Universität Salzburg — Übungsleiter: Schäler/Kwitt/Resmerita

Datenbanken 1 – PS (501.073)

Projektabgabe – Teil 2 — Abzugeben bis **19.05.2021**

In folgender Angabe finden Sie die Lösung (ER-Diagramm) zu **Teil 1** des PS Projektes. Ihre Aufgabe besteht nun darin, dieses ER-Diagramm in **SQL** umzusetzen (beispielsweise indem Sie das Diagramm in ein entsprechendes Schema übersetzen und dies als Vorlage zur Erstellung der SQL Statements benutzen). Konkret bedeutet dies, die Tabellen (mittels `CREATE TABLE` Anweisungen, etc.) zu generieren.

Erstellen Sie dazu eine Datei `create.sql`, welche **alle** Anweisungen enthält. *Bitte achten Sie auf den korrekten Dateinamen*. Die `create.sql` Datei ist die einzige Datei die abgegeben werden soll.

Hinweis: Die `create.sql` Datei **muss** von PostgreSQL geladen werden können. Zum Testen könnten Sie auf Ihrem eigenen System beispielsweise eine Datenbank `mydb` anlegen und dann Ihre `create.sql` Datei laden:

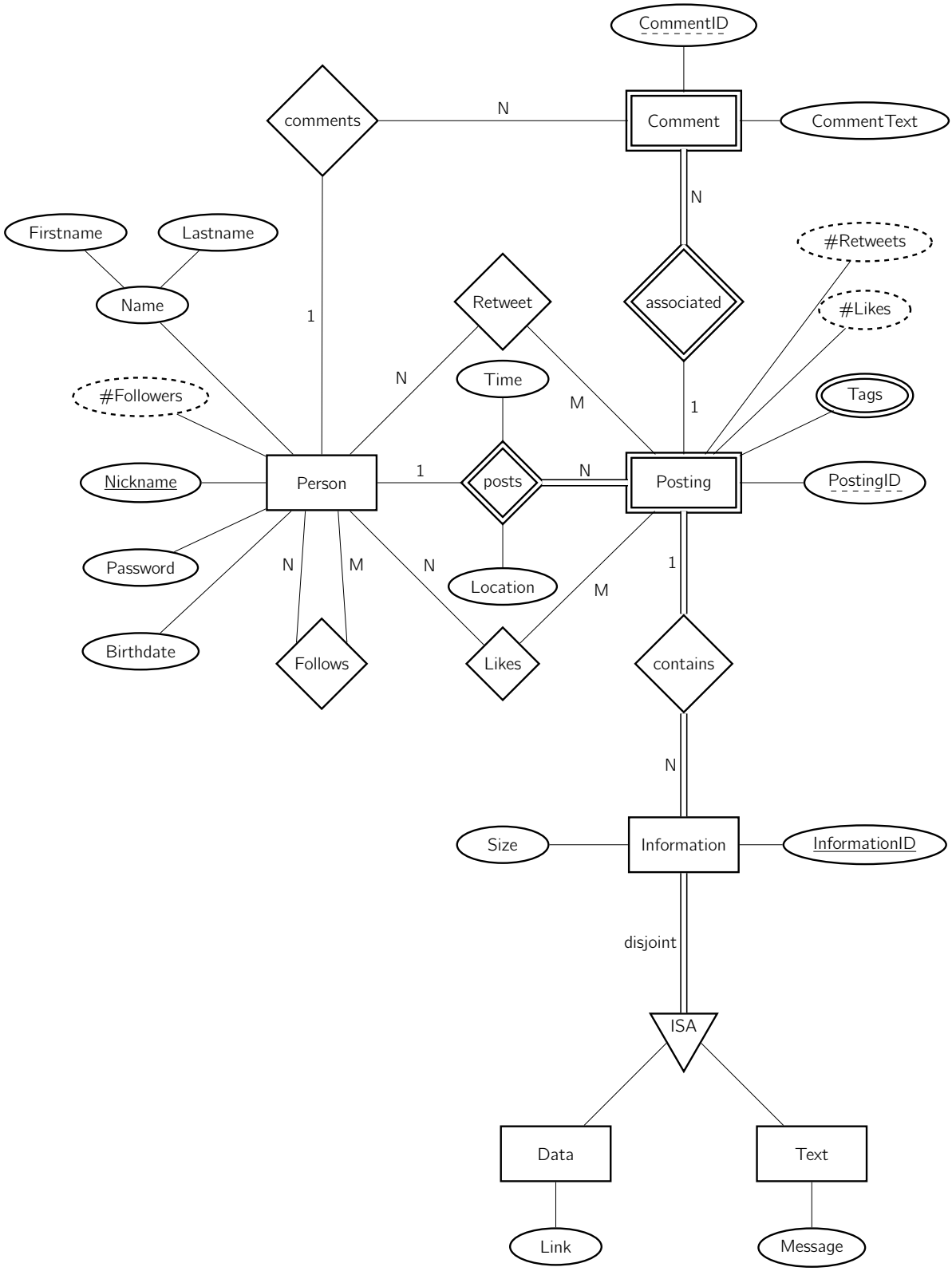
```
createdb mydb
psql -d mydb -f create.sql
```

Wir werden die Abgaben **automatisch** überprüfen. Zunächst muss Ihre `create.sql` Datei unter PostgreSQL ausführbar sein, sonst wird Ihre Abgabe nicht weiter überprüft. Es gibt jeweils **0.6 Punkte** (also **3 Punkte** gesamt) für jede der folgenden Kategorien:

- jede Tabelle existiert mit richtigem Namen
- jedes Attribut existiert mit richtigem Namen und Datentyp
- jeder Primärschlüssel existiert
- jeder Fremdschlüssel existiert
- jede Beschränkung der erlaubten Datenwerte wurde angelegt

In folgender Tabelle finden Sie die Namen der Entitäten und Attribute (bitte genau einhalten), sowie die zu verwendenden Datentypen und Einschränkungen:

Entität/Attribut	Datentyp	Einschränkungen
Nickname	TEXT	
Firstname	TEXT	
Lastname	TEXT	
Password	TEXT	NOT NULL & länger als 0 Zeichen lang
Birthdate	TIMESTAMP	
PostingID	BIGINT	
CommentID	BIGINT	
CommentText	TEXT	
Time	TIMESTAMP	NOT NULL
Location	TEXT	
InformationID	BIGINT	
Size	BIGINT	
Link	TEXT	NOT NULL
Message	TEXT	NOT NULL & länger als 0 Zeichen lang
Tag	TEXT	NOT NULL & länger als 0 Zeichen lang


CommentID
comments
N
Comment
CommentText
N
Firstname
Lastname
#Retweets
Name
1
Retweet
associated
#Likes
N
Time
M
1
Tags
#Followers
Person
1
posts
N
Posting
PostingID
Nickname
Password
N
M
N
Location
M
1
Birthdate
Follows
Likes
contains
N
Size
Information
InformationID
disjoint
ISA
Data
Text
Link
Message

Wichtige Hinweise

Bitte folgende Konvention beachten: Attribute die in einer Relation Fremdschlüsselattribute sind, sollen den **gleichen** Namen erhalten wie in der Relation wo diese Attribute Primärschlüssel sind. Hier ein Beispiel:

```
CREATE TABLE Foo (
  A TEXT,
  B TEXT,
  CONSTRAINT foo_pk PRIMARY KEY (A)
);

CREATE TABLE Bar (
  A TEXT,
  ID BIGINT,
  CONSTRAINT bar_fk FOREIGN KEY(A) REFERENCES Foo(A),
  CONSTRAINT bar_pk PRIMARY KEY(A, ID)
);
```

In diesem Beispiel ist das Attribut `A` in `Bar` ein Fremdschlüsselattribut zu `Foo(A)`. Laut der oben genannten Konvention, sind also beide Attribute mit `A` benannt.

Für das mehrwertige Attribut `Tags` (siehe ER-Diagramm) wird ein eigener Entitätstyp erstellt; `Tag` (Datentyp siehe Tabelle) stellt darin das Attribut dar. Um auf Zeichenlänge > 0 zu überprüfen, benutzen Sie die Funktion `char_length`.

Bei Beziehungen die als Entitäten abgebildet werden, also `Follows`, `Likes` und `Retweet` (also alle N:M), ist folgende Namensgebung zu beachten (die Bedeutung der angeführten Namen ergibt sich aus den entsprechenden Fremdschlüsselbeziehungen): Bei `Follows` soll `FollowerNickname` und `FolloweeNickname` verwendet werden. Bei `Likes` soll `PostNickname` und `LikeeNickname` verwendet werden. Bei `Retweet` soll `PostNickname` und `RetweetNickname` verwendet werden.

Bei der Entität `Comment` soll `CommenterNickname` verwendet werden. Die Namen der anderen Attribute ergeben sich aus der anfänglich genannten Konvention und den entsprechenden Fremdschlüsselbeziehungen.

Abgabedetails

Abgegeben wird die Datei `create.sql`.