問13,

すべての生徒の苗字（漢字）を抽出し、"Family name"カラムに、  
名前（漢字）を抽出し、"Given\_name"カラムに値を入れてくださいという問題です．

生徒の名前という文字列を分割したりする必要がある．

この時，何とかして機械的に「苗字」と「名前」をわけて抽出できかを考えてみます．

まずは，データを見てます，

するとこのデータでは，どうやら苗字と名前の間に半角スペースが存在していることが確認できます．

なのでこの半角スペースを基準に苗字と名前に分割できないか考えていきます．

またこれは後ほどを述べることにはなりますがこの半角スペースだけですべてのレコードを分解することはできないので，何が原因か問題かということも同時に考えながら進めていきます．

P2-

まずはいきなり苗字と名前を抽出することはできないので

苗字のみを抽出することから考えていきます．

そのための構文はこちらになりますが，今回文字列の操作にあたり，substringとcharaindexの二つの関数を使っています．

まずは，それぞれの関数の機能について説明しますと

substring 関数というのは対象とした文字列から指定した位置で文字を切り出すことができる関数です．

構文はこちらの通りで，対象の文字列，切り出し開始位置，切り取る文字数と指定します．

今回であれば文字列から苗字をまずきりだすことを考えるので，第一引数には名前のある列名，第二引数の切り出し開始位置には1を指定することとなります．

また，半角スペースを基準に苗字と名前をわけることを考えているので，第二引数の切りとる文字数を指定するためにも，半角スペースが前から何文字目にいるかという情報もそれぞれのレコードで必要となってきます．

そこで，この２つ目のキャラインデックスという関数で検索したい文字が文字列の何番目の位置にいるかという情報を取得します，

Charaindexの

使い方としては検索対象の文字，今回であれは半角スペースを第一引数に，

第二引数に，文字列を指定します．すると，文字列の中で、検索した値が前から何番目にいるかを表示できます．

ここから上の構文に戻りますが

苗字を取りたいので substring 関数の最初の引数に名前の入ってる[ネーム]を指定し，

第二引数には1，３つめの引数には，半角スペースの位置-1とすることで苗字のみを抽出しています．

引く1をしないと半角スペースも合わせて切り出されるので，注意してください．

これで苗字は切り出されるかと思うので実行してみます．

すると，エラーとなって関数が実行できません，

今回半角スペースの位置を基準にいくつか関数使用しているので、そこから見てみると

データに半角スペースがない人がいることがこちらの構文で確認できます。

前ページ

CHARINDEX関数で検索する文字、今回であれば半角スペースが存在しない場合、実は0が返ってきます。

そしてこの帰り値-1をsubstrngの第二引数に指定していますが、substrngの第二引数には正の値しかとることができないという決まりになっているのでそのためのエラーというわけでした。

このままでは姓名を抽出できないでの対処をします。

苗字と名前の切り出す方法は、先ほどのsubstring,charaindexを使う方針で進めようと思うので，今回はこの苗字と名前の間に半角スペースがないデータについては，前回の講義で行ったように更新して、半角スペースを挿入することとします。

半角スペースがデータは3件のみなので，それぞれの名前ごとにUPDATE句で更新します．

一点注意点ですが，今回は練習用のサンプルデータであるためUPDATEでデータを更新しましたが，実際このようにデータを更新する際は要注意．お客様のデータであれば事前に確認するようにしてください，

また，今回の構文を更新無しで実行する方法も後述します．

まずは，UPDATEで更新した場合，について解説を進めますが

先程の構文で半角スペースのないデータは更新できたので，

もう一度まずは苗字を切り出します，

先程の構文を実行し，結果を確認すると問題なく実行できていることがわかるかと思います，

また，名前に関しては回答では文字列の右端から指定した文字だけを切り取るという処理になっています．

まず，右端から指定した文字数だけ取り出すにはrightという関数を用います．

右側から切り出す文字数は，

全体の文字数から，半角スペースまでの文字数を引いた値を指定しています，

こうすることで，半角スペースよりあとの文字だけを取り出しています．

こちらもためしに実行すると名前だけを抽出できていることが確認できるかと思います．

苗字と名前を抽出できたので

問題に戻って，もとデータの[Family name]カラムに、苗字を，

[Given\_name]カラムに名前を入れていきます，

更新の際は，前回ように一括で処理を行います．

こちらは苗字の更新となっていますが，UPDATE句でもとのテーブル，

更新したいカラム名をset以降に指定しますい．

更新の値には先ほどの構文をつかいます，

一括更新のために，もとのテーブルと，更新したい値を抽出したテーブルとをJOIN句で結合することとなるので， 全生徒に固有の番号である[temp\_no]も，先程の構文で選択していることに注意してください．

名前に関しても構文の構造は同じです．

ただし，更新先のカラム名が[Given\_name]カラム，ということと，この中身の構文が名前を抽出するカラムになっていることに注意してください．

以上で問題の内容を実装出来ました，

P.8

最後に，今回，半角スペースのないデータを更新するという方法を実施しましたが

UPDATEによる更新なしで，今回の処理を実施したい場合に，replace関数を用いる方法をご紹介します．

この構文全体で．苗字と名前の更新処を一つの構文で記述していますが，

Replace関数を使って，苗字と名前を抽出する部分はこちらのJOIN句以降になります．ここを見る，

replace関数というのはカラムに対して，指定した文字列を別の指定した文字列で置き換える関数になります．

苗字と名前を抽出する方法は，先程の構文と同じ方法をもちいるとすると

from句でreplace関数を使い，半角スペースのないデータの置き換えを行います．

置き換えはこちらの部分で，先程のUPDATE句を使う方法で半角スペースのない3つのデータを更新したように，[name]カラムにReplace関数を三回使用します．

この際，関数が入れ子になっていますが，

中身としては一番内側のReplace関数で[name]カラムの松本さんを半角スペースを含む名称に置き換え，その置き換えを実施したカラムに対して次は中西さん，高野さんと半角スペースを含む名称で置き換えて，一気に3つの置き換えを行います．

こうすることで，全データの[name]の苗字と名前の間に半角スペースが入ったので，

先程の構文で苗字と名前が抽出できます．

そうしましたら更新についても，先程と同様で，set句でそれぞれ苗字と名前のカラムを

指定しています．

直接元データの更新をせずに同じことをできるのでぜひ覚えておいてね．

問１４

五教科すべてにおいて平均点以上を獲得している生徒の名前と、  
その五教科得点を抽出してくださいという問題です．

この問題を解くときに意識して欲しいことは二つあります

一つ目抽出条件についてです．

今回は5教科全てにおいて平均点以上の生徒という条件です．これに基づいて対象のレコードを絞り込めば良いです.

ここまでは問題ないかと思いますが，もう一つのポイントとしてはこの平均点を設定する方法です.

単純な方法としては各教科の平均点を一旦AVERAGE 関数を使って算出し,

その後，where句の条件にその算出した値を直接指定すると方法です．

ただし，この方法を使ってしまうと,例えばデータの内容が変わると当然平均点は変わる可能性があるので，問題の条件で抽出することができず，再度，平均点を算出し，コードを書き換える必要がでてきます．

これでは面倒なのでこの平均点を算出するという処理も一連のコードの中に組み込んでしまいます．

そうすることでデータの内容が変わっても，コードを書き換える一つがなくなり，

より汎用的なコードにすることができます．なので今回はその方法ご回答としてご紹介します.

P10

回答はこちらになります．ポイントはこのウェア句の部分です．

各教科で平均点を算出するという構造，この部分ですね，を直接where句の中に記述しています．

このようにすることで先ほども申し上げたようにデータの内容が変わったとしても毎回そのデータをもとに平均点が算出され，その算出された値を条件にデータが絞り込まれるのでより汎用的なコードになっていると言えるかと思います