**A3. PL/pgSQL. Procediments emmagatzemats i triggers.**

**Escriviu el codi PL/pgSQL per a la resolució de cada un dels exercicis següents. Per a cada exercici heu de mostrar una captura de pantalla demostrant l’execució del bloc anònim i el seu resultat.**

**La pràctica s’ha de fer des de el pgAdmin4.**

L’elecció del bloc d’exercicis es realitzarà en funció del següents càlcul:

D són els dos darrers números del DNI de l’alumne.

A són els dos darrers números del vostre any de naixement.

M és el número que correspon al mes de la vostra data de naixement

N = (D+A+M) mod 3

|  |  |
| --- | --- |
| N | BLOC |
| 0 | A |
| 1 | B |
| 2 | C |

**És molt important que especifiqueu el vostre DNI i la fórmula mitjançant la que heu obtingut el número d’exercici a resoldre.**

Índex

[BLOC A. 1](#_Toc112333646)

[BLOC B. 4](#_Toc112333647)

[BLOC C 7](#_Toc112333648)

# BLOC A.

**BDD HR**

**Exercici 1**. ( 1 punt ) Fes un procediment emmagatzemat que rebi com a paràmetre d’entrada un codi d’empleat i posi el nom en un paràmetre de sortida.

**Exercici 2.** ( 2 punts )Fes un procediment emmagatzemat que rebi com a paràmetre d’entrada un codi d’empleat i mostri per consola la següent informació:

*Salari: <salari\_de\_l’empleat>*

A continuació ha de mostrar la següent informació en funció del salari de l’empleat,

* si el salari és 0 euros llavors ha de mostrar: *Treballa gratis!*
* si el salari és < 5000 euros llavors ha de mostrar: *És un becari.*
* Si el salari és < 10000 euros llavors ha de mostrar: *Està en la mitja.*
* Si el salari és superior o igual a 10000 euros llavors ha de mostrar: *Cobra massa !*

Exemple: Per a l’empleat 124 hauria de mostrar:

Salari: 5800

Està en la mitja.

**Exercici 3.** ( 2 punts ) Fes un procediment emmagatzemat que rebi com a paràmetres un número de departament, i un import i incrementi el salari de tots els empleats del departament indicat, amb el salari que tenien més l’import indicat.

**Exercici 4**. ( 1,5 punts ) Crear una funció que rebi com a paràmetre un salari i que retorni calculat el percentatge de la comissió que li aplicaríem en funció d’aquest salari:

- Si el salari és menor o igual a 3000, la comissió que s'aplicarà serà d'un 0.15

- Si la quantitat es troba entre 3001 i 7000, s'aplicarà un 0.1

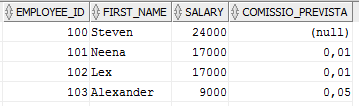
- Si el salari es troba entre 7001 i 10000, s'aplicarà un 0.05

- Si el salari es troba entre 10001 i 20000, s'aplicarà un 0.01

- Si el salari és major de 20000 no té comissió (és NULL).

La funció només retornarà un valor en concret, però no modificarà res de la base de dades. Aquesta funció s'anomenarà **f\_calcularComissio**. Utilitza un CASE per calcular el valor.

Fes un SELECT de la taula d’empleats on utilitzis aquesta funció i mostris quina comissió tindria cada empleat en funció del seu salari. Mostra el codi d’empleat, nom, salari i la comissió prevista.



Etc...

**Exercici 5.** (0,5 punts) Implementa el control d’excepcions en els següents exercicis:

* Exercici 1: Si no es troba l’empleat retorna una cadena buida ‘ ’
* Exercicis 2 i 3: Si no es troba l’empleat escriure un missatge indicant-ho.
* Exercici 3: Si l’operació d’actualització falla retorna un missatge d’error indicant-ho.

**Exercici 6.** (1 punts)

Volem fer una taula d’auditoria per saber si un usuari ha fet alguna modificació sobre la taula DEPARTMENTS (ha afegit, eliminat o modificat alguna cosa).

Per fer-ho farem la següent taula:

Taula: Auditoria\_Departments

|  |  |  |
| --- | --- | --- |
| **ATRIBUT** | **TIPUS** | **DESCRIPCIO** |
| usuari | VARCHAR2(30) | Usuari que ha fet el UPDATE |
| data\_hora | TIMESTAMP | Data-Hora en que es fa el UPDATE |
| operacio | CHAR(1) | I: Insert; U: Update; D: Delete |

Fes un únic trigger per registrar el que es demana. ( El trigger només s’ha de disparar una vegada, encara que les operacions afectin a més d’una fila)

**Exercici 7.** (2 punts)

Executa les dues sentències següents:

**alter table employees**

**add column num\_dependents numeric default 0;**

**update employees e**

**set num\_dependents = (select count(\*)**

**from dependents d**

**where d.employee\_id = e.employee\_id);**

Crear un trigger que mantingui actualitzat el camp num\_dependents de la taula employees automàticament en cas de inserir, modificar o eliminar un fill/a (dependent).

Heu de tenir en compte que:

* Si inserim un nou fill/a s’ha de sumar a la columna num\_dependents.
* Si eliminen un fill/a s’ha de restar de la columna num\_dependents.
* Si modifiquem el camp employee\_id d’un fill/a ... (pensa-ho).

# BLOC B.

**BDD PAGILA**

**Exercici 1**. ( 1 punt ) Fes un procediment emmagatzemat que rebi com a paràmetre d’entrada un codi de client i posi el nom en un paràmetre de sortida.

**Exercici 2.** ( 2 punts )Fes un procediment emmagatzemat que rebi com a paràmetre d’entrada un codi d’actor i mostri per consola la següent informació:

*Pel·lícules on ha participat: <numero\_de\_pel·lícules>*

A continuació ha de mostrar la següent informació en funció del número de pel·lícules en que ha participat:

* si el número de pel·lícules és 1 llavors ha de mostrar: *És un becari.*
* si el número de pel·lícules és <= 3 llavors ha de mostrar: *És un actor de repartiment.*
* Si el número de pel·lícules és <= 5 llavors ha de mostrar: *És un secundari.*
* Si el número de pel·lícules és superior a 5 llavors ha de mostrar: *El protagonista de la peli !*

Exemple: Per a l’actor 186 hauria de mostrar:

Pel·lícules on ha participat: 2

És un actor de repartiment.

**Exercici 3.** ( 2 punts ) Fes un procediment emmagatzemat que rebi com a paràmetres un identificador de categoria, i un número de dies i incrementi el temps de lloguer (rental\_duration) de totes les pel·lícules de la categoria indicada, amb el temps de lloguer que tenen més el número de dies indicat.

**Exercici 4**. ( 1,5 punts ) Crear una funció que rebi com a paràmetre un preu de lloguer i el cost de substitució i que retorni calculat el cost de substitució (amb dos decimals) en funció del preu de lloguer aplicant un factor de correcció segons s’indica a continuació:

- Si el preu de lloguer és 0.99, el cost de substitució serà multiplicat per 1,1

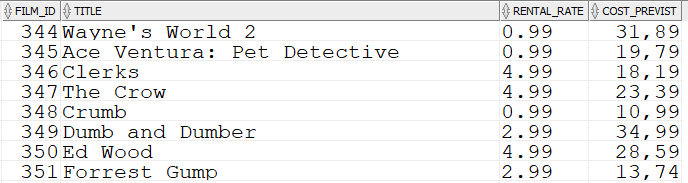
- Si el preu de lloguer és 2.99, el cost de substitució serà multiplicat per 1,25

- Si el preu de lloguer és 4,99, el cost de substitució serà multiplicat per 1,3

- En qualsevol altre cas el cost de substitució serà NULL.

La funció només retornarà un valor en concret, però no modificarà res de la base de dades. Aquesta funció s'anomenarà **f\_calcularCost**. Utilitza un CASE per calcular el valor.

Fes un SELECT de la taula film on utilitzis aquesta funció i mostris quin cost de substitució tindria cada pel·lícula en funció del preu de lloguer. Mostra el codi de pel·lícula, títol, preu del lloguer i el cost de substitució previst.



Etc...

**Exercici 5.** (0,5 punts) Implementa el control d’excepcions en els següents exercicis:

* Exercici 1: Si no es troba el client retorna una cadena buida ‘ ’
* Exercicis 2 i 3: Si no es troba l’actor o la categoria escriure un missatge indicant-ho.
* Exercici 3: Si l’operació d’actualització falla retorna un missatge d’error indicant-ho.

**Exercici 6.** (1 punts)

Volem fer una auditoria sobre el camp rental\_rate de la taula FILM de manera que sapiguem si un usuari ha fet modificacions sobre aquest camp.

Per fer-ho, volem crearem una taula com la següent

Taula: Auditoria\_EmplSalari

|  |  |  |
| --- | --- | --- |
| **ATRIBUT** | **TIPUS** | **DESCRIPCIO** |
| usuari | VARCHAR2(30) | Usuari que ha fet el UPDATE |
| data\_hora | TIMESTAMP | Data-Hora en que es fa el UPDATE |
| filmid | NUMBER | Película sobre la que s’ha fet UPDATE |
| rentalrate\_old | NUMBER | Preu lloguer anterior |
| rentalrate\_new | NUMBER | Preu lloguer actual |

\*usuari + data Hora = clau primària

Crea un trigger que enregistri l’usuari que ha modificat el salari de l’empleat

**Ajuda:**

Per enregistrar l’usuari que ha fet la operació, utilitzeu la variable de sistema **CURRENT\_USER**:

insert into table\_name (usuari) values (**current\_user**)

**Exercici 7.** (2 punts)

Executa les dues sentències següents:

**alter table film**

**add column stock numeric default 0;**

**update film f**

**set stock = (select count(\*)**

**from inventory i**

**where i.film\_id = f.film\_id);**

Crear un trigger que mantingui actualitzat el camp stock de la taula film automàticament en cas de inserir, modificar o eliminar una còpia (inventory).

Heu de tenir en compte que:

* Si inserim una nova còpia (inventory\_id) s’ha de sumar a la columna stock.
* Si eliminen una còpia existent (inventory\_id) ha de restar de la columna stock.
* Si modifiquem el camp film\_id d’una còpia ... (pensa-ho).

# BLOC C

**BDD ADVENTUREWORKS**

**Exercici 1**. ( 1 punt ) Fes un procediment emmagatzemat que rebi com a paràmetre d’entrada un codi de persona (taula person de l’esquema person) i posi el nom complert (firstname, middlename, lastname) en un paràmetre de sortida.

**Exercici 2.** ( 2 punts )Fes un procediment emmagatzemat que rebi com a paràmetre d’entrada un codi d’un producte (taula product de l’esquema production) i mostri per consola la següent informació:

*Fabricacions: <numero\_de\_fabricacions>*

El *número de fabricacions* sortirà de les fulles de treball acabades (taula workorder de l’esquema production, amb la condició que el camp *duedate* estigui informat, ja que indica la data que s’ha acabat la fabricació)

A continuació ha de mostrar la següent informació en funció del número de fabricacions:

* si el número de fabricacions és < 50 llavors ha de mostrar: *Baixa producció.*
* si el número de pel·lícules és >=50 i <100 llavors ha de mostrar: *Producció ocasional.*
* Si el número de pel·lícules és >= 100 i <= 500 llavors ha de mostrar: *Producció estàndard.*
* Si el número de pel·lícules és superior a 500 llavors ha de mostrar: *Alta producció.*

Exemple: Per al producte 750 hauria de mostrar:

Fabricacions: 206

Producció estàndard.

**Exercici 3.** ( 2 punts ) Fes un procediment emmagatzemat que rebi com a paràmetres un codi d’oferta (taula specialoffer de l’esquema sales), i un número d’unitats i incrementi la quantitat màxima que s’ha de comprar (maxqty) de totes les ofertes que tenen maxqty diferent de 0, amb la quantitat que tenen més el número d’unitats.

**Exercici 4**. ( 2,5 punts Crear una funció que rebi com a paràmetre una quota de vendes i que retorni calculat el percentatge de la comissió que li aplicaríem en funció d’aquestes vendes:

- Si les vendes són menors a 1 milió, no s’aplicarà comissió (NULL)

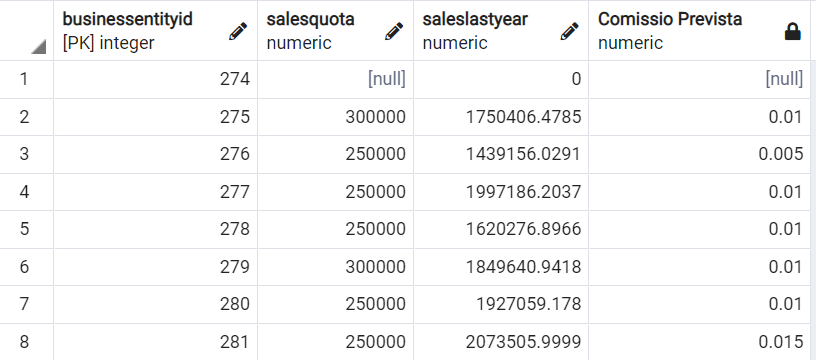
- Si les vendes es troben entre 1 milió +1 i 1 milió i mig, s'aplicarà un 0.005

- Si les vendes es troben entre 1 milió i mig +1 i 2 milions, s'aplicarà un 0.01

- Si les vendes són majors que 2 milions +1 llavors s’aplicarà un 0.015

La funció només retornarà un valor en concret, però no modificarà res de la base de dades. Aquesta funció s'anomenarà **f\_calcularComissio**. Utilitza un CASE per calcular el valor.

Fes un SELECT de la taula *sales.salesperson* on utilitzis aquesta funció i mostris quina comissió tindria cada venedor en funció de les vendes registrades l’últim any. Mostra el codi de venedor, quota de vendes, vendes de l’últim any i la comissió prevista.



Etc...

**Exercici 5.** (0,5 punts) Implementa el control d’excepcions en els següents exercicis:

* Exercici 1: Si no es troba la persona retorna una cadena buida ‘ ’
* Exercicis 2 i 3: Si no es troba el producte o l’oferta escriure un missatge indicant-ho.
* Exercici 3: Si l’operació d’actualització falla retorna un missatge d’error indicant-ho.

**Exercici 6.** (1 punts)

Volem fer una auditoria sobre el camp standardprice de la taula *purchasing.productvendor* de manera que sapiguem si un usuari ha fet modificacions sobre aquest camp.

Per fer-ho, volem crearem una taula com la següent

Taula: Auditoria\_ProdVendor

|  |  |  |
| --- | --- | --- |
| **ATRIBUT** | **TIPUS** | **DESCRIPCIO** |
| usuari | VARCHAR2(30) | Usuari que ha fet el UPDATE |
| data\_hora | TIMESTAMP | Data-Hora en que es fa el UPDATE |
| Product\_id | NUMBER | Producte |
| Vendor\_id | NUMBER | Vendor |
| standardprice\_old | NUMBER | Standardprice anterior |
| standardprice\_new | NUMBER | Standardprice actual |

\*usuari + data Hora = clau primària

Crea un trigger que enregistri l’usuari que ha modificat el salari de l’empleat

**Ajuda:**

Per enregistrar l’usuari que ha fet la operació, utilitzeu la variable de sistema **CURRENT\_USER**:

insert into table\_name (usuari) values (**current\_user**)

**Exercici 7.** (2 punts)

Volem crear un trigger que mantingui actualitzat el camp standardcost de la taula product automàticament, en cas de un nou preu estàndard s’ha d’actualitzar la columna standardcost sempre i quan la data d’inici del preu (startdate) sigui anterior o igual a la data del sistema.