

SQL (Part 1)

January 29, 2019

Data Science CSCI 1951A

Brown University

Instructor: Ellie Pavlick

HTAs: Wennie Zhang, Maulik Dang, Gurnaaz Kaur

Announcements

- Waitlist—should be mostly set now, unless people drop
- Watch website for links to readings/slides/resources. I will post slides shortly after lecture.
- iClicker-for-credit will begin on Tuesday
- Final project groups—thread on Piazza, elsewhere?

Announcements

- This lecture should cover most of what you need for HW1
 - A bit of a laundry list—apologies in advance :)
- Nested queries/optimization will be on Tuesday
- For more, this is a very useful inventory of all SQL commands/usage: <https://www.w3schools.com/sql/>

Last Time...

- Conceptual Modeling, Database Design, ER Diagrams
- Relational Algebra (more about this on Tuesday)
- And lingering questions...?

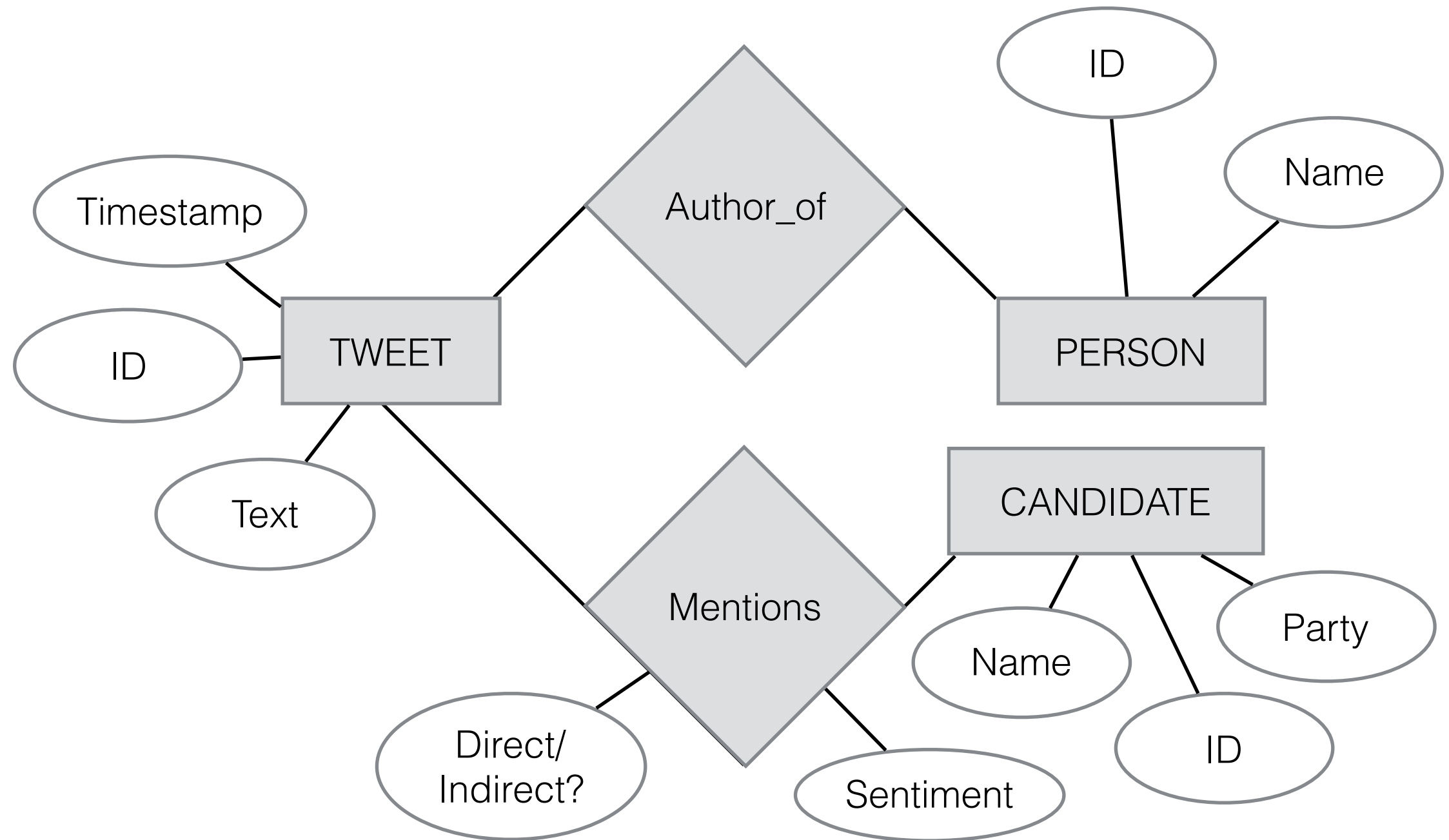
SQL

SQL

- Data Definition Language (DDL): Defining data types and Relation Schemas (intensions!)
- Data Manipulation and Query Language (DML):
 - Populating/updating data bases (extensions!)
 - Querying data bases

Creating and Manipulating Tables

Creating and Manipulating Tables



Data Types

Data Types

- Numeric: INT, FLOAT, REAL, DOUBLE

Data Types

- Numeric: INT, FLOAT, REAL, DOUBLE
- Character Strings: CHAR(n), VARCHAR(n), CLOB(size)

Data Types

- Numeric: INT, FLOAT, REAL, DOUBLE
- Character Strings: CHAR(n), VARCHAR(n), CLOB(size)
 - CLOB(2MB) for large objects e.g. documents/web pages

Data Types

- Numeric: INT, FLOAT, REAL, DOUBLE
- Character Strings: CHAR(n), VARCHAR(n), CLOB(size)
 - CLOB(2MB) for large objects e.g. documents/web pages
- Bit Strings: BIT(n), BIT VARYING(n), BLOB

Data Types

- Numeric: INT, FLOAT, REAL, DOUBLE
- Character Strings: CHAR(n), VARCHAR(n), CLOB(size)
 - CLOB(2MB) for large objects e.g. documents/web pages
- Bit Strings: BIT(n), BIT VARYING(n), BLOB
 - BLOB(20MB) e.g. for images

Data Types

- Numeric: INT, FLOAT, REAL, DOUBLE
- Character Strings: CHAR(n), VARCHAR(n), CLOB(size)
 - CLOB(2MB) for large objects e.g. documents/web pages
- Bit Strings: BIT(n), BIT VARYING(n), BLOB
 - BLOB(20MB) e.g. for images
- Boolean

Data Types

- Numeric: INT, FLOAT, REAL, DOUBLE
- Character Strings: CHAR(n), VARCHAR(n), CLOB(size)
 - CLOB(2MB) for large objects e.g. documents/web pages
- Bit Strings: BIT(n), BIT VARYING(n), BLOB
 - BLOB(20MB) e.g. for images
- Boolean
- Dates: DATE, TIME, TIMESTAMP, TIME WITH TIME ZONE

Creating Tables

Creating Tables

TWEET: <ID, Time, Text>

Creating Tables

TWEET: <ID, Time, Text>

```
create table TWEET (  
  ID INT,  
  Time TIMESTAMP,  
  Text ???  
);
```

Creating Tables

TWEET: <ID, Time, Text>

```
create table TWEET (  
  ID INT,  
  Time TIMESTAMP,  
  Text ???  
);
```

CHAR(n), VARCHAR(n), CLOB(size) ??

Creating Tables

TWEET: <ID, Time, Text>

```
create table TWEET (  
  ID INT,  
  Time TIMESTAMP,  
  Text VARCHAR(140)  
);
```

Creating Tables

PERSON: <Handle, Name>

```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000)  
);
```

Creating Tables

PERSON: <Handle, Name, ProfilePic>

```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
  ProfilePic ???  
);
```

Creating Tables

PERSON: <Handle, Name, ProfilePic>

```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
  ProfilePic BLOB(20MB),  
);
```


Creating Tables

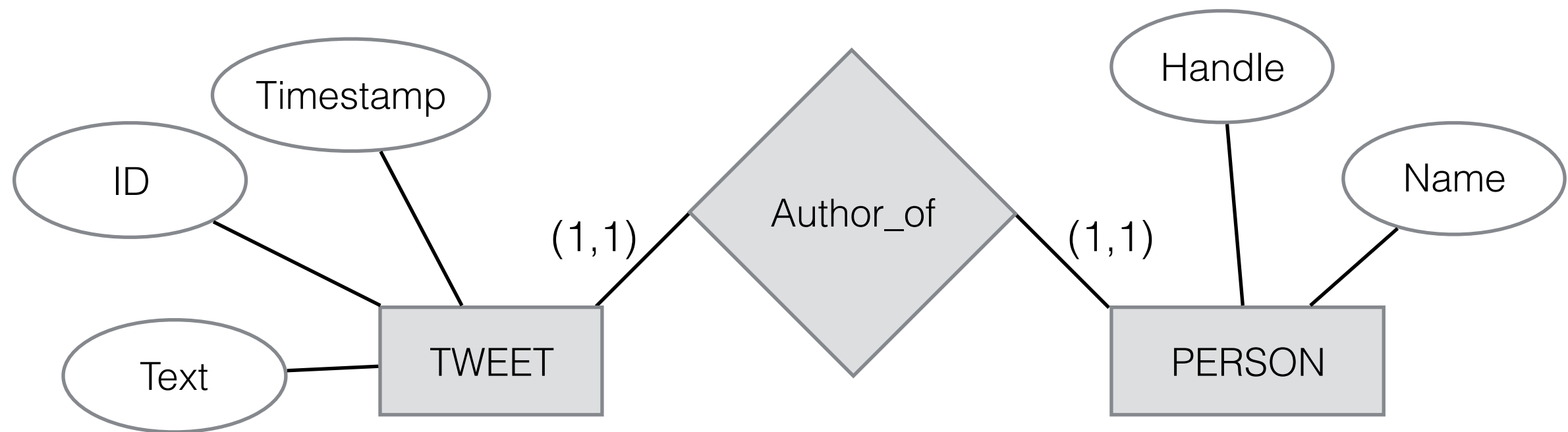
PERSON: <Handle, Name, ProfilePic, ProfilePage>

```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
  ProfilePic BLOB(20MB),  
  ProfilePage ???  
);
```

Creating Tables

PERSON: <Handle, Name, ProfilePic, ProfilePage>

```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
  ProfilePic BLOB(20MB),  
  ProfilePage CLOB(20MB)  
);
```

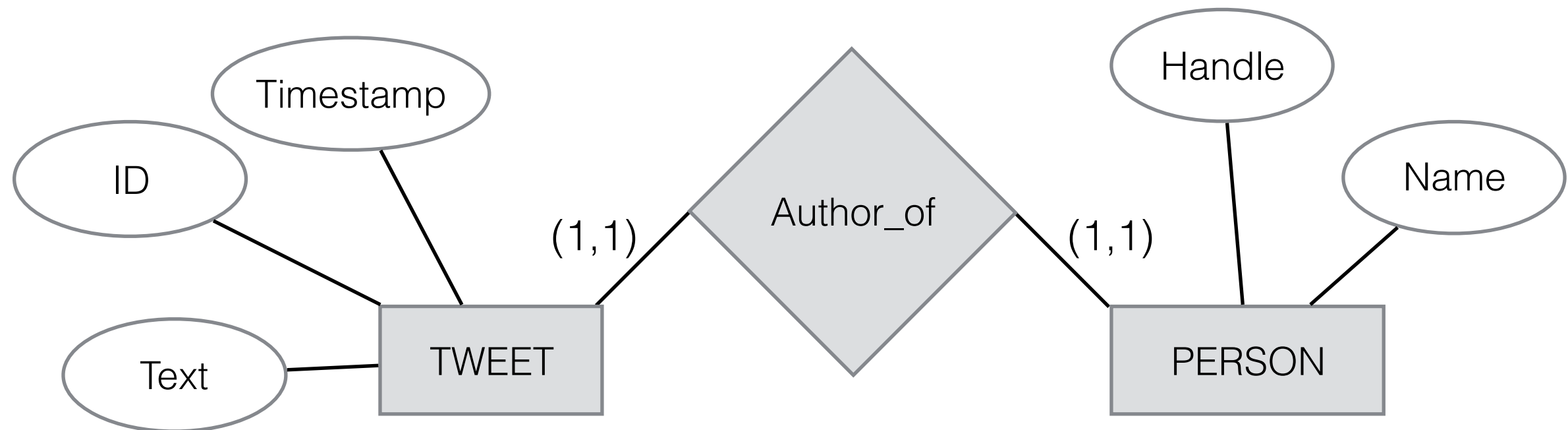


Clicker Question!

TWEET: <ID:INT, Time:TIMESTAMP, Text:VARCHAR(140)>

PERSON: <Handle:VARCHAR(100), Name:VARCHAR(1000)>

```
create table AUTHOR (  
    ???  
);
```



TWEET: <ID:INT, Time:TIMESTAMP, Text:VARCHAR(140)>

PERSON: <Handle:VARCHAR(100), Name:VARCHAR(1000)>

```

create table AUTHOR (
  Tweet INT,
  Person VARCHAR(100),
);
  
```

(a)

```

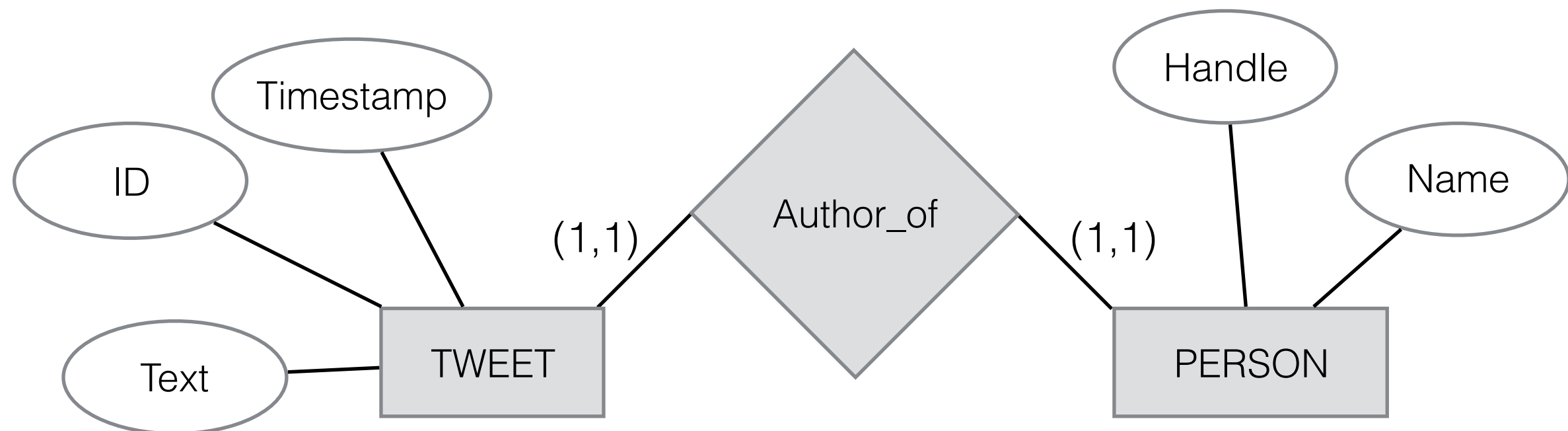
create table AUTHOR (
  Tweet INT,
  Person INT,
);
  
```

(b)

```

create table AUTHOR (
  Tweet INT,
  Person VARCHAR(1000),
);
  
```

(c)



TWEET: <ID:INT, Time:TIMESTAMP, Text:VARCHAR(140)>

PERSON: <Handle:VARCHAR(100), Name:VARCHAR(1000)>

```
create table AUTHOR (
  Tweet INT,
  Person VARCHAR(100),
);
```

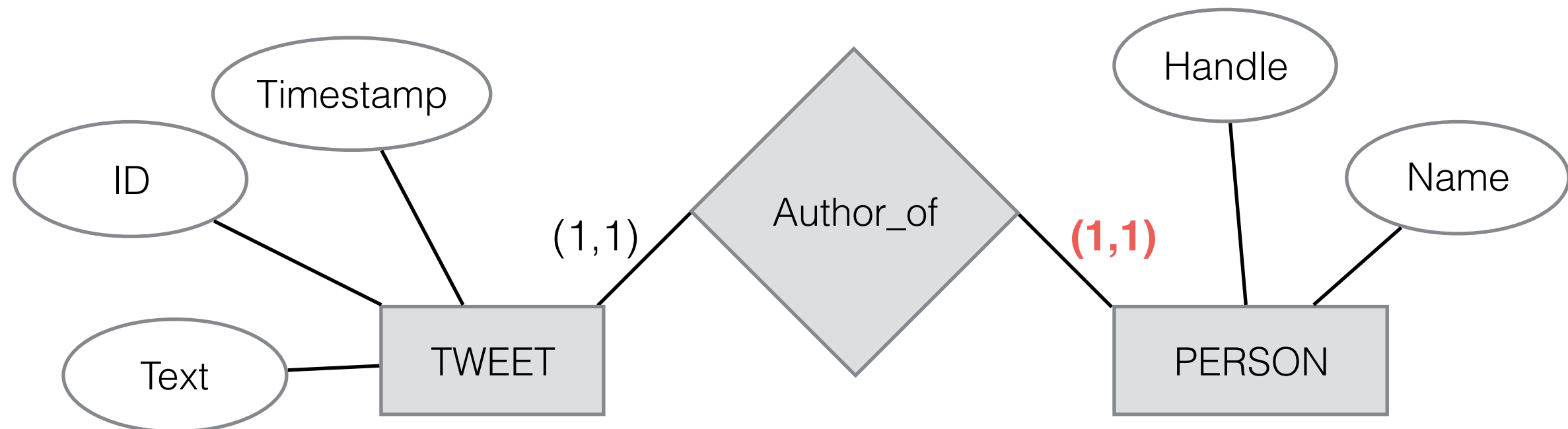
(a)

```
create table AUTHOR (
  Tweet INT,
  Person INT,
);
```

(b)

```
create table AUTHOR (
  Tweet INT,
  Person VARCHAR(1000),
);
```

(c)



TWEET: <ID:INT, Time:TIMESTAMP, Text:VARCHAR(140)>

PERSON: <Handle:VARCHAR(100), Name:VARCHAR(1000)>

```

create table AUTHOR (
  Tweet INT,
  Person VARCHAR(100),
);
  
```

(a)

```

create table AUTHOR (
  Tweet INT,
  Person INT,
);
  
```

Should use handle because
they will be unique.

```

create table AUTHOR (
  Tweet INT,
  Person VARCHAR(1000),
);
  
```

(c)

Inserting Tuples

Inserting Tuples

TWEET: <ID:INT, Time:TIMESTAMP, Text:VARCHAR(140)>

ID	Timestamp	Text

Inserting Tuples

TWEET: <ID:INT, Time:TIMESTAMP, Text:VARCHAR(140)>

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey

```
insert into TWEET values (  
    389472,  
    2019-01-01 12:34:56,  
    "hey");
```

```
create table TWEET (  
  ID INT,  
  Time TIMESTAMP,  
  Text VARCHAR(140)  
);
```

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
NULL	2019-01-01 12:34:57	lol

```
insert into TWEET(Timestamp, Text) values(  
  2019-01-01 12:34:57,  
  "lol");
```

```
create table TWEET (  
  ID INT,  
  Time TIMESTAMP,  
  Text VARCHAR(140)  
);
```

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
NULL	2019-01-01 12:34:57	lol

```
insert into TWEET(Timestamp, Text) values(  
  2019-01-01 12:34:57,  
  "lol");
```

```
create table TWEET (  
  ID INT DEFAULT 0,  
  Time TIMESTAMP,  
  Text VARCHAR(140)  
);
```

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
0	2019-01-01 12:34:57	lol

```
insert into TWEET(TimeStamp, Text) values(  
  2019-01-01 12:34:57,  
  "lol");
```

```
create table TWEET (  
  ID INT NOT NULL,  
  Time TIMESTAMP,  
  Text VARCHAR(140)  
);
```



ID	Timestamp
389472	2019-01-01 12:34:56

```
insert into TWEET (Timestamp, Text) values (  
  2019-01-01 12:34:57,  
  "lol");
```

Keys

Keys

TWEET: <ID, Time, Text>

```
create table TWEET (  
  ID INT,  
  Time TIMESTAMP,  
  Text VARCHAR(140),  
);
```

Keys

TWEET: <ID, Time, Text>

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP,  
  Text VARCHAR(140),  
);
```


Keys

TWEET: <ID, Time, Text>

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP,  
  Text VARCHAR(140),  
);
```

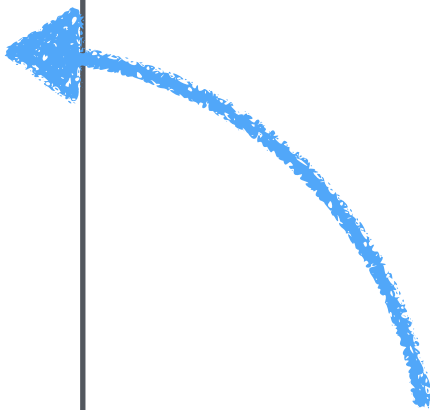


Enforces
"NOT NULL"

Keys

TWEET: <ID, Time, Text>

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP,  
  Text VARCHAR(140),  
);
```



Enforces
Uniqueness

Keys

TWEET: <ID, Time, Text>

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP,  
  Text VARCHAR(140),  
);
```

=

```
create table TWEET (  
  ID INT,  
  Time TIMESTAMP,  
  Text VARCHAR(140),  
  PRIMARY KEY (ID)  
);
```

Keys

```
create table AUTHOR (  
  Tweet INT,  
  Person VARCHAR(100),  
  PRIMARY KEY (Tweet, Person)  
);
```

Clicker Lightning Round!

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D

Clicker Lightning Round!



(a)



(b)

TWEET

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP ,  
  Text VARCHAR(140)  
);
```

```
insert into TWEET  
values(5, "2019-01-01 12:34:57", "lol");
```

Clicker Lightning Round!



TWEET

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP ,  
  Text VARCHAR(140)  
);
```

```
insert into TWEET  
values(5, "2019-01-01 12:34:57", "lol");
```

Clicker Lightning Round!



(a)



(b)

TWEET

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D

```
create table TWEET (  
ID INT PRIMARY KEY,  
Time TIMESTAMP ,  
Text VARCHAR(140)  
);
```

```
insert into TWEET  
values (E7w3WKVDB, "2019-01-01 12:34:57", "lol");
```


Clicker Lightning Round!



(a)



(b)

TWEET

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D

```
create table TWEET (  
ID INT PRIMARY KEY,  
Time TIMESTAMP ,  
Text VARCHAR(140)  
);
```

data type mismatch

```
insert into TWEET  
values (E7w3WKVDB, "2019-01-01 12:34:57", "lol");
```

Clicker Lightning Round!



(a)



(b)

TWEET

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D

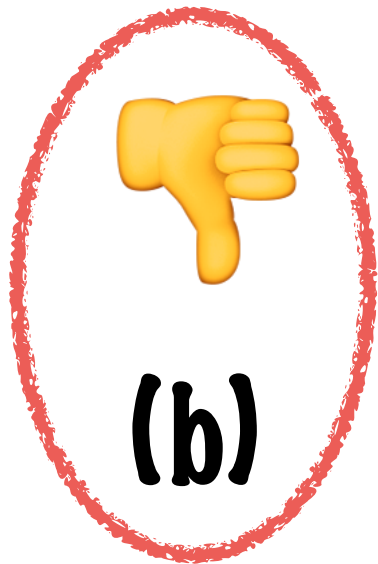
```
create table TWEET (  
ID INT PRIMARY KEY,  
Time TIMESTAMP ,  
Text VARCHAR(140)  
);
```

```
insert into TWEET (Timestamp, Text)  
values ("2019-01-01 12:34:57", "lol");
```

Clicker Lightning Round!



(a)



(b)

TWEET

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP ,  
  Text VARCHAR(140)  
);
```

requires NOT NULL

```
insert into TWEET (Timestamp, Text)  
values ("2019-01-01 12:34:57", "lol");
```

Clicker Lightning Round!



(a)



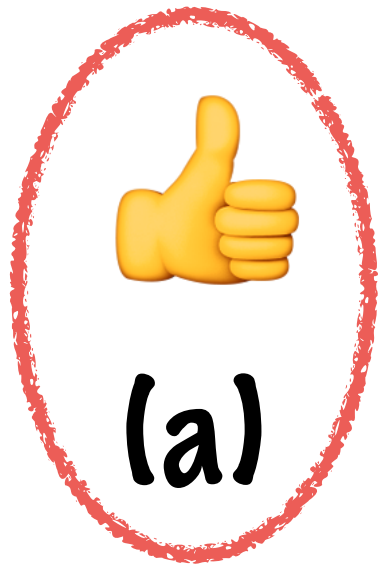
(b)

TWEET

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D

```
create table TWEET (  
  ID INT NOT NULL,  
  Time TIMESTAMP,  
  Text VARCHAR(140) DEFAULT "lol"  
);  
  
insert into TWEET(ID, Text)  
values(389472, "2019-01-01 12:34:57");
```

Clicker Lightning Round!



TWEET

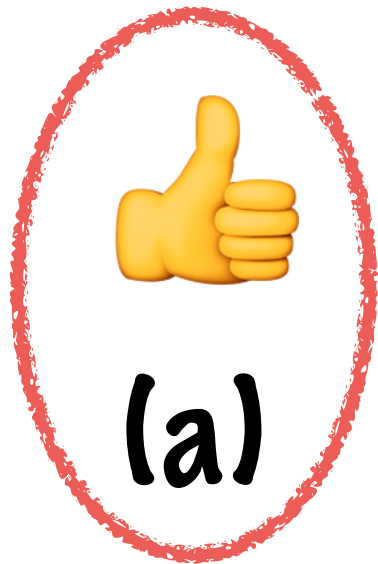
ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D

```
create table TWEET (  
  ID INT NOT NULL,  
  Time TIMESTAMP,  
  Text VARCHAR(140) DEFAULT "lol"  
);
```

```
insert into TWEET(ID, Text)  
values(389472, "2019-01-01 12:34:57");
```

weird, but
technically
fine

Clicker Lightning Round!



ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
389472	NULL	2019-01-01 12:34:57

```
create table TWEET (  
  ID INT NOT NULL,  
  Time TIMESTAMP,  
  Text VARCHAR(140) DEFAULT "lol"  
);
```

```
insert into TWEET(ID, Text)  
values(389472, "2019-01-01 12:34:57");
```

weird, but
technically
fine

Clicker Lightning Round!



(a)



(b)

TWEET

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D

```
create table TWEET (  
ID INT PRIMARY KEY,  
Time TIMESTAMP ,  
Text VARCHAR(140)  
);
```

```
insert into TWEET  
values (389472, "2019-01-04 12:14:37", "ugh");
```

Clicker Lightning Round!



(a)



(b)

TWEET

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP ,  
  Text VARCHAR(140)  
);
```

```
insert into TWEET  
values (389472, "2019-01-04 12:14:37", "ugh");
```

primary key
needs to be
unique

Keys

```
create table TWEET (  
  ID INT,  
  Time TIMESTAMP,  
  Text VARCHAR(140)  
);
```

```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
);
```

```
create table AUTHOR (  
  Tweet INT,  
  Person VARCHAR(100),  
  PRIMARY KEY (Tweet, Person)  
);
```

Keys

```
create table TWEET (  
  ID INT,  
  Time TIMESTAMP,  
  Text VARCHAR(140)  
);
```

```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
);
```

```
create table AUTHOR (  
  Tweet INT,  
  Person VARCHAR(100),  
  PRIMARY KEY (Tweet, Person)  
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID),  
  FOREIGN KEY (Person) REFERENCES PERSON(Handle),  
);
```

Referential Integrity

```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
  PRIMARY KEY (Handle)  
);
```

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP ,  
  Text VARCHAR(140)  
);
```

```
create table AUTHOR (  
  Tweet INT, Person VARCHAR(100),  
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID),  
  FOREIGN KEY (Person) REFERENCES PERSON(Handle),  
);
```

TWEET

ID	Timestamp	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 12:34:57	1951A 4 lyfe

PERSON

Handle	Name
m	Maulik
w	Wennie
g	Gurnaaz

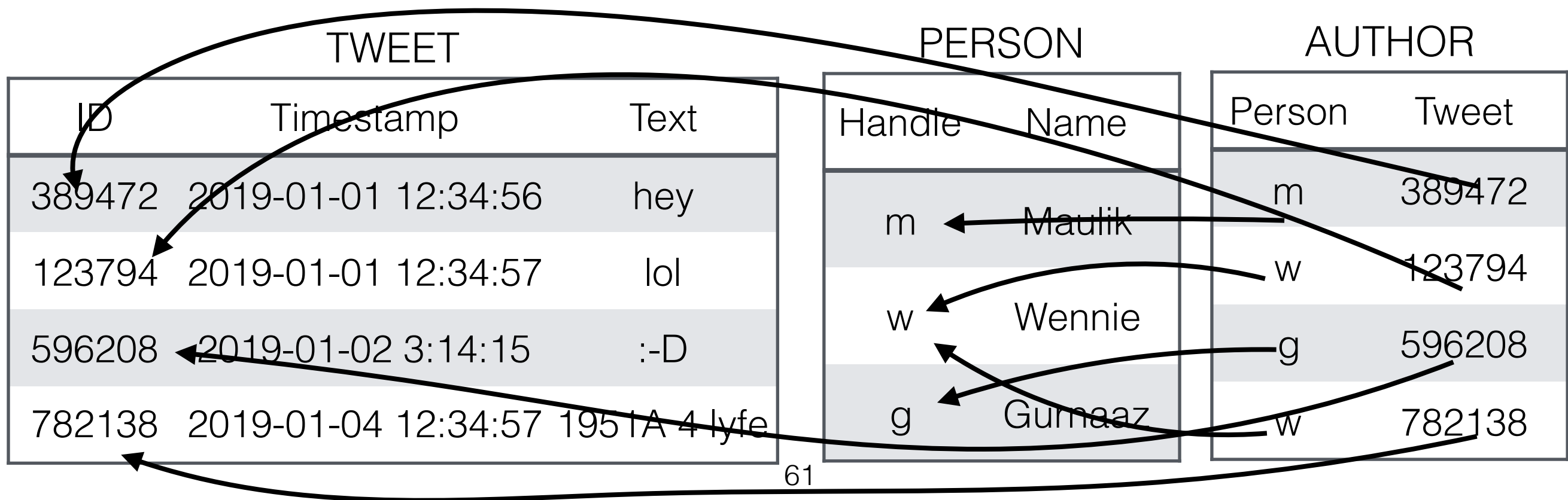
AUTHOR

Person	Tweet
m	389472
w	123794
g	596208
w	782138

```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
  PRIMARY KEY (Handle)  
);
```

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP ,  
  Text VARCHAR(140)  
);
```

```
create table AUTHOR (  
  Tweet INT, Person VARCHAR(100),  
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID),  
  FOREIGN KEY (Person) REFERENCES PERSON(Handle),  
);
```

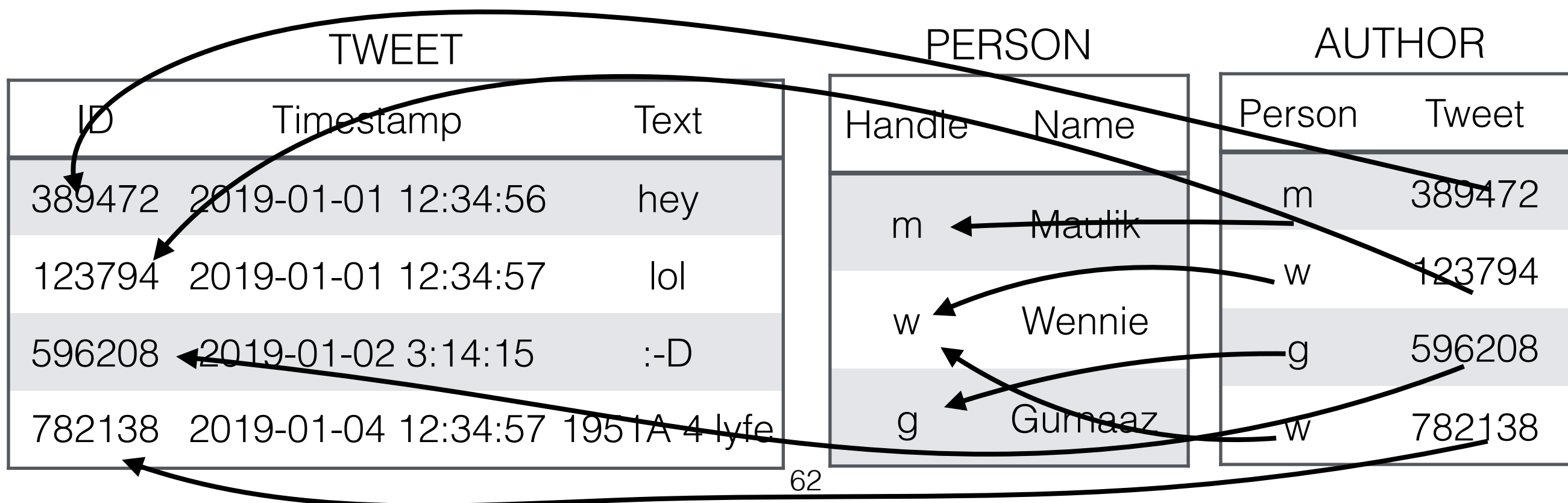


```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
  PRIMARY KEY (Handle)  
);
```

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP ,  
  Text VARCHAR(140)  
);
```

```
create table AUTHOR (  
  Tweet INT, Person VARCHAR(100),  
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID),  
  FOREIGN KEY (Person) REFERENCES PERSON(Handle),  
);
```

```
DELETE FROM TWEET WHERE ID = "596208"
```

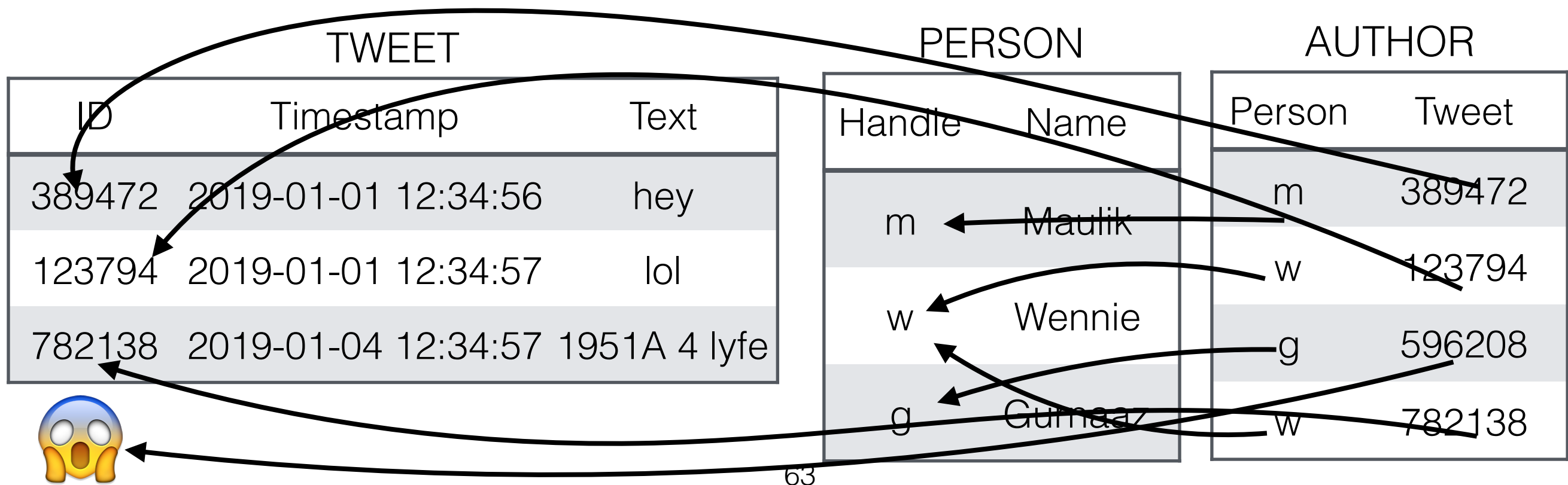


```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
  PRIMARY KEY (Handle)  
);
```

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP ,  
  Text VARCHAR(140)  
);
```

```
create table AUTHOR (  
  Tweet INT, Person VARCHAR(100),  
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID),  
  FOREIGN KEY (Person) REFERENCES PERSON(Handle),  
);
```

```
DELETE FROM TWEET WHERE ID = "596208"
```

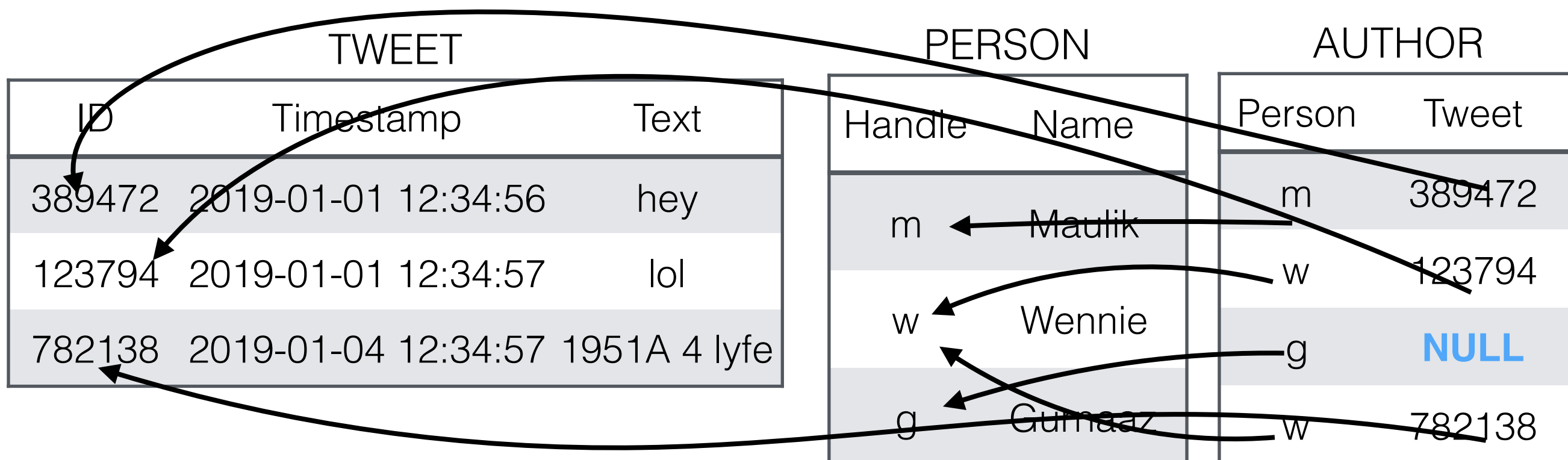


```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
  PRIMARY KEY (Handle)  
);
```

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP ,  
  Text VARCHAR(140)  
);
```

```
create table AUTHOR (  
  Tweet INT, Person VARCHAR(100),  
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID) ON DELETE SET NULL,  
  FOREIGN KEY (Person) REFERENCES PERSON(Handle),  
);
```

```
DELETE FROM TWEET WHERE ID = "596208"
```

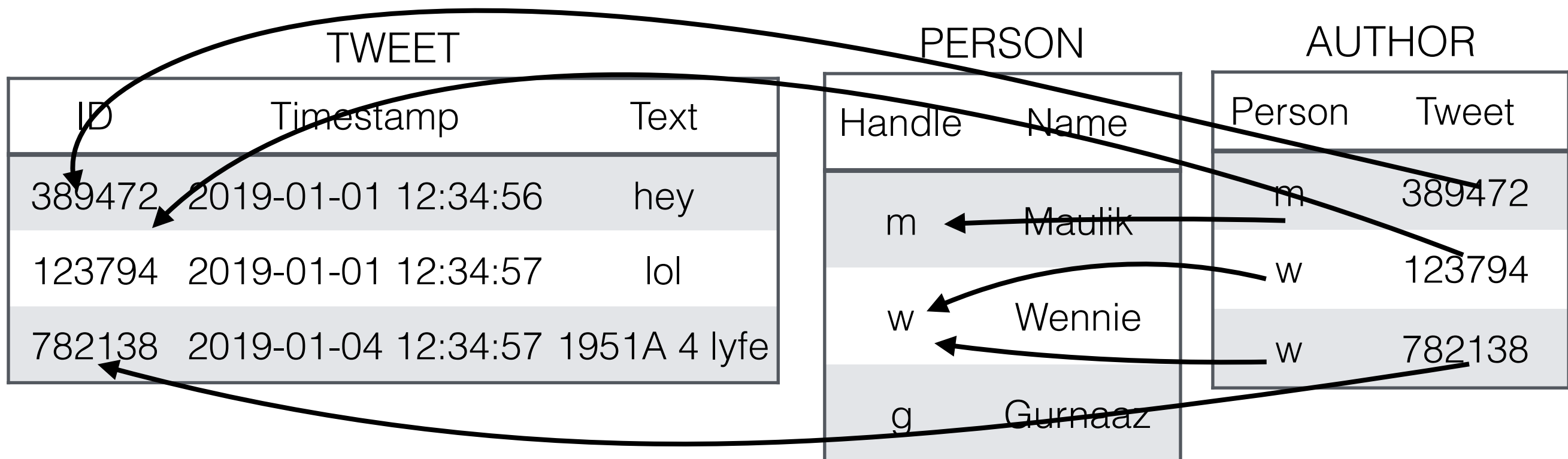



```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
  PRIMARY KEY (Handle)  
);
```

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP ,  
  Text VARCHAR(140)  
);
```

```
create table AUTHOR (  
  Tweet INT, Person VARCHAR(100),  
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID) ON DELETE CASCADE,  
  FOREIGN KEY (Person) REFERENCES PERSON(Handle),  
);
```

```
DELETE FROM TWEET WHERE ID = "596208"
```



```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
  PRIMARY KEY (Handle)  
);
```

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP ,  
  Text VARCHAR(140)  
);
```

```
create table AUTHOR (  
  Tweet INT, Person VARCHAR(100),  
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID) ON DELETE RESTRICT,  
  FOREIGN KEY (Person) REFERENCES PERSON(Handle),  
);
```

DELETE FROM TWEET WHERE ID = "596208"

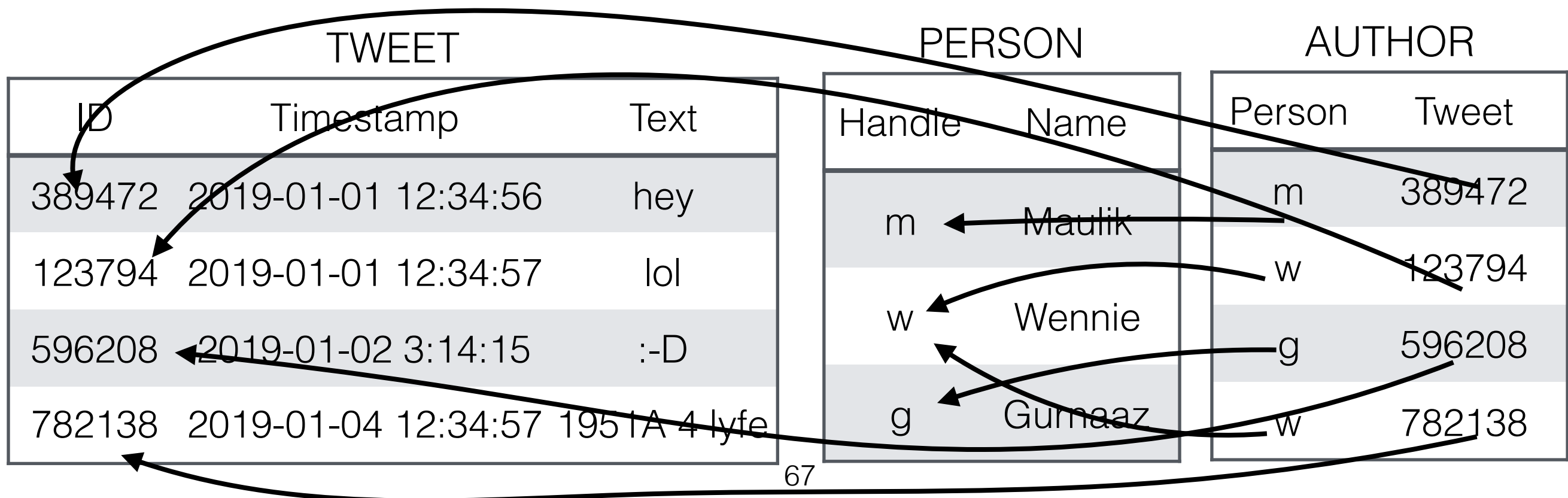


```
create table PERSON (  
  Handle VARCHAR(100),  
  Name VARCHAR(1000),  
  PRIMARY KEY (Handle)  
);
```

```
create table TWEET (  
  ID INT PRIMARY KEY,  
  Time TIMESTAMP ,  
  Text VARCHAR(140)  
);
```

```
create table AUTHOR (  
  Tweet INT, Person VARCHAR(100),  
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID)  
  ON DELETE RESTRICT  
  ON UPDATE CASCADE,  
  FOREIGN KEY (Person) REFERENCES PERSON(Handle),  
);
```

```
DELETE FROM TWEET WHERE ID = "596208"
```



Clicker Question!

```
create table PERSON (  
  Handle VARCHAR(100), Name VARCHAR(1000),  
  FOREIGN KEY (Author) REFERENCES PERSON(Handle) ON DELETE CASCADE,  
);  
create table TWEET (  
  ID INT, Text VARCHAR(140), Author VARCHAR(100),  
  FOREIGN KEY (Author) REFERENCES PERSON(Handle) ON DELETE CASCADE,  
);  
create table RETWEET (  
  Person VARCHAR(100), Tweet INT,  
  FOREIGN KEY (Person) REFERENCES PERSON(Handle) ON DELETE SET NULL,  
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID) ON DELETE SET NULL,  
);
```

PERSON

Handle	Name
m	Maulik
w	Wennie
g	Gurnaaz

TWEET

ID	Text	Author
1	hey	g
2	lol	g
3	:-D	w

RETWEET

Person	Tweet
m	1
m	2
w	1

```

create table PERSON (
  Handle VARCHAR(100), Name VARCHAR(1000),
  FOREIGN KEY (Author) REFERENCES PERSON(Handle) ON DELETE CASCADE,
);
create table TWEET (
  ID INT, Text VARCHAR(140), Author VARCHAR(100),
  FOREIGN KEY (Author) REFERENCES PERSON(Handle) ON DELETE CASCADE,
);
create table RETWEET (
  Person VARCHAR(100), Tweet INT,
  FOREIGN KEY (Person) REFERENCES PERSON(Handle) ON DELETE SET NULL,
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID) ON DELETE SET NULL,
);

```

PERSON

Handle	Name
m	Maulik
w	Wennie
g	Gurnaaz

TWEET

ID	Text	Author
1	hey	g
2	lol	g
3	:-D	w

RETWEET

Person	Tweet
m	1
m	2
w	1

```
DELETE FROM PERSON WHERE Handle = "g"
```

What happens...?

Clicker Question!

PERSON

TWEET

RETWEET

(a)

Handle	Name
m	Maulik
w	Wennie

ID	Text	Author
3	:-D	w

Person	Tweet
--------	-------

(b)

Handle	Name
m	Maulik
w	Wennie

ID	Text	Author
1	hey	NULL
2	lol	NULL
3	:-D	w

Person	Tweet
m	1
m	2
w	1

(c)

Handle	Name
m	Maulik
w	Wennie

ID	Text	Author
3	:-D	w

Person	Tweet
m	NULL
m	NULL
w	NULL

Clicker Question!

PERSON

TWEET

RETWEET

(a)

Handle	Name
m	Maulik
w	Wennie

ID	Text	Author
3	:-D	w

Person	Tweet
--------	-------

(b)

Handle	Name
m	Maulik
w	Wennie

ID	Text	Author
1	hey	NULL
2	lol	NULL
3	:-D	w

Person	Tweet
m	1
m	2
w	1

(c)

Handle	Name
m	Maulik
w	Wennie

ID	Text	Author
3	:-D	w

Person	Tweet
m	NULL
m	NULL
w	NULL

```

create table PERSON (
  Handle VARCHAR(100), Name VARCHAR(1000),
  FOREIGN KEY (Author) REFERENCES PERSON(Handle) ON DELETE CASCADE,
);
create table TWEET (
  ID INT, Text VARCHAR(140), Author VARCHAR(100),
  FOREIGN KEY (Author) REFERENCES PERSON(Handle) ON DELETE CASCADE,
);
create table RETWEET (
  Person VARCHAR(100), Tweet INT,
  FOREIGN KEY (Person) REFERENCES PERSON(Handle) ON DELETE SET NULL,
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID) ON DELETE SET NULL,
);

```

PERSON

Handle	Name
m	Maulik
w	Wennie
g	Gurnaaz

TWEET

ID	Text	Author
1	hey	g
2	lol	g
3	:-D	w

RETWEET

Person	Tweet
m	1
m	2
w	1

```
DELETE FROM PERSON WHERE Handle = "g"
```

What happens...?


```

create table PERSON (
  Handle VARCHAR(100), Name VARCHAR(1000),
  FOREIGN KEY (Author) REFERENCES PERSON(Handle) ON DELETE CASCADE,
);
create table TWEET (
  ID INT, Text VARCHAR(140), Author VARCHAR(100),
  FOREIGN KEY (Author) REFERENCES PERSON(Handle) ON DELETE CASCADE,
);
create table RETWEET (
  Person VARCHAR(100), Tweet INT,
  FOREIGN KEY (Person) REFERENCES PERSON(Handle) ON DELETE SET NULL,
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID) ON DELETE SET NULL,
);

```

PERSON

Handle	Name
m	Maulik
w	Wennie

TWEET

ID	Text	Author
1	hey	g
2	lol	g
3	:-D	w

RETWEET

Person	Tweet
m	1
m	2
w	1

```
DELETE FROM PERSON WHERE Handle = "g"
```

What happens...?

```

create table PERSON (
  Handle VARCHAR(100), Name VARCHAR(1000),
  FOREIGN KEY (Author) REFERENCES PERSON(Handle) ON DELETE CASCADE,
);
create table TWEET (
  ID INT, Text VARCHAR(140), Author VARCHAR(100),
  FOREIGN KEY (Author) REFERENCES PERSON(Handle) ON DELETE CASCADE,
);
create table RETWEET (
  Person VARCHAR(100), Tweet INT,
  FOREIGN KEY (Person) REFERENCES PERSON(Handle) ON DELETE SET NULL,
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID) ON DELETE SET NULL,
);

```

PERSON

Handle	Name
m	Maulik
w	Wennie

TWEET

ID	Text	Author
3	:-D	w

RETWEET

Person	Tweet
m	1
m	2
w	1

```
DELETE FROM PERSON WHERE Handle = "g"
```

What happens...?

```

create table PERSON (
  Handle VARCHAR(100), Name VARCHAR(1000),
  FOREIGN KEY (Author) REFERENCES PERSON(Handle) ON DELETE CASCADE,
);
create table TWEET (
  ID INT, Text VARCHAR(140), Author VARCHAR(100),
  FOREIGN KEY (Author) REFERENCES PERSON(Handle) ON DELETE CASCADE,
);
create table RETWEET (
  Person VARCHAR(100), Tweet INT,
  FOREIGN KEY (Person) REFERENCES PERSON(Handle) ON DELETE SET NULL,
  FOREIGN KEY (Tweet) REFERENCES TWEET(ID) ON DELETE SET NULL,
);

```

PERSON

Handle	Name
m	Maulik
w	Wennie

TWEET

ID	Text	Author
3	:-D	w

RETWEET

Person	Tweet

```
DELETE FROM PERSON WHERE Handle = "g"
```

What happens...?

Questions so far?

SQL for Data Querying and Analysis

Basic Template

```
SELECT <attribute list>  
FROM <table list>  
[ WHERE <condition> ]  
[ GROUP BY <attribute list> ]  
[ HAVING <condition> ]  
[ ORDER BY <attribute list> ];
```

Relational Algebra

- σ selection
- π project
- \cup union
- $-$ minus
- \times cross
- ρ rename

TWEET

ID	Time	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 15:04:57	1951A 4 lyfe
127890	2019-01-04 17:30:07	hey
173902	2019-01-05 3:34:18	i <3 1951A
893110	2019-01-06 12:21:53	i <3 1951A

```
SELECT *
FROM TWEET
WHERE Text = "hey"
```



```
SELECT <attribute list>
FROM <table list>
WHERE <condition>;
```

ID	Time	Text
389472	2019-01-01 12:34:56	hey
127890	2019-01-04 17:30:07	hey

σ selection

TWEET

ID	Time	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 15:04:57	1951A 4 lyfe
127890	2019-01-04 17:30:07	hey
173902	2019-01-05 3:34:18	i <3 1951A
893110	2019-01-06 12:21:53	i <3 1951A

```
SELECT ID, Time
FROM TWEET
WHERE Text = "hey"
```



```
SELECT <attribute list>
FROM <table list>
WHERE <condition>;
```



ID	Time
389472	2019-01-01 12:34:56
127890	2019-01-04 17:30:07

σ selection + π project

Relational Algebra

Join

dom0 = {  ,  }

dom1 = {  ,  }

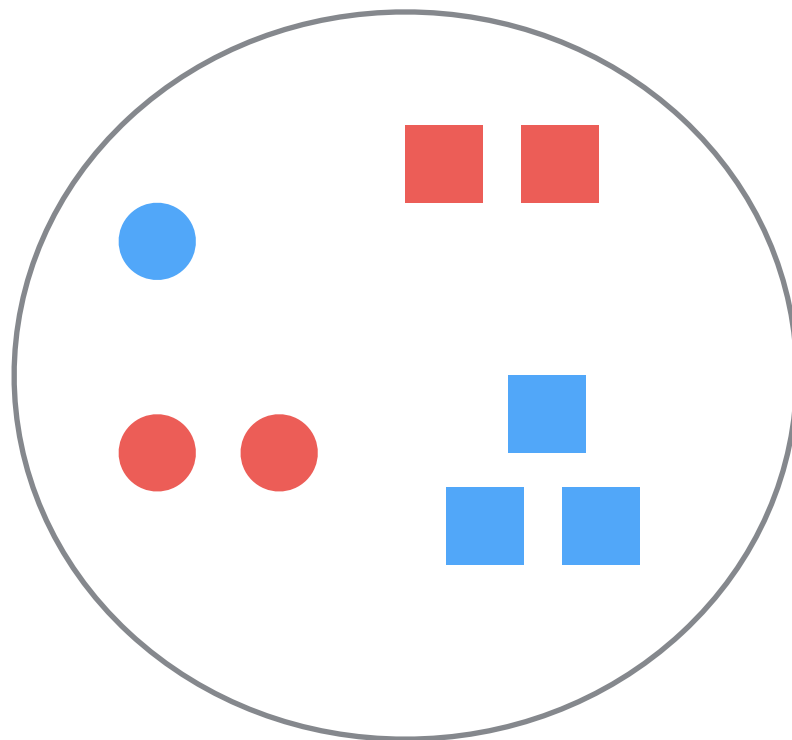
dom2 = {  ,  ,  }

dom3 = {blue, red, circle, square, 1, 2, 3}

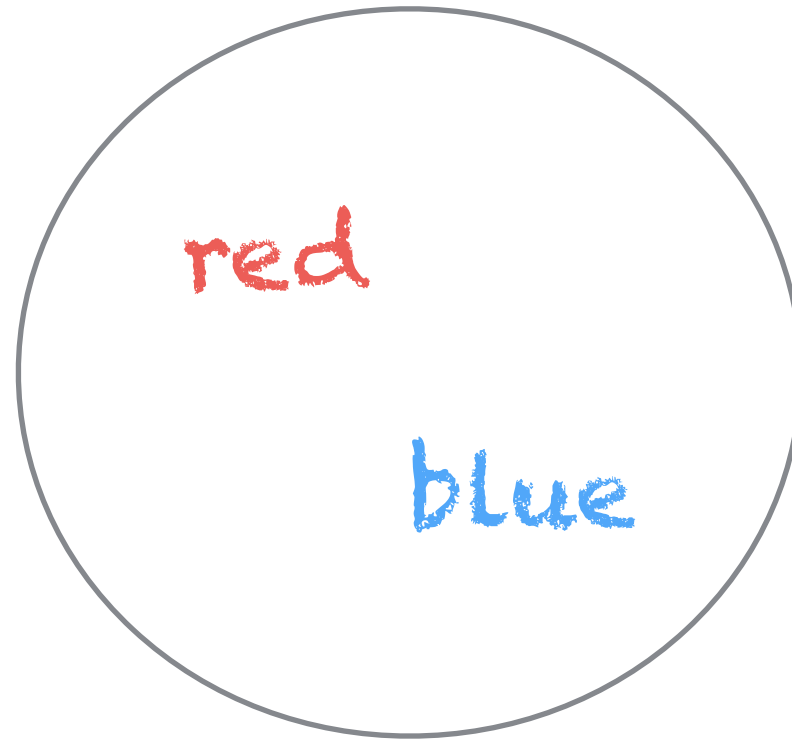
R: (A0:dom0, A1:dom1, A2:dom2)

S: (A0:dom0, A1:dom3)

R



S



Find the names of the colors of each element of R.

Relational Algebra

Join

dom0 = {  ,  }

dom1 = { (), [] }

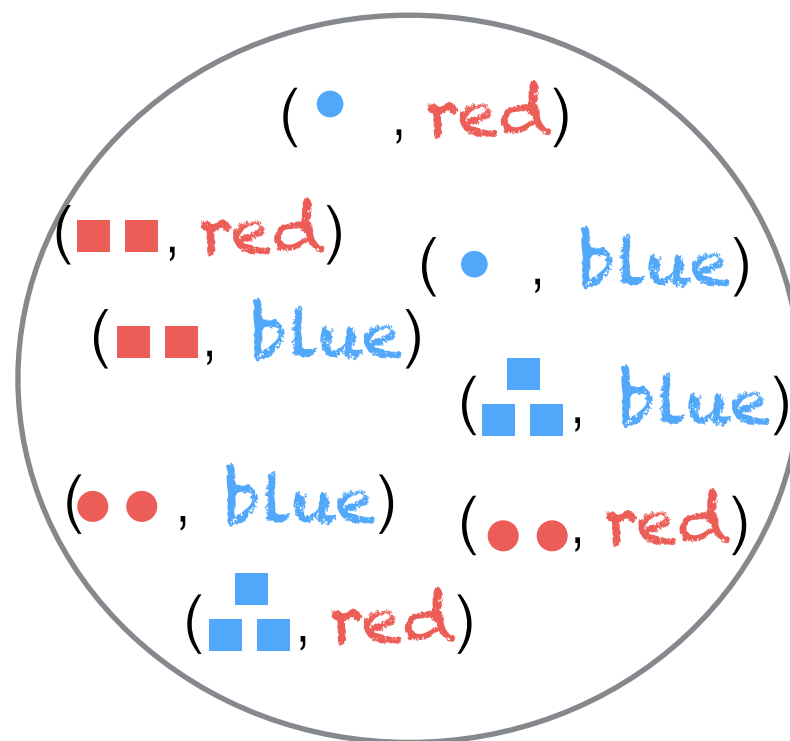
dom2 = { ✓, ✓✓, ✓✓✓ }

dom3 = {blue, red, circle, square, 1, 2, 3}

R: (A0:dom0, A1:dom1, A2:dom2)

S: (A0:dom0, A1:dom3)

$R \times S$



Find the names of the colors of each element of R.

Relational Algebra

Join

dom0 = {  ,  }

dom1 = { (), [] }

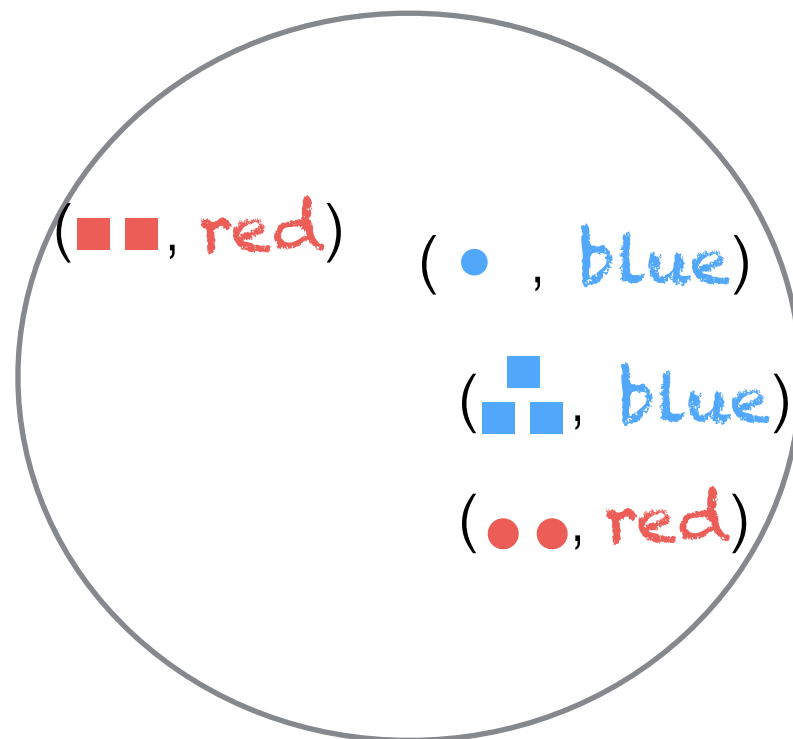
dom2 = { ✓, ✓✓, ✓✓✓ }

dom3 = {blue, red, circle, square, 1, 2, 3}

R: (A0:dom0, A1:dom1, A2:dom2)

S: (A0:dom0, A1:dom3)

$$\sigma_{R.A0 = S.A0}(R \times S)$$



Find the names of the colors of each element of R.

Relational Algebra

Join

dom0 = {  ,  }

dom1 = { (), [] }

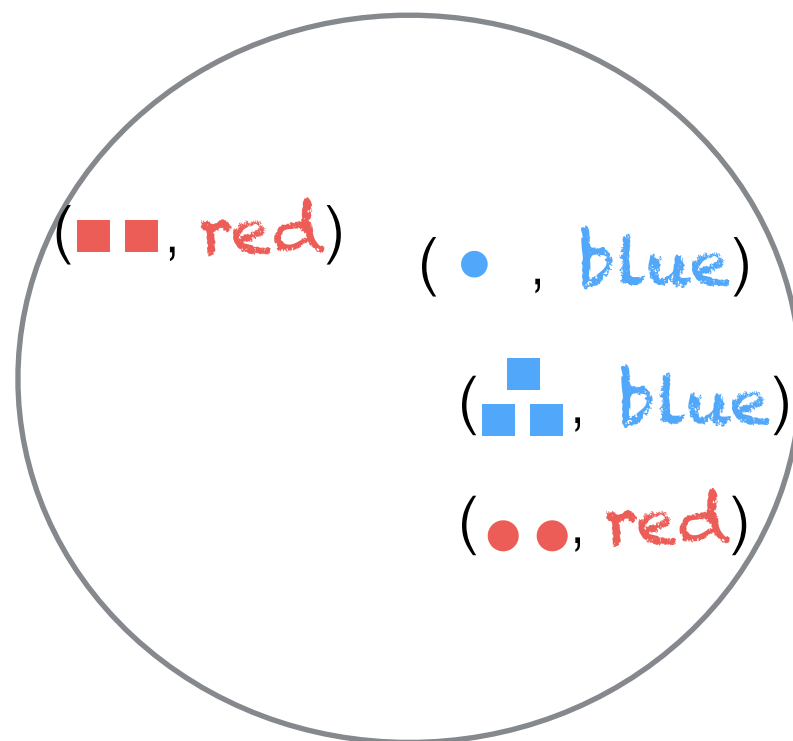
dom2 = { ✓, ✓✓, ✓✓✓ }

dom3 = {blue, red, circle, square, 1, 2, 3}

R: (A0:dom0, A1:dom1, A2:dom2)

S: (A0:dom0, A1:dom3)

$$R \bowtie_{R.A0 = S.A0} S$$



Find the names of the colors of each element of R.

TWEET

ID	Time	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 15:04:57	1951A 4 lyfe
127890	2019-01-04 17:30:07	hey
173902	2019-01-05 3:34:18	i <3 1951A
893110	2019-01-06 12:21:53	i <3 1951A

```
SELECT ID, Text
FROM Tweet,
      Author
WHERE ID = Tweet
```

AUTHOR

Person	Tweet
m	389472
w	123794
g	596208
w	782138
w	127890
g	173902
m	893110

TWEET

ID	Time		Text
389472	2019-01-01	12:34:56	hey
123794	2019-01-01	12:34:57	lol
596208	2019-01-02	3:14:15	:-D
782138	2019-01-04	15:04:57	1951A 4 lyfe
127890	2019-01-04	17:30:07	hey
173902	2019-01-05	3:34:18	i <3 1951A
893110	2019-01-06	12:21:53	i <3 1951A

```
SELECT ID, Text
FROM Tweet,
      Author
WHERE ID = Tweet
```



AUTHOR

Person	Tweet
m	389472
w	123794
g	596208
w	782138
w	127890
g	173902
m	893110

ID	Time	Text	Person	Tweet
389472	2019-01-01	hey	m	389472
123794	2019-01-01	lol	w	123794
596208	2019-01-02	:-D	g	596208
782138	2019-01-04	1951A 4 lyfe	w	782138
127890	2019-01-04	hey	w	127890
173902	2019-01-05	i <3 1951A	g	173902
893110	2019-01-06	i <3 1951A	m	893110

TWEET

ID	Time	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 15:04:57	1951A 4 lyfe
127890	2019-01-04 17:30:07	hey
173902	2019-01-05 3:34:18	i <3 1951A
893110	2019-01-06 12:21:53	i <3 1951A

```
SELECT ID, Text
FROM Tweet,
      Author
WHERE ID = Tweet
```

AUTHOR

Person	Tweet
m	389472
w	123794
g	596208
w	782138
w	127890
g	173902
m	893110

"Join Condition"

TWEET

ID	Time	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 15:04:57	1951A 4 lyfe
127890	2019-01-04 17:30:07	hey
173902	2019-01-05 3:34:18	i <3 1951A
893110	2019-01-06 12:21:53	i <3 1951A

```
SELECT ID, Text
FROM Tweet,
      Author
WHERE ID = Tweet
```

AUTHOR

Person	Tweet
m	389472
w	123794
g	596208
w	782138
w	127890
g	173902
m	893110

"Join Condition"

No condition → cross product

TWEET

ID	Time	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 15:04:57	1951A 4 lyfe
127890	2019-01-04 17:30:07	hey
173902	2019-01-05 3:34:18	i <3 1951A
893110	2019-01-06 12:21:53	i <3 1951A

```
SELECT ID, Text
FROM Tweet AS t,
      Author AS a
WHERE t.ID = a.Tweet
```

AUTHOR

Person	Tweet
m	389472
w	123794
g	596208
w	782138
w	127890
g	173902
m	893110

aliasing

TWEET

ID	Time	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 15:04:57	1951A 4 lyfe
127890	2019-01-04 17:30:07	hey
173902	2019-01-05 3:34:18	i <3 1951A
893110	2019-01-06 12:21:53	i <3 1951A

```
SELECT ID, Text
FROM Tweet AS t,
      Author AS a
WHERE t.ID = a.ID
```

AUTHOR

Person	ID
m	389472
w	123794
g	596208
w	782138
w	127890
g	173902
m	893110

aliasing (to avoid ambiguity)

TWEET

ID	Time	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 15:04:57	1951A 4 lyfe
127890	2019-01-04 17:30:07	hey
173902	2019-01-05 3:34:18	i <3 1951A
893110	2019-01-06 12:21:53	i <3 1951A

```
SELECT ID, Text
FROM Tweet AS t,
      Author AS a
WHERE t.ID = a.ID
```

AUTHOR

Person	ID
m	389472
w	123794
g	596208
w	782138
w	127890
g	173902
m	893110

aliasing (to avoid ambiguity)

p rename

TWEET

ID	Time	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 15:04:57	1951A 4 lyfe
127890	2019-01-04 17:30:07	hey
173902	2019-01-05 3:34:18	i <3 1951A
893110	2019-01-06 12:21:53	i <3 1951A

```
SELECT ID, Text
FROM Tweet AS t,
      Author AS a
WHERE t.ID = a.Tweet
      AND a.Person = "w"
```

AUTHOR

Person	Tweet
m	389472
w	123794
g	596208
w	782138
w	127890
g	173902
m	893110

TWEET

ID	Time	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 15:04:57	1951A 4 lyfe
127890	2019-01-04 17:30:07	hey
173902	2019-01-05 3:34:18	i <3 1951A
893110	2019-01-06 12:21:53	i <3 1951A

```
SELECT ID, Text
FROM Tweet AS t,
      Author AS a
WHERE t.ID = a.Tweet
      AND a.Person = "w"
```



AUTHOR

Person	Tweet
m	389472
w	123794
g	596208
w	782138
w	127890
g	173902
m	893110

ID	Time	Text	Person	Tweet
389472	2019-01-01	hey	m	389472
123794	2019-01-01	lol	w	123794
596208	2019-01-02	:-D	g	596208
782138	2019-01-04	1951A 4 lyfe	w	782138
127890	2019-01-04	hey	w	127890
173902	2019-01-05	i <3 1951A	g	173902
893110	2019-01-06	i <3 1951A	m	893110

TWEET

ID	Time	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 15:04:57	1951A 4 lyfe
127890	2019-01-04 17:30:07	hey
173902	2019-01-05 3:34:18	i <3 1951A
893110	2019-01-06 12:21:53	i <3 1951A

```
SELECT ID, Text
FROM Tweet AS t,
      Author AS a
WHERE t.ID = a.Tweet
      AND a.Person = "w"
```



AUTHOR

Person	Tweet
m	389472
w	123794
g	596208
w	782138
w	127890
g	173902
m	893110

ID	Time	Text	Person	Tweet
123794	2019-01-01 12:34:57	lol	w	123794
782138	2019-01-04 15:04:57	1951A 4 lyfe	w	782138
127890	2019-01-04 17:30:07	hey	w	127890

TWEET

ID	Time	Text
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
596208	2019-01-02 3:14:15	:-D
782138	2019-01-04 15:04:57	1951A 4 lyfe
127890	2019-01-04 17:30:07	hey
173902	2019-01-05 3:34:18	i <3 1951A
893110	2019-01-06 12:21:53	i <3 1951A

AUTHOR

Person	Tweet
m	389472
w	123794
g	596208
w	782138
w	127890
g	173902
m	893110

```
SELECT ID, Text
FROM Tweet AS t,
      Author AS a
WHERE t.ID = a.Tweet
      AND a.Person = "w"
```



ID	Text
123794	lol
782138	1951A 4 lyfe
127890	hey

(non)Clicker Question!

PERSON

Handle	Name
m	Maulik
w	Wennie
g	Gurnaaz

RETWEET

Person	Tweet
m	1
m	2
w	1

Find names of people who
have retweeted.

(non)Clicker Question!

PERSON

Handle	Name
m	Maulik
w	Wennie
g	Gurnaaz

RETWEET

Person	Tweet
m	1
m	2
w	1

Pst. Hint →

Person
Maulik
Maulik
Wennie

(non)Clicker Question!

PERSON

Handle	Name
m	Maulik
w	Wennie
g	Gurnaaz

RETWEET

Person	Tweet
m	1
m	2
w	1

```
SELECT Name
FROM PERSON AS p,
      RETWEET AS r
WHERE r.Person = p.Handle
```

JOINS

```
SELECT ID, Text  
FROM TWEET, AUTHOR  
WHERE ID = Tweet AND Person = "w"
```

JOINS

```
SELECT ID, Text  
FROM TWEET, AUTHOR  
WHERE ID = Tweet AND Person = "w"
```

=

```
SELECT ID, Text  
FROM (TWEET JOIN AUTHOR ON ID = Tweet)  
WHERE Person = "w"
```

JOINS

```
SELECT ID, Text
FROM TWEET, AUTHOR
WHERE ID = Tweet AND Person = "w"
```

=

```
SELECT ID, Text
FROM (TWEET JOIN AUTHOR ON ID = Tweet)
WHERE Person = "w"
```

JOINS

TWEET

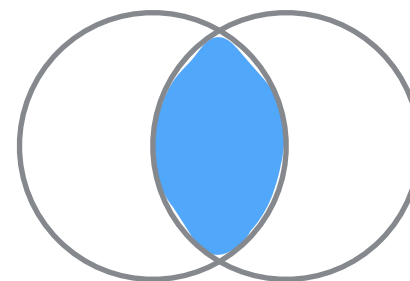
ID	Text
389472	hey
596208	:-D
782138	1951A 4 lyfe
173902	i <3 1951A
893110	i <3 1951A

AUTHOR

Person	Tweet
m	389472
g	596208
g	173902
m	893110
w	672109

```
SELECT ID, Text
FROM (TWEET JOIN AUTHOR
      ON ID = Tweet)
WHERE Person = "w"
```

Person	Tweet	ID	Text
m	389472	389472	hey
g	596208	596208	:-D
g	173902	173902	i <3 1951A
m	893110	893110	i <3 1951A
w	672109		
		782138	1951A 4 lyfe



Inner Join

JOINS

TWEET

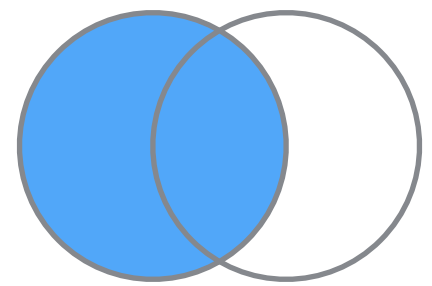
ID	Text
389472	hey
596208	:-D
782138	1951A 4 lyfe
173902	i <3 1951A
893110	i <3 1951A

AUTHOR

Person	Tweet
m	389472
g	596208
g	173902
m	893110
w	672109

```
SELECT ID, Text
FROM (TWEET JOIN AUTHOR
      ON ID = Tweet)
WHERE Person = "w"
```

Person	Tweet	ID	Text
m	389472	389472	hey
g	596208	596208	:-D
g	173902	173902	i <3 1951A
m	893110	893110	i <3 1951A
w	672109		
		782138	1951A 4 lyfe



JOINS

Left Outer Join

TWEET

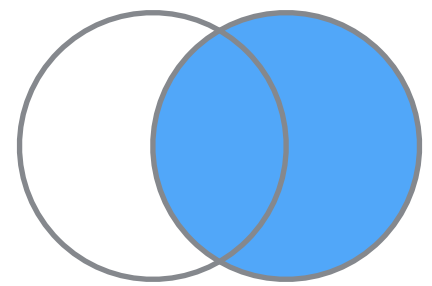
ID	Text
389472	hey
596208	:-D
782138	1951A 4 lyfe
173902	i <3 1951A
893110	i <3 1951A

```
SELECT ID, Text
FROM (TWEET LEFT OUTER JOIN AUTHOR
      ON ID = Tweet)
WHERE Person = "w"
```

AUTHOR

Person	Tweet
m	389472
g	596208
g	173902
m	893110
w	672109

Person	Tweet	ID	Text
m	389472	389472	hey
g	596208	596208	:-D
g	173902	173902	i <3 1951A
m	893110	893110	i <3 1951A
w	672109		
NULL	NULL	782138	1951A 4 lyfe



JOINS

Right Outer Join

TWEET

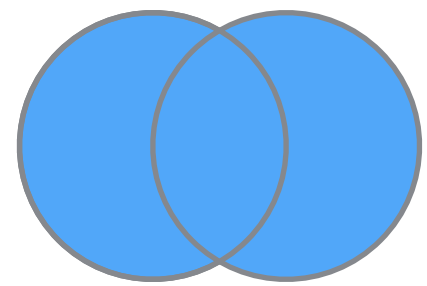
ID	Text
389472	hey
596208	:-D
782138	1951A 4 lyfe
173902	i <3 1951A
893110	i <3 1951A

```
SELECT ID, Text
FROM (TWEET RIGHT OUTER JOIN AUTHOR
      ON ID = Tweet)
WHERE Person = "w"
```

AUTHOR

Person	Tweet
m	389472
g	596208
g	173902
m	893110
w	672109

Person	Tweet	ID	Text
m	389472	389472	hey
g	596208	596208	:-D
g	173902	173902	i <3 1951A
m	893110	893110	i <3 1951A
w	672109	NULL	NULL
		782138	1951A 4 lyfe



JOINS

Full Outer Join

TWEET

ID	Text
389472	hey
596208	:-D
782138	1951A 4 lyfe
173902	i <3 1951A
893110	i <3 1951A

```
SELECT ID, Text
FROM (TWEET FULL OUTER JOIN AUTHOR
      ON ID = Tweet)
WHERE Person = "w"
```

AUTHOR

Person	Tweet
m	389472
g	596208
g	173902
m	893110
w	672109

Person	Tweet	ID	Text
m	389472	389472	hey
g	596208	596208	:-D
g	173902	173902	i <3 1951A
m	893110	893110	i <3 1951A
w	672109	NULL	NULL
NULL	NULL	782138	1951A 4 lyfe

JOINS

Natural Join

TWEET

ID	Text
389472	hey
596208	:-D
782138	1951A 4 lyfe
173902	i <3 1951A
893110	i <3 1951A

```
SELECT ID, Text
FROM (TWEET JOIN AUTHOR)
WHERE Person = "w"
```

AUTHOR

Person	Tweet
m	389472
g	596208
g	173902
m	893110
w	672109

assumes condition is
ALL PAIRS of attributes with
matching names

JOINS

Natural Join

TWEET

ID	Text
389472	hey
596208	:-D
782138	1951A 4 lyfe
173902	i <3 1951A
893110	i <3 1951A

```
SELECT ID, Text
FROM (TWEET JOIN AUTHOR)
WHERE Person = "w"
```

AUTHOR

Person	Tweet
m	389472
g	596208
g	173902
m	893110
w	672109

if no matches, forms full
cross product

JOINS

Natural Join

TWEET

ID	Text
389472	hey
596208	:-D
782138	1951A 4 lyfe
173902	i <3 1951A
893110	i <3 1951A

```
SELECT ID, Text
FROM (TWEET AS t(tweetid, text) JOIN
      AUTHOR AS a(person, tweetid))
```

AUTHOR

Person	Tweet
m	389472
g	596208
g	173902
m	893110
w	672109

Person	tweetid	tweetid	Text
m	389472	389472	hey
g	596208	596208	:-D
g	173902	173902	i <3 1951A
m	893110	893110	i <3 1951A
w	672109	NULL	NULL
NULL	NULL	782138	1951A 4 lyfe

Natural (Inner) Join

JOINS

TWEET

ID	Text
389472	hey
596208	:-D
782138	1951A 4 lyfe
173902	i <3 1951A
893110	i <3 1951A

```
SELECT ID, Text
FROM (TWEET AS t(tweetid, text) JOIN
      AUTHOR AS a(person, tweetid))
```

AUTHOR

Person	Tweet
m	389472
g	596208
g	173902
m	893110
w	672109

Person	tweetid	tweetid	Text
m	389472	389472	hey
g	596208	596208	:-D
g	173902	173902	i <3 1951A
m	893110	893110	i <3 1951A

Natural (Inner) Join

JOINS

TWEET

ID	Text
389472	hey
596208	:-D
782138	1951A 4 lyfe
173902	i <3 1951A
893110	i <3 1951A

```
SELECT ID, Text
FROM (TWEET AS t(tweetid, text) JOIN
      AUTHOR AS a(person, tweetid))
```

AUTHOR

Person	Tweet
m	389472
g	596208
g	173902
m	893110
w	672109

Person	tweetid	Text
m	389472	hey
g	596208	:-D
g	173902	i <3 1951A
m	893110	i <3 1951A

Natural (Inner) Join

JOINS

TWEET

ID	Text
389472	hey
596208	:-D
782138	1951A 4 lyfe
173902	i <3 1951A
893110	i <3 1951A

```
SELECT ID, Text
FROM (TWEET AS t(tweetid, foo) JOIN
      AUTHOR AS a(foo, tweetid)
```

AUTHOR

Person	Tweet
m	389472
g	596208
g	173902
m	893110
w	672109

foo	tweetid	foo
m	389472	hey
g	596208	:-D
g	173902	i <3 1951A
m	893110	i <3 1951A

Natural (Inner) Join

JOINS

TWEET

ID	Text
389472	hey
596208	:-D
782138	1951A 4 lyfe
173902	i <3 1951A
893110	i <3 1951A

```
SELECT ID, Text
FROM (TWEET AS t(tweetid, foo) JOIN
      AUTHOR AS a(foo, tweetid)
```

foo

tweetid

AUTHOR

Person	Tweet
m	389472
g	596208
g	173902
m	893110
w	672109

Clicker Question!

STUDENT

ID	Name
1	Wennie
2	Maulik
3	Gurnaaz
4	Jens
5	Erin

GRADES

Student	Course	Grade
1	32	A
2	1951A	A
6	32	A

```
SELECT Name, Course
FROM (STUDENT LEFT OUTER JOIN GRADES ON ID = Student)
```

Name	Course
Wennie	32
Maulik	1951A
NULL	32

(a)

Name	Course
Wennie	32
Maulik	1951A
Gurnaaz	NULL
Jens	NULL
Erin	NULL

(b)

Clicker Question!

STUDENT

ID	Name
1	Wennie
2	Maulik
3	Gurnaaz
4	Jens
5	Erin

GRADES

Student	Course	Grade
1	32	A
2	1951A	A
6	32	A

```
SELECT Name, Course
FROM (STUDENT LEFT OUTER JOIN GRADES ON ID = Student)
```

Name	Course
Wennie	32
Maulik	1951A
NULL	32

(a)

Name	Course
Wennie	32
Maulik	1951A
Gurnaaz	NULL
Jens	NULL
Erin	NULL

(b)

Clicker Question!

STUDENT

ID	Name
1	Wennie
2	Maulik
3	Gurnaaz
4	Jens
5	Erin

GRADES

Student	Course	Grade
1	32	A
2	1951A	A
6	32	A

Name	Course
Wennie	32
Maulik	1951A
NULL	32

(a)

```
SELECT Name, Course
FROM (STUDENT RIGHT OUTER JOIN GRADES ON ID = Student)
```

(b)

```
SELECT Name, Course
FROM (STUDENT NATURAL JOIN GRADES ON ID = Student)
```

Clicker Question!

STUDENT

ID	Name
1	Wennie
2	Maulik
3	Gurnaaz
4	Jens
5	Erin

GRADES

Student	Course	Grade
1	32	A
2	1951A	A
6	32	A

Name	Course
Wennie	32
Maulik	1951A
NULL	32

(a)

```
SELECT Name, Course  
FROM (STUDENT RIGHT OUTER JOIN GRADES ON ID = Student)
```

(b)

```
SELECT Name, Course  
FROM (STUDENT NATURAL JOIN GRADES ON ID = Student)
```

ORDER BY

TWEET

ID	Time	Text
782138	2019-01-04 15:04:57	1951A 4 lyfe
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
127890	2019-01-04 17:30:07	hey
893110	2019-01-06 12:21:53	i <3 1951A
596208	2019-01-02 3:14:15	:-D
173902	2019-01-05 3:34:18	i <3 1951A

```
SELECT Text
FROM Tweet
ORDER BY Time
```

Text
hey
lol
:-D
1951A 4 lyfe
hey
i <3 1951A
i <3 1951A

ORDER BY

TWEET

ID	Time	Text
782138	2019-01-04 15:04:57	1951A 4 lyfe
389472	2019-01-01 12:34:56	hey
123794	2019-01-01 12:34:57	lol
127890	2019-01-04 17:30:07	hey
893110	2019-01-06 12:21:53	i <3 1951A
596208	2019-01-02 3:14:15	:-D
173902	2019-01-05 3:34:18	i <3 1951A

```
SELECT Text
FROM Tweet
ORDER BY ID
```

Text
lol
hey
i <3 1951A
hey
:-D
1951A 4 lyfe
i <3 1951A

GROUP BY

TWEET

ID	Likes	Text
782138	1,000	1951A 4 lyfe
389472	10	hey
123794	100	lol
127890	0	hey
893110	8,000,000	i <3 1951A
596208	1	:-D
173902	1,000,000,000	i <3 1951A

```
SELECT Text,  
Count(*), AVG(Likes)  
FROM Tweet  
GROUP BY Text
```

Text	Count(*)	AVG(Likes)
lol	1	100
hey	2	5
i <3 1951A	2	504,000,000
:-D	1	1
1951A 4 lyfe	1	1,000

GROUP BY

TWEET

ID	Likes	Text
782138	1,000	1951A 4 lyfe
389472	10	hey
123794	100	lol
127890	0	hey
893110	8,000,000	i <3 1951A
596208	1	:-D
173902	1,000,000,000	i <3 1951A

```
SELECT Text,  
Count(*), AVG(Likes)  
FROM Tweet  
GROUP BY Text
```

Text	Count(*)	AVG(Likes)
lol	1	100
hey	2	5
i <3 1951A	2	504,000,000
:-D	1	1
1951A 4 lyfe	1	1,000

SUM, MIN, MAX,
COUNT, AVG

HAVING

TWEET

ID	Likes	Text
782138	1,000	1951A 4 lyfe
389472	10	hey
123794	100	lol
127890	0	hey
893110	8,000,000	i <3 1951A
596208	1	:-D
173902	1,000,000,000	i <3 1951A

```
SELECT Text,  
Count(*), AVG(Likes)  
FROM Tweet  
GROUP BY Text  
HAVING COUNT(*) > 1
```

Text	Count(*)	AVG(Likes)
hey	2	5
i <3 1951A	2	504,000,000

LIKE

TWEET

ID	Likes	Text
782138	1,000	1951A 4 lyfe
389472	10	hey
123794	100	lol
127890	0	hey
893110	8,000,000	i <3 1951A
596208	1	:-D
173902	1,000,000,000	i <3 1951A

```
SELECT Text, Count(*),  
AVG(Likes)  
FROM Tweet  
WHERE Text LIKE '%1951A%'  
GROUP BY Text
```

Text	Count(*)	AVG(Likes)
1951A 4 lyfe	1	1,000
i <3 1951A	2	504,000,000

