Customer

Cloud, Corporate, Store

You can query the `customer` endpoint to retrieve customer information. Recall that customer records are different from memberships; customers refer to end users that have placed orders or made purchases anywhere in the POS system; whereas memberships are customers that have actively joined the loyalty programs and gift card features of POS.

The `customer` endpoint has multiple types of filters that can be used when querying for customers. Usage of these filters are subject to different permission levels.

The caller must have `ViewCustomers` permission to call this endpoint.

Return a list of customers:

```
query MyQuery {
  customer(first: 50, skip: 0) {
    hasMoreResults
    limitedByPermission
    results {
      id
      memberId
      status
      fullName
    }
  }
}
```

This example illustrates a few concepts:

- Paging is implemented using the `first` and `skip` input parameters. Paging is recommended to avoid downloading large numbers of results in a single API call. If no limit is specified, the server

will enforce an arbitrary limit.

- The `hasMoreResults` value is set to true if it appears there are more results that can be paged.
- The `limitedByPermission` value is set to true if the caller's permissions prevents a full list of customers from being retrieved.

> The customers endpoint has many more result fields that are available, including fields for contact and address information. Use GraphiQL or your favorite GraphQL tool to explore the full list of fields.

Return the customer that matches a specific customer ID:

```
query MyQuery {
  customer(id: "02499497793190625396") {
    results {
      id
      memberId
      status
      fullName
    }
  }
}
```

Return the customer that matches a specific membership ID:

```
query MyQuery {
  customer(memberId: "000010049") {
    results {
      id
      memberId
      status
      fullName
    }
  }
}
```

Return the contact phone number for a customer matching the specified customer ID:

```
query MyQuery {
  customer(id: "04499096027743125505") {
    results {
      id
      phone {
        raw
        normalized
        display
        isValid
        region
      }
    }
  }
}
```

There are multiple ways of representing phone numbers:

- The `raw` value was the original text entered for the phone number.
- The `normalized` value is the entered number converted to the standard E164 format, if possible. This is good for storing the number in a database, for example.
- The `display` value is the entered number converted to a format that is appropriate to the system's culture. This is good for displaying the number in a user interface, for example.

Return the shipping addresses (aka Ship To addresses) for a customer matching the specified customer ID:

```
query MyQuery {
  customer(id: "04499096027743125505") {
    results {
      id
      shipping {
        addresses {
          code
          address {
            addressLine1
            addressLine2
```

```
            addressLine3
            city
            stateCode
            state
            countryCode
            country
            zip
          }
        }
      }
    }
  }
}
```

Of note:

- The `code` value is a unique identifier for each shipping address.
- States and countries are represented either by an official name or system code.
    - In most cases – depending on POS configuration – the code values are the official 2 character ISO codes; e.g. TX for Texas, US for USA.
    - Code values may be null if they are unknown.