# In-Class Practice – Day 16 (Data Types and Data Functions)

**Converting Dates and Numbers to String for formatting**

1. Copy/Paste and run the statement below and then format according to the examples below

**select sysdate as unformatted,**

**sysdate as formatted1,**

**sysdate as formatted2,**

**sysdate as formatted3,**

**sysdate as formatted4**

**from dual;**

* 1. *formatted1:* OCT-2019
  2. *formatted2:* October 23, 2019 *Note: ignore extra space to right of month for now. We’ll work on that later*
  3. *formatted3*: 10/23/19 06:10:01
  4. *formatted4*: WED, 10-23-2019

1. Copy/paste and run the statement below and then format according to the examples below. Add lines as needed

**select invoice\_total as unformatted**

**,invoice\_total as formatted1**

**from invoices;**

* 1. *formatted1*: $999.99
  2. *formatted2*: In this example see if you can fix the ### issue arising in formatted1
  3. *formatted3*: In this example copy the format of the previous column but remove decimals

**Using String functions to format characters**

1. Copy/paste and run the statement below and then format according to the examples below. Add lines as needed

**select vendor\_name,**

**vendor\_state,**

**to\_char(invoice\_date,'Month DD, YYYY') as invoice\_date,**

**invoice\_total**

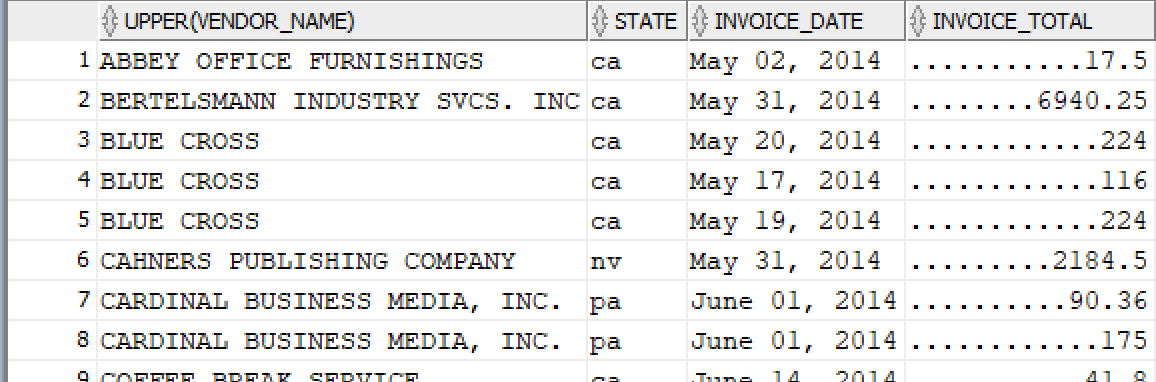
**from vendors v inner join invoices i**

**on v.vendor\_id = i.vendor\_id**

**order by v.vendor\_name;**

* 1. Update vendor\_name to be ALL CAPS
  2. Update vendor\_state to by lowercase
  3. Trim the trailing spaces after the invoice\_date Month. *Hint: You’ll have to trim Month by itself and concat to the other day and year part of the date.*
  4. Left pad invoice\_total with 15 periods (e.g. ‘.’)

NOTE: You should end up with something like this:



**More character Functions**

1. Copy/paste and run the statement below and then format according to the examples below. Add lines as needed

**select vendor\_phone as unformatted\_phone,**

**vendor\_phone as phone\_replace,**

**vendor\_phone as phone\_substr,**

**vendor\_phone as phone\_length**

**from vendors ;**

* 1. Update the 2nd phone like so (i.e 800-555-1205) using the REPLACE function. *Hint: You can nest a function inside another function.*
  2. Update the 3rd phone the same way (i.e. 800-555-1205) using the SUBSTR function
  3. Update the 4th phone to just return the LENGTH of the cell value.

1. Select the product\_name from the Products table. We want you to parse the first word of the column into a second column called *Brand* which contains the first word in the product\_name. After you do this parse the all text that comes after the first blank space in product\_name into a 3rd column called *Instrument\_Name.* ***Hint: You’ll want to consider using a combination of INSTR and SUBSTR for this***

**Numeric functions**

1. Copy/paste and run the statement below and then format according to example output shown in picture

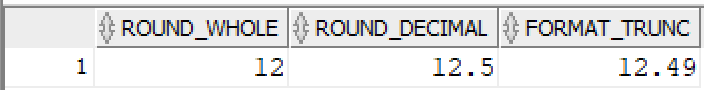
**SELECT 12.4999 as Round\_Whole,**

**12.4999 as Round\_Decimal,**

**12.4999 as Format\_TRUNC**

**FROM DUAL;**

Result:



**Date Functions**

1. Copy/paste and run the statement below and then format according to instructions below

**Select SYSDATE - TO\_DATE('01-JAN-22') as Days\_since\_New\_Year,**

**SYSDATE - TO\_DATE('01-JAN-22') as Days\_Rounded**

**from DUAL;**

* 1. Update 2nd date to be rounded to a whole number
  2. Add a 3rd column that uses the MONTHS\_BETWEEN function to capture the number of months between today’s date and Jan 1 of this year
  3. Add a column that returns today’s date plus 30
  4. Add a column that uses the ADD\_MONTHS function to add one month today’s date.

**Date/Time Searches**

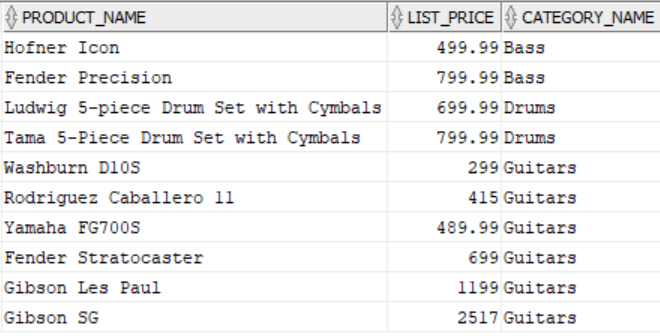
1. Select all records from the **date\_sample** table where the start\_date is Feb-28-06. If you get no records consider how the timestamp affects this and how you should update your query to account for that.

**CASE Function – Simple**

1. Select the product\_name and list\_price from products. Add a third column that uses a CASE based on the category\_id column that returns the following:
   1. When category\_id is 1, return 'Guitars'
   2. When category\_id is 2, return 'Bass'
   3. When category\_id is 3, return 'Drums'
   4. When category\_id is 4, return 'Keyboard'

Once you have the CASE working, update query to sort by category\_name price ascending

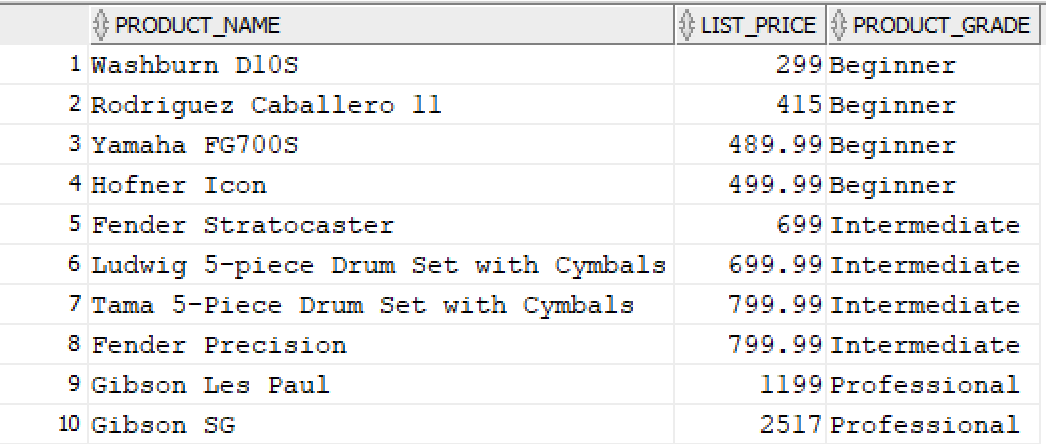
Results should look something like this:



**CASE Function – Search**

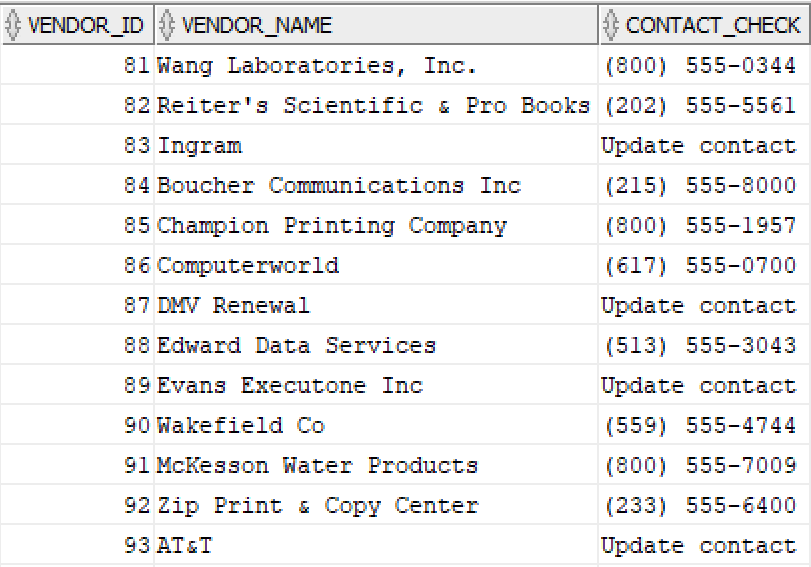
1. Select the product\_name and list\_price from products. Add a third column that uses a CASE that makes the following evaluations: *Hint: You will need to use a search Case so that slightly changes the syntax.*
   1. When list\_price >= 1000, return 'Professional'
   2. When list\_price >= 500, return 'Intermediate'
   3. Everything else, return 'Beginner'

Results should look something like this:

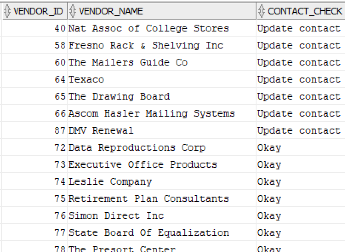


**Handling Nulls**

1. Pull a report that checks vendor contact is up-to-date like so. *Hint: Use NVL for this*

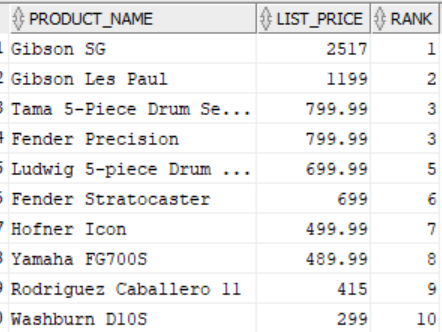


1. Make a copy of the previous query and update it to use the NVL2 function to show not null values as ‘Okay’ and then sort by contact\_check. Results will look something like this



**RANK and DENSE RANK**

1. Pull a list of products (name and list price) and their rank by list\_price descending. Results should look like this for RANK.



NOTE: Try using DENSE\_RANK and discuss the difference with RANK.

**Row Number**

1. Try running the query below:

SELECT ROW\_NUMBER() OVER(ORDER BY vendor\_name)

AS row\_number, vendor\_name

FROM vendors;

After you run the query make this query output an in-line view. That means I want you to make the query a subquery. i.e. SELECT \* FROM (subquery)

**If you get done early…you can either help out your neighbors OR work on extra practice below**

1. Without using the slides to help you, update the code to use the ANSI standard CAST function to convert the data in the same way it’s currently doing it

**SELECT invoice\_id,**

**invoice\_date,**

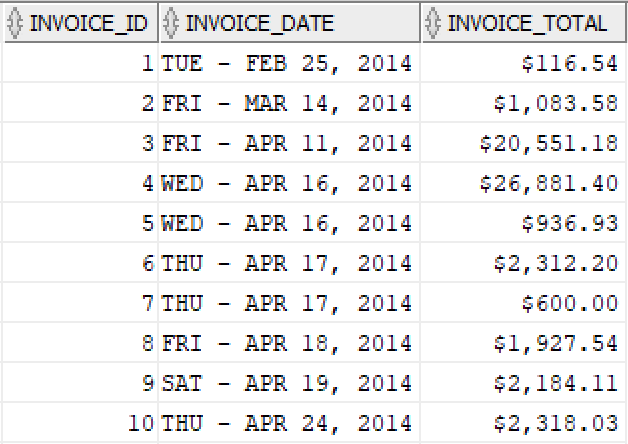
**invoice\_total,**

**to\_char(invoice\_date) AS varchar\_date,**

**to\_number(invoice\_total) AS integer\_total**

**FROM invoices;**

1. Write a query that returns an output like this

****

1. Copy/paste and run the statement below and then format according to instructions below

**select SYSDATE as current\_hour,**

**SYSDATE as hours\_left**

**from dual;**

* 1. Update current\_hour column to return only the current hour using the HH24 mask
  2. Update the hours\_left column to return the hours left in the day