Managing the product and availability data - ingestion model

Endpoint	Ingestion model	Real-time model	Additional notes
/products/modified-since	hourly		Used for ingestion. Frequency of updates: no longer than hourly (recommended every 15-20 min). Correct method: cursor parameter instead of the modified-since parameter to make sure that no products are skipped. If used, the partner must not call the /products/{product-code} endpoint in real-time but use only the ingested data.
/products/bulk			Not included in the workflows. May be used only for some edge cases when it's necessary to get details of a few products at the time. This is stated in our API documentation (here).
/products/{product-code}			Not used.
/availability/schedules/modified-since	hourly		Used for ingestion. Frequency of updates: no longer than hourly

			(recommended every 15-20 min). Correct method: cursor parameter instead of the modified-since parameter to make sure that no products are skipped. If used, the partner must not call the /availability/schedules/{product-code} endpoint in real-time but use only the ingested data.
/availability/schedules/bulk			Not included in the workflows. May be used only for some edge cases when it's necessary to get details of a few products at the time. This is stated in our API documentation (here).
/availability/schedules/{product-code}			Not used.
/products/search			Ingested data from the /products/modified-since endpoint should be used to create the search functionality and pull the products displayed on search pages. However if the search functionality hasn't been implemented based on the ingested data, it is ok to use the /products/search endpoint - the following rules apply: - This endpoint must not be used for ingestion, only to make real-time calls when the customer

			is searching for products from a specific destination. ● Results can be cached for maximum 1 hour. ● Pagination rules apply to these endpoints: No more than 50 products can be requested in a single call ("count": 50). The first request should be done with "start": 1, the second request must include the value of the start parameter for the next result you wish to see. For example, if the first request was done with "start": 1 and "count": 50, the second request will be done with "start": 51 and "count": 50, the third one with "start": 101 and "count": 50 etc. ● Paginate through the search results (using the start and the count paremeters) only when the customer wants to move to the next page with search results to see more products. The first request should be used to retrieve and display maximum 50 products and if the customer wants to move to the next page with search results, another request to the search endpoint should be done to request additional

			products. Partners shouldn't pull automatically all products from each destination when the user initiates the search, this would be an incorrect usage of the endpoint and would result in long load times.
/search/freetext			Ingested data from the /products/modified-since endpoint should be used to create the search functionality and pull the products displayed on search pages. However if the search functionality hasn't been implemented based on the ingested data, it is ok to use the /search/freetext endpoint - the following rules apply: ● This endpoint must not be used for ingestion, only to make real-time calls. ● Results can be cached for maximum 1 hour. ● Pagination rules apply to these endpoints: No more than 50 products can be requested in a single call ("count": 50). The first request should be done with "start": 1, the second request must include the value of the start parameter for the next result you wish to see. For example, if the

			first request was done with "start": 1 and "count": 50, the second request will be done with "start": 51 and "count": 50, the third one with "start": 101 and "count": 50 etc. ● Paginate through the search results (using the start and the count paremeters) only when the customer wants to move to the next page with search results to see more products. The first request should be used to retrieve and display maximum 50 products and if the customer wants to move to the next page with search results, another request to the search endpoint should be done to request additional products. Partners shouldn't pull automatically all products from each destination when the user initiates the search, this would be an incorrect usage of the endpoint and would result in long load times.

Managing the product and availability data - real-time search model

Endpoint	Ingestion model	Real-time model	Additional notes
/products/modified-since			Not used.
/products/bulk			Not included in the workflows. May be used only for some edge cases when it's necessary to get details of a few products at the time. This is stated in our API documentation (here).
/products/{product-code}		x	Not used to ingest all products but only in real-time for a single product selected from search. Must not be used if the partner is already ingesting with /products/modified.since. Results can be cached for 1 hour.
/availability/schedules/modified-since			Not used.
/availability/schedules/bulk			Not included in the workflows. May be used only for some edge cases when it's necessary to get details of a few products at the time. This is stated in our API documentation (here).
/availability/schedules/{product-code}		x	Not used to ingest availability for all products but only in real-time for a single product selected from search.

			Must not be used if the partner is already ingesting with /products/modified.since. Results can be cached for 1 hour.
/products/search		x	Must not be used for ingestion, only to make real-time calls when the customer is searching for products from a specific destination. Results can be cached for maximum 1 hour. ● Pagination rules apply to these endpoints: No more than 50 products can be requested in a single call ("count": 50). The first request should be done with "start": 1, the second request must include the value of the start parameter for the next result you wish to see. For example, if the first request was done with "start": 1 and "count": 50, the second request will be done with "start": 51 and "count": 50, the third one with "start": 101 and "count": 50 etc. ● Paginate through the search results (using the start and the count paremeters) only when the customer wants to move to the next page

			with search results to see more products. The first request should be used to retrieve and display maximum 50 products and if the customer wants to move to the next page with search results, another request to the search endpoint should be done to request additional products. Partners shouldn't pull automatically all products from each destination when the user initiates the search, this would be an incorrect usage of the endpoint and would result in long load times.
/search/freetext		x	Must not be used for ingestion, only to make real-time calls. Results can be cached for maximum 1 hour. ● Pagination rules apply to these endpoints: No more than 50 products can be requested in a single call ("count": 50). The first request should be done with "start": 1, the second request must include the value of the start parameter for the next result you wish to see. For example, if the first request was done with "start": 1 and

			"count": 50, the second request will be done with "start": 51 and "count": 50, the third one with "start": 101 and "count": 50 etc. ● Paginate through the search results (using the start and the count paremeters) only when the customer wants to move to the next page with search results to see more products. The first request should be used to retrieve and display maximum 50 products and if the customer wants to move to the next page with search results, another request to the search endpoint should be done to request additional products. Partners shouldn't pull automatically all products from each destination when the user initiates the search, this would be an incorrect usage of the endpoint and would result in long load times.

Other endpoints - applicable to both ingestion and real-time search models

Endpoint	Ingestion model	Real-time model	Additional notes
/exchange-rates	daily		Exchange rates must be refreshed based on the expiry

			timestamp (at the moment it's once a day), not multiple times a day and not in real-time.
/products/tags	weekly		Must be cached and refreshed weekly.
/reviews/product	weekly		Reviews must not be indexed as it's Viator Unique Content. Review data must be cached, not retrieved in real-time as there is a lot of data. The recommended frequency of updates for all reviews is weekly, could be done also monthly as this data doesn't change frequently (for example daily would be too frequent). In addition to a full refresh of all reviews, partners should refresh the reviews for a particular product when they identify a different review count returned in the product content response than saved on their end previously. The provider of the reviews (Viator/Tripadvisor) must be indicated, even when only the review rating is displayed without review text.
/suppliers/search/product-codes	weekly		Must be cached and refreshed weekly.
/v1/taxonomy/destinations	weekly		Must be cached and refreshed weekly.

/v1/taxonomy/attractions	weekly		Must be cached and refreshed weekly.
/products/booking-questions	monthly		Must be cached and refreshed monthly.
/locations/bulk	monthly		All locations must be cached and refreshed monthly (weekly would be too often) in addition to on-demand calls to this endpoint for any new locations found in the product content response.
/v1/product/photos	monthly		We recommend using the /reviews/product endpoint instead for traveller photos. If this endpoint is used to get user photos, this data must be cached (monthly refresh should be enough)
/availability/check		x	This endpoint must be called in real-time in the booking flow when the customer selects a date and passenger mix, it must not be used to ingest availability or pricing data. In case of pricing differences between previously quoted price and the new price from the /availability/check response, the new price must be applied to the booking (this shouldn't trigger the booking flow to be canceled, instead the new price should be communicated). In case the booking hold is not supported, it's essential to call the /availability/check endpoint again in the booking flow, right before submitting the booking

			request, to double-check the price and availability (even if this check was done already at an earlier stage). Pricing may change at any time, for example while the customer is providing all details for the booking, that's why it must be verified again before booking.
/bookings/hold		x	This endpoint can be used in combination with the /bookings/book endpoint, not the /bookings/cart/book endpoint. This endpoint shouldn't be called immediately prior to booking to check availability - as for this purpose the /availability/check endpoint should be used and not all products support availability hold; it should be called when the customer is moving to checkout or about to provide payment details, to hold availability and pricing for a few minutes. Timestamps returned for both availability and pricing hold should be verified and a new hold should be done in case the first hold expired and the booking hasn’t been made yet (3 times should be enough). In case the partner is not using the /bookings/hold endpoint, they should make another real-time check with the

			/availability/check endpoint right before booking (calling the /bookings/book endpoint) as there is a risk of a price change during the booking flow. Only the following combination of endpoints is allowed: /bookings/hold + /bookings/book or /bookings/cart/hold + /bookings/cart/book.
/bookings/book		x	This endpoint can be used in combination with the /bookings/hold endpoint, not the /bookings/cart/hold endpoint. The booking status from the /bookings/book response must be verified before confirming the booking to the customer. Viator voucher from the response to the booking endpoint must be shared with customers (custom vouchers are not allowed). Only the following combination of endpoints is allowed: /bookings/hold + /bookings/book or /bookings/cart/hold + /bookings/cart/book.
/bookings/cart/hold		x	This endpoint can be used in combination with the /bookings/cart/book endpoint,

			not the /bookings/book endpoint. This endpoint shouldn't be called immediately prior to booking to check availability - as for this purpose the /availability/check endpoint should be used and not all products support availability hold; it should be called when the customer is moving to checkout or about to provide payment details, to hold availability and pricing for a few minutes. Timestamps returned for both availability and pricing hold should be verified and a new hold should be done in case the first hold expired and the booking hasn’t been made yet (3 times should be enough). In case the partner is not using the /bookings/hold endpoint, they should make another real-time check with the /availability/check endpoint right before booking (calling the /bookings/book endpoint) as there is a risk of a price change during the booking flow. Only the following combination of endpoints is allowed: /bookings/hold + /bookings/book or /bookings/cart/hold + /bookings/cart/book.

/bookings/cart/book		x	This endpoint can be used in combination with the /bookings/cart/hold endpoint, not the /bookings/hold endpoint. The booking status from the /bookings/book response must be verified before confirming the booking to the customer. Viator voucher from the response to the booking endpoint must be shared with customers (custom vouchers are not allowed). Only the following combination of endpoints is allowed: /bookings/hold + /bookings/book or /bookings/cart/hold + /bookings/cart/book.
/bookings/status		x	This endpoint should be used for periodic checks of the booking status when the /bookings/book response returns a pending status (manual confirmation type products) - hourly checks should be enough, every few minutes would be too often. This endpoint shouldn't be used to verify the booking status of all bookings. The booking endpoint returns the booking status. Confirmed bookings could be cancelled by the supplier later and supplier cancellations should be pulled with the /bookings/modified-since endpoint (the process to

			automate supplier cancellations is described in this article). When it comes to supplier-initiated amendments, they are not supported in the API and will be communicated via email (not to the customer but to the merchant partner using the email address provided in the partner dashboard for Supplier communications). This endpoint must be called also whenever the booking endpoint returns an error or the response times out, or for any other reason it's not clear if the booking has been confirmed (it must be called prior to re-booking).
/bookings/modified-since		Every 2-5 min	This endpoint is used to automate the process for supplier cancellations (the process to automate supplier cancellations is described in this article). When this endpoint is not used, the partner must be able to provide 24h customer support to inform travelers about canceled bookings as soon as possible. In order to stop the cancellation emails, this endpoint must be called no longer than every 5 minutes and it's necessary to verify the acknowledgeBy timestamp from the response - it indicates the time by when the

			acknowledgement for this cancellation notification must be sent using the /bookings/modified-since/acknowledge endpoint. If not done within the time specified, an email about the cancellation will be sent to the email address provided in the partner dashboard for Supplier communications. This endpoint shouldn't be called in real-time but every 2-5 min.
/bookings/modified-since/acknowledge		Every 2-5 min (based on timestamps from the /bookings/modified-since/acknowledge)	It is necessary to acknowledge that the cancellation notification has been received within the required time - indicated in the acknowledgeBy timestamp returned in the /bookings/modified-since response (5 min).
/bookings/cancel-reasons	monthly		Must be cached and refreshed monthly.
/bookings/{booking-reference}/cancel-quote		x	This endpoint must be called in real-time before canceling the booking to check the refund amount and communicate it to the customer. It's not enough to display the cancellation policy wording as it doesn't guarantee that travelers will understand the amount that will be refunded, i.e. they may not calculate correctly the time left to the start time based on the supplier's time zone. That's why it's important to double-check the

			refund amount using this endpoint to ensure 100% accuracy.
/bookings/{booking-reference}/cancel		x	All bookings must be canceled via the API using this endpoint (not by contacting Viator support)
Affiliate API only			
/v1/search/attractions	weekly/ monthly		Must be cached and refreshed weekly or monthly.
/v1/attraction	weekly/ monthly		Must be cached and refreshed weekly or monthly.
/v1/attraction/products	weekly/ monthly		Must be cached and refreshed weekly or monthly.
/v1/support/customercare	weekly/ monthly		Must be cached and refreshed weekly or monthly.