GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

LECTURE 3:

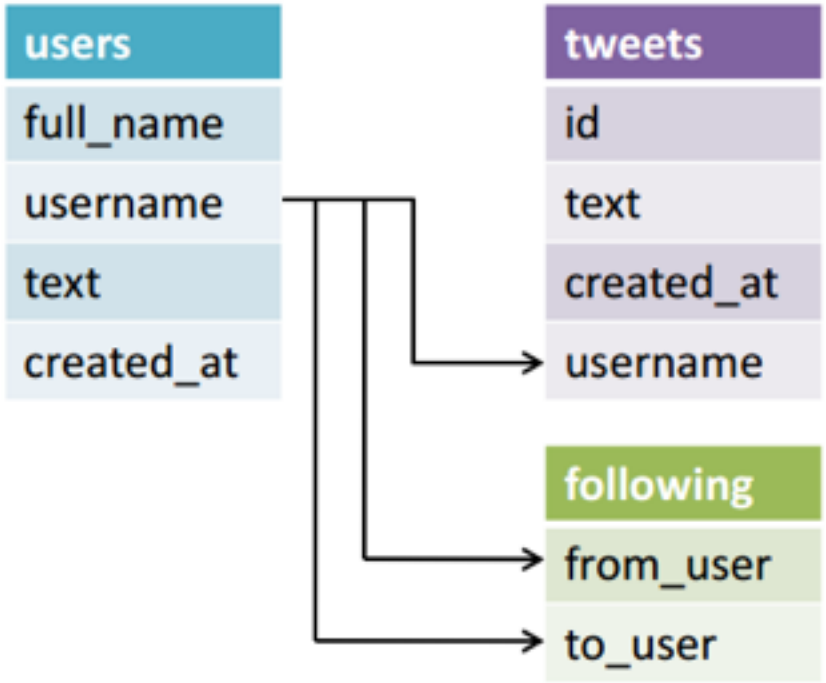
I. RELATIONAL DATABASES
II. SQL EXERCISES
III. MYSQL & PYTHON TUTORIAL

I. RELATIONAL DATABASES

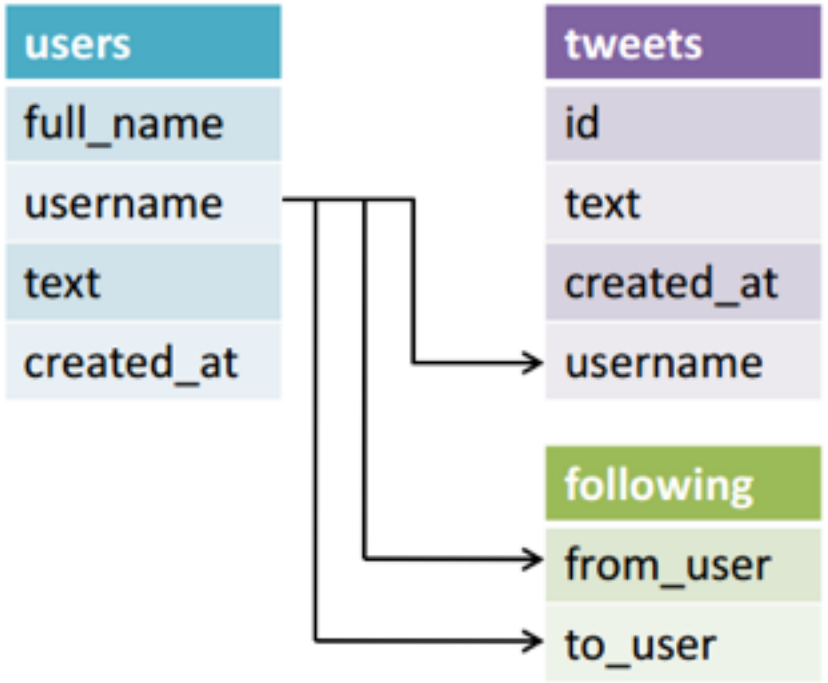
Relational database are traditionally organized in the following manner:

*A database has **tables** which represent individual entities or objects*

*Tables have a predefined **schema** - rules that tell it what columns exist and what they look like*

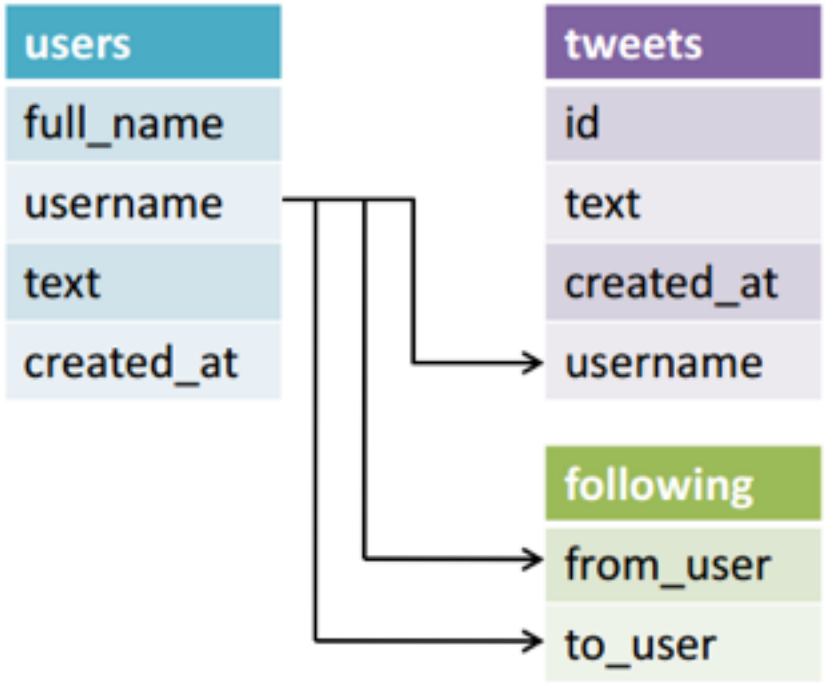

users
full_name
username
text
created_at
tweets
id
text
created_at
username
following
from_user
to_user

Each table should have a **primary key** column– a unique identifier for that row


users
full_name
username
text
created_at
tweets
id
text
created_at
username
following
from_user
to_user

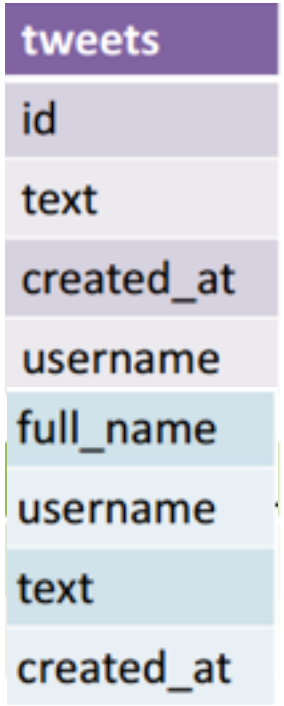
*Each table should have a **primary key** column- a unique identifier for that row*

*Additionally each table can have a **foreign key** column- an id that links this to table to another*


users
full_name
username
text
created_at
tweets
id
text
created_at
username
following
from_user
to_user

We could have had a table structure as follow:

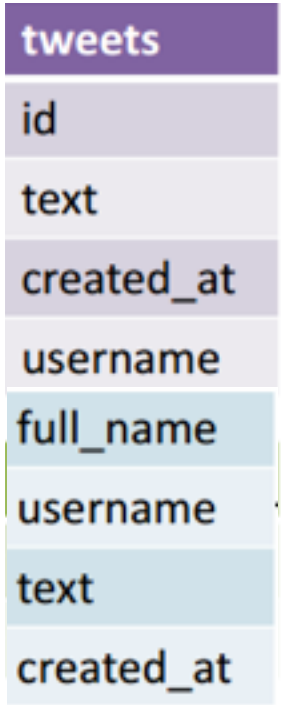
Why is this different?



We could have had a table structure as follow:
Why is this different?

We would repeat the user information on each row.

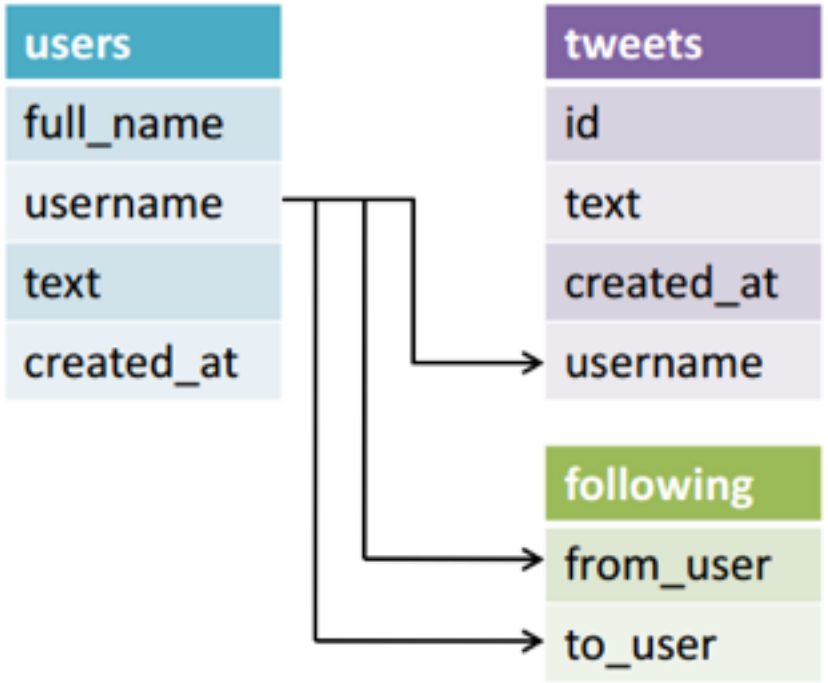
This is called denormalization

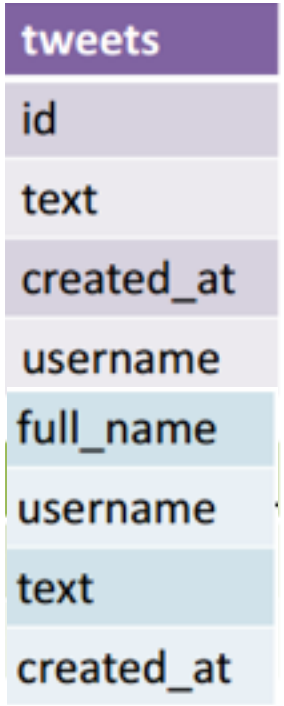


Normalized Data: Many tables to reduce redundant or repeated data in a table

Denormalized Data:

Wide data, fields are often repeated but removes the need to join together multiple tables


users
full_name
username
text
created_at
tweets
id
text
created_at
username
following
from_user
to_user


tweets
id
text
created_at
username
full_name
username
text
created_at

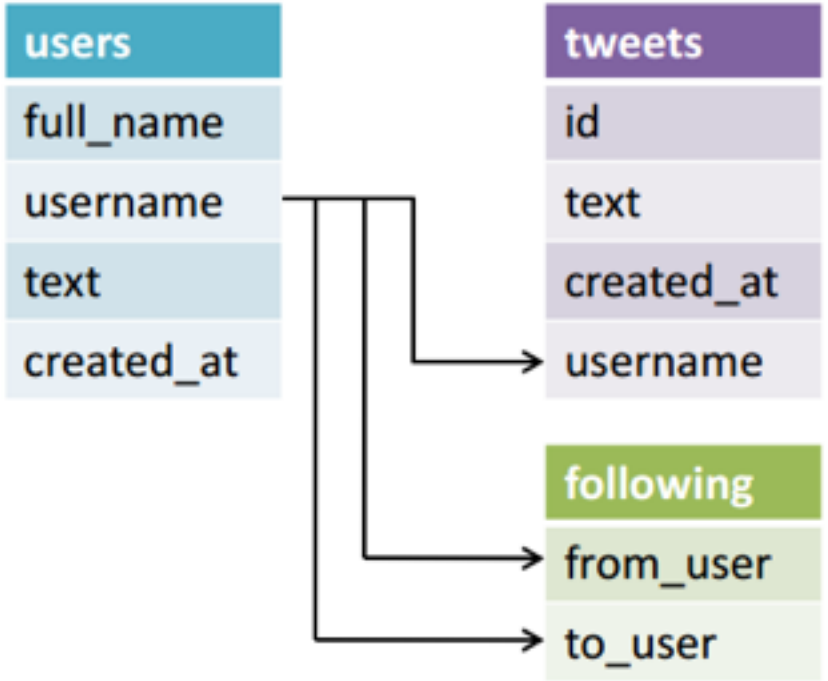
Normalized Data: Many tables to reduce redundant or repeated data in a table

Denormalized Data:
Wide data, fields are often repeated but removes the need to join together multiple tables

Trade off of speed vs. storage

Q: Why are normalized tables (possibly) slower to read?

Q: Why are normalized tables (possibly) slower to read?

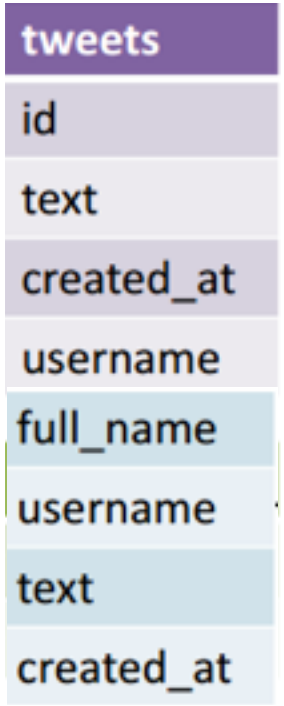


Q: Why are normalized tables (possibly) slower to read?

A: We'll have to get data from multiple tables to answer some questions.

Q: Why are denormalized tables (possibly) slower to write?

Q: Why are denormalized tables (possibly) slower to write?



Q: Why are denormalized tables (possibly) slower to write?

A: We'll have to write more information on each write

SQL is a query language to load, retrieve and update data in relational databases

SELECT: *Allows you to* ***retrieve*** *information from a table*

Syntax:

SELECT col1, col2 FROM table WHERE <some condition>

Example:
SELECT poll_title, poll_date FROM polls WHERE romney_pct > obama_pct

GROUP BY: *Allows you to* ***aggregate*** *information from a table*

Syntax:

SELECT col1, AVG(col2) FROM table GROUP BY col1

Example:
SELECT poll_date, AVG(obama_pct) FROM polls GROUP BY poll_date

GROUP BY: *Allows you to* ***aggregate*** *information from a table*

Syntax:

SELECT col1, AVG(col2) FROM table GROUP BY col1

Example:
SELECT poll_date, AVG(obama_pct) FROM polls GROUP BY poll_date

GROUP BY: *Allows you to* ***aggregate*** *information from a table*

Syntax:

SELECT col1, AVG(col2) FROM table GROUP BY col1

There are usually a few common built-in operations:
SUM, AVG, MIN, MAX, COUNT

JOIN: *Allows you to* ***combine*** *multiple tables*

Syntax:

```
SELECT table1.col1, table1.col2, table2.col2
FROM  table1 JOIN table2 ON table1.col1 = table2.col2
```

JOIN: *Allows you to* ***combine*** *multiple tables*

Syntax:

```
SELECT table1.col1, table1.col2, table2.col2
FROM  (JOIN table1, table2 ON table1.col1 = table2.col2)
```

INSERT: *Allows you to* ***add*** *data to tables*

Syntax and Example:

```
INSERT INTO <table> (col1, col2)
VALUES( ...)

INSERT INTO classroom (first_name, last_name)
VALUES('John', 'Doe');
```

II. SQL EXERCISES

III. MYSQL & PYTHON TUTORIAL