B. Analytics Studio

By**GauthamT**

Last Updated:**Aug 28, 2025**

Introduction

Enterprises that have undergone a digital transformation, or who are planning one, understand that their data is their most valuable asset. Most enterprises list bad customer experience as their reason for leaving a service behind. It has become critical for brands to harness, harvest, and analyze their customer experience data. For years, successful brands have been leveraging their data from their CRMs, websites, POS systems, surveys, and social media to better understand their customers. Unfortunately, many companies have overlooked the value of the authentic unified multichannel conversations between their customers, agents, and bots; the source of which could be various speech and text conversations like a phone (VoIP telephony), SMS texting, email, web chat, social media, video conferencing, etc. Due to the size and complexity of working with unstructured voice and text data, contact center interactions have proven difficult for brands to deeply analyze.

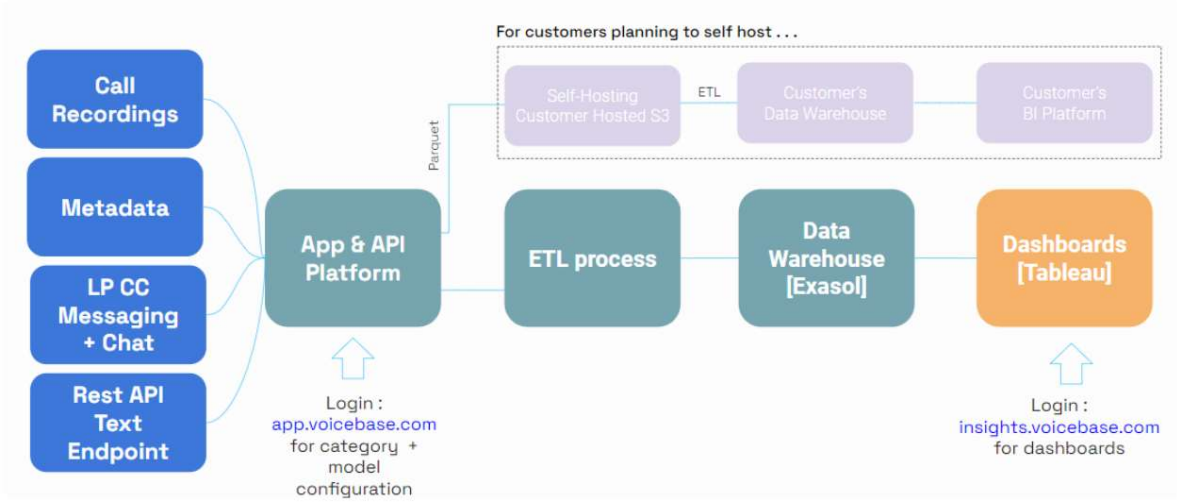
Enter Analytics Studio, by LivePerson, a leader in delivering intent-driven conversational experiences at scale, recently announced a groundbreaking new solution that enables data-driven enterprises to harness the power in their omnichannel call recordings and text conversations that until now have been hiding in plain sight. Analytics Studio is a tightly integrated, turn-key analytics platform that converts conversations into rich actionable data within minutes. The product captures conversations through LivePerson's Conversational Cloud where it can collate asynchronous messaging and synchronous chats, and also speech through LivePerson's VoiceBase platform which has a powerful speech-to-text and speech analytics API where calls are transcribed.

The conversations are measured with metrics like sentiment, silence, survey results, MCS, and NPS and tagged with customizable key events like conversation drivers, and competitor mentions. The data from each conversation is combined with the metadata of their customers so that each conversation can be tagged with meaningful information about the customer's, agent's, and bot's behavior impacting an interaction's outcome. Next, through a simple online configuration, this rich data is transformed and offered back to the customer in a relational data schema that customers can host in their own Snowflake, RedShift, Exasol, or other data management solutions.

This approach provides clients with complete control and ownership of their data. Analytics Studio pairs all of this data with a complete suite of detailed Tableau dashboards

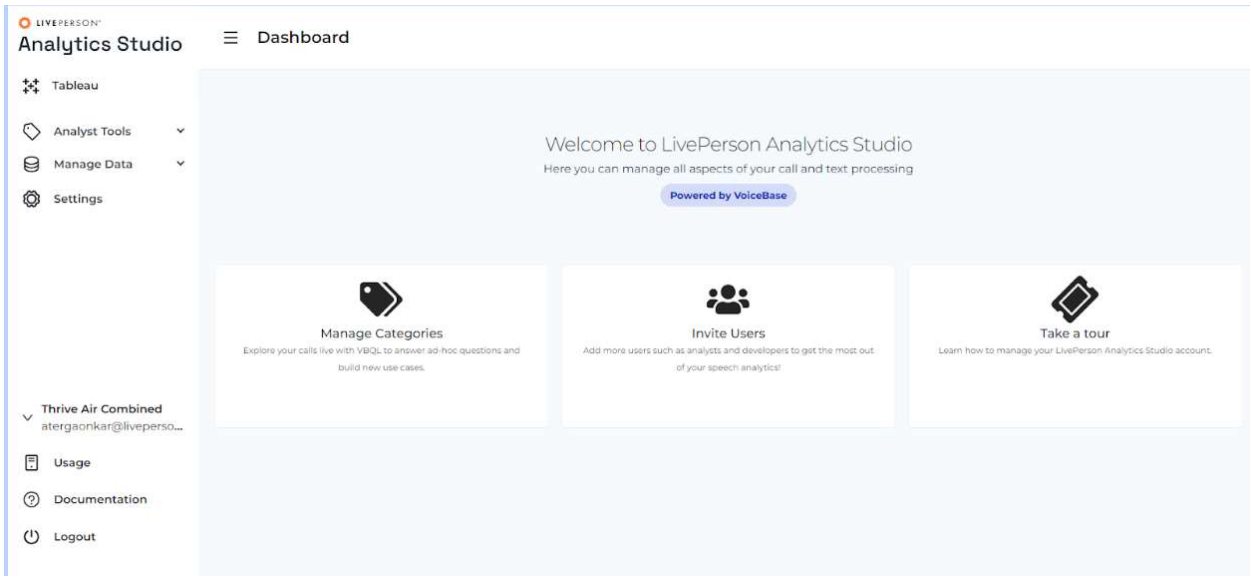
which can be customized to fit the individual needs of each customer.To increase speed to insight, Analytics Studio also offers a hosted version; because we understand that constraints on resources and priorities can often become bottlenecks to even the most promising projects. With this option, LivePerson does the hard work for you by managing the data warehouse, Tableau Server, and dashboard maintenance. This allows business users to get immediate answers from their data and begin reaping the ROI rewards.

Architecture



Getting Started with Analytics Studio

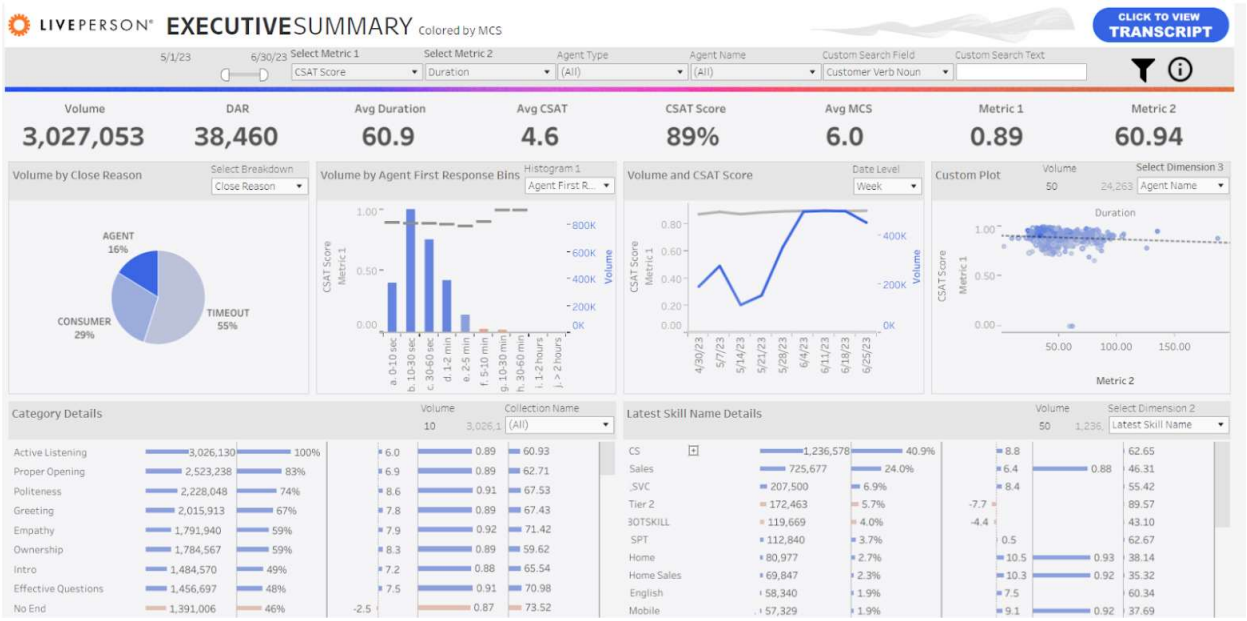
Once setup, **Analytics Studio** can be accessed from here.



The left-side navigation bar grants entry to different features. The availability of these features depends on assigned roles, and not all users will have access to every feature.We plan to integrate Analytics Studio into LivePerson Conversational Cloud, soon!

The Dashboards

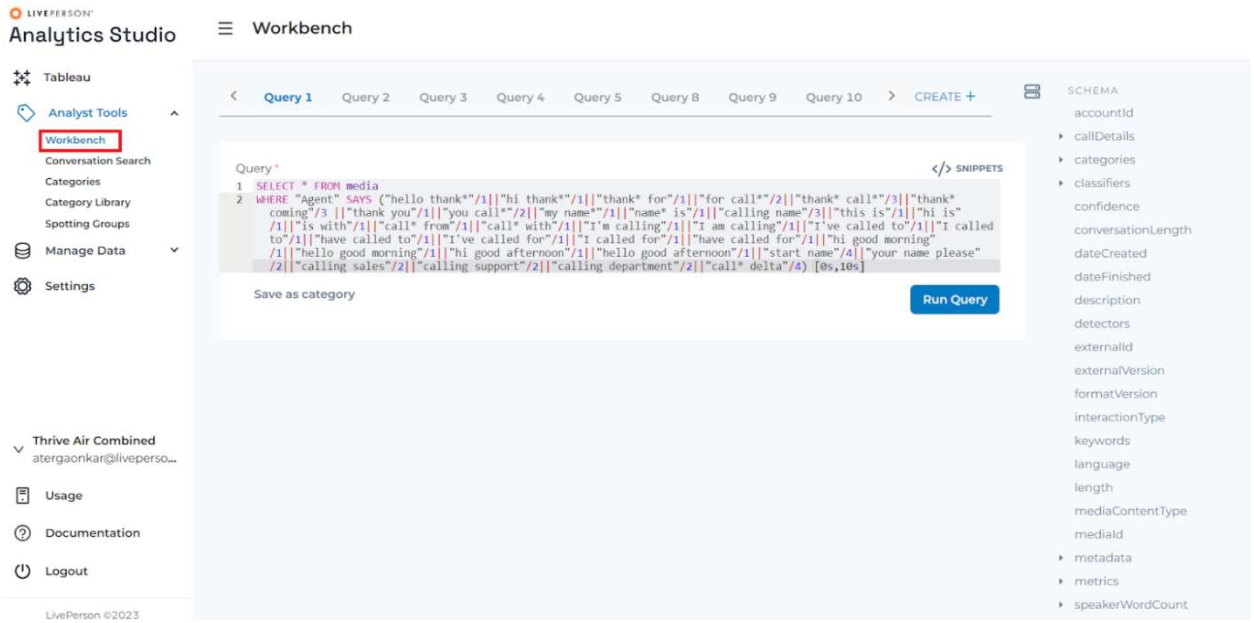
The topmost icon on the navigation bar provides users access to Tableau dashboards.



Users will get access to the following dashboards - Executive SummaryAgent ComparisonAgent ScorecardCategory ComparisonRepeat ContactsTransformation OverviewTrend ComparisonTrend SummaryDimension DrilldownDownloadable ReportCustom Report

Analytics Workbench

Analytics Workbench is an important part of Analytics Studio that helps you tag your conversations with a proprietary query language known as **VoiceBase Query Language (VBQL)**.



Analytics Studio enables you to apply speech analytics to conversations and categorize it according to business definitions. Each definition is called a category and it consists of a conditional statement that is applied to the transcription results and any corresponding metadata provided with the recording.

Analytics Studio runs the categories when the conversation is processed and returns a boolean result where a 1 equals a match for the category definition and a 0 when it does not match.

Categories are especially useful as a discovery tool when used in conjunction with the Analyst Workbench available in your Analytics Studio account. Categories are created using LivePerson's proprietary query language, VBQL.

Analytics Workbench is an important part of Analytics Studio that helps you tag your conversations with a proprietary query language known as **VoiceBase Query Language (VBQL)**. The tool allows you to analyze, tag (known as **Categories** in the product), and optimize insights around voice and messaging data. VBQL allows you to write powerful, flexible SQL-like queries to access and filter your data.

To access the Analytics Workbench, log in to Analytics Studio and go to **Analyst Tools > Workbench** on the left-hand menu.

Glossary of Key Concepts

Analytics Workbench

The Analytics Workbench is the name of the tool/UI that you will use to investigate your voice, messaging, or chat data, create Queries and Categories, and listen to or read transcripts of Media files using the Media Player.

Anchor Word

Anchor Words are keywords that a business identifies in the Discovery phase that 'anchor' or act as a starting point for building out robust and thorough Categories. Anchor Words usually relate to business key performance indicators, checkpoints, alerts, and concepts. Anchor Words give you a starting point to build from when building a catalog of Variant Words.

Bearer Tokens

Bearer Tokens are unique access tokens that provide access to your account, data libraries, and Content.

Media Player

The Media Player is a tool within the Analytics Workbench that allows you to inspect or listen to specific media or messaging files and investigate specific data points and insights about that file.

The Media Player is also available on the BI Dashboards and can be embedded using a simple script and Bearer Token.

Categories

The Queries could be saved as Categories and that can run on all new data put into your account.

The Categories help you tag your conversations and are useful to analyze and gather insights.

Fields

Fields are data points about your uploaded files. This can include the length of the file, transitions in the conversation, the word count of the conversation, the total amount of silence in the conversation, 'over talk' (did one person talk over the other), and more.

Media

Media includes all audio or messaging data and their associated metadata. When Media is uploaded to Analytics Studio, it is automatically uploaded into a default Media library named ‘media’. This library/name can then be accessed using VBQL to gain insights, and eventually build your Categories.

MediaId

MediaId (usually written ‘mediaId’) is the unique identifier assigned to all data coming into the system, whether speech, messaging, or chat. The Media ID is used to identify each piece of Media and the insights and metadata associated with the Media.

Metrics

Metrics are data points about the conversations in your uploaded Media. There are more than 50 Paralinguistic Metrics available including the sentiment of the participants in the conversation, silence and ‘over talk’ of the individual participants in voice data, the vocabulary of the participants, and more. Not all metrics are suitable for use with messaging, but some are appropriate and may be used.

Queries

Queries are used to retrieve data that fit search parameters. They are equivalent to searches that would be run in your favorite search engine. The key difference is that the VBQL language is specifically designed to search through Media data including audio, transcripts, and metadata.

Transcriptions

A Transcription is a written script of the voice or messaging data that you’ve uploaded, produced by VoiceBase’s AI-powered analytics engine and Natural Language Processing. The Queries and Categories that you will create will be searching the TranVariant Words Variant Words are alternate ways to describe your Anchor Words and concepts. These can be synonyms, antonyms, related concepts, expanded concepts, and/or more verbose concepts of your Anchor Words. Anchor Words and Variant Words will be the basis for most of your Categories. While building a catalog of Variant Words, you will also naturally get exposed to additional Anchor Words and concepts that can then be folded into existing and new Categories and Queries.

Analytics 101

Know your data

Conversation Analytics is most successful when based on a deep understanding of your data. Once your initial data has been uploaded, the audio, text, and metadata are

accessible through the VoiceBase API, and will automatically begin processing. The first layer of processing uses an AI-powered analytics engine and Natural Language Processing (NLP) to provide Transcriptions, Redactions, and Metrics. The second layer of processing is the Knowledge Extraction layer. This is where all new data added to your secure data bucket going forward is processed with the Categories that you'll create (more on Categories later).
Before you start building Categories, take the time to listen to a few calls and read the transcriptions or read through your conversations if they are messaging or chat media. Once you've seen for yourself how speech and transcription can differ, you can also start to understand the principle of Indicative Language. Indicative Language is a method of using the context around the keywords to understand what the participant was saying. By paying attention to the terms, tenses, and phrases surrounding the targeted keywords, you can better understand how to build a robust and comprehensive query, which will ultimately help you build better Categories.

Tip: If there are keywords like brands, names, and other unique words that tend to be mistranscribed you can add the words to the Keyword List (a.k.a. Custom Vocabulary List) to introduce the words to the Transcription engine.

Business Goals

Your Business Goals will determine how you want to analyze your voice and speech data. You will apply the nuances of your Business Goals by creating robust and fine-tuned Categories, specific to the analysis outcomes you need from your voice or messaging data. There are many different types of Categories that can be written in VBQL.
We break these items down into 3 types:

- Checkpoints
- Alerts
- Concepts

Some Categories act as mandatory **Checkpoints** you'd expect to happen in a conversation. For example, a greeting might be expected for all inbound calls or messages. That greeting should also be branded. These tasks are triggers to ensure something either does or does not, happen during a call. These items are often showcased in scorecards by agents, contact centers, or other focus.

In addition to checkpoints, VBQL also allows the classification of **Alerts**. Alerts are warnings to you about items like holds, transfers, and telecom or confusion issues. They act as flags to help understand why a checkpoint may have been missed or what

confounding factors may have influenced the call. Alerts can be quite handy to have access to in real-time and are often showcased in monitoring reports and performance-centric dashboards.

The final category type covers **Concepts**. Concepts encompass products, services, emotions, and other noteworthy events within speech or messaging analytics. Concepts themselves may have natural hierarchies and relationships, such as actions and products/services. Someone may call for a refund for a product or to cancel a service. The specific service ("voicemail") may be a part of a package of offerings ("call plan") with a set number of actions ("cancel" or "upgrade"). All quoted items are examples of concepts we'd want to see in relation to each other. Often, concepts may make up a series of analyses and dashboards to delve into the intricacies and relationships. Organization of concepts into natural buckets or hierarchies improves their ease of use.

Similarly, it's important to understand how the data returned from the Category will be used. There are many dashboards that can illustrate and track business goals. It's important to craft your Categories with an idea in mind of who or what will be utilizing the data. Tip: Keep the size of your Voice Data Library and how sensitive your Queries are in mind when deciding to opt for immediate alerts. Things that are statistically rare can happen more than you would expect with a large data stream. For this and other reasons, we recommend reports and dashboards over simple alerts.

Discovery

Ultimately, one of the main goals of setting up an effective Analytics system is creating your customized and robust Categories. The first phase of category development is searching for Anchor Words and Variant Words. Searching for these provides a window into the Indicative Language around what you are searching for. This data is important when making comprehensive Categories. The goal during this phase is to collect as much data about the Category requirements and discover new variants.

Explore and Iterate

Two of the keys to creating effective Queries and Categories are breaking ideas down into discrete pieces of logic and testing each piece as you make changes.
For example, when building a Query or Category around appointment setting we can break down the Query into several pieces: looking for words pertaining to appointments, setting a time, and verbiage indicating "going to" or "coming into" a location. By writing and testing them as individual Queries more of the Queries can be reused, testing becomes easier, and we can still use operators (AND, OR, &&, ||, etc.) to combine these Queries into more robust Categories that we can save to run automatically when new data comes in. Testing

as you make changes to Queries helps identify which Word Variants and phrases are most important to the Query and reviewing the results can lead to discovering even more ways the same concept can be expressed.

Tip: There are also many other things you can look at besides the words in the media: the amount of silence, intensity (indicated as CAPITALIZED words in the Transcription), timing, and other metadata, all of which are accessible in Queries and Results.

Building VBQL Queries and Categories

Queries Vs. Categories

Before we write our first Queries, it's important to understand the difference between Queries and Categories. Queries and Categories both use the VBQL and are ways to search for data within your voice or messaging Media library. The difference between the two rests on how and when the VBQL is executed. Queries are a manual way to search through your data. Every Query search is executed only when the **Run Query** button is pressed. Running a Query is very similar to how you would search in your favorite search engine. In contrast, Categories execute their VBQL every time new data enters your VoiceBase system.

The Simplest Query

We'll start with something simple. Copy and paste the following into the VBQL Editor:

```
SELECT * FROM media
```

Here's a breakdown of this query:

- SELECT is the command that sets what data is returned by the query. SELECT is the most common command you will use and indicates you want to go retrieve data for you from your database.
- * is a wildcard character that is used to designate that you'd like to select all columns/Fields for your data.
- FROM is the keyword to set which Media library is being queried
- media is the default library for ingested data.

Your results should appear at the bottom of the VBQL Editor after running this Query.

Searching for an Anchor Word

Let's modify the previous example to search for a specific word. Let's look for someone, anyone, saying the word "error." "error" is what we would call an Anchor Word. It isn't robust enough to save as a final Category, but it's a good start for Discovery and Category development.

SELECT * FROM media WHERE * SAYS "error"

WHERE is the command to set the search parameters you'd like to find in your data. When you use a WHERE clause/command, you first specify who is saying the keyword you're looking for, or, as we did in the example above, we use the wildcard '*' to indicate either participant could be saying our keyword. Then you use the SAYS clause and specify the keyword you're looking for, using quotes to indicate if it's a string as necessary.

Note: Depending on your familiarity with programming and/or query languages, you may or may not be familiar with the concept of a string. A string is a type of data that a computer recognizes as a string of characters (letters, numbers, and even symbols) designated as those characters wrapped in either single or double quotes. You'll note that after the WHERE command, we wrapped the word "error" in double quotes. Wrapping words in either single or double quotes indicates to the VBQL Query Editor that everything between the quotes is a string. Since the data you are trying to find, in this case, the keyword "error", is also considered a string data type in VoiceBase, you'll want to make sure you wrap your word in quotes in your query. Also note, the VBQL is not case sensitive, meaning searching for "error" searches for: "error", "Error", "eRROr", etc.

Only the String datatype requires the surrounding quotes.

Simplifying the Return

Using the wildcard character (*) is a powerful way to return as much data as possible, as quickly as possible. The main downside to the wildcard character is that it doesn't give you fine control over exactly what specific data you'd like to see. As mentioned in the 'Query Tabs – The Results Table' section, the Query results table displays columns/Fields based on which columns/Fields are SELECTed in the Query. VBQL Reference Guide for a full list and descriptions. We're going to be selecting only the mediaId in our next query. It is the ID for the specific Media item in the database. To return only the mediaId for each call in the return it would be added to the SELECT part of the Query. In this case we will replace the * wildcard with mediaId (which is case-sensitive):

SELECT mediaId FROM media WHERE * SAYS "error"

You'll note now that our results table now only includes the system default field/column for the Play button, as well as our only SELECTed column: mediaId.

Viewing Hits

Next, we will evolve our query by adding the 'hits()' field/column to our SELECT statement. One thing that will make viewing the results easier and more efficient will be adding text snippets of the hits in the voice data. We can do that by selecting the hits() data from the results. Like how we added mediaId, we're going to be adding hits() to the SELECT portion of our Query:

SELECT mediaId, hits() FROM media WHERE * SAYS "error"

Our results will now include a button that will take you to a JSON window showing what phrases were hit in the Query and when they occurred in each piece of audio.
Click on the '…' button for any row/item under the hits() column to expand the output.

Note: This data is formatted using a syntax called JSON (a.k.a. JavaScript Object Notation). You don't need to be a JSON expert in order to read the results, you just need to remember the basic structure of JSON. JSON formats our results by grouping the names of an attribute with its specific data point. These are commonly referred to as name/value pairs. Then, if a group of name/value pairs can be nested within a greater category, they have commonly organized that way (this can sometimes be called a parent and child relationship). For example, in our results, 'fieldValues' is the parent group, while 'startMs', 'endMs', 'startRatio', 'line', 'participant', and 'text' are all the child name/value pairs under this group/parent. You can learn more about JSON here.

The information we're going to be looking at right now is under the "fieldValues" section. The key name/value pairs to look at are: "startMs" and "endMs" are the start and end times of the hit in milliseconds respectively, "participant" indicates which line the audio came from and "text" is the Transcription of what triggered the hit. Note: The "participant" name/value pair is only useful in stereo audio where the speech data is split between the left and right audio channels. Names and channels are configurable.

Searching by Participant

For Media with stereo audio or in a messaging conversation, a Query may specify the name of the participant (Agent, Consumer, Bot, etc.):

SELECT mediaId, hits() FROM media WHERE "Consumer" SAYS "error"

Note: Agent, Consumer, and participant are the default keywords that can be redefined in the VoiceBase configuration.
Another option is to use an asterisk (*) as a wildcard, to include hits for any participant within the conversation:

SELECT mediaId, hits() FROM media WHERE * SAYS "error"

If the audio is rendered in mono, the phrase is simply attributed to "participant".

Searching by Metadata

Visible in the Schema Viewer, there is a list of available metadata values that can be included in Queries with the WHERE clause. One common use for Querying metadata is to make sure a call is valid for a certain Query by checking on the call length. There are several ways to do this:

- to use the length in milliseconds
- to look at the number of transitions between the Agent and the Consumer
- to check the length by word (example shown below)

count SELECT mediaId, hits() FROM media WHERE "Agent" SAYS "may be recorded" AND wordCount > 3000

Changing the LIMIT

By default, the Analytics Workbench limits the results from a query to the top 10 conversations. For testing and development, it is good to look at a larger sample size. Using the LIMIT clause will allow you to specify how many results you want to be returned; the maximum is 50. Note: It will always show the total number of conversations with hits and the size of the call library on top of the results window. Also note, that the LIMIT clause doesn’t require you to use another operator like AND or OR, it can be used by itself just like in the example below:

SELECT mediaId, hits() FROM media WHERE "Agent" SAYS "may be recorded"/1 LIMIT 50

Saving as a Category

After you've developed and tested your Query it's time to save it as a Category. The process for saving your Query as a Category is as follows:

- Click on **Save as category** under the VBQL editor
- Determine which "Collection" name is most appropriate, as this will be important when viewed in Tableau
- Choose the appropriate Collection name from the dropdown
- Fill out all other required and optional fields in your new Category tab and save the new Category

Voila! You now have a new Category that will begin automatically processing each new piece of Media entering in the product.

Planning a collection

- Collections determine how you will filter your categories.
- Which categories would you like to see together?
    - We can also use Groups as another field to filter by.
    - Collections must be chosen from a dropdown, but Groups can include anything.
- Plan ahead so you don't end up with collections that only have 1-2 categories.
- Some commonly used collections include:
    - Call Driver
    - Agent Quality
    - Alerts
    - Products
    - Issues

Expanding the Query with Operators

Let’s start to expand our query and look at the different results we can get. Here’s the original query again:
SELECT mediaId, HITS() FROM media WHERE "agent" SAYS "thank you"
We’re going to focus on the operators we can use after SAYS:

The AND operator is used to search two terms in the entire transcript. If the query is, WHERE "agent" SAYS "thank you" AND "survey" we are looking searching the entire transcript for where the agent says “thank you” and “survey”.

Using the && (double ampersand) operator is a little different than AND. This time, the query is WHERE "agent" SAYS ("thank you" && "survey") and we’re going to look at the speaker turn where the agent says “thank you” and “survey”. For example, if the agent were to say, “thank you for calling. Would you help us by taking a survey?”

Where AND and && search for two terms together, the OR operator is used to search for one term or the other. Like AND, this is going to search the entire transcript. The query could be, WHERE "agent" SAYS "thank you" OR "thanks"

The || (double pipes or pipes) operator is different because it looks at the speaker turn like the && operator. The query here could be, WHERE "agent" SAYS ("thank you" || "thanks")

The AND NOT operator is used when you want to find a specific term and not another term that might be similar. For example, we might want to find where agents are specifically saying “thanks for calling” and not “appreciate you calling”. Again, like AND or OR, this is going to search the entire transcript.

The ! (bang) operator, like the && and || operators, is going to search the speaker turn. Our query is going to look like this:
WHERE "agent" SAYS ("thanks for calling" && ! "appreciate you calling")

Using * (wildcard) will help increase matches and simplify your queries. Let’s say we wanted to find “week”, “weekend”, “weekday”, or “weeks”. The query would look like WHERE "agent" SAYS "week*"

The SLOP and / operators are broader wildcard queries. Using SLOP or / allows you to fill in the blanks with phrases. For example, you could use a query like:
WHERE "agent" SAYS ("thanks for calling" || "thank you for calling" || "thanks so much for calling") but a simpler way would be to use:
WHERE "agent" SAYS ("thank* calling"/3)

Above, we’re using a * (wildcard) for the word "thank" to cover "thank" and "thanks". the /3 allows for 3 wildcard terms between "thank*" and "calling". So, "thanks for calling”, “thank you for calling”, “thanks so much for calling” would all match here.

[beginningtime s, endingtime s]

Time constraints. Useful for finding something during the first xx seconds of a conversation or during the last xx seconds of a conversation.

Adding Operators to the Query

Our initial VBQL query was:

SELECT mediaId, HITS() FROM media WHERE "agent" SAYS "thank you"

The above query is going to find all conversations where the agent says “thank you”. That could be at any point during the conversation. Now that we’ve learned how we can expand our query to gather more meaningful results, let’s expand our query. Maybe I want to find all forms of gratitude by the agent AND I want the query to focus on the last 30 seconds of the call. Using our operators from above, we can now do this:

SELECT mediaId, HITS() FROM media WHERE "agent" SAYS ("thank*"||"you for call*"/2||"appreciate*||"wonderful day"/2||"great day"/2||"good morning"/2||"good after*"/2||"good night"/2||"enjoy day"/2||"enjoy week*"/2||"great week*"/2||"good week*"/2||"cheers") [-30s,]

With the above query, I’m now looking for any instance where the agent expresses gratitude by saying something like, “thanks for calling”, “have a good weekend”, “enjoy the day” or “great day” during the last 30 seconds of the call.

Examples

A simple, catch-all query

VBQL

Date Range

Find records within a given date range

SELECT * FROM media WHERE dateCreated > '2022-06-06T00:00:00.000Z' AND dateCreated < '2002-08-08T23:59:59.000Z' ORDER BY dateCreated DESC

Greater than Date

Find records after a given date range

SELECT * FROM media WHERE dateCreated > "2022-06-06T08:00:00.000Z"

Media ID

Find a record using the mediaId

WHERE mediaId="4da3b01e-d954-4aeb-a3b3-d059a1da93d1"

All conversations

Finds all records

SELECT * FROM media WHERE * ORDER BY dateCreated DESC

Interaction Type

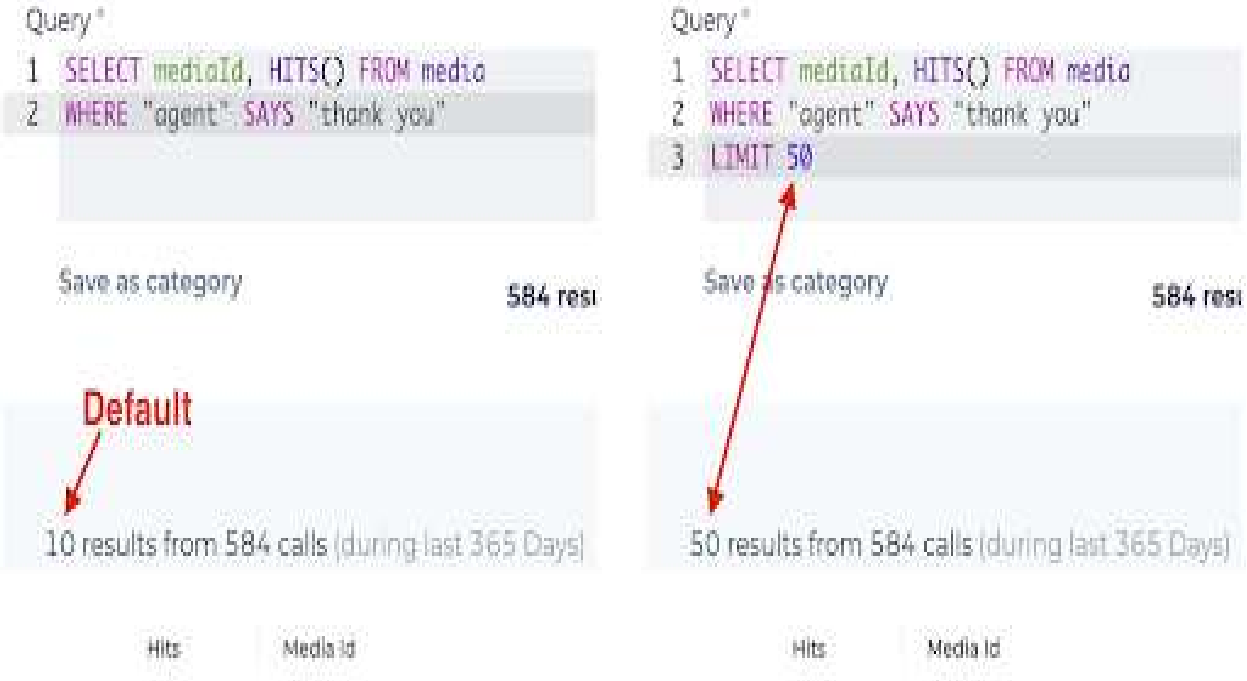
Find all records by speech (voice) or chat

SELECT *, HITS() FROM media WHERE interactionType="speech"

WHERE interactionType="chat"

LIMIT

Use LIMIT (row_count) to increase the numbers of records returned in a search. The default result is 10 rows. Using LIMIT, you can increase the number of rows. The maximum to return is 50 results.



limit

Orders by

Orders results by specific field(s) for ad-hoc query building. Options include; field_name, alias_name, ASC (ascending), or DESC (descending)

In this example, we are using ORDER BY to sort by the Date Created column in DESC (descending) order.

order by

Filter by METADATA

Utilize metadata fields within queries to specify a subset of conversations to analyze. Examples would be metadata.department="international shipping" or metadata.campaign_name= EOY 50% Off"

Iterative Category Building

Our recommendation for building categories is a workflow like the following:

1. Import a base list of categories.
2. Process several thousand conversations (anywhere from 10K-100K works fine)
3. Review imported categories against processed conversations.
4. Review business use cases and build new categories.
5. Compare processed conversations with existing categories and add additional categories as needed.
6. Re-apply categories to conversations that have already been processed.
7. Turn conversation processing back on.

Do's and Don'ts

Do's

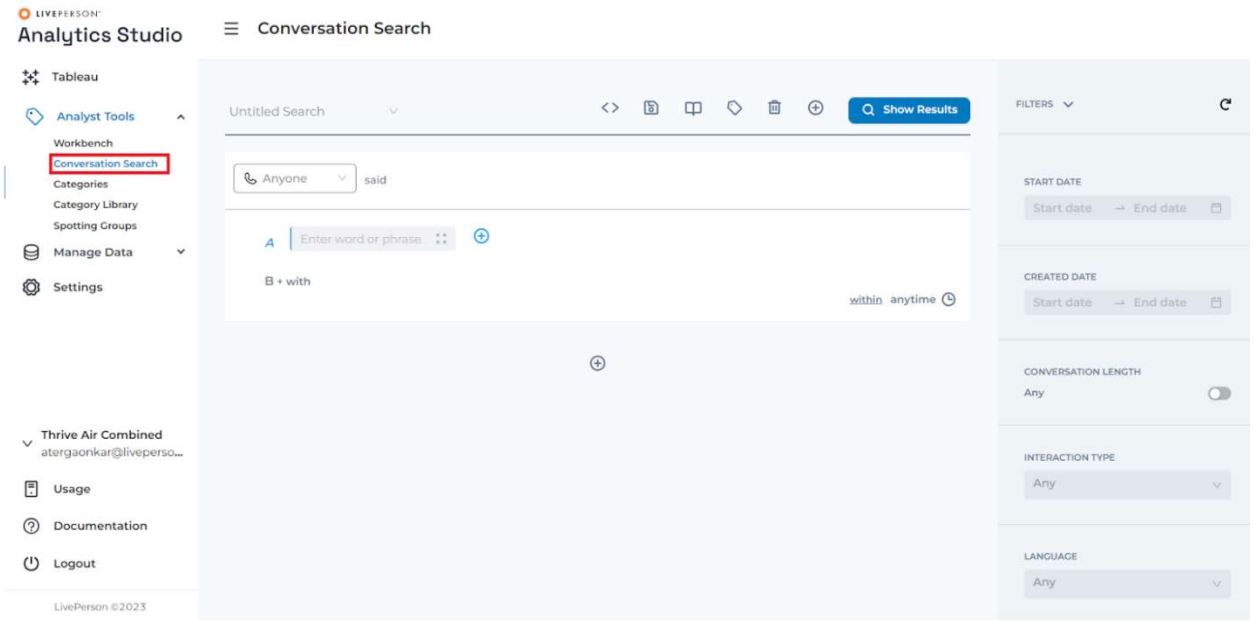
- Build categories that hit on > 1% of conversations
- Limit the collections

Don'ts

- Blindly moving category rules over from different platforms.
- Building categories with only a single word.
- Categories don't need to take weeks to build, but they do require some thought.

Conversation Search

Although the Analytics Workbench is the first choice for most wanting to search conversations and create categories, some prefer a simplified UI that abstracts the complexities of VBQL. **Conversation Search** is a lightweight visual search UI that extends the VBQL Analytics Workbench.

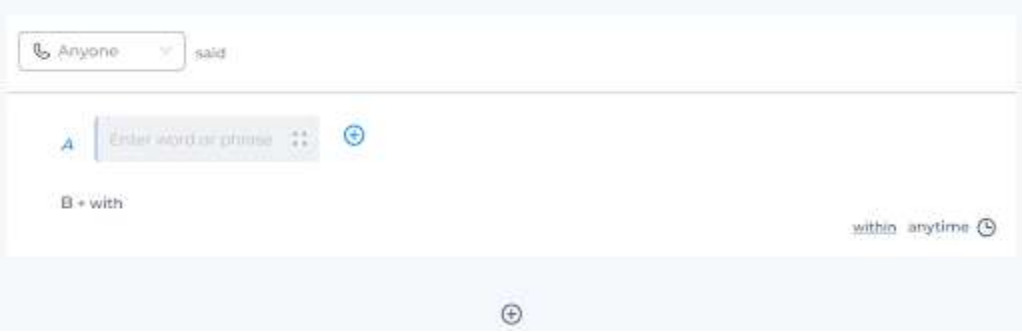


Although the Analytics Workbench is the first choice for most wanting to search conversations and create categories, some prefer a simplified UI that abstracts the complexities of VBQL. Conversation Search is a lightweight visual search UI that extends the VBQL Analytics Workbench.

To access the Conversation Search, log in to Analytics Studio and go to **Analyst Tools > Conversation Search** on the left-hand menu.

The Search

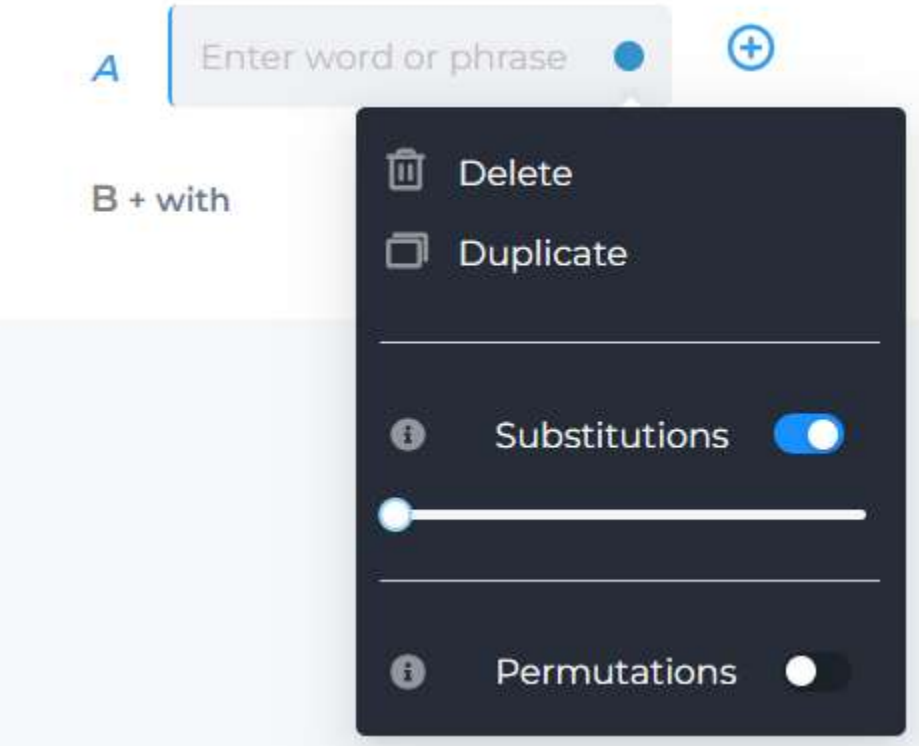
Each white cards represent participant turns. A turn is a segment of conversation, typically what could be spoken in 20 seconds. Each turn could be an ‘Agent’ or ‘Bot’ or ‘Consumer’ or ‘Anyone’. Every operator within the white card is a turn-level operator. The “or” would translate to || and the “with” would translate to &&. The colored/lettered sub-sections are groups of terms, the blue A section would be bookended with parentheses to group these together and within the participant's turn, from where A starts and letter n ends there would be parentheses bookending them together. The ‘Within anytime’ in the lower right corner of a white card is the timeframe this participant turn should be tagged within.



Conversation Search 1

A turn can be added by clicking on the Add Turn operator between the white cards. The operator could have one of the three values - ‘And’, ‘Or’ ‘Andnot’.

The Word Menu



Substitutions represent what we know as SLOP in VBQL, when turn on a slider from 1-10 will appear for the user to pick how many substitutions they would like to allow.

Permutations represent what we know as wildcard (*) in VBQL, when turned on I would like to apply some logic that replaces the last letter, of words with 4 letters or more, with the wildcard (*).

The Filters

All filters on the right are applied to the entire search.

The Top Menu

The top menu comprises multiple useful features as shown below

Preview VBQL

A click of this button will show you the Query that could either be copied or exported as a Category. This is a very useful feature to learn the VBQL syntax.

Save Search

The search could be saved for future use.

Bookmarked Searches

Saved searches could be bookmarked for ease of access.

Export as Category

A one-click button to export the search as a Category.

Delete

To delete a search from saved searches.

New Search

To start a new search.

Categories

Analytics Studio allows you to analyze, tag, and optimize insights around voice and messaging data. The tags are called Categories.

The **Categories** menu allows you to access categories, create new categories, import predefined categories, and much more!

The bar chart at the top shows the percentage of conversations tagged to each Collection. The **Add Category** button at the top allows you to create new categories. However, we would suggest the users use Workbench to first try and run the VBQLs before they create the categories.

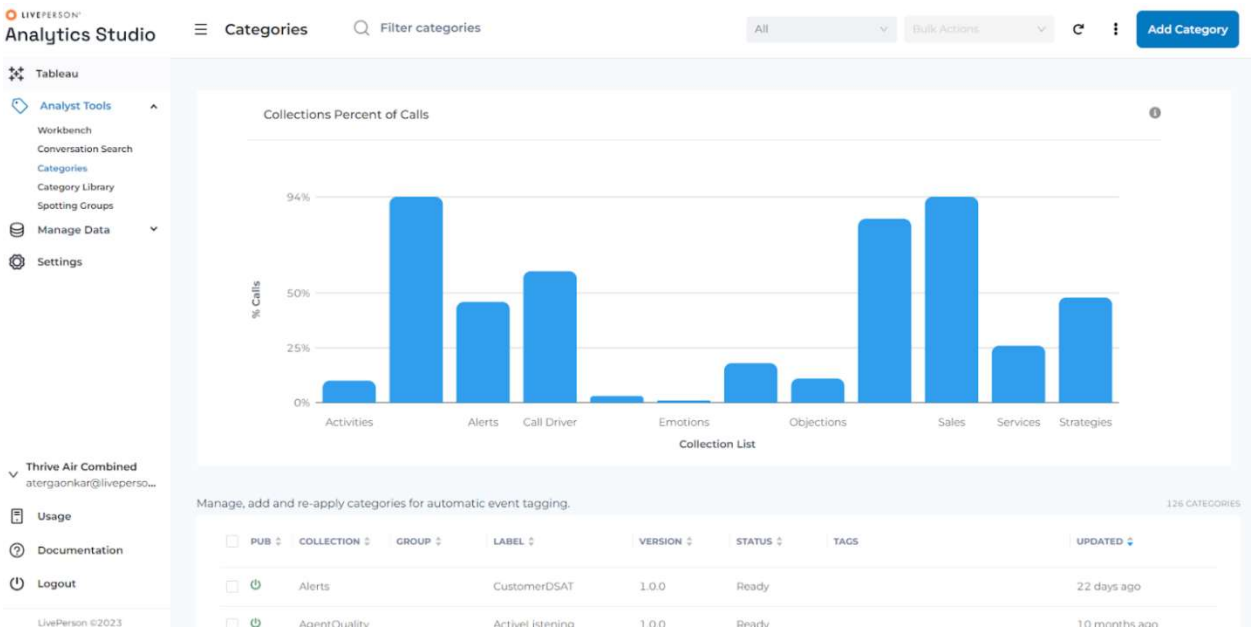
The 'Bulk Action' menu provides users with various options on the selected categories.

1. Re-apply Categories

Upon creating a new category, it will affect only upcoming conversations. To extend the category's impact to previous conversations, users must pick a category/categories and specify a timeframe for the categories to take effect. The "Re-apply Categories" feature offers this choice.

1. Delete Categories : The selected categories will be deleted.
2. Publish & Unpublish Categories

Users have the option to generate categories and opt not to make them public. This will maintain the categories in a prepared state without tagging them in conversations.



FAQs