**DaySim Input Data Documentation**

The purpose of this chapter is to document the fields expected on the input data files to DaySim, with information on which ones are mandatory vs. optional, the formats, the relevant configuration settings, etc.

**An Overview of the DaySim Input/Output Structure**

The diagram below indicates that there are two main steps to importing data into DaySim. The first involves input of “raw” input data files into DaySims “RawConverter” class, which sets up various relationships between data items and adds some fields to certain files to convert them from the “raw” format to the standard DaySim input/output format for the specific data type. If the configuration command “ShouldRunRawConverter” is True, or if there are no files present in the user-specified working directory, then DaySim will run the RawConverter and write the files in specified text format to the working directory.

Raw Data Files

RawConverter

Working Text Files

Import and indexing

Working Binary Files

DaySim simulation run

Output Text Files

The second step is import and indexing. If the user specifies that a specific file should be imported, or if a new working file is created by running the RawConverter, or if there is no binary format file present in the working directory, then DaySim will import the file and write two binary files to the working directory (with .bin and .pk extensions). These binary imported files allow fast data access and cross-referencing during the model simulation run.

After a simulation is completed, DaySim writes output text files to the user-specified Output directory. A feature of DaySim is that the output text files have the same format as the working text files, so they could be used as input files to a subsequent run.

The types of files and the way they are dealt with are summarized below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **File Type** | **Use Raw-Converter?** | **Import?** | **Output?** | **Section # below** | **Comments** |
| Zone | Yes | Yes | No | 1 |  |
| Parcel | Yes | Yes | No | 2,3 | Buffering step is described in #3 |
| ParcelNode | Yes | Yes | No | 3 | Required if using node-node distance |
| NodeDistances | No | Direct | No | 3 | Required if using node-node distance |
| NodeIndex | No | Direct | No | 3 | Required if using node-node distance |
| TransitStopArea | Yes | Yes | No | 4 | Required if using stop areas skims |
| NodeStopAreaIndex | No | Direct | No | 5 | Required if using stop area skims |
| ParkAndRideNode | Yes | Yes | No | 6 | Required if using park & ride lot choice |
| IXXI | Yes | No | No | 7 | Data is transferred to the Zone file |
| Roster | No | Direct | No | 8 | CSV file listing input skim matrices |
| RosterCombinations | No | Direct | No | 8 | CSV file used with Roster |
| Household | Yes | Yes | Yes | 9 | Output in all application runs |
| Person | Yes | Yes | Yes | 10 | Output in all application runs |
| HouseholdDay | Yes\* | Yes\*\* | Yes | 11 | Output in all application runs |
| PersonDay | Yes\* | Yes\*\* | Yes | 12 | Output in all application runs |
| Tour | Yes\* | Yes\*\* | Yes | 13 | Output in all application runs |
| Trip | Yes\* | Yes\*\* | Yes | 14 | Output in all application runs |
| JointTour | Yes\* | Yes\*\* | Yes | 15 | Only relevant for H version runs |
| PartialHalfTour | Yes\* | Yes\*\* | Yes | 15 | Only relevant for H version runs |
| FullHalfTour | Yes\* | Yes\*\* | Yes | 15 | Only relevant for H version runs |
| TDMTripList | No | No | Yes | 16 | Customizable trip output file |

\* Raw inputs only required in Estimation mode (typically from household survey data)

\*\* Import only relevant for Estimation mode, or when using outputs from a previous run as inputs

For each of the file types where RawConversion is relevant (“Yes” in the “Use Raw Converter?” column above), there are two configuration inputs:

* Raw<FileType>Path (string file name and path)
* Raw<FileType>Delimiter (ASCII code, 9=tab, 32=space, 44=comma)

where <FileType> is substituted with the string in the FileType column above. These give the file names and delimiters for the raw input files. The assumed path for the files is given by the configuration input “BasePath”.

For each of the file types above where Import is relevant (“Yes” in the “Import?” column above), there are three additional configuration inputs:

* Import<FileType> (False or True)
* Input<FileType>Path (string file name and path)
* Input<FileType>Delimiter (ASCII code, 9=tab, 32=space, 44=comma)

where <FileType> is again substituted with the string in the FileType column above. These give the file names and delimiters for the working text files. The path for the working directory is specified by the configuration input “WorkingSubpath”, along with the “BasePath”.

If the configuration parameters above are not included, the default names of the working text files are “\_<FileType>.tsv”, and the default delimiter is a Tab. Import<FileType> is always set to true if a new working text file has been created (“ShouldRunRawConversion” is True), or if there are no binary files from previous imports (“\_<FileType>.bin” and “\_<FileType>.pk”) in the working directory.

A “safe” and relatively simple way to run DaySim is to always set “ShouldRunRawConversion” to “true” and to omit the “Import<FileType>xxx” configuration lines above. In that case, DaySim will always run both the raw conversion and import steps on all files, and will use the default working file names.

For each of the file types above where output is relevant (“Yes” in the “Output” column above), there are two more configuration inputs:

* Output<FileType>Path (string file name and path)
* Output<FileType>Delimiter (ASCII code, 9=tab, 32=space, 44=comma)

where <FileType> is again substituted with the string in the FileType column above. These give the file names and delimiters for the output text files. The path for the output directory is specified by the configuration input “OutputSubpath”, along with the “BasePath”.

The input files above that do not use the DaySim raw conversion and import conventions are NodeDistances, NodeIndex, NodeStopAreaIndex, Roster and RosterCombinations. Each of those files uses its own specific configuration settings and data reading code, and each is read in on every run, if specified.

Each data structure listed in the table above is defined in more detail in one of the sections below, with the section numbers listed in the Table.

1. **The raw “Zone” file**

This is often referred to as the “taz index” file. Its main purpose is to indicate to DaySim which zone numbers are valid. If there are gaps in the zone numbering (unused zone numbers), then this file is used to set up a mapping from the nominal zone numbers to an internal zone numbering that is used to compress the amount of memory used to store zone-to-zone skim matrices in memory.

**Relevant DaySim configuration parameters:**

* RawZonePath (string file name and path)
* RawZoneDelimiter (ASCII code, 9=tab, 32=space, 44=comma)

**Format**: ASCII delimited, with a header record.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| Zone\_ID | 1 – 9999999 | The zone ID number used in the network software that produces skims. Values must be unique positive integers, in ascending order. |
| Zone\_ordinal | 1 – 9999999 | A zone index number internal to DaySim, which is mapped to Zone\_id. Values must be unique positive integers in ascending order. Value will generally begin at 1 with no gaps in numbering, although gaps are allowed. |
| Dest\_eligible | 0 or 1 | A binary variable indicating whether or not a zone is eligible as a destination in Daysim. Zones that are not eligible as destinations are external zones or special park and ride lot zones. |
| External | 0 – 99 | This variable was originally used as a binary variable to indicate external zones, but was not used in the DaySim code, so it is now used to indicate a District mapping of the zones. Including a district mapping is optional – only necessary if ODShadowPricing is used, or other region-specific variables that are District-based |
| xcoord | 1 – 9999999 | **OPTIONAL** –only needed if Transit Stop Areas are used for transit skims. The x coordinate of the zone centroid, in integer length units (typically SPF). |
| ycoord | 1 – 9999999 | **OPTIONAL** –only needed if Transit Stop Areas are used for transit skims. The x coordinate of the zone centroid, in integer length units (typically SPF). |

1. **The unbuffered raw “Parcel” file**

The word “parcel” is used interchangeably for parcels or microzones, which are treated essentially the same by DaySim in terms of input data. (The word “parcel” is used in describing the files for legacy reasons, although the word “microzone” would be more appropriate—since a parcel can be a microzone, but a microzone is not always a parcel.)

The raw unbuffered parcel file is not input directly to DaySim, but is an input to the buffering process, which produces the buffered parcel file for DaySim.

**Format**: ASCII delimited, with a header record. “Real >=0” below indicates a non-negative number which does not need to be an integer value.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| parcelid | 1 – 9999999 | The parcel ID number. Values must be unique positive integers, in ascending order. (Gaps are allowed, but not efficient for memory.) |
| xcoord\_p | 1-999999999 | The x coordinate of the parcel centroid, in integer length units (typically SPF). |
| ycoord\_p | 1-999999999 | The y coordinate of the parcel centroid, in integer length units (typically SPF). |
| sqft\_p | 0-999999999 | The area of the parcel, in thousands of square length units (typically sqf, does not need to be an integer) |
| taz\_p | 1-9999999 | The zone that the parcel is in. Must be a valid zone\_id in the “zone” file |
| lutype\_p | 0-9999999 | A land use type value. This variable currently has no mandatory use in the DaySim code, so is available for region-specific usage. |
| hh\_p | Real >=0 | The number of households residing on a parcel. |
| stugrad\_p | Real >=0 | The number of grade school (K-8) students enrolled in schools on a parcel |
| stuhgh\_p | Real >=0 | The number of high school (9-12) students enrolled in schools on a parcel. If this is not available separately, then set to 0 & put the number of K-12 students in stugrd\_p |
| stuuni\_p | Real >=0 | The number of university/college students enrolled in schools on a parcel. |
| empedu\_p | Real >=0 | The number of educational employees working on a parcel |
| empfoo\_p | Real >=0 | The number of food service employees working on a parcel |
| empgov\_p | Real >=0 | The number of government employees working on a parcel |
| empind\_p | Real >=0 | The number of industrial employees working on a parcel |
| empmed\_p | Real >=0 | The number of medical employees working on a parcel |
| empofc\_p | Real >=0 | The number of (other) office employees working on a parcel |
| empret\_p | Real >=0 | The number of retail employees working on a parcel |
| empsvc\_p | Real >=0 | The number of (other) service employees working on a parcel |
| empoth\_p | Real >=0 | The number of other sector employees working on a parcel. Typically contains construction, agriculture, mining. |
| emptot\_p | Real >=0 | The total number of employees working on a parcel. Should equal the sum of the 9 previous fields. |
| parkdy\_p | Real >=0 | The number of paid public off-street parking spaces on a parcel with per day pricing. (May overlap with parkhr\_p if have both types of pricing.) |
| parkhr\_p | Real >=0 | The number of paid public off-street parking spaces on a parcel with per hour pricing. (May overlap with parkdy\_p if have both types of pricing.) |
| ppricdyp | Real >=0 | The average price of public off-street parking spaces on a parcel with per day pricing. (In cents per day) |
| pprichrp | Real >=0 | The average price of public off-street parking spaces on a parcel with per hour pricing. (In cents per hour) |

If the data is to be prepared at the microzone level, there are two ways to do that. The first is to aggregate up from parcel data. The second is to disaggregate from zone-level data to a Census block/zone intersection using zone level totals plus Census and LEHD household and employment data at the block level. RSG has created a standalone tool to do this step, and it also allows for different groupings and transformations of employment categories from whatever is available at the zone level to the 9 DaySim categories.

1. **The buffered raw “Parcel” file**

This is created through a series of buffering steps, which also create the node-to-node distances used in DaySim, if relevant.

There are currently three methods for doing short-distance calculations in both buffering (in RSG’s stand-alone buffering tool) and in DaySim:

1. Using X-Y coordinate orthogonal distance: This is the default option if no all-streets network is available or one wishes to avoid the effort of processing it
2. Using node-to-node distances: This is the most accurate and recommended option, using the shortest-path distance between nodes on an-all streets network rather than simple X-Y based distances.
3. Using circuity-based measures for each parcel: This method creates a “circuity surface” using 24 radially distributed points around each parcel centroid. Since this option requires the same all-streets network inputs as the node-to-node option (b), but is less accurate and requires an older stand-alone tool to create the measures, this option is not recommended, but it is still supported in DaySim.

If one choose the node-to-node distances method (b), then a few extra files must be input to DaySim:

**The raw “ParcelNode” file**

**Relevant DaySim configuration parameters:**

* RawParcelNodePath (string file name and path)
* RawParcelNodeDelimiter (ASCII code, 9=tab, 32=space, 44=comma)

**Format**: ASCII delimited, with a header record.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| parcelid | 1 - 9999999 | The parcel ID number. This file must have the same number of records as the raw parcel file, with the parcel IDs in the same order. |
| node\_id | 1-999999999 | The id of the nearest all-streets network node. This is an integer that does not need to be unique in this file. (Two or more parcels may share the same nearest node.) |

**The “NodeDistances” file**

**Relevant DaySim configuration parameters:**

* NodeDistancesPath (string file name and path)
* NodeDistancesDelimiter (ASCII code, 9=tab, 32=space, 44=comma, 0=binary, HD5 is also possible-need to look at latest settings to see how that is indicated in config file)

**Format**: ASCII delimited, with a header record

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| ANodeID | 1 - 9999999 | The ANode ID, from the all-streets network. For efficiency, should be a node present in the ParcelNode file. |
| BNodeID | 1 - 9999999 | The BNode ID, from the all-streets network. For efficiency, should be a node present in the ParcelNode file. The file must be sorted first by ANodeID and then by BNodeID within ANodeID. |
| Distance | 1-999999999 | The node-node shortest path distance, in length units (typically feet) |

**The “NodeIndex” file**

**Relevant DaySim configuration parameters:**

* NodeIndexPath (string file name and path)
* NodeIndexDelimiter (ASCII code, 9=tab, 32=space, 44=comma)

**Format**: ASCII delimited, with a header record

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| ANodeID | 1 - 9999999 | The ANode ID, from the all-streets network. For efficiency, should be a node present in the ParcelNode file. |
| FirstRecord | 1-999999999 | The position in the NodeNodeDistances file for the first record with ANodeID as the A node |
| LastRecord | 1-999999999 | The position in the NodeNodeDistances file for the last record with ANodeID as the A node (LastRecord >= FirstRecord) |

**Creating the buffered parcel file:** The buffering tool uses either the parcel/microzone centroid coordinates if method (a) is used, or the node-based files listed above if method (b) is used. The tool calculates buffered totals for two different buffers around each parcel (or microzone). The buffers use step functions or distance decay functions as specified by the user. For distance decay functions, there is a maximum buffer distance also provided as user input. In addition to the raw parcel file, the parcel-node correspondences and the node-to-node distances, the buffering uses 3 additional input files:

* 1. **Intersections:** A list of intersections, with node ID, XY coordinates, and number of links terminating at the node. (This can be the same file as the node list used above, but it should contain all intersections, regardless of whether or not it is a nearest node to any parcel, and it contains the extra information on number of links ending at the node.)
  2. **Transit stops:** A list of transit stops, with stop ID, XY coordinates, and the type of transit serving the stop (bus, LRT, etc.)
  3. **Open space areas (optional):** A list of parks, sports fields and other public recreational areas, with an ID and the size, in square feet. If this file is not provided, the buffered variables for open space will be 0.

The buffered “RawParcel” file contains the same data as the raw parcel file, but adds a number of new fields….

**Relevant DaySim configuration parameters:**

* RawParcelPath (string file name and path)
* RawParcelDelimiter (ASCII code, 9=tab, 32=space, 44=comma)
* UseShortDistanceNodeToNodeMeasures (set True if method (b) is used and the node distance files are available)
* UseShortDistanceCircuityMeasures (set True if method (c) is used and circuity measures are available on the parcel file. If UseShortDistanceNodeToNodeMeasures is also set True, it will override this one)
* MaximumBlendingDistance (the maximum distance, in miles, to which the short distance measures are used to adjust zone-to-zone skim-based travel times)
* DestinationScale( set to 0 if “parcels” are actually parcels, 1 if “parcels” are microzones—such as Census blocks—or set to 2 if “parcels” are the same as TAZs – This setting affects whether or not “intra-parcel” trips are allowed in destination choice models.)

**Format**: ASCII delimited, with a header record. “Real >=0” below indicates a non-negative number which does not need to be an integer value.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| parcelid | 1 - 9999999 | The parcel ID number. Values must be unique positive integers, in ascending order. (Gaps are allowed, but not efficient for memory.) |
| xcoord\_p | 1-999999999 | The x coordinate of the parcel centroid, in integer length units (typically SPF). |
| etc  etc. | etc.  etc. | etc.  etc. …. Variables containing the same data as the raw parcel file |
| pprichrp | Real >=0 | The average price of public off-street parking spaces on a parcel with per hour pricing. (In cents per hour) |
| hh\_1 | Real >=0 | The number of households residing in buffer 1. |
| stugrad\_1 | Real >=0 | The number of grade school (K-8) students enrolled in schools in buffer 1 |
| stuhgh\_1 | Real >=0 | The number of high school (9-12) students enrolled in schools in buffer 1. |
| stuuni\_1 | Real >=0 | The number of university/college students enrolled in schools in buffer 1. |
| empedu\_1 | Real >=0 | The number of educational employees working in buffer 1 |
| empfoo\_1 | Real >=0 | The number of food service employees working in buffer 1 |
| empgov\_1 | Real >=0 | The number of government employees working in buffer 1 |
| empind\_1 | Real >=0 | The number of industrial employees working in buffer 1 |
| empmed\_1 | Real >=0 | The number of medical employees working in buffer 1 |
| empofc\_1 | Real >=0 | The number of (other) office employees working in buffer 1 |
| empret\_1 | Real >=0 | The number of retail employees working in buffer 1 |
| empsvc\_1 | Real >=0 | The number of (other) service employees working in buffer 1 |
| empoth\_1 | Real >=0 | The number of other sector employees working in buffer 1. |
| emptot\_1 | Real >=0 | The total number of employees working in buffer 1. |
| parkdy\_1 | Real >=0 | The number of paid public off-street parking spaces in buffer 1 with per day pricing. |
| parkhr\_1 | Real >=0 | The number of paid public off-street parking spaces in buffer 1 with per hour pricing. |
| ppricdy1 | Real >=0 | The average price of public off-street parking spaces in buffer 1 with per day pricing. (In cents per day) |
| pprichr1 | Real >=0 | The average price of public off-street parking spaces in buffer 1 with per hour pricing. (In cents per hour) |
| nodes1\_1 | Real >=0 | The number of intersections with 1 connecting link in buffer 1 |
| nodes3\_1 | Real >=0 | The number of intersections with 3 connecting links in buffer 1 |
| nodes4\_1 | Real >=0 | The number of intersections with 4 or more connecting links in buffer 1 |
| tstops\_1 | Real >=0 | The number of transit stops in buffer 1 |
| nparks\_1 | Real >=0 | The number of public open space areas in buffer 1 |
| aparks\_1 | Real >=0 | The average size of public open space areas in buffer 1 (in square length units) |
| hh\_2 | Real >=0 | The number of households residing in buffer 2. |
| stugrad\_2 | Real >=0 | The number of grade school (K-8) students enrolled in schools in buffer 2 |
| stuhgh\_2 | Real >=0 | The number of high school (9-12) students enrolled in schools in buffer 2. |
| stuuni\_2 | Real >=0 | The number of university/college students enrolled in schools in buffer 2. |
| empedu\_2 | Real >=0 | The number of educational employees working in buffer 2 |
| empfoo\_2 | Real >=0 | The number of food service employees working in buffer 2 |
| empgov\_2 | Real >=0 | The number of government employees working in buffer 2 |
| empind\_2 | Real >=0 | The number of industrial employees working in buffer 2 |
| empmed\_2 | Real >=0 | The number of medical employees working in buffer 2 |
| empofc\_2 | Real >=0 | The number of (other) office employees working in buffer 2 |
| empret\_2 | Real >=0 | The number of retail employees working in buffer 2 |
| empsvc\_2 | Real >=0 | The number of (other) service employees working in buffer 2 |
| empoth\_2 | Real >=0 | The number of other sector employees working in buffer 2. |
| emptot\_2 | Real >=0 | The total number of employees working in buffer 2. |
| parkdy\_2 | Real >=0 | The number of paid public off-street parking spaces in buffer 2 with per day pricing. |
| parkhr\_2 | Real >=0 | The number of paid public off-street parking spaces in buffer 2 with per hour pricing. |
| ppricdy2 | Real >=0 | The average price of public off-street parking spaces in buffer 2 with per day pricing. (In cents per day) |
| pprichr2 | Real >=0 | The average price of public off-street parking spaces in buffer 2 with per hour pricing. (In cents per hour) |
| nodes1\_2 | Real >=0 | The number of intersections with 1 connecting link in buffer 2 |
| nodes3\_2 | Real >=0 | The number of intersections with 3 connecting links in buffer 2 |
| nodes4\_2 | Real >=0 | The number of intersections with 4 or more connecting links in buffer 2 |
| tstops\_2 | Real >=0 | The number of transit stops in buffer 2 |
| nparks\_2 | Real >=0 | The number of public open space areas in buffer 2 |
| aparks\_2 | Real >=0 | The average size of public open space areas in buffer 2 (in square length units) |
| dist\_lbus | Real >=0 | The distance to the nearest local bus stop in miles (999 If no stops within 3 miles) |
| dist\_ebus | Real >=0 | The distance to the nearest premium bus stop in miles (999 If no stops within 3 miles) |
| dist\_crt | Real >=0 | The distance to the nearest commuter rail station in miles (999 If no stops within 3 miles) |
| dist\_fry | Real >=0 | The distance to the nearest ferry terminal in miles (999 If no stops within 3 miles) |
| dist\_lrt | Real >=0 | The distance to the nearest light rail stop in miles (999 If no stops within 3 miles) |
| dist\_park | Real >=0 | The distance to the edge of the nearest public open space area (999 If no areas within 3 miles, or if no open space file supplied) |

If the Circuity short distance method (c) is being used, the buffering program and DaySim will also expect an extra 24 fields to be present the end of the parcel file records, with the circuity ratio (shortest path distance/straight line distance) for 3 distances in each of 8 radial directions. As mentioned above, we do not recommend using the Circuity method unless the user is already familiar with it and using it.

1. **The raw “TransitStopArea” file**

This file is **optiona**l, and is only used if the transit skims are indicated in the Roster file to be stop are-to-stop are rather than zone-to-zone.

**Relevant DaySim configuration parameters:**

* RawTransitStopAreaPath (string file name and path)
* RawTransitStopAreaDelimiter (ASCII code, 9=tab, 32=space, 44=comma)

**Format**: ASCII delimited, with a header record.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| Zone\_ID | 1 - 9999999 | The stop area ID number used in the network software that produces skims. Values must be unique positive integers, in ascending order. |
| Zone\_ordinal | Integer | Not currently used by DaySim – any integer value is ok. |
| Dest\_eligible | Integer | Not currently used by DaySim – any integer value is ok. |
| External | Integer | Not currently used by DaySim – any integer value is ok. |
| Xcoord | 1 – 9999999 | The x coordinate of the transit stop area, in integer length units (typically SPF). |
| Ycoord | 1 – 9999999 | The y coordinate of the zone centroid, in integer length units (typically SPF). |

1. **The “NodeStopAreaIndex” file**

This file is **optiona**l, and is only used if the transit skims are indicated in the Roster file to be stop are-to-stop are rather than zone-to-zone. It is very similar in structure to the “NodeDistances” file, in that it contains 2 IDs and a distance, in feet.

**Relevant DaySim configuration parameters:**

* NodeStopAreaIndexPath (string file name and path)
* NodeStopAreaIndexDelimiter (ASCII code, 9=tab, 32=space, 44=comma)
* MaximumParcelToStopAreaDistance (to limit run time, the maximum distance in feet that the DaySim parcel-to-parcel walk-to-transit path choice code will allow from node to stop area on either end)
* MaximumStopAreasToSearch (to limit run time, the maximum number of stop areas to include in search at either end for the DaySim parcel-to-parcel walk-to-transit path choice)
* MaximumParcelToStopAreaDistanceParkAndRide (to limit run time, the maximum distance in feet that the DaySim parcel-to-parcel park and ride transit path choice code will allow from node to stop area on the non-home end)
* MaximumStopAreasToSearchParkAndRide (to limit run time, the maximum number of stop areas to include in search for the DaySim parcel-to-parcel park and ride transit path choice at the non-home end)

**Format**: ASCII delimited, with a header record.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| parcelid | 1 - 9999999 | The parcel/microzone ID, which must be a valid id in the RawParcel file. The records must be sorted by this field, as DaySim indexes the file as it is read in. |
| stopareaid | 1 - 9999999 | The transit stop area id, which must be a valid id in the Raw TransitStopArea file. The file does not need to be sorted in any way by this id. |
| distance | 1-999999999 | The parcel/microzone to stop area all-streets shortest path distance, in feet. The file does not need to sorted by this field, but if one is using the “MaximumStopAreasToSearch” setting, then it would be best to sort by increasing distance wthin parceled, so that DaySim uses the closest stop areas first in the search. |

1. **The raw “ParkAndRideNode” file**

This file is for park and ride lot/path choice in DaySim. It is **optional**, and not needed if either (a) the park and ride mode is not included in the model for the region, or (b) the park and ride skim matrices are prepared in the network software rather than using the path choice in DaySim.

**Relevant DaySim configuration parameters:**

* RawParkAndRideNodePath (string file name and path)
* RawParkAndRidePathDelimiter (ASCII code, 9=tab, 32=space, 44=comma)
* ShouldReadParkAndRideNodeSkim (if true, DaySim will get park and ride mode LOS from zone-to-zone skims listed in the Roster, rather than using lot/path choice)
* MaximumMilesToDriveToParkAndRide (to save run time, a limit on the drive search radius for partk and ride lot/path choice)
* MaximumRatioDriveToParkAndRdieVersusDriveToDestination (to save run time and avoid illogical lot choices, a limit on the ratio of the drive access distance to park and ride and the drive access distance to the final destination)

**Format**: ASCII delimited, with a header record.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| Node\_ID | 1 - 9999999 | The park and ride node ID number. Values must be unique positive integers, in ascending order. |
| Zone\_ID | 1 - 9999999 | The zone that the lot is associated in. Must be a zone ID present in the raw Zone file. May be either an internal (destination-eligible zone) or a special park and ride zone, which allows more accurate zone-to-zone skims for park and ride |
| xcoord | 1 – 9999999 | The x coordinate of the lot, in integer length units (typically SPF). |
| ycoord | 1 – 9999999 | The y coordinate of the lot, in integer length units (typically SPF). |
| capacity | 0 – 9999999 | The number of parking spaces in the lot. A value of 0 makes the lot unavailable as a choice option, but can be useful for including a lot as a placeholder for future/alternative scenarios |
| cost | 0 – 9999999 | The daily parking cost for the lot, in hundredths of monetary units (typically this is cents) |

1. **The “IXXI” zonal file**

This is a file that DaySim uses for the work location model, to set the percent of workers living in each zone that work outside the region, and the percent of jobs in each zone that are filled by workers living outside the region. DaySim does not select a usual workplace or simulate internal home-work tours for the I-X fraction of workers, and makes the X-I fraction of jobs unavailable as usual work locations for region residents. Both fractions tend to be larger towards the edges of modeled regions.

**Relevant DaySim configuration parameters:**

* IxxiPath (string file name and path)
* IxxiDelimiter (ASCII code, 9=tab, 32=space, 44=comma)
* IxxiFirstLineIsHeader (if True, DaySim expects a header record)

**Format**: ASCII delimited, with optional header.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| Zone\_ID | 1 - 9999999 | The zone ID number. Values must be unique positive integers, in ascending order. There must be the same number of records in the same order as in the raw Zone file. |
| ix\_frac | 0.00 – 1.00 | The fraction of workers living in the zone that have a usual work location outside the modeled region (not in a destination-eligible zone) |
| xi\_frac | 0.00 – 1.00 | The fraction of jobs in the zone that are filled by workers living outside of the region (not in a destination-eligible zone) |

1. **The “Roster” and “RosterCombinations” files**

The Roster file is a very convenient and flexible way to define which skim matrices are used for all modes/path types and level-of-service (LOS) variables, for all times of day. The RosterCombinations file is used together with the Roster file, and tells which mode-path type combinations are valid in the Roster file. A “path type” is essentially a “submode” in DaySim, such as tolled versus non-tolled networks for auto modes or local bus versus light rail (plus optional bus) networks for transit.

**Relevant DaySim configuration parameters:**

* RosterCombinationsPath (string file name and path)
* RosterPath (string file name and path)
* SkimDelimiter (ASCII code, 9=tab, 32=space, 44=comma; only relevant for “text-ij” skim format)
* VotVeryLowLow (the VOT, in dollars per hour, that is the boundary between the “VeryLow” and “Low” VotGroups in the Roster
* VotLowMedium (the VOT, in dollars per hour, that is the boundary between the “Low” and “Medium” VotGroups in the Roster
* VotMediumHigh (the VOT, in dollars per hour, that is the boundary between the “Medium” and “High” VotGroups in the Roster
* VotHighVeryHigh (the VOT, in dollars per hour, that is the boundary between the “High” and “VeryHigh” VotGroups in the Roster

The RosterCombinations file must be in .CSV format, and is a matrix where the columns give the vaild mode labels for the Roster entries, and the rows give the valid “path type” labels for the Roster entries. An example is shown below. A TRUE entry means that the mode/path type combination is valid for the matrix entries in the Roster, and a FALSE entry means that it is not valid. A TRUE entry does not mean that the path type is required, however, so using the file below would not require separate “no-tolls” matrices for the auto modes, or separate “ferry” matrices for transit – but it would allow them.

A few other things to note in this example…

There is a “park-and-ride” mode listed, but that does not require that the user provide skims for park and ride. (That is an option, but DaySim can also use its internal lot/path choice to create park and ride paths from the “sov” and “transit” skims.)

All of the path types are FALSE for the “school-bus” mode. Typically, the school bus mode uses the “hov3” skims in the models, so no separate “school-bus” skims are listed in the Roster.

The “other” mode is basically a placeholder, not currently used in the models. For BKR, we may add a “taxi-uber” mode, but that will use “hov2” skims, so all of the path types could remain FALSE.

Some DaySim users let the network software choose the best transit path, so only use a single path type. In that case, they often list the path type for all of the “transit” skims as “local-bus”, but it would also be possible to list them all as “full-network”, and change the RosterCombinations file so that only the “full-network” row is TRUE under transit, and all of the other rows are FALSE.

A user can add or substituted different path type names, and even different mode names, but that could require substantial corresponding changes to the DaySim code—essentially every call to PathTypeModel includes a mode label and path type label, and that combination must be TRUE in the RosterCombinations file.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| # | Walk | bike | Sov | hov2 | hov3 | transit | park-and-ride | school-bus | other |
| full-network | TRUE | TRUE | TRUE | TRUE | TRUE | FALSE | FALSE | FALSE | FALSE |
| no-tolls | TRUE | TRUE | TRUE | TRUE | TRUE | FALSE | FALSE | FALSE | FALSE |
| local-bus | FALSE | FALSE | FALSE | FALSE | FALSE | TRUE | TRUE | FALSE | FALSE |
| light-rail | FALSE | FALSE | FALSE | FALSE | FALSE | TRUE | TRUE | FALSE | FALSE |
| premium-bus | FALSE | FALSE | FALSE | FALSE | FALSE | TRUE | TRUE | FALSE | FALSE |
| commuter-rail | FALSE | FALSE | FALSE | FALSE | FALSE | TRUE | TRUE | FALSE | FALSE |
| ferry | FALSE | FALSE | FALSE | FALSE | FALSE | TRUE | TRUE | FALSE | FALSE |

The Roster file must also be in CSV format, and must include the columns listed below and shown in the example below.

* **#variable**: A variable label, as referred to in DaySim code
* **mode**: A mode label, as referred to in the DaySim code and present in the RosterCombinations file
* **path-type**: A path type label, as referred to in the DaySim code, and the mode/path type combination must be TRUE in the RosterCombinations file In the example, the “walk”, “bike”, “sov”, “hov2” and “hov3” are input only for the “full-network” path type, while separate “transit” skims are input for five different path types.
* **vot-group**: A value of time class, with boundaries set in the configuration file. Valid names are very-low, low, medium, high, very-high, or all.
* **start-minute** and **end-minute:** The time period for which the skim matrix applies, in minutes past midnight. For example, 0-1439 is the entire day, and 360-539 is 6:00 am to 8:59 am. For each variable/mode/path-type/vot-group combination, the skims should cover all minutes from 0 to 1439. An example is the last 5 lines above for toll, for 5 time periods. The last period 1110-179 spans midnight, and is 6:30 pm to 2:59 am.
* **length:**  The “zone” system used for the matrix. “maxzone” uses Zone, “transitstop” uses TransitStopArea, “null” just returns a value of 0 instead of reading in a matrix (in which case the “file-type” and “name” columns should also be “null”).
* **file-type**: The format of the file. Valid types are:
  + **text-ij :** Text files with a record for each I-J zone pair that can contain more than 1 skim variable. Column 1 of the text file is always the I zone and column 2 the J zone.
  + **transcad**: Native binary format written by TransCAD-requires a valid TransCAD license to be installed when running.
  + **cube**: Native binary format written by Cube. Requires a valid Cube license to be installed when running.
  + **emme**: Native binary format written to EMME databanks.
  + **visum-binary** : Native binary format written by Visum.
  + **hdf5** : HDF5 format, which can be written by various network packages or converters.
  + **omx** : Open Matrix format (modified HDF5), which can be written by various network packages or converters.
  + **bin :** A custom DaySim binary format, which is fastest to load, but requires pre-processing of the matrices.
  + **null :** Returns a value of 0. (“length” and “name” should also be set to “null”)
* **name**: The matrix file name (plus the table name for HDF5 or OMX files). The directory path is assumed to be the same as for the Roster file. Note that different roster rows may refer to the same file—such as the first four rows in the example using “walkSkim.h5/1”. In that case, the same matrix is used for multiple variables, but only read and stored in memory once. In the example, the toll matrices use the same table number, but different file names for the five time periods.
* **field**: The matrix number on the file for the particular variable. For “text-ij”, it is the column number. For “cube” it is the Cube matrix number.
* **transpose**: This indicates that the origin and destination zones for the matrix should be “switched” and the transpose used. For example, the transpose of the AM peak transit matrices can be used to represent the PM peak period.
* **blend-variable**: This is the variable to use for “short distance blending”. It is only relevant for the walk, bike and auto modes, but not for transit. In practice, this should always be set as “distance”, which is what the DaySim code assumes.
* **blend-path-type**: This is the path type to use for “short distance blending”. If it is set to “null”, then DaySim assumes that the “blend-path-type” is the same as the “path-type” entry on the same row. It would be possible to use a different path type for the same mode if no “distance” matrix was available for “path type”, but in almost all cases, the user should leave this as “null”
* **factor**: This allows one variable to be set as a factored version of another variable. In the example below, “walk” “time” is set to use the same matrix as “walk” “distance”, but with a factor of 20, which assumes a walk speed of 20 minutes per mile. Similarly, “bike” “time” is set to use the same matrix as “bike” “distance”, but with a factor of 6, which assumes a bike speed of 6 minutes per mile. If the entry is “null”, the default factor is 1.0.

**Roster file example: Part 1 (not the complete roster – most auto variables shown only for the AM peak period)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| #variable | mode | path-type | vot-group | start-minute | end-minute | length | file-type | name | field | transpose | blend-variable | blend-path-type | factor | scaling |
| distance | walk | full-network | all | 0 | 1439 | maxzone | hdf5 | walkSkim.h5/1 | 1 | FALSE | distance | null | null | TRUE |
| time | walk | full-network | all | 0 | 1439 | maxzone | hdf5 | walkSkim.h5/1 | 1 | FALSE | distance | null | 20 | TRUE |
| distance | bike | full-network | all | 0 | 1439 | maxzone | hdf5 | walkSkim.h5/1 | 1 | FALSE | distance | null | null | TRUE |
| time | bike | full-network | all | 0 | 1439 | maxzone | hdf5 | walkSkim.h5/1 | 1 | FALSE | distance | null | 6 | TRUE |
| distance | sov | full-network | all | 360 | 539 | maxzone | hdf5 | HWYALLAM.h5/20 | 1 | FALSE | distance | null | null | TRUE |
| ivtime | sov | full-network | all | 360 | 539 | maxzone | hdf5 | HWYALLAM.h5/19 | 1 | FALSE | distance | null | null | TRUE |
| toll | sov | full-network | all | 360 | 539 | maxzone | hdf5 | HWYALLAM.h5/43 | 1 | FALSE | distance | null | 0.01 | FALSE |
| distance | hov2 | full-network | all | 360 | 539 | maxzone | hdf5 | HWYALLAM.h5/27 | 1 | FALSE | distance | null | null | TRUE |
| ivtime | hov2 | full-network | all | 360 | 539 | maxzone | hdf5 | HWYALLAM.h5/26 | 1 | FALSE | distance | null | null | TRUE |
| toll | hov2 | full-network | all | 360 | 539 | maxzone | hdf5 | HWYALLAM.h5/44 | 1 | FALSE | distance | null | 0.01 | FALSE |
| distance | hov3 | full-network | all | 360 | 539 | maxzone | hdf5 | HWYALLAM.h5/34 | 1 | FALSE | distance | null | null | TRUE |
| ivtime | hov3 | full-network | all | 360 | 539 | maxzone | hdf5 | HWYALLAM.h5/33 | 1 | FALSE | distance | null | null | TRUE |
| toll | hov3 | full-network | all | 180 | 359 | maxzone | hdf5 | HWYALLEA.h5/45 | 1 | FALSE | distance | null | 0.01 | FALSE |
| toll | hov3 | full-network | all | 360 | 539 | maxzone | hdf5 | HWYALLAM.h5/45 | 1 | FALSE | distance | null | 0.01 | FALSE |
| toll | hov3 | full-network | all | 540 | 929 | maxzone | hdf5 | HWYALLMD.h5/45 | 1 | FALSE | distance | null | 0.01 | FALSE |
| toll | hov3 | full-network | all | 930 | 1109 | maxzone | hdf5 | HWYALLPM.h5/45 | 1 | FALSE | distance | null | 0.01 | FALSE |
| toll | hov3 | full-network | all | 1110 | 179 | maxzone | hdf5 | HWYALLEV.h5/45 | 1 | FALSE | distance | null | 0.01 | FALSE |

**Roster file example: Part 2 (not the complete roster – all transit variables shown only for the AM peak period)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| #variable | mode | path-type | vot-group | start-minute | end-minute | length | file-type | name | field | transpose | blend-variable | blend-path-type | factor | scaling |
| ivtime | transit | local-bus | all | 360 | 539 | maxzone | hdf5 | TRNWLWAM.h5/22 | 1 | FALSE | null | null | null | TRUE |
| iwaittime | transit | local-bus | all | 360 | 539 | maxzone | hdf5 | TRNWLWAM.h5/11 | 1 | FALSE | null | null | null | TRUE |
| xwaittime | transit | local-bus | all | 360 | 539 | maxzone | hdf5 | TRNWLWAM.h5/12 | 1 | FALSE | null | null | null | TRUE |
| nboard | transit | local-bus | all | 360 | 539 | maxzone | hdf5 | TRNWLWAM.h5/17 | 1 | FALSE | null | null | null | TRUE |
| fare | transit | local-bus | all | 360 | 539 | maxzone | hdf5 | TRNWLWAM.h5/18 | 1 | FALSE | null | null | 0.01 | FALSE |
| ivtime | transit | light-rail | all | 360 | 539 | maxzone | hdf5 | TRNWMWAM.h5/22 | 1 | FALSE | null | null | null | TRUE |
| iwaittime | transit | light-rail | all | 360 | 539 | maxzone | hdf5 | TRNWMWAM.h5/11 | 1 | FALSE | null | null | null | TRUE |
| xwaittime | transit | light-rail | all | 360 | 539 | maxzone | hdf5 | TRNWMWAM.h5/12 | 1 | FALSE | null | null | null | TRUE |
| nboard | transit | light-rail | all | 360 | 539 | maxzone | hdf5 | TRNWMWAM.h5/17 | 1 | FALSE | null | null | null | TRUE |
| fare | transit | light-rail | all | 360 | 539 | maxzone | hdf5 | TRNWMWAM.h5/18 | 1 | FALSE | null | null | 0.01 | FALSE |
| ivtime | transit | premium-bus | all | 360 | 539 | maxzone | hdf5 | TRNWPWAM.h5/24 | 1 | FALSE | null | null | null | TRUE |
| iwaittime | transit | premium-bus | all | 360 | 539 | maxzone | hdf5 | TRNWPWAM.h5/11 | 1 | FALSE | null | null | null | TRUE |
| xwaittime | transit | premium-bus | all | 360 | 539 | maxzone | hdf5 | TRNWPWAM.h5/12 | 1 | FALSE | null | null | null | TRUE |
| nboard | transit | premium-bus | all | 360 | 539 | maxzone | hdf5 | TRNWPWAM.h5/17 | 1 | FALSE | null | null | null | TRUE |
| fare | transit | premium-bus | all | 360 | 539 | maxzone | hdf5 | TRNWPWAM.h5/18 | 1 | FALSE | null | null | 0.01 | FALSE |
| ivtime | transit | commuter-rail | all | 360 | 539 | maxzone | hdf5 | TRNWBWAM.h5/24 | 1 | FALSE | null | null | null | TRUE |
| iwaittime | transit | commuter-rail | all | 360 | 539 | maxzone | hdf5 | TRNWBWAM.h5/11 | 1 | FALSE | null | null | null | TRUE |
| xwaittime | transit | commuter-rail | all | 360 | 539 | maxzone | hdf5 | TRNWBWAM.h5/12 | 1 | FALSE | null | null | null | TRUE |
| nboard | transit | commuter-rail | all | 360 | 539 | maxzone | hdf5 | TRNWBWAM.h5/17 | 1 | FALSE | null | null | null | TRUE |
| fare | transit | commuter-rail | all | 360 | 539 | maxzone | hdf5 | TRNWBWAM.h5/18 | 1 | FALSE | null | null | 0.01 | FALSE |
| ivtime | transit | ferry | all | 360 | 539 | maxzone | hdf5 | TRNWFWAM.h5/24 | 1 | FALSE | null | null | null | TRUE |
| iwaittime | transit | ferry | all | 360 | 539 | maxzone | hdf5 | TRNWFWAM.h5/11 | 1 | FALSE | null | null | null | TRUE |
| xwaittime | transit | ferry | all | 360 | 539 | maxzone | hdf5 | TRNWFWAM.h5/12 | 1 | FALSE | null | null | null | TRUE |
| nboard | transit | ferry | all | 360 | 539 | maxzone | hdf5 | TRNWFWAM.h5/17 | 1 | FALSE | null | null | null | TRUE |
| fare | transit | ferry | all | 360 | 539 | maxzone | hdf5 | TRNWFWAM.h5/18 | 1 | FALSE | null | null | 0.01 | FALSE |

* **scaling**: This last setting causes some confusion, so deserves careful explanation. DaySim stores all matrix values in memory as 2-byte unsigned integers, which can take values between 0 and roughly 65,000. Those values are assumed to be hundredths of miles for distance, hundredths of minutes for times, hundredths of dollars (cents) for costs, and hundredths of boardings for transit boardings or transfers. So, the maximum skim values that can be stored in memory are roughly 650 miles, minutes, dollars, or boardings, which is sufficient for regional models. (Any values larger than this are capped at the maximum value.)

In most cases, the input matrices are in units of miles, minutes, or dollars. In that case, “scaling” should be set to TRUE, and DaySim will scale the values by 100 when storing them in memory, and then “unscale” them back to the original units when accessing them from memory. That is done for most of the walk, bike, and auto variables in the example, as well as transit times and boardings (“nboard”).

In the example, the toll and fare matrices were already in units of cents rather than dollars, so no scaling is necessary and “scaling” is FALSE for those variables. However, the DaySim code assumes that these costs are in dollars, so a factor of 0.1 is necessary to convert the cost in cents to a cost in dollars. In general, the rules are:

***In summary, if the matrices are in units of miles, minutes or dollars, the proper setting of “scaling” is “TRUE” and “factor” is “null” (unless converting walk distance to walk time, or a similar conversion).***

***If the matrices are in cents, or they are already pre-scaled to hundredths of miles or hundredths of minutes, the proper setting of “scaling” is “FALSE” and “factor” is “0.01”.***

1. **The raw “Household” file**

A raw household file is required input for DaySim. It can be a synthetic population for application mode, or from a travel survey for estimation mode.

Of the variables listed below, only five (hhno, hhsize, hhincome, hhparcel, and hhexpfac) are strictly needed by DaySim as inputs on the raw data file. One (hhvehs) is predicted by DaySim. The rest can be computed based on other data files or aren’t currently used in model application. DaySim also adds two variables to the working and output household file (“fraction\_with\_jobs\_outside”, which is a property of the residence zone, and “zone\_id” which is DaySim’s internal zone ID corresponding to “hhtaz”).

**Relevant DaySim configuration parameters:**

* RawHouseholdPath (string file name and path)
* RawHouseholdDelimiter (ASCII code, 9=tab, 32=space, 44=comma)

**Format**: ASCII delimited, with a header record.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| Hhno | 1 - 9999999 | The household ID number. Values must be unique positive integers, and should be in ascending order. |
| Hhsize | 1 – 99 | The number of persons in the household. Must equal the number of person records for the household in the raw person file |
| Hhvehs | 0 - 99 | The number of autos in the household. (This could be made optional as input, as it is predicted by DaySim.) |
| Hhwkrs | 0 – 99 | The number of workers in the household. (This could be made optional as input, as it is computed by DaySim from the person records.) |
| Hhftw | 0 – 99 | The number of HH members with person type=full-time worker. (This could be optional as input, as it is computed by DaySim.) |
| Hhptw | 0 – 99 | The number of HH members with person type=part-time worker. (This could be optional as input, as it is computed by DaySim.) |
| Hhret | 0 – 99 | The number of HH members with person type=retired adult. (This could be optional as input, as it is computed by DaySim.) |
| hhoad | 0 – 99 | The number of HH members with person type=other non-working adult. (This could be optional as input, as it is computed by DaySim.) |
| hhuni | 0 – 99 | The number of HH members with person type=university student. (This could be optional as input, as it is computed by DaySim.) |
| hhhsc | 0 – 99 | The number of HH members with person type=grade school student age 16+. (This could be optional as input, as it is computed by DaySim.) |
| hh515 | 0 – 99 | The number of HH members with person type=child age 5-15. (This could be optional as input, as it is computed by DaySim.) |
| hhcu5 | 0 – 99 | The number of HH members with person type=child age 0-4. (This could be optional as input, as it is computed by DaySim.) |
| hhincome | -1 – 9999999 | The household annual income, in integer dollars. (A negative value is interpreted as missing data in DaySim estimation mode.) |
| hhownrent | 1 – 9 | Household own versus rent status. (This could be optional as input, as it is not currently used in the DaySim model code.) |
| hhrestype | 1 – 9 | Household residence building type. (This could be optional as input, as it is not currently used in the DaySim model code.) |
| hhparcel | 1 – 9999999 | The ID of the parcel on which the household lives. Must be a parcel ID found in the raw parcel file. |
| hhhtaz | 1 – 9999999 | The ID of the zone in which the household lives. (This could be optional as input, as the Parcel file has the parcel-zone correspondence.) |
| hhexpfac | Real >= 0 | The expansion factor for the household – a non-negative real number. (Is typically 1.0 in a synthetic population.) |
| samptype | 0 - 99 | The type of sample used. (This could be optional as input, as it is not used in the DaySim model code, but can be useful with survey data in model estimation to identify different sample types.) |

1. **The raw “Person” file**

A raw person file is required input for DaySim. It is typically for a synthetic population in application mode, or from a travel survey in estimation mode.

Of the variables listed below, only seven (hhno, pno, pptyp, pagey, pgend, pwtyp, pstyp) are needed by DaySim as inputs in the raw data file. Four (pwpcl, pspcl, ptpass and ppaidprk) are predicted by DaySim. The rest are computed based on other data files or aren’t currently used in model application, and can be coded as -1 by the user. DaySim also adds one variable to the beginning of each record in the output person file (“ID” which is a sequential, unique person ID created by DaySim)

**Relevant DaySim configuration parameters:**

* RawPersonPath (string file name and path)
* RawPersonDelimiter (ASCII code, 9=tab, 32=space, 44=comma)

**Format**: ASCII delimited, with a header record.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| hhno | 1 - 9999999 | The household ID number. Values must be unique positive integers, and should be in ascending order. Must be present in the Household file. |
| pno | 1 – 99 | The person sequence number within the household. Values must be unique positive integers within a household, and in ascending order from 1 up to “hhsize” on the Household file. |
| pptyp | 1 - 8 | Person type (1= full time worker, 2 =part time worker, 3=non-worker age 65+, 4 = other non-working adult, 5 = university student, 6 = grade school student/child age 16+, 7 = child age 5-15, 8 = child age 0-4. (There could be a switch to make this optional and compute it within DaySim for synthetic populations based on ACS PUMS. For other survey data, the coding and rules may be more variable and better done outside DaySim.) |
| pagey | 0 – 99 | Age in years (integer) |
| pgend | 1 – 9 | Gender (1=male, 2=female, 9=missing data for estimation) |
| pwtyp | 0 - 2 | Worker type (0=non-worker, 1=full time worker, 2=part time worker) |
| pwpcl | -1 - 9999999 | Usual work location parcel ID. -1 for none/missing, otherwise must be a valid parcel ID present in the Parcel file. |
| pwtaz | -1 - 9999999 | Usual work location zone ID. (This could be optional as input, as the Parcel file has the parcel-zone correspondence.) |
| pwautime | -1 - 9999999 | The 1-way peak auto travel time between the residence and usual work parcels (a real number of minutes, -1 if no usual work location. Could be optional as input, used as output for calibration.) |
| pwaudist | -1 - 9999999 | The 1-way peak auto travel distance between the residence and usual work parcels (a real number of miles, -1 if no usual work location. Could be optional as input, used as output for calibration.) |
| pstyp | 0 - 2 | Worker type (0=non-student, 1=full time student, 2=part time student if known – part-time distinction not used in DaySim code) |
| pspcl | -1 - 9999999 | Usual school location parcel ID. -1 for none/missing, otherwise must be a valid parcel ID present in the Parcel file. |
| pstaz | -1 - 9999999 | Usual school location zone ID. (This could be optional as input, as the Parcel file has the parcel-zone correspondence.) |
| psautime | -1 - 9999999 | The 1-way peak auto travel time between the residence and usual school parcels (a real number of minutes, -1 if no usual school location. Could be optional as input, used as output for calibration.) |
| psaudist | -1 - 9999999 | The 1-way peak auto travel distance between the residence and usual school parcels (a real number of miles, -1 if no usual school location. Could be optional as input, used as output for calibration.) |
| puwmode | -1 – 9 | The usual mode used to work. (This is optional, as it is a placeholder for possible models that may be added to DaySim in the future.) |
| puwarrp | -1 – 9 | The usual arrival period at work. (This is optional, as it is a placeholder for possible models that may be added to DaySim in the future.) |
| puwdepp | -1 – 9 | The usual departure period from work. (This is optional, as a placeholder for possible models that may be added to DaySim in the future.) |
| ptpass | 0 – 1 | Transit pass ownership (0=no, 1=yes. This is predicted by DaySim, so could be an optional input in application mode.) |
| ppaidprk | 0 – 1 | Worker has to pay to park at work (0=no, 1=yes. This is predicted by DaySim, so could be an optional input in application mode.) |
| pdiary | 0 – 1 | Survey respondent used their diary? (0=no, 1=yes. This is only relevant for survey data in estimation, so could be optional in application mode.) |
| pproxy | 0 – 1 | Survey responses by proxy? (1=no, 2=yes, 3=by mail, 9=missing. This is only relevant for survey data in estimation, so could be optional in application mode.) |
| psexpfac | Real >= 0 | The expansion factor for the person – a non-negative real number. (In application mode, this could be optional, since it is set equal to hhexpfac) |

1. **The “HouseholdDay” file**

A raw HouseholdDay file is not used as input for DaySim in application mode, but it is required in estimation mode. An output HouseholdDay file is produced in application mode. This file is mainly relevant for the “H” version of the DaySim models which has models of joint tour and half tour generation that do not belong to single persons in the household. This file is still produced in the Default version of the models, but contains no useful information.

DaySim adds one variable to the beginning of each record in the output household-day file (“ID” which is a sequential, unique ID created by DaySim)

**Relevant DaySim configuration parameters:**

* RawHouseholdDayPath (string file name and path)
* RawHouseholdDayDelimiter (ASCII code, 9=tab, 32=space, 44=comma)

**Format**: ASCII delimited, with a header record.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| hhno | 1 - 9999999 | The household ID number. Values must be unique positive integers, and should be in ascending order. Must be present in the Household file. |
| day | 1 – 99 | The survey day sequence, if using multi-day survey data, or if DaySim were programmed to simulate multiple days per household (which is not a current feature). |
| dow | 1 - 7 | The day of the week, which is relevant for survey data, but is not currently used in the DaySim models. |
| jttours | 0 - 99 | The number of fully joint tour records output for the household. (Is set only in the H version of the DaySim models.) |
| phtours | 0 - 99 | The number of partially joint half tour records output for the household. (Is set only in the H version of the DaySim models.) |
| fhtours | 0 - 99 | The number of fully joint half tour records output for the household. (Is set only in the H version of the DaySim models.) |
| hdexpfac | Real >= 0 | The expansion factor for the household-day – a non-negative real number. (Is set equal to hhexpfac in application mode) |

1. **The “PersonDay” file**

A raw PersonDay file is not used as input for DaySim in application mode, but it is required in estimation mode. An output PersonDay file is created in application mode.

DaySim adds three variable to the beginning of each record in the output PersonDay file (“ID” which is a sequential, unique ID created by DaySim, “person\_ID” which is a link to the “ID” field in the output Person file, and “household\_day\_ID”, which is a link to the “ID” field in the output HouseholdDay file)

**Relevant DaySim configuration parameters:**

* RawPersonDayPath (string file name and path)
* RawPersonDayDelimiter (ASCII code, 9=tab, 32=space, 44=comma)

**Format**: ASCII delimited, with a header record.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| hhno | 1 - 9999999 | The household ID number. Values must be unique positive integers, and should be in ascending order. Must be present in the Household file. |
| pno | 1 – 99 | The person sequence number within the household. Values must be unique positive integers within a household, and match the hhno/pno combinations in the Person file |
| day | 1 – 99 | The survey day sequence, if using multi-day survey data, or if DaySim were programmed to simulate multiple days per household (which is not a current feature). |
| beghom | 0 – 1 | A flag if the survey diary day begins at home. (Not currently relevant for application mode, where all days are simulated to begin at home.) |
| endhom | 0 – 1 | A flag if the survey diary day ends at home. (Not currently relevant for application mode, where all days are simulated to end at home.) |
| hbtours | 0 – 99 | The total number of home-based tour records predicted for the person-day. |
| wbtours | 0 – 99 | The total number of work-based subtour records predicted for the person-day. |
| uwtours | 0 – 99 | The total number of home-based work tours predicted to go to the usual workplace in the person-day |
| wktours | 0 – 99 | The number of home-based work tours predicted in the person-day |
| sctours | 0 – 99 | The number of home-based school tours predicted in the person-day |
| estours | 0 – 99 | The number of home-based escort tours predicted in the person-day |
| pbtours | 0 – 99 | The number of home-based personal business tours predicted in the person-day (also includes medical tours in the Default models) |
| shtours | 0 – 99 | The number of home-based shopping tours predicted in the person-day |
| mltours | 0 – 99 | The number of home-based meal tours predicted in the person-day |
| sotours | 0 – 99 | The number of home-based social tours predicted in the person-day (also includes recreational tours in the Default models) |
| retours | 0 – 99 | The number of home-based recreation tours predicted in the person-day. (Is only predicted by the H version of the models.) |
| metours | 0 – 99 | The number of home-based medical tours predicted in the person-day. . (Is only predicted by the H version of the models.) |
| wkstops | 0 – 99 | The number of home-based work stops predicted in the person-day |
| scstops | 0 – 99 | The number of home-based school stops predicted in the person-day |
| esstops | 0 – 99 | The number of home-based escort stops predicted in the person-day |
| pbstops | 0 – 99 | The number of home-based personal business stops predicted in the person-day (also includes medical stops in the Default models) |
| shstops | 0 – 99 | The number of home-based shopping stops predicted in the person-day |
| mlstops | 0 – 99 | The number of home-based meal stops predicted in the person-day |
| sostops | 0 – 99 | The number of home-based social stops predicted in the person-day (also includes recreational stops in the Default models) |
| restops | 0 – 99 | The number of home-based recreation stops predicted in the person-day. (Is only predicted by the H version of the models.) |
| mestops | 0 – 99 | The number of home-based medical stops predicted in the person-day. (Is only predicted by the H version of the models.) |
| wkathome | 0 – 1439 | The number of minutes spent working at home during the day. (Is only predicted by the H version of the models.) |
| pdexpfac | Real >= 0 | The expansion factor for the household-day – a non-negative real number. (Is set equal to hhexpfac in application mode) |

1. **The “Tour” file**

A raw Tour file is not used as input for DaySim in application mode, but it is required in estimation mode. A DaySim format Tour file is produced as output in application mode.

DaySim adds three variable to the beginning of each record in the output Tour file (“ID” which is a sequential, unique ID created by DaySim, “person\_ID” which is a link to the “ID” field in the output Person file, and “person\_day\_ID”, which is a link to the “ID” field in the output PersonDay file)

**Relevant DaySim configuration parameters:**

* RawTourPath (string file name and path)
* RawTourDelimiter (ASCII code, 9=tab, 32=space, 44=comma)

**Format**: ASCII delimited, with a header record.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| hhno | 1 - 9999999 | The household ID number. Values must be unique positive integers, and should be in ascending order. Must be present in the Household file. |
| pno | 1 – 99 | The person sequence number within the household. Values must be unique positive integers within a household, and match the hhno/pno combinations in the Person file |
| day | 1 – 99 | The survey day sequence, if using multi-day survey data, or if DaySim were programmed to simulate multiple days per household (which is not a current feature). |
| tour | 1 – 99 | The tour sequence within the person-day. |
| jtindex | 0 – 99 | Links to the sequence number of the tour in the JointTour file records for the HouseholdDay. (Only relevant for the H version of the models.) |
| parent | 0 - 99 | If it is a work-based subtour, the “tour” sequence number of the “parent” work tour, otherwise 0. |
| subtours | 0 - 99 | For home-based work tours, the number of work-based subtours made from the work activity of that tour. |
| pdpurp | 1 - 9 | The tour primary destination purpose (1=work, 2=school, 3=escort, 4=personal business (& medical), 5=shopping, 6=meal, 7=social (& recreation), 8=recreation (H version only) 9=medical (H version only) |
| tlvorig | 0 - 1439 | The time leaving the (sub)tour origin, in minutes after midnight (or hours\*100+minute for estimation mode) |
| tardest | 0 - 1439 | The time arriving at the (sub)tour destination, in minutes after midnight (or hours\*100+minute for estimation mode) |
| tlvdest | 0 - 1439 | The time leaving the (sub)tour destination, in minutes after midnight (or hours\*100+minute for estimation mode) |
| tarorig | 0 - 1439 | The time arriving back at the (sub)tour origin, in minutes after midnight (or hours\*100+minute for estimation mode) |
| toadtyp | 1 - 5 | Tour origin address type (1=home, 2=usual work location, 3=usual school location, 4=other location in region, 5=out of region/missing (survey data only) |
| tdadtyp | 1 - 5 | Tour destination address type (1=home, 2=usual work location, 3=usual school location, 4=other location in region, 5=out of region/missing (survey data only) |
| topcl | -1 - 9999999 | Tour origin parcel ID. Must be a valid parcel ID present in the Parcel file. |
| totaz | -1 - 9999999 | Tour origin zone ID. Must be a valid zone ID present in the Zone file. |
| tdpcl | -1 - 9999999 | Tour destination parcel ID. Must be a valid parcel ID present in the Parcel file. |
| tdtaz | -1 - 9999999 | Tour destination zone ID. Must be a valid zone ID present in the Zone file. |
| tmodetp | 1 - 8 | Tour main mode type (1=walk, 2=bike, 3=sov, 4=hov 2, 5=hov 3+, 6=walk to transit, 7=park and ride, 8=school bus, 9=TNC, 10=other – survey only) |
| tpathtp | 1 - 8 | Tour main mode path type (1=full network, 2=no-toll network, 3=local bus, 4=light rail, 5=premium bus, 6=commuter rail, 7=ferry) |
| tautotime | -1 - 9999999 | The one-way auto travel time between the origin and destination (a real number of minutes) |
| tautocost | -1 - 9999999 | The one-way auto toll cost between the origin and destination (a real number of dollars) |
| tautodist | -1 - 9999999 | The one-way auto travel distance between the origin and destination (a real number of miles) |
| tripsh1 | 1 - 99 | The number of trips segments on the half tour to the destination. |
| tripsh2 | 1 - 99 | The number of trips segments on the half tour from the destination. |
| phtindx1 | 0 - 99 | Links to the sequence number of the first half tour in the PartialHalfTour file records for the HouseholdDay. (Only relevant for the H version of the models.) |
| phtindx2 | 0 - 99 | Links to the sequence number of the second half tour in the PartialHalfTour file records for the HouseholdDay. (Only relevant for the H version of the models.) |
| fhtindx1 | 0 - 99 | Links to the sequence number of the first half tour in the FullHalfTour file records for the HouseholdDay. (Only relevant for the H version of the models.) |
| fhtindx2 | 0 - 99 | Links to the sequence number of the second half tour in the FullHalfTour file records for the HouseholdDay. (Only relevant for the H version of the models.) |
| toexpfac | Real >= 0 | The expansion factor for the tour – a non-negative real number. (Is set equal to hhexpfac in application mode) |

1. **The “Trip” file**

A raw Trip file is not used as input for DaySim in application mode, but it is required in estimation mode. A DaySim format Trip file is produced as output in application mode.

DaySim adds two variables to the beginning of each record in the output Trip file (“ID” which is a sequential, unique ID created by DaySim, and “tour\_ID” which is a link to the “ID” field in the output Tour file). It also adds a “VOT” field to the end of the output Trip file, which is the value of time (dollars per hour) used in simulating the choices for the trip.

**Relevant DaySim configuration parameters:**

* RawTripPath (string file name and path)
* RawTripDelimiter (ASCII code, 9=tab, 32=space, 44=comma)

**Format**: ASCII delimited, with a header record.

|  |  |  |
| --- | --- | --- |
| **Header label** | **Valid values** | **Description, comments** |
| hhno | 1 - 9999999 | The household ID number. Values must be unique positive integers, and should be in ascending order. Must be present in the Household file. |
| pno | 1 – 99 | The person sequence number within the household. Values must be unique positive integers within a household, and match the hhno/pno combinations in the Person file |
| day | 1 – 99 | The survey day sequence, if using multi-day survey data, or if DaySim were programmed to simulate multiple days per household (which is not a current feature). |
| tour | 1 – 99 | The tour sequence within the person-day. Must match a tour present for the person-day in the Tour file. |
| half | 1 –2 | The half tour (1=to the destination, 2=from the destination) |
| tseg | 1 - 99 | The trip sequence number within the half tour. |
| tsvid | 1 - 99 | Links to a travel survey trip ID (not relevant in application mode) |
| opurp | 0 – 10 | The purpose at the trip origin (0=home, 1=work, 2=school, 3=escort, 4=personal business (& medical), 5=shopping, 6=meal, 7=social (& recreation), 8=recreation (H version only) 9=medical (H version only), 10=change mode at a park and ride lot |
| dpurp | 0 – 10 | The purpose at the trip destination (0=home, 1=work, 2=school, 3=escort, 4=personal business (& medical), 5=shopping, 6=meal, 7=social (& recreation), 8=recreation (H version only) 9=medical (H version only), 10=change mode at a park and ride lot |
| oadtyp | 1 – 6 | Trip origin address type (1=home, 2=usual work location, 3=usual school location, 4=other location in region, 5=out of region/missing (survey data only), 6=inserted change mode location for park and ride |
| dadtyp | 1 – 6 | Trip dest. address type (1=home, 2=usual work location, 3=usual school location, 4=other location in region, 5=out of region/missing (survey data only), 6=inserted change mode location for park and ride |
| opcl | -1 - 9999999 | Trip origin parcel ID. Must be a valid parcel ID present in the Parcel file. |
| otaz | -1 - 9999999 | Trip origin zone ID. Must be a valid zone ID present in the Zone file. |
| dpcl | -1 - 9999999 | Trip destination parcel ID. Must be a valid parcel ID present in the Parcel file. |
| dtaz | -1 - 9999999 | Trip destination zone ID. Must be a valid zone ID present in the Zone file. |
| mode | 1 - 8 | Trip mode (1=walk, 2=bike, 3=sov, 4=hov 2, 5=hov 3+, 6=walk to transit, 7=park and ride, 8=school bus, 9=TNC, 10=other – survey only) |
| pathtype | 1 - 8 | Trip path type (1=full network, 2=no-toll network, 3=local bus, 4=light rail, 5=premium bus, 6=commuter rail, 7=ferry) |
| dorp | 0 - 999 | For auto trips, 1=driver, 2=passenger; for transit trips, is set to the total walk access+egress time, in integer minutes |
| deptm | 0 – 1439 | The trip departure time, in minutes after midnight (or hours\*100+minute for estimation mode) |
| arrtm | 0 – 1439 | The trip arrival time, in minutes after midnight (or hours\*100+minute for estimation mode) |
| endacttm | 0 – 1439 | The end time of the destination activity, in minutes after midnight (or hours\*100+minute for estimation mode) |
| travtime | -1 - 9999999 | The travel time by the trip mode and path type (a real number of minutes) |
| travcost | -1 - 9999999 | The travel cost by the trip mode and path type (a real number of dollars) |
| travdist | -1 - 9999999 | The network distance between the trip origin and destination (a real number of miles, SOV distance used for transit trips) |
| trexpfac | Real >= 0 | The expansion factor for the trip – a non-negative real number. (Is set equal to hhexpfac in application mode) |

1. **The “JointTour”, “PartialHalfTour” and “FullHalfTour” files**

Only relevant for the H version of DaySim – to be completed.

1. **The “TDMTripList” file**

The TDMTripList file is an extra trip-level output file that can be more easily customized by the user, compared to the main Trip output file. It can be made to include information that is not on the main trip file, such as separate fields for different types of travel times and costs. There is no “raw” TDMTripList file for input.

**Relevant DaySim configuration parameters:**

* ShouldOutputTDMTripsList (set to True to output a file)
* OutputTDMTripListPath (string file name and path)
* TDMTripListDelimiter (ASCII code, 9=tab, 32=space, 44=comma)
* UseTransimsTDMTripListFormat (set to True to use a specific format)

**Format**: ASCII delimited, with a header record.

The variables and labels on the file are set by the user in the “TDMTripListExporter.cs” class, which requires knowing how to edit the C# code.