**There are five workouts**

**To be completed before Sunday 30-Jan-2021**

**Workout-5 (Postgresql DVDRENTAL Database)**

**Problem Statement**: Write a function which will accept two inputs and return a grade depending on the inputs provided with the below criteria.

**Function Name**: calculate\_grade

**Inputs:**

1. rate numeric
2. cost numeric

**Logic:**

|  |  |  |
| --- | --- | --- |
| value of "rate" | value of "cost" | grade \* |
| less than 1.00 | less than 10 | A |
| less than 1.00 | 10 or more but less than 20 | B |
| less than 1.00 | 20 or more | C |
| 1 or more but less than 4 | less than 12 | D |
| 1 or more but less than 4 | 12 or more but less than 24 | E |
| 1 or more but less than 4 | 24 or more | F |
| 4 or more | less than 13.5 | G |
| 4 or more | 13.5 or more but less than 19.2 | H |
| 4 or more | 19.2 or more | K |

\* A is minimum and K is maximum grade

**Workout-6 (Postgresql DVDRENTAL Database)**

**Problem Statement**: Write a procedure which will populate a target table for all actor information stated below.

**Target table name:** actor\_performance

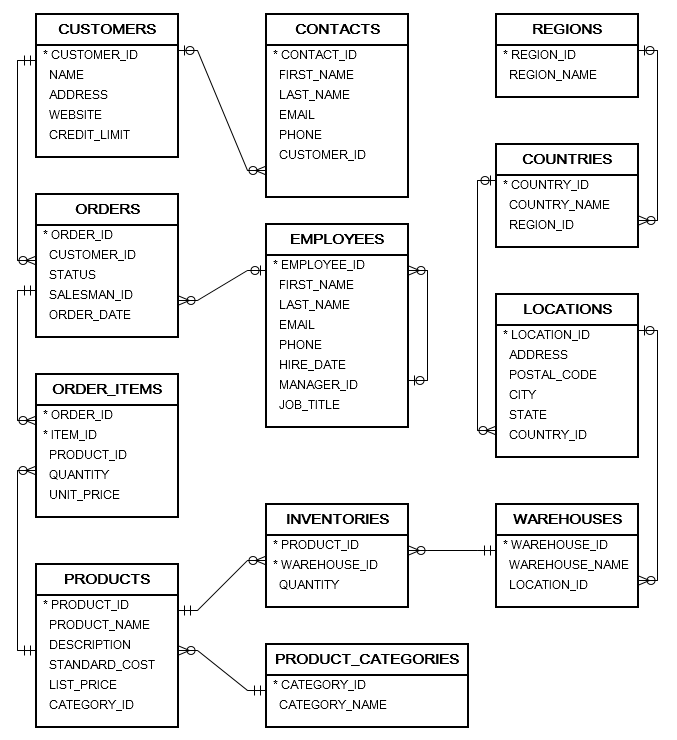
**Target table columns:**

|  |  |  |
| --- | --- | --- |
| column name | column type | description |
| actor\_id | integer | actor\_id from actor table |
| actor\_name | character varying | concatenation of first\_name and last\_name from actor |
| max\_grade | character(1) | any value from A to K |
| min\_grade | character(1) | any value from A to K |
| ratings | character varying | comma separated ratings from film table |

**Hint**: Use tables – actor, film\_actor, film

For each actor, find all the films done by that actor and then find all the grades from the above function, *calculate\_grade*, using rental\_rate and replacement\_cost. Between the grades, A is the min grade and K is the maximum grade. Use these values to store min\_grade and max\_grade columns of *actor\_performance* table. Also find the comma separated values of rating from *film* table per actor.

**Use the below schema diagram for better understanding of oracle table associations.**



**Workout-7 (Oracle Sample Data, Oracle database)**

**Problem Statement**: Write a function which take input two values and return a string.

**Function name:** get\_assesment

**Input:** amount numeric, ranges numeric

**Output:** Function will take the amount and ranges as inputs and find those amounts in list\_price of products table such that list\_price lies between amount+ranges and amount-ranges. After finding the rows from products table, it will find the category\_names using category\_id along with company names from product\_name (use 1st word). Concatenate the category\_name and company name using "@" sign [category\_name@ company name]. All these strings will be concatenated using "OR" and return the value.

**Hint**: Use tables – products, product\_categories,

**Workout-8 (Oracle Sample Data, Oracle database)**

**Problem Statement**: Write a procedure to populate a table having columns customer\_id, customer\_name and purchase\_posibilities varchar2(4000).

**Hint**: Use tables – customers,orders, order\_items,products

**Rule to fillup purchase\_posibilities:** find all the purchases per customer. Find average of standard\_cost and average of list\_price. Send average of standard\_cost as amount and difference of averages between standard\_cost and list\_price as ranges in the above function. The output of the function will be the purchase\_posibilities.

**Workout-9 Design Database**

Consider a Medicine shop management system. System must have the below capabilities:

1. Keep medicine information.
2. Keep seller information.
3. Keep purchase information by which it can increase stock.
4. Sell medicine and keep information for sell along with seller and purchaser.
5. Able to provide current stock of all medicines.
6. Can produce purchase report for a specified period time.
7. Able to find Moving Averages.

Try to design the required tables along with columns which you may find best suitable from your practical experience.

Try to define table constraints (PK, FK, UNIQUE, CHECK) which may be imposed to check bad data entry.

Try to guess what indexes are required to build for faster queries.

Try to define some function for getting current stock.

Try to define procedure for outputs (output tables may also be defined).

Some queries:

* select all medicine sold by any seller for a period of time.
* select all medicine sold in a period of time.
* select whine is sold very frequently and sold maximum in a period of time.
* find voucher by buyer mobile number or name.
* find which medicines are expired and about to expire in some days.
* find which seller sold maximum medicines in a period of time.
* find profit in a period of time.