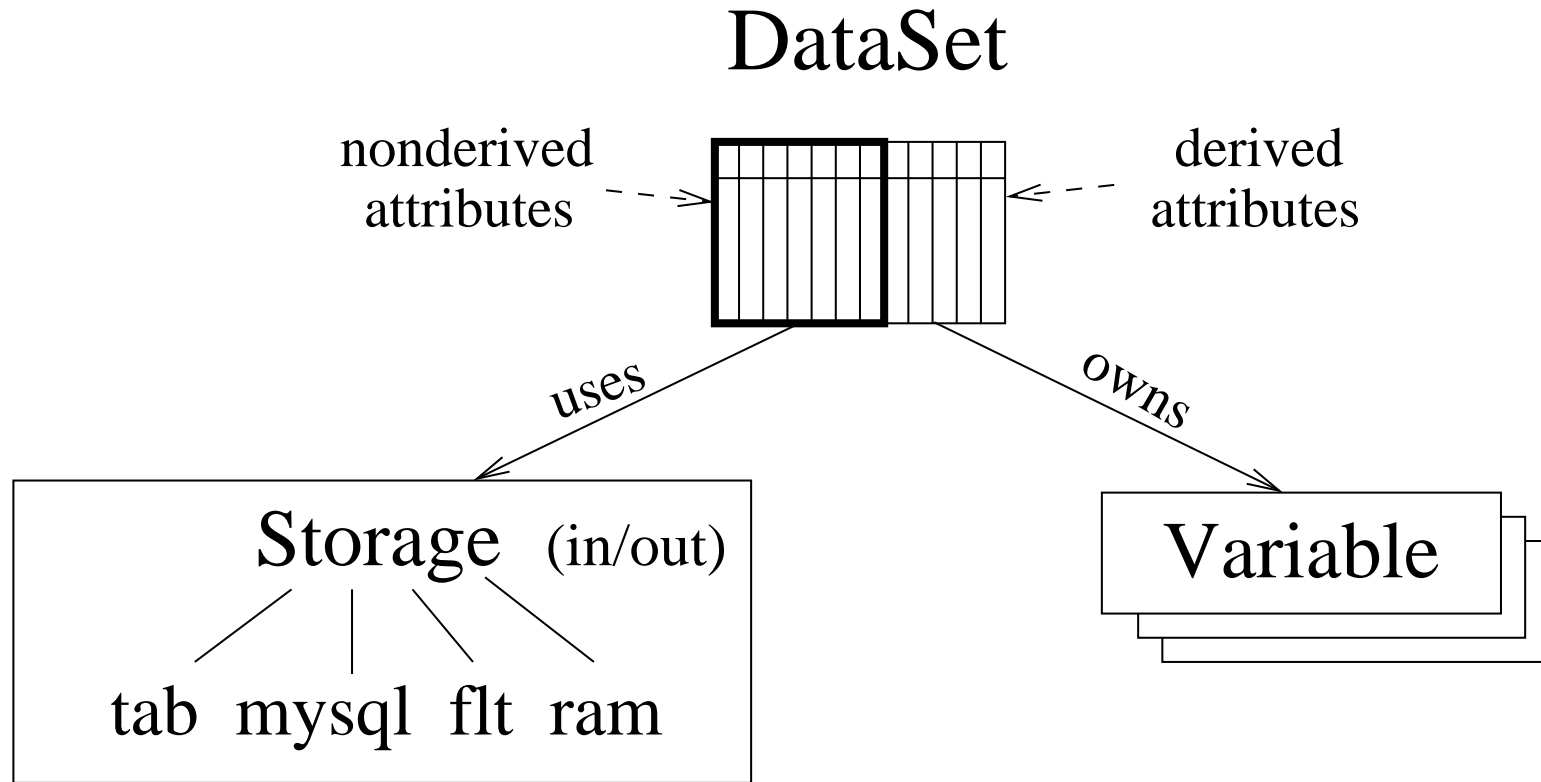


DataSet



DataSet

- Has a unique name (dataset name, entity name)

DataSet

- Has a unique name (dataset name, entity name)
- **Nonderived attributes:**
specified as **dataset_name.attribute_name**

DataSet

- Has a unique name (dataset name, entity name)
- **Nonderived attributes:**
specified as **dataset_name.attribute_name**
e.g. 'gridcell.distance_to_highway'

DataSet

- Has a unique name (dataset name, entity name)
- **Nonderived attributes:**
specified as **dataset_name.attribute_name**
e.g. 'gridcell.distance_to_highway'
- **Derived attributes (variables):**
specified as **package_name.dataset_name.attribute_name**

DataSet

- Has a unique name (dataset name, entity name)
- **Nonderived attributes:**
specified as **dataset_name.attribute_name**
e.g. 'gridcell.distance_to_highway'
- **Derived attributes (variables):**
specified as **package_name.dataset_name.attribute_name**
e.g. 'opus.urbansim.gridcell.is_near_highway'

DataSet

- Has a unique name (dataset name, entity name)
- **Nonderived attributes:**
specified as **dataset_name.attribute_name**
e.g. 'gridcell.distance_to_highway'
- **Derived attributes (variables):**
specified as **package_name.dataset_name.attribute_name**
e.g. 'opus.urbansim.gridcell.is_near_highway'
→ module of that name must exist (variable implementation)

DataSet

- Has a unique name (dataset name, entity name)
- **Nonderived attributes:**
specified as **dataset_name.attribute_name**
e.g. 'gridcell.distance_to_highway'
- **Derived attributes (variables):**
specified as **package_name.dataset_name.attribute_name**
e.g. 'opus.urbansim.gridcell.is_near_highway'
→ module of that name must exist (variable implementation)
e.g. opus/urbansim/gridcell/is_near_highway.py

DataSet

- Has a unique name (dataset name, entity name)
- **Nonderived attributes:**
specified as **dataset_name.attribute_name**
e.g. `'gridcell.distance_to_highway'`
- **Derived attributes (variables):**
specified as **package_name.dataset_name.attribute_name**
e.g. `'opus.urbansim.gridcell.is_near_highway'`
→ module of that name must exist (variable implementation)
e.g. `opus/urbansim/gridcell/is_near_highway.py`
- **InteractionSet**
 - DataSet with no nonderived attributes

DataSet

- Has a unique name (dataset name, entity name)
- **Nonderived attributes:**
specified as **dataset_name.attribute_name**
e.g. `'gridcell.distance_to_highway'`
- **Derived attributes (variables):**
specified as **package_name.dataset_name.attribute_name**
e.g. `'opus.urbansim.gridcell.is_near_highway'`
→ module of that name must exist (variable implementation)
e.g. `opus/urbansim/gridcell/is_near_highway.py`
- **InteractionSet**
 - DataSet with no nonderived attributes
 - Derived attributes stored as 2-D arrays,

DataSet

- Has a unique name (dataset name, entity name)
- **Nonderived attributes:**
specified as **dataset_name.attribute_name**
e.g. `'gridcell.distance_to_highway'`
- **Derived attributes (variables):**
specified as **package_name.dataset_name.attribute_name**
e.g. `'opus.urbansim.gridcell.is_near_highway'`
→ module of that name must exist (variable implementation)
e.g. `opus/urbansim/gridcell/is_near_highway.py`
- **InteractionSet**
 - DataSet with no nonderived attributes
 - Derived attributes stored as 2-D arrays,
specified as e.g.
`'opus.urbansim.household_gridcell.cost_to_income_ratio'`

Predefined datasets in UrbanSim

class name	dataset_name (default)	in_table_name (default)	id_name (default)
CitySet*	city	cities	city_id
ControlTotalSet (households)	control_total	annual_household_control_totals	year
ControlTotalSet (jobs)	control_total	annual_employment_control_totals	year, sector_id
CountySet*	county	counties	county_id
DevelopmentConstraintSet	development_constraint	development_constraints	constraint_id
DevelopmentEventSet	development_event	development_events	grid_id, scheduled_year
DevelopmentGroupSet	development_group	development_type_groups	group_id
DevelopmentProjectSet	development_project	–	project_id
DevelopmentTypeSet	development_type	development_types, development_type_group_definitions	development_type_id
EmploymentSectorGroupSet	employment_sector_group	employment_adhoc_sector_groups	group_id
EmploymentSectorSet	employment_sector	sector_id	employment_sectors
FazdistrictSet*	fazdistrict	–	fazdistrict_id
FazSet*	faz	fazes	faz_id
GridcellSet*	gridcell	gridcells	grid_id
HouseholdCharacteristicSet	household_characteristic	household_characteristics_for_ht	–
HouseholdSet	household	households	household_id
JobSet	job	jobs	job_id
LargeAreaSet*	large_area	large_areas	large_area_id
NeighborhoodSet*	neighborhood	neighborhoods	neighborhood_id
PlanTypeGroupSet	plan_type_group	plan_type_groups	group_id
PlanTypeSet	plan_type	plan_types	plan_type_id
RaceSet	race	race_names	race_id
RateSet (households)	rate	annual_relocation_rates_for_households	age_min, income_min
RateSet (jobs)	rate	annual_relocation_rates_for_jobs	sector_id
RecentDevelopmentEventSet	recent_development_event	development_event_history	grid_id, scheduled_year
TargetVacancySet	target_vacancy	target_vacancies	year
TravelDataSet	travel_data	travel_data	from_zone_id, to_zone_id
ZoneSet*	zone	zones	zone_id

Predefined interaction sets in UrbanSim

class name	dataset_name (default)
DevelopmentProjectGridcellSet	development_project_gridcell
HouseholdGridcellSet	household_gridcell
HouseholdNeighborhoodSet	household_neighborhood
HouseholdZoneSet	household_zone
JobGridcellSet	job_gridcell

Built-in Functions

- Unary transformations:

Built-in Functions

- Unary transformations:
sqrt, log, ln, ln_bounded, squared, powDDD

Built-in Functions

- Unary transformations:

sqrt, log, ln, ln_bounded, squared, powDDD

e.g. `'opus.core.func.sqrt(gridcell.distance_to_cbd) as sqrt_distance_to_cbd'`

Built-in Functions

- Unary transformations:

sqrt, log, ln, ln_bounded, squared, powDDD

e.g. `'opus.core.func.sqrt(gridcell.distance_to_cbd) as sqrt_distance_to_cbd'`

- Aggregation and disaggregation:

Built-in Functions

- Unary transformations:

sqrt, log, ln, ln_bounded, squared, powDDD

e.g. `'opus.core.func.sqrt(gridcell.distance_to_cbd) as sqrt_distance_to_cbd'`

- Aggregation and disaggregation:

over geographic units, e.g. county → zone → gridcell

Built-in Functions

- **Unary transformations:**

sqrt, log, ln, ln_bounded, squared, powDDD

e.g. `'opus.core.func.sqrt(gridcell.distance_to_cbd) as sqrt_distance_to_cbd'`

- **Aggregation and disaggregation:**

over geographic units, e.g. county → zone → gridcell

- **aggregate** (sum, mean, variance, standard_deviation, minimum, maximum, center_of_mass)

Built-in Functions

- **Unary transformations:**

sqrt, log, ln, ln_bounded, squared, powDDD

e.g. `'opus.core.func.sqrt(gridcell.distance_to_cbd) as sqrt_distance_to_cbd'`

- **Aggregation and disaggregation:**

over geographic units, e.g. county → zone → gridcell

- **aggregate** (sum, mean, variance, standard_deviation, minimum, maximum, center_of_mass)

`'zone:opus.core.func.aggregate(gridcell.capacity)'`

Built-in Functions

- **Unary transformations:**

sqrt, log, ln, ln_bounded, squared, powDDD

e.g. `'opus.core.func.sqrt(gridcell.distance_to_cbd) as sqrt_distance_to_cbd'`

- **Aggregation and disaggregation:**

over geographic units, e.g. county → zone → gridcell

- **aggregate** (sum, mean, variance, standard_deviation, minimum, maximum, center_of_mass)

`'zone:opus.core.func.aggregate(gridcell.capacity)'`

`'zone:opus.core.func.aggregate(opus.urbansim.gridcell.is_near_cbd, maximum)'`

Built-in Functions

- **Unary transformations:**

sqrt, log, ln, ln_bounded, squared, powDDD

e.g. `'opus.core.func.sqrt(gridcell.distance_to_cbd) as sqrt_distance_to_cbd'`

- **Aggregation and disaggregation:**

over geographic units, e.g. `county → zone → gridcell`

- **aggregate** (sum, mean, variance, standard_deviation, minimum, maximum, center_of_mass)

 - `'zone:opus.core.func.aggregate(gridcell.capacity)'`

 - `'zone:opus.core.func.aggregate(opus.urbansim.gridcell.is_near_cbd, maximum)'`

 - `'county:opus.core.func.aggregate(gridcell.capacity, sum, [zone])'`

Built-in Functions

- Unary transformations:

sqrt, log, ln, ln_bounded, squared, powDDD

e.g. `'opus.core.func.sqrt(gridcell.distance_to_cbd) as sqrt_distance_to_cbd'`

- Aggregation and disaggregation:

over geographic units, e.g. county → zone → gridcell

- **aggregate** (sum, mean, variance, standard_deviation, minimum, maximum, center_of_mass)

 - `'zone:opus.core.func.aggregate(gridcell.capacity)'`

 - `'zone:opus.core.func.aggregate(opus.urbansim.gridcell.is_near_cbd, maximum)'`

 - `'county:opus.core.func.aggregate(gridcell.capacity, sum, [zone])'`

- **aggregate_all** (sum, mean, ...) for alldata

Built-in Functions

- **Unary transformations:**

sqrt, log, ln, ln_bounded, squared, powDDD

e.g. `'opus.core.func.sqrt(gridcell.distance_to_cbd) as sqrt_distance_to_cbd'`

- **Aggregation and disaggregation:**

over geographic units, e.g. `county → zone → gridcell`

- **aggregate** (sum, mean, variance, standard_deviation, minimum, maximum, center_of_mass)

`'zone:opus.core.func.aggregate(gridcell.capacity)'`

`'zone:opus.core.func.aggregate(opus.urbansim.gridcell.is_near_cbd, maximum)'`

`'county:opus.core.func.aggregate(gridcell.capacity, sum, [zone])'`

- **aggregate_all** (sum, mean, ...) for alldata

`'opus.core.func.aggregate_all(gridcell.capacity)'`

Built-in Functions

- **Unary transformations:**

sqrt, log, ln, ln_bounded, squared, powDDD

e.g. `'opus.core.func.sqrt(gridcell.distance_to_cbd) as sqrt_distance_to_cbd'`

- **Aggregation and disaggregation:**

over geographic units, e.g. `county → zone → gridcell`

- **aggregate** (sum, mean, variance, standard_deviation, minimum, maximum, center_of_mass)

 - `'zone:opus.core.func.aggregate(gridcell.capacity)'`

 - `'zone:opus.core.func.aggregate(opus.urbansim.gridcell.is_near_cbd, maximum)'`

 - `'county:opus.core.func.aggregate(gridcell.capacity, sum, [zone])'`

- **aggregate_all** (sum, mean, ...) for alldata

 - `'opus.core.func.aggregate_all(gridcell.capacity)'`

- **disaggregate**

Built-in Functions

- **Unary transformations:**

sqrt, log, ln, ln_bounded, squared, powDDD

e.g. `'opus.core.func.sqrt(gridcell.distance_to_cbd) as sqrt_distance_to_cbd'`

- **Aggregation and disaggregation:**

over geographic units, e.g. `county → zone → gridcell`

- **aggregate** (sum, mean, variance, standard_deviation, minimum, maximum, center_of_mass)

 - `'zone:opus.core.func.aggregate(gridcell.capacity)'`

 - `'zone:opus.core.func.aggregate(opus.urbansim.gridcell.is_near_cbd, maximum)'`

 - `'county:opus.core.func.aggregate(gridcell.capacity, sum, [zone])'`

- **aggregate_all** (sum, mean, ...) for alldata

 - `'opus.core.func.aggregate_all(gridcell.capacity)'`

- **disaggregate**

 - `'gridcell:opus.core.func.disaggregate(county.population, [zone])'`

Built-in Functions (cont.)

- Number of agents:
number_of_agents

Built-in Functions (cont.)

- Number of agents:

number_of_agents

e.g. `'gridcell:opus.core.func.number_of_agents(household)'`