Databases for Data Science

Fall 2022 - Lecture 01

What’s a Database?

Persistent storage:

Data that outlives any one program

Databases Course

General skills

- Database and programming technical skill
- Engagement with data
- Judgment, synthesis, critical thinking, problem solving
- Teamwork, communication

Databases Course

Technical skills

- CRUD - create, read, update, and delete data (SQL)
- Explore data and answer questions (more SQL, python)
- Display data (python, web tools)
- Linux shell interface
- Clean data (more SQL, Linux shell)

Communications

Slack: #databases-f21 or DMs

Email: wcorning@ncf.edu

Office Hours

In-person at HNS 105: TBD

Remotely by appointment (email / Slack)

when2meet.com/?16574394-HcDPf

Coursework

Graded work:

- Weekly homework assignments
- Final project
- Final exam 10/12
- Class participation

Please read the policies on Canvas regarding deadlines, academic honesty, and **citing reference materials**.

command line prompt

Accessing NCF Linux server

```
$ ssh wcorning@cs1.ncf.edu

password:*********
```

Your temporary password: tbd

Change your password first thing:

```
$ passwd
```

REMEMBER YOUR PASSWORD!

(or, better yet, use a password manager!)

If you’re locked out, email it@ncf.edu, cc: me

Recommended tools

- Standalone code editor: VSCode
  - Can run remotely via SSH - see https://code.visualstudio.com/docs/remote/ssh
  - Native support for Jupyter notebooks

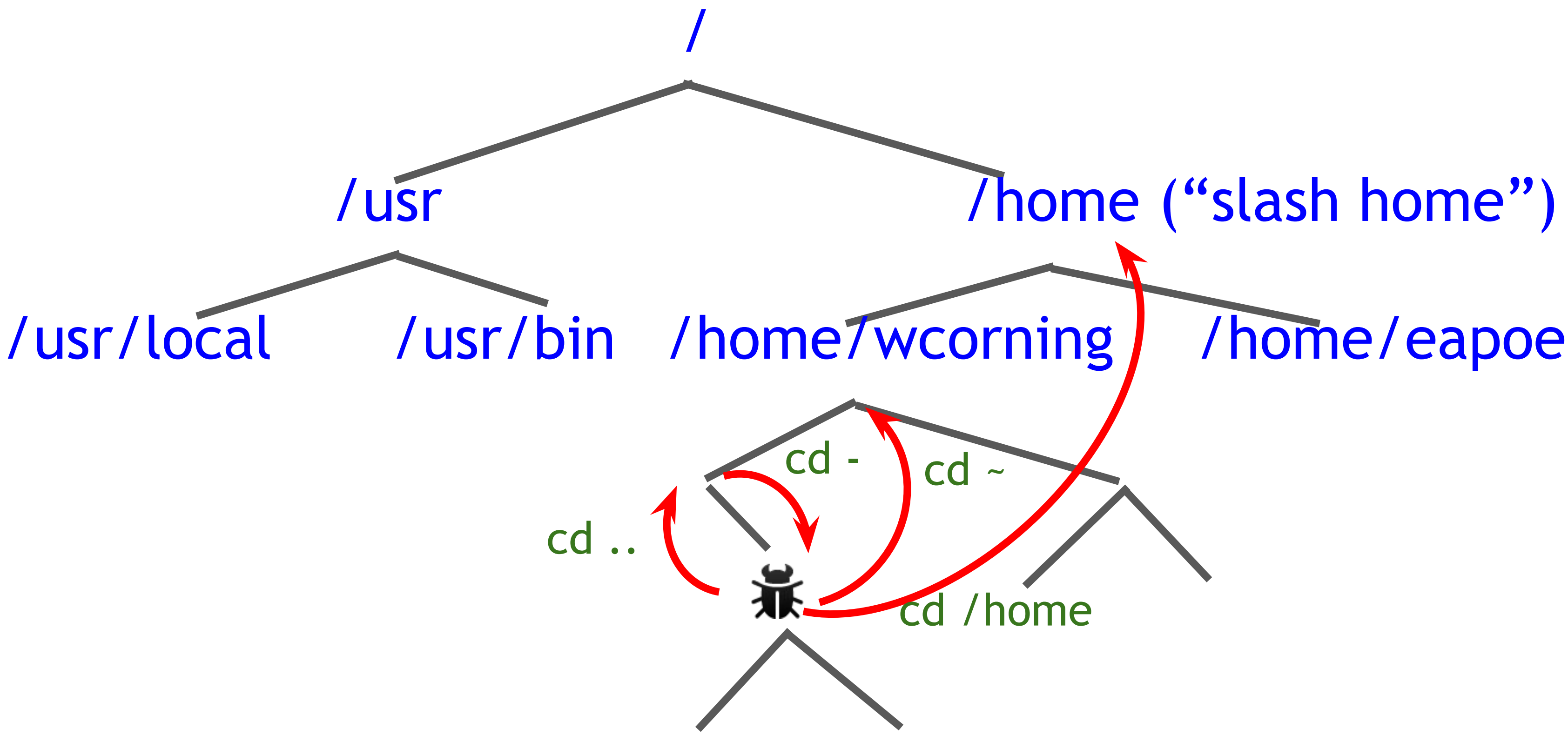
- Terminal text editor: `nano`
  - (or vim / emacs, if you're feeling ambitious)

- Local development on Windows: Windows Subsystem for Linux

- Source control: `git` with GitHub

Linux command line

cd: change directory



/

/usr

/home ("slash home")

/usr/local

/usr/bin

/home/wcorning

/home/eapoe

cd -

cd ~

cd ..

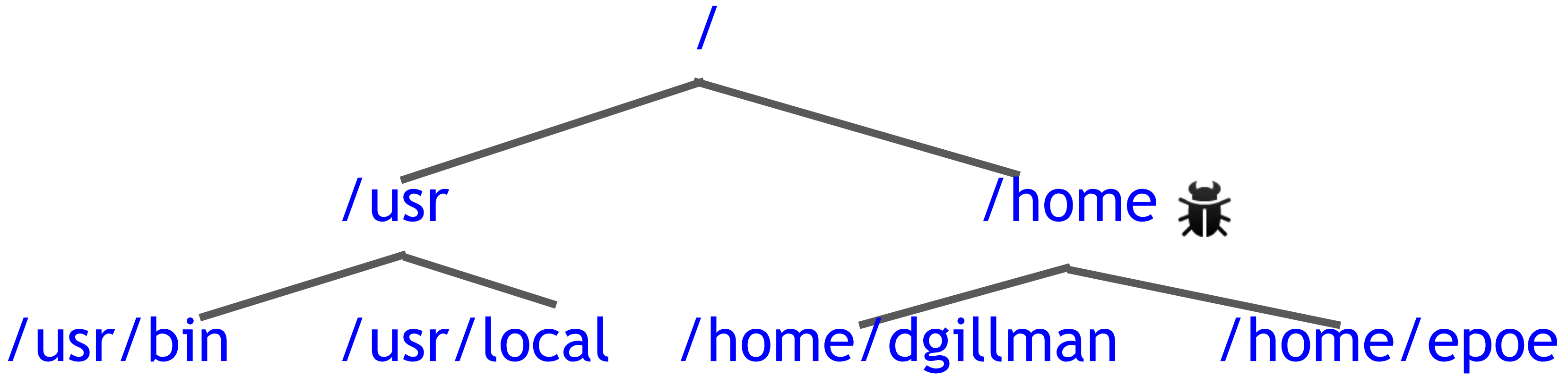
cd /home

Linux command line

pwd: where am I?

Linux command line

ls: list files and directories



/

/usr

/home 🐞

/usr/bin

/usr/local

/home/dgillman

/home/epoe

```
$ ls
dgillman wcorning
$ ls /usr
bin local
```

Shell

Command-line interface program

Linux: bash (Bourne-again shell)

It’s a program (executable file). Where is it?

```
$ which bash
```

Are all programs in /usr/bin? No.

How do you tell bash to run a program? Type its name.

How does bash know where to find the program? If you don’t give the full path, bash looks in the folders in PATH.

What’s PATH? A string variable (you can program in bash)

```
$ echo $PATH
```

Shell

To stop a program, type

Control key + c simultaneously -- “Ctrl-C” or “^C”

To exit the shell (or a looping script), Ctrl-D

File permission

bash permission (and other info)

```
$ ls -l /usr/bin/bash
-rwxr-xr-x. 1 root root 1219248 Nov  8  2019 /usr/bin/bash
```

```
-r-xr-xr-x
```

directory owner (u) group (g) other (o)

r: can read
w: can write
x: can execute

Example: `$ chmod u+x hello.py`

Linux command for reading a file

`cat` or `more` or `less`

`more` scrolls from top to bottom

- `<space>` to page down, `<return>` to line down

`less` is more: it scrolls up and down and searches

- `h` for help
- `q` to quit

Linux command

`grep`

`grep` searches for a string in a file, returns matching lines, matches limited regular expressions

`egrep` (extended `grep`) is a fully functional regular expression search tool.

Command syntax “meta-syntax”

Or, How to find out how to use a command

```
man command
```

Meta-syntax varies:

- `command [optional arguments] required argument`
- `<command> [optional arguments] <required argument>`

“Metasyntactic variables”

```
foo, bar, baz
```

How to log into foo.network.ncf.edu….

Save your hands!

History and command-line editing

`history`

Shows the last n commands

`↑, ctl-r`

Step backward, search backward for a command

`←, →, ctl-b, ctl-f, M-b, M-f`

Navigate in the current line

`Backspace, ctl-d, ctl-k, M-d, ctl-y`

Edit the current line

Google is your friend

What’s the command for that, again?

Don’t be afraid to search *linux copy file* if you need to

- But, once you know the syntax, check for what it does!

Database CRUD

Four types of database operations

- Create: table (creates empty table), index
- Read: table <- today
- Update: table (insert or alter data or schema)
- Delete: table, index

PostGres (PostGreSQL)

A Relational Database

Seven Databases, chapter 2

- Create a database:

```
$ createdb book
```

- Enter the Postgres command-line shell:

```
$ psql book
```

Relational database:

Contains tables ("relations")

Tables contain rows (key values and their attributes)

psql client

- Meta commands begin with \

`mydb=> \?`	help with meta commands
`mydb=> \d (\l)`	list tables (list databases)

- SQL

`mydb=> \h`	help with SQL

- Exit the shell

```
\q
```

What if you make a mistake?

1. Delete the database.

```
mydb=# drop database mydb;
```

2. Delete a table. (Need to recreate it.)

```
mydb=# drop table foo;
```

2. Clear a table of all its data.

```
mydb=# delete from foo;
```

2. Delete only certain records (rows) from a table.

```
mydb=# delete from foo where bar = 6;
```

Chronic homelessness:
What is its impact on the Tampa jail?

Hillsborough County Sheriff's Office

Queried and scraped
(scrape: download text from a web page)

booking_dates:	without arrest details
bookings:	with arrest details

Read

The R in CRUD

```
select <columns>
from <table>
[where <condition>];
```

Homelessness Data

Start exploring in Postgres

… and exploring Postgres in the process:

`psql homelessness` - open Postgres client in homelessness database

`\? (\h)` - what Postgres meta-commands (SQL commands) are there?

`\d (\l)` - what tables (databases) are there?

`\d <table>` - what's in this table?

`\c <database>` - switch to new database

Questions about the data?

1. How many unique arrestees are there?
2. What is the difference between the two arrestees tables?
3. What's the date range of arrests?
4. How many arrests had more than one charge?
5. What's the most common charge?
6. Do homeless people get arrested on nuisance charges more than other people?
7. Are there places where homeless people are more likely to get arrested?

[dataset][questions about dataset]

[dataset/world][methodology question - we could have asked/answered this before collecting data]

[world][can answer with data?] [can ask without data - has to do with the world]

What questions can the data answer?

1. How many unique arrestees are there?
    - How to id an arrestee?
2. How many arrests had more than one charge?
    - Do the arrests list charges?
3. What’s the most common charge?
    - Per arrest or per charge?
4. Do homeless people get arrested on nuisance charges more than other people?
    - Can I tell if someone is homeless?
5. Are there places where homeless people are more likely to get arrested?
    - Does the data say where the arrest occurred?

Questions about the data?

1. Empty arrestees table
2. Why are there more charges than bookings?
3. Why are there fewer booking_dates than bookings?
4. What signifies homelessness? Lack of address?
5. Can homeless status be given by the data?
6. Who has been arrested over ten times?
7. How many arrests are there by race, gender, or ethnicity?
8. What do the column names mean?

[dataset][questions about dataset]
[dataset/world][methodology question - we could have asked/answered this before collecting data]
[world][can answer with data?] [can ask without data - has to do with the world]

What questions can the data answer?

1. Who has been arrested over ten times?
   - Can you identify the person arrested from the data?
2. How many arrests are there by race, gender, or ethnicity?
   - What are “race”, “gender”, and “ethnicity” in this data?
3. What is the average length of stay in jail?
   - Can you subtract dates? How?
4. Where do most of the arrests take place?
   - Does the data identify location of arrest? What is “address”?
5. What do homeless people get charged with?
   - Does the data say who is homeless? Is there a way to guess?
6. How many arrests were there for “disorderly conduct” in 2010?
   - Where is charge? Where is arrest year?

`where` Conditions

Select out certain rows

Pose any condition out of curiosity - there’s a where clause for that

```
select * from booking_dates where name = 'MAN, FLORIDA';
```

boolean expression (true or false)

Who was arrested on Christmas?

```
select * from booking_dates where arrestdate = '2019-12-25';

select * from booking_dates where arrestdate in
('2015-12-25','2014-12-25','2013-12-25');
select * from booking_dates where arrestdate like ‘%-12-25’;
```

Which Christmases were there arrests?

select distinct arrestdate from booking_dates where arrestdate like '%-12-25';

What exactly does 'distinct' do?

It makes sure no two *rows* are the same.

```
select distinct name, arrestdate from booking_dates where arrestdate like
'%-12-25';
              name               | arrestdate
---------------------------------+------------
 ABBOTT,STEVEN CRAIG             | 2009-12-25
 ACOSTA,ERIK WILFREDO            | 2009-12-25
```

Same dates, distinct rows.

`like`

matching strings

Has Florida Man been arrested in Tampa?

```
select * from booking_dates
where name like '%FLORIDA%';
```

```
select * from booking_dates
where name ilike '%florida%';
```

Aggregate functions

`sum(), avg(), min(), max()`

```
select name, min(arrestDate)
from bookings
group by 1 (i.e., group by 1st selected column: name)
limit 5;
```

Select statement with
Aggregation, Group By, Order By, Limit

```
select min(arrestdate), max(arrestdate) from booking_dates;
```

```
select name, sum(1) from booking_dates group by 1 limit 5;
                                               (^ column 1)
```

```
select soid, name, sum(one) from
  (select soid, name, 1 one from bookings)
group by 1, 2
order by 3
limit 10;
```

Shorthand for `sum(one):` `sum(1)` or `count(*)`

(`count(*)` ignores `null` rows - we’ll see them shortly.)

Arrests per person

```
select name, count(*) from booking_dates group by 1 order by 2 desc limit
5;
           name            | count
---------------------------+-------
 KELLY,WILLIAM MICHAEL     |   157
 PARSON,JOHN               |   113
 MASTERS,PRESTON EUGENE    |   112
 SKILLEN,ALBERT GAY        |   108
 HARRIS,FREDERICK TIMOTHY  |   101
```

Bug: Two people can have the same name
(Use SOID! But it’s not in this table! What to do?)

Arrests per person

Q: Number of arrests per person, by name?

Difficulties

- Two people can have the same name
- Can we use SOID? It’s not in this booking_dates
- …?

Copy (\Copy)

`\copy <table> to|from <file> [with] <options>;`

- `\copy` is client-based; `copy` is server-based
- `\copy` uses `copy`'s options

example: fill the `arrestees` table

Run psql from the directory containing the file with the arrestee data.

```
$ cd /usr/share/databases/Homelessness/Jail/sql
$ psql homelessness
homelessness=# copy arrestees from arrestees.csv
homelessness-# with csv, header;
```

`quote` option: Why are names quoted in arrestees.csv?

Inner (Comma) Join

pairs of rows from two tables that match condition

```
select <columns> from <table1>, <table2>
where <table1.field> = <table2.field>;
```

Q: Number of arrests per person, by name?

General SQL development strategy:

Start big, e.g. with no where clause:

```
select * from arrestees, booking_dates
where arrestees.name = booking_dates.name;
```

Pare down:

```
select soid, a.name from arrestees a, booking_dates bd
where a.name = bd.name;
```

Inner (Comma) Join

pairs of rows from two tables that match condition

```
select <columns> from <table1>, <table2>
where <table1.field> = <table2.field>;
```

Q: Number of arrests per person, by name?

General SQL development strategy:

Start big. E.g. start with no where clause:

```
select * from arrestees, booking_dates
where arrestees.name = booking_dates.name;
```

Pare down:

```
select soid, a.name from arrestees a, booking_dates bd
where a.name = bd.name;
```

Temporary tables

Arrests per person, by name

```
select soid, name, sum(1)
from soid_name_arrest
group by 1,2;
```

But `soid_name_arrest` doesn’t exist -- create it first:

```
homelessness=# create temporary table soid_name_arrest as
select soid, a.name name
from arrestees a, booking_dates bd
where a.name = bd.name;
SELECT 1393792
```

Nested tables

No need to create a temporary table

Arrests per person, by name

```
select soid, name, sum(1)
from
    (select soid, a.name name
     from arrestees a, booking_dates bd
     where a.name = bd.name) a
group by 1,2;
```

Inner (Comma) Join

Q: How many arrests were there for “disorderly conduct” in 2010?

Try this one yourself!

Inner (Comma) Join

Q: Number of arrests per person, by name, dob, ethnicity?

Try this yourself! What tables do we need?

Did we answer the question?

Q: Number of arrests per person, by name, dob, ethnicity?

Does the query give one row for each soid, name, dob, e?

Does the query give all soid, name, dob, e?

Does the query count all arrests for each soid, name, dob, e?

How to correct these problems?

What if you make a mistake?

1. Delete the database.

```
book=# drop database book;
```

2. Delete a table. (Need to recreate it.)

```
sqlite> drop table cities;
```

2. Clear a table of all its data.

```
sqlite> delete from prez1st;
```

2. Delete only certain records (rows) from a table.

```
sqlite> delete from prez1st where id = 6;
```

null

missing values

Q: How long do people stay in jail?

```
create temporary table booking_time as
select b.soid, bd.name, bd.arrestdate, b.releasedate
from bookings b, booking_dates bd
where b.bookingnumber = bd.bookingnumber;

select * from booking_time;
```

Left [Outer] Join

all rows from left table
paired with matching rows or null row

```
create temporary table booking_time as
select b.soid, bd.name, bd.arrestdate, b.releasedate
from bookings b left outer join booking_dates bd
~~where~~ on b.bookingnumber = bd.bookingnumber;
```

Join Aliases

```
select ... from a inner join b on...
select ... from a join b on...
select ... from a, b where... ("comma join")

select ... from a left outer join b on...
select ... from a left join b on…

select ... from a cross join b;
select ... from a, b; (where is optional, on isn't)
```

coalesce(a,b)

if a is null, b; else a

Get arrest date if possible; use release date if not.

```
select b.soid, bd.name,
 coalesce(bd.arrestdate, b.releasedate) arr_rel_date,
  b.releasedate ...
```

Get arrest date if possible; flag if not.

```
select b.soid, bd.name,
 coalesce(bd.arrestdate, 'ARREST DATE MISSING') arr_date,
  b.releasedate ...
```

count(expression)

result: number of non-null values

```
select count(releasedate) from bookings;
```

vs. count(*): number of non-null rows

```
select count(*) from bookings;
```

Number of rows per value of column:

What question does this query answer?

```
select name, count(*)
from bookings
group by name;
```

Operations on `null`

```
book=> insert into books values (1, null);
book=> select id, (title <> '') tfn from books;
 id | tfn
----+------
  1 | t
  1 |
  3 | f
(3 rows)

book=> select id, tfn from (select id, (title <> '')
tfn from books) a where tfn is not null;
 id | tfn
----+------
  1 | t
  3 | f
(2 rows)
```

Q: average number of charges per arrest by year

How to get year: (see special date functions and string functions)

```
select extract(year from arrestdate) yr from booking_dates;
```

Start big:

```
select bd.bookingnumber, extract(year from arrestdate) yr,
   c.charge
from booking_dates bd, charges c
where c.bookingnumber = bd.bookingnumber limit 5;
```

Q: average number of charges per arrest by year (cont.)

Pare down:

```
select bd.bookingnumber, extract(year from arrestdate) yr,count(*)
from booking_dates bd, charges c
where c.bookingnumber = bd.bookingnumber
group by 1, 2 limit 5;
```

Get answer:

```
select yr, sum(1) numArrests, round(avg(charges), 2) from
    (select bd.bookingnumber, extract(year from arrestdate) yr,
       count(*) charges
     from booking_dates bd, charges c
     where c.bookingnumber = bd.bookingnumber
     group by 1, 2) a
group by 1 order by 1;
```

Query took 5sec - why?

explain

`explain`: shows the query plan:

- how Postgres will read tables
- how Postgres will join tables
- cost estimates of planned steps

more later

```
explain select * from booking_dates bd, charges c
where c.bookingnumber = bd.bookingnumber;
                    QUERY PLAN
----------------------------------------------------------
 Hash Join (cost=125197.62..287712.65 rows=6749583 width=123)
```

Estimate of cost… 288K what?

Units of cost

explain analyze

shows the estimates and runs the query

```
explain analyze select * from booking_dates bd, charges c
where c.bookingnumber = bd.bookingnumber;
-----------------------------------------------------------
Hash Join  (cost=125197.62..287712.65 rows=6749583 width=123)
(actual time=1604.019..4075.828 rows=2767040 loops=1)
 Planning Time: 0.456 ms
 Execution Time: 4193.551 ms
```

About 70,000 estimated cost units per second? Ballpark.

case and where...in

case = if… then

Q: How to define “homeless”?

```
select address, count(*) from bookings
group by 1 order by 2 desc limit 10;

create temporary table homelessaddresses(address text);

insert into homelessaddresses
select distinct address from bookings
where address in ('HOMELESS','UNK','UNKNOWN');
```

(Could use `union`)

case, cont.

Q: How to define “homeless”?

```
create temporary table booking_homeless as
select bookingnumber,
    case
      when address in (select address from homelessaddress)
      then true
      else false
     end homeless
...;
```

where... in (select...)

Q: Is every charge in `charges` associated with a booking in `bookings`?

```
select count(*) from charges where bookingnumber in (select
bookingnumber from bookings) ;
```

Alternatively: `exists`

```
select count(*) from charges where exists
 (select bookingnumber from bookings
  where bookings.bookingnumber = charges.bookingnumber);
```

where... not in (select…)

Slow -- an oddity of Postgres

```
select * from charges where not bookingnumber in (select
bookingnumber from bookings) ;
```

```
explain select * from charges where bookingnumber not in
(select bookingnumber from bookings) ;
                        QUERY PLAN
----------------------------------------------
 Gather  (cost=1000.00..24456271513.76 rows=1427924 width=60)
```

How many seconds? 24billion/100K = 240,000sec ~ 3 days

Table Constraints

Integrity, uniqueness guarantees

How can we avoid asking blue questions?

Constraints:

- no two are the same
- non-null
- foreign key references

```
create table bookings (
    bookingnumber integer primary key,
    agency text,
    …
    soid integer
);
```

More Table Constraints

Foreign keys, check

```
create table charges (
    bookingnumber integer references bookings,
    chargetype text check (chargetype != ''),
    charge text,
    court text,
    casenumber text
);
```

Should this table have a primary key, too? No.
Should charge be non-null?

Alter Table

Apply constraints after table creation

```
alter table charges add check (chargetype is not null);
alter table charges add check (charge is not null);

alter table arrestees primary key (soid);

alter table booking_dates foreign key (bookingnumber)
    references bookings (bookingnumber);
```