In-Class Practice – (SELECT)

Basic Select

1. Select all records and all columns from *Vendors* table. ***Compare with your partner to see that you used the same syntax and got the same count.***
2. Then select just the ID, name, address, city, state, zip from *Vendors*. Select all rows. **Compare with your partner to see that you get similar results.**

String Expressions and Column Alias

3. Select all vendors (ID, Name) and then their Address location but concatenate their address into this format: **Address; City, State Zip**. NOTE: Do not include alias this time
4. Add a column alias to this newly created column called **Vendor_Address**

Arithmetic Expressions

5. Using the *Invoices* table, pull ***vendor_id***, ***invoice total***, ***payment total***, and the ***difference of invoice_total & payment_total*** (i.e. invoice_total – payment_total). NOTE: Do not include alias this time
6. Rename the calculated column (i.e. invoice_total – payment_total) to **"Amount_Owed"**
7. Add in a WHERE clause that results in only the first 5 rows being returned (hint: Use the ROWNUM pseudo column.

DUAL Table

8. Go **select * from dual**. Note that this table is a built-in table with DBMSs that allows you to test different types of string and arithmetic expressions without actually having tables in your schema.
9. Now use the dual table to mess around with the following:
   - Select an explicit string like 'hello, my name is [your name here]'
   - Find the answer to 18*36

Scalar Functions

10. Using the DUAL table, return the following
    - Leverage the SYSDATE function to return today's date
    - Use the MOD function to find the remainder of 3942/17
    - Use the TO_CHAR function to change the format of SYSDATE to "MM/DD/YYYY"
    - Now let's all be British and change it to "DD/MM/YYYY"
11. Using *Invoices* table, select ***invoice_id***, ***invoice_number***, ***invoice_total***, ***invoice_date***, and a ***calculated column called "Days til Due"*** that returns the ***number of days until the invoice_date***. Hint: Use SYSDATE function. Also show days as a whole number using a 2nd function
12. Select the average invoice amount from invoices using the AVG function. Then round to the nearest 2 decimals.
13. Using *Customers_OM,* select customer initials into one column called Initials and in another, a formatted phone number (e.g. 212-555-4800) call phone_num. TIP: First write this out in pseudo code to determine what functions and logic you need

DISTINCT

14. What is the distinct number of vendors in *Vendors* table
15. What is the distinct number of vendor_IDs on invoices. TIP: First gather a list of vendor_ids in *Invoices* table and then add in the distinct keyword.

Above and beyond

16. How could you use MOD function to determine if a number is even or odd? For example, update the following SQL statements by utilizing MOD function in the SELECT portion of the statement to figure out if number is even or odd. What number would you divide by and what result could you expect to show for even or odd numbers?

`Select 23482` `From dual;`	`Select 23481` `From dual;`

17. Using the *Invoices* table, select ***invoice_id***, ***invoice_number***, ***invoice_total***, ***invoice_date***, and a ***calculated column called "Weeks til Due"*** that returns the ***number of weeks until the invoice_date in one column and the number of additional days beyond the week*** (Hint: the remainder after dividing the "Days til Due" by 7, use MOD).

18. Using the *Invoices* table, use a select statement to output one single column of text that lists the concatenation of Invoice number, date due, and payment total. (Hint: use the TO_CHAR function to convert all data into text)