MINING VIRTUAL UNIVERSES IN A RELATIONAL DATABASE

With examples from the (milli-)Millennium Run Database(s)

Gerard Lemson
MPA Garching, Germany
Matt’s categorization of questions

- Questions:
  - Phrase questions in terms of physics
  - Formulate question in terms of data
  - Translate in terms of existing tools

- Simple Questions
  - Can be answered using available data+tools

- Hard questions
  - Those for which tools do not exist

- Impossible
  - Those for which data is not sufficient
Use database to make questions simple(r)

- Provide data and tools for accessing them
  - yt is such an approach tailored for yt-like data

- We try to make questions as simple as possible
  - Data represented in a database using standard relational techniques
  - Query tool through online interfaces.
  - Standard query language SQL makes translating physics question into data question simple
It’s not always as simple as it seems. One should *try* to understand the data when using the database.

**Interpreting Millennium Run Halo Mergers:**
*Beware All Ye Who Enter Here*
by
Kevin Bundy, Tommaso Treu, Richard Ellis

Thanks to a tremendous effort by the Virgo Consortium, the astronomy community now has direct access via the internet to one of the largest cosmological dark matter simulations ever constructed. Using an SQL interface, the vast depths of the Millennium Simulation (Springel et al. 2005) can be plumbed using this website to retrieve information about the growth of dark matter structure as well as results on galaxy evolution as

See
Kevin Bundy, Tommaso Treu, Richard Ellis

http://astro.berkeley.edu/~kbundy/millennium/
Raw data:
- Particles

FOF groups and Subhalos

Mock images

Density fields

Subhalo merger trees

Mock catalogues

Synthetic galaxies (SAM)

ISSAC 2012 SDSC, San Diego, USA
MOTIVATION:
WHY RELATIONAL DATABASE
Analysis and Databases
(courtesy Alex Szalay)

- Much statistical analysis of data deals with
  - Creating uniform samples
  - Data filtering
  - Assembling relevant subsets
  - Estimating completeness
  - Censoring bad data
  - Counting and building histograms
  - Generating Monte-Carlo subsets
  - Likelihood calculations
  - Hypothesis testing

- Traditionally these are performed on files
- Most of these tasks are much better done inside a database
Relational database offers ...

- Encapsulation of data in terms of logical structure
  - no need to know about internals of data storage
- Standard query language for finding information
- Advanced query optimizers (indexes, clustering)
- Transparent internal parallelization
- Authenticated remote access for multiple users at same time

Especially important
- Forces one to think carefully about data structure
- Speeds up path from science question to answer
  - Makes more questions simple
- Facilitates communication
  - query code is clean(er)
RDB CONCEPTS
Relational database stores data in \textit{relations} (= tables)
Tables

- Tables have names
  - Full path: `<database-name>.[<schema-name>].<table-name>`

- Related data values are stored in rows

- Rows have columns
  - All the same for a given table

- Columns have names and data types
  - Data types have SQL names: SMALLINT, INTEGER, BIGINT, REAL, FLOAT, DECIMAL, CHAR(10), VARCHAR(100), CLOB, BLOB, DATETIME, TIME, TIMESTAMP, ...

- Rows often have a unique identifier consisting of the values of >= 1 columns: *primary key*
### Primary Key Column

- **galaxyid**
- **haloid**
- **descendantid**
- **redshift**
- **x**

### Foreign Key Columns

- **y**
- **z**
- **np**

<table>
<thead>
<tr>
<th>galaxyid</th>
<th>haloid</th>
<th>descendantid</th>
<th>redshift</th>
<th>x</th>
<th>y</th>
<th>z</th>
<th>np</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>0.0</td>
<td>6.5757904</td>
<td>13.08604</td>
<td>25.33813</td>
<td>51984</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0.019932542</td>
<td>6.587909</td>
<td>13.099106</td>
<td>25.301092</td>
<td>51288</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.041403063</td>
<td>6.597178</td>
<td>13.111782</td>
<td>25.252974</td>
<td>51052</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2</td>
<td>0.064493395</td>
<td>6.615912</td>
<td>13.12013</td>
<td>25.204876</td>
<td>51169</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>3</td>
<td>0.08928783</td>
<td>6.6276503</td>
<td>13.1303835</td>
<td>25.152872</td>
<td>50870</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>4</td>
<td>0.11588337</td>
<td>6.6414022</td>
<td>13.1400175</td>
<td>25.09534</td>
<td>50468</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>5</td>
<td>0.14436343</td>
<td>6.658701</td>
<td>13.149509</td>
<td>25.03174</td>
<td>50168</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>6</td>
<td>0.17489761</td>
<td>6.642237</td>
<td>13.170146</td>
<td>24.927555</td>
<td>50485</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>7</td>
<td>0.20754863</td>
<td>6.6424794</td>
<td>13.18374</td>
<td>24.83325</td>
<td>49888</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>8</td>
<td>0.24246909</td>
<td>6.6978354</td>
<td>13.176765</td>
<td>24.781622</td>
<td>48275</td>
</tr>
</tbody>
</table>
Database
- Many tables in >=1 schemas.
- Related through foreign keys
- Why so complex?
Normalization

- Consider storing galaxies, with info about their sub-halo as well as the FOF groups these live in. Note, a subhalo contains \(\geq 1\) galaxies, a FOF group \(\geq 0\) subhalos.
One table: redundancy

<table>
<thead>
<tr>
<th>galId</th>
<th>mStar</th>
<th>magB</th>
<th>X</th>
<th>haloid</th>
<th>np</th>
<th>hX</th>
<th>vMax</th>
<th>fofId</th>
<th>nSub</th>
<th>m200</th>
<th>fX</th>
</tr>
</thead>
<tbody>
<tr>
<td>112</td>
<td>0.215</td>
<td>-17.9</td>
<td>7.6</td>
<td>6625</td>
<td>100</td>
<td>7.6</td>
<td>165</td>
<td>123</td>
<td>2</td>
<td>445.77</td>
<td>7.6</td>
</tr>
<tr>
<td>113</td>
<td>0.038</td>
<td>-15.6</td>
<td>7.4</td>
<td>6625</td>
<td>100</td>
<td>7.6</td>
<td>165</td>
<td>123</td>
<td>2</td>
<td>445.77</td>
<td>7.6</td>
</tr>
<tr>
<td>154</td>
<td>0.173</td>
<td>-17.1</td>
<td>7.65</td>
<td>6626</td>
<td>65</td>
<td>7.9</td>
<td>130</td>
<td>123</td>
<td>2</td>
<td>445.77</td>
<td>7.6</td>
</tr>
<tr>
<td>221</td>
<td>1.20</td>
<td>-20.7</td>
<td>35.1</td>
<td>7883</td>
<td>452</td>
<td>35.1</td>
<td>200</td>
<td>456</td>
<td>2</td>
<td>101.32</td>
<td>35.1</td>
</tr>
<tr>
<td>223</td>
<td>0.225</td>
<td>-19.7</td>
<td>35.0</td>
<td>7883</td>
<td>452</td>
<td>35.1</td>
<td>200</td>
<td>456</td>
<td>2</td>
<td>101.32</td>
<td>35.1</td>
</tr>
<tr>
<td>225</td>
<td>0.04</td>
<td>-17.5</td>
<td>34.9</td>
<td>7883</td>
<td>452</td>
<td>35.1</td>
<td>200</td>
<td>456</td>
<td>2</td>
<td>101.32</td>
<td>35.1</td>
</tr>
<tr>
<td>278</td>
<td>1.54</td>
<td>-19.4</td>
<td>35.2</td>
<td>7884</td>
<td>255</td>
<td>35.2</td>
<td>190</td>
<td>456</td>
<td>2</td>
<td>101.32</td>
<td>35.1</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Normalization

### Galaxy

| gallId | haloid | mStar | magB | X  | ...
|--------|--------|-------|------|----|-----
| 112    | 6625   | 0.215 | -17.9| 7.6| ... |
| 113    | 6625   | 0.038 | -15.6| 7.4| ... |
| 154    | 6626   | 0.173 | -17.1| 7.65| ... |
| 221    | 7883   | 1.20  | -20.7| 35.1| ... |
| 223    | 7883   | 0.225 | -19.7| 35.0| ... |
| 225    | 7883   | 0.04  | -17.5| 34.9| ... |
| 278    | 7884   | 1.54  | -19.4| 35.2| ... |
| ...    | ...    | ...   | ...  | ...| ... |

### SubHalo

| haloid | fofId | Np  | X   | vMax | ...
|--------|-------|-----|-----|------|-----
| 6625   | 123   | 100 | 7.6 | 165  | ... |
| 6626   | 123   | 65  | 7.9 | 130  | ... |
| 7883   | 456   | 452 | 35.1| 200  | ... |
| 7884   | 456   | 255 | 35.2| 190  | ... |
| 9885   | 789   | 30  | 67.0| 110  | ... |
| ...    | ...   | ... | ... | ...  | ... |

### FOF

| fofId | nSub | m200  | X  | ...
|-------|------|-------|----|-----
| 123   | 2    | 445.77| 7.6| ... |
| 456   | 2    | 101.32| 35.1| ... |
| 789   | 1    | 70.0  | 67.0| ... |
| ...   | ...  | ...   | ...| ... |

ISSAC 2012 SDSC, San Diego, USA
DATABASE DESIGN
Data model features

- Each object its table
  - properties are columns
  - each a unique identifier
- Relations implemented through *foreign keys*,
  - pointers to unique identifier column
  - FOF to mesh cell it lies in
  - Subhalo to its FOF group
  - galaxy to its subhalo etc
- Special design needed for
  - Hierarchical relations: merger trees
  - Spatial relations: multi-dimensional indexes required
  - Support for random sample selection
Motivation for data model

1. Return the (B-band luminosity function of) galaxies residing in halos of mass between $10^{13}$ and $10^{14}$ solar masses.
2. Return the galaxy content at $z=3$ of the progenitors of a halo identified at $z=0$.
3. Return all the galaxies within a sphere of radius 3Mpc around a particular halo.
4. Return the complete halo merger tree for a halo identified at $z=0$.
5. Find positions and velocities for all galaxies at redshift zero with B-luminosity, colour and bulge-to-disk ratio within given intervals.
6. Find properties of all galaxies in haloes of mass $10^{14}$ at redshift 1 which have had a major merger (mass-ratio < 4:1) since redshift 1.5.
7. Find all the $z=3$ progenitors of $z=0$ red ellipticals (i.e. B-V > 0.8 B/T > 0.5).
8. Find the descendents at $z=1$ of all LBG’s (i.e. galaxies with SFR > 10 Msun/yr) at $z=3$.
9. Make a list of all haloes at $z=3$ which contain a galaxy of mass > $10^{10.5}$ Msun which is a progenitor of BCG's in $z=0$ cluster of mass > $10^{14.5}$.
10. Find all $z=3$ galaxies which have NO $z=0$ descendant.
11. Return the complete galaxy merging history for a given $z=0$ galaxy.
12. Find all the $z=2$ galaxies which were within 1Mpc of a LBG (i.e. SFR > 10Msun/yr) at some previous redshift.
13. Find the multiplicity function of halos depending on their environment (overdensity of density field smoothed on certain scale).
14. Find the dependency of halo formation times on environment (“Gao-effect”).
millimil database/schema @ISSACTAP

\[ \text{MMSnapshots} \rightarrow \text{MMField} \]

\[ \text{MMSnapshotids} \rightarrow \text{Snapshots} \rightarrow \text{DSubHalo} \leftarrow \text{Bower2006a} \]

\[ \text{Guo2010a} \leftarrow \text{SubHalo} \rightarrow \text{DeLucia2006a} \]

\[ \text{FOF} \rightarrow \text{MPAHalo} \rightarrow \text{DHalo} \rightarrow \text{Bower2006a} \]
Database tuning: Indexes

- Performance: disk IO is bottleneck
- Avoid it as much as possible, but can not store whole DB in memory
- To find rows of interest, avoid scanning complete tables
  - sequential scan \( \sim O(N) \)
  - \( \sim 10 \) min for galaxy tables (\( 10^9 \) rows, 250 GB)
- Binary search might speed up: requires ordering
  - \( \sim O(\log(N)) \)
- Can only order in one way
  - create external data structure
- INDEX
  - ordered according to \( \geq 1 \) columns, with direct pointer to row.
  - Bookmark lookup may be avoided
## Indexes

<table>
<thead>
<tr>
<th>galaxyid</th>
<th>haloid</th>
<th>descendantid</th>
<th>snapnum</th>
<th>redshift</th>
<th>coldgas</th>
<th>stellarMass</th>
<th>bulgeMass</th>
<th>mag_b</th>
<th>mag_v</th>
<th>mag_r</th>
<th>mag_i</th>
<th>mag_k</th>
<th>phkey</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>63</td>
<td>0.5</td>
<td>0.043965917</td>
<td>39.046864</td>
<td>34.046272</td>
<td>-22.992061</td>
<td>-22.992061</td>
<td>-23.712515</td>
<td>-23.712515</td>
<td>-25.889254</td>
<td>673</td>
<td>6.745794</td>
<td>13.08804</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>62</td>
<td>0.019032562</td>
<td>0.019032562</td>
<td>34.046864</td>
<td>34.046272</td>
<td>-22.992061</td>
<td>-22.992061</td>
<td>-23.712515</td>
<td>-23.712515</td>
<td>-25.889254</td>
<td>673</td>
<td>6.745794</td>
<td>13.08804</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>66</td>
<td>0.041403063</td>
<td>0.01826254</td>
<td>34.75536</td>
<td>33.75536</td>
<td>-22.04472</td>
<td>-23.052692</td>
<td>-23.76902</td>
<td>-23.76902</td>
<td>-25.926197</td>
<td>673</td>
<td>6.597178</td>
<td>13.1176</td>
</tr>
</tbody>
</table>

- **snapnum**, **stellarMass**, **galaxyid**
- **mag_b**
- **snapnum**, **x**
Databases we will consider

- Databases @ MPA
  - SQLServer:
    `<database>.[<schema>].<table>`

- Millimil @ ISSACTAP
  - Postgres:
    `<schema>.<table>`
  - Mirror of millimil+MMSnapshots @MPA

- Documented at
  - `http://gavo.mpa-garching.mpg.de/Millennium/Help`
Interfaces and Tools

- **Millennium Databases @GAVO**
  - [http://gavo.mpa-garching.mpg.de/Millennium](http://gavo.mpa-garching.mpg.de/Millennium)
  - [http://gavo.mpa-garching.mpg.de/MyMillennium](http://gavo.mpa-garching.mpg.de/MyMillennium) (auth, MyDB)
  - Wget, R, IDL

- **Millimil++ @SDSC**
  - TAP interface (M. Egger @MPA)
  - psql (hands on sessions)

- **TOPCAT**
  - Millennium query interface
  - TAP client interface
  - Visualisation via SAMP
QUERYING THE DATABASE: SQL
SQL

- **Sequential Query Language**
- Filtering, combining, sub-setting of tables
- Functions, procedures, aggregations
- Data manipulation: insert/update/delete
- A query produces tabular results, which can be used as tables again in sub-queries, or stored in a database
- Table creation...
Table creation statement

create table MPAHalo (  
    haloId bigint not null,  
    descendantId bigint , -- foreign key  
    lastProgenitorId bigint , -- foreign key  
    snapnum integer, redshift real,  
    x real,y real,z real,  
    np integer, velDisp real, vmax real,  
    ...,  
    primary key (haloId)  
) ;
SELECT ... FROM ... WHERE ...

1.
```
select *
from millimil.snapshots
```

2.
```
select snapnum, redshift, np
from millimil.MPAHalo
```

3.
```
select *
from millimil.MPAHalo
where redshift = 0
```
WHERE conditions

0 = <> != < > <= >=
0 np between 100 and 200
0 name like 'Primack'
0 a=b and d=e
0 a=b or e=d
0 id in (1,2,3)
0 a is null
0 a is not null
0 exists ... (later)
Find galaxies in a slice in X,Y,Z at redshift 0
select snapnum as snapshotIndex, redshift as z, np as numberOfParticles from millimil.MPAHalo

<table>
<thead>
<tr>
<th>snapshotIndex</th>
<th>z</th>
<th>numberOfParticles</th>
</tr>
</thead>
<tbody>
<tr>
<td>63</td>
<td>0.0</td>
<td>51984</td>
</tr>
<tr>
<td>62</td>
<td>0.019932542</td>
<td>51288</td>
</tr>
<tr>
<td>61</td>
<td>0.041403063</td>
<td>51052</td>
</tr>
<tr>
<td>60</td>
<td>0.064493395</td>
<td>51169</td>
</tr>
<tr>
<td>59</td>
<td>0.08928783</td>
<td>50870</td>
</tr>
<tr>
<td>58</td>
<td>0.11588337</td>
<td>50468</td>
</tr>
<tr>
<td>57</td>
<td>0.14438343</td>
<td>50168</td>
</tr>
<tr>
<td>56</td>
<td>0.17489761</td>
<td>50485</td>
</tr>
<tr>
<td>55</td>
<td>0.20754863</td>
<td>49888</td>
</tr>
<tr>
<td>54</td>
<td>0.24246909</td>
<td>48275</td>
</tr>
</tbody>
</table>
Color-magnitude for random sample of galaxies

\[
\text{select } \text{mag}_{\text{bdust}} - \text{mag}_{\text{vdust}} \text{ as } B_V
\]

from mpagalaxies..delucia2006a

where snapnum = 63

and random between 100000 and 101000
Aggregation: count, sum, max, min, avg, stddev

```
select count(*) as num, max(stellarmass) as maxmass, avg(stellarmass) as avgmass
from millimil.delucia2006a
where snapnum = 63 and type = 1
```
```sql
select h.*
from Halos h
order by h.snapnum desc, h.x asc
```
```sql
SELECT redshift, type, COUNT(*) AS numGal, AVG(stellarMass) AS m_avg, MAX(stellarMass) AS m_max
FROM millimil.DeLucia2006a
GROUP BY redshift, type
ORDER BY redshift, type
```
FOF multiplicity function, $z=0,1,2,3$

\[
\text{select snapnum, .1*floor(log10(np)/.1) as lognp, count(*) as num from mfield..fof where snapnum in (63,41,32,27) group by snapnum, .1*floor(log10(np)/.1) order by 1,2}
\]
JOINs

```sql
select h.haloid, g.stellarMass
from millimil.guo2010a g, millimil.mpahalo h
where h.np = 1000
and g.haloid = h.haloid
```

---

<table>
<thead>
<tr>
<th>Guo2010a</th>
<th>MPAHalo</th>
</tr>
</thead>
<tbody>
<tr>
<td>gallid</td>
<td>haloid</td>
</tr>
<tr>
<td>112</td>
<td>6625</td>
</tr>
<tr>
<td>113</td>
<td>6625</td>
</tr>
<tr>
<td>154</td>
<td>6626</td>
</tr>
<tr>
<td>221</td>
<td>7883</td>
</tr>
<tr>
<td>223</td>
<td>7883</td>
</tr>
<tr>
<td>225</td>
<td>7883</td>
</tr>
<tr>
<td>278</td>
<td>7884</td>
</tr>
</tbody>
</table>
Galaxies in massive halos

```sql
select h.haloId, g.*
from millimil.DeLucia2006a g
, millimil.MPAHalo h
where h.snapnum = 63
and h.np between 10000 and 11000
and g.haloId = h.haloId
```
Direct progenitors of massive halos (self-join)

```
select prog.*
from MPAHalo prog,
    MPAHalo des
where des.haloId = prog.descendantId
    and des.np > 10000
    and des.snapnum = 63
```
Calculate the conditional luminosity function in B of galaxies in FOF groups containing about 1000 particles at redshifts 0, 1, 2, 3.

```sql
SELECT f.snapnum AS B, COUNT(*) AS num
FROM mfield..fof
JOIN mfield..fofsubhalo sh ON f.fofid = sh.fofid
JOIN mpagalaxies..delucia2006a g ON sh.subhaloid = g.subhaloid
WHERE f.np BETWEEN 1000 AND 1010
AND f.snapnum IN (27, 32, 40, 63)
GROUP BY f.snapnum, .1*floor(g.mag_bDust/.1)
ORDER BY 1, 2
```

ISSAC 2012 SDSC, San Diego, USA
Sub-select

```sql
SELECT g.galaxyId, x, y, z, stellarMass
FROM millimil.DeLucia2006a g,
     (SELECT haloId
      FROM mpahalo
      WHERE snapnum = 63
      ORDER BY np DESC
      LIMIT 10) mh
WHERE g.haloId = mh.haloId;
```
Comparing 2 L-Galaxies models: galaxy by galaxy

DeLucia2006a vs Bertone2007a

\[ m_\text{bulge} / m_\ast \]

\[ V, B-V \]

ISSAC 2012 SDSC, San Diego, USA
Compare L-Galaxies (MPA) to GalForm (Durham) models: halo by halo
select ...
from mpagalaxies..delucia2006a d, dhalotrees..dhalo dsubhalo dsh, dgalaxies..bower2006a b
where d.snapnum = 63 and d.type = 0 and dsh.subhaloid = d.subhaloid and b.dhaloid = dsh.dhaloid and b.type = 0
Some special design features in the Millennium Databases

(next lecture?)

Identifiers
Environment
Trees
(Spatial)
Identifiers

- Parent-child relations reflected in identifiers avoid need for associative tables
  - FOFs in snapnums
    - $\text{fofId}=\text{snapnum} \times 10^{10} + \text{filenr} \times 10^6 + \text{rank-in-file}$
  - Subhalos in FOFs
    - $\text{subhaloid} = \text{fofId} \times 10^6 + \text{rank-in-fof}$
  - Particles in FOFs (mini-Mil-II)
    - $\text{particleId} = \text{fofId} \times 10^6 + \text{rank-in-fof}$
    - Global id for tracking of orbits
Environment

“find void galaxies”

- Environment as density field on $256^3$ grid
- Smoothed at various scales
  - CIC
  - G_5, G10
- Objects know their grid cell
select snapnum, .01*floor(f.g5/.01) as g5, count(*) as num
from mfield..mfield f
where f.snapnum in (63,41,32,27)
group by snapnum,.01*floor(f.g5/.01)
order by 1,2
FOF mass multiplicity function, conditioned on density in environment

```sql
select .1*floor(log10(fof.np)/.1) as lognp,
      count(*) as num
from mfield..mfield f,
     mfield..fof fof
where fof.snapnum=f.snapnum
    and fof.phkey = f.phkey
    and f.snapnum = 63
    and f.g5 between 1 and 1.1
group by .1*floor(log10(fof.np)/.1)
order by 1

(and similar for g5 = 0.5,2.5)
```
conditional multiplicity functions
\( \rho / \langle \rho \rangle = 0.5, 1, 2, 5 \)
Time evolution on merger trees

particles

halos
Galaxies

Table: mpagalaxies..delucia2006a
Galaxy ID = 415000584000000

[Diagram with scattered points and color bars for stellar mass and B-V.]
Trees in a database

- Recursion only partially supported
  - And not efficient
- Special solution
  - Indexing based on depth-first-order of progenitors
- Pointers to
  - descendant
  - last progenitor (finding all progenitors)
  - main leaf (finding main progenitors)
    - trees are getting very large ($10^8$)
    - branches ~100
  - tree root
    - finding descendants. indexing on intervals?
Main branches

- Track the object
- Pointer to main leaf
Merger trees:

```sql
select prog.*
from galaxies des,
     galaxies prog
where des.galaxyId = 0
    and prog.galaxyId between des.galaxyId and des.lastProgenitorId
```

Main progenitors:

```sql
select prog.*
from galaxies des,
     galaxies prog
where des.galaxyId = 0
    and prog.galaxyId between des.galaxyId and des.mainLeafId
```

Descendants:

```sql
select des.*
from galaxies des,
     galaxies prog
where prog.galaxyId = 41
    and des.galaxyId between prog.treeRootId and prog.galaxyId
    and prog.galaxyId between des.galaxyId and des.lastProgenitorId
```
Merger tree rooted in particular halo (in Millennium-II database)

```
MERGER TREE ROOTED IN PARTICULAR HALO (IN MILLENNIUM-II DATABASE)
```

```
SELECT p.mainleafid-d.mainleafid AS leaf, prog.* FROM millenniumII..halotree d, millennium..halotree p WHERE d.subhaloid = 670000003758000000 AND p.haloId BETWEEN d.haloId AND d.lastProgenitorId
```
Evolution of mass

Mass vs. Snapnum

Time
Find the dependency of halo formation times on environment $\delta_5 < Z_{\text{form}}$.

$\langle Z_{\text{form}} \rangle$
Spatial queries

Find all halos in a subvolume of space:

\[ 10 \leq x < 20 \]
\[ 20 \leq y < 30 \]
\[ 0 \leq z < 10 \]
select x,y,z
from millimil.mpahalo
where snapnum = 63
    and x between 10 and 20
    and y between 20 and 30
    and z between 0 and 10

Inefficient, even when indexed !
<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.001083</td>
<td>42.471325</td>
<td>24.673561</td>
</tr>
<tr>
<td>15.001247</td>
<td>58.420914</td>
<td>42.722874</td>
</tr>
<tr>
<td>15.002215</td>
<td>38.042484</td>
<td>29.557423</td>
</tr>
<tr>
<td>15.002735</td>
<td>50.487785</td>
<td>57.716877</td>
</tr>
<tr>
<td>15.002753</td>
<td>20.000177</td>
<td>8.21466</td>
</tr>
<tr>
<td>15.005095</td>
<td>13.637599</td>
<td>16.135191</td>
</tr>
<tr>
<td>15.006593</td>
<td>22.170828</td>
<td>48.242783</td>
</tr>
<tr>
<td>15.011488</td>
<td>24.824438</td>
<td>19.773285</td>
</tr>
<tr>
<td>15.011741</td>
<td>48.099907</td>
<td>11.500685</td>
</tr>
<tr>
<td>15.011868</td>
<td>23.312265</td>
<td>27.858799</td>
</tr>
<tr>
<td>15.013065</td>
<td>23.969515</td>
<td>18.883507</td>
</tr>
<tr>
<td>15.013158</td>
<td>56.041866</td>
<td>40.82894</td>
</tr>
<tr>
<td>15.014361</td>
<td>59.503357</td>
<td>45.31733</td>
</tr>
<tr>
<td>15.017322</td>
<td>46.257664</td>
<td>44.37695</td>
</tr>
<tr>
<td>15.018202</td>
<td>27.333895</td>
<td>9.441319</td>
</tr>
</tbody>
</table>
Spatial indexes

- Performance of finding things is improved if those things are co-located on disk: ordering, indices
- Co-locating a 3D configuration of points on a 1D disk can only be done approximately
- Space filling curves: Peano-Hilbert, Z-curve
- See Tamas’ talk
Simpler: Zones
Zone index

- Coarse sampling of points in multiple dimensions allows simple multi-dimensional ordering

- \( ix = \text{floor}(x/10\text{Mpc}) \)
- \( iy = \text{floor}(y/10\text{Mpc}) \)
- \( iz = \text{floor}(z/10\text{Mpc}) \)

- index on (snapnum,ix,iy,iz,x,y,z,galaxyId)
**ix=1 and iy=2 and iz=0**

<table>
<thead>
<tr>
<th>IX</th>
<th>IY</th>
<th>IZ</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.061804</td>
<td>20.891907</td>
<td>4.4156647</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.069336</td>
<td>23.437601</td>
<td>9.812217</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.100678</td>
<td>20.905642</td>
<td>4.613036</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.173968</td>
<td>22.368838</td>
<td>0.1832</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.194122</td>
<td>20.675834</td>
<td>8.034463</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.2500305</td>
<td>24.246683</td>
<td>1.6651521</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.365576</td>
<td>23.290754</td>
<td>9.404872</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.372606</td>
<td>20.203691</td>
<td>2.0006201</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.524696</td>
<td>21.039974</td>
<td>2.80077</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.583943</td>
<td>22.344622</td>
<td>9.421347</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.6358385</td>
<td>26.785904</td>
<td>9.881406</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.6638322.829983</td>
<td>7.137772</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.673803</td>
<td>26.918291</td>
<td>3.302736</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.717824</td>
<td>22.365341</td>
<td>9.221828</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.847992</td>
<td>24.700747</td>
<td>1.389664</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.883896</td>
<td>22.593819</td>
<td>7.277129</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.9104126.531118</td>
<td>2.5693457</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15.916905</td>
<td>27.137867</td>
<td>4.289855</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>16.047333</td>
<td>28.938115.414605</td>
<td>68</td>
</tr>
</tbody>
</table>
Back to Matt’s categorization of questions.

○ What are the hard questions in our approach?
  ● SQL does not support them though data does.
  ● Solution: download lots of our data, write your own code.
  ● Ask DB managers to add more functions to your DB.
    E.g. Spatial3D, many more @JHU

○ What are impossible questions?
  ● Not supported by our data.
  ● Solution:
    1. create your own data (L-Galaxies online, light-cones online etc.)
    2. Find it elsewhere (Next lecture, VO)
THANKS TO THE ORGANIZERS AND THANK YOU.

Acknowledgment:
Thanks to Matthias Egger for building the TAP interface. GL and Matthias Egger are supported by Advanced Grant 246797 GALFORMOD from the European Research Council.
Hands on 1

- Running SQL queries
- Teach TAP interface
- Teach TOPCAT interface