先ほどと同様にpd.mergeを用いてジョインを行なっています。

1行目でcustomer\_masterと、2行目でitem\_masterと結合を行なっています。

先頭5行の出力結果を見ると、顧客情報、商品情報が付与されているのが確認 できます。

これで、4種類6個のデータを1つに結合し、分析できるデータに整形できました。

しかし、結合した影響で売上 (price) が落ちてしまっているので、売上を計算する必要があります。

## // ノック5:必要なデータ列を作ろう

売上列を作るためには、quantityとitem\_priceの掛け算で計算できます。 計算した後、確認のため、quantity、item\_price、price列の先頭5行を出 力してみましょう。

join\_data["price"] = join\_data["quantity"] \* join\_data["item\_price"]
join\_data[["quantity", "item\_price", "price"]].head()

## ■図1-6:売上列の作成

## 

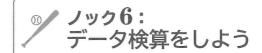
1行目でpandasのデータフレーム型の掛け算を実行しています。データフレーム型の計算では、行ごと(横方向)に計算が実行されます。

先頭5行目の出力結果を見ると、quantityが2の行のpriceが単価の2倍になっており、しっかり計算が実行できているのが確認できました。

これで、一通りのデータ加工は完了しました。

ただし、データ加工は、一歩間違えると集計ミスが起き、数字のズレを生みます。 間違ったデータを提供することは、会社の経営に大きな影響を及ぼし、最悪の場合、 会社が傾くこともあります。また、個人でみても、データで語るデータサイエンティ ストが誤ったデータを出すというのは、顧客からの信頼を失います。データを結 合したりする度に、**件数の確認等を行うことを心がけてください**。また、なるベ くデータの検算ができる列を探し、**検算を実行**するようにしましょう。

今回のケースでは、price列で簡単なデータ検算が行えそうなので、やってみましょう。



データ加工前のtransactionデータにおけるpriceと、データ加工後に計算に よって作成したprice列は合計すると同じ値になるはずです。

細かくデータを見ていくケースもありますが、今回は、簡易的にそれぞれの price 合計の値を確認してみましょう。

print(join\_data["price"].sum())
print(transaction["price"].sum())

出力すると、971135000が2つ表示され、完全に一致していることが確認できました。

また、下記のように記述し、True/Falseで確認しても良いでしょう。

join\_data["price"].sum() == transaction["price"].sum()

これで、データの検算も無事終了しました。

繰り返しになりますが、誤ったデータで分析しないように、データ加工の検算 は常に意識してください。

それでは、いよいよデータ分析に移っていきます。