Beschränkungen
Wertebereich	Text
Werte-Einschränkung	Nein
Wertevergabe	Benutzereingabe
Default-Wert	Nein
NULL-Wert-erlaubt	Nein
Duplikate erlaubt	Nein
Primärschlüssel	Nein
Sonst	-
Beispielwerte	'cat', 'drachenlordImKnast', 'WM2020' ...

Tabelle 10: Tags.created_at

created_at	vom System automatisch erstellter Timestamp bei Erstellung des Tags
Wertebereich	Datum & Zeit
Werte-Einschränkung	Nein
Wertevergabe	System, bei Erstellung des Accounts
Default-Wert	Ja, sysdate
NULL-Wert-erlaubt	Ja
Duplikate erlaubt	Ja
Primärschlüssel	Nein
Sonst	-
Beispielwerte	'11.11.2021 23:59:59' ...

4.3 Beziehungen

Aus der Analyse von Instagram ergeben sich die nachfolgend beschriebenen Beziehungen für unser Projekt SEISMIC.

4.3.1 posten

Ist eine Binäre Beziehung zwischen Users und Photos.

- Jeder User kann ein oder mehrere Fotos posten
- Jedes Foto wird von genau einem User gepostet

Tabelle 11: Beziehung posten

Beteiligung		Kardinalität
Users	0	N
	Jeder User muss kein Foto posten	Jeder User kann mehrere Fotos posten
Photos	1	1
	Jedes Foto muss von einem User gepostet werden	Jedes Foto wird von höchstens einem User gepostet

4.3.2 liken

Ist eine Binäre Beziehung zwischen Users und Photos.

- Jeder User kann ein oder mehrere Fotos liken
- Jedes Photo kann von mehreren Usern geliked werden

Tabelle 12: Beziehung liken

Beteiligung		Kardinalität
Users	0	N
	Jeder User muss kein Foto liken	Jeder User kann mehrere Fotos liken
Photos	0	M

Beteiligung	Kardinalität
Jedes Foto muss nicht von einem User geliked werden	Jedes Foto kann von mehreren User geliked werden

4.3.3 comments

Ist eine Binäre Beziehung zwischen Users und Photos.

- Jeder User kann ein oder mehrere Fotos kommentieren
- Jedes Photo kann von mehreren Usern kommentiert werden

Tabelle 13: Beziehung comments

	Beteiligung	Kardinalität
Users	0	N
	Jeder User muss kein Foto kommentieren	Jeder User kann mehrere Fotos kommentieren
Photos	0	M
	Jedes Foto muss nicht von einem User kommentiert werden	Jedes Foto kann von mehreren Usern kommentiert werden

4.3.4 photo_tags

Ist eine Binäre Beziehung zwischen Tags und Photos.

- Jeder Tag muss einem oder mehreren Fotos zugeordnet sein
- Jedes Photo kann mit mehreren Tags gekennzeichnet werden

Tabelle 14: Beziehung photo_tags

	Beteiligung	Kardinalität
Tags	1	N
	Jeder Tag muss einem Foto zugeordnet sein	Jeder Tag kann mehreren Fotos zugeordnet sein
Photos	0	M

Beteiligung	Kardinalität
Jedes Foto muss nicht mit einem Tag gekennzeichnet werden	Jedes Foto kann mit mehreren Tags gekennzeichnet werden

4.3.5 follows

Ist eine Unäre Beziehung von Users

- Jeder Follower kann mehreren Followees folgen
- Jeder Followee kann mehrere Follower besitzen

Tabelle 15: Beziehung follows

Beteiligung		Kardinalität
Users	0	N
	Jeder follower muss keinem followee folgen	Jeder follower kann mehreren followees folgen
Users	0	M
	Jeder followee muss keinen follower besitzen	Jeder followee kann von mehreren followern gefolgt werden

4.4 Typische Operationen

- Benutzer login
- Anzeige Kommentare eines Fotos
- Ausgabe aller Followees des Benutzers
- Ausgabe der Followers des Benutzers
- Ausgabe aller likes eines Bildes

4.5 weitere relevante Informationen

Im nachfolgend sind weitere relevanten Informationen aufgelistet, die bei der Erstellung der Datenbank und Auswahl des Datenbanksystems zu beachten sind.



4.5.1 Datenmengen

Über das Datenvolumen der Datenbank lässt sich an dieser Stelle noch keine ganz genaue Aussage treffen. Unter anderem fehlen Informationen, in welchem Format und Auflösung die Bilder später gespeichert werden und ob auch noch Videos und andere Metadaten hinzu kommen werden.

Es ist allerdings davon auszugehen das die Datenbank schnell mehrere TB groß werden kann.

4.5.2 Gesetzliche Bestimmungen

Da wir uns im Internet bewegen sind die jeweiligen Rechtsbestimmungen aller Länder zu beachten. Besonders in Hinsicht auf Datenschutz und möglicher (Anti-) Zensurgesetze.

Wir werden uns in der ersten Phasen nur auf den europäischen Markt konzentrieren und nehmen die Europäische Datenschutz-Grundverordnung sowie nationale Regelungen als mindestmaß für unser Konzept.

Zudem sind strafrechtliche Bestimmungen der einzelnen Länder zu beachten.

4.5.3 Sicherung der Datenbank

Wir können uns keine Downtimes der Datenbank erlauben, da wir uns auf einem globalen Markt bewegen. Die Datenbank wird als micro-service in einem orchestrierten Container-cluster laufen. Sie wird durch Replizierung gegen einen Ausfall gesichert und jeweils asynchron auf externe Sicherungshubs verteilt.

4.5.4 Zeichensatz

Als Zeichensatz werden wir UTF-8 einsetzen.

5 Entity Relationship-Modell

5.1 Gesamtdarstellung

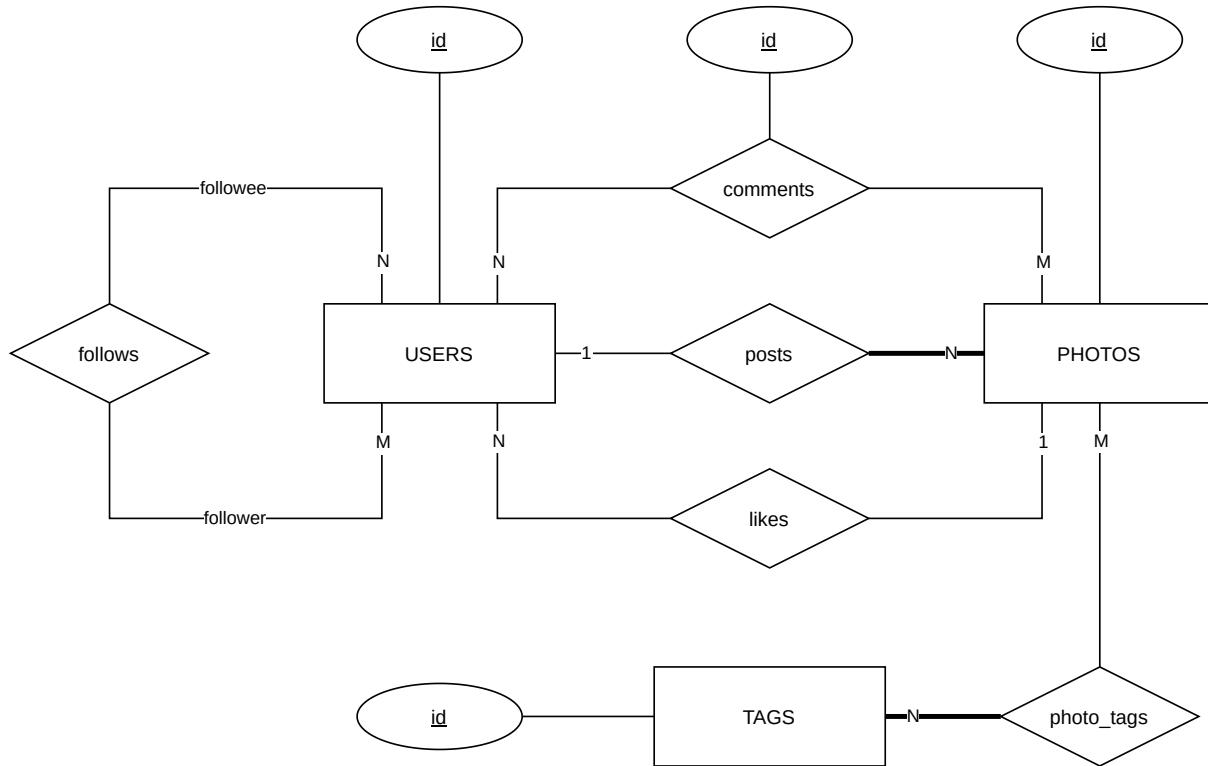


Abbildung 5: Gesamtübersicht SEISMIC

5.2 Detaildarstellung

Jede Entität wird nachfolgend detailliert mit den dazugehörigen Attributen dargestellt.

5.2.1 USERS

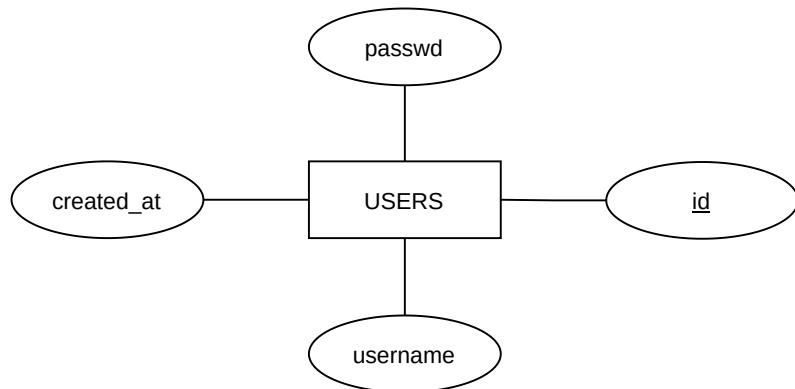


Abbildung 6: SEISMIC - Entität USERS

5.2.2 PHOTOS

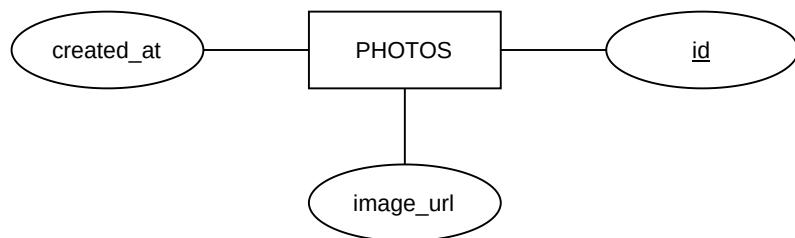


Abbildung 7: SEISMIC - Entität PHOTOS

5.2.3 TAGS

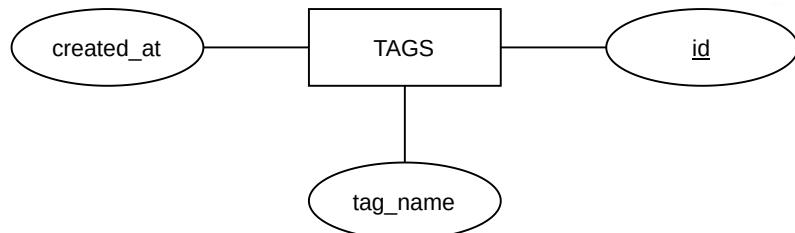


Abbildung 8: SEISMIC - Entität TAGS



6 Relationenmodell

Users ([id](#) , passwd NN , username NN , created_at)

Photos ([id](#) , [image_url](#) NN , [created_at](#) , [user_id](#) NN)

FK: Photos.user_id => PK: Users.id

Tags ([id](#) , tag_name NN , created_at)

Comments ([id](#), [comment_text](#) NN, [user_id](#) NN, [photo_id](#) NN, [created_at](#))

FK: Comments.user_id => PK: Users.id

FK: Comments.photo_id => PK: Photos.id

Likes (user_id , photo_id , created_at)

FK: Likes.user_id => PK: Users.id

FK: Likes.photo_id => PK: Photos.id

Photo_tags (photo_id , tag_id)

FK: Photo_tags.photo_id => PK: Photos.id

FK: Photo_tags.tag_id => PK: Tags.id

Follows (follower_id , followee_id)

FK: Follows.follower_id => PK: Users.id

FK: Follows.followee_id => PK: Users.id

Abbildung 9: SEISMIC - Relationenmodell

7 Normalisierung

An dieser Stelle sollen nur die ersten 3 Normalformen betrachtet werden.

1. Normalform Alle Attribute weisen nur atomare Werte auf.
2. Normalform: Da die 1. Normalform erfüllt ist und alle Primärschlüssel aus künstlichen Schlüsseln gebildet werden, liegt die 2. Normalform vor.
3. Normalform: Die 2. Normalform liegt vor und es existieren keine Nichtschlüsselattribute, die transitiv vom Primärschlüssel abhängen. Somit ist die 3. Normalform ebenfalls gegeben.



8 Datenbankobjekte

Zum Erstellen der Datenbankobjekte wird ein Benutzer benötigt, der über ausreichende Rechte verfügt.

8.1 Tabellen

- löschen bereits vorhandener Tabellen
- Erstellung aller vorher aufgelisteten Tabellen von SEISMIK

Script: [setup/db_table_install.sql](#)

8.2 Dummy-Datensätze

- entfernen aller vorhandenen Datensätzen in allen Tabellen
- einfügen von Dummy-Datensätze in jede Tabelle
- speichern (COMMIT) der Transaktion

Skript: [setup/insert_data.sql](#)

9 Datenmanipulation

Die Manipulation der Daten erfolgt durch interaktive Eingabe der Werte.

9.1 Foto posten

Es wird ein neues Foto (derzeit noch link) des angemeldeten Benutzers (Login) in der Tabelle `photos` gespeichert. Der Benutzer wird zum Datenupload (derzeit noch url) des Fotos aufgefordert. Der Primärschlüssel wird vom DBMS selbst verwaltet. Am Ende werden die Änderungen gespeichert (COMMIT).

Skript: [update/post.sql](#)

9.2 Foto löschen

Es wird ein ausgewähltes Foto (derzeit noch link) des angemeldeten Benutzers (Login) aus der Tabelle `photos` gelöscht. Der Benutzer wählt die ID des Fotos das er löschen möchte aus. Dann wird dass Foto in der Tabelle `photos` gelöscht. Anschließend werden in allen Beziehungen/Tabellen (`likes`, `comments`, `photo_tags`) die jeweiligen zugehörigen Datensätze gelöscht. Am Ende wird die Transaktion abgeschlossen (COMMIT).

Skript: [update/post.sql](#)

9.3 Liken

Es werden dem Benutzer alle Fotos die er noch nicht geliked hat angezeigt. Der eingeloggte Benutzer wählt eine Foto ID aus (`photos`). Dann wird in der Tabelle `likes` die ID des eingeloggten Benutzers sowie die ID des gewählten Fotos eingetragen. Am Ende werden die Änderungen gespeichert (COMMIT). Die beiden gesetzten ID sind jeweils Fremdschlüssel mit Referenz zu `users` und `photos`.

Skript: [update/like.sql](#)

9.4 Unlikken

Dem Benutzer werden alle Fotos angezeigt die er geliked hat. Er wird aufgefordert eine ID eines Fotos das er unliken will zu wählen. Es werden dann in der Tabelle `likes` die beiden Fremdschlüssel `user_id` und `photo_id` gelöscht. Diese Transaktion wird anschließend beendet (COMMIT). Beide Werte sind jeweils Fremdschlüssel mit Referenz auf `users` und `photos`.

Skript: [update/unlike.sql](#)



9.5 Folgen

Am Anfang erhält der Benutzer eine Übersicht derjenigen Nutzer denen er Folgen kann (der Benutzer und bereits gefolgte Benutzer werden nicht angezeigt). Er wählt dann die `id` des Benutzers dem er folgen will. Es werden danach in der Tabelle `follows` jeweils die ID des Benutzers `follower_id` sowie die ID des ausgewählten Benutzers `followee_id` eingetragen. Die Transaktion wird anschließend beendet (COMMIT). Jede ID ist jeweils ein Fremdschlüssel mit Referenz zu `users`.

Script: [update/follow.sql](#)

9.6 Unfollow

Der Benutzer wählt aus einer Liste der bereits gefolgten Benutzern die `followee_id` aus. Anschließend wird in der Tabelle `follows` der Datensatz mit der ausgewählten `followee_id` und der ID des Benutzers `follower_id` gelöscht. Am Ende werden die Änderungen gespeichert (COMMIT). Die beiden gesetzten ID sind jeweils Fremdschlüssel mit einer Referenz zu `users`.

Script: [update/unfollow.sql](#)



10 Datenabfrage

Eine Auflistung der derzeitigen Datenabfragen sollen nachfolgend vorgestellt werden.

10.1 Foto des Users anzeigen

Welche Fotos hat der User bereits gepostet, mit Gesamtanzahl der Likes und Comments je Foto.

Script: [read/userphotos.sql](#)

10.2 Kommentare eines Fotos

Der User erhält eine Auswahl aller Fotos mit Kommentaren > 0 und != NULL. Er wählt dann ein Foto via ID aus. Es wird dem User nach abgeschlossener Eingabe eine Tabelle mit allen Kommentaren zu diesem Foto präsentiert, absteigend sortiert nach gepostetem Datum und wer diesen post erstellt hat.

Script: [read/neusteK.sql](#)

10.3 Top 5 liked Fotos

Eine Übersicht der Top 5 gelikten Fotos (die obersten 5 Zeilen [keine Mehrfachplazierungen])

Script: [read/top5.sql](#)

11 Menugestaltung

Die Verwaltung der Datenbankobjekte, die Datenmanipulation und die Datenabfrage kann menugesteuert ausgeführt werden. Die Menugestaltung wird mit den Möglichkeiten, die `sqlcl` bietet, realisiert.

11.1 Ordner- und Datei-Struktur

Die Menugestaltung basiert auf der nachfolgend dargestellten Struktur.

```
⌚ AxBook ➜ ~/Doc/IT3/projects_IT33/seismik/scripts on ↵ main [📝 ? ]
↳ ls --tree

└── read/
    ├── menu.sql
    ├── neusteK.sql
    ├── top5.sql
    └── userphotos.sql

└── setup/
    ├── db_table_install.sql
    ├── insert_data.sql
    └── menu.sql

└── update/
    ├── delete.sql
    ├── follow.sql
    ├── like.sql
    ├── menu.sql
    ├── post.sql
    ├── unfollow.sql
    └── unlike.sql

    ├── asciiArt
    ├── menu.sql
    ├── quit.sql
    └── seismic.sh

    ├── start.sql
    └── user.sql
```

Abbildung 10: Ordner-/Dateistruktur

11.2 Menu-Übersicht

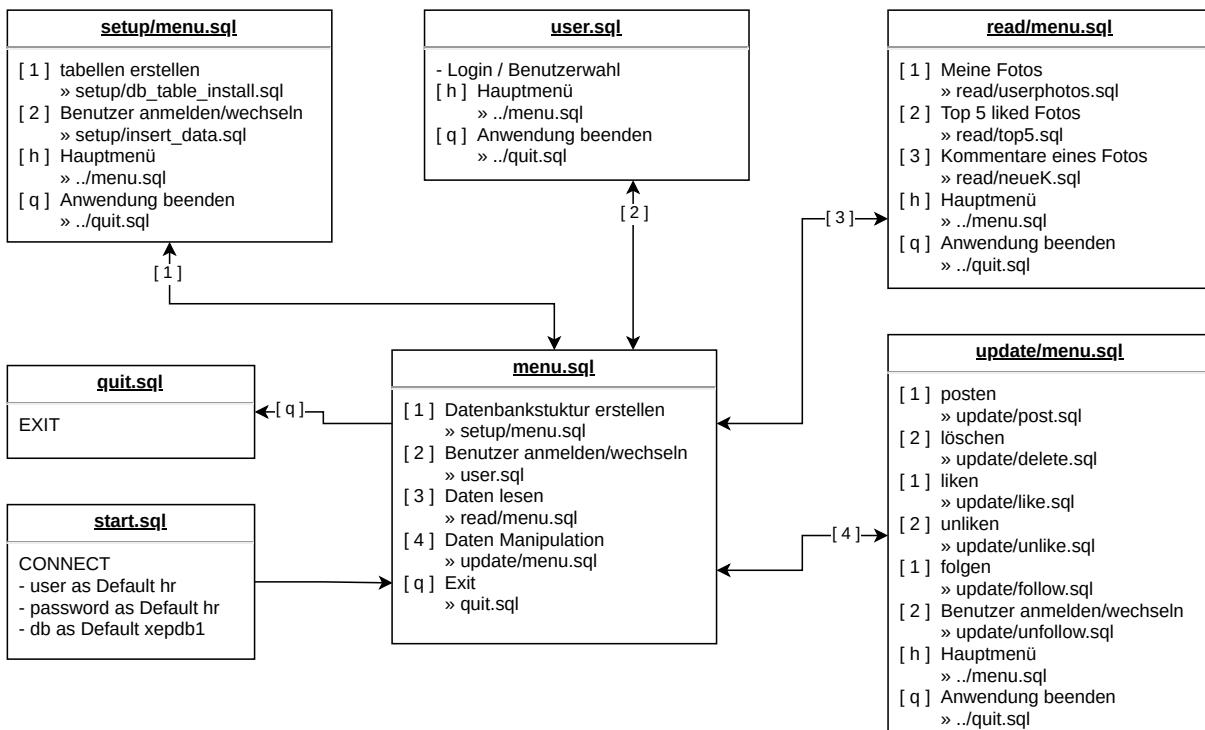


Abbildung 11: Menugestaltung

Zum Starten wird das shellskript: `seismik.sh` ausgeführt. Es sorgt dafür, dass das Programm `sqlcl` ohne eine Anmeldung an die Datenbank gestartet wird. Nach erfolgreichem Start der Anwendung wird das Skript: `start.sql` ausgeführt.

Im Skript: `start.sql` werden die folgenden Substitutionsvariablen generiert:

- `usr` => Name des Datenbankbenutzers (Default: `hr`)
- `passwd` => Passwort des Datenbankbenutzers (Default: `hr`)
- `db` => Datenbankname der Datenbank (Default: `XEPDB1`)

Statt dem vorgeschlagenen Default-Wert kann der Benutzer interaktiv einen neuen Wert vergeben. Der Benutzer wird danach mit der Datenbank verbunden und anschließend wird das Hauptmenü durch Aufrufen des Skriptes: `menu.sql` präsentiert.

11.2.1 Hauptmenü



Abbildung 12: Hauptmenü

Im Hauptmenü kann zwischen folgenden Menupunkten gewählt werden, deren Auswahl in den jeweiligen eckigen Klammerpaar angegeben wird:

Tabelle 16: Hauptmenü

Menupunkt	Bemerkung	Skript
Datenbankstruktur bearbeiten	Untermenu: Datenbankstruktur	setup/menu.sql
Benutzerauswahl / -wechsel	Ruft das Script Login auf	user.sql
Lesen spezifischer Datensätze	Untermenu: Daten lesen	read/menu.sql
ändern von Datensätzen	Untermenu: Daten ändern	update/menu.sql

11.2.2 Untermenü: Datenbankstruktur



Abbildung 13: Menu Datenbankstruktur

Im Untermenü: Datenbankstruktur kann zwischen folgenden Menupunkten gewählt werden, deren Auswahl in den jeweiligen eckigen Klammerpaar angegeben wird:

Tabelle 17: Untermenü - Datenbankstruktur

Menupunkt	Bemerkung	Skript
Datenbankstruktur erstellen	Alle Tabellen werden zuerst gelöscht und danach erstellt. Anschließend wird in das Untermenü zurückgekehrt.	db_table_install.sql
Beispieldatensätze einfügen	Die Tabellen werden am Anfang geleert. Anschließend werden die Dummy-Datensätze eingefügt und die Transaktion wird mit einem COMMIT abgeschlossen.	insert_data.sql
zurück zum Hauptmenu	Ruft das Hauptmenu auf.	./menu.sql
Anwendung beenden	Verlässt sqlcl.	./quit.sql

11.2.3 Script: Benutzerauswahl / -wechsel

```
© d.schwarz          © Axklen
.....
SEISMIC
Login
.....
Benutzer ID      BENUTZER      BILDER
-----
1 Kenton_Kirlin      4
2 Axklen            1
3 Poloi.al          3
4 knastjunge        2
5 falsche-Wurst     1
7 kleinbein         1
8 Watank233          3
Wählen Sie eine Benutzer ID => 
```

Abbildung 14: Benutzerauswahl / -wechsel

In diesem Script bekommt der Benutzer eine Übersicht der zur Verfügung stehenden Benutzeraccounts. Der Benutzer wird aufgefordert eine ID eines Benutzeraccounts zu wählen um sich dann mit diesem Account "einzuloggen".

Script: [user.sql](#)

11.2.4 Untermenü: Lesen spezifischer Datensätze



Abbildung 15: Datensätze Lesen

Tabelle 18: Untermenü - Datensätze Lesen

Menupunkt	Bemerkung	Skript
Meine Fotos	Der Benutzer wählt eine Foto ID aus. Ihm wird dann das Foto mit Details angezeigt.	userphotos.sql
Top 5 Liked Fotos	Dem Benutzer wird eine Liste mit den Top 5 gelikten Fotos angezeigt.	top5.sql
Kommentare eines Fotos	Dem Benutzer wird eine Liste aller Fotos mit Kommentaren angezeigt. Er kann eines wählen und bekommt dann alle Kommentare zu diesem Foto angezeigt.	neusteK.sql
zurück	kehrt in das Untermenü zurück.	menu.sql

Menupunkt	Bemerkung	Skript
zurück zum Hauptmenu	Ruft das Hauptmenü auf.	.../menu.sql
Anwendung beenden	Verlässt sqlcl.	.../quit.sql

11.2.5 Untermenü: ändern von Datensätzen

```
© d.schwarz          © Axklen
.....
SEISMIC
.....
👤 Kenton_Kirlin  📄 4  ↗ 2  ↙ 2  💬 1
.....
[ 1 ]  🖼 posten
[ 2 ]  ✖ Löschen
[ 3 ]  ❤ liken
[ 4 ]  ❤ unliken
[ 5 ]  ↗ folgen
[ 6 ]  ⚡ unfollow
.....
[ h ]  🖼 ZURÜCK zum Hauptmenü
[ q ]  ✖ Anwendung BEENDEN
.....
Ihre Auswahl => [
```

Abbildung 16: Datensätze ändern

Tabelle 19: Untermenü - Datensätze ändern

Menupunkt	Bemerkung	Skript
posten	Der Benutzer wird zur Eingabe der url eines Fotos, das er posten will, aufgefordert.	posten.sql
löschen	Der Benutzer wird zur Auswahl einer ID eines Fotos, das er löschen will aufgefordert.	delete.sql
liken	Der Benutzer wird zur Eingabe einer Foto ID	like.sql



Menupunkt	Bemerkung	Skript
unlike	aufgefordert. Anschließend wird das Foto geliked	
folgen	Der Benutzer wird zur Eingabe einer Foto ID aufgefordert. Anschließend wird das Foto nicht mehr vom Benutzer geliked.	unlike.sql
unfollow	Der Benutzer wird zur Eingabe der Benutzer ID des Benutzers dem er folgen will aufgefordert.	follow.sql
zurück	Der Benutzer wird zur Eingabe der Benutzer ID des Benutzers dem er nicht mehr folgen will aufgefordert.	unfollow.sql
zurück zum Hauptmenu	kehrt in das Untermenü zurück.	menu.sql
Anwendung beenden	Ruft das Hauptmenü auf.	../menu.sql
	Verlässt sqlcl.	../quit.sql



12 Quellenverzeichnis

12.1 Abbildung 1-4

<https://instagram.com> (Seite 7-10)

12.2 Logo

<https://logopixel.com/>