Vertex AI Search for retail is being renamed to Vertex AI Search for commerce. We're updating content to reflect the new branding.

About catalogs and products

This page provides best practices for creating your catalog information and populating your catalog data.

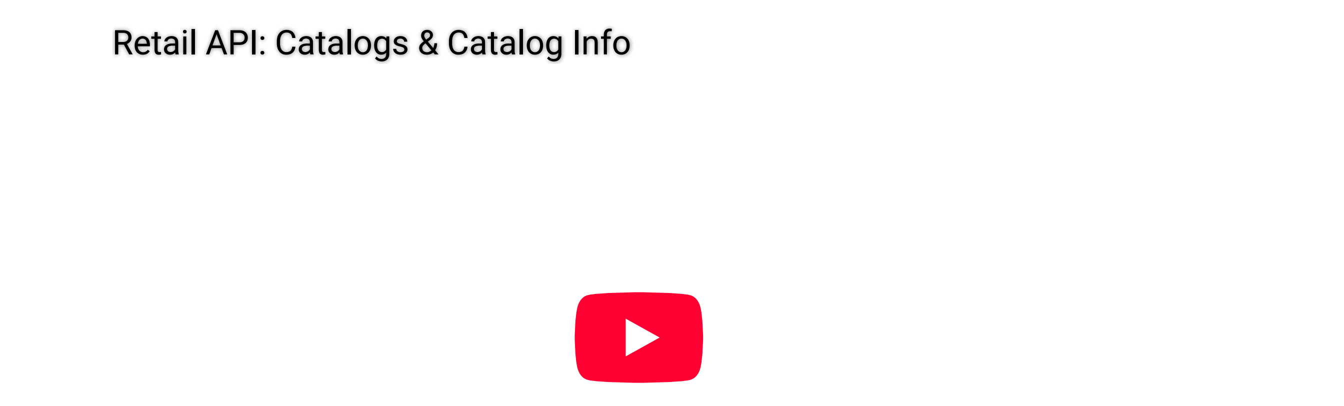
Overview

The catalog is a collection of product objects.

The catalog data you import into Vertex AI Search for commerce has a direct effect on the quality of the resulting model, and therefore on the quality of search and recommendation results. In general, the more accurate and specific catalog information you can provide, the higher quality your model.

Your catalog should be kept up to date. You can upload catalog changes as often as needed; ideally, every day for catalogs with a high rate of change. You can upload (patch) existing product items; only the changed fields will be updated. There is no charge for uploading catalog information. For more information, see Keeping your catalog up to date (/retail/docs/upload-catalog#current).

Note: There is only one catalog per Google Cloud project. It is named `default_catalog`. You don't need to create the default catalog; it is there by default.



Catalog branches

Warning: If you use recommendations, do not use multiple catalog branches. Branch switching can negatively impact prediction results.

Catalog branches with search

If you use search, you can use catalog branches to test new data that you've uploaded offline before making it live on your site.

You can use up to three branches, identified as `0`, `1`, and `2`. Your live site points to `default_branch` for its catalog data. Specify which branch is your live `default_branch` (the default is set to branch `0`) using either `setDefaultBranch` or the Data tab in Search for commerce console. Your site then uses the catalog data provided by the branch that `default_branch` points to.

As an example, say `default_branch` is set to branch ID `0`, so your site is using the catalog data that you've uploaded to that branch. You can upload new catalog data to branch `1` and preview it. After you've confirmed that the catalog has been uploaded correctly, you can switch to branch `1` as the live `default_branch`.

The catalog cache can take up to 30 minutes to update after branch switching.

If you use recommendations, we recommend using only the default branch due to the update delay during branch switching. If the data difference between branches is large, update delay can negatively impact prediction results.

Required product information

The following fields are required; you must provide values for them when you create product items in your catalog. They should also correspond with the values used in your internal product database, and should accurately reflect the product represented, because they are included in training your models.

In some cases, other fields are also required. Refer to the complete list of all product fields on the `Product` (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products) reference page.

All product information you provide can be used to improve the quality of recommendations and search results. Be sure to provide as many fields as possible.

Field	Notes
`name`	The full, unique resource name of the product. Required for all `Product` methods except for `import`. During import, the name is automatically generated and does not need to be manually provided.
`id`	The product ID used by your product database. The ID field must be unique across your entire catalog. The same value is used when you record a user event, and is also returned by the `predict` and `search` methods.
`title`	Product title from your product database. A UTF-8 encoded string. Limited to 1250 characters.
`categories`	Product categories. Every product must be assigned to at least one category. If a product belongs to more than one category, repeat the field for each category. The value must be a non-empty UTF-8 encoded string with a length limit of 5,000 characters. Always specify the full category path, for example: `["Sports & Fitness > Athletic Clothing > Shoes"]`.

Product description enrichment

Vertex AI Search for commerce uses product URLs to enrich product descriptions. Product URLs enhance catalogs by:

- **Enriching product data**: Vertex AI Search for commerce extracts supplemental information by crawling the product's URI, the unique resource identifier behind the exact location of each product on the web (URL). This process helps in derive additional details and signals from the linked web pages. The deeper understanding of products gained through URI crawling directly contributes to the quality of the catalog's data.
- **Improving search quality and relevance**: Web signals gathered from the crawled URIs are used to enhance search quality. The backend utilizes the crawled information, including how the URI was interacted with on the web, such as when your user clicks backlinks, for relevance and popularity scoring in search results.
- **Providing a foundation for model training**: Besides product ID, title, category hierarchy, and price, URLs are considered one of the primary fields used as inputs for model training.

Note: To help ensure quality service and prevent degraded performance, it is strongly recommended to provide a valid URI for each product.

To maximize the benefits of product URLs, adhere to these best practices:

- The linked web page must be publicly accessible and load correctly, not located behind a login or authentication wall.
- Each URI should be unique and consistently point to the correct product's web page. Its content should accurately reflect the product information in the catalog. It is also recommended to maintain the same top-level domain name across all product URIs

Catalog categories

This section describes the catalog structure and how to designate it for use in taxonomy and filtering.

Catalog structure

The `categories` field in the catalog should contain the most granular category path for each product. Parent categories are not required and should not be included.

Here is an example of how to structure your catalog:

- **Correct**: `categories: ["Flowers, Cards, Occasion > Seasonal Items > Christmas"]`
- **Incorrect**: `categories: ["Flowers, Cards, Occasion", "Flowers, Cards, Occasion > Seasonal Items", "Flowers, Cards, Occasion > Seasonal Items > Christmas"]`

Parent categories

The parent categories for a given product should not be included in the `categories` field. Filtering for browsing should be handled using other custom attributes (/retail/docs/attribute-config).

Category naming

Category names should be chosen carefully to avoid introducing incorrect keywords and improve performance. Using more specific and accurate terms will enhance relevance and reduce issues.

- **Recommended**: *Frozen Food > Frozen Fruits*

- **Not recommended**: *Frozen Fruits & Vegetables > Frozen Fruits*

Project-specific, catalog-level structure

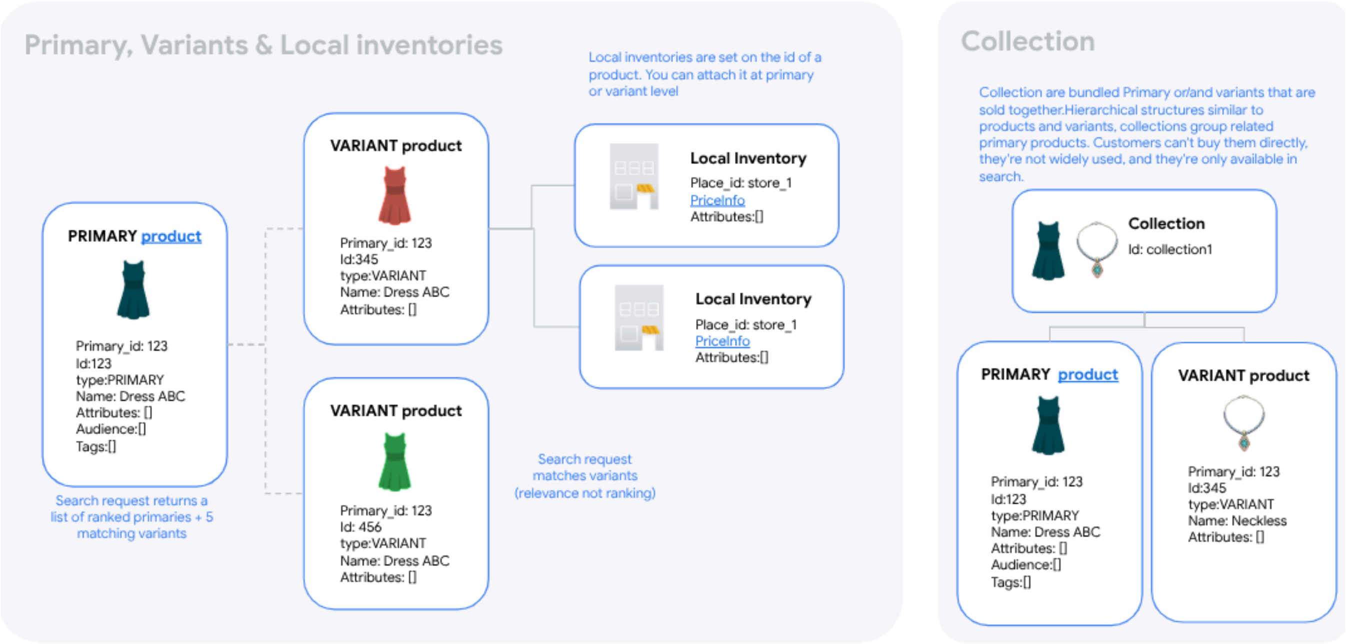
Create a single catalog for each language. If you operate in multiple countries, you can use the same catalog to provide search results in different countries.

Provide prices, which are determined through local inventories, in the same currency across countries. If prices differ among countries, create a local inventory for each country. Specify its prices there.

To optimize your search results, provide each country name as the `SearchRequest.entity` and `UserEvent.entity`. Use country entities for ranking purposes only.

Product hierarchy overview

When managing your product catalog in Vertex AI Search for commerce, understanding how attributes are handled for primary and variant products is crucial for effective search and recommendations. Product SKU designations determine catalog hierarchy.



Local inventories are set on the id of a product. You can attach it at primary or variant level

Search request returns a list of ranked primaries + 5 matching variants

Search request matches variants (relevance not ranking)

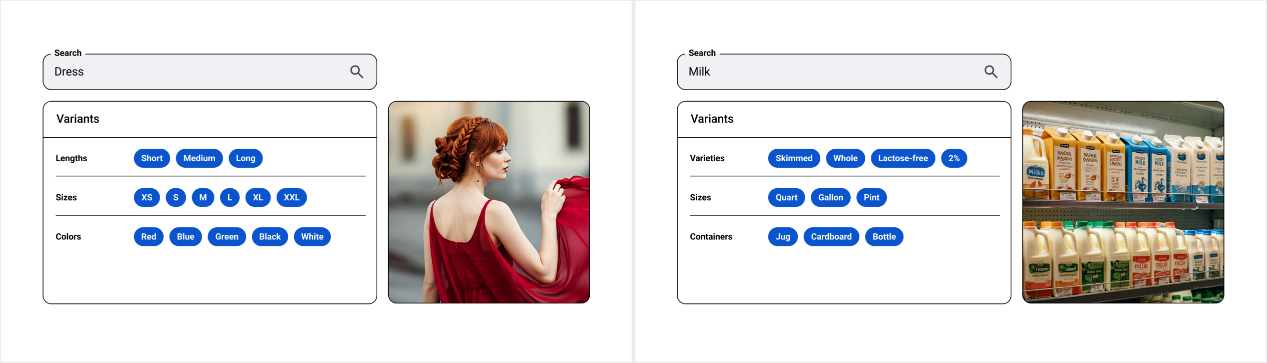
Collection are bundled Primary or/and variants that are sold together.Hierarchical structures similar to products and variants, collections group related primary products. Customers can't buy them directly, they're not widely used, and they're only available in search.

Product designation types

There are three product designation types:

1. **Primary** or parent items are returned in recommendation or search results and serve as logical containers or groups of similar items. Primaries can be individual (SKU-level) items and groups of similar items (SKU groups).
2. **Variant** or child items are specific, individual versions of a SKU-group primary product. For example, if the primary product is *V-neck shirt,* variants could be *Brown V-neck shirt, size XL* and *White V-neck shirt, size S*.
3. **Collection** items are bundles of primary products or variant products, such as a jewelry set with a necklace, earrings, and a ring. Hierarchical structures similar to products and variants, collections group related primary products. Customers can't buy them directly, they're not widely used, and they're only available in search.

Note: The system strictly enforces primary to variant taxonomic relationships, and incorrect mapping will cause import failures. This product `type` is immutable, so changing it requires deleting and recreating the product.



Product examples

As an example, according to these product designation types, grocery items are better cataloged as primary products, each consisting of a single SKU product, such as"`bananas, fresh`".

On the other hand, tee-shirts would be better structured hierarchically, as primaries with their corresponding set of variants. Each variant represents an individual SKU (for each size) and each primary item represents a group of SKUs, where each SKU is a different size for one overarching tee-shirt style. This organization by SKU structure allows the search results and recommendation panels to show a range of tee-shirt styles. It allows the shopper to drill down on a particular primary (style) to select the variant (size) to purchase.

Collections group related products that a customer might buy. To accurately represent them in the reranking model, Vertex AI Search for commerce has a logic that credits them with purchases. For example: A shopper clicks on products in a bedsheets set, then adds to cart or purchases a primary product in that collection. The collection is credited with that purchase, and the model accurately represents the popularity and value of collections.

There is also a variant-only catalog type, which is now deprecated. This catalog type can only be used with recommendations. For the variant-only catalog, the `ingestionProductType` is set to `variant` during import. A primary is inferred for each variant, based on a primary product ID specified for each variant.

Set up your product catalog

When planning your product catalog, you need to decide whether it contains products designated as only primaries, primaries and variants, or a mixture of the two arrangements. Think of it in terms of your products' SKU structure. Your products can be primary items, which may or may not have variants.

Based on how your product SKUs are designated, consider your options for setting up your product catalog:

- **You want your SKU to be shown as an individual search result or recommendation**: SKU=primary
- **Your SKU should be part of a group of similar SKUs**: SKU=variant, group of SKUs=primary
- **A mixture of both combinations**: SKU=primary, SKU=variant, group of SKUs=primary

If your product detail page shows an option, size, or color selector, these options are typically uploaded as variants into your product catalog. Consider whether or not you want different types of the same product with different attributes such as size and color to appear as a single search result or separate ones. For example, for a book you want to decide if you want a hard cover SKU and a soft cover SKU of the same book to appear as separate search results (SKU = primary), or as one (SKU = variant, group of SKUs = primary).

When setting up your product catalog, keep in mind that recommendation and search results only return primary items.

Minimal primary products

If you determine that your catalog should have both primaries and variants, that is, SKU groups and SKUs, but you only have SKUs now, you need to create primaries for the SKU groups. These primaries are sometimes called virtual primaries or fake primaries.

These primaries only need to contain minimal information: `id`, `title`, and `categories`.

If `type` is not specified, the product type defaults to primary. If you are importing, you don't need to specify `name`. For more information, see the preceding section, Required product information (#required-fields).

Catalog import

If your catalog is in Cloud Storage or BigQuery or some other storage, then do a bulk data import.

For detailed information about how to upload a catalog, see Import catalog information (/retail/docs/upload-catalog).

Product inventory

Product inventory encompasses:

- Price, both the current and original prices
- Availability, such as in stock, out-of-stock, back ordered, and pre-ordered
- Quantity available
- Fulfillment information such as pickup-in-store, ship-to-store, and next-day-delivery

There are two levels of inventory: product-level and local.

Product-level inventory

For retailers who only sell online, inventory is specified at the product level. Price, availability, and other inventory data is set for each product in the catalog.

For more information about product-level inventory, including how to maintain inventory data, see Update inventory for Vertex AI Search for commerce (/retail/docs/inventory-updates).

Local inventory

Retailers who have brick-and-mortar stores and an online store need to keep inventory information on a per-store basis. They use local inventory to do this.

There are two product fields that can be used to store local inventory. Both fields are lists of locations (place IDs) with associated inventory information:

- **Product.fulfillmentInfo** (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.FIELDS.fulfillment_info). Pickup and shipping methods at each store location
- **Product.localInventories** (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.FIELDS.local_inventories). Price information, product attributes, and pickup and shipping methods at each store location

You can use either or both fields for your store-level information.

For more information about local inventories, see Update local inventory for Vertex AI Search for commerce (/retail/docs/local-inventory-updates).

Primary-variant-inventory structure

The primary-variant-inventory data structure is composed of primary, variant, and local inventory products:

- **Primary products**: Primary product data is stored without prices.
- **Variant products (with the lowest price across the country)**: For example, the variant for the primary (color, size) should have the lowest price across country. Variant price data is rolled up to the primary product data, the primary price in turn being used for ranking purposes. Location-specific prices are ignored.
- **Local inventory (region or store-specific pricing)**: Use the price information from the local inventory in reranking at serving time
- **Primary product attributes**: Primary products must only include attributes common to all their associated variants.

Catalog data quality metrics

The **Data quality** (https://console.cloud.google.com/ai/retail/catalogs/default_catalog/data/) page in the Search for commerce console assesses if you need to update catalog data to improve the quality of search results and unlock search performance tiers.

The following table describes the quality metrics that Vertex AI Search for commerce uses to help you evaluate your product data. For details about how to view data quality metrics and search performance tiers in the Search for commerce console, see Unlock search performance tiers (/retail/docs/data-quality).

Catalog quality metric	Quality rule
URI is present and accessible (#product-urls)	Product has a valid `Product.uri` (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.FIELDS.uri). ⁊ be accessible and match your domain.
Meets time conformance	`Product.availableTime` (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.FIELDS.avail before current time, and `Product.expireTime` (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.FIELDS.expir current time.
Searchable attribute is present	Product has at least one `attribute` (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.FIELDS.attrib searchable.
Description is present	Product has non-empty `Product.description` (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.FIELDS.desc
Title consists of at least two words	`Product.title` (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Pro consists of at least two words.
Has variant with image	The `variant` (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.Type.ENUM_ product has at least one `Product.image` (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.FIELDS.imag ignore this metric if all your products are at `primary`

Catalog quality metric	Quality rule
	(/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.Type.ENUM_ level.
Has variant with price info	The **`variant`** (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.Type.ENUM_ product has **`Product.priceInfo`** (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.FIELDS.price may ignore this metric if all your products are at **`primary`** (/retail/docs/reference/rest/v2/projects.locations.catalogs.branches.products#Product.Type.ENUM_ level.

Product schema

When importing a catalog from BigQuery, use the following Vertex AI Search for commerce product schema to create a BigQuery table with the correct format and load it with your catalog data. Then, import the catalog (/retail/docs/upload-catalog#bq).

Vertex AI Search for commerce product schema

BigQuery (#bigquery) More

```
[
  {
    "name": "name",
    "type": "STRING",
    "mode": "NULLABLE"
  },
  {
    "name": "id",
    "type": "STRING",
    "mode": "REQUIRED"
  },
  {
    "name": "type",
    "type": "STRING",
    "mode": "NULLABLE"
  },
  {
    "name": "primaryProductId",
    "type": "STRING",
    "mode": "NULLABLE"
  },
  {
    "name": "collectionMemberIds",
    "type": "STRING",
    "mode": "REPEATED"
  },
  {
    "name": "gtin",
    "type": "STRING",
    "mode": "NULLABLE"
  },
  {
    "name": "categories",
    "type": "STRING",
    "mode": "REPEATED"
  },
  {
    "name": "title",
    "type": "STRING",
    "mode": "REQUIRED"
  },
  {
    "name": "brands",
    "type": "STRING",
    "mode": "REPEATED"
  },
  {
    "name": "description",
    "type": "STRING",
    "mode": "NULLABLE"
  },
  {
    "name": "languageCode",
    "type": "STRING",
    "mode": "NULLABLE"
  },
  {
    "name": "attributes",
    "type": "RECORD",
    "mode": "REPEATED",
    "fields": [
      {
        "name": "key",
        "type": "STRING",
        "mode": "NULLABLE"
      },
      {
        "name": "value",
        "type": "RECORD",
        "mode": "NULLABLE",
        "fields": [
          {
            "name": "text",
            "type": "STRING",
            "mode": "REPEATED"
          },
          {
            "name": "numbers",
            "type": "FLOAT",
            "mode": "REPEATED"
          }
        ]
      }
    ]
  },
  {
    "name": "tags",
    "type": "STRING",
    "mode": "REPEATED"
  },
  {
    "name": "priceInfo",
    "type": "RECORD",
    "mode": "NULLABLE",
    "fields": [
      {
        "name": "currencyCode",
        "type": "STRING",
        "mode": "NULLABLE"
      },
      {
        "name": "price",
        "type": "FLOAT",
        "mode": "NULLABLE"
      },
      {
        "name": "originalPrice",
        "type": "FLOAT",
        "mode": "NULLABLE"
      },
      {
        "name": "cost",
        "type": "FLOAT",
        "mode": "NULLABLE"
      },
      {
        "name": "priceEffectiveTime",
        "type": "STRING",
        "mode": "NULLABLE"
      },
      {
        "name": "priceExpireTime",
        "type": "STRING",
        "mode": "NULLABLE"
      }
    ]
  },
  {
    "name": "rating",
    "type": "RECORD",
    "mode": "NULLABLE",
    "fields": [
      {
        "name": "ratingCount",
        "type": "INTEGER",
        "mode": "NULLABLE"
      },
      {
        "name": "averageRating",
        "type": "FLOAT",
        "mode": "NULLABLE"
      },
      {
        "name": "ratingHistogram",
        "type": "INTEGER",
        "mode": "REPEATED"
      }
    ]
  },
  {
    "name": "expireTime",
    "type": "STRING",
    "mode": "NULLABLE"
  },
  {
    "name": "ttl",
    "type": "RECORD",
    "mode": "NULLABLE",
    "fields": [
      {
        "name": "seconds",
        "type": "INTEGER",
        "mode": "NULLABLE"
      },
      {
        "name": "nanos",
        "type": "INTEGER",
        "mode": "NULLABLE"
      }
    ]
  },
  {
    "name": "availableTime",
    "type": "STRING",
    "mode": "NULLABLE"
  },
  {
    "name": "availability",
    "type": "STRING",
    "mode": "NULLABLE"
  },
  {
    "name": "availableQuantity",
    "type": "INTEGER",
    "mode": "NULLABLE"
  },
  {
    "name": "fulfillmentInfo",
    "type": "RECORD",
    "mode": "REPEATED",
    "fields": [
      {
        "name": "type",
        "type": "STRING",
        "mode": "NULLABLE"
      },
      {
        "name": "placeIds",
        "type": "STRING",
        "mode": "REPEATED"
      }
    ]
  },
  {
    "name": "uri",
    "type": "STRING",
    "mode": "NULLABLE"
  },
  {
    "name": "images",
    "type": "RECORD",
    "mode": "REPEATED",
    "fields": [
      {
        "name": "uri",
        "type": "STRING",
        "mode": "REQUIRED"
      },
      {
        "name": "height",
        "type": "INTEGER",
        "mode": "NULLABLE"
      },
      {
        "name": "width",
        "type": "INTEGER",
        "mode": "NULLABLE"
      }
    ]
  },
  {
    "name": "audience",
    "type": "RECORD",
    "mode": "NULLABLE",
    "fields": [
      {
        "name": "genders",
        "type": "STRING",
        "mode": "REPEATED"
      },
      {
        "name": "ageGroups",
        "type": "STRING",
        "mode": "REPEATED"
      }
    ]
  },
  {
    "name": "colorInfo",
    "type": "RECORD",
    "mode": "NULLABLE",
    "fields": [
      {
        "name": "colorFamilies",
        "type": "STRING",
        "mode": "REPEATED"
      },
      {
        "name": "colors",
        "type": "STRING",
        "mode": "REPEATED"
      }
    ]
  },
  {
    "name": "sizes",
    "type": "STRING",
    "mode": "REPEATED"
  },
  {
    "name": "materials",
    "type": "STRING",
    "mode": "REPEATED"
  },
  {
    "name": "patterns",
    "type": "STRING",
    "mode": "REPEATED"
  },
  {
    "name": "conditions",
    "type": "STRING",
    "mode": "REPEATED"
  },
  {
    "name": "publishTime",
    "type": "STRING",
    "mode": "NULLABLE"
  },
  {
    "name": "promotions",
    "type": "RECORD",
    "mode": "REPEATED",
    "fields": [
      {
        "name": "promotionId",
        "type": "STRING",
        "mode": "NULLABLE"
      }
    ]
  }
]
```

Note: For reranking on inventory prices, use `placeId` in the search request.

Last updated 2025-08-25 UTC.