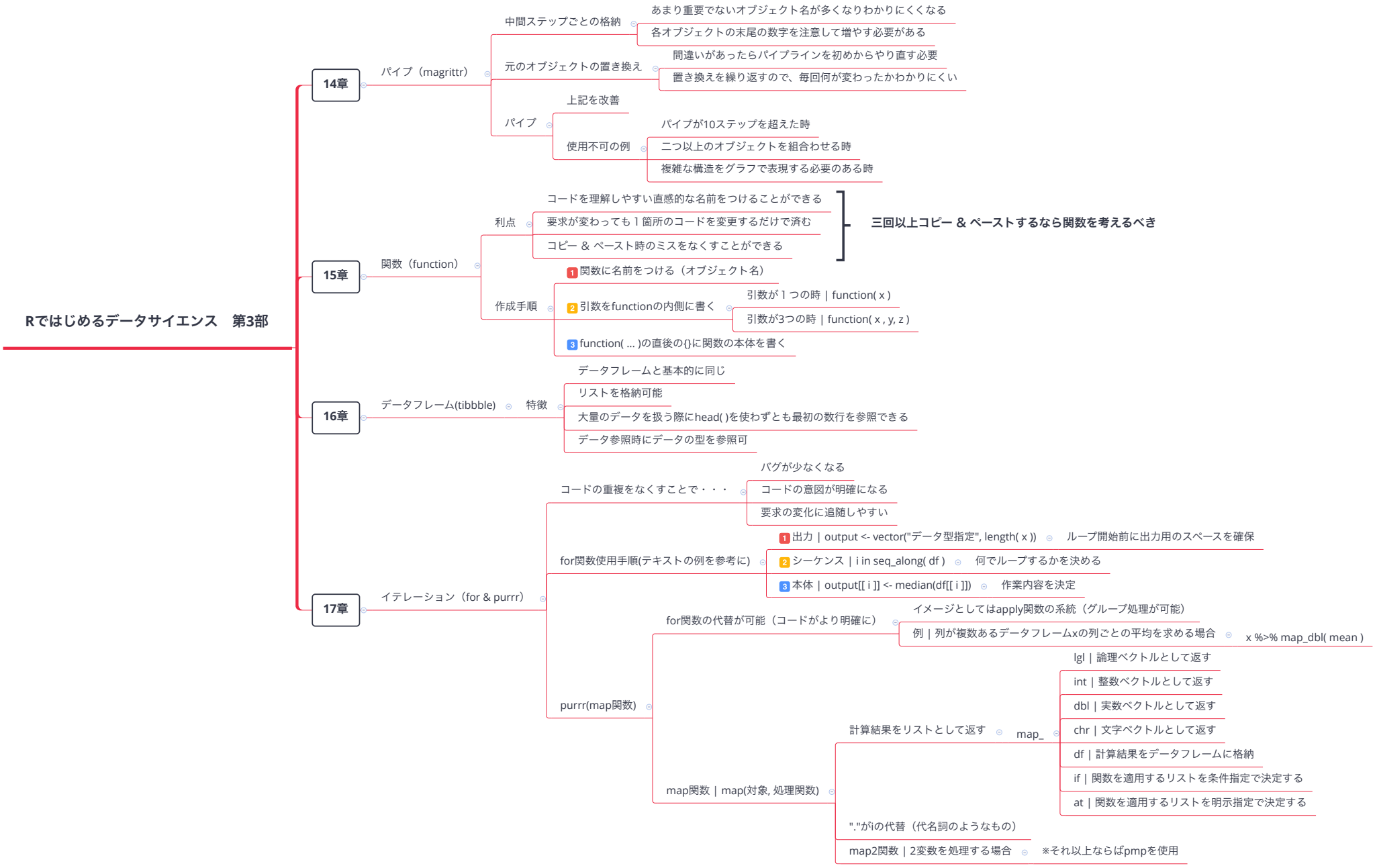
Rではじめるデータサイエンス　第3部
14章
パイプ（magrittr）
中間ステップごとの格納
あまり重要でないオブジェクト名が多くなりわかりにくくなる
各オブジェクトの末尾の数字を注意して増やす必要がある
元のオブジェクトの置き換え
間違いがあったらパイプラインを初めからやり直す必要
置き換えを繰り返すので、毎回何が変わったかわかりにくい
パイプ
上記を改善
使用不可の例
パイプが10ステップを超えた時
二つ以上のオブジェクトを組合わせる時
複雑な構造をグラフで表現する必要のある時
15章
関数（function）
利点
コードを理解しやすい直感的な名前をつけることができる
要求が変わっても１箇所のコードを変更するだけで済む
コピー & ペースト時のミスをなくすことができる
三回以上コピー & ペーストするなら関数を考えるべき
作成手順
1 関数に名前をつける（オブジェクト名）
2 引数をfunctionの内側に書く
引数が１つの時｜function(x)
引数が3つの時｜function(x , y, z)
3 function(...)の直後の{}に関数の本体を書く
16章
データフレーム(tibbble)
特徴
データフレームと基本的に同じ
リストを格納可能
大量のデータを扱う際にhead( )を使わずとも最初の数行を参照できる
データ参照時にデータの型を参照可
17章
イテレーション（for & purrr）
コードの重複をなくすことで・・・
バグが少なくなる
コードの意図が明確になる
要求の変化に追随しやすい
for関数使用手順(テキストの例を参考に)
1 出力｜output <- vector("データ型指定", length(x))　ループ開始前に出力用のスペースを確保
2 シーケンス｜i in seq_along(df)　何でループするかを決める
3 本体｜output[[i]] <- median(df[[i]])　作業内容を決定
purrr(map関数)
for関数の代替が可能（コードがより明確に）
イメージとしてはapply関数の系統（グループ処理が可能）
例｜列が複数あるデータフレームxの列ごとの平均を求める場合　x %>% map_dbl(mean)
map関数｜map(対象, 処理関数)
計算結果をリストとして返す
map_
lgl｜論理ベクトルとして返す
int｜整数ベクトルとして返す
dbl｜実数ベクトルとして返す
chr｜文字ベクトルとして返す
df｜計算結果をデータフレームに格納
if｜関数を適用するリストを条件指定で決定する
at｜関数を適用するリストを明示指定で決定する
"."がiの代替（代名詞のようなもの）
map2関数｜2変数を処理する場合　※それ以上ならばpmpを使用