Storage

Mechanisms for persistent storage

In memory data structures

```
names = ['Alice', 'Bob', 'Charlie']
courses = ['Introduction to EE', 'Applied Mech', 'Calculus']
rels = [('Alice', 'Introduction to EE'),
        ('Bob', 'Calculus'),
        ('Alice', 'Calculus'),
        ('Charlie', 'Applied Mech')]
```

In memory data structures

```
names = ['Alice', 'Bob', 'Charlie']
courses = ['Introduction to EE', 'Applied Mech', 'Calculus']
rels = [('Alice', 'Introduction to EE'),
        ('Bob', 'Calculus'),
        ('Alice', 'Calculus'),
        ('Charlie', 'Applied Mech')]
```

- Error prone - easy to make mistakes in entry or referencing
- Does not scale
- Duplicate names?

In memory data structures - Keys

```
names = {0: 'Alice', 1: 'Bob', 2: 'Charlie'}
courses = {0: 'Introduction to EE',
           1: 'Applied Mech',
           2: 'Calculus'}
rels = [(0, 0),
        (1, 2),
        (0, 2),
        (2, 1)]
```

In memory data structures - Keys

```
names = {0: 'Alice', 1: 'Bob', 2: 'Charlie'}
courses = {0: 'Introduction to EE',
           1: 'Applied Mech',
           2: 'Calculus'}
rels = [(0, 0),
        (1, 2),
        (0, 2),
        (2, 1)]
```

- Data entry errors less likely
- Duplicates not a problem - Unique **Key**

Objects

```
class Student:
    idnext = 0 # Class variable
    def __init__(self, name):
        self.name = name
        self.id = idnext
        idnext = idnext + 1
```

- Auto-initialize ID to ensure unique
- Functions to set/get values

Objects

```
class Student:
    idnext = 0 # Class variable
    def __init__(self, name, hostel):
        self.name = name
        self.id = idnext
       self.hostel = hostel
        idnext = idnext + 1
```

- Add a new field to object easily

Persistence?

- In memory data structures lost when server shut down or restarted
- Save to disk? Structured data?
  - Python Pickle and similar modules
  - CSV - comma separated values
  - TSV - tab separated values
- Essentially same as spreadsheets: limited flexibility

Spreadsheet

- Naturally represent tabular data
- Extension, adding fields easy
- Separate sheet for relationships

Spreadsheet

- Naturally represent tabular data
- Extension, adding fields easy
- Separate sheet for relationships

Problems:

- Lookups, cross-referencing harder than dedicated database
- Stored procedures - limited functionality
- Atomic operations - no clear definition

Relational Databases - SQL

- From IBM ~ 1970s
- Data stored in Tabular format:
  - Columns of tables: fields (name, address, department, …)
  - Rows of tables: individual entries (student1, student2, …)

Unstructured databases - NoSQL

- Easily add/change fields
- Arbitrary data
- NoSQL
  - MongoDB
  - CouchDB
  - ….
- Flexible, but potential loss of validation