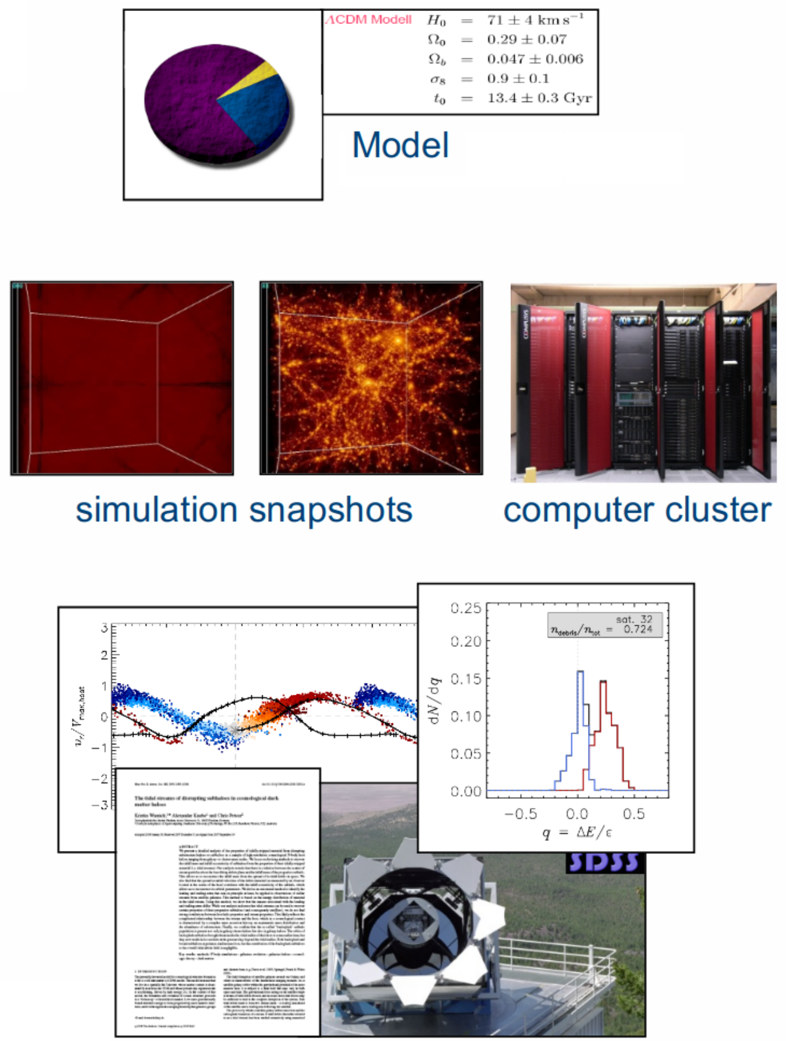
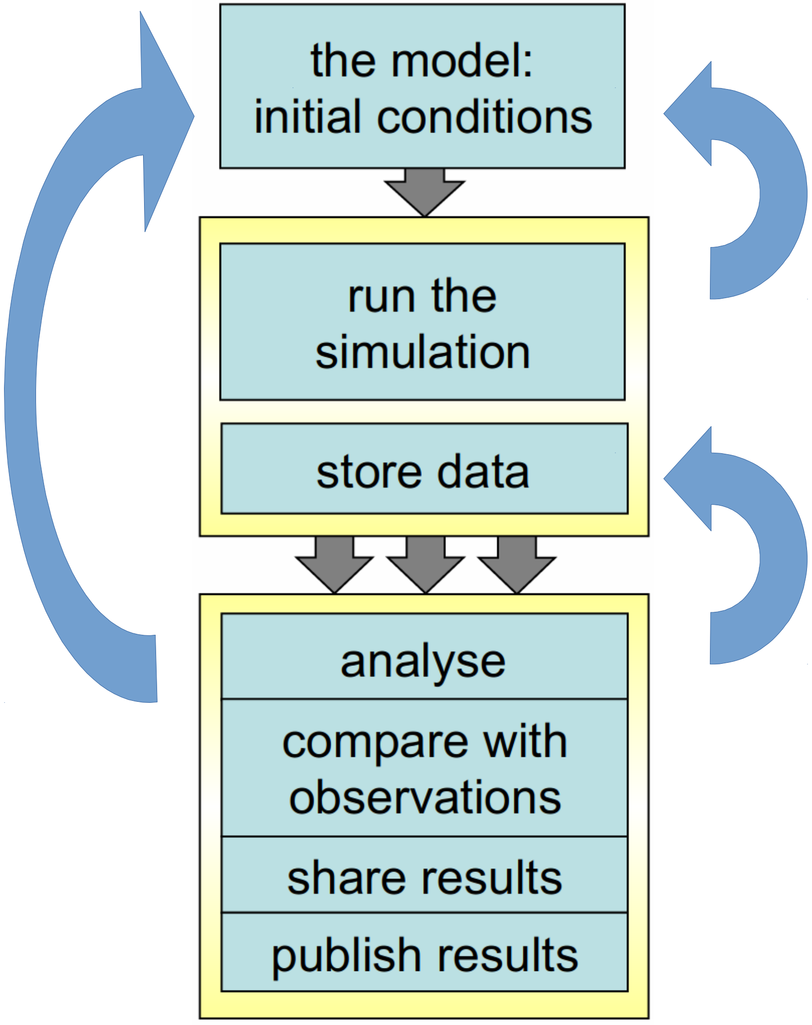
SQL Database for Research 101

University of Strathclyde

Motivation: workflow is clear, right?





the model: initial conditions

run the simulation

store data

analyse

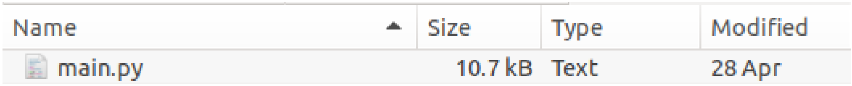
compare with observations

share results

publish results

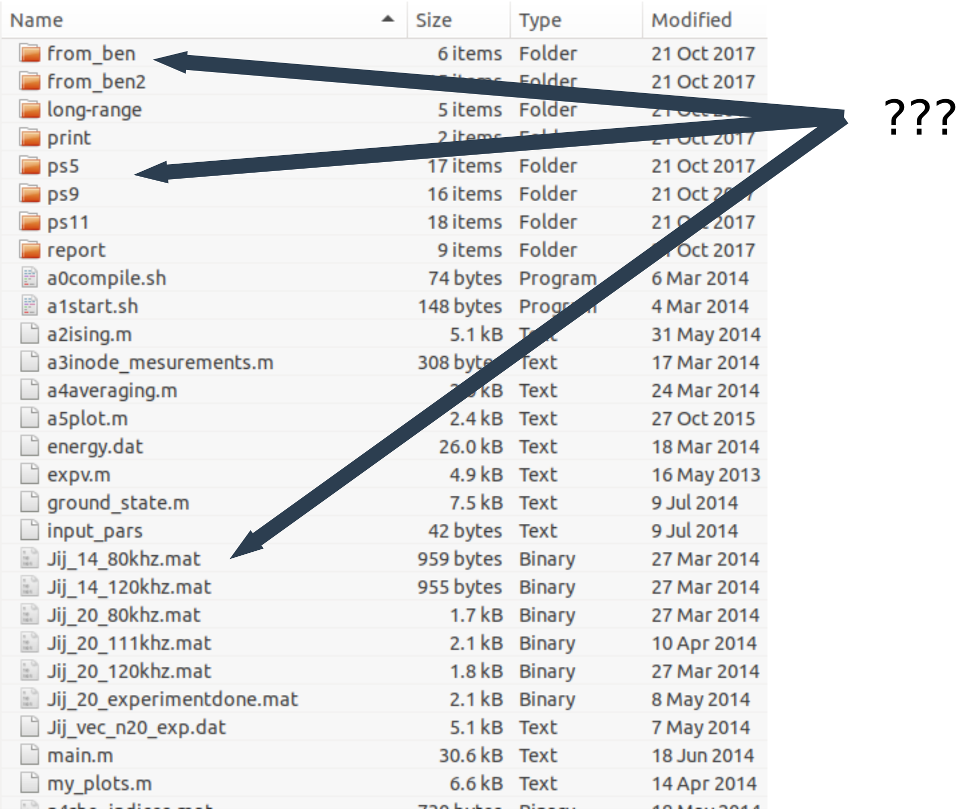
Motivation: organization is important

Project beginning



Name	Size	Type	Modified
main.py	10.7 kB	Text	28 Apr

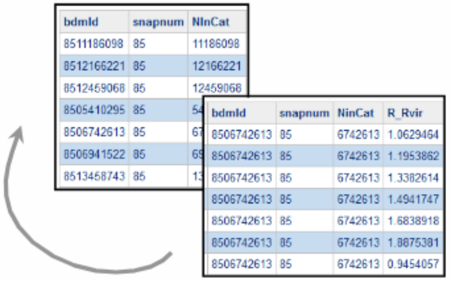
Project end



Name	Size	Type	Modified
from_ben	6 items	Folder	21 Oct 2017
from_ben2	[illegible] items	Folder	21 Oct 2017
long-range	5 items	Folder	21 Oct 2017
print	2 items	Folder	21 Oct 2017
ps5	17 items	Folder	21 Oct 2017
ps9	16 items	Folder	21 Oct 2017
ps11	18 items	Folder	21 Oct 2017
report	9 items	Folder	21 Oct 2017
a0compile.sh	74 bytes	Program	6 Mar 2014
a1start.sh	148 bytes	Program	4 Mar 2014
a2ising.m	5.1 kB	Text	31 May 2014
a3inode_mesurements.m	308 bytes	Text	17 Mar 2014
a4averaging.m	[illegible] kB	Text	24 Mar 2014
a5plot.m	2.4 kB	Text	27 Oct 2015
energy.dat	26.0 kB	Text	18 Mar 2014
expv.m	4.9 kB	Text	16 May 2013
ground_state.m	7.5 kB	Text	9 Jul 2014
input_pars	42 bytes	Text	9 Jul 2014
Jij_14_80khz.mat	959 bytes	Binary	27 Mar 2014
Jij_14_120khz.mat	955 bytes	Binary	27 Mar 2014
Jij_20_80khz.mat	1.7 kB	Binary	27 Mar 2014
Jij_20_111khz.mat	2.1 kB	Binary	10 Apr 2014
Jij_20_120khz.mat	1.8 kB	Binary	27 Mar 2014
Jij_20_experimentdone.mat	2.1 kB	Binary	8 May 2014
Jij_vec_n20_exp.dat	5.1 kB	Text	7 May 2014
main.m	30.6 kB	Text	18 Jun 2014
my_plots.m	6.6 kB	Text	14 Apr 2014

Simulation Databases can help

- store results of simulations in database, as tables and links between them
- Why?
  - simulations produce TB of data => hard to handle and share
  - post-processing results have variety of formats, individual software for reading
  - visibility of data?
  - reproducibility of data?



bdmId	snapnum	NInCat
8511186098	85	11186098
8512166221	85	12166221
8512459068	85	12459068
8505410295	85	54
8506742613	85	67
8506941522	85	69
8513468743	85	13

bdmId	snapnum	NinCat	R_Rvir
8506742613	85	6742613	1.0629464
8506742613	85	6742613	1.1953862
8506742613	85	6742613	1.3382614
8506742613	85	6742613	1.4941747
8506742613	85	6742613	1.6838918
8506742613	85	6742613	1.8875381
8506742613	85	6742613	0.9454057

Just get the subset you need, do (basic) calculations directly on the database server

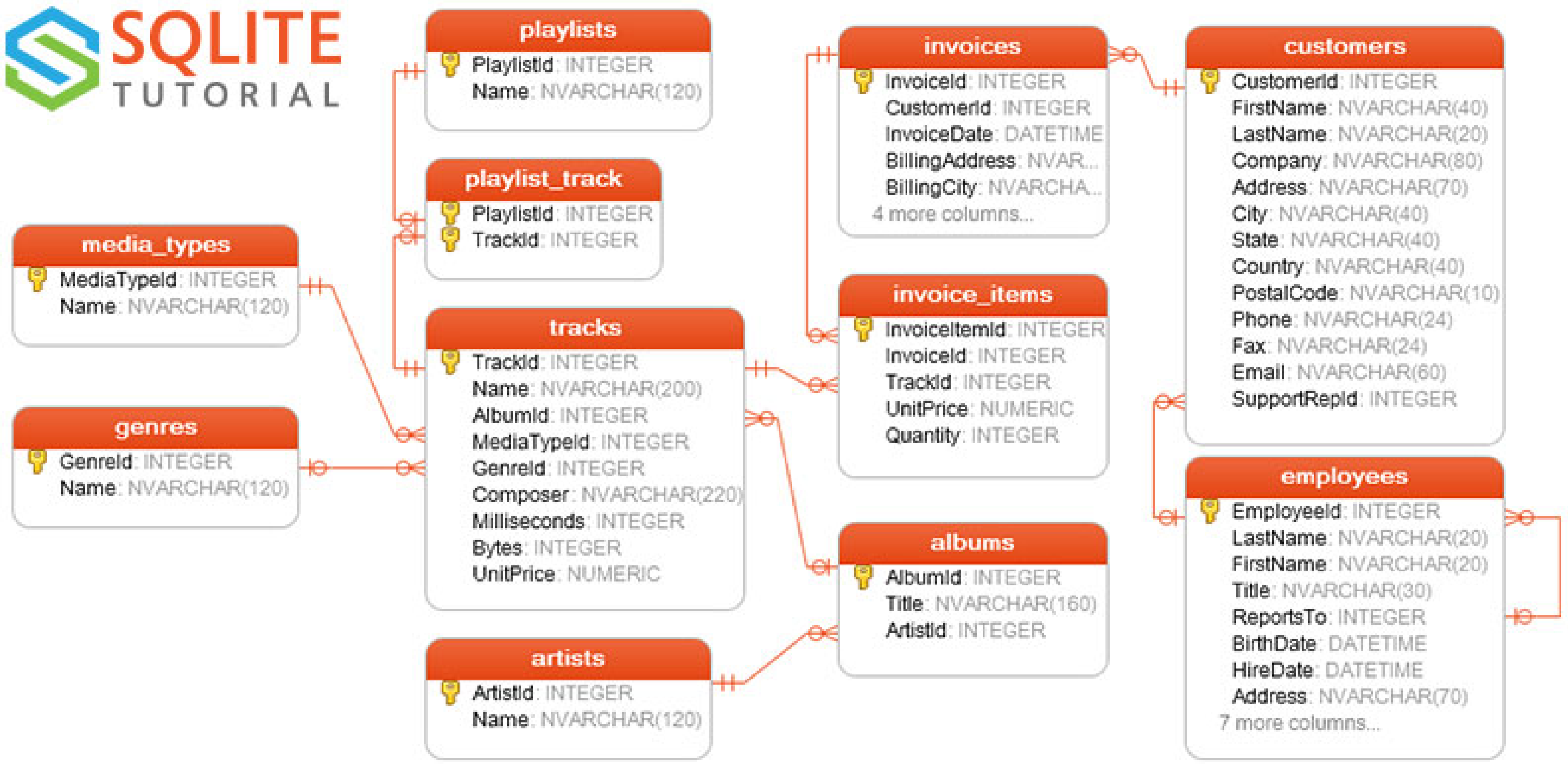
Uniform data format, SQL as standard

```
select top 20 * from MDR1..FOF
where snapnum=85
order by mass desc
```

extracts 20 most massive FOF groups at z=0

< 1 s

Example Database



SQLITE TUTORIAL

playlists
PlaylistId: INTEGER
Name: NVARCHAR(120)

playlist_track
PlaylistId: INTEGER
TrackId: INTEGER

media_types
MediaTypeId: INTEGER
Name: NVARCHAR(120)

genres
GenreId: INTEGER
Name: NVARCHAR(120)

tracks
TrackId: INTEGER
Name: NVARCHAR(200)
AlbumId: INTEGER
MediaTypeId: INTEGER
GenreId: INTEGER
Composer: NVARCHAR(220)
Milliseconds: INTEGER
Bytes: INTEGER
UnitPrice: NUMERIC

artists
ArtistId: INTEGER
Name: NVARCHAR(120)

invoices
InvoiceId: INTEGER
CustomerId: INTEGER
InvoiceDate: DATETIME
BillingAddress: NVAR...
BillingCity: NVARCHA...
4 more columns...

invoice_items
InvoiceItemId: INTEGER
InvoiceId: INTEGER
TrackId: INTEGER
UnitPrice: NUMERIC
Quantity: INTEGER

albums
AlbumId: INTEGER
Title: NVARCHAR(160)
ArtistId: INTEGER

customers
CustomerId: INTEGER
FirstName: NVARCHAR(40)
LastName: NVARCHAR(20)
Company: NVARCHAR(80)
Address: NVARCHAR(70)
City: NVARCHAR(40)
State: NVARCHAR(40)
Country: NVARCHAR(40)
PostalCode: NVARCHAR(10)
Phone: NVARCHAR(24)
Fax: NVARCHAR(24)
Email: NVARCHAR(60)
SupportRepId: INTEGER

employees
EmployeeId: INTEGER
LastName: NVARCHAR(20)
FirstName: NVARCHAR(20)
Title: NVARCHAR(30)
ReportsTo: INTEGER
BirthDate: DATETIME
HireDate: DATETIME
Address: NVARCHAR(70)
7 more columns...

SQL Language

- **SQL: Structured Query Language**
- **Originally called SEQUEL**
  - **S**tructured **E**nglish **Que**ry **L**anguage
  - Designed/Implemented at IBM Research for an experimental DBMS called **System R**.
- **SQL is now a standard**
  - American National Standards Institute (ANSI)
  - International Standards Organization (ISO)

There are different flavours of SQL, but their syntax is very similar

SQL Language

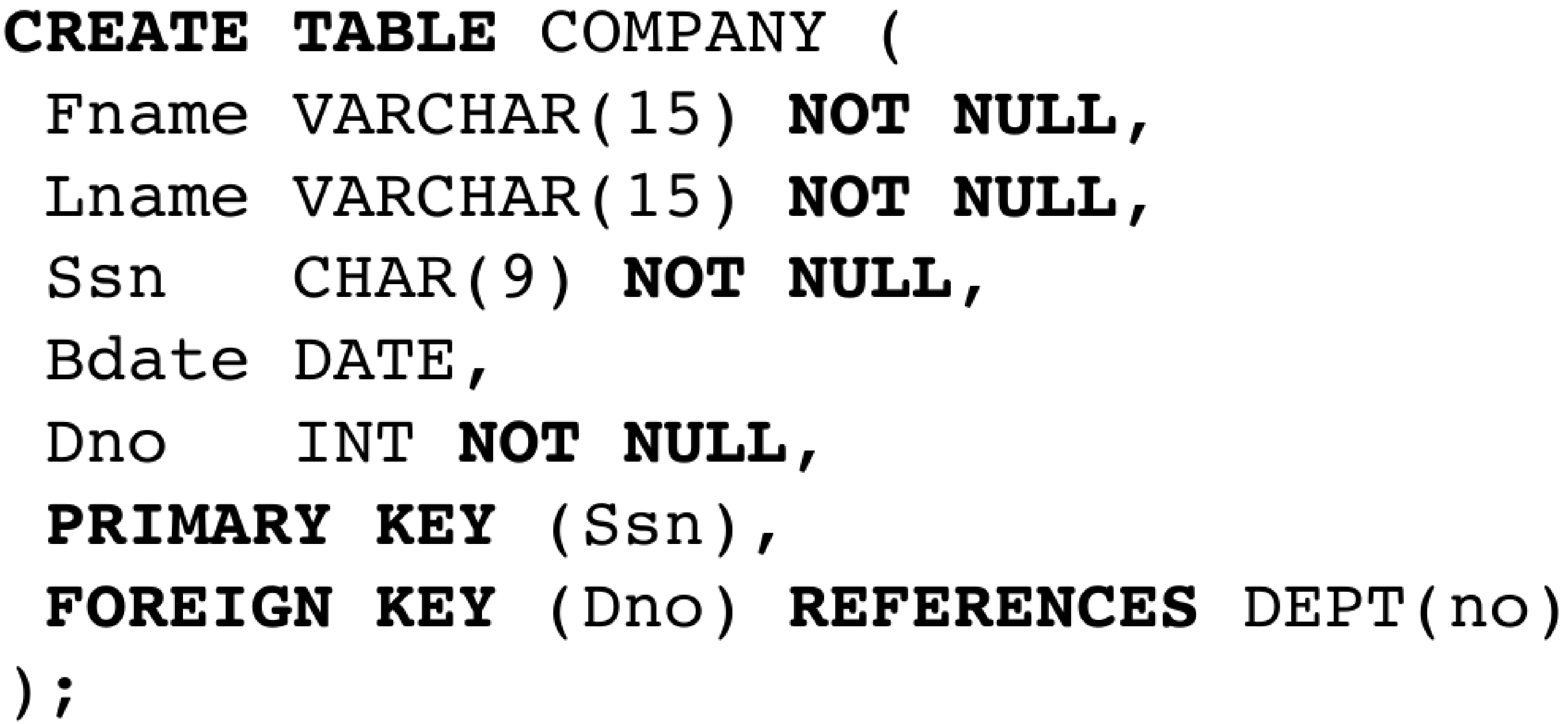
- **Declarative Language**

A user *only* specifies *what* the result is to be...

The database figures out *how* to retrieve the result!

This allows for greater flexibility in the language
and more opportunity for an SQL compiler
to optimize queries to achieve increased performance!

Create Table



```
CREATE TABLE COMPANY (
 Fname VARCHAR(15) NOT NULL,
 Lname VARCHAR(15) NOT NULL,
 Ssn   CHAR(9) NOT NULL,
 Bdate DATE,
 Dno   INT NOT NULL,
 PRIMARY KEY (Ssn),
 FOREIGN KEY (Dno) REFERENCES DEPT(no)
);
```

Data Types

- **Numeric**
- **Character string**
- **Bit string (BLOB)**
- **Boolean**
- **Time**

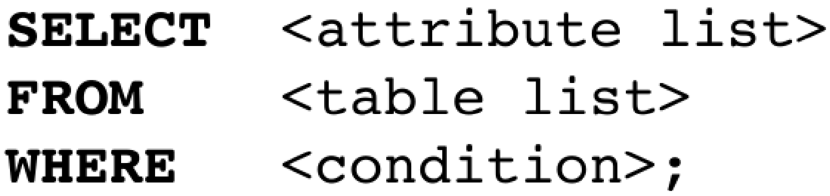
- integer
- float
- double
- quad
- ...

Non-standard formats can be saved in binary:

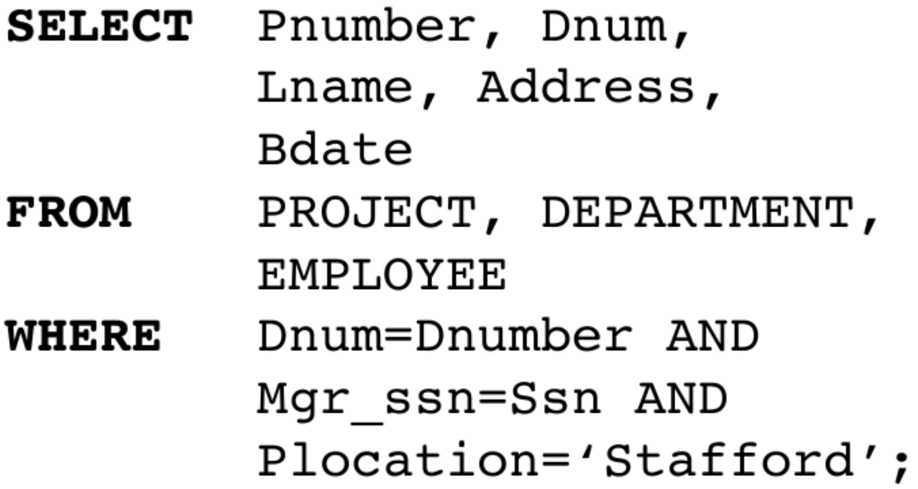
- images
- numpy arrays
- objects
- ...

Retrival Queries

SELECT-FROM-WHERE Structure



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



```
SELECT   Pnumber, Dnum,
         Lname, Address,
         Bdate
FROM     PROJECT, DEPARTMENT,
         EMPLOYEE
WHERE    Dnum=Dnumber AND
         Mgr_ssn=Ssn AND
         Plocation='Stafford';
```

Research application

- **All** modern languages have **A**pplication **P**rogramming **I**nterface for SQL databases
- Python solutions:
  - sqlite3 – interface SQLite
  - SQLAlchemy – Python SQL toolkit
  - pandas – data analysis toolkit
  - more…
- Let’s look at the practical…