Implementing Customer Billing

This page explains how to configure and operate the TapeTrack billing process for a Customer.

The billing process converts TapeTrack journal activity into one Customer-specific statistics file, applies the Customer's billing rules, creates an XML invoice file, and imports that file into QuickBooks Desktop.



This procedure is intended for TapeTrack administrators and billing operators. Complete the configuration and acceptance-test sections for each Customer before including that Customer in a production billing run.

Billing Process

The billing process has four stages:

1. TapeTrack records operational activity in the journal database.
2. TMSS10LogStatsExtractDB extracts activity for the billing period into a Customer-specific `.ttstats` file.
3. TMSS10LogStatsProcess applies the billing definition and mapping files and creates an XML invoice file.
4. The TapeTrack QuickBooks Interface imports the XML file and creates an invoice in QuickBooks Desktop.

Stage	Input	Output	Purpose
Journal capture	TapeTrack transactions	Journal records	Records storage, movement, inventory, notes, and closed consignments.
Statistics extraction	Journal database	`<Customer-ID>.ttstats`	Produces daily and summary usage records for one Customer.
Billing calculation	`.ttstats`, `.ttsdef`, and `.ttmap` files	Customer invoice XML	Converts operational quantities into billable items.
Invoice import	Customer invoice XML	QuickBooks invoice	Creates one invoice for the Customer.



The complete process is Customer-scoped. Each statistics file and each invoice XML file must contain only one Customer.

Before You Begin

Confirm that the following components are installed and available:

- A functioning TapeTrack Framework Server with journal activity for the Customer.
- TMSS10LogStatsExtractDB.
- TMSS10LogStatsProcess.
- A billing definition file, normally with the `.ttsdef` extension.
- A Customer description map and, when package-based rules are used, a package map.
- The TapeTrack QuickBooks Interface on a Windows computer with access to QuickBooks Desktop.
- QuickBooks Customers and Items that match the values which will be produced by the billing rules.

The billing administrator should have an approved contract or rate schedule that defines:

- Which storage locations and media types are billable.
- Whether storage is billed by total, high-water, or low-water quantity.
- Required rounding, minimum, threshold, and cap values.
- Billable movement, inventory, and journal categories.
- Fixed, monthly, quarterly, or annual charges.
- The QuickBooks Customer name or ListID and Item names.

Define the Customer Billing Profile

Create and approve a billing profile before editing the billing configuration.

Setting	Customer value	Example
TapeTrack Customer-ID	*Required*	`OS03`
TapeTrack Customer Description	*Required*	`ACME Data Services`
QuickBooks Customer reference	*Required*	`ACME Data Services`
QuickBooks reference type	FullName or ListID	`FullName`
Billing package	Optional	`STANDARD`
Billable Media-ID values	*Required*	`LT`, `LTO`
Billable Repository-ID values	*Required*	`COV`, `RVLT`
Storage basis	Total, high-water, or low-water	`High-water`
Rounding increment	Contract value	`50`
Minimum quantity	Contract value	`0`
Included threshold	Contract value	`0`
Maximum quantity	Contract value	`0` for no cap
QuickBooks Item names	*Required*	`A - Storage:QBROTATIONBOX`
Fixed charges	Optional	Monthly account fee

Store the approved profile with the billing configuration so that a later rule change can be traced to a contract requirement.

Configure Mapping Files

Customer Description Map

The Customer description map supplies the $CUSTOMER2 variable used by billing rules. Add one line containing the TapeTrack Customer-ID followed by the value expected by QuickBooks.

Customer.ttmap

```
# TapeTrack Customer-ID    QuickBooks Customer name
A099                      ACME Data Services
```

When TMSS10LogStatsProcess is run with the -M option, $CUSTOMER contains the Customer-ID from the .ttstats file and $CUSTOMER2 contains the mapped description.



If a rule outputs $CUSTOMER2 and no map entry exists, the processor reports that no value was found. Do not import the resulting invoice until the mapping has been corrected and the XML regenerated.

Package Map

Create a package map when the billing definition filters rules by package. The package assigned to the Customer must match the package filter in the applicable billing rules.

If the Customer does not use package-based filtering, confirm whether the site's standard package map or wildcard rules should apply.

Configure Billing Rules

Billing rules are stored in a .ttsdef file. The processor supports the following directives:

Directive	Use
AddStorageChargeItem	Bills storage or slot quantities by package, Customer, media, and repository.
AddInventoryChargeItem	Bills inventory quantities by package, Customer, and media.
AddMovementChargeItem	Bills movements by package, Customer, media, source repository, and destination repository.
AddJournalChargeItem	Converts journal categories and quantities into invoice items.
AddAppendFile	Adds approved external invoice XML content, normally for fixed or recurring charges.

Use the processor's -l option to display the rule signatures supported by the installed version:

```
TMSS10LogStatsProcess -l
```

Storage Rule Example

The storage rule signature is:

```
AddStorageChargeItem(
  packageFilter,
  customerFilter,
  mediaFilter,
  repositoryFilter,
  roundUpFactor,
  minimumUnits,
  threshold,
  cap,
  model,
  outputCustomer,
  outputItemCode,
  descriptionOverride,
  costOverrideInCents,
  continueFlag
);
```

The following example bills Customer A099 for high-water storage of LT media in repository COV, rounded to the next 50 units:

US01.ttsdef

```
AddStorageChargeItem("*", "US01", "LTO", "OFFS",
                     10, 0, 0, 0, 1,
                     "$CUSTOMER2", "A - Storage:QBRACKSTORAGE",
                     "LT media storage", 0, 0);
```

In this example:

- * allows any package.
- US01 restricts the rule to one Customer.
- LTO and OFFS select the Media-ID and Repository-ID.
- 10 rounds a non-zero quantity up to the next group of 10.
- Model 1 selects real high-water storage.
- $CUSTOMER2 outputs the mapped QuickBooks Customer name.
- A cost override of 0 outputs a rate of 0.00; the QuickBooks Item default rate can then apply.
- A Continue value of 0 marks matched data as spent so that a later storage rule cannot bill it again.

Calculation Order

For a matching rule, the processor calculates the selected model and then applies adjustments in this order:

1. Round up.
2. Enforce the minimum.
3. Subtract the threshold, or set the quantity to zero when it does not exceed the threshold.
4. Apply the cap.

If the resulting quantity is zero, no invoice item is produced.

Billing Models

Data type	Model 0	Model 1	Model 2	Additional storage models
Storage	Daily total	Real high-water	Real low-water	3 = slot total; 4 = slot high-water; 5 = slot low-water
Movement	Total movements	Movement high-water	Movement low-water	Not applicable
Inventory	Total inventory	Inventory high-water	Inventory low-water	Not applicable
Journal	Total quantity	Journal high-water	Journal low-water	Not applicable

Rule order matters. When Continue is false, matching data is marked as spent and cannot be billed by a later rule of the same type. Review general rules and Customer exceptions together.

Pricing

A rule cost override is expressed in cents. When the output cost is `0.00`, the QuickBooks Interface omits the QuickBooks `Rate` element, allowing the default price configured for the QuickBooks Item to apply.

Confirm which system is the authoritative price source before testing:

- Use a non-zero rule cost when TapeTrack must supply the invoice rate.
- Use a zero rule cost when QuickBooks Item pricing must supply the rate.

Never use `99999.99` as a valid price. The QuickBooks Interface treats that value as a signal to skip the invoice line.

Perform an Acceptance Test

Test a new Customer separately before including it in the normal billing run.

Extract Customer Statistics

Create a dedicated test output directory and run the extractor for a known date range. The following

command is a template; replace every value in angle brackets with the value required by the installed environment:

```
TMSS10LogStatsExtractDB \
  -h <journal-database-home> \
  -R <date-range> \
  -o <test-output-directory> \
  -c A099 \
  -C <customer-translation-map>
```

Use `-H` when the contract bills the maximum snapshot count observed during each day. Use `-m` only when the agreed process requires billing a Volume in multiple locations on the same day.

The accepted `-R` date-range syntax is supplied by the installed TapeTrack framework and may differ between environments. Confirm it from the installed utility's help before running production billing.

The extractor creates `A099.ttstats`. Confirm that:

- The first line is a `C` record for the expected Customer-ID.
- The file contains the expected billing dates.
- Media and repository values match the Customer billing profile.
- Daily storage, movement, inventory, and journal activity is plausible.
- Final rows dated `00000000` reconcile with the daily data.

Final `00000000` rows are summaries for validation. The billing processor deliberately ignores those rows when calculating storage, movement, and inventory charges.

Generate the Invoice XML

Process the Customer statistics file with the approved definition and maps:

```
TMSS10LogStatsProcess \
  -i <test-output-directory>/A099.ttstats \
  -c <billing-definition>.ttsdef \
  -M Customer.ttmap \
  -P <package-map>.ttmap \
  > <test-output-directory>/A099.xml
```

Omit `-P` only when package mapping is not required. The `-N` option suppresses the XML declaration, stylesheet reference, and `<tapetrack_data>` wrapper and should not be used for a normal standalone QuickBooks import file.

Validate the XML

Before importing the XML, verify:

- The file is well-formed XML.
- Every `<item>` contains the same `<customer>` value.
- The Customer value exists in QuickBooks as the selected FullName or ListID.
- Every `<itemcode>` exists in QuickBooks.
- Quantities agree with the contract's model, rounding, minimum, threshold, and cap.
- Duplicate item codes are intentional; the interface does not merge duplicate lines.
- Zero rates are intentional and the corresponding QuickBooks Items have the correct default rates.
- Fixed or recurring items included by `AddAppendFile` are present once and contain valid XML.

Import into a QuickBooks Test Company

1. Open the appropriate QuickBooks Desktop company file.
2. Open the TapeTrack QuickBooks Interface.
3. Confirm the `CustomerIsListID`, `IsToBeEmailed`, and `IsToBePrinted` preferences.
4. Drag the tested Customer XML file onto the interface window.
5. Wait for the interface to report the QuickBooks invoice number.
6. Open the invoice in QuickBooks and compare every line with the source XML.
7. Confirm that QuickBooks applied the expected default rate where the XML cost was `0.00`.

If the interface reports an error, double-click the error entry and retain the QuickBooks response XML for troubleshooting.

Approve the Customer for Production

Do not enable production billing until the following items have been approved:

- Customer and QuickBooks mapping.
- Billing package assignment.
- Billable media, repositories, movements, inventory, and journal categories.
- Storage models and calculation adjustments.
- Item codes and pricing source.
- Fixed and recurring charges.
- Test-period `.ttstats` reconciliation.
- XML invoice review.
- QuickBooks test invoice review.

Record the approved versions of the `.ttsdef`, Customer map, package map, and append files.

Monthly Billing Procedure

1. Confirm the billing period and Customer scope.
2. Take a versioned copy of the active billing definition, maps, and append files.
3. Run `TMSS10LogStatsExtractDB` for the billing period.
4. Review extractor logs and confirm that every expected Customer produced a `.ttstats` file.
5. Reconcile daily data with the final `00000000` summary rows.
6. Run `TMSS10LogStatsProcess` once for each Customer file.

7. Validate each XML file before import.
8. Import each XML file using the TapeTrack QuickBooks Interface.
9. Record the returned QuickBooks invoice number against the Customer XML file.
10. Compare the created invoice with the XML, including rates supplied by QuickBooks.
11. Retain the audit package for the billing period.

The audit package should contain:

- Source `.ttstats` files.
- The billing definition used.
- Customer and package maps used.
- Referenced append files.
- Final invoice XML files.
- Extractor and processor logs.
- QuickBooks invoice numbers and any error response XML.

Troubleshooting

Symptom	**Checks**
Customer statistics file was not created	Confirm the date range, Customer filter, Customer-ID translation map, journal activity, and output-directory permissions.
Invoice XML contains no items	Confirm that a rule matches the Customer, package, media, repository, or category. Check whether rounding, minimum, threshold, or cap processing reduced the count to zero.
Wrong Customer is used in QuickBooks	Check `$CUSTOMER2` in the Customer map and the `CustomerIsListID` preference.
QuickBooks rejects the import	Confirm that the Customer and Item references exist, the XML is valid, and the QuickBooks connection is active. Review the stored QuickBooks response XML.
Invoice quantities are unexpected	Confirm the model number, `-H` and `-m` extractor options, rule order, Continue value, and the calculation adjustments. Remember that billing uses daily rows rather than `00000000` summary rows.
Duplicate invoice lines are created	Check for overlapping rules or duplicate append content. The interface does not combine repeated Item codes.
Rate is missing from the invoice request	A cost of `0.00` intentionally omits the QuickBooks Rate. Confirm the QuickBooks Item's default price.
Consignment charge is missing	The extractor writes R records for closed consignments, but the documented processor does not consume R records. Confirm whether consignments are billed through another rule or process.

Known Limitations

- Closed consignment R records are not processed by the documented version of `TMSS10LogStatsProcess`.
- Journal note text is retained in the statistics file, but billing rules use the journal category and quantity rather than the free-text note.
- The QuickBooks Interface ignores the XML `<total>` field.
- Append files are inserted directly into the processor output after variable substitution. Keep them version-controlled and validate their XML.
- A zero cost can result in QuickBooks-side pricing. The final invoice must therefore be reconciled

in QuickBooks, not only against the processor's XML totals.

Related Pages

- Implementation and Planning Guide
- TapeMaster Configuration
- Adding and Maintaining Customers
- Customer Object

From:
https://docs.tapetrack.com/ - **TapeTrack Documentation**

Permanent link:
https://docs.tapetrack.com/cookbook/billing_process_temp?rev=1787107379

Last update: **2026/08/19 02:42**