General Transit Feed Specification Reference

Revised Dec 8, 2022. See Revision History for more details.

This document defines the format and structure of the files that comprise a GTFS dataset.

Table of Contents

- trips.txt
- stop_times.txt
- calendar.txt
- calendar_dates.txt
- fare_attributes.txt
- fare_rules.txt
- timeframes.txt
- fare_media.txt
- fare_products.txt
- fare_leg_rules.txt
- fare_transfer_rules.txt
- areas.txt
- stop_areas.txt
- shapes.txt
- frequencies.txt
- transfers.txt
- pathways.txt
- levels.txt
- translations.txt
- feed_info.txt
- attributions.txt

Document Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", “SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

Term Definitions

This section defines terms that are used throughout this document.

- **Dataset** - A complete set of files defined by this specification reference. Altering the dataset creates a new version of the dataset. Datasets should be published at a public, permanent URL, including the zip file name. (e.g., https://www.agency.org/gtfs/gtfs.zip).
- **Record** - A basic data structure comprised of a number of different field values describing a single entity (e.g. transit agency, stop, route, etc.). Represented, in a table, as a row.
- **Field** - A property of an object or entity. Represented, in a table, as a column.
- **Field value** - An individual entry in a field. Represented, in a table, as a single cell.
- **Service day** - A service day is a time period used to indicate route scheduling. The exact definition of service day varies from agency to agency but service days often do not correspond with calendar days. A service day may exceed 24:00:00 if service begins on one day and ends on a following day. For example, service that runs from 08:00:00 on Friday to 02:00:00 on Saturday, could be denoted as running from 08:00:00 to 26:00:00 on a single service day.
- **Text-to-speech field** - The field should contain the same information than its parent field (on which it falls back if it is empty). It is aimed to be read as text-to-speech, therefore, abbreviation should be either removed ("St" should be either read as "Street" or "Saint"; "Elizabeth I" should be "Elizabeth the first") or kept to be read as it ("JFK Airport" is said abbreviated).
- **Leg** - Travel in which a rider boards and alights between a pair of subsequent locations along a trip.
- **Journey** - Overall travel from origin to destination, including all legs and transfers in-between.
- **Sub-journey** - Two or more legs that comprise a subset of a journey.
- **Fare product** - Purchassable fare products that can be used to pay for or validate travel.

Presence

Presence conditions applicable to fields and files:

- **Required** - The field or file must be included in the dataset and contain a valid value for each record.
- **Optional** - The field or file may be omitted from the dataset.
- **Conditionally Required** - The field or file must be included under conditions outlined in the field or file description.
- **Conditionally Forbidden** - The field or file must not be included under conditions outlined in the field or file description.
- **Recommended** - The field or file may be omitted from the dataset, but it is a best practice to include it. Before omitting this field or file, the best practice should be carefully evaluated and the full implications of omission should be understood.

Field Types

- **Color** - A color encoded as a six-digit hexadecimal number. Refer to https://htmlcolorcodes.com to generate a valid value (the leading "#" must not be included).
  Example: `FFFFFF` for white, `000000` for black or `0039A6` for the A,C,E lines in NYMTA.
- **Currency code** - An ISO 4217 alphabetical currency code. For the list of current currency, refer to https://en.wikipedia.org/wiki/ISO_4217#Active_codes.
  Example: `CAD` for Canadian dollars, `EUR` for euros or `JPY` for Japanese yen.
- **Currency amount** - A decimal value indicating a currency amount. The number of decimal places is specified by ISO 4217 for the accompanying Currency code. All financial calculations should be processed as decimal, currency, or another equivalent type suitable for financial calculations depending on the programming language used to consume data. Processing currency amounts as float is discouraged due to gains or losses of money during calculations.
- **Date** - Service day in the YYYYMMDD format. Since time within a service day may be above 24:00:00, a service day may contain information for the subsequent day(s).
  Example: `20180913` for September 13th, 2018.
- **Email** - An email address.
  Example: `example@example.com`
- **Enum** - An option from a set of predefined constants defined in the "Description" column.
  Example: The `route_type` field contains a `0` for tram, a `1` for subway...
- **ID** - An ID field value is an internal ID, not intended to be shown to riders, and is a sequence of any UTF-8 characters. Using only printable ASCII characters is recommended. An ID is labeled "unique ID" when it must be unique within a file. IDs defined in one .txt file are often referenced in another .txt file. IDs that reference an ID in another table are labeled "foreign ID".
  Example: The `stop_id` field in stops.txt is a "unique ID". The `parent_station` field in stops.txt is a "foreign ID referencing `stops.stop_id`".
- **Language code** - An IETF BCP 47 language code. For an introduction to IETF BCP 47, refer to http://www.rfc-editor.org/rfc/bcp/bcp47.txt and http://www.w3.org/International/articles/language-tags/.
  Example: `en` for English, `en-US` for American English or `de` for German.
- **Latitude** - WGS84 latitude in decimal degrees. The value must be greater than or equal to -90.0 and less than or equal to 90.0.
  Example: `41.890169` for the Colosseum in Rome.
- **Longitude** - WGS84 longitude in decimal degrees. The value must be greater than or equal to -180.0 and less than or equal to 180.0.
  Example: `12.492269` for the Colosseum in Rome.
- **Float** - A floating point number.
- **Integer** - An integer.
- **Phone number** - A phone number.
- **Time** - Time in the HH:MM:SS format (H:MM:SS is also accepted). The time is measured from "noon minus 12h" of the service day (effectively midnight except for days on which daylight savings time changes occur). For times occurring after midnight on the service day, enter the time as a value greater than 24:00:00 in HH:MM:SS.
  Example: `14:30:00` for 2:30PM or `25:35:00` for 1:35AM on the next day.
- **Text** - A string of UTF-8 characters, which is aimed to be displayed and which must therefore be human readable.

- **Timezone** - TZ timezone from the https://www.iana.org/time-zones. Timezone names never contain the space character but may contain an underscore. Refer to http://en.wikipedia.org/wiki/List_of_tz_zones for a list of valid values.
  Example: `Asia/Tokyo`, `America/Los_Angeles` *or* `Africa/Cairo`.
- **URL** - A fully qualified URL that includes http:// or https://, and any special characters in the URL must be correctly escaped. See the following http://www.w3.org/Addressing/URL/4_URI_Recommentations.html for a description of how to create fully qualified URL values.

Field Signs

Signs applicable to Float or Integer field types:

- **Non-negative** - Greater than or equal to 0.
- **Non-zero** - Not equal to 0.
- **Positive** - Greater than 0.

Example: ***Non-negative float*** *- A floating point number greater than or equal to 0.*

Dataset Attributes

The **primary key** of a dataset is the field or combination of fields that uniquely identify a row. `Primary key (*)` is used when all provided fields for a file are used to uniquely identify a row. `Primary key (none)` means that the file allows only one row.

Example: the `trip_id` *and* `stop_sequence` *fields make the primary key of stop_times.txt.*

Dataset Files

This specification defines the following files:

File Name	Presence	Description
agency.txt	**Required**	Transit agencies with service represented in this dataset.
stops.txt	**Required**	Stops where vehicles pick up or drop off riders. Also defines stations and station entrances.
routes.txt	**Required**	Transit routes. A route is a group of trips that are displayed to riders as a single service.
trips.txt	**Required**	Trips for each route. A trip is a sequence of two or more stops that occur during a specific time period.
stop_times.txt	**Required**	Times that a vehicle arrives at and departs from stops for each trip.
calendar.txt	**Conditionally Required**	Service dates specified using a weekly schedule with start and end dates. Conditionally Required: - **Required** unless all dates of service are defined in calendar_dates.txt. - Optional otherwise.
calendar_dates.txt	**Conditionally Required**	Exceptions for the services defined in the calendar.txt. Conditionally Required: - **Required** if calendar.txt is omitted. In which case

File Name	Presence	Description
		calendar_dates.txt must contain all dates of service. - Optional otherwise.
fare_attributes.txt	Optional	Fare information for a transit agency's routes.
fare_rules.txt	Optional	Rules to apply fares for itineraries.
timeframes.txt	Optional	Date and time periods to use in fare rules for fares that depend on date and time factors.
fare_media.txt	Optional	To describe the fare media that can be employed to use fare products. File fare_media.txt describes concepts that are not represented in fare_attributes.txt and fare_rules.txt. As such, the use of fare_media.txt is entirely separate from files fare_attributes.txt and fare_rules.txt.
fare_products.txt	Optional	To describe the different types of tickets or fares that can be purchased by riders. File fare_products.txt describes fare products that are not represented in fare_attributes.txt and fare_rules.txt. As such, the use of fare_products.txt is entirely separate from files fare_attributes.txt and fare_rules.txt.
fare_leg_rules.txt	Optional	Fare rules for individual legs of travel. File fare_leg_rules.txt provides a more detailed method for modeling fare structures. As such, the use of fare_leg_rules.txt is entirely separate from files fare_attributes.txt and fare_rules.txt.
fare_transfer_rules.txt	Optional	Fare rules for transfers between legs of travel. Along with fare_leg_rules.txt, file fare_transfer_rules.txt provides a more detailed method for modeling fare structures. As such, the use of fare_transfer_rules.txt is entirely separate from files fare_attributes.txt and fare_rules.txt.
areas.txt	Optional	Area grouping of locations.
stop_areas.txt	Optional	Rules to assign stops to areas.
shapes.txt	Optional	Rules for mapping vehicle travel paths, sometimes referred to as route alignments.
frequencies.txt	Optional	Headway (time between trips) for headway-based service or a compressed representation of fixed-schedule service.
transfers.txt	Optional	Rules for making connections at transfer points between routes.
pathways.txt	Optional	Pathways linking together locations within stations.
levels.txt	**Conditionally Required**	Levels within stations. Conditionally Required: - **Required** when describing pathways with elevators (`pathway_mode=5`). - Optional otherwise.
translations.txt	Optional	Translations of customer-facing dataset values.

File Name	Presence	Description
feed_info.txt	Optional	Dataset metadata, including publisher, version, and expiration information.
attributions.txt	Optional	Dataset attributions.

File Requirements

The following requirements apply to the format and contents of the dataset files:

- All files must be saved as comma-delimited text.
- The first line of each file must contain field names. Each subsection of the Field Definitions section corresponds to one of the files in a GTFS dataset and lists the field names that may be used in that file.
- All file and field names are case-sensitive.
- Field values must not contain tabs, carriage returns or new lines.
- Field values that contain quotation marks or commas must be enclosed within quotation marks. In addition, each quotation mark in the field value must be preceded with a quotation mark. This is consistent with the manner in which Microsoft Excel outputs comma-delimited (CSV) files. For more information on the CSV file format, see http://tools.ietf.org/html/rfc4180. The following example demonstrates how a field value would appear in a comma-delimited file:
  - **Original field value:** `Contains "quotes", commas and text`
  - **Field value in CSV file:** `"Contains ""quotes"", commas and text"`
- Field values must not contain HTML tags, comments or escape sequences.
- Extra spaces between fields or field names should be removed. Many parsers consider the spaces to be part of the value, which may cause errors.
- Each line must end with a CRLF or LF linebreak character.
- Files should be encoded in UTF-8 to support all Unicode characters. Files that include the Unicode byte-order mark (BOM) character are acceptable. See http://unicode.org/faq/utf_bom.html#BOM for more information on the BOM character and UTF-8.
- All dataset files must be zipped together. The files must reside at the root level directly, not in a subfolder.

Field Definitions

agency.txt

File: **Required**

Primary key (`agency_id`)

Field Name	Type	Presence	Description
`agency_id`	Unique ID	**Conditionally Required**	Identifies a transit brand which is often synonymous with a transit agency. Note that in some cases, such as when a single agency operates multiple separate services, agencies and brands are distinct. This document uses the term "agency" in place of "brand". A dataset may contain data from multiple agencies. Conditionally Required: - **Required** when the dataset contains data for multiple transit agencies. - Recommended otherwise.

Field Name	Type	Presence	Description
agency_name	Text	**Required**	Full name of the transit agency.
agency_url	URL	**Required**	URL of the transit agency.
agency_timezone	Timezone	**Required**	Timezone where the transit agency is located. If multiple agencies are specified in the dataset, each must have the same agency_timezone.
agency_lang	Language code	Optional	Primary language used by this transit agency. Should be provided to help GTFS consumers choose capitalization rules and other language-specific settings for the dataset.
agency_phone	Phone number	Optional	A voice telephone number for the specified agency. This field is a string value that presents the telephone number as typical for the agency's service area. It may contain punctuation marks to group the digits of the number. Dialable text (for example, TriMet's "503-238-RIDE") is permitted, but the field must not contain any other descriptive text.
agency_fare_url	URL	Optional	URL of a web page that allows a rider to purchase tickets or other fare instruments for that agency online.
agency_email	Email	Optional	Email address actively monitored by the agency's customer service department. This email address should be a direct contact point where transit riders can reach a customer service representative at the agency.

stops.txt

File: **Required**

Primary key (stop_id)

Field Name	Type	Presence	Description
stop_id	Unique ID	**Required**	Identifies a location: stop/platform, station, entrance/exit, generic node or boarding area (see location_type). Multiple routes may use the same stop_id.
stop_code	Text	Optional	Short text or a number that identifies the location for riders. These codes are often used in phone-based transit information systems or printed on signage to make it easier for riders to get information for a particular location. The stop_code may be the same as stop_id if it is public facing. This field should be left empty for locations without a code presented to riders.
stop_name	Text	**Conditionally Required**	Name of the location. The stop_name should match the agency's rider-facing name for the location as printed on a timetable, published online, or

Field Name	Type	Presence	Description
			represented on signage. For translations into other languages, use `translations.txt`. When the location is a boarding area (`location_type=4`), the `stop_name` should contains the name of the boarding area as displayed by the agency. It could be just one letter (like on some European intercity railway stations), or text like “Wheelchair boarding area” (NYC’s Subway) or “Head of short trains” (Paris’ RER). Conditionally Required: - **Required** for locations which are stops (`location_type=0`), stations (`location_type=1`) or entrances/exits (`location_type=2`). - Optional for locations which are generic nodes (`location_type=3`) or boarding areas (`location_type=4`).
`tts_stop_name`	Text	Optional	Readable version of the `stop_name`. See "Text-to-speech field" in the Term Definitions for more.
`stop_desc`	Text	Optional	Description of the location that provides useful, quality information. Should not be a duplicate of `stop_name`.
`stop_lat`	Latitude	**Conditionally Required**	Latitude of the location. For stops/platforms (`location_type=0`) and boarding area (`location_type=4`), the coordinates must be the ones of the bus pole — if exists — and otherwise of where the travelers are boarding the vehicle (on the sidewalk or the platform, and not on the roadway or the track where the vehicle stops). Conditionally Required: - **Required** for locations which are stops (`location_type=0`), stations (`location_type=1`) or entrances/exits (`location_type=2`). - Optional for locations which are generic nodes (`location_type=3`) or boarding areas (`location_type=4`).
`stop_lon`	Longitude	**Conditionally Required**	Longitude of the location. For stops/platforms (`location_type=0`) and boarding area (`location_type=4`), the coordinates must be the ones of the bus pole — if exists — and otherwise of where the travelers are boarding the

Field Name	Type	Presence	Description
			vehicle (on the sidewalk or the platform, and not on the roadway or the track where the vehicle stops). Conditionally Required: - **Required** for locations which are stops (`location_type=0`), stations (`location_type=1`) or entrances/exits (`location_type=2`). - Optional for locations which are generic nodes (`location_type=3`) or boarding areas (`location_type=4`).
`zone_id`	ID	**Conditionally Required**	Identifies the fare zone for a stop. If this record represents a station or station entrance, the `zone_id` is ignored. Conditionally Required: - **Required** if providing fare information using fare_rules.txt - Optional otherwise.
`stop_url`	URL	Optional	URL of a web page about the location. This should be different from the `agency.agency_url` and the `routes.route_url` field values.
`location_type`	Enum	Optional	Location type. Valid options are: `0` (or blank) - **Stop** (or **Platform**). A location where passengers board or disembark from a transit vehicle. Is called a platform when defined within a `parent_station`. `1` - **Station**. A physical structure or area that contains one or more platform. `2` - **Entrance/Exit**. A location where passengers can enter or exit a station from the street. If an entrance/exit belongs to multiple stations, it may be linked by pathways to both, but the data provider must pick one of them as parent. `3` - **Generic Node**. A location within a station, not matching any other `location_type`, that may be used to link together pathways define in pathways.txt. `4` - **Boarding Area**. A specific location on a platform, where passengers can board and/or alight vehicles.
`parent_station`	Foreign ID referencing `stops.stop_id`	**Conditionally Required**	Defines hierarchy between the different locations defined in `stops.txt`. It contains the ID of the parent location, as followed: - **Stop/platform** (`location_type=0`):

Field Name	Type	Presence	Description
			the `parent_station` field contains the ID of a station. - **Station** (`location_type=1`): this field must be empty. - **Entrance/exit** (`location_type=2`) or **generic node** (`location_type=3`): the `parent_station` field contains the ID of a station (`location_type=1`) - **Boarding Area** (`location_type=4`): the `parent_station` field contains ID of a platform. Conditionally Required: - **Required** for locations which are entrances (`location_type=2`), generic nodes (`location_type=3`) or boarding areas (`location_type=4`). - Optional for stops/platforms (`location_type=0`). - Forbidden for stations (`location_type=1`).
`stop_timezone`	Timezone	Optional	Timezone of the location. If the location has a parent station, it inherits the parent station's timezone instead of applying its own. Stations and parentless stops with empty `stop_timezone` inherit the timezone specified by `agency.agency_timezone`. If `stop_timezone` values are provided, the times in stop_times.txt should be entered as the time since midnight in the timezone specified by `agency.agency_timezone`. This ensures that the time values in a trip always increase over the course of a trip, regardless of which timezones the trip crosses.
`wheelchair_boarding`	Enum	Optional	Indicates whether wheelchair boardings are possible from the location. Valid options are: For parentless stops: `0` or empty - No accessibility information for the stop. `1` - Some vehicles at this stop can be boarded by a rider in a wheelchair. `2` - Wheelchair boarding is not possible at this stop. For child stops: `0` or empty - Stop will inherit its `wheelchair_boarding` behavior from the parent station, if specified in the parent. `1` - There exists some accessible path

Field Name	Type	Presence	Description
			from outside the station to the specific stop/platform. `2` - There exists no accessible path from outside the station to the specific stop/platform. For station entrances/exits: `0` or empty - Station entrance will inherit its `wheelchair_boarding` behavior from the parent station, if specified for the parent. `1` - Station entrance is wheelchair accessible. `2` - No accessible path from station entrance to stops/platforms.
`level_id`	Foreign ID referencing `levels.level_id`	Optional	Level of the location. The same level may be used by multiple unlinked stations.
`platform_code`	Text	Optional	Platform identifier for a platform stop (a stop belonging to a station). This should be just the platform identifier (eg. "G" or "3"). Words like “platform” or "track" (or the feed’s language-specific equivalent) should not be included. This allows feed consumers to more easily internationalize and localize the platform identifier into other languages.

routes.txt

File: **Required**

Primary key (`route_id`)

Field Name	Type	Presence	Description
`route_id`	Unique ID	**Required**	Identifies a route.
`agency_id`	Foreign ID referencing `agency.agency_id`	**Conditionally Required**	Agency for the specified route. Conditionally Required: - **Required** if multiple agencies are defined in agency.txt. - Recommended otherwise.
`route_short_name`	Text	**Conditionally Required**	Short name of a route. Often a short, abstract identifier (e.g., "32", "100X", "Green") that riders use to identify a route. Both `route_short_name` and `route_long_name` may be defined. Conditionally Required: - **Required** if `routes.route_long_name` is empty. - Recommended if there is a brief service designation. This should be the

Field Name	Type	Presence	Description
			commonly-known passenger name of the service, and should be no longer than 12 characters.
`route_long_name`	Text	**Conditionally Required**	Full name of a route. This name is generally more descriptive than the `route_short_name` and often includes the route's destination or stop. Both `route_short_name` and `route_long_name` may be defined. Conditionally Required: - **Required** if `routes.route_short_name` is empty. - Optional otherwise.
`route_desc`	Text	Optional	Description of a route that provides useful, quality information. Should not be a duplicate of `route_short_name` or `route_long_name`. *Example: "A" trains operate between Inwood-207 St, Manhattan and Far Rockaway-Mott Avenue, Queens at all times. Also from about 6AM until about midnight, additional "A" trains operate between Inwood-207 St and Lefferts Boulevard (trains typically alternate between Lefferts Blvd and Far Rockaway).*
`route_type`	Enum	**Required**	Indicates the type of transportation used on a route. Valid options are: `0` - Tram, Streetcar, Light rail. Any light rail or street level system within a metropolitan area. `1` - Subway, Metro. Any underground rail system within a metropolitan area. `2` - Rail. Used for intercity or long-distance travel. `3` - Bus. Used for short- and long-distance bus routes. `4` - Ferry. Used for short- and long-distance boat service. `5` - Cable tram. Used for street-level rail cars where the cable runs beneath the vehicle (e.g., cable car in San Francisco). `6` - Aerial lift, suspended cable car (e.g., gondola lift, aerial tramway). Cable transport where cabins, cars, gondolas or open chairs are suspended by means of one or more cables. `7` - Funicular. Any rail system designed for steep inclines. `11` - Trolleybus. Electric buses that

Field Name	Type	Presence	Description
			draw power from overhead wires using poles. `12` - Monorail. Railway in which the track consists of a single rail or a beam.
`route_url`	URL	Optional	URL of a web page about the particular route. Should be different from the `agency.agency_url` value.
`route_color`	Color	Optional	Route color designation that matches public facing material. Defaults to white (`FFFFFF`) when omitted or left empty. The color difference between `route_color` and `route_text_color` should provide sufficient contrast when viewed on a black and white screen.
`route_text_color`	Color	Optional	Legible color to use for text drawn against a background of `route_color`. Defaults to black (`000000`) when omitted or left empty. The color difference between `route_color` and `route_text_color` should provide sufficient contrast when viewed on a black and white screen.
`route_sort_order`	Non-negative integer	Optional	Orders the routes in a way which is ideal for presentation to customers. Routes with smaller `route_sort_order` values should be displayed first.
`continuous_pickup`	Enum	Optional	Indicates that the rider can board the transit vehicle at any point along the vehicle's travel path as described by `shapes.txt`, on every trip of the route. Valid options are: `0` - Continuous stopping pickup. `1` or empty - No continuous stopping pickup. `2` - Must phone agency to arrange continuous stopping pickup. `3` - Must coordinate with driver to arrange continuous stopping pickup. Values for `routes.continuous_pickup` may be overridden by defining values in `stop_times.continuous_pickup` for specific `stop_time`s along the route.
`continuous_drop_off`	Enum	Optional	Indicates that the rider can alight from the transit vehicle at any point along the vehicle's travel path as described by `shapes.txt`, on every trip of the route. Valid options are: `0` - Continuous stopping drop off.

Field Name	Type	Presence	Description
			`1` or empty - No continuous stopping drop off. `2` - Must phone agency to arrange continuous stopping drop off. `3` - Must coordinate with driver to arrange continuous stopping drop off. Values for `routes.continuous_drop_off` may be overridden by defining values in `stop_times.continuous_drop_off` for specific `stop_time`s along the route.
`network_id`	ID	Optional	Identifies a group of routes. Multiple rows in routes.txt may have the same `network_id`.

trips.txt

File: **Required**

Primary key (`trip_id`)

Field Name	Type	Presence	Description
`route_id`	Foreign ID referencing `routes.route_id`	**Required**	Identifies a route.
`service_id`	Foreign ID referencing `calendar.service_id` or `calendar_dates.service_id`	**Required**	Identifies a set of dates when service is available for one or more routes.
`trip_id`	Unique ID	**Required**	Identifies a trip.
`trip_headsign`	Text	Optional	Text that appears on signage identifying the trip's destination to riders. Should be used to distinguish between different patterns of service on the same route. If the headsign changes during a trip, values for `trip_headsign` may be overridden by defining values in `stop_times.stop_headsign` for specific `stop_time`s along the trip.
`trip_short_name`	Text	Optional	Public facing text used to identify the trip to riders, for instance, to identify train numbers for commuter rail trips. If riders do not commonly rely on trip names, `trip_short_name` should be empty. A `trip_short_name` value, if provided, should uniquely identify a trip within a service day; it should not be used for destination names or limited/express designations.
`direction_id`	Enum	Optional	Indicates the direction of travel for a trip. This field should not be used in routing; it provides a way to separate trips by direction when publishing time tables. Valid options are:

Field Name	Type	Presence	Description

			`0` - Travel in one direction (e.g. outbound travel).
			`1` - Travel in the opposite direction (e.g. inbound travel).
			
			Example: The `trip_headsign` and `direction_id` fields may be used together to assign a name to travel in each direction for a set of trips. A trips.txt file could contain these records for use in time tables:
			`trip_id,...,trip_headsign,direction_id`
			`1234,...,Airport,0`
			`1505,...,Downtown,1`

`block_id`	ID	Optional	Identifies the block to which the trip belongs. A block consists of a single trip or many sequential trips made using the same vehicle, defined by shared service days and `block_id`. A `block_id` may have trips with different service days, making distinct blocks. See the example below. To provide in-seat transfers information, transfers of `transfer_type 4` should be provided instead.
`shape_id`	Foreign ID referencing `shapes.shape_id`	**Conditionally Required**	Identifies a geospatial shape describing the vehicle travel path for a trip. Conditionally Required: - **Required** if the trip has a continuous pickup or drop-off behavior defined either in `routes.txt` or in `stop_times.txt`. - Optional otherwise.
`wheelchair_accessible`	Enum	Optional	Indicates wheelchair accessibility. Valid options are: `0` or empty - No accessibility information for the trip. `1` - Vehicle being used on this particular trip can accommodate at least one rider in a wheelchair. `2` - No riders in wheelchairs can be accommodated on this trip.
`bikes_allowed`	Enum	Optional	Indicates whether bikes are allowed. Valid options are: `0` or empty - No bike information for the trip. `1` - Vehicle being used on this particular trip can accommodate at least one bicycle. `2` - No bicycles are allowed on this trip.

Example: Blocks and service day

The example below is valid, with distinct blocks every day of the week.

route_id	trip_id	service_id	block_id	*(first stop time)*	*(last stop time)*
red	trip_1	mon-tues-wed-thurs-fri-sat-sun	red_loop	22:00:00	22:55:00
red	trip_2	fri-sat-sun	red_loop	23:00:00	23:55:00
red	trip_3	fri-sat	red_loop	24:00:00	24:55:00
red	trip_4	mon-tues-wed-thurs	red_loop	20:00:00	20:50:00
red	trip_5	mon-tues-wed-thurs	red_loop	21:00:00	21:50:00

Notes on above table:

- On Friday into Saturday morning, for example, a single vehicle operates `trip_1`, `trip_2`, and `trip_3` (10:00 PM through 12:55 AM). Note that the last trip occurs on Saturday, 12:00 AM to 12:55 AM, but is part of the Friday "service day" because the times are 24:00:00 to 24:55:00.
- On Monday, Tuesday, Wednesday, and Thursday, a single vehicle operates `trip_1`, `trip_4`, and `trip_5` in a block from 8:00 PM to 10:55 PM.

stop_times.txt

File: **Required**

Primary key (`trip_id`, `stop_sequence`)

Field Name	Type	Presence	Description
`trip_id`	Foreign ID referencing `trips.trip_id`	**Required**	Identifies a trip.
`arrival_time`	Time	**Conditionally Required**	Arrival time at the stop (defined by `stop_times.stop_id`) for a specific trip (defined by `stop_times.trip_id`) in the time zone specified by `agency.agency_timezone`, not `stops.stop_timezone`. If there are not separate times for arrival and departure at a stop, `arrival_time` and `departure_time` should be the same. For times occurring after midnight on the service day, enter the time as a value greater than 24:00:00 in HH:MM:SS. If exact arrival and departure times (`timepoint=1` or empty) are not available, estimated or interpolated arrival and departure times (`timepoint=0`) should be provided. Conditionally Required: - **Required** for the first and last stop in a trip (defined by `stop_times.stop_sequence`). - **Required** for `timepoint=1`. - Optional otherwise.

Field Name	Type	Presence	Description
departure_time	Time	**Conditionally Required**	Departure time from the stop (defined by stop_times.stop_id) for a specific trip (defined by stop_times.trip_id) in the time zone specified by agency.agency_timezone, not stops.stop_timezone. If there are not separate times for arrival and departure at a stop, arrival_time and departure_time should be the same. For times occurring after midnight on the service day, enter the time as a value greater than 24:00:00 in HH:MM:SS. If exact arrival and departure times (timepoint=1 or empty) are not available, estimated or interpolated arrival and departure times (timepoint=0) should be provided. Conditionally Required: - **Required** for timepoint=1. - Optional otherwise.
stop_id	Foreign ID referencing stops.stop_id	**Required**	Identifies the serviced stop. All stops serviced during a trip must have a record in stop_times.txt. Referenced locations must be stops/platforms, i.e. their stops.location_type value must be 0 or empty. A stop may be serviced multiple times in the same trip, and multiple trips and routes may service the same stop.
stop_sequence	Non-negative integer	**Required**	Order of stops for a particular trip. The values must increase along the trip but do not need to be consecutive. *Example: The first location on the trip could have a stop_sequence=1, the second location on the trip could have a stop_sequence=23, the third location could have a stop_sequence=40, and so on.*
stop_headsign	Text	Optional	Text that appears on signage identifying the trip's destination to riders. This field overrides the default trips.trip_headsign when the headsign changes between stops. If the headsign is displayed for an entire trip, trips.trip_headsign should be used instead. A stop_headsign value specified for one stop_time does not apply to subsequent

Field Name	Type	Presence	Description
			stop_times in the same trip. If you want to override the trip_headsign for multiple stop_times in the same trip, the stop_headsign value must be repeated in each stop_time row.
pickup_type	Enum	Optional	Indicates pickup method. Valid options are: 0 or empty - Regularly scheduled pickup. 1 - No pickup available. 2 - Must phone agency to arrange pickup. 3 - Must coordinate with driver to arrange pickup.
drop_off_type	Enum	Optional	Indicates drop off method. Valid options are: 0 or empty - Regularly scheduled drop off. 1 - No drop off available. 2 - Must phone agency to arrange drop off. 3 - Must coordinate with driver to arrange drop off.
continuous_pickup	Enum	Optional	Indicates that the rider can board the transit vehicle at any point along the vehicle's travel path as described by shapes.txt, from this stop_time to the next stop_time in the trip's stop_sequence. Valid options are: 0 - Continuous stopping pickup. 1 or empty - No continuous stopping pickup. 2 - Must phone agency to arrange continuous stopping pickup. 3 - Must coordinate with driver to arrange continuous stopping pickup. If this field is populated, it overrides any continuous pickup behavior defined in routes.txt. If this field is empty, the stop_time inherits any continuous pickup behavior defined in routes.txt.
continuous_drop_off	Enum	Optional	Indicates that the rider can alight from the transit vehicle at any point along the vehicle's travel path as described by shapes.txt, from this stop_time to the next stop_time in the trip's stop_sequence. Valid options are: 0 - Continuous stopping drop off. 1 or empty - No continuous stopping drop off. 2 - Must phone agency to arrange continuous stopping drop off.

Field Name	Type	Presence	Description
			`3` - Must coordinate with driver to arrange continuous stopping drop off. If this field is populated, it overrides any continuous drop-off behavior defined in `routes.txt`. If this field is empty, the `stop_time` inherits any continuous drop-off behavior defined in `routes.txt`.
`shape_dist_traveled`	Non-negative float	Optional	Actual distance traveled along the associated shape, from the first stop to the stop specified in this record. This field specifies how much of the shape to draw between any two stops during a trip. Must be in the same units used in shapes.txt. Values used for `shape_dist_traveled` must increase along with `stop_sequence`; they must not be used to show reverse travel along a route. Recommended for routes that have looping or inlining (the vehicle crosses or travels over the same portion of alignment in one trip). See `shapes.shape_dist_traveled`. *Example: If a bus travels a distance of 5.25 kilometers from the start of the shape to the stop,* `shape_dist_traveled`*=*`5.25`*.*
`timepoint`	Enum	Recommended	Indicates if arrival and departure times for a stop are strictly adhered to by the vehicle or if they are instead approximate and/or interpolated times. This field allows a GTFS producer to provide interpolated stop-times, while indicating that the times are approximate. Valid options are: `0` - Times are considered approximate. `1` or empty - Times are considered exact.

calendar.txt

File: **Conditionally Required**

Primary key (`service_id`)

Field Name	Type	Presence	Description
`service_id`	Unique ID	**Required**	Identifies a set of dates when service is available for one or more routes. Each `service_id` value must be unique in a calendar.txt file.
`monday`	Enum	**Required**	Indicates whether the service operates on all Mondays in the date range specified by the `start_date` and `end_date` fields. Note that exceptions for particular dates may be listed in calendar_dates.txt. Valid options are:

Field Name	Type	Presence	Description
			`1` - Service is available for all Mondays in the date range. `0` - Service is not available for Mondays in the date range.
`tuesday`	Enum	**Required**	Functions in the same way as `monday` except applies to Tuesdays
`wednesday`	Enum	**Required**	Functions in the same way as `monday` except applies to Wednesdays
`thursday`	Enum	**Required**	Functions in the same way as `monday` except applies to Thursdays
`friday`	Enum	**Required**	Functions in the same way as `monday` except applies to Fridays
`saturday`	Enum	**Required**	Functions in the same way as `monday` except applies to Saturdays.
`sunday`	Enum	**Required**	Functions in the same way as `monday` except applies to Sundays.
`start_date`	Date	**Required**	Start service day for the service interval.
`end_date`	Date	**Required**	End service day for the service interval. This service day is included in the interval.

calendar_dates.txt

File: **Conditionally Required**

Primary key (`service_id`, `date`)

The calendar_dates.txt table explicitly activates or disables service by date. It may be used in two ways.

- Recommended: Use calendar_dates.txt in conjunction with calendar.txt to define exceptions to the default service patterns defined in calendar.txt. If service is generally regular, with a few changes on explicit dates (for instance, to accommodate special event services, or a school schedule), this is a good approach. In this case `calendar_dates.service_id` is a foreign ID referencing `calendar.service_id`.
- Alternate: Omit calendar.txt, and specify each date of service in calendar_dates.txt. This allows for considerable service variation and accommodates service without normal weekly schedules. In this case `service_id` is an ID.

Field Name	Type	Presence	Description
`service_id`	Foreign ID referencing `calendar.service_id` or ID	**Required**	Identifies a set of dates when a service exception occurs for one or more routes. Each (`service_id`, `date`) pair may only appear once in calendar_dates.txt if using calendar.txt and calendar_dates.txt in conjunction. If a `service_id` value appears in both calendar.txt and calendar_dates.txt, the information in calendar_dates.txt modifies the service information specified in calendar.txt.
`date`	Date	**Required**	Date when service exception occurs.
`exception_type`	Enum	**Required**	Indicates whether service is available on the date specified in the date field. Valid options are: `1` - Service has been added for the specified date. `2` - Service has been removed for the specified date.

Field Name	Type	Presence	Description
			Example: Suppose a route has one set of trips available on holidays and another set of trips available on all other days. One `service_id` *could correspond to the regular service schedule and another* `service_id` *could correspond to the holiday schedule. For a particular holiday, the calendar_dates.txt file could be used to add the holiday to the holiday* `service_id` *and to remove the holiday from the regular* `service_id` *schedule.*

fare_attributes.txt

File: **Optional**

Primary key (`fare_id`)

Versions
There are two modelling options for describing fares. GTFS-Fares V1 is the legacy option for describing minimal fare information. GTFS-Fares V2 is an updated method that allows for a more detailed account of an agency's fare structure. Both are allowed to be present in a dataset, but only one method should be used by a data consumer for a given dataset. It is recommended that GTFS-Fares V2 takes precedence over GTFS-Fares V1.

The files associated with GTFS-Fares V1 are:
- fare_attributes.txt
- fare_rules.txt

The files associated with GTFS-Fares V2 are:
- fare_media.txt
- fare_products.txt
- fare_leg_rules.txt
- fare_transfer_rules.txt

Field Name	Type	Presence	Description
`fare_id`	Unique ID	**Required**	Identifies a fare class.
`price`	Non-negative float	**Required**	Fare price, in the unit specified by `currency_type`.
`currency_type`	Currency code	**Required**	Currency used to pay the fare.
`payment_method`	Enum	**Required**	Indicates when the fare must be paid. Valid options are: `0` - Fare is paid on board. `1` - Fare must be paid before boarding.
`transfers`	Enum	**Required**	Indicates the number of transfers permitted on this fare. Valid options are: `0` - No transfers permitted on this fare. `1` - Riders may transfer once.

Field Name	Type	Presence	Description
			2 - Riders may transfer twice. empty - Unlimited transfers are permitted.
agency_id	Foreign ID referencing agency.agency_id	**Conditionally Required**	Identifies the relevant agency for a fare. Conditionally Required: - **Required** if multiple agencies are defined in agency.txt. - Recommended otherwise.
transfer_duration	Non-negative integer	Optional	Length of time in seconds before a transfer expires. When transfers=0 this field may be used to indicate how long a ticket is valid for or it may be left empty.

fare_rules.txt

File: **Optional**

Primary key (*)

The fare_rules.txt table specifies how fares in fare_attributes.txt apply to an itinerary. Most fare structures use some combination of the following rules:

- Fare depends on origin or destination stations.
- Fare depends on which zones the itinerary passes through.
- Fare depends on which route the itinerary uses.

For examples that demonstrate how to specify a fare structure with fare_rules.txt and fare_attributes.txt, see FareExamples in the GoogleTransitDataFeed open source project wiki.

Field Name	Type	Presence	Description
fare_id	Foreign ID referencing fare_attributes.fare_id	**Required**	Identifies a fare class.
route_id	Foreign ID referencing routes.route_id	Optional	Identifies a route associated with the fare class. If several routes with the same fare attributes exist, create a record in fare_rules.t for each route. *Example: If fare class "b" is valid on route "TSW" and "TSE", the fare_rules.txt file would contain these records for the fare class:* fare_id,route_id b,TSW b,TSE
origin_id	Foreign ID referencing stops.zone_id	Optional	Identifies an origin zone. If a fare class has multiple origin zones create a record in fare_rules.txt for each origin_id. *Example: If fare class "b" is valid for all travel originating from eit zone "2" or zone "8", the fare_rules.txt file would contain these records for the fare class:* fare_id,...,origin_id b,...,2 b,...,8
destination_id	Foreign ID referencing stops.zone_id	Optional	Identifies a destination zone. If a fare class has multiple destinat zones, create a record in fare_rules.txt for each destination_i

Field Name	Type	Presence	Description

			Example: The `origin_id` and `destination_id` fields could be used together to specify that fare class "b" is valid for travel between zones 3 and 4, and for travel between zones 3 and 5, t *fare_rules.txt file would contain these records for the fare class:*
			`fare_id,...,origin_id,destination_id`
			`b,...,3,4`
			`b,...,3,5`

`contains_id`	Foreign ID referencing `stops.zone_id`	Optional	Identifies the zones that a rider will enter while using a given fare class. Used in some systems to calculate correct fare class.
			
			Example: If fare class "c" is associated with all travel on the GRT route that passes through zones 5, 6, and 7 the fare_rules.txt wo contain these records:
			`fare_id,route_id,...,contains_id`
			`c,GRT,...,5`
			`c,GRT,...,6`
			`c,GRT,...,7`
			Because all `contains_id` zones must be matched for the fare to apply, an itinerary that passes through zones 5 and 6 but not zo 7 would not have fare class "c". For more detail, see https://code.google.com/p/googletransitdatafeed/wiki/FareExamp in the GoogleTransitDataFeed project wiki.

timeframes.txt

File: **Optional**

Primary key (*)

Used to describe fares that can vary based on the time of day, the day of the week, or a particular day in the year. Timeframes can be associated with fare products in `fare_leg_rules.txt`.
There must not be overlapping time intervals for the same `timeframe_group_id` and `service_id` values.

Field Name	Type	Presence	Description
`timeframe_group_id`	ID	**Required**	Identifies a timeframe or set of timeframes.
`start_time`	Time	**Conditionally Required**	Defines the beginning of a timeframe. The interval includes the start time. Values greater than `24:00:00` are forbidden. An empty value in `start_time` is considered `00:00:00`. Conditionally Required: - **Required** if `timeframes.end_time` is defined. - **Forbidden** otherwise
`end_time`	Time	**Conditionally Required**	Defines the end of a timeframe. The interval does not include the end time. Values greater than `24:00:00` are forbidden. An

Field Name	Type	Presence	Description
			empty value in `end_time` is considered `24:00:00`. Conditionally Required: - **Required** if `timeframes.start_time` is defined. - **Forbidden** otherwise
`service_id`	Foreign ID referencing `calendar.service_id` or `calendar_dates.service_id`	**Required**	Identifies a set of dates that a timeframe is in effect.

Timeframe Local Time Semantics

- When evaluating a fare event's time against `timeframes.txt`, the event time is computed in local time using the local timezone, as determined by the `stop_timezone`, if specified, of the stop or parent station for the fare event. If not specified, the feed's agency timezone should be used instead.
- The "current day" is the current date of the fare event's time, computed relative to the local timezone. The "current day" may be different from the service day of a fare leg's trip, especially for trips that extend past midnight.
- The "time-of-day" for the fare event is computed relative to "current day" using GTFS Time field-type semantics.

fare_media.txt

File: **Optional**

Primary Key (`fare_media_id`)

To describe the different fare media that can be employed to use fare products. Fare media are physical or virtual holders used for the representation and/or validation of a fare product.

Field Name	Type	Presence	Description
`fare_media_id`	Unique ID	**Required**	Identifies a fare media.
`fare_media_name`	Text	Optional	Name of the fare media. For fare media which are transit cards (`fare_media_type =2`) or mobile apps (`fare_media_type =4`), the `fare_media_name` should be included and should match the rider-facing name used by the organizations delivering them.
`fare_media_type`	Enum	**Required**	The type of fare media. Valid options are: `0` - None. Used when there is no fare media involved in purchasing or validating a fare product, such as paying cash to a driver or conductor with no physical ticket provided. `1` - Physical paper ticket that allows a passenger to take either a certain number of pre-purchased trips or unlimited trips within a fixed period of time. `2` - Physical transit card that has stored tickets, passes or monetary value. `3` - cEMV (contactless Europay, Mastercard and Visa) as an open-loop token container for account-based ticketing.

Field Name	Type	Presence	Description
			4 - Mobile app that have stored virtual transit cards, tickets, passes, or monetary value.

fare_products.txt

File: **Optional**

Primary Key (fare_product_id, fare_media_id)

To describe the different types of tickets or fares that can be purchased by riders.

Field Name	Type	Presence	Description
fare_product_id	ID	**Required**	Identifies a fare product.
fare_product_name	Text	Optional	The name of the fare product as displayed to riders.
fare_media_id	Foreign ID referencing fare_media.fare_media_id	Optional	Identifies a fare media that can be employed to use the fare product during the trip. When fare_media_id is empty, it is considered that the fare media is unknown.
amount	Currency amount	**Required**	The cost of the fare product. May be negative to represent transfer discounts. May be zero to represent a fare product that is free.
currency	Currency code	**Required**	The currency of the cost of the fare product.

fare_leg_rules.txt

File: **Optional**

Primary Key (network_id, from_area_id, to_area_id, from_timeframe_group_id, to_timeframe_group_id, fare_product_id)

Fare rules for individual legs of travel.

Fares in fare_leg_rules.txt must be queried by filtering all the records in the file to find rules that match the leg to be traveled by the rider.

To process the cost of a leg:

1. The file fare_leg_rules.txt must be filtered by the fields that define the characteristics of travel, these fields are:
    - fare_leg_rules.network_id
    - fare_leg_rules.from_area_id
    - fare_leg_rules.to_area_id
    - fare_leg_rules.from_timeframe_group_id
    - fare_leg_rules.to_timeframe_group_id

2. If the leg exactly matches a record in fare_leg_rules.txt based on the characteristics of travel, that record must be processed to determine the cost of the leg.

3. If no exact matches are found, then empty entries in `fare_leg_rules.network_id`, `fare_leg_rules.from_area_id`, and `fare_leg_rules.to_area_id` must be checked to process the cost of the leg:
    - An empty entry in `fare_leg_rules.network_id` corresponds to all networks defined in `routes.txt` excluding the ones listed under `fare_leg_rules.network_id`
    - An empty entry in `fare_leg_rules.from_area_id` corresponds to all areas defined in `areas.area_id` excluding the ones listed under `fare_leg_rules.from_area_id`
    - An empty entry in `fare_leg_rules.to_area_id` corresponds to all areas defined in `areas.area_id` excluding the ones listed under `fare_leg_rules.to_area_id`
4. If the leg does not match any of the rules described above, then the fare is unknown.

Field Name	Type	Presence	Description
`leg_group_id`	ID	Optional	Identifies a group of entries in fare_leg_rules.txt. Used to describe fare transfer rules between `fare_transfer_rules.from_leg_group_id` and `fare_transfer_rules.to_leg_group_id`. Multiple entries in fare_leg_rules.txt may belong to the same `fare_leg_rules.leg_group_id`. The same entry in fare_leg_rules.txt (not including `fare_leg_rules.leg_group_id`) must not belong to multiple `fare_leg_rules.leg_group_id`.
`network_id`	Foreign ID referencing `routes.network_id`	Optional	Identifies a route network that applies for the fare leg rule. If there are no matching `fare_leg_rules.network_id` values to the `network_id` being filtered, empty `fare_leg_rules.network_id` will be matched by default. An empty entry in `fare_leg_rules.network_id` corresponds t all networks defined in `routes.txt` excluding the ones listed under `fare_leg_rules.network_id`
`from_area_id`	Foreign ID referencing `areas.area_id`	Optional	Identifies a departure area. If there are no matching `fare_leg_rules.from_area_id` values to th `area_id` being filtered, empty `fare_leg_rules.from_area_id` will be matched by default. An empty entry in `fare_leg_rules.from_area_id` correspond to all areas defined in `areas.area_id`

Field Name	Type	Presence	Description
			excluding the ones listed under `fare_leg_rules.from_area_id`
`to_area_id`	Foreign ID referencing `areas.area_id`	Optional	Identifies an arrival area. If there are no matching `fare_leg_rules.to_area_id` values to the `area_id` being filtered, empty `fare_leg_rules.to_area_id` will be matched by default. An empty entry in `fare_leg_rules.to_area_id` corresponds t all areas defined in `areas.area_id` excludin the ones listed under `fare_leg_rules.to_area_id`
`from_timeframe_group_id`	Foreign ID referencing `timeframes.timeframe_group_id`	Optional	Defines the timeframe for the fare validation event at the start of the fare leg. The “start time” of the fare leg is the time at which the event is scheduled to occur. For example, the time could be the scheduled departure time of a bus at the start of a fare le where the rider boards and validates their fare For the rule matching semantics below, the start time is computed in local time, as determined by Local Time Semantics of timeframes.txt. The stop or station of the fare leg’s departure event should be used for timezone resolution, where appropriate. For a fare leg rule that specifies a `from_timeframe_group_id`, that rule will match a particular leg if there exists at least one record in `timeframes.txt` where all of th following conditions are true - The value of `timeframe_group_id` is equal to the `from_timeframe_group_id` value. - The set of days identified by the record’s `service_id` contains the “current day” of the fare leg’s start time. - The “time-of-day” of the fare leg's start time greater than or equal to the record’s `timeframes.start_time` value and less than the `timeframes.end_time` value. An empty `fare_leg_rules.from_timeframe_group_i` indicates that the start time of the leg does no affect the matching of this rule.
`to_timeframe_group_id`	Foreign ID referencing `timeframes.timeframe_group_id`	Optional	Defines the timeframe for the fare validation event at the end of the fare leg. The “end time” of the fare leg is the time at which the event is scheduled to occur. For

Field Name	Type	Presence	Description
			example, the time could be the scheduled arrival time of a bus at the end of a fare leg where the rider gets off and validates their far For the rule matching semantics below, the end time is computed in local time, as determined by Local Time Semantics of timeframes.txt. The stop or station of the fare leg's arrival event should be used for timezon resolution, where appropriate. For a fare leg rule that specifies a `to_timeframe_group_id`, that rule will match a particular leg if there exists at least one record in `timeframes.txt` where all of the following conditions are true - The value of `timeframe_group_id` is equal to the `to_timeframe_group_id` value. - The set of days identified by the record's `service_id` contains the "current day" of the fare leg's end time. - The "time-of-day" of the fare leg's end time i greater than or equal to the record's `timeframes.start_time` value and less than the `timeframes.end_time` value. An empty `fare_leg_rules.to_timeframe_group_id` indicates that the end time of the leg does not affect the matching of this rule.
`fare_product_id`	Foreign ID referencing `fare_products.fare_product_id`	**Required**	The fare product required to travel the leg.

fare_transfer_rules.txt

File: **Optional**

Primary Key (`from_leg_group_id, to_leg_group_id, fare_product_id, transfer_count, duration_limit`)

Fare rules for transfers between legs of travel defined in `fare_leg_rules.txt`.

To process the cost of a multi-leg journey:

1. The applicable fare leg groups defined in `fare_leg_rules.txt` should be determined for all individual legs of travel based on the rider's journey.
2. The file `fare_transfer_rules.txt` must be filtered by the fields that define the characteristics of the transfer, these fields are:
    - `fare_transfer_rules.from_leg_group_id`
    - `fare_transfer_rules.to_leg_group_id`
3. If the transfer exactly matches a record in `fare_transfer_rules.txt` based on the characteristics of the transfer, then that record must be processed to determine the transfer cost.

4. If no exact matches are found, then empty entries in `from_leg_group_id` or in `to_leg_group_id` must be checked to process the transfer cost:
    - An empty entry in `fare_transfer_rules.from_leg_group_id` corresponds to all leg groups defined under `fare_leg_rules.leg_group_id` excluding the ones listed under `fare_transfer_rules.from_leg_group_id`
    - An empty entry in `fare_transfer_rules.to_leg_group_id` corresponds to all leg groups defined under `fare_leg_rules.leg_group_id` excluding the ones listed under `fare_transfer_rules.to_leg_group_id`

5. If the transfer does not match any of the rules described above, then there is no transfer arrangement and the legs are considered separate.

Field Name	Type	Presence	Description
`from_leg_group_id`	Foreign ID referencing `fare_leg_rules.leg_group_id`	Optional	Identifies a group of pre-transfer fare leg rules If there are no matching `fare_transfer_rules.from_leg_group_id` to the `leg_group_id` being filtered, empty `fare_transfer_rules.from_leg_group_id` matched by default. An empty entry in `fare_transfer_rules.from_leg_group_id` corresponds to all leg groups defined under `fare_leg_rules.leg_group_id` excluding th listed under `fare_transfer_rules.from_leg_group_id`
`to_leg_group_id`	Foreign ID referencing `fare_leg_rules.leg_group_id`	Optional	Identifies a group of post-transfer fare leg rules If there are no matching `fare_transfer_rules.to_leg_group_id` va the `leg_group_id` being filtered, empty `fare_transfer_rules.to_leg_group_id` wi matched by default. An empty entry in `fare_transfer_rules.to_leg_group_id` corresponds to all leg groups defined under `fare_leg_rules.leg_group_id` excluding th listed under `fare_transfer_rules.to_leg_group_id`
`transfer_count`	Non-zero integer	**Conditionally Forbidden**	Defines how many consecutive transfers the tr rule may be applied to. Valid options are: `-1` - No limit. `1` or more - Defines how many transfers the tra rule may span. If a sub-journey matches multiple records with different `transfer_count`s, then the rule with minimum `transfer_count` that is greater than equal to the current transfer count of the sub-j is to be selected.

Field Name	Type	Presence	Description
			Conditionally Forbidden: - **Forbidden** if `fare_transfer_rules.from_leg_group_id` not equal `fare_transfer_rules.to_leg_gr` - **Required** if `fare_transfer_rules.from_leg_group_id` `fare_transfer_rules.to_leg_group_id`.
`duration_limit`	Positive integer	Optional	Defines the duration limit of the transfer. Must be expressed in integer increments of se If there is no duration limit, `fare_transfer_rules.duration_limit` mu empty.
`duration_limit_type`	Enum	**Conditionally Required**	Defines the relative start and end of `fare_transfer_rules.duration_limit`. Valid options are: `0` - Between the departure fare validation of th current leg and the arrival fare validation of the leg. `1` - Between the departure fare validation of th current leg and the departure fare validation of next leg. `2` - Between the arrival fare validation of the cu leg and the departure fare validation of the nex `3` - Between the arrival fare validation of the cu leg and the arrival fare validation of the next le Conditionally Required: - **Required** if `fare_transfer_rules.duration_limit` is d - **Forbidden** if `fare_transfer_rules.duration_limit` is e
`fare_transfer_type`	Enum	**Required**	Indicates the cost processing method of transf between legs in a journey: Valid options are: `0` - From-leg `fare_leg_rules.fare_product` plus `fare_transfer_rules.fare_product_` AB. `1` - From-leg `fare_leg_rules.fare_product` plus `fare_transfer_rules.fare_product_` to-leg `fare_leg_rules.fare_product_id`; A B. `2` - `fare_transfer_rules.fare_product_id` Cost processing interactions between multiple transfers in a journey:

`fare_transfer_type`	Processing A > B	Proc B > C
`0`	A + AB	S + B

Field Name	Type	Presence	Description
			`1` Processing A > B `2` Processing AB `fare_transfer_type` Where S indicates the total processed cost of preceding leg(s) and transfer(s).
`fare_product_id`	Foreign ID referencing `fare_products.fare_product_id`	Optional	The fare product required to transfer between legs. If empty, the cost of the transfer rule is 0.

areas.txt

File: **Optional**

Primary key (`area_id`)

Defines area identifiers.

Field Name	Type	Presence	Description
`area_id`	Unique ID	**Required**	Identifies an area. Must be unique in areas.txt.
`area_name`	Text	**Optional**	The name of the area as displayed to the rider.

stop_areas.txt

File: **Optional**

Primary key (`*`)

Assigns stops from stops.txt to areas.

Field Name	Type	Presence	Description
`area_id`	Foreign ID referencing `areas.area_id`	**Required**	Identifies an area to which one or multiple `stop_id`s belong. The same `stop_id` may be defined in many `area_id`s.
`stop_id`	Foreign ID referencing `stops.stop_id`	**Required**	Identifies a stop. If a station (i.e. a stop with `stops.location_type=1`) is defined in this field, it is assumed that all of its platforms (i.e. all stops with `stops.location_type=0` that have this station defined as `stops.parent_station`) are part of the same area. This behavior can be overridden by assigning platforms to other areas.

shapes.txt

File: **Optional**

Primary key (`shape_id`, `shape_pt_sequence`)

Shapes describe the path that a vehicle travels along a route alignment, and are defined in the file shapes.txt. Shapes are associated with Trips, and consist of a sequence of points through which the vehicle passes in order. Shapes do not need to intercept the location of Stops exactly, but all Stops on a trip should lie within a small distance of the shape for that trip, i.e. close to straight line segments connecting the shape points.

Field Name	Type	Presence	Description
shape_id	ID	**Required**	Identifies a shape.
shape_pt_lat	Latitude	**Required**	Latitude of a shape point. Each record in shapes.txt represents a shape poin define the shape.
shape_pt_lon	Longitude	**Required**	Longitude of a shape point.
shape_pt_sequence	Non-negative integer	**Required**	Sequence in which the shape points connect to form the shape. Values must along the trip but do not need to be consecutive. *Example: If the shape "A_shp" has three points in its definition, the shapes.tx contain these records to define the shape:* shape_id,shape_pt_lat,shape_pt_lon,shape_pt_sequence A_shp,37.61956,-122.48161,0 A_shp,37.64430,-122.41070,6 A_shp,37.65863,-122.30839,11
shape_dist_traveled	Non-negative float	Optional	Actual distance traveled along the shape from the first shape point to the poi in this record. Used by trip planners to show the correct portion of the shape Values must increase along with shape_pt_sequence; they must not be use reverse travel along a route. Distance units must be consistent with those us stop_times.txt. Recommended for routes that have looping or inlining (the vehicle crosses o the same portion of alignment in one trip). If a vehicle retraces or crosses the route alignment at points in the course of shape_dist_traveled is important to clarify how portions of the points in sh line up correspond with records in stop_times.txt. *Example: If a bus travels along the three points defined above for A_shp, the shape_dist_traveled values (shown here in kilometers) would look like thi* shape_id,shape_pt_lat,shape_pt_lon,shape_pt_sequence,shape_dis A_shp,37.61956,-122.48161,0,0 A_shp,37.64430,-122.41070,6,6.8310 A_shp,37.65863,-122.30839,11,15.8765

frequencies.txt

File: **Optional**

Primary key (trip_id, start_time)

Frequencies.txt represents trips that operate on regular headways (time between trips). This file may be used to represent two different types of service.

- Frequency-based service (exact_times=0) in which service does not follow a fixed schedule throughout the day. Instead, operators attempt to strictly maintain predetermined headways for trips.
- A compressed representation of schedule-based service (exact_times=1) that has the exact same headway for trips over specified time period(s). In schedule-based service operators try to strictly adhere to a schedule.

Field Name	Type	Presence	Description
trip_id	Foreign ID referencing trips.trip_id	**Required**	Identifies a trip to which the specified headway of service applies.

Field Name	Type	Presence	Description
start_time	Time	**Required**	Time at which the first vehicle departs from the first stop of the trip with the specified headway.
end_time	Time	**Required**	Time at which service changes to a different headway (or ceases) at the first stop in the trip.
headway_secs	Positive integer	**Required**	Time, in seconds, between departures from the same stop (headway) for the trip, during the time interval specified by start_time and end_time. Multiple headways may be defined for the same trip, but must not overlap. New headways may start at the exact time the previous headway ends.
exact_times	Enum	Optional	Indicates the type of service for a trip. See the file description for more information. Valid options are: 0 or empty - Frequency-based trips. 1 - Schedule-based trips with the exact same headway throughout the day. In this case the end_time value must be greater than the last desired trip start_time but less than the last desired trip start_time + headway_secs.

transfers.txt

File: **Optional**

Primary key (from_stop_id, to_stop_id, from_trip_id, to_trip_id, from_route_id, to_route_id)

When calculating an itinerary, GTFS-consuming applications interpolate transfers based on allowable time and stop proximity. Transfers.txt specifies additional rules and overrides for selected transfers.

Fields from_trip_id, to_trip_id, from_route_id and to_route_id allow higher orders of specificity for transfer rules. Along with from_stop_id and to_stop_id, the ranking of specificity is as follows:

1. Both trip_ids defined: from_trip_id and to_trip_id.
2. One trip_id and route_id set defined: (from_trip_id and to_route_id) or (from_route_id and to_trip_id).
3. One trip_id defined: from_trip_id or to_trip_id.
4. Both route_ids defined: from_route_id and to_route_id.
5. One route_id defined: from_route_id or to_route_id.
6. Only from_stop_id and to_stop_id defined: no route or trip related fields set.

For a given ordered pair of arriving trip and departing trip, the transfer with the greatest specificity that applies between these two trips is chosen. For any pair of trips, there should not be two transfers with equally maximal specificity that could apply.

Field Name	Type	Presence	Description
from_stop_id	Foreign ID referencing stops.stop_id	**Conditionally Required**	Identifies a stop or station where a connection between routes begins. If this field refers to a station, the transfer rule applies to all its child stops. Refering to a station is forbiden for transfer_types 4 and 5.
to_stop_id	Foreign ID referencing stops.stop_id	**Conditionally Required**	Identifies a stop or station where a connection between routes ends. If this field refers to a station, the transfer rule applies to all child stops. Refering to a

Field Name	Type	Presence	Description
			station is forbiden for `transfer_types` 4 and 5.
`from_route_id`	Foreign ID referencing `routes.route_id`	Optional	Identifies a route where a connection begins. If `from_route_id` is defined, the transfer will apply to the arriving trip on the route for the given `from_stop_id`. If both `from_trip_id` and `from_route_id` are defined, the `trip_id` must belong to the `route_id`, and `from_trip_id` will take precedence.
`to_route_id`	Foreign ID referencing `routes.route_id`	Optional	Identifies a route where a connection ends. If `to_route_id` is defined, the transfer will apply to the departing trip on the route for the given `to_stop_id`. If both `to_trip_id` and `to_route_id` are defined, the `trip_id` must belong to the `route_id`, and `to_trip_id` will take precedence.
`from_trip_id`	Foreign ID referencing `trips.trip_id`	**Conditionally Required**	Identifies a trip where a connection between routes begins. If `from_trip_id` is defined, the transfer will apply to the arriving trip for the given `from_stop_id`. If both `from_trip_id` and `from_route_id` are defined, the `trip_id` must belong to the `route_id`, and `from_trip_id` will take precedence. REQUIRED if `transfer_type` is `4` or `5`.
`to_trip_id`	Foreign ID referencing `trips.trip_id`	**Conditionally Required**	Identifies a trip where a connection between routes ends. If `to_trip_id` is defined, the transfer will apply to the departing trip for the given `to_stop_id`. If both `to_trip_id` and `to_route_id` are defined, the `trip_id` must belong to the `route_id`, and `to_trip_id` will take precedence. REQUIRED if `transfer_type` is `4` or `5`.
`transfer_type`	Enum	**Required**	Indicates the type of connection for the specified (`from_stop_id`, `to_stop_id`) pair. Valid options are: `0` or empty - Recommended transfer

Field Name	Type	Presence	Description
			point between routes. `1` - Timed transfer point between two routes. The departing vehicle is expected to wait for the arriving one and leave sufficient time for a rider to transfer between routes. `2` - Transfer requires a minimum amount of time between arrival and departure to ensure a connection. The time required to transfer is specified by `min_transfer_time`. `3` - Transfers are not possible between routes at the location. `4` - Passengers can transfer from one trip to another by staying onboard the same vehicle (an "in-seat transfer"). More details about this type of transfer below. `5` - In-seat transfers are not allowed between sequential trips. The passenger must alight from the vehicle and re-board. More details about this type of transfer below.
`min_transfer_time`	Non-negative integer	Optional	Amount of time, in seconds, that must be available to permit a transfer between routes at the specified stops. The `min_transfer_time` should be sufficient to permit a typical rider to move between the two stops, including buffer time to allow for schedule variance on each route.

Linked trips

The following applies to `transfer_type=4` and `=5`, which are used to link trips together, with or without in-seats transfers.

The trips linked together MUST be operated by the same vehicle. The vehicle MAY be coupled to, or uncoupled from, other vehicles.

If both a linked trips transfer and a block_id are provided and they produce conflicting results, then the linked trips transfer shall be used.

The last stop of `from_trip_id` SHOULD be geographically close to the first stop of `to_trip_id`, and the last arrival time of `from_trip_id` SHOULD be prior but close to the first departure time of `to_trip_id`. The last arrival time of `from_trip_id` MAY be later than the first departure time of `to_trip_id` in case the `to_trip_id` trip is occurring the subsequent service day.

Trips MAY be linked 1-to-1 in the regular case, but MAY also be linked 1-to-n, n-to-1, or n-to-n to represent more complex trip continuations. For example, two train trips (trip A and trip B in the diagram below) can merge into a single train trip (trip C) after a vehicle coupling operation at a common station:

- In a 1-to-n continuation, the `trips.service_id` for each `to_trip_id` MUST be identical.
- In an n-to-1 continuation, the `trips.service_id` for each `from_trip_id` MUST be identical.
- n-to-n continuations must respect both constraints.
- Trips may be linked together as part of multiple distinct continuations, provided that the `trip.service_id` MUST NOT overlap on any day of service.

pathways.txt

File: **Optional**

Primary key (`pathway_id`)

Files pathways.txt and levels.txt use a graph representation to describe subway or train stations, with nodes representing locations and edges representing pathways.

To navigate from the station entrance/exit (a node represented as a location with `location_type=2`) to a platform (a node represented as a location with `location_type=0` or empty), the rider will move through walkways, fare gates, stairs, and other edges represented as pathways. Generic nodes (nodes represented with `location_type=3`) can be used to connect pathways throughout a station.

Pathways must be defined exhaustively in a station. If any pathways are defined, it is assumed that all pathways throughout the station are represented. Therefore, the following guidelines apply:

- No dangling locations: If any location within a station has a pathway, then all locations within that station should have pathways, except for platforms that have boarding areas (`location_type=4`, see guideline below).
- No pathways for a platform with boarding areas: A platform (`location_type=0` or empty) that has boarding areas (`location_type=4`) is treated as a parent object, not a point. In such cases, the platform must not have pathways assigned. All pathways should be assigned for each of the platform's boarding areas.
- No locked platforms: Each platform (`location_type=0` or empty) or boarding area (`location_type=4`) must be connected to at least one entrance/exit (`location_type=2`) via some chain of pathways. Stations not allowing a pathway to the outside of the station from a given platform are rare.

Field Name	Type	Presence	Description
`pathway_id`	Unique ID	**Required**	Identifies a pathway. Used by systems as an internal identifier for the record. Must be unique in the dataset. Different pathways may have the same values for `from_stop_id` and `to_stop_id`. *Example: When two escalators are side-by-side in opposite directions, or when a stair set and elevator go from the same place to the same place, different `pathway_id` may have the same `from_stop_id` and `to_stop_id` values.*
`from_stop_id`	Foreign ID referencing `stops.stop_id`	**Required**	Location at which the pathway begins. Must contain a `stop_id` that identifies a platform (`location_type=0` or empty), entrance/exit (`location_type=2`), generic node (`location_type=3`) or boarding area (`location_type=4`).

Field Name	Type	Presence	Description
			Values for `stop_id` that identify stations (`location_type=1`) are forbidden.
`to_stop_id`	Foreign ID referencing `stops.stop_id`	**Required**	Location at which the pathway ends. Must contain a `stop_id` that identifies a platform (`location_type=0` or empty), entrance/exit (`location_type=2`), generic node (`location_type=3`) or boarding area (`location_type=4`). Values for `stop_id` that identify stations (`location_type=1`) are forbidden.
`pathway_mode`	Enum	**Required**	Type of pathway between the specified (`from_stop_id`, `to_stop_id`) pair. Valid options are: `1` - Walkway. `2` - Stairs. `3` - Moving sidewalk/travelator. `4` - Escalator. `5` - Elevator. `6` - Fare gate (or payment gate): A pathway that crosses into an area of the station where proof of payment is required to cross. Fare gates may separate paid areas of the station from unpaid ones, or separate different payment areas within the same station from each other. This information can be used to avoid routing passengers through stations using shortcuts that would require passengers to make unnecessary payments, like directing a passenger to walk through a subway platform to reach a busway. `7`- Exit gate: A pathway exiting a paid area into an unpaid area where proof of payment is not required to cross.
`is_bidirectional`	Enum	**Required**	Indicates the direction that the pathway can be taken: `0` - Unidirectional pathway that can only be used from `from_stop_id` to `to_stop_id`. `1` - Bidirectional pathway that can be used in both directions. Exit gates (`pathway_mode=7`) must not be bidirectional.
`length`	Non-negative float	Optional	Horizontal length in meters of the pathway from the origin location (defined in `from_stop_id`) to the destination location (defined in `to_stop_id`). This field is recommended for walkways (`pathway_mode=1`), fare gates

Field Name	Type	Presence	Description
			(`pathway_mode=6`) and exit gates (`pathway_mode=7`).
`traversal_time`	Positive integer	Optional	Average time in seconds needed to walk through the pathway from the origin location (defined in `from_stop_id`) to the destination location (defined in `to_stop_id`). This field is recommended for moving sidewalks (`pathway_mode=3`), escalators (`pathway_mode=4`) and elevator (`pathway_mode=5`).
`stair_count`	Non-null integer	Optional	Number of stairs of the pathway. A positive `stair_count` implies that the rider walk up from `from_stop_id` to `to_stop_id`. And a negative `stair_count` implies that the rider walk down from `from_stop_id` to `to_stop_id`. This field is recommended for stairs (`pathway_mode=2`). If only an estimated stair count can be provided, it is recommended to approximate 15 stairs for 1 floor.
`max_slope`	Float	Optional	Maximum slope ratio of the pathway. Valid options are: `0` or empty - No slope. `Float` - Slope ratio of the pathway, positive for upwards, negative for downwards. This field should only be used with walkways (`pathway_mode=1`) and moving sidewalks (`pathway_mode=3`). *Example: In the US, 0.083 (also written 8.3%) is the maximum slope ratio for hand-propelled wheelchair, which mean an increase of 0.083m (so 8.3cm) for each 1m.*
`min_width`	Positive float	Optional	Minimum width of the pathway in meters. This field is recommended if the minimum width is less than 1 meter.
`signposted_as`	Text	Optional	Public facing text from physical signage that is visible to riders. May be used to provide text directions to riders, such as 'follow signs to '. The text in `singposted_as` should appear exactly how it is printed on the signs.

Field Name	Type	Presence	Description
			When the physical signage is multilingual, this field may be populated and translated following the example of `stops.stop_name` in the field definition of `feed_info.feed_lang`.
`reversed_signposted_as`	Text	Optional	Same as `signposted_as`, but when the pathway is used from the `to_stop_id` to the `from_stop_id`.

levels.txt

File: **Conditionally Required**

Primary key (`level_id`)

Describes levels in a station. Useful in conjunction with `pathways.txt`, and is required for navigating pathways with elevators (`pathway_mode=5`).

Field Name	Type	Presence	Description
`level_id`	Unique ID	**Required**	Identifies a level in a station.
`level_index`	Float	**Required**	Numeric index of the level that indicates its relative position. Ground level should have index `0`, with levels above ground indicated by positive indices and levels below ground by negative indices.
`level_name`	Text	Optional	Name of the level as seen by the rider inside the building or station. *Example: Take the elevator to "Mezzanine" or "Platform" or "-1".*

translations.txt

File: **Optional**

Primary key (`table_name`, `field_name`, `language`, `record_id`, `record_sub_id`, `field_value`)

In regions that have multiple official languages, transit agencies/operators typically have language-specific names and web pages. In order to best serve riders in those regions, it is useful for the dataset to include these language-dependent values.

Field Name	Type	Presence	Description
`table_name`	Enum	**Required**	Defines the table that contains the field to be translated. Allowed values are: - `agency` - `stops` - `routes` - `trips` - `stop_times` - `pathways` - `levels` - `feed_info` - `attributions`

Field Name	Type	Presence	Description
			Any file added to GTFS will have a `table_name` value equivalent to the file name, as listed above (i.e., not including the `.txt` file extension).
`field_name`	Text	**Required**	Name of the field to be translated. Fields with type `Text` may be translated, fields with type `URL`, `Email` and `Phone number` may also be “translated” to provide resources in the correct language. Fields with other types should not be translated.
`language`	Language code	**Required**	Language of translation. If the language is the same as in `feed_info.feed_lang`, the original value of the field will be assumed to be the default value to use in languages without specific translations (if `default_lang` doesn't specify otherwise). *Example: In Switzerland, a city in an officially bilingual canton is officially called “Biel/Bienne”, but would simply be called “Bienne” in French and “Biel” in German.*
`translation`	Text or URL or Email or Phone number	**Required**	Translated value.
`record_id`	Foreign ID	**Conditionally Required**	Defines the record that corresponds to the field to be translated. The value in `record_id` must be the first or only field of a table's primary key, as defined in the primary key attribute for each table and below: - `agency_id` for `agency.txt` - `stop_id` for `stops.txt`; - `route_id` for `routes.txt`; - `trip_id` for `trips.txt`; - `trip_id` for `stop_times.txt`; - `pathway_id` for `pathways.txt`; - `level_id` for `levels.txt`; - `attribution_id` for `attribution.txt`. Fields in tables not defined above should not be translated. However producers sometimes add extra fields that are outside the official specification and these unofficial fields may be translated. Below is the recommended way to use `record_id` for those tables: - `service_id` for `calendar.txt`; - `service_id` for `calendar_dates.txt`; - `fare_id` for `fare_attributes.txt`; - `fare_id` for `fare_rules.txt`; - `shape_id` for `shapes.txt`; - `trip_id` for `frequencies.txt`; - `from_stop_id` for `transfers.txt`.

Field Name	Type	Presence	Description
			Conditionally Required: - **Forbidden** if `table_name` is `feed_info`. - **Forbidden** if `field_value` is defined. - **Required** if `field_value` is empty.
`record_sub_id`	Foreign ID	**Conditionally Required**	Helps the record that contains the field to be translated when the table doesn't have a unique ID. Therefore, the value in `record_sub_id` is the secondary ID of the table, as defined by the table below: - None for `agency.txt`; - None for `stops.txt`; - None for `routes.txt`; - None for `trips.txt`; - `stop_sequence` for `stop_times.txt`; - None for `pathways.txt`; - None for `levels.txt`; - None for `attributions.txt`. Fields in tables not defined above should not be translated. However producers sometimes add extra fields that are outside the official specification and these unofficial fields may be translated. Below is the recommended way to use `record_sub_id` for those tables: - None for `calendar.txt`; - `date` for `calendar_dates.txt`; - None for `fare_attributes.txt`; - `route_id` for `fare_rules.txt`; - None for `shapes.txt`; - `start_time` for `frequencies.txt`; - `to_stop_id` for `transfers.txt`. Conditionally Required: - **Forbidden** if `table_name` is `feed_info`. - **Forbidden** if `field_value` is defined. - **Required** if `table_name=stop_times` and `record_id` is defined.
`field_value`	Text or URL or Email or Phone number	**Conditionally Required**	Instead of defining which record should be translated by using `record_id` and `record_sub_id`, this field can be used to define the value which should be translated. When used, the translation will be applied when the fields identified by `table_name` and `field_name` contains the exact same value defined in field_value. The field must have **exactly** the value defined in `field_value`. If only a subset of the value matches `field_value`, the translation won't be applied. If two translation rules match the same record (one with `field_value`, and the other one with `record_id`), the rule with `record_id` takes precedence.

Field Name	Type	Presence	Description
			Conditionally Required: - **Forbidden** if `table_name` is `feed_info`. - **Forbidden** if `record_id` is defined. - **Required** if `record_id` is empty.

feed_info.txt

File: **Recommended** (**Required** if `translations.txt` is provided)

Primary key (none)

The file contains information about the dataset itself, rather than the services that the dataset describes. In some cases, the publisher of the dataset is a different entity than any of the agencies.

If both referencing methods (`record_id`, `record_sub_id`) and `field_value` are used to translate the same value in 2 different rows, the translation provided with (`record_id`, `record_sub_id`) takes precedence.

Field Name	Type	Presence	Description
`feed_publisher_name`	Text	**Required**	Full name of the organization that publishes the dataset. This may be the same as one of the `agency.agency_name` values.
`feed_publisher_url`	URL	**Required**	URL of the dataset publishing organization's website. This may be the same as one of the `agency.agency_url` values.
`feed_lang`	Language code	**Required**	Default language used for the text in this dataset. This setting helps GTFS consumers choose capitalization rules and other language-specific settings for the dataset. The file `translations.txt` can be used if the text needs to be translated into languages other than the default one. The default language may be multilingual for datasets with the original text in multiple languages. In such cases, the `feed_lang` field should contain the language code `mul` defined by the norm ISO 639-2, and a translation for each language used in the dataset should be provided in `translations.txt`. If all the original text in the dataset is in the same language, then `mul` should not be used. --- *Example: Consider a dataset from a multilingual country like Switzerland, with the original `stops.stop_name` field populated with stop names in different languages. Each stop name is written according to the dominant language in that stop's geographic location, e.g. `Genève` for the French-speaking city of Geneva, `Zürich` for the German-speaking city of Zurich, and `Biel/Bienne` for the bilingual city of Biel/Bienne. The dataset `feed_lang` should be `mul` and translations would be provided in*

Field Name	Type	Presence	Description
			translations.txt, in German: Genf, Zürich and Biel; in French: Genève, Zurich and Bienne; in Italian: Ginevra, Zurigo and Bienna; and in English: Geneva, Zurich and Biel/Bienne.
default_lang	Language code	Optional	Defines the language that should be used when the data consumer doesn't know the language of the rider. It will often be en (English).
feed_start_date	Date	Recommended	The dataset provides complete and reliable schedule information for service in the period from the beginning of the feed_start_date day to the end of the feed_end_date day. Both days may be left empty if unavailable. The feed_end_date date must not precede the feed_start_date date if both are given. It is recommended that dataset providers give schedule data outside this period to advise of likely future service, but dataset consumers should treat it mindful of its non-authoritative status. If feed_start_date or feed_end_date extend beyond the active calendar dates defined in calendar.txt and calendar_dates.txt, the dataset is making an explicit assertion that there is no service for dates within the feed_start_date or feed_end_date range but not included in the active calendar dates.
feed_end_date	Date	Recommended	(see above)
feed_version	Text	Recommended	String that indicates the current version of their GTFS dataset. GTFS-consuming applications can display this value to help dataset publishers determine whether the latest dataset has been incorporated.
feed_contact_email	Email	Optional	Email address for communication regarding the GTFS dataset and data publishing practices. feed_contact_email is a technical contact for GTFS-consuming applications. Provide customer service contact information through agency.txt. It's recommended that at least one of feed_contact_email or feed_contact_url are provided.
feed_contact_url	URL	Optional	URL for contact information, a web-form, support desk, or other tools for communication regarding the GTFS dataset and data publishing practices. feed_contact_url is a technical contact for GTFS-consuming applications. Provide customer service contact information through agency.txt. It's recommended that at least one of feed_contact_url or feed_contact_email are provided.

attributions.txt

File: **Optional**

Primary key (`attribution_id`)

The file defines the attributions applied to the dataset.

Field Name	Type	Presence	Description
`attribution_id`	Unique ID	Optional	Identifies an attribution for the dataset or a subset of it. This is mostly useful for translations.
`agency_id`	Foreign ID referencing `agency.agency_id`	Optional	Agency to which the attribution applies. If one `agency_id`, `route_id`, or `trip_id` attribution is defined, the other ones must be empty. If none of them is specified, the attribution will apply to the whole dataset.
`route_id`	Foreign ID referencing `routes.route_id`	Optional	Functions in the same way as `agency_id` except the attribution applies to a route. Multiple attributions may apply to the same route.
`trip_id`	Foreign ID referencing `trips.trip_id`	Optional	Functions in the same way as `agency_id` except the attribution applies to a trip. Multiple attributions may apply to the same trip.
`organization_name`	Text	**Required**	Name of the organization that the dataset is attributed to.
`is_producer`	Enum	Optional	The role of the organization is producer. Valid options are: `0` or empty - Organization doesn't have this role. `1` - Organization does have this role. At least one of the fields `is_producer`, `is_operator`, or `is_authority` should be set at `1`.
`is_operator`	Enum	Optional	Functions in the same way as `is_producer` except the role of the organization is operator.
`is_authority`	Enum	Optional	Functions in the same way as `is_producer` except the role of the organization is authority.
`attribution_url`	URL	Optional	URL of the organization.
`attribution_email`	Email	Optional	Email of the organization.
`attribution_phone`	Phone number	Optional	Phone number of the organization.