Universitat Oberta de Catalunya

uoc.edu

Disseny i implementació de la base de dades per a una aplicació de planificació de producció

Adrián Merinas Diez
Grau en Enginyeria Informàtica
Bases de dades

Jordi Ferrer Duran
Cristina Pérez Solà

10 de Juny de 2019

FITXA DEL TREBALL FINAL

Títol del treball:	*Disseny i implementació de la base de dades per a una aplicació de planificació de producció*
Nom de l'autor:	*Adrián Merinas Diez*
Nom del consultor/a:	*Jordi Ferrer Duran*
Nom del PRA:	*Cristina Pérez Solà*
Data de lliurament (mm/aaaa):	*06/2019*
Titulació o programa:	*Grau en Enginyeria Informàtica*
Àrea del Treball Final:	*Bases de dades*
Idioma del treball:	*Català*
Paraules clau	*base de dades (database), cadena de subministrament (supply chain), magatzem de dades (data warehouse)*

Resum del Treball (màxim 250 paraules): *Amb la finalitat, context d'aplicació, metodologia, resultats i conclusions del treball*

Aquest treball dóna resposta a les necessitats de l'empresa de producció i distribució de begudes emergent CAT-Cola que ha decidit simplificar i agilitzar els seus complexos processos en la seva cadena de subministrament i així millorar la seva posició i ser més competitiva en el seu mercat objectiu.

Amb aquest objectiu, el treball analitza els requisits del client, n'identifica els punts clau, transcorre les diferents etapes fonamentals de disseny d'una base de dades, tria el sistema gestor de base de dades més conveninet i elabora la documentació associada a tot aquest procés.

El resultat d'aquesta implementació es composa en: un script de creació de base de dades i inicialització de dades de prova, una memòria amb tota la metodologia seguida i el seu desenvolupament i uns annexos i scripts associats comentats amb el codi escrit.

Durant el transcurs del treball s'han fonamentat els coneixements adquirits d'un gran rang d'assignatures. S'han après nous processos interns sobre tecnologies utilitzades i ha permès trobar nous enfocs en el disseny de base de dades.

Abstract (in English, 250 words or less):

This undergraduate dissertation answers the needs of CAT-Cola, an emergent beverage production and Distribution company that decided to simplify and streamline its complex processes in its supply chain and thus, improve its position and become more competitive in its target market.

Consequently, the dissertation analyses the client's requirements, identifies the key points, goes through the different fundamental stages of data base design, chooses the most convenient data base management system and develops the associated documentation associated with this process.

The result of this implementation is composed of a data base creation script and its initialization of test data, a memory with the methodology followed and its development, annexes and associated commented scripts with the written code.

The knowledge acquired thorough a wide range of subjects during the undergraduate program has provided the foundation for this dissertation. New internal processes on applied technologies have been learnt and it has allowed finding new approaches in the setting of database design.

Índex

Llista de figures

1. Introducció

1.1 Context i justificació del Treball

El treball es basa en una hipotètica empresa emergent en el sector de la producció i distribució de begudes anomenada CAT-Cola.

CAT-Cola, que necessita guanyar quota del mercat a la seva competència, ha determinat que necessita una aplicació de planificació de producció per optimitzar els seus processos de producció.

Aquest programari permetrà a l'empresa simplificar i agilitzar els seus complexos processos que es troben dins la seva cadena de subministrament, i així, millorar la seva competència dins del seu mercat.

Amb aquest objectiu, el treball es centrarà, exclusivament, en dissenyar i implementar la base de dades que utilitzarà aquesta aplicació de planificació.

Al final del treball, i a partir de l'enunciat que se'ns ha proveït, es recollirà i analitzaran els requisits, s'elaborarà un disseny conceptual, un disseny lògic i un disseny físic on s'implementarà i s'optimitzarà sobre un SGBD (Sistema Gestor de Bases de Dades).

1.2 Objectius del Treball

- Identificar i analitzar els requisits que ens demana la companyia CAT-Cola a partir dels enunciats de les PACs.
- Elaborar el disseny conceptual de la base de dades amb el recolzament d'un llenguatge de modelització.
- Elaborar el disseny lògic de la base de dades a partir del disseny conceptual.
- Triar el SGBD més convenient a la nostra situació.
- Elaborar el disseny físic on s'implementarà el disseny lògic al SGBD triat.
- Inicialitzar un conjunt de dades que s'adaptin a la implementació i optimitzar les consultes i processos segons les necessitats del client.
- Generar i entregar la documentació necessària.

1.3 Enfocament i mètode seguit

En el mercat actual, qualsevol nova empresa podria optar per implementar un ERP (Planificador de Recursos Empresarials) que ja es trobi consensuat com, per exemple, un de programari lliure com és Odoo (antic OpenERP) o d'altres de programari propietari com pot ser SAP.

Tot i així, per poder operar aquests ERPs es necessiten experts que coneguin la seva estructura interna i que siguin capaços d'adaptar-los a les necessitats del client.

El nostre client, només necessita un mòdul concret del ERP relacionat amb la cadena de subministrament i, tot i que podria aquest mòdul individual i adaptar-lo d'algun ERP, el temps i el cost per comprar la llicència, llogar els seus serveis i/o trobar experts que dominin el ERP concret triat pot ser major que crear el programari des de bon inici.

Per altra banda, la creació d'un programari personalitzat per una empresa en els seus estadis inicials permet que aquesta tingui major rang d'opcions a l'hora de triar les tecnologies que més li convinguin.

Consegüentment, i com que el treball s'ha de basar en els coneixements adquirits en el Grau, s'ha decidit per desenvolupar un producte nou de base utilitzant un SGBD i els principis teòrics que ens ha proveït els estudis.

1.4 Planificació del Treball

Per a la planificació del Treball he tingut en compte les fases en què consta el disseny de qualsevol base de dades. A la [Figura 2] es desglossa el Treball en un seguit de tasques i es defineixen els objectius de cadascuna per cada entrega.

Aquestes tasques es troben prioritzades en una escala de colors tal i com apareix a la [Figura 1].

Prioritat
Baixa
Mitjana
Alta

Figura 1 Correspondència de Colors amb la seva prioritat

Tasca	Descripció
PAC 1: Pla de Treball	Es recullen i s'analitzen els requisits del treball per obtenir uns objectius concrets i planificar-los.
Recollida i anàlisis de Requisits	Es parla amb el client per trobar les seves necessitats i s'analitzen per extreure els objectius.
Objectius i Planificació	Obtinguts els objectius s'encapsulen en tasques concretes i es planifiquen en el calendari i amb un cost d'hores.
PAC 2: Disseny Conceptual	Es basa en conceptualitzar els requisits del client i generar un esquema d'alt nivell amb aquestes especificacions.
Conceptualització	Com que el disseny conceptual es basarà en un model Entitat-Relació (ER), la conceptualització extraurà aquells conceptes que s'utilitzin per dissenyar aquest diagrama.
Diagrama UML, ER	Disseny d'un diagrama d'alt nivell amb l'ajuda d'un llenguatge unificat de modelització (UML). En el nostre cas, utilitzarem el model relacional.

Documentació PAC 2	La documentació és un procés progressiu, com a tal, es portarà a terme en conjunt amb les altres tasques de l'entrega. S'explicaran les decisions i l'atomització que s'ha pres a l'hora de crear el diagrama i triar les entitats i relacions.
PAC 3: Disseny Lògic i Físic	El disseny conceptual es transforma a un disseny lògic dependent del tipus de tecnologia (en aquest cas, una base de dades relacional). Posteriorment, s'implementa en el disseny físic al SGBD triat.
Traducció a Taules i Columnes	Es tria la tecnologia de base de dades i es transforma el disseny conceptual en un disseny lògic que es basarà en tecnologies de base de dades relacionals.
Tria del SGBD	Un cop triem la tecnologia, escollim quin és el Sistema Gestor de Base de Dades que més ens convé, basat en aquesta tecnologia.
Scripts de Taules al SGBD	Escriptura del codi relacionat amb la creació de taules provinents de les Entitats de la fase anterior.
Scripts CRUD al SGBD	Escriptura del codi relacionat amb la creació, lectura, actualització i eliminació de dades de les taules.
Documentació PAC 3	S'adjuntarà el codi generat i el resultat del pas entre dissenys.
Lliurament Final: Optimització	Aquesta última fase es procedeix a fer els últims ajustos per millorar el rendiment de la base de dades i preparar la documentació final del Treball.
Scripts de Inicialització de dades	Són els scripts necessaris per poder inicialitzar la base de dades amb informació rellevant pel client.
Scripts de Procediments	És el codi de creació de procediments que facilitarà el manteniment de les dades de les taules. Creació, lectura, actualització i eliminació de les dades emmagatzemades a les taules.
Scripts de Consultes	Aquests scripts s'utilitzaran per provar les consultes més demanades pel client i optimitzar-les.
Scripts de Índexs i altres	És altre codi necessari per l'optimització de la base de dades.
Revisió i Presentació del Treball	Revisió de tot el que s'ha fet i creació de la presentació del Treball pel Tribunal.
Documentació Lliurament Final	Codi d'optimització de la base de dades, producte final i memòria.
Tribunal d'Avaluació	Preparar les respostes del Tribunal i publicar aquest al repositori institucional de la UOC.

Figura 2 Tasques en què consta el Treball i la seva descripció.

Pel que concerneix als recursos necessaris per portar a terme el treball, aquests es troben explicats a la [Figura 3].

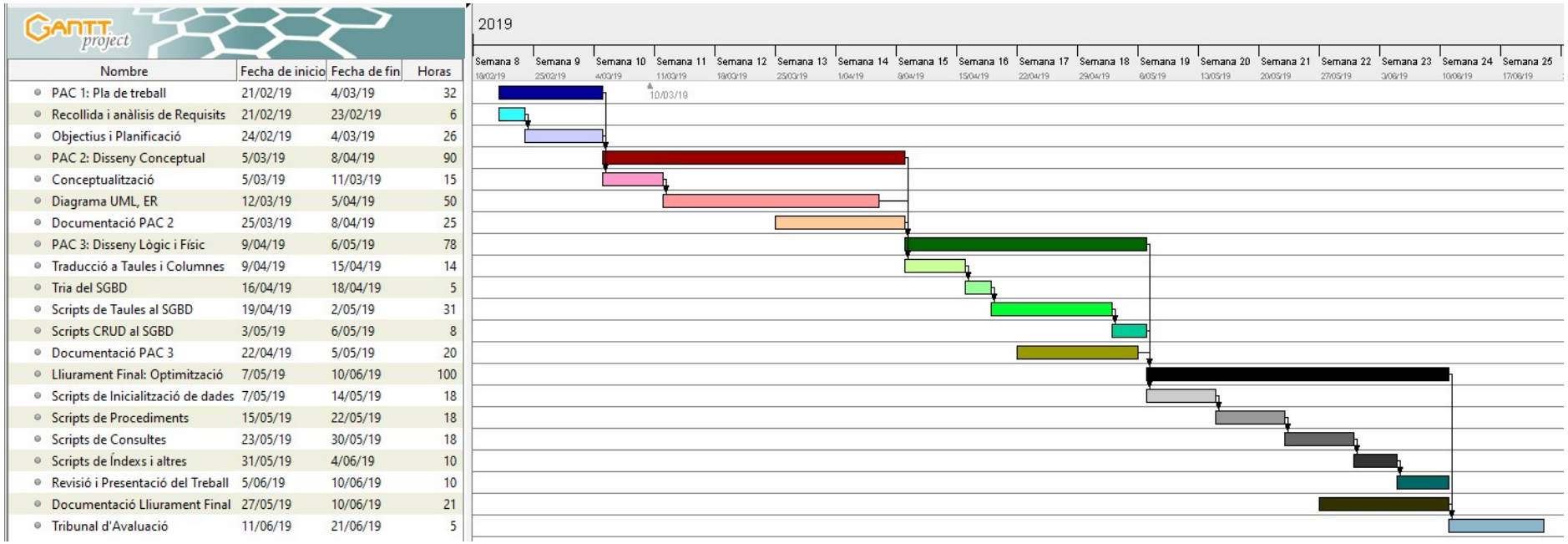
Recurs	Descripció
Programari de gestió de projectes	Programari per dissenyar un diagrama de Gantt
Eina de modelatge relacional	Programari de modelatge utilitzant UML per fer el disseny conceptual de la base de dades
SGBD	Sistema Gestor de Base de Dades que s'utilitzarà per implementar la base de dades
Servidor	Faré servir el meu propi ordinador que farà de Servidor on es trobarà instal·lat el SGBD

Figura 3 Llistat de Recursos necessaris pel Treball.

A la [Figura 4] es troba detallada la planificació de les tasques en funció a l'espai temporal des de la data d'inici fins a la data final del treball. També es troben especificades les hores necessàries per completar cada tasca.

S'ha de tenir en compte que les tasques PAC 1, PAC 2, PAC 3 i Lliurament Final es tracten de la suma de les subtasques que es troben sota seu. La dedicació total d'aquestes tasques resulta en 300 hores de treball.

L'exclusió dels dies festius és degut a que es tracta d'un treball acadèmic i, on més temps es disposarà per treballar-hi és, precisament, els dies festius. Si es tractés d'un projecte empresarial, s'haurien de tenir en compte els dies festius del calendari on els treballadors tindrien festiu o només es trobessin serveis mínims.



Nombre	Fecha de inicio	Fecha de fin	Horas
PAC 1: Pla de treball	21/02/19	4/03/19	32
Recollida i anàlisis de Requisits	21/02/19	23/02/19	6
Objectius i Planificació	24/02/19	4/03/19	26
PAC 2: Disseny Conceptual	5/03/19	8/04/19	90
Conceptualització	5/03/19	11/03/19	15
Diagrama UML, ER	12/03/19	5/04/19	50
Documentació PAC 2	25/03/19	8/04/19	25
PAC 3: Disseny Lògic i Físic	9/04/19	6/05/19	78
Traducció a Taules i Columnes	9/04/19	15/04/19	14
Tria del SGBD	16/04/19	18/04/19	5
Scripts de Taules al SGBD	19/04/19	2/05/19	31
Scripts CRUD al SGBD	3/05/19	6/05/19	8
Documentació PAC 3	22/04/19	5/05/19	20
Lliurament Final: Optimització	7/05/19	10/06/19	100
Scripts de Inicialització de dades	7/05/19	14/05/19	18
Scripts de Procediments	15/05/19	22/05/19	18
Scripts de Consultes	23/05/19	30/05/19	18
Scripts de Índexs i altres	31/05/19	4/06/19	10
Revisió i Presentació del Treball	5/06/19	10/06/19	10
Documentació Lliurament Final	27/05/19	10/06/19	21
Tribunal d'Avaluació	11/06/19	21/06/19	5

Figura 4 Diagrama de Gantt de la planificació del Treball utilitzant el programari GanttProject.

1.5 Breu sumari de productes obtinguts

Els productes resultants del desenvolupament d'aquest treball composen per:

- Base de dades: Script de creació i inicialització de la base de dades generada com a solució al problema que s'ha plantejat (bdd_export.sql).
- Mèmoria del Treball: On es troba explicada la metodologia seguida, el desenvolupament d'aquest, la seva composició, els inconvenients més significatius i les solucions aplicades.
- Annexos i scripts associats: Els annexos es troben formats per tot el conjunt de scripts programats a mà amb comentaris.

1.6 Breu descripció dels altres capítols de la memòria

- Capítol 1: Introducció – Obtenció dels requisits del client, enfocament i metodologia del treball i la seva planificació.
- Capítol 2: Disseny Conceptual – Identificació d'entitats, traducció a un disseny conceptual relacional i decisions presses.
- Capítol 3: Disseny Lògic – Traducció del disseny conceptual al disseny lògic en forma de taules i normalització d'aquest.
- Capítol 4: Disseny Físic – Elecció del SGBD, creació de l'espai de taules i usuari administrador, creació de scripts de taules.
- Capítol 5: Optimització – Disseny i creació de les consultes demanades pel client, creació de scripts de procediments ABM i auxiliars per un futur ús d'una aplicació, inicialització 'automàtica' de dades i conjunt de proves.
- Capítol 6: Seguiment de la planificació – Explicació del procés de desenvolupament del treball, inconvenients trobats i solucions optades.
- Capítol 7: Conclusions – Resultats d'aprenentatge adquirits durant el treball i millores futures.
- Capítol 8: Glossari – Definició de conceptes utilitzats durant el treball.
- Capítol 9: Bibliografia – Referències bibliogràfiques utilitzades en el transcurs del treball.
- Capítol 10: Annexos – Codi escrit a mà amb comentaris utilitzat per generar la base de dades.

2. Disseny Conceptual

2.1 Preàmbul

Com es comenta a la introducció, el disseny d'una base de dades segueix un seguit de fases preestablertes que ens facilita la seva tasca d'implementació. Aquestes fases també ens permeten evitar possibles errors de disseny que, més endavant, poguessin resultar en una major inversió de recursos per resoldre'ls. [1, pàg. 4]

La primera fase és la de recollida i anàlisis de requisits. Aquesta etapa és dedicada a extreure la informació necessària del client i els grups d'interès sobre els requisits i restriccions que necessita el seu sistema per funcionar correctament. [2, pàg 66-73] Com que el treball es basa en un enunciat, es considerarà que aquesta part es troba resolta ja que aquest conté tota la informació necessària d'aquests requisits.

Conseqüentment, es segueix amb la següent fase que és el disseny conceptual.

En el disseny conceptual s'extreuen els conceptes claus recollits a l'anterior fase i es modela un esquema conceptual utilitzant diagrames UML, ER o relacionals per visualitzar el sistema.

Existeixen moltes eines per modelar aquests esquemes i, tot i que MagicDraw hauria sigut molt bona opció, la seva llicència propietària dificulta el fet d'utilitzar-lo. Aquest programari té una versió de prova però el nombre d'entitats es troba limitat.

Degut a que el treball tindrà un alt nombre d'entitats i es vol adjuntar el diagrama complet, al final s'ha triat una eina en línia gratuïta sense aquestes restriccions anomenada draw.io[1] amb prestacions similars a les ofertes per MagicDraw.

Un cop triada l'eina, ja es pot començar a desenvolupar el diagrama per visualitzar el sistema que es vol implementar pel nostre client.

2.2 Conceptualització d'Entitats i Relacions

En aquest apartat, s'extreuen les Entitats més importants i les Relacions que es donen entre aquestes i es conceptualitzen en un diagrama relacional.

Existeixen diverses metodologies de conceptualització i normalment es tracta d'un procés iteratiu. L'enfocament principal que s'ha seguit a l'hora de trobar les entitats i les seves relacions ha sigut l'estratègia de 'Bottom-up' on s'han trobat primer les taules principals i les relacions més fortes i després les derivades. [3, pàg. 62-67]

[1] Enllaç al recurs online: https://www.draw.io/

Abans de començar, però, s'ha d'aclarir que la idea inicial era dissenyar un ERD seguint les pautes de les assignatures de disseny de base de dades del grau.

Els diagrames utilitzats en aquestes assignatures són una mescla de nomenclatures i diagrames UML, ER i relacionals. En els diagrames UML no existeix el concepte de clau primària i els ERD no existeix el concepte de clau forana. Aquests, són dos models independents de la tecnologia que s'hagi d'aplicar posteriorment i, en els diagrames d'aquestes assignatures es veu clarament que es troben enfocats a bases de dades relacionals.

Tot i que fer un diagrama UML o ER ens permetria, més endavant, utilitzar-ho de base si canviéssim de tecnologia, els canvis portats a terme a la seva implementació probablement el farien irrecognoscible i el més òptim seria començar-lo de nou.

Resumint, com que l'objectiu del treball serà implementar el sistema en una base de dades relacional, s'ha decidit escollir dissenyar un diagrama relacional de base amb terminologia de cardanilitats basada en els diagrames de Chen.[4, pàg. 112] Aquest diagrama s'adapta a les necessitats del treball més convenientment que els altres mencionats, agilitzarà el procés de disseny i és més acurat que els diagrames observats en les assignatures pel nostre objectiu.

El disseny relacional resultant del nostre sistema es pot observar a la [Figura 5].

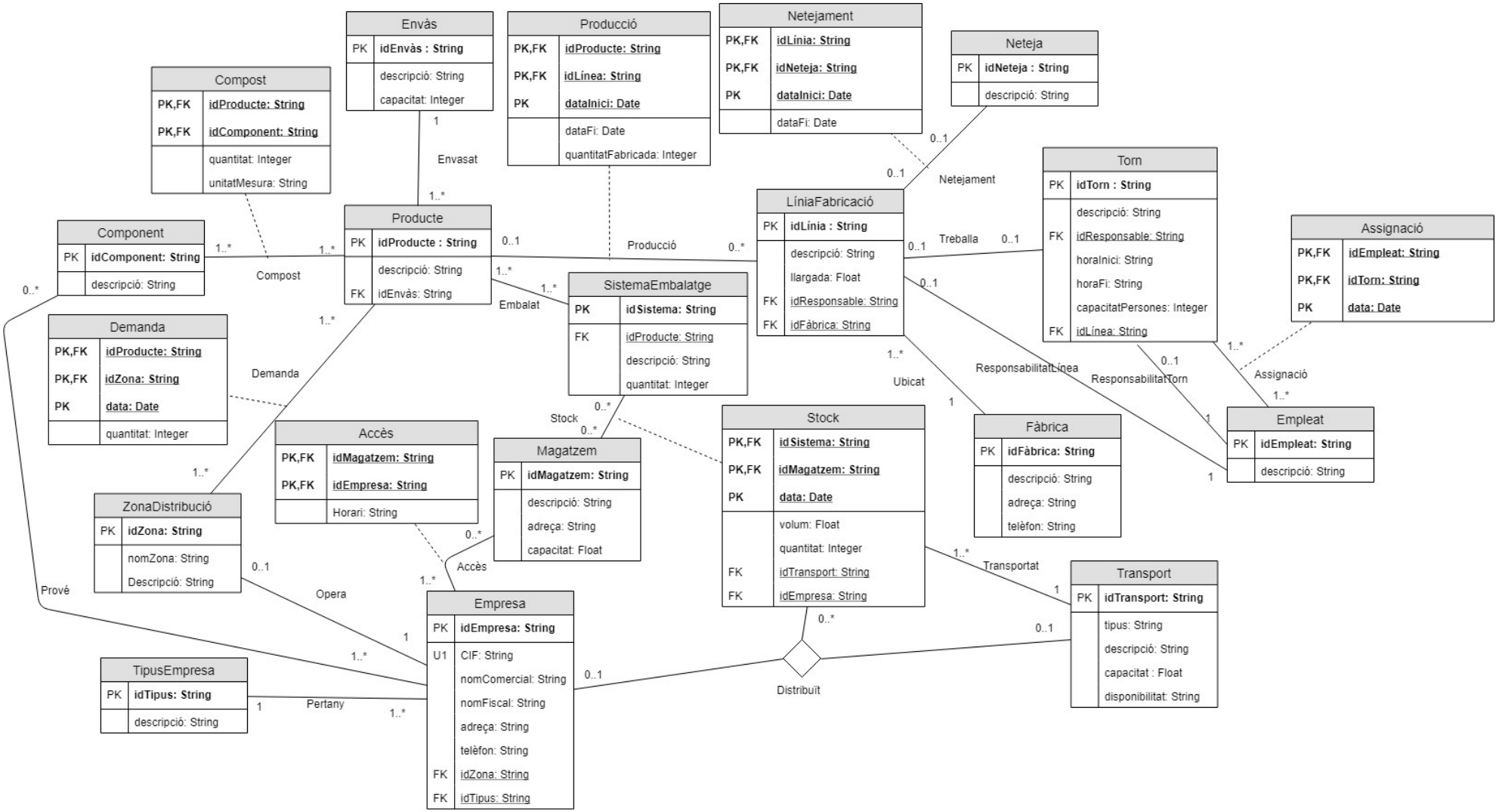


Figura 5 Diagrama relacional del sistema creat amb el recurs online draw.io.

2.3 Decisions de disseny

En aquest apartat s'explicaran les decisions més rellevants per les que s'han optat a l'hora de dissenyar el model relacional del sistema.

Com que el treball es basa en l'aplicació d'aquitectures de magatzem de dades, s'utilitzarà un esquema d'estrella. En aquest tipus d'esquema, les taules principals s'anomenarien taules dimensionals on es tracten dades descriptives dels subjectes de la companyia i les taules que aglutinen claus foranes, aquestes taules s'anomenarien taules factuals que contenen dades quantificades. [5, pàg. 393-395]

Es considera que gran part de les cardinalitats han de ser, com a mínim, de u. Per exemple, en la relació Compost, es considera que un Component ha de tenir un Producte assignat per existir i també ha de ser així a la inversa. Si un Component no s'utilitza en cap Producte, no hi ha raó per guardar-lo a la base de dades. Per tant, el registre no existirà i només es guardaran les dades rellevants.

Per altra banda, no es creu necessari fer una Entitat que fos la Data o fer Entitats amb diversos trossos de Data (dia, mes, any, etc.). Per aquesta raó, existeixen diverses Entitats associatives que tenen com a clau primària una data sense necessitat de fer una relació ternària amb una entitat Data.

El diagrama intenta tenir en compte les consultes més demanades que es troben als requisits del client. Per exemple, per saber el producte més produït durant un any, podem extreure aquesta informació a partir de l'Entitat associativa Producció que indica la quantitat fabricada de Producte en una LíneaFabricació en el període de temps que ha estat en marxa.

Pel que correspon al Producte, es considera que l'Envàs que el conté és el més indicat per tenir l'atribut de capacitat (els mil·lilitres de producte). Com que un Producte pot estar compost de diversos Components, l'Entitat Compost guarda quins són els components, la quantitat per la que es troben formats i la unitat de mesura per si es tracta d'un component sòlid, gasós o líquid.

L'entitat SistemaEmbalatge indica els diferents tipus d'agregació que tindrà cada Producte. Per exemple, un producte que sigui una llauna de llimonada anirà en paquets de vuit llaunes. He considerat que tot els productes passaran per aquest sistema d'embalatge degut a què considero que l'empresa no enviarà o vendrà productes individuals sinó paquets d'aquests.

Si fos un sistema més realista, aquests productes o petites agrupacions s'haurien d'agrupar en pallets i portar un codi de barres per identificar-los. Tot i així, el treball no deixa de ser una simplificació i petita interpretació d'un sistema real.

Per mantenir un registre de la capacitat lliure als magatzems, existeix l'Entitat associativa Stock on es guarda el volum i quantitat de SistemaEmbalatge per

Magatzem. Com he comentat, considero que són els SistemaEmbalatge els que acaben sent les unitats mínimes d'emmagatzematge i transport.

Aquest Stock ha sigut transportat per l'empresa, en un primer moment, per un mètode de Transport, com es veu en la relació Transportat.

Empresa és una sola entitat on es guarden les dades de tot tipus d'empreses (proveïdores, distribuïdores, etc.). Les que són distribuïdores, tenen un horari d'accés que es guarda a l'Entitat associativa Accés.

El Stock emmagatzemat és transportat per una Empresa distribuïdora als punts de venta. La relació Distribuït ens relaciona el Stock (paquets de producte) que una Empresa distribuïdora ha distribuït i mitjançant quin Transport ho ha fet.

Les línies de fabricació s'han de netejar i he considerat que l'Entitat Neteja són grups de treball independents de la LíniaFabricació fins que se'ls assigna una a través de Netejament. Aquestes poden no estar treballant en una LíneaFabricació en un moment donat, però quan ho fan, es guarda a l'Entitat associativa Netejament.

Els Torns són els que treballen a les línies de fabricació i en aquests, se'ls assignen Empleats. He considerat que un Empleat se'l pot assignar diversos Torns en una mateixa data. Per tant, he creat una Entitat associativa Assignació que guarda els Torns que un usuari ha treballat i que també servirà per saber si un Torn es troba ple d'Empleats.

3. Disseny Lògic

3.1 Preàmbul

Com s'explica a l'anterior apartat, a l'hora de realitzar el disseny conceptual s'ha triat fer un diagrama relacional i, per tant, utilitzar el model relacional que és la base de les bases de dades relacionals on s'acabarà implementant.

Tot i que la tria del model seria pròpia d'aquest apartat, les incongruències exposades anteriorment i l'agilitat i facilitats que ha proporcionat la realització del diagrama relacional ha fet que es decantés cap aquesta resolució. Tot i així, dins el disseny lògic encara es troben apartats que no s'han abordat i que es veuran a continuació.

3.2 Entitats, relacions i atributs

En aquest apartat es traduiran totes les entitats, relacions i atributs. En concret, s'escriuran totes les entitats i els seus atributs que, després de la seva normalització, s'implementaran com a taules i columnes al SGBD triat. [1]

Per poder interpretar el significat dels diferents formats dels atributs utilitzarem la [Figura 6]. A l'hora de la seva implementació, hi haurà restriccions addicionals però, de moment, aquestes són les restriccions més importants.

Si un atribut és clau primària i forana, la prioritat serà la simbologia de la primària.

Les entitats que guarden una data d'inici i una de fi suposarem, per simplicitat, que no hi ha un estat intermedi on la data de fi es trobi com a NULL. Per tant, aquests registres es faran un sol cop amb el seu inici i final.

Format	Significat
Atribut	Clau primària
Atribut	Clau forana
Atribut	Valor que pot ser NULL

Figura 6 Format dels atributs de les entitats.

Entitats

Envàs(idEnvàs, descripció, capacitat)

Producte(idProducte, descripció, idEnvàs)
On idEnvàs referencia Envàs (idEnvàs)

Component(idComponent, descripció)

ZonaDistribució(idZona, nomZona, descripció)

SistemaEmbalatge(idSistema, idProducte, descripció, quantitat)
On idProducte referencia Producte(idProducte)

TipusEmpresa(idTipus, descripció)

Empresa(idEmpresa, CIF, nomComercial, nomFiscal, adreça, telèfon, **idZona**, idTipus)
On idZona referencia ZonaDistribució(idZona)
idTipus referencia TipusEmpresa(idTipus)

Empleat(idEmpleat, descripció)

Fàbrica(idFàbrica, descripció, adreça, telèfon)

LíniaFabricació(idLínea, descripció, llargada, idResponsable, idFàbrica)
On idResponsable referencia Empleat(idEmpleat)
idFàbrica referencia Fàbrica(idFàbrica)

Neteja(idNeteja, descripció)

Torn(idTorn, descripció, idResponsable, horaInici, horaFi, capacitatPersones, idLínea)
On idResponsable referencia Empleat(idEmpleat)
idLínea referencia LíneaFabricació (idLínea)

Transport(idTransport, tipus, descripció, capacitat, disponibilitat)

Magatzem(idMagatzem, descripció, adreça, capacitat)

Entitats derivades de relacions

Compost(idProducte, idComponent, quantitat, unitatMesura)
On idProducte referencia Producte(idProducte)
idComponent referencia Component(idComponent)

Demanda(idProducte, idZona, data, quantitat)
On idProducte referencia Producte(idProducte)
idZona referencia ZonaDistribució (idZona)

Accès(idMagatzem, idEmpresa, Horari)
On idMagatzem referencia Magatzem(idMagatzem)
idEmpresa referencia Empresa(idEmpresa)

Producció(idProducte, idLínea, dataInici, dataFi, quantitatFabricada)
On idProducte referencia Producte(idProducte)
idLínea referencia LíneaFabricació(idLínea)

Netejament(idLínea, idNeteja, dataInici, dataFi)
On idLínea referencia LíneaFabricació(idLínea)
idNeteja referencia Neteja(idNeteja)

Assignació(idEmpleat, idTorn, data)
On idEmpleat referencia Empleat(idEmpleat)

idTorn referencia Torn(idTorn)

Stock(idSistema, idMagatzem, data, volum, quantitat, idTransport, **idEmpresa**)
On idSistema referencia SistemaEmbalatge(idSistema)
idMagatzem referencia Magatzem (idMagatzem)
idTransport referencia Transport(idTransport)
idEmpresa referencia Empresa(idEmpresa)

3.2 Normalització

La normalització és un procés per evitar redundàncies i anomalies en la inserció, actualització i esborrat de dades. [6]

Aquest procés es basa en la descomposició d'entitats per crear-ne de noves a partir de la divisió dels atributs de la primera. [7]

Si bé quan ja es té certa experiència, l'obtenció d'entitats normalitzades és en certa mesura trivial, ara seguirem les diferents etapes per confirmar si es troba o no normalitzat el nostre model lògic.

Primera Forma Normal (1FN)

En aquesta forma normal, no poden haver més de dos valors en un sol atribut per una única clau primària. [6]

Si s'han triat correctament les claus primàries, aquesta forma normal ja hauria de complir-se i, de fet, es compleix per totes les Entitats.

Posem l'exemple d'Empresa, que tots els atributs tinguin un sol valor per una clau idEmpresa significa que un sol idEmpresa només pot tenir un CIF, un nomComercial, un nomFiscal, una adreça, un telèfon, un idTipus i farem que només actuï sobre una sola ZonaDistribució amb un sol valor de idZona.

Per tant, totes les Entitats es troben en 1FN.

Segona Forma Normal (2FN)

Es troba en aquesta forma normal si, primerament, es trobava en 1FN i tots els atributs que no són clau candidata (U1, U2, etc.) depenen exclusivament de la clau primària. [6]

Si s'han escollit correctament les claus primàries, totes les taules que tinguin una sola clau primària es troben, per defecte, en segona forma normal.

Posem d'exemple, el SistemaEmbalatge. Tant la descripció com la quantitat depenen del idSistema i del idProducte. Si la quantitat només depengués d'un d'ells, significaria que per un sistema d'embalatge en concret, qualsevol producte tindria les mateixes unitats independentment del producte que fos.

Si les repassem, totes les Entitats es troben en 2FN.

Tercera Forma Normal (3FN)

Es troba en aquesta forma normal si, primerament, es trobava en 2FN i no existeix cap atribut que no sigui clau candidata i que depengui de la clau primària a través d'un altre atribut que no sigui clau primària. [6]

Posem l'exemple de Transport. Si per alguna raó la disponibilitat depengués de la capacitat i aquesta depèn de la clau primària, l'Entitat no es trobaria en 3FN.

Si repassem les Entitats, veurem que totes depenen directament de la seva clau primària, per tant, es troben en 3FN.

Forma Normal de Boyce-Codd (BCNF)

Es troba en aquesta forma normal si, primerament, es trobava en 3FN i tots els determinants són claus candidates. Els determinants són atributs que determinen el valor d'un altre atribut. [6]

Aquest cas tant particular només es pot donar quan existeix una clau composta i una possible clau candidata composta encavalcada (utilitza un atribut en comú amb la clau composta).

En el nostre cas, no existeix cap clau candidata composta. Totes les claus candidates compostes són les úniques que hi poden haver. Per tant, es troben en BCNF.

Quarta Forma Normal (4FN)

Es troba en aquesta forma normal si, primerament, es trobava en BCNF i no presenta dependències multivaluades independents. [6]

Una Entitat pot trobar-se en BCNF i no trobar-se en 4FN. Es pot donar aquest cas quan estem barrejant conceptes en una sola taula.

Utilitzem Distribuït com a exemple. Si quin transports utilitza una empresa per transportar sistemes d'embalatge i l'altre concepte fos la data que s'han fet moviments en un magatzem, ens trobaríem en aquest cas.

El nostre objectiu, però, és el d'emmagatzemar els moviments de sistemes d'embalatge de les empreses, amb el medi de transport que s'ha utilitzat per fer-ho, la data quan s'ha realitzat i de quin magatzem s'han retirat, per tant, és un sol concepte.

Totes les Entitats compleixen la 4FN.

Cinquena Forma Normal (5FN)

Es troba en aquesta forma normal si, primerament, es trobava en 4FN i no presenta dependències de projecció-combinació. Simplificant, això significa que les taules no es poden fer més petites amb claus diferents. Per tant, aquesta forma normal només afecta a les Entitats amb clau composta. [6]

Les Entitats que han sorgit d'una relació on les seves claus primàries són la composició de les de les Entitats que enllacen les podem descartar perquè si s'ha creat l'Entitat associativa és perquè és necessària i els atributs que no són clau no poden anar a cap de les altres Entitats.

La resta d'atributs de les Entitats amb clau composta depenen totalment de la clau composta i no es poden descomposar sense perdre el seu sentit.

Per tant, podem concloure que les Entitats es troben en 5FN.

4. Disseny Físic

4.1 Preàmbul

Aquesta és la fase de implementació del model lògic a un suport físic. La informació que recollirem o generarem s'ha de guardar en un suport físic de forma persistent i són els SGBD els que realitzen la gestió del manteniment del seu emmagatzematge. [7, pàg. 1-2]

En aquest apartat, triarem el SGBD més convenient, generarem la base de dades i implementarem les Entitats anteriors en forma de taules i columnes.

4.2 Tria del SGBD

A la fase del model lògic s'ha escollit el model relacional de gestió de bases de dades. Com a tal, haurem de triar un SGBD que es basi en el model relacional, per tant, paradigmes com bases de dades orientades a objectes, gràfiques o d'altres queden descartades.

Dins els possibles SGBD basats en el model relacional com són SQL Server de Microsoft, MySql o altres, s'ha decantat pel SGBD Oracle Database 11g Express Edition que és la que s'ha treballat a les assignatures del curs.

Per accedir a la base de dades i gestionar-la s'utilitzarà el IDE anomenat Oracle SQL Developer que proporciona Oracle gratuïtament i la consola de comandes.

La instal·lació de l'entorn es considerarà fóra del nostre treball.

4.3 Creació de l'espai de taules i esquema

L'edició Express de Oracle només permet crear una sola base de dades que es configura durant la instal·lació juntament amb l'usuari administrador 'SYSTEM'.

Per tant, per poder treballar de forma separada a la resta de taules i informació guardada a la base de dades, treballarem amb els espais de taules i els esquemes.

Les bases de dades es troben compostes pels espais de taules o *tablespaces* que són una separació lògica que, en el seu conjunt, guarden tota la informació de la base de dades. A la mateixa vegada, els espais de taules es troben compostes per fitxers de dades o *datafiles* que són les estructures físiques associades a un espai de taules. [8]

Diferents objectes com, per exemple, les taules poden trobar-se repartits entre diferents espais de taules, però només es crearà un espai de taules nou per tenir recollits tots els objectes d'aquest treball.

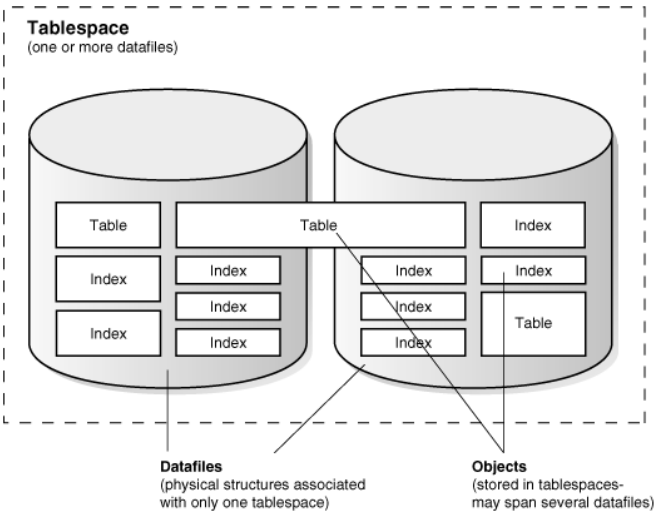


Figura 7 Espai de taules, fitxers de dades i objectes.

Els dos tipus d'espai de taules més importants són [9]:

- Permanents: Utilitzats per guardar dades permanents.
- Temporals: Utilitzats per guardar dades temporals de les sessions.

Es crearà un espai de dades permanent nou i s'utilitzarà l'espai de dades temporal TEMP que ve per defecte.

Oracle SQL considera els esquemes o *schemes* sinònims a usuaris. Es necessita un esquema per poder iniciar sessió i treballar sobre la base de dades.

Els esquemes són estructures de dades lògiques que es guarden i s'assignen a un espai de taules. [10]

Els passos que seguirem per preparar el nostre entorn de treball seran:

1. Entrar a la sessió de l'usuari administrador SYSTEM.
2. Crear l'espai de taules permanent TFG.
3. Crear l'usuari TFGAdmin assignat a aquest espai de taules.
4. Donar permisos a l'usuari TFGAdmin.
5. Canviar la sessió a l'usuari TFGAdmin.

A l'Annex I es mostren els scripts per preparar l'entorn de treball. Per fer l'últim punt, només ens hem de desconnectar de la sessió SYSTEM i iniciar-ne una nova amb l'usuari TFGAdmin.

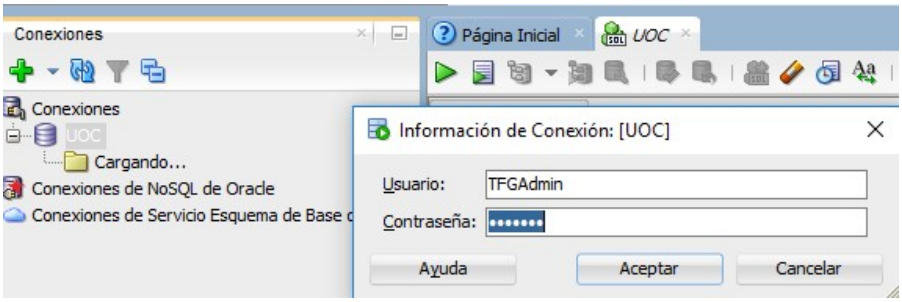


Figura 8 Inici de sessió a la connexió de la nostra base de dades amb el nou usuari.

4.4 Creació de scripts de taules

Amb la sessió iniciada com l'usuari o esquema TFGAdmin, procedirem a crear les taules que s'han modelat amb anterioritat.

A l'Annex II es pot veure el script utilitzat per crear les taules. La seva creació es troba en el mateix ordre que es mostren a l'apartat 3.2. A la [Figura 9] es pot observar el resultat de la creació de les taules.

Per crear les taules s'ha utilitzat el Llenguatge de Definició de Dades de SQL (DDL) que es troba estandarditzat i és independent del SGDB utilitzat. [11, pàg. 248]

Com a criteri general, s'han creat les taules i les seves columnes sense accents i caràcters regionals, ja que molts programes no els reconeixen correctament i, per ordre general, es solen evitar.

La [Figura 10] s'ha obtingut a partir del Data Modeler del IDE Oracle SQL Developer que ens permet generar el diagrama relacional a partir d'un esquema.

Totes les taules es troben distribuïdes de forma que coincideixin amb les Entitats del model relacional de la [Figura 5]. Es pot observar com les Taules i les relacions coincideixen en la seva totalitat.

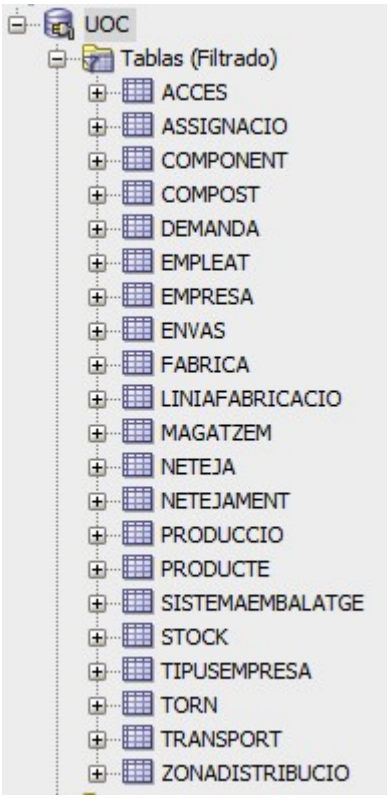


Figura 9 Resultat de la creació de taules.

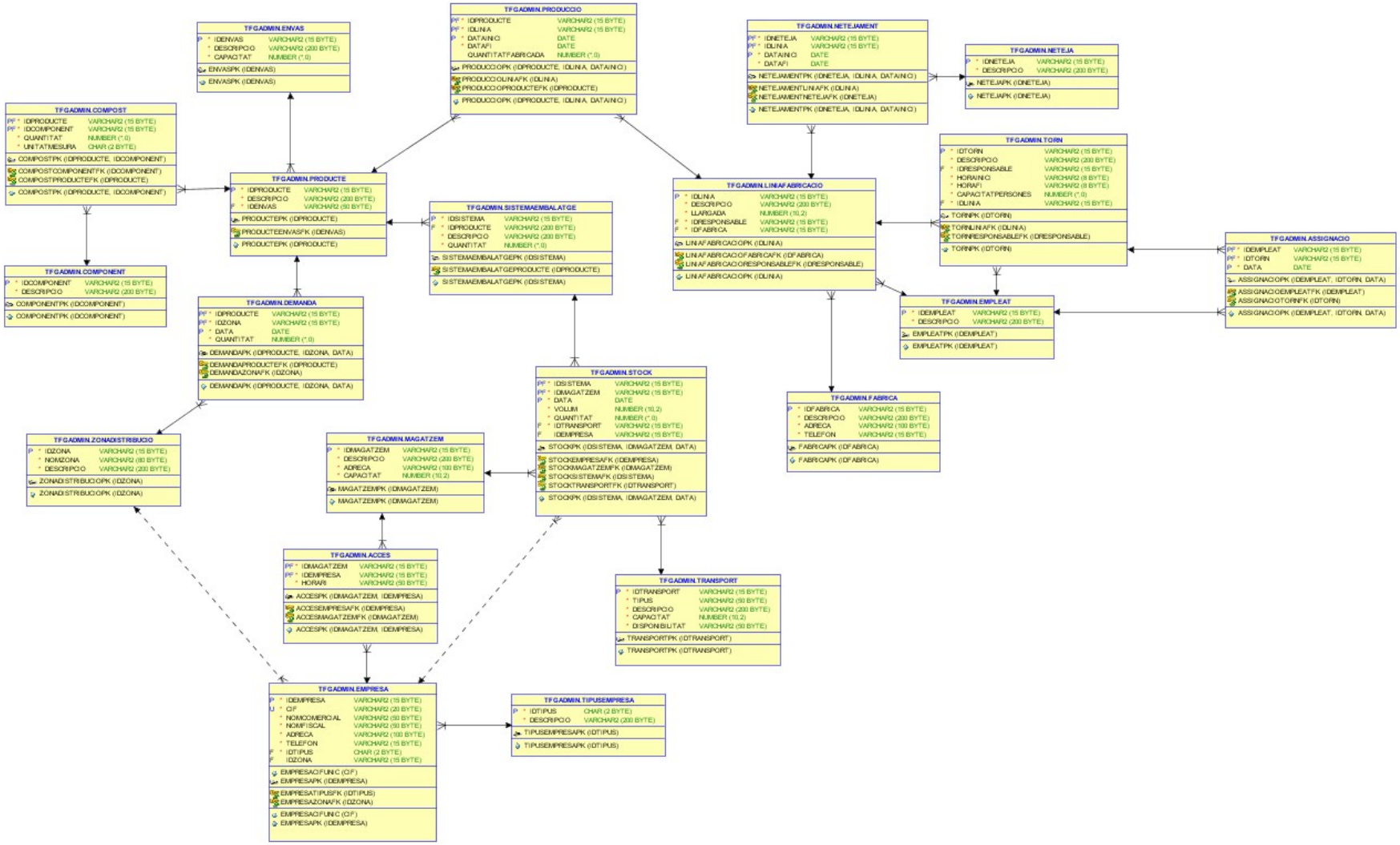


Figura 10 Diagrama relacional obtingut del Data Modeler de Oracle SQL Developer

5. Optimització

5.1 Preàmbul

Aquest aparat es troba dedicat a la generació d'eines per facilitar la tasca de consulta i generació de dades als desenvolupadors i usuaris de l'aplicació final.

En la tasca relacionada amb les consultes que demana el client, s'ha decidit que les dades agregades es generaran cada cop que executin un dels procediments de les taules relacionades amb les consultes i el resultat, es guardarà en noves taules que seran les consultades.

Aquest enfoc permet que cada cop que hi hagi una inserció, modificació o esborrat, les dades agregades d'aquestes taules s'actualitzin i sempre que es consultin es trobaran actualitzades.

Si féssim una consulta amb una clàusula d'agrupació i ho guardéssim com a vista, cada cop que fos cridada hauria de fer els càlculs d'agrupació corresponents. Això significaria que per una base de dades amb una mida considerable, els recursos podrien ser absorbits per aquestes consultes.

Si fos una vista materialitzada, aquesta s'hauria de refrescar cada cert temps, així que els càlculs s'haurien de realitzar igualment. Tot i que els càlculs es trobessin espaiats o es fessin en períodes de poca activitat, en el transcurs entre períodes no es garanteix que les dades es trobin actualitzades.

Per tant, les consultes realitzades es faran sobre les taules expressament dissenyades per donar-les resposta. Per facilitar la feina de ser referenciades, es guardaran amb un nom com a vistes simples.

5.2 Disseny de consultes

S'ha considerat adient separar les taules de consulta de la resta de taules que constitueixen el nucli del sistema perquè la tipologia de consultes pot variar segons la necessitat de l'organització de forma separada al nucli de l'organització.

Aquesta separació física en la documentació també permet diferenciar fàcilment les taules del nucli operacional a les consultes considerades com més emprades que, en definitiva, tenen objectius diferents (unes recullen les transicions i operacions de l'organització i les altres són més estadístiques). De fet, es podrien considerar com dos mòduls diferents.

A continuació ens mourem consulta per consulta explicant les decisions de disseny. Els scripts d'aquest apartat es poden consultar a l'Annex III.

A la següent figura es mostra el disseny d'aquestes taules creat amb el Data Modeler.

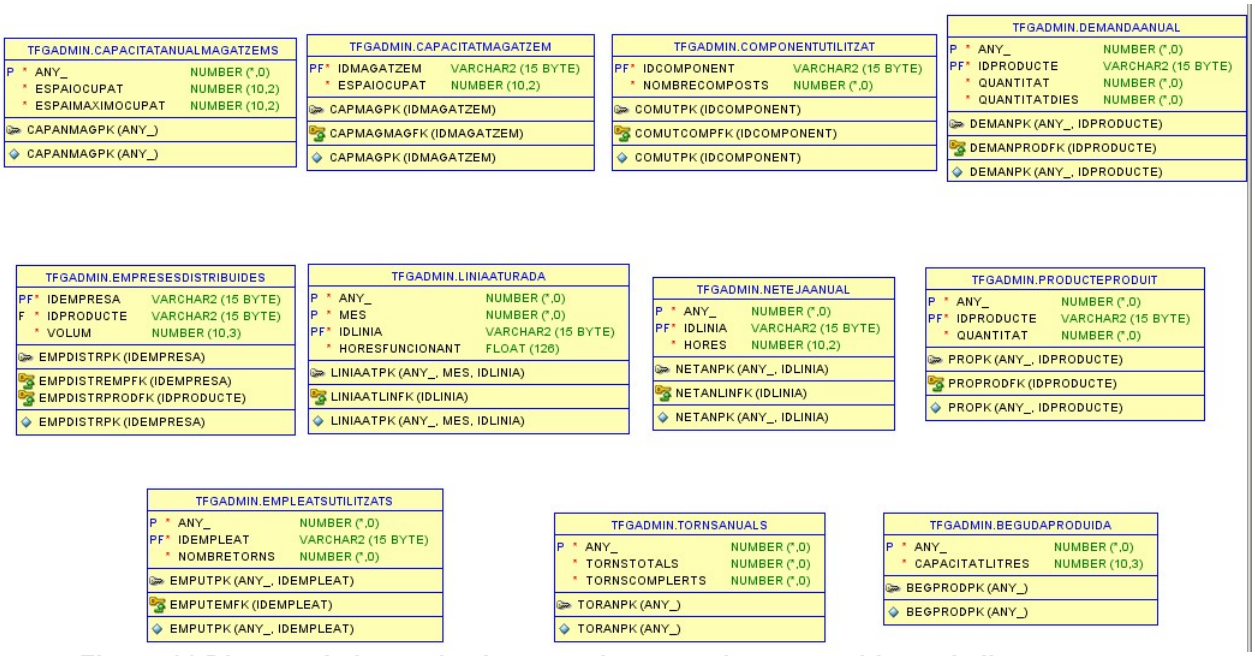


Figura 11 Disseny de les taules base per les consultes requerides pel client.

Producte més produït

Aquesta consulta ens demana que donat un any qualsevol, indiquem quin és el producte més produït.

Per aquest fi, s'ha creat la taula ProducteProduit que s'omplirà a partir dels registres de la taula Produccio que és la que emmagatzema totes les operacions de generació de productes.

Cada operació sobre un registre de Produccio modificarà la quantitat pel producte i any d'aquell registre.

Línia menys aturada

Aquesta consulta demana que, donat un mes qualsevol, s'indiqui la línia amb menor temps d'aturada.

L'enunciat no indica si es tracta d'un mes i any qualsevol o la suma de temps de tots els mesos passats. Interpretaré que sí es vol l'any perquè no sembla tenir sentit la suma de temps del mateix mes per tots els anys en operació. Potser una línia estava en obres durant un mes d'un any concret i aquest fet s'arrossegaria durant la resta dels anys.

La taula LiniaAturada ens indica les hores de funcionament d'una liniaFabricacio per mes i per any. Amb aquesta taula, podrem indicar un any i un mes concret i trobar la línia (o línies) que aquell any i mes hagin treballat més (o menys).

Capacitat d'emmagatzematge

Suposaré que amb 'una seu qualsevol' es refereix a un magatzem qualsevol i que amb 'un moment qualsevol' es refereix al moment de la consulta. Aquesta segona presumpció es deu a que segurament la consulta s'utilitza per saber quin magatzem té espai per portar els productes que s'acaben de fabricar. No veig quina raó podria haver per consultar la capacitat disponible d'una data anterior.

Per tant, la consulta a respondre serà: Donat un magatzem, obtenir la seva capacitat d'emmagatzematge al moment de la consulta.

Per aquest fi, s'ha creat la taula capacitatMagatzem que registra l'espai que es troba ocupat a cada magatzem de forma actualitzada.

Cada cop que hi hagi un moviment de Stock, s'actualitzarà l'espai ocupat d'aquesta taula.

Empleats més utilitzats

Aquesta consulta ens demana els deu primers empleats que han ocupat més torns en l'any actual.

S'utilitzarà la taula EmpleatsUtilitzats que emmagatzema l'any, l'empleat i el nombre de torns que ha participat per aquesta consulta.

Cada cop que s'assigni un torn amb Assignació, el comptador de torns augmentarà en aquesta taula per l'empleat i any corresponents.

Increment de demanda

Amb aquesta consulta es vol obtenir el percentatge d'increment de la demanda mitjà de productes d'un any respecte l'any anterior.

Suposaré que amb 'demanda mitjana' es refereix a la quantitat de demanda dividit pels dies que s'ha fet aquesta demanda i que quan menciona els productes, es vol saber aquest increment per producte.

La taula Demanda ens permet afegir una quantitat de demanda per dia, per tant, la demanda mitjana d'un any seria la suma de les quantitats partit pel nombre de dies d'aquell any concret.

La nova taula DemandaAnual guardarà la quantitat de dies i de producte inserit a la taula Demanda durant el transcurs dels anys.

Component més utilitzat

En aquest cas, es demana quin és el component més utilitzat en les fórmules de producció de productes.

Per fer aquesta consulta disposem de la taula ComponentUtilitzat que guardarà els components i el nombre de registres que apareix en la taula Compost.

Capacitat emmagatzematge anual

Es demana que indiquem l'any en el qual la capacitat d'emmagatzematge útil en tots els magatzems de l'empresa hagi sigut més petita.

S'interpretarà aquesta consulta com l'any on un dels seus dies va haver menor capacitat en el conjunt de magatzems que en qualsevol altre dia d'un altre any.

Per aquesta consulta s'utilitzaran els mateixos procediments que per la taula capacitatMagatzem. Cada cop que es modifiqui el valor d'un magatzem en aquesta taula, provocarà que també es modifiqui la quantitat total de tots els magatzems que es guardarà a la taula capacitatAnualMagatzems.

Aquesta taula guardarà la capacitat total ocupada en tots els magatzems donat un any concret.

Beguda produïda percentual anual

Aquesta consulta ens demana obtenir el percentatge d'increment (o decrement) dels litres de beguda produïda a l'any en curs respecte l'anterior.

En aquest cas, s'ha creat la taula anomenada BegudaProduida que guarda el volum en litres de beguda per any. Cada cop que es mouen registres a Produccio, es suma a l'any del registre la capacitat de l'envàs per la quantitat de producte produït.

Temps de Neteja Anual

Ens demanen saber l'any amb el major temps de neteja emprat en cada línia de fabricació.

S'ha interpretat aquesta consulta com, donat un any, saber quina és la línia que s'ha emprat més temps en netejar durant tot l'any.

Per tant, s'ha creat la taula NetejaAnual on es guarda l'any, la línia que s'ha netejat i el total d'hores invertides en la seva neteja. Cada cop que es modifiquin registres a Netejament, s'actualitzarà aquesta taula.

Percentatge de torns incomplerts

En aquest cas, ens demanen el percentatge de torns de producció que no es troben complerts en un any qualsevol.

Per complir amb aquest requisit, s'ha creat la taula TornsAnuals on es guarda l'any, el nombre de torns totals i el nombre de torns coberts.

Cada cop que s'assignin torns a Assignacio, es recalcularà el nombre de torns i mirarà si la inserció o eliminació afecta al torn registrat.

Empreses de distribució

En aquesta consulta ens demanen que trobem les cinc primeres empreses de distribució segons la quantitat total de líquid distribuït de qualsevol producte.

Suposaré que es refereix que volen saber les cinc primeres empreses que més volum de líquid hagin distribuït fins el moment de la consulta i de quin ha sigut aquest producte.

He cregut que és més interessant saber el producte que més s'ha distribuït per saber l'èxit de cada producte.

Si el que es demana en realitat és saber tota la quantitat distribuïda independentment del producte, encara es pot obtenir de la taula empresesDistribuides tot i que llavors sí serà necessari fer una clàusula d'agrupació que no es troben permeses en aquest treball o fer una altra taula per aquesta consulta en concret.

Per aquest requisit, s'ha creat la taula EmpresesDistribuides que emmagatzema l'empresa distribuïdora, el producte i el volum total en litres del producte que ha mogut. Cada cop que es registri una sortida en el Stock, augmentarà el producte distribuït per l'empresa distribuïdora que l'hagi tret.

D'aquesta forma, es poden ordenar les empreses filtrant-les per producte i obtenir aquelles que més hagin distribuït fins el moment.

5.3 Creació de scripts ABM i auxiliars

En aquest apartat es començarà a treballar amb PL/SQL que és el llenguatge procedimental que utilitza Oracle. També es crearan procediments representatius de creació, actualització i esborrat de les taules més representatives. Si no es triés aquest subconjunt de procediments, tenint en compte que són 3 operacions (sense tenir en compte els procediments de lectura) multiplicat pel nombre de taules del nostre model, el conjunt total seria molt ampli i gran part serien repeticions només variant el nombre de paràmetres i el nom de la taula afectada.

Tenint això en compte, es considera que només és necessari preparar un subconjunt de procediments per demostrar el procés, preparar la base de dades per les consultes específiques i continuar amb altres parts del treball.

En general, s'han dissenyat els procediments per a que tractin excepcions guardant el resultat de la crida dins una variable de sortida anomenada RSP. Aquesta variable mostrarà OK si es crida el procediment i es mostra el contingut de la variable i la seva execució ha sigut correcta. De no ser correcta, RSP guardarà ERROR: i el contingut de la variable SQLERRM (que es trobarà

ple quan entri a l'excepció del procediment) o un error personalitzat degut a la tria de disseny. [12]

Com que s'ha guiat pels valors que es mostren en els exemples de l'enunciat, la majoria de les claus consten d'un caràcter seguit de zeros i el nombre d'ordre d'inserció a la taula, p.ex. C0001.

Entre les funcions auxiliars, existeix tornaIdTaula que, utilitzant SQL dinàmic, cerco el últim id de qualsevol taula (amb un únic id), el tallo i li augmento el nombre per a la nova inserció. Si no existeix un primer registre, entra a l'excepció NO_DATA_FOUND i retorna el caràcter inicial del identificador amb zeros fins el 1.

La versió 11g del SGBD de Oracle té certes decisions de disseny i mancances que han hagut de ser sortejades mitjançant programació o consultes SQL alternatives. Un d'aquests exemples és que el paràmetre rownum s'assigna abans de fer l'ordenació i per tant, no es pot triar directament les primeres files per ordenació i és necessari fer una subconsulta que s'utilitzarà com a taula a consultar. [13] Un altre cas, és el fet que no es puguin assignar columnes numèriques auto-incrementables, per això, s'ha hagut de modificar el procediment tornaIdTaula només per incorporar la generació del primer nombre auto-numèric de les taules que ho necessitin (només taula LOG).

Per emmagatzemar les crides a procediments, s'ha creat una nova taula LOG que es troba fora de l'anàlisi del sistema. En aquesta taula aïllada es guarden les crides que es realitzen als procediments. Cada procediment té sentències d'inserció en aquesta taula i el procediment crearLOG és l'única excepció en aquest aspecte (ja que podria provocar un bucle infinit), en el tractament d'excepcions (només és una inserció) i també en l'aspecte de la variable RSP.

Els scripts de creació de procediments, funcions auxiliars i creació de la taula LOG es poden consultar a l'Annex IV.

En general, la majoria de procediments de modificació i esborrat han de desfer el que han fet els procediments de creació. Per aquesta raó, existeixen variables que guarden els valors dels registres abans que siguin modificats i esborrats per poder rectificar la resta de taules implicades.

Ara es farà un resum més concret dels procediments que intervenen en cadascuna de les consultes.

Producte més produït

El procediment crearProduccio genera un registre a la taula Produccio. A partir de la quantitat registrada, també la inserta pel producte i l'any de la dataFi a la taula ProducteProduit.

El procediment modificarProduccio modifica els valors dels camps de la taula Produccio. Agafa l'any de dataFi abans de modificar el registre i la quantitatFabricada i li resta al registre de ProducteProduit. Després agafa l'any

de la dataFi de modificació i afegeix (o crea) el registre a ProducteProduit. Aquest procediment, per tant, té en compte les produccions nocturnes que es donin al canviar l'any quan són modificades.

Per últim, el procediment esborrarProduccio elimina el registre de Produccio, treu la quantitat que s'havia fabricat de ProducteProduit i elimina els registres a 0.

Línia menys aturada

Consultarem la taula LiniaAturada però com que l'omplirem a partir de la Produccio, ampliarem els mètodes crearProduccio, modificarProduccio i esborrarProduccio.

S'ha creat un procediment auxiliar actualitzaLiniaAturada que serveix per crear, sumar o restar les hores de funcionament de la línia per les dates especificades. Aquest procediment facilita la tasca de cridar-lo als tres anteriors.

Quan el paràmetre i_SumaResta és 1, actualitza la línia existent restant-li la diferència de les hores per cada mes entre les dues dates i_dataInici i i_dataFi. Aquest paràmetre s'utilitza per eliminar les hores de funcionament en el cas de modificarProduccio i esborrarProduccio.

Quan el paràmetre és 2, ens permet crear o sumar les hores de producció en els procediments crearProduccio i modificarProduccio.

Es podria dir que el procediment es troba dividit en dues parts:

- Si la data d'inici i la de fi cauen al mateix mes, es crea o actualitza el registre amb una simple sentència.
- Si la data d'inici i de fi cauen en diferents mesos, es fa un bucle per tots els mesos entre aquestes dates i es calcula la suma d'hores en una subdivisió de tres parts:
    - Es sumen les hores del primer mes entre la data d'inici (que pot caure enmig del mes) i el final de mes.
    - Es sumen les hores de tots els mesos entremig.
    - Es sumen les hores entre el primer dia de l'últim mes i la data de fi (que pot caure enmig del mes).

Tot i que la consulta es tracta de la línia més aturada, el més convenient pel disseny era obtenir les hores treballades per línia i després ordenarles inversament.

Capacitat d'emmagatzematge

S'han creat els procediments crearStock, modificarStock i esborrarStock per donar resposta a aquesta consulta.

El procediment crearStock inserta a la taula Stock el registre amb els paràmetres indicats. Abans d'això, comprova que el magatzem on s'insereixi

tingui espai disponible. Si és així, insereix (o suma) també a capacitatMagatzem l'espai que ocuparà aquest stock. En cas contrari, cridarà una excepció personalitzada per avisar que el magatzem es desbordaria si aquest stock hi arribés.

El procediment modificarStock fa la mateixa comprovació però trient el valor anterior a la capacitat.

Per últim, esborrarStock no ha de fer cap comprovació degut a què només s'esborra el registre de Stock i es redueix l'espai de capacitatMagatzem.

Empleats més utilitzats

Com que tots els camps de la taula Assignacio són clau primària, obligaré que si es vol fer una modificació, primer s'esborri el registre i es creï un de nou. Per aquesta raó, per aquest apartat, no existeix un procediment de modificació.

Per tant, en aquest apartat només s'han creat els procediments crearAssignacio i esborrarAssignacio.

A crearAssignacio genera el registre corresponent a Assignacio i crea un registre a empleatsUtilitzats amb un comptador de u o li suma u al comptador si ja existia un registre per aquell any i empleat.

A esborrarAssignacio, elimina el registre de Assignacio i resta una unitat al comptador de empleatsUtilitzats.

Increment de demanda

Per aquesta consulta s'han creat els procediments crearDemanda, modificarDemanda i esborrarDemanda.

A crearDemanda es genera el registre a Demanda segons els paràmetres especificats. També s'aprofita per crear el registre a demandaAnual pel idProducte concret, l'any del registre de Demanda i la quantitat de la Demanda. En cas que ja existeixi un registre, es suma la quantitat.

A esborrarDemanda s'esborra el registre de Demanda i se li resta la quantitat anteriorment imputada.

Finalment, a modificarDemanda, s'esborra la quantitat anterior i es suma la nova quantitat a les dues taules.

Com he comentat, s'ha considerat que la consulta es vol fer per producte ja que, d'aquesta forma, sembla més pràctica per saber quins productes es venen millor o pitjor.

Si es tractés de tota la demanda independentment del producte, només s'hauria de modificar o crear una taula que no tingués en compte el idProducte però el procés seria el mateix.

Component més utilitzat

El procediment crearCompost fa la inserció d'un registre a la taula Compost amb els seus camps corresponents. Si no existeix registre a ComponentUtilitzat, el crea i, si existeix, en suma una unitat.

El procediment esborrarCompost elimina el Compost indicat i treu una unitat a la taula ComponentUtilitzat del Compost del registre esborrat. També esborra tots aquells registres que quedin a 0 (en un principi, només el que s'acaba de modificar).

També s'ha creat el procediment modificarCompost que només modifica les dades del Compost indicat però no té repercussió a ComponentUtilitzat.

Capacitat emmagatzematge anual

Per aquesta consulta es reutilitzaran els procediments crearStock, modificarStock i esborrarStock.

Els procediments s'han ampliat modificant l'espai ocupat a l'any del registre inserit, modificat o esborrat a Stock.

Com que també es poden fer registres negatius per guardar les sortides de les empreses distribuïdores, el camp espaiMaximOcupat és qui guarda l'espai màxim que s'ha ocupat durant aquell any.

Bàsicament cada cop que s'insereix, modifica o esborra un registre, capacitatAnualMagatzems es veu modificat guardant el valor del moment i el valor més alt de capacitat que s'ha arribat.

Per poder realitzar aquesta funcionalitat, agafa l'espi que es troba ocupat al magatzem i li suma el nou espai d'ocupació (en la inserció del registre). Si aquest valor és major que el registre existent (o si no hi ha registre), es guarda aquest nou valor.

En les modificaions, es comprova que el valor anterior, restant el volum que se li havia sumat i sumant el nou volum dóni o no un major espai màxim. De ser així, es reemplaça.

A l'esborrat, es resta de l'espai màxim ocupat, el volum anterior.

Beguda produïda percentual anual

Es tornaran a ampliar els procediments crearStock, modificarStock i esborrarStock per donar solució a aquesta consulta.

En aquest cas, com que a Stock entren sistemes d'emmagatzematge, el que es fa és multiplicar la quantitat de producte del sistema per la capacitat de l'envàs del producte (que es troba en mil·lilitres) dividit per mil (per trobar els litres).

Un cop trobats els litres del registre, s'insereix (crearStock), es modifica (crearStock i modificarStock) o es resta (esborrarStock) de l'any del registre de BegudaProduida.

Temps de Neteja Anual

S'han creat els procediments crearNetejament, modificarNetejament i esborrarNetejament.

A crearNetejament es crea el registre a la taula Netejament amb els paràmetres de la seva taula i també s'afegeix (o es suma) a un registre de NetejaAnual la diferència entre els dies de dataInici i dataFi (en hores).

A modificarNetejament, s'esborren les hores inserides anteriorment i s'afegeixen les hores després de la modificació.

A esborrarNetejament, es resten les hores i s'esborra el registre si es queda sense hores.

Percentatge de torns incomplerts

S'han aprofitat els procediments crearAssignacio i esborrarAssignacio.

A crearAssignacio es mira els torns que hi ha, la capacitat del torn del registre inserit i les persones que es troben a aquest torn. Si la capacitat i les persones que es troben en el torn són les mateixes, augmenta el comptador de torns complets.

A esborrarAssignacio fa la mateixa comprovació amb el nombre de persones abans d'esborrar el registre. Si coincideix, es resta al comptador.

Empreses de distribució

S'han tornat a ampliar els procediments crearStock, modificarStock i esborrarStock.

En aquest cas, les quantitats i el volum són negatius. Per tant, s'ha tingut en compte per no afectar a la resta de consultes. També s'ha tingut en compte per sumar-los a la taula EmpresesDistribuides que és la que guarda el volum (en litres) distribuïts per les empreses distribuïdores.

El procés és similar a la consulta de la beguda produïda, només que ara s'ha de tenir en compte que els valors són negatius i que el registre inserit ha d'haver-hi una empresa.

5.5 Creació de consultes

Les consultes que donen resposta a les qüestions plantejades anteriorment es troben en el mateix ordre que s'han descrit en els apartats anteriors a l'Annex V.

Com s'ha comentat, algunes tenen paràmetres (com l'any) que s'han de modificar per la persona que les consulta. A les consultes ja venen donats alguns valors d'exemple.

Per les limitacions que té Oracle SQL indicades anteriorment, la majoria de consultes que han de limitar la quantitat de registres tenen subconsultes que són les que ordenen els registres.

A continuació s'expliquen més en detall cadascuna de les consultes.

Component més utilitzat

La consulta troba el primer component amb més participació en compostos i el nombre de compostos que en participa.

La subconsulta ordena els components pel nombre de composts que participa descendentment (major participació adalt).

Amb la clàusula WHERE s'aconsegueix limitar la resposta al primer d'aquests registres.

Producte més produït

Troba el producte més produït amb la seva quantitat produïda per un any concret.

La subconsulta ordena els productes descendentment en funció a la quantitat (els que tenen més quantitat adalt) i la seva clàusula WHERE s'indica l'any que es vol consultar.

Amb la clàusula WHERE de la consulta principal s'aconsegueix limitar la resposta al primer d'aquests registres.

Línia més aturada

Troba la primera línia amb més hores aturades per l'any i mes indicats i mostra la línia, les hores que ha estat funcionant i les hores d'aturada respecte el total d'hores d'aquell mes.

La subconsulta agafa la línia, l'any, mes, hores que es troba funcionant i fa el càlcul de totes les hores d'aquell mes i li resta les hores que ha funcionat per obtenir les hores d'aturada i ho ordena segons aquestes hores descendentment (la que té més hores d'aturada adalt).

La subconsulta és on s'ha de filtrar l'any i el mes que es vol consultar.

Amb la clàusula WHERE de la consulta principal s'aconsegueix limitar la resposta al primer d'aquests registres.

Capacitat emmagatzematge actual magatzem

Troba el magatzem, el seu espai ocupat, la seva capacitat màxima i l'espai disponible del magatzem indicat en el moment de consulta.

Empleats més utilitzats

Troba els 10 empleats més utilitzats i el nombre de torns que ha realitzat cadascú per l'any indicat.

La subconsulta ordena segons el nombre de torns que han realitzat els empleats descendentment (qui ha treballat més torns adalt) i filtra per l'any que s'indiqui.

Amb la clàusula WHERE de la consulta principal s'aconsegueix limitar la resposta als deu primers d'aquests registres.

Increment percentual de demanda anual

Troba la demanda mitjana l'any anterior (quantitat en relació als dies de demanda registrats) i el seu increment respecte l'any anterior per un any i producte concret.
Hi ha dues subconsultes, la de l'any que es vol consultar i la de l'any anterior.

Les subconsultes troben la quantitat, els dies de registre de la demana i la demanda mitjana per l'any i producte demanats. Ambdós s'ha d'indicar també el producte que es vol consultar.

Any menor capacitat emmagatzematge

Troba l'any i l'espai ocupat on el total de l'espai ocupat entre tots els magatzems s'ha trobat al mínim.

La subconsulta ordena segons l'espai ocupat ascendentment (els registres amb menor espai ocupat adalt).

Amb la clàusula WHERE de la consulta principal s'aconsegueix limitar la resposta al primer d'aquests registres.

Increment percentual de beguda produida

Troba la quantitat de litres produïts d'aquest any i de l'any anterior i percentatge d'increment (o decrement) entre aquestes quantitats.

Hi ha dues subconsultes, la de l'any actual i la de l'any anterior. Les subconsultes troben la quantitat de litres produïts de l'any de consulta.

Any amb major temps de neteja

Troba l'any, la línia i les hores dedicades de neteja on s'ha dedicat més hores per l'any indicat.

La subconsulta ordena les línies segons les hores de neteja descendentment (les que han dedicat més temps adalt). S'ha de filtrar per l'any que es vol consultar.

Amb la clàusula WHERE de la consulta principal s'aconsegueix limitar la resposta al primer d'aquests registres.

Percentatge de torns incomplerts

Troba el nombre de torns totals realitzats, els d'aquests que es troben complerts i el percentatge de torns incomplerts d'un any indicat.

Aquesta és una consulta simple a tornsAnuals on només s'ha d'indicar l'any a consultar.

Primeres cinc empreses de distribució de producte

Troba les primeres cinc empreses que més volum han distribuït i de quin producte ha sigut.
La subconsulta ordena les línies segons el volum descendentment (major volum distribuït adalt).

Amb la clàusula WHERE de la consulta principal s'aconsegueix limitar la resposta als cinc primers d'aquests registres.

5.6 Inicialització de dades

Necessitem un conjunt de dades suficientment gran per poder provar les consultes que ens demana el client.

Si fos una implementació real, el propi client ens proporcionaria les dades correctes de la seva organització, ja sigui extraient fitxers de dades per trasllat d'un sistema seu anterior o tenint treballadors introduint dades a la versió inicial de proves del nostre sistema.

Com que no disposem d'un client real, la informació que s'introduirà a la base de dades serà completament fictícia i no té perquè respectar dades correctes o realistes. El seu principal i únic objectiu és proveir una base per poder realitzar i provar les consultes demanades. Per tant, la majoria d'aquestes seran escollides aleatòriament. Tot i així, es procurarà que, com a mínim, s'adeqüin al model de l'organització del nostre client.

Per generar les dades, s'ha començat per introduint-les a les taules de les Entitats que no provenen de les relacions. D'aquestes, les prioritàries són aquelles que es poden considerar com les més primitives sense cap referència a una altra taula.

En les taules relacionades amb les consultes (les que deriven de relacions entre Entitats) s'ha generat un seguit de scripts automatitzats per poder generar una quantitat considerable de dades.

Aquests scripts són aleatoris i criden els procediments comentats anteriorment. Per tant, si hi ha alguna restricció que incompleixen en la generació d'algun registre, aquest donarà error i es guardarà a la taula LOG.

Les dades resultants d'aquest apartat, es troba contingut al fitxer de còpia de la base de dades que es troba adjunt a aquest treball. Els scripts de dades aleatòries i de SELECT i DELETE utilitzats per tornar-les a generar es mostren a l'Annex VI.

5.7 Proves i coherència de dades

En aquest apartat s'ha escollit un subconjunt de totes les possibles proves. El conjunt de proves testejades és major però només s'han escollit les més significatives.
Es tracta d'una tasca molt àmplia i reiterativa, fet que produeix que el testeig de la base de dades pugui allargar-se indefinidament, arreglant punts febles i tornant a testejar o millorant el disseny anterior.

També existeixen diferents enfocs i metodologies de testeig de programari informàtic però, aquest treball, s'enfocarà a verificar que la generació, modificació i eliminació de dades a partir dels procediments sigui correcta i que les consultes mostren les dades corresponents segons la definició que s'ha indicat anteriorment.

Com que és necessari l'execució de codi i mostrar el seu resultat per verificar que aquesta s'ha portat a terme correctament, és l'únic apartat que, a més de tenir un annexe amb tot el codi, es mostrarà el codi prova a prova.

S'ha de recalcar que les dades resultants que es mostren de les proves poden variar en funció a les dades que es trobin guardades en aquell moment a la base de dades. Tot i així, el comportament i la funcionalitat serà la mateixa.

A l'Annex VI es troba tot el codi relatiu a aquest apartat.

Proves en la generació de dades i execució de procediments

Generació de claus primàries

El primer procediment que provarem serà el de generació de claus primàries. El resultat de la [Figura 12] es mostra a la [Figura 13] per a les dades d'aquell moment.

```
/*
Aquesta prova permet verificar el correcte funcionament del procediment crucial tornaIdTaula
*/
--Mostra els registres de Log i Component ordenats inversament per veure l'última clau
select * from LOG ORDER BY idLog DESC;
select * from component order by idcomponent desc;
--Instruccions PL/SQL
DECLARE
    id varchar(15);
BEGIN
    --Torna el nombre enter següent
    id:=tornaIdTaula('LOG','idLog');
    DBMS_OUTPUT.put_line(id);
    --Torna la combinació de lletra i nombre enter següent de llargada 5 (per defecte)
    id:=tornaIdTaula('component','idComponent','C');
    DBMS_OUTPUT.put_line(id);
    --Torna la combinació de lletra i nombre enter següent amb llargada 10
    id:=tornaIdTaula('component','idComponent','C',10);
    DBMS_OUTPUT.put_line(id);
END;
```

Figura 12 Codi associat a la prova del procediment de generació de claus primàries.

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA
1	245	CREARSTOCK	12/04/2019 20:09:34	S0002,M0004,23-03-2018 20:09:34,-5

	IDCOMPONENT	DESCRIPCIO
1	C0010	extracte de fruites

```
246
C0011
C000000011

Procedimiento PL/SQL terminado correctamente.
```

Figura 13 Resultat de l'execució del codi associat a la prova del procediment de generació de claus primàries.

Com podem observar, l'última clau del log és 245 i l'última clau de component és C0010. La prova retorna correctament les pròximes claus 246 i C0011 (o C000000011 si s'indiquen deu dígits).

Taules d'una sola clau

El codi de la [Figura 14] permet comprovar el funcionament dels procediments crearComponent, modificarComponent i esborrarComponent. Com he comentat, aquests procediments s'han creat com a exemples per a taules d'una sola clau o aïllades amb dades que poden existir sense referències. El resultat de les proves es mostra a la [Figura 15].

```
/*
Aquesta prova permet verificar procediments de gestió de dades d'una taula
simple d'una sola clau.
*/
--Elimina el log per tenir-ho net per la prova
DELETE FROM Log;
--Mostra la llista de components inicial
SELECT * FROM component;
DECLARE
    RSP varchar(200);
BEGIN
    --Crea dos components, proval i prova2
    crearComponent('proval',RSP);
    DBMS_OUTPUT.put_line(RSP);
    crearComponent('prova2',RSP);
    DBMS_OUTPUT.put_line(RSP);
    --modifica el primer component que hauria de tenir clau C0011
    modificarComponent('C0011','prova3',RSP);
    DBMS_OUTPUT.put_line(RSP);
    --Esborra el segon component
    esborrarComponent('C0012',RSP);
    DBMS_OUTPUT.put_line(RSP);
END;
--Mostra el log dels processos
SELECT * FROM Log;
--Mostra els components finals
SELECT * FROM component;
--Elimina el segon component de prova
DELETE component WHERE idComponent='C0011';
```

Figura 14 Codi associat a la prova de procediments de generació de dades a taules aïllades.

```
245 filas eliminado

 9 C0009  mescla herbes
10 C0010  extracte de fruites

OK
OK
OK
OK

Procedimiento PL/SQL terminado correctamente.
```

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
1	1	CREARCOMPONENT	08/05/2019 21:24:16	proval	OK
2	2	CREARCOMPONENT	08/05/2019 21:24:16	prova2	OK
3	3	MODIFICARCOMPONENT	08/05/2019 21:24:16	C0011,prova3	OK
4	4	ESBORRARCOMPONENT	08/05/2019 21:24:16	C0012	OK

	IDCOMPONENT	DESCRIPCIO
1	C0011	prova3

```
1 fila eliminado
```

Figura 15 Resultat de l'execució del codi associat a la prova de procediments de generació de dades a taules aïllades.

Com podem observar, es mostra 'OK' com a resposta a l'execució de cada procediment del codi. A més, en el Log es pot veure l'ordre d'execució, la data, els paràmetres utilitzats i el resultat.

Quan consultem la taula, només veiem el component 'C0011' que no s'ha esborrat i que havia sigut modificat amb anterioritat.

A partir d'aquest punt, les proves es troben dividides en la comprovació dels procediments individuals com en la [Figura 14] i la generació de dades massiva utilitzada per poder provar, en la següent secció, les consultes sobre un conjunt de dades suficient.

Taules relacionades amb Compost

Ara es provaran els procediments crearCompost, modificarCompost i esborrarCompost.

A la [Figura 16] es mostra l'eliminació de les dades actuals, la consulta de les taules implicades quan es gestiona la taula Compost i l'execució individual de cada procediment.

```
--Esborrat de les taules implicades
delete componentUtilitzat;
delete compost;
delete LOG;

--Consultes de les taules implicades
select * from component;
select * from producte;
select * from compost;
select * from componentUtilitzat;
select * from LOG;

DECLARE
    RSP varchar(200);
BEGIN
    --Creació de compostos
    crearCompost('P0005','C0001',600,'mL',RSP);
    DBMS_OUTPUT.put_line(RSP);
    crearCompost('P0004','C0002',300,'mG',RSP);
    DBMS_OUTPUT.put_line(RSP);
    --Modificació d'un dels compostos creats
    modificarCompost('P0004','C0002',500,'mG',RSP);
    DBMS_OUTPUT.put_line(RSP);
    --Esborrat d'un dels compostos
    esborrarCompost('P0005','C0001',RSP);
    DBMS_OUTPUT.put_line(RSP);
END;
```

Figura 16 Codi associat a les proves individuals dels procediments de gestió de la taula Compost.

Quan s'executa el codi PL/SQL tenim el resultat de la [Figura 17] al consultar les taules modificades (compost, componentUtilitzat i Log) que ens verifica que s'ha executat correctament.

	IDPRODUCTE	IDCOMPONENT	QUANTITAT	UNITATMESURA
1	P0004	C0002	500	mG

	IDCOMPONENT	NOMBRECOMPOSTS
1	C0002	1

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
1	3	MODIFICARCOMPOST	26/05/19	P0004,C0002,500,mG	OK
2	4	esborrarCompost	26/05/19	P0005,C0001	OK
3	1	CREARCOMPOST	26/05/19	P0005,C0001,600,mL	OK
4	2	CREARCOMPOST	26/05/19	P0004,C0002,300,mG	OK

Figura 17 Comprovació del resultat de l'execució del codi de la [Figura 16].

A la [Figura 18] es mostra el codi de generació massiva de dades per aquesta taula.

```
DECLARE
    RSP varchar(200);
    producte varchar(10);
    component varchar(10);
    unitatMesura varchar(5);
BEGIN
    --Bucle de 1 a 20
    FOR i in 1..20
    LOOP
        --Omple les variables producte i component amb valors aleatoris entre els 5 primers registres
        producte:='P'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0');
        component:='C'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0');
        --Si és el primer component (aigua) la unitat de mesura és mL
        IF component LIKE 'C0001' THEN
            unitatMesura:='mL';
        --En cas contrari, és mG
        ELSE
            unitatMesura:='mG';
        END IF;
        --Crea un compost amb dades aleatòries
        crearCompost(producte,component,ROUND(dbms_random.value(100,1000),0),unitatMesura,RSP);
        --Mostra el resultat de l'execució anterior
        DBMS_OUTPUT.put_line(RSP);
    END LOOP;
END;
```

Figura 18 Codi de generació de dades massives per la taula Compost.

Si l'executem i comprovem les taules implicades (compost, componentUtilitzat i Log) veiem el resultat de la [Figura 19].

	IDPRODUCTE	IDCOMPONENT	QUANTITAT	UNITATMESURA
1	P0003	C0004	498	mG
2	P0005	C0003	149	mG
3	P0003	C0005	251	mG
4	P0005	C0002	398	mG
5	P0003	C0001	882	mL
6	P0003	C0002	862	mG
7	P0001	C0004	125	mG
8	P0002	C0003	897	mG
9	P0005	C0001	714	mL
10	P0002	C0002	274	mG
11	P0004	C0002	643	mG
12	P0004	C0003	914	mG
13	P0005	C0004	416	mG
14	P0001	C0002	653	mG

	IDCOMPONENT	NOMBRECOMPOSTS
1	C0004	3
2	C0003	3
3	C0005	1
4	C0002	5
5	C0001	2

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
1	1	CREARCOMPOST	26/05/2019 13:27:38	P0003,C0004,498,mG	OK
2	2	CREARCOMPOST	26/05/2019 13:27:38	P0005,C0003,149,mG	OK
3	3	CREARCOMPOST	26/05/2019 13:27:38	P0003,C0005,251,mG	OK
4	4	CREARCOMPOST	26/05/2019 13:27:38	P0005,C0002,398,mG	OK
5	5	CREARCOMPOST	26/05/2019 13:27:38	P0003,C0001,882,mL	OK
6	6	CREARCOMPOST	26/05/2019 13:27:38	P0003,C0002,862,mG	OK
7	7	CREARCOMPOST	26/05/2019 13:27:38	P0001,C0004,125,mG	OK
8	8	CREARCOMPOST	26/05/2019 13:27:38	P0002,C0003,897,mG	OK
9	9	CREARCOMPOST	26/05/2019 13:27:38	P0005,C0001,714,mL	OK
10	10	CREARCOMPOST	26/05/2019 13:27:38	P0002,C0002,274,mG	OK
11	11	CREARCOMPOST	26/05/2019 13:27:38	P0004,C0002,643,mG	OK
12	12	CREARCOMPOST	26/05/2019 13:27:38	P0002,C0002,126,mG	ERROR:ORA-00001: unique constraint (TFGADMIN.COMPOSTPK) violated
13	13	CREARCOMPOST	26/05/2019 13:27:38	P0002,C0002,338,mG	ERROR:ORA-00001: unique constraint (TFGADMIN.COMPOSTPK) violated
14	14	CREARCOMPOST	26/05/2019 13:27:38	P0004,C0003,914,mG	OK
15	15	CREARCOMPOST	26/05/2019 13:27:38	P0001,C0004,197,mG	ERROR:ORA-00001: unique constraint (TFGADMIN.COMPOSTPK) violated
16	16	CREARCOMPOST	26/05/2019 13:27:38	P0002,C0002,787,mG	ERROR:ORA-00001: unique constraint (TFGADMIN.COMPOSTPK) violated
17	17	CREARCOMPOST	26/05/2019 13:27:38	P0005,C0004,416,mG	OK
18	18	CREARCOMPOST	26/05/2019 13:27:38	P0002,C0002,796,mG	ERROR:ORA-00001: unique constraint (TFGADMIN.COMPOSTPK) violated
19	19	CREARCOMPOST	26/05/2019 13:27:38	P0002,C0003,715,mG	ERROR:ORA-00001: unique constraint (TFGADMIN.COMPOSTPK) violated
20	20	CREARCOMPOST	26/05/2019 13:27:38	P0001,C0002,653,mG	OK

Figura 19 Resultat de consultar les taules implicades després del seu esborrat i execució del codi de la [Figura 18].

Com podem observar, les dades entre les taules concorden. La taula componentUtilitzat té correctament cada component que s'utilitza i les vegades que s'utilitza a la taula compost.

Com que es tracta d'una generació aleatòria de dades sobre un petit conjunt de dades de les taules principals, veurem en moltes proves que hi ha col·lisions.

L'última part de la [Figura 19] es veuen aquestes col·lisions de claus primàries però l'excepció és tractada correctament i la resta d'insercions es creen sense problemes.

Gestió de Producció

En aquesta secció es mostra el codi per provar els procediments crearProduccio, modificarProduccio i esborrarProduccio i les seves taules implicades.

A la [Figura 20] es mostra el codi utilitzat per provar els procediments individuals.

```
--Esborrat de les taules implicades
DELETE Produccio;
DELETE ProducteProduit;
DELETE LOG;
DELETE LiniaAturada;

--Consulta de les taules implicades
SELECT * FROM Producte;
SELECT * FROM LiniaFabricacio;
SELECT * FROM Produccio;
SELECT * FROM ProducteProduit;
SELECT * FROM LOG;
SELECT * FROM LiniaAturada;

DECLARE
    RSP varchar(200);
BEGIN
    --Creació de produccions de producte
    crearProduccio('P0001','L0001',TO_DATE('27/03/2018 8:00:00'),TO_DATE('28/03/2018 14:00:00'),50,RSP);
    DBMS_OUTPUT.put_line(RSP);
    crearProduccio('P0001','L0001',TO_DATE('28/03/2018 8:00:00'),TO_DATE('28/04/2018 14:00:00'),100,RSP);
    DBMS_OUTPUT.put_line(RSP);
    crearProduccio('P0002','L0002',TO_DATE('27/03/2018 8:00:00'),TO_DATE('27/03/2018 14:00:00'),25,RSP);
    DBMS_OUTPUT.put_line(RSP);

    --Modificació d'una de les línies anteriors
    modificarProduccio('P0001','L0001',TO_DATE('27/03/2018 8:00:00'),TO_DATE('27/03/2018 14:00:00'),20,RSP);
    DBMS_OUTPUT.put_line(RSP);

    --Esborrat d'una de les línies anteriors
    esborrarProduccio('P0002','L0002',TO_DATE('27/03/2018 8:00:00'),RSP);
    DBMS_OUTPUT.put_line(RSP);
END;
```

Figura 20 Codi associat a les proves individuals dels procediments de gestió de Producció.

Quan s'executa el codi i es consulta a les taules implicades (Produccio, ProducteProduit, Log i LiniaAturada), resulta en les dades de la [Figura 21].

	IDPRODUCTE	IDLINIA	DATAINICI	DATAFI	QUANTITATFABRICADA
1	P0001	L0001	27/03/2018 08:00:00	27/03/2018 14:00:00	20
2	P0001	L0001	28/03/2018 08:00:00	28/04/2018 14:00:00	100

	ANY_	IDPRODUCTE	QUANTITAT
1	2018	P0001	120

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
1	1	ACTUALITZALINIAATURADA	26/05/2019 15:17:57	L0001,27-03-2018 08:00:00,28-03-2018 14:00:00,2	OK
2	2	crearProduccio	26/05/2019 15:17:57	P0001,L0001,27-03-2018 08:00:00,28-03-2018 14:00:00,50	OK
3	3	ACTUALITZALINIAATURADA	26/05/2019 15:17:57	L0001,28-03-2018 08:00:00,28-04-2018 14:00:00,2	OK
4	4	crearProduccio	26/05/2019 15:17:57	P0001,L0001,28-03-2018 08:00:00,28-04-2018 14:00:00,100	OK
5	5	ACTUALITZALINIAATURADA	26/05/2019 15:17:57	L0002,27-03-2018 08:00:00,27-03-2018 14:00:00,2	OK
6	6	crearProduccio	26/05/2019 15:17:57	P0002,L0002,27-03-2018 08:00:00,27-03-2018 14:00:00,25	OK
7	7	ACTUALITZALINIAATURADA	26/05/2019 15:17:57	L0001,27-03-2018 08:00:00,28-03-2018 14:00:00,1	OK
8	8	ACTUALITZALINIAATURADA	26/05/2019 15:17:57	L0001,27-03-2018 08:00:00,27-03-2018 14:00:00,2	OK
9	9	MODIFICARPRODUCCIO	26/05/2019 15:17:57	P0001,L0001,27-03-2018 08:00:00,27-03-2018 14:00:00,20	OK
10	10	ACTUALITZALINIAATURADA	26/05/2019 15:17:57	L0002,27-03-2018 08:00:00,27-03-2018 14:00:00,1	OK
11	11	ESBORRARPRODUCCIO	26/05/2019 15:17:57	P0002,L0002,27-03-2018 08:00:00	OK

	ANY_	MES	IDLINIA	HORESFUNCIONANT
1	2018	3	L0001	70
2	2018	4	L0001	662

Figura 21 Resultat d'executar el codi de la [Figura 20].

A la taula Produccio hi ha dos registres amb una quantitat total de 120 unitats del producte P0001 tal i com es mostra a la taula ProducteProduit. També es pot comprovar que s'han executat totes les operacions i que les hores de funcionament són les correctes, fins i tot, després de la modificació i esborrat.

Podem confirmar, per tant, que s'han generat les dades de forma coherent entre taules.

A la [Figura 22] es veu el codi per a la generació aleatòria de dades massives utilitzant el procediment crearProduccio.

```
DECLARE
    RSP varchar(200);
BEGIN
    FOR i in 1..50
    LOOP
        --Creació de produccions de producte aleatoris dels 5 primers productes i 5 primeres línies de producció
        -- entre 10 i 48 /24 hores anteriors al dia actual i 10 i 24 /24 dies posteriors al dia actual
        -- i una quantitat entre 10 i 50
        crearProduccio('P'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),'L'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),
                       SYSDATE-ROUND(dbms_random.value(10,48),0)/24,SYSDATE+ROUND(dbms_random.value(10,24),0)/24,
                       ROUND(dbms_random.value(10,50),0),RSP);
        DBMS_OUTPUT.put_line(RSP);
    END LOOP;
END;
```

Figura 22 Codi de generació de dades massives per a la taula Produccio.

El resultat de consultar les mateixes taules es troba a la [Figura 23] (només el final de les taules).

Com s'ha comentat amb anterioritat, hi ha hagut un parell de registres que han generat excepcions per coincidir claus primàries però han sigut tractats adequadament.

	IDPRODUCTE	IDLINIA	DATAINICI	DATAFI	QUANTITATFABRICADA
35	P0003	L0001	25/05/2019 10:24:18	27/05/2019 01:24:18	10
36	P0004	L0002	25/05/2019 16:24:18	27/05/2019 10:24:18	37
37	P0003	L0005	25/05/2019 21:24:18	27/05/2019 15:24:18	16
38	P0004	L0004	25/05/2019 15:24:18	27/05/2019 09:24:18	29
39	P0005	L0004	25/05/2019 15:24:18	27/05/2019 12:24:18	15
40	P0004	L0002	25/05/2019 15:24:18	27/05/2019 13:24:18	10
41	P0002	L0003	25/05/2019 14:24:18	27/05/2019 15:24:18	14
42	P0001	L0001	25/05/2019 02:24:18	27/05/2019 10:24:18	23
43	P0003	L0003	25/05/2019 02:24:18	27/05/2019 03:24:18	28
44	P0005	L0005	26/05/2019 00:24:18	27/05/2019 15:24:18	38
45	P0001	L0001	26/05/2019 00:24:18	27/05/2019 06:24:18	43
46	P0004	L0001	25/05/2019 07:24:18	27/05/2019 01:24:18	30
47	P0004	L0002	24/05/2019 22:24:18	27/05/2019 06:24:18	15
48	P0002	L0003	24/05/2019 16:24:18	27/05/2019 02:24:18	47

	ANY_	IDPRODUCTE	QUANTITAT
1	2019	P0002	376
2	2019	P0003	237
3	2019	P0004	585
4	2019	P0005	113
5	2019	P0001	66

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
86	87	ACTUALITZALINIAATURADA	26/05/2019 15:24:18	L0003,25-05-2019 02:24:18,27-05-2019 03:24:18,2	OK
87	88	crearProduccio	26/05/2019 15:24:18	P0003,L0003,25-05-2019 02:24:18,27-05-2019 03:24:18,28	OK
88	89	ACTUALITZALINIAATURADA	26/05/2019 15:24:18	L0005,26-05-2019 00:24:18,27-05-2019 15:24:18,2	OK
89	90	crearProduccio	26/05/2019 15:24:18	P0005,L0005,26-05-2019 00:24:18,27-05-2019 15:24:18,38	OK
90	91	ACTUALITZALINIAATURADA	26/05/2019 15:24:18	L0001,26-05-2019 00:24:18,27-05-2019 06:24:18,2	OK
91	92	crearProduccio	26/05/2019 15:24:18	P0001,L0001,26-05-2019 00:24:18,27-05-2019 06:24:18,43	OK
92	93	ACTUALITZALINIAATURADA	26/05/2019 15:24:18	L0001,25-05-2019 07:24:18,27-05-2019 01:24:18,2	OK
93	94	crearProduccio	26/05/2019 15:24:18	P0004,L0001,25-05-2019 07:24:18,27-05-2019 01:24:18,30	OK
94	95	ACTUALITZALINIAATURADA	26/05/2019 15:24:18	L0002,24-05-2019 22:24:18,27-05-2019 06:24:18,2	OK
95	96	crearProduccio	26/05/2019 15:24:18	P0004,L0002,24-05-2019 22:24:18,27-05-2019 06:24:18,15	OK
96	97	ACTUALITZALINIAATURADA	26/05/2019 15:24:18	L0003,24-05-2019 16:24:18,27-05-2019 02:24:18,2	OK
97	98	crearProduccio	26/05/2019 15:24:18	P0002,L0003,24-05-2019 16:24:18,27-05-2019 02:24:18,47	OK
98	77	ACTUALITZALINIAATURADA	26/05/2019 15:24:18	L0004,25-05-2019 15:24:18,27-05-2019 09:24:18,2	OK

	ANY_	MES	IDLINIA	HORESFUNCIONANT
1	2019	5	L0001	311
2	2019	5	L0002	524
3	2019	5	L0004	459
4	2019	5	L0003	604
5	2019	5	L0005	217

Figura 23 Resultat de consultar les taules implicades de l'execució del codi de la [Figura 22].

Gestió de Stock

En aquest apartat es verifiquen els procediments crearStock, modificarStock i esborrarStock juntament amb les seves taules associades. A la [Figura 24] tenim el codi associat a les proves individuals d'aquests procediments.

```
--Esborrat de les taules implicades
DELETE Stock;
DELETE capacitatMagatzem;
DELETE LOG;
DELETE capacitatAnualMagatzems;
DELETE begudaProduida;
DELETE empresesDistribuides;

--Consulta de les taules implicades
SELECT * FROM empresa;
SELECT * FROM empresesDistribuides;
SELECT * FROM begudaProduida;
SELECT * FROM capacitatAnualMagatzems;
SELECT * FROM Stock;
SELECT * FROM capacitatMagatzem;
SELECT * FROM magatzem;
SELECT * FROM LOG;

--Instruccions PL/SQL
DECLARE
    RSP varchar(200);
BEGIN
    --Crea tres stocks determinats
    crearStock('S0003','M0001',TO_DATE('10-04-2019 20:10:00','DD-MM-YYYY HH24-MI-SS'),50,6,'T0001',NULL,RSP);
    DBMS_OUTPUT.put_line(RSP);
    crearStock('S0003','M0002',TO_DATE('09-04-2019 20:10:00','DD-MM-YYYY HH24-MI-SS'),50,6,'T0001',NULL,RSP);
    DBMS_OUTPUT.put_line(RSP);
    crearStock('S0003','M0003',TO_DATE('08-04-2019 20:10:00','DD-MM-YYYY HH24-MI-SS'),100,6,'T0001',NULL,RSP);
    DBMS_OUTPUT.put_line(RSP);
    crearStock('S0003','M0001',TO_DATE('11-04-2019 20:10:00','DD-MM-YYYY HH24-MI-SS'),-20,-6,'T0001','EP001',RSP);
    DBMS_OUTPUT.put_line(RSP);
    --Modifica un dels stocks anteriors
    modificarStock('S0003','M0002',TO_DATE('09-04-2019 20:10:00','DD-MM-YYYY HH24-MI-SS'),80,6,'T0001',NULL,RSP);
    DBMS_OUTPUT.put_line(RSP);
    --Esborrra un dels stocks anteriors
    esborrarStock('S0003','M0003',TO_DATE('08-04-2019 20:10:00','DD-MM-YYYY HH24-MI-SS'),RSP);
    DBMS_OUTPUT.put_line(RSP);
END;
```

Figura 24 Codi associat a les proves individuals de gestió de Stock.

El resultat d'executar el codi anterior es pot observar a la [Figura 25] on es mostren els resultats de consultar les taules implicades (Stock, empresesDistribuides, begudaProduida, capacitatAnualMagatzems, Log).

Podem veure que s'han generat i rectificat les dades correctament.

	IDSISTEMA	IDMAGATZEM	DATA	VOLUM	QUANTITAT	IDTRANSPORT	IDEMPRESA
1	S0003	M0001	10/04/2019 20:10:00	50	6	T0001	(null)
2	S0003	M0002	09/04/2019 20:10:00	80	6	T0001	(null)
3	S0003	M0001	11/04/2019 20:10:00	-20	-6	T0001	EP001

	IDEMPRESA	IDPRODUCTE	VOLUM
1	EP001	P0002	0,9

	ANY_	CAPACITATLITRES
1	2019	1,8

	ANY_	ESPAIOCUPAT	ESPAIMAXIMOCUPAT
1	2019	110	110

	IDMAGATZEM	ESPAIOCUPAT
1	M0001	30
2	M0002	80

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
1	1	CREARSTOCK	26/05/2019 15:54:48	S0003,M0001,10-04-2019 20:10:00,50,6,T0001,	OK
2	2	CREARSTOCK	26/05/2019 15:54:48	S0003,M0002,09-04-2019 20:10:00,50,6,T0001,	OK
3	3	CREARSTOCK	26/05/2019 15:54:48	S0003,M0003,08-04-2019 20:10:00,100,6,T0001,	OK
4	4	CREARSTOCK	26/05/2019 15:54:48	S0003,M0001,11-04-2019 20:10:00,-20,-6,T000...	OK
5	5	MODIFICARSTOCK	26/05/2019 15:54:48	S0003,M0002,09-04-2019 20:10:00,80,6,T0001,	OK
6	6	ESBORRARSTOCK	26/05/2019 15:54:48	S0003,M0003,08-04-2019 20:10:00	OK

Figura 25 Resultat d'executar el codi de la [Figura 24].

A la [Figura 26] es troba el codi de generació de dades aleatòries de forma massiva.

Es troba dividit en dos apartats, la creació de dades d'entrada als magatzems i la sortida o distribució d'aquests per les empreses distribuïdores.

```
DECLARE
    RSP varchar(200);
BEGIN
    --Bucle de 1 fins a 100
    FOR i in 1..100
    LOOP
        --Crea un stock amb un dels 5 primers registres de Sistema de forma aleatòria, un dels 5 primers registres de Magatzem aleatoris,
        -- a una data de i dies abans d'avui, amb un volum aleatori entre el 1 i 10, una quantitat entre el 1 i el 50 i un transport
        -- aleatori entre els 5 primers
        crearStock('S'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),'M'||
        LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),SYSDATE-i,dbms_random.value(1,10),dbms_random.value(1,50),
        'T'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),NULL,RSP);
        DBMS_OUTPUT.put_line(RSP);
        --És el mateix, l'únic que la data és just d'un any anterior
        crearStock('S'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),'M'||
        LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),ADD_MONTHS(SYSDATE,-12)-i,dbms_random.value(1,10),dbms_random.value(1,50),
        'T'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),NULL,RSP);
        DBMS_OUTPUT.put_line(RSP);
    END LOOP;

    --Bucle de 1'1 al 20
    FOR i in 1..20
    LOOP
        --Fa el mateix que els anteriors, només que la quantitat és negativa i s'informa d'una empresa entre les 8 primeres aleatòriament
        crearStock('S'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),'M'||
        LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),SYSDATE-i,-dbms_random.value(1,10),-dbms_random.value(1,10),
        'T'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),'EP'||LPAD(ROUND(dbms_random.value(1,8),0),3,'0'),RSP);
        DBMS_OUTPUT.put_line(RSP);
        --El mateix que l'anterior però amb data d'un any enrere
        crearStock('S'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),'M'||
        LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),ADD_MONTHS(SYSDATE,-12)-i,-dbms_random.value(1,10),-dbms_random.value(1,10),
        'T'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),'EP'||LPAD(ROUND(dbms_random.value(1,8),0),3,'0'),RSP);
        DBMS_OUTPUT.put_line(RSP);
    END LOOP;
END;
```

Figura 26 Codi de generació de dades aleatòries de forma massiva per a la gestió de Stock.

A la [Figura 27] es pot observar una mostra de les dades creades amb el codi anterior seguint el mateix ordre de taules consultades.

	IDSISTEMA	IDMAGATZEM	DATA	VOLUM	QUANTITAT	IDTRANSPORT	IDEMPRESA
197	S0004	M0004	16/02/2019 16:02:30	9,21	9	T0002	(null)
198	S0002	M0004	16/02/2018 16:02:30	5,63	30	T0001	(null)
199	S0003	M0004	15/02/2019 16:02:30	5,71	34	T0002	(null)
200	S0003	M0002	15/02/2018 16:02:30	2,59	3	T0004	(null)
201	S0003	M0004	25/05/2019 16:02:30	-5,04	-7	T0004	EP007
202	S0003	M0004	25/05/2018 16:02:30	-4,55	-3	T0001	EP006
203	S0002	M0002	24/05/2019 16:02:30	-4,32	-7	T0003	EP003
204	S0003	M0002	24/05/2018 16:02:30	-3,6	-3	T0004	EP007
205	S0001	M0004	23/05/2019 16:02:30	-3,9	-8	T0004	EP004
206	S0003	M0003	22/05/2019 16:02:30	-9,33	-8	T0002	EP007
207	S0002	M0004	22/05/2018 16:02:30	-4,95	-10	T0003	EP001

	IDEMPRESA	IDPRODUCTE	VOLUM
1	EP007	P0002	0,924
2	EP006	P0002	0,379
3	EP003	P0001	0,062
4	EP004	P0001	0,482
5	EP001	P0001	0,231
6	EP005	P0002	1,253
7	EP002	P0002	0,589
8	EP008	P0003	3,056

	ANY_	CAPACITATLITRES
1	2019	660,58
2	2018	526,177

	ANY_	ESPAIOCUPAT	ESPAIMAXIMOCUPAT
1	2019	437,82	554,36
2	2018	418,55	536,13

	IDMAGATZEM	ESPAIOCUPAT
1	M0002	246,08
2	M0003	273,65
3	M0004	183,63
4	M0005	75,76
5	M0001	77,25

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
104	104	CREARSTOCK	26/05/2019 16:02:30	S0005,M0002,04-04-2018 16:02:30,2,4141416721...	OK
105	105	CREARSTOCK	26/05/2019 16:02:30	S0005,M0004,03-04-2019 16:02:30,4,889758108...	OK
106	106	CREARSTOCK	26/05/2019 16:02:30	S0005,M0002,03-04-2018 16:02:30,8,864136453...	OK
107	107	CREARSTOCK	26/05/2019 16:02:30	S0003,M0003,02-04-2019 16:02:30,2,599462799...	OK
108	108	CREARSTOCK	26/05/2019 16:02:30	S0002,M0002,02-04-2018 16:02:30,8,133371712...	OK
109	109	CREARSTOCK	26/05/2019 16:02:30	S0003,M0003,01-04-2019 16:02:30,4,811762788...	OK
110	110	CREARSTOCK	26/05/2019 16:02:30	S0004,M0004,01-04-2018 16:02:30,7,615863124...	OK
111	111	CREARSTOCK	26/05/2019 16:02:30	S0003,M0002,31-03-2019 16:02:30,1,802922410...	OK
112	112	CREARSTOCK	26/05/2019 16:02:30	S0002,M0001,31-03-2018 16:02:30,5,585122229...	OK
113	113	CREARSTOCK	26/05/2019 16:02:30	S0003,M0003,30-03-2019 16:02:30,6,586127444...	OK

Figura 27 Mostra de dades generades a partir de l'execució del codi de la [Figura 26]

Gestió d'assignacions de torns

En aquesta secció es comprovarà el funcionament dels procediments crearAssignacio i esborrarAssignacio i les seves taules implicades.

A la [Figura 28] es mostra el codi per a la comprovació individual d'aquests procediments.

```
--Esborrat de les taules implicades
DELETE Assignacio;
DELETE empleatsUtilitzats;
DELETE LOG;
DELETE tornsAnuals;

--Consulta de les taules implicades
SELECT * FROM empleat;
SELECT * FROM torn;
SELECT * FROM Assignacio;
SELECT * FROM tornsAnuals;
SELECT * FROM empleatsUtilitzats;
SELECT * FROM LOG;

DECLARE
    RSP varchar(200);
BEGIN
    --Creació d'Assignacions
    crearAssignacio('EM001','T0001',SYSDATE,RSP);
    DBMS_OUTPUT.put_line(RSP);
    crearAssignacio('EM002','T0002',SYSDATE,RSP);
    DBMS_OUTPUT.put_line(RSP);
    --Esborrat d'un dels anteriors registres
    esborrarAssignacio('EM001','T0001',RSP);
    DBMS_OUTPUT.put_line(RSP);
END;
```

Figura 28 Codi associat a les proves individuals dels procediments de gestió d'assignacions.

Si executem el codi anterior trobarem unes dades similars a les que es mostren a la [Figura 29] a les taules pertinents (assignació, tornsAnuals, empleatsUtilitzats, Log).

	IDEMPLEAT	IDTORN	DATA
1	EM002	T0002	26/05/2019 17:17:59

ANY_	TORNSTO...	TORNSCO...

	ANY_	IDEMPLEAT	NOMBRETORNS
1	2019	EM002	1

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
1	1	CREARASSIGNACIO	26/05/2019 17:17:59	EM001,T0001,26-05-2019 17:17:59	OK
2	2	CREARASSIGNACIO	26/05/2019 17:17:59	EM002,T0002,26-05-2019 17:17:59	OK
3	3	ESBORRARASSIGNACIO	26/05/2019 17:17:59	EM001,T0001,26-05-2019 17:17:59	OK

Figura 29 Resultat d'executar el codi de la [Figura 28].

Els tornsAnuals no tenen registre perquè el procediment esborrarAssignacio elimina expressament els registres que no tenen torns complets.

A la [Figura 30] es mostra el codi per la generació aleatòria de dades massives a partir del procediment crearAssignacio.

```
--Instruccions PL/SQL
DECLARE
    RSP varchar(200);
BEGIN
    --Bucle de 0 a 100
    FOR i in 0..100
    LOOP
        --Crea una assignació d'un empleat aleatori entre els 15 primers amb un torn dels 4 primers
        -- a una data del dia actual menys i (nombre del bucle)
        crearAssignacio('EM'||LPAD(ROUND(dbms_random.value(1,15),0),3,'0'),'T'||
        LPAD(ROUND(dbms_random.value(1,4),0),4,'0'),SYSDATE-i,RSP);
        DBMS_OUTPUT.put_line(RSP);
        --El mateix que en cas anterior però la data és d'un any abans
        crearAssignacio('EM'||LPAD(ROUND(dbms_random.value(1,15),0),3,'0'),'T'||
        LPAD(ROUND(dbms_random.value(1,4),0),4,'0'),ADD_MONTHS(SYSDATE,-12)-i,RSP);
        DBMS_OUTPUT.put_line(RSP);
    END LOOP;
END;
```

Figura 30 Codi per a la generació aleatòria de dades massives de la gestió d'assignacions.

A la [Figura 31] es pot observar una mostra de cadascuna de les taules implicades en el mateix ordre d'abans.

	IDEMPLEAT	IDTORN	DATA
192	EM008	T0002	05/05/2019 17:21:48
193	EM008	T0002	18/05/2019 17:21:48
194	EM008	T0003	20/02/2018 17:21:48
195	EM008	T0003	28/02/2018 17:21:48
196	EM008	T0003	01/03/2018 17:21:48
197	EM008	T0003	07/04/2018 17:21:48
198	EM008	T0003	11/04/2018 17:21:48
199	EM008	T0003	16/04/2018 17:21:48
200	EM008	T0003	06/05/2018 17:21:48

	ANY_	TORNSTOTALS	TORNSCOMPLERTS
1	2019	5	3
2	2018	5	1

	ANY_	IDEMPLEAT	NOMBRETORNS
20	2019	EM002	7
21	2018	EM011	9
22	2019	EM009	7
23	2018	EM004	10
24	2018	EM014	8
25	2019	EM015	2
26	2018	EM003	7
27	2019	EM001	2
28	2019	EM004	5
29	2019	EM010	5
30	2018	EM001	1

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
142	142	CREARASSIGNACIO	26/05/2019 17:21:48	EM006,T0003,17-03-2018 17:21:48	OK
143	143	CREARASSIGNACIO	26/05/2019 17:21:48	EM009,T0002,16-03-2019 17:21:48	OK
144	144	CREARASSIGNACIO	26/05/2019 17:21:48	EM004,T0003,16-03-2018 17:21:48	OK
145	145	CREARASSIGNACIO	26/05/2019 17:21:48	EM012,T0003,15-03-2019 17:21:48	OK
146	146	CREARASSIGNACIO	26/05/2019 17:21:48	EM005,T0004,15-03-2018 17:21:48	OK
147	147	CREARASSIGNACIO	26/05/2019 17:21:48	EM010,T0003,14-03-2019 17:21:48	OK
148	148	CREARASSIGNACIO	26/05/2019 17:21:48	EM003,T0003,14-03-2018 17:21:48	OK
149	149	CREARASSIGNACIO	26/05/2019 17:21:48	EM007,T0002,13-03-2019 17:21:48	OK
150	150	CREARASSIGNACIO	26/05/2019 17:21:48	EM007,T0003,13-03-2018 17:21:48	OK
151	151	CREARASSIGNACIO	26/05/2019 17:21:48	EM005,T0002,12-03-2019 17:21:48	OK

Figura 31 Resultat de consultar les taules implicades després de l'execució del codi de la [Figura 30].

Gestió de la demanda

En aquesta secció es comprovaran els procediments de crearDemanda, modificarDemanda i esborrarDemanda i les seves taules associades.

```
--Esborrat de les taules implicades
DELETE Demanda;
DELETE DemandaAnual;
DELETE LOG;

--Consulta de les taules implicades
SELECT * FROM Producte;
SELECT * FROM ZONADISTRIBUCIO;
SELECT * FROM Demanda;
SELECT * FROM DemandaAnual;
SELECT * FROM LOG;

DECLARE
    RSP varchar(200);
BEGIN
    --Creació de demanda
    crearDemanda('P0001','Z0001',SYSDATE,20,RSP);
    DBMS_OUTPUT.put_line(RSP);
    crearDemanda('P0002','Z0001',SYSDATE,20,RSP);
    DBMS_OUTPUT.put_line(RSP);
    --Modificació de demanda
    modificarDemanda('P0002','Z0001',SYSDATE,40,RSP);
    DBMS_OUTPUT.put_line(RSP);
    --Esborrat de demanda
    esborrarDemanda('P0001','Z0001',SYSDATE,RSP);
    DBMS_OUTPUT.put_line(RSP);
END;
```

Figura 32 Codi associat a les proves individuals dels procediments de gestió de demanda.

	IDPRODUCTE	IDZONA	DATA	QUANTITAT
1	P0002	Z0001	26/05/2019 18:16:23	40

	ANY_	IDPRODUCTE	QUANTITAT	QUANTITATDIES
1	2019	P0002	40	1

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
1	1	CREARDEMANDA	26/05/2019 18:16:23	P0001,Z0001,26-05-2019 18:16:23,20	OK
2	2	CREARDEMANDA	26/05/2019 18:16:23	P0002,Z0001,26-05-2019 18:16:23,20	OK
3	3	MODIFICARDEMANDA	26/05/2019 18:16:23	P0002,Z0001,26-05-2019 18:16:23,40	OK
4	4	ESBORRARDEMANDA	26/05/2019 18:16:23	P0001,Z0001,26-05-2019 18:16:23	OK

Figura 33 Resultat de consultar les taules afectades per l'execució del codi de la [Figura 32].

A les anteriors figures es comprova el funcionament correcte de l'execució dels procediments comentats i les dades generades a les taules demanda, demandaAnual i Log.

```
--Instruccions PL/SQL
DECLARE
    RSP varchar(200);
BEGIN
    --Bucle que va de 0 a 100
    FOR i in 0..100
    LOOP
        --Crea una demanda d'un dels 5 primers productes aleatòriament, d'una de les 5 primeres zones aleatòriament,
        -- de la data actual menys i (nombre del bucle) dies, amb una quantitat entre 1 i 50
        crearDemanda('P'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),'Z'||
        LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),SYSDATE-i,dbms_random.value(1,50),RSP);
        DBMS_OUTPUT.put_line(RSP);
        --El mateix que abans però d'un any anterior
        crearDemanda('P'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),'Z'||
        LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),add_months(SYSDATE,-12)-i,dbms_random.value(1,50),RSP);
        DBMS_OUTPUT.put_line(RSP);
    END LOOP;
END;
```

Figura 34 Codi de generació aleatòria massiva de dades per la gestió de Demanda.

	IDPRODUCTE	IDZONA	DATA	QUANTITAT
82	P0003	Z0005	16/04/2018 18:20:26	35
83	P0004	Z0003	15/04/2019 18:20:26	10
84	P0004	Z0003	15/04/2018 18:20:26	42
85	P0001	Z0004	14/04/2019 18:20:26	8
86	P0003	Z0002	14/04/2018 18:20:26	7
87	P0002	Z0004	13/04/2019 18:20:26	4
88	P0001	Z0001	13/04/2018 18:20:26	17
89	P0004	Z0001	12/04/2019 18:20:26	26
90	P0003	Z0003	12/04/2018 18:20:26	30
91	P0004	Z0002	11/04/2019 18:20:26	35

	ANY_	IDPRODUCTE	QUANTITAT	QUANTITATDIES
1	2019	P0001	472	18
2	2018	P0004	626	20
3	2019	P0002	736	28
4	2018	P0005	403	21
5	2019	P0003	689	24
6	2018	P0001	332	15
7	2018	P0002	530	21
8	2019	P0004	486	20
9	2018	P0003	594	24
10	2019	P0005	278	11

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
16	16	CREARDEMANDA	26/05/2019 18:20:26	P0004,Z0004,19-05-2018 18:20:26,24,88703018...	OK
17	17	CREARDEMANDA	26/05/2019 18:20:26	P0002,Z0003,18-05-2019 18:20:26,27,52620169...	OK
18	18	CREARDEMANDA	26/05/2019 18:20:26	P0003,Z0005,18-05-2018 18:20:26,30,53033253...	OK
19	19	CREARDEMANDA	26/05/2019 18:20:26	P0001,Z0004,17-05-2019 18:20:26,29,79181789...	OK
20	20	CREARDEMANDA	26/05/2019 18:20:26	P0003,Z0004,17-05-2018 18:20:26,46,86003092...	OK
21	21	CREARDEMANDA	26/05/2019 18:20:26	P0004,Z0005,16-05-2019 18:20:26,49,78372010...	OK
22	22	CREARDEMANDA	26/05/2019 18:20:26	P0003,Z0002,16-05-2018 18:20:26,11,15688934...	OK
23	23	CREARDEMANDA	26/05/2019 18:20:26	P0004,Z0002,15-05-2019 18:20:26,19,04455865...	OK
24	24	CREARDEMANDA	26/05/2019 18:20:26	P0005,Z0004,15-05-2018 18:20:26,4,692028092...	OK
25	25	CREARDEMANDA	26/05/2019 18:20:26	P0001,Z0003,14-05-2019 18:20:26,5,874048117...	OK

Figura 35 Resultat de consultar les dades generades pel codi de la [Figura 34].

A la [Figura 34] i [Figura 35] es mostra el codi de generació aleatòria massiva de dades de gestió de demanda i la consulta de les dades generades a les taules en el mateix ordre que abans, respectivament.

Gestió de netejament

En aquest apartat es verifiquen els procediments crearNetejament, modificarNetejament i esborrarNetejament juntament amb les taules implicades.

```
--Esborrat de les taules implicades
DELETE netejaAnual;
DELETE LOG;
DELETE netejament;

--Consulta de les taules implicades
SELECT * FROM neteja;
SELECT * FROM liniaFabricacio;
SELECT * FROM netejaAnual;
SELECT * FROM netejament;
SELECT * FROM LOG;

DECLARE
    RSP varchar(200);
BEGIN
    --Creació de netejament
    crearNetejament('N0001','L0001',SYSDATE,SYSDATE+20/24,RSP);
    DBMS_OUTPUT.put_line(RSP);
    crearNetejament('N0002','L0002',SYSDATE,SYSDATE+20/24,RSP);
    DBMS_OUTPUT.put_line(RSP);
    --Modificació de netejament
    modificarNetejament('N0001','L0001',SYSDATE,SYSDATE+10/24,RSP);
    DBMS_OUTPUT.put_line(RSP);
    --Esborrat de netejament
    esborrarNetejament('N0002','L0002',SYSDATE,RSP);
    DBMS_OUTPUT.put_line(RSP);
END;
```

Figura 36 Codi associat a la comprovació individuals dels procediments de gestió de netejament.

	ANY_	IDLINIA	HORES
1	2019	L0001	10

	IDNETEJA	IDLINIA	DATAINICI	DATAFI
1	N0001	L0001	26/05/2019 18:26:17	27/05/2019 04:26:17

	IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
1	1	CREARNETEJAMENT	26/05/2019 18:26:17	N0001,L0001,26-05-2019 18:26:17,27-05-2019 ...	OK
2	2	CREARNETEJAMENT	26/05/2019 18:26:17	N0002,L0002,26-05-2019 18:26:17,27-05-2019 ...	OK
3	3	MODIFICARNETEJAMENT	26/05/2019 18:26:17	N0001,L0001,26-05-2019 18:26:17,27-05-2019 ...	OK
4	4	ESBORRARNETEJAMENT	26/05/2019 18:26:17	N0002,L0002,26-05-2019 18:26:17	OK

Figura 37 Resultat de consultar les taules implicades després de l'execució del codi de la [Figura 36].

Les taules consultades a la [Figura 37] són netejaAnual, netejament i Log en aquest ordre.

```
--Instruccions PL/SQL
DECLARE
    RSP varchar(200);
BEGIN
    --Bucle que va de 0 a 100
    FOR i in 0..100
    LOOP
        --Crea un netejament d'un dels 5 primers grups de neteja aleatòriament, d'una de les 5 primeres línies de producció,
        -- d'una data d'inici d'avui menys i (valor de bucle) dies i de finalització unes hores aleatòries després (es passen a dies)
        crearNetejament('N'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),'L'||
        LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),SYSDATE-i,SYSDATE-i+dbms_random.value(1,24)/24,RSP);
        DBMS_OUTPUT.put_line(RSP);
        --El mateix que abans però les dates són d'un any anterior.
        crearNetejament('N'||LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),'L'||
        LPAD(ROUND(dbms_random.value(1,5),0),4,'0'),ADD_MONTHS(SYSDATE,-12)-i,ADD_MONTHS(SYSDATE,-12)-i+dbms_random.value(1,24)/24,RSP);
        DBMS_OUTPUT.put_line(RSP);
    END LOOP;
END;
```

Figura 38 Codi de generació automàtica massiva de dades per la gestió de netejament.

	ANY_	IDLINIA	HORES
1	2019	L0004	283,25
2	2018	L0003	309,85
3	2018	L0002	409,53
4	2018	L0004	339,42
5	2019	L0003	263,95
6	2019	L0002	345,43
7	2019	L0005	147,17
8	2018	L0005	163,46
9	2019	L0001	135,85
10	2018	L0001	88,38

	IDNETEJA	IDLINIA	DATAINICI	DATAFI
43	N0002	L0005	30/04/2019 18:40:19	01/05/2019 01:19:09
44	N0002	L0004	30/04/2018 18:40:19	30/04/2018 22:07:18
45	N0005	L0002	29/04/2019 18:40:19	30/04/2019 12:41:11
46	N0004	L0001	29/04/2018 18:40:19	30/04/2018 04:49:57
47	N0004	L0005	28/04/2019 18:40:19	29/04/2019 01:46:08
48	N0004	L0003	28/04/2018 18:40:19	29/04/2018 12:27:52
49	N0002	L0005	27/04/2019 18:40:19	28/04/2019 03:49:49
50	N0004	L0004	27/04/2018 18:40:19	28/04/2018 10:27:05
51	N0005	L0004	26/04/2019 18:40:19	27/04/2019 06:28:57
52	N0002	L0002	26/04/2018 18:40:19	27/04/2018 03:35:04
53	N0005	L0002	25/04/2019 18:40:19	26/04/2019 08:45:08

IDLOG	NOMPROC	DATAEXECUCIO	ENTRADA	SORTIDA
116	CREARNETEJAMENT	26/05/2019 18:40:19	N0003,L0003,30-03-2018 18:40:19,31-03-2018 ...	OK
117	CREARNETEJAMENT	26/05/2019 18:40:19	N0004,L0004,29-03-2019 18:40:19,30-03-2019 ...	OK
118	CREARNETEJAMENT	26/05/2019 18:40:19	N0005,L0003,29-03-2018 18:40:19,30-03-2018 ...	OK
119	CREARNETEJAMENT	26/05/2019 18:40:19	N0002,L0003,28-03-2019 18:40:19,29-03-2019 ...	OK
120	CREARNETEJAMENT	26/05/2019 18:40:19	N0002,L0005,28-03-2018 18:40:19,28-03-2018 ...	OK
121	CREARNETEJAMENT	26/05/2019 18:40:19	N0001,L0004,27-03-2019 18:40:19,28-03-2019 ...	OK
122	CREARNETEJAMENT	26/05/2019 18:40:19	N0004,L0003,27-03-2018 18:40:19,28-03-2018 ...	OK
123	CREARNETEJAMENT	26/05/2019 18:40:19	N0003,L0003,26-03-2019 18:40:19,26-03-2019 ...	OK
124	CREARNETEJAMENT	26/05/2019 18:40:19	N0002,L0004,26-03-2018 18:40:19,27-03-2018 ...	OK

Figura 39 Resultat de consultar les taules implicades en la gestió de netejament després de l'execució del codi de la [Figura 38].

Finalment, amb la [Figura 38] i [Figura 39] tenim la comprovació de la generació de dades aleatòries automàtiques massives per omplir les dades de gestió de netejament.

Proves en les consultes dels requisits de client

Aquest apartat és, bàsicament, l'execució de les consultes amb les dades generades anteriorment i la comprovació que el resultat concorda correctament amb la seva definició.

Component més utilitzat

Si executem la consulta del component més utilitzat amb les dades actuals, ens donarà el resultat de la [Figura 40]. El segon apartat de la imatge es mostra la subconsulta on es troben tots els components utilitzats per ordre de participació en compostos.

	IDCOMPONENT	NOMBRECOMPOSTS
1	C0002	5

	IDCOMPONENT	NOMBRECOMPOSTS
1	C0002	5
2	C0004	3
3	C0003	3
4	C0001	2
5	C0005	1

Figura 40 Resultat de la consulta de component més utilitzat i llista de tots els components participants a compostos.

Producte més produït

En aquest cas, la consulta s'ha fet sobre l'any 2019. La segona part de la [Figura 41] mostra la llista de tots els productes produïts aquell any.

	IDPRODUCTE	QUANTITAT
1	P0004	585

	IDPRODUCTE	QUANTITAT
1	P0004	585
2	P0002	376
3	P0003	237
4	P0005	113
5	P0001	66

Figura 41 Resultat de la consulta de producte més produït i llistat de productes produïts per l'any 2019.

Línia més aturada

El primer resultat és l'execució de la consulta per l'any 2019 i el mes 5 (maig). El segon resultat és la subconsulta que mostra totes les línies ordenades per les seves hores aturades ordenades descendentment.

	IDLINIA	HORESFUNCIONANT	HORESATURADA
1	L0005	217	527

	IDLINIA	ANY_	MES	HORESFUNCIONANT	HORESATURADA
1	L0005	2019	5	217	527
2	L0001	2019	5	311	433
3	L0004	2019	5	459	285
4	L0002	2019	5	524	220
5	L0003	2019	5	604	140

Figura 42 Resultat de la consulta de la línia més aturada i la subconsulta amb l'ordenació de les línies.

Capacitat emmagatzematge actual magatzem

En aquesta consulta es tria el magatzem M0001 i es consulta quina és la seva capacitat, espai ocupat i espai disponible en el moment de consulta.

	IDMAGATZEM	ESPAIOCUPAT	CAPACITAT	ESPAIDISPONIBLE
1	M0001	77,25	500	422,75

Figura 43 Resultat de la consulta de capacitat emmagatzematge actual d'un magatzem concret.

Top 10 empleats més utilitzats

La [Figura 44] mostra els deu primers empleats que més han sigut assignats a torns per l'any 2019. He creat quinze empleats i si s'executa la subconsulta es veurà que tots tenen algun torn assignat (amb les dades actuals).

	IDEMPLEAT	NOMBRETORNS
1	EM011	12
2	EM007	10
3	EM003	8
4	EM014	8
5	EM012	8
6	EM006	8
7	EM002	7
8	EM009	7
9	EM005	7
10	EM008	6

Figura 44 Resultat de la consulta dels empleats més utilitzats.

Increment percentual de demanda anual

La [Figura 45] mostra que la demanda anual pel 2019 ha augmentat aproximadament un 4% respecte l'any anterior pel producte P0002. Si es volen retallar els decimals només s'hauria d'utilitzar la comanda ROUND.

	DEMANDAMITJANYANTERIOR	DEMANDAMITJANAANYACTUAL	INCRDECRPERCENTANYACTUAL
1	25,238095238095238095238095238095238095	26,285714285714285714285714285714285714	4,150943396226415094339622641509433962

Figura 45 Resultat de la consulta d'increment percentual de demanda anual per un producte concret.

Any menor capacitat emmagatzematge

La [Figura 46] mostra l'any on la suma de la capacitat dels magatzems ha sigut menor en un any concret. El següent resultat mostra les dades pels dos anys pels que he creat les dades.

	ANY_	ESPAIOCUPAT
1	2018	418,55

	ANY_	ESPAIOCUPAT
1	2018	418,55
2	2019	437,82

Figura 46 Resultat de la consulta de l'any amb menor capacitat d'emmagatzematge i la seva subconsulta.

Increment percentual de beguda produïda

En aquest cas, la [Figura 47] mostra la relació de beguda produïda l'any actual (2019) respecte l'anterior.

	CAPACITATLITRESANYANTERIOR	CAPACITATLITRESANYACTUAL	INCRDECRPERCENTANYACTUAL
1	526,177	660,58	25,543305769731478190798913673535711367 09

Figura 47 Resultat d'executar la consulta d'increment percentual de beguda produïda.

Any amb major temps de neteja

En aquest cas, la consulta mostra la línia que ha necessitat més temps de neteja en comparació amb les altres d'altres o el mateix any, en concret, l'any 2019.

	IDLINIA	HORES	ANY_
1	L0002	345,43	2019

	IDLINIA	HORES	ANY_
1	L0002	345,43	2019
2	L0004	283,25	2019
3	L0003	263,95	2019
4	L0005	147,17	2019
5	L0001	135,85	2019

Figura 48 Resultat de la consulta de la línia amb major temps de neteja per un any concret i la seva subconsulta.

Percentatge de torns incomplerts

El resultat de la següent figura mostra el nombre de torns per l'any 2019, els que es troben incomplerts i el percentatge que representen aquests.

	TORNSTOTALS	TORNSCOMPLERTS	PERCENTATGETORNSINCOMPLERTS
1	5	3	40

Figura 49 Resultat de la consulta de percentatge de torns incomplerts per un any en concret.

Top 5 empreses de distribució

A la [Figura 50] podem comprovar quines són les 5 primeres empreses distribuïdores i també la resta d'empreses distribuïdores participants fins el moment (subconsulta).

	IDEMPRESA	IDPRODUCTE	VOLUM
1	EP008	P0003	3,056
2	EP005	P0002	1,253
3	EP007	P0002	0,924
4	EP002	P0002	0,589
5	EP004	P0001	0,482

	IDEMPRESA	IDPRODUCTE	VOLUM
1	EP008	P0003	3,056
2	EP005	P0002	1,253
3	EP007	P0002	0,924
4	EP002	P0002	0,589
5	EP004	P0001	0,482
6	EP006	P0002	0,379
7	EP001	P0001	0,231
8	EP003	P0001	0,062

Figura 50 Resultat de la consulta de les 5 primeres empreses distribuïdores i de la seva subconsulta.

6. Seguiment de la planificació

En aquest apartat descric com ha sigut el desenvolupament d'aquest treball descrivint només, els fets més importants de cada tasca i subtasca.

6.1 PAC 1: Pla de treball (21/02/2019 – 04/03/2019)

Recollida i anàlisis de Requisits

Vaig començar llegint l'enunciat del treball i repassant el temari de les assignatures relacionades amb les bases de dades i de gestió de projectes.

A l'assignatura de gestió de projectes es divideix el projecte en els apartats d'iniciació, planificació, execució, seguiment i control i, finalment, tancament.

Com que només es treballen els tres primers apartats anteriors, el treball és en realitat més ampli (incorpora la creació d'una aplicació) i només ens centrem en desenvolupar exclusivament la part de la base de dades, vaig considerar que la millor estructuració per la memòria seria les fases de disseny d'una base de dades que sempre es compleixen en la creació d'una base de dades relacional.

Objectius i Planificació

En la planificació del treball, vaig tenir en compte les 300 hores de dedicació que es demanen. Com que les entregues parcials es trobaven ben definides, només calia repartir aquestes hores proporcionalment als períodes que ocupaven cada PAC.

Pel que fa al diagrama de Gantt, tenia clar des de bon principi que GanttProject seria l'eina que utilitzaria. Tot i que vaig tenir certs inconvenients per poder crear la columna d'hores i d'organitzar certes subtasques dins el Gantt, en aquest període no vaig tenir grans impediments i vaig seguir la planificació.

6.2 PAC 2: Disseny Conceptual (05/03/2019 – 08/04/2019)

Conceptualització

Segons la planificació, aquest apartat el vaig centrar al disseny conceptual. El primer que vaig fer era trobar les diferents entitats que participaven en el treball i la seva relació entre aquestes.

Un cop les vaig tenir clares, vaig crear-ne d'altres indirectes per reforçar les principals o per respondre a les necessitats del client com, per exemple, donar resposta a les seves consultes o optimitzar l'ús de les entitats.

Aquest és l'apartat on més dubtes em van sorgir i el que millor havia d'elaborar ja que es tracta de la base del treball i una incorrecta decisió de disseny al principi pot implicar invertir molt més temps en una rectificació posterior quan el projecte es troba més avançat.

El primer gran inconvenient amb el que em vaig trobar era com implementar les entitats SistemaEmbalatge i Transport amb la resta, especialment amb Producte, Magatzem i Empresa.

Primerament no vaig tenir en compte el SistemaEmbalatge i el Transport per desconeixença a què es referien i com funcionaven conjuntament amb la resta del sistema.

Per altra banda, desconeixia on guardar dades que em permetessin resoldre consultes com les 5 primeres empreses de distribució o les que tracten sobre l'emmagatzematge útil.

Un cop resolts els dubtes, i de la iteració de fer proves en apartats posteriors, vaig trobar que el diagrama actual resolia totes les especificacions.

Diagrama UML, ER

Per poder dissenyar aquest diagrama d'Entitats i Relacions m'hagués agradat utilitzar MagicDraw, una eina que em trobava habituat a utilitzar per la creació d'aquests diagrames conceptuals.

Com que MagicDraw té licència propietària i l'edició que disposava es trobava limitada per ser de prova, vaig haver de cercar alternatives. A raó d'això, vaig trobar l'eina en línia draw.io que oferia les suficients funcionalitats pel disseny del diagrama del treball.

Draw.io no té les mateixes facilitats que MagicDraw a l'hora de crear el diagrama així que moltes funcionalitats les he hagut de simular i fer manualment però el resultat va resultar prou adequat.

En aquest apartat també es trobava la polèmica de realitzar un diagrama Entitat-Relació, l'híbrid que es troba en els materials d'alguna assignatura o el diagrama relacional que vaig acabar escollint i que les raons d'elecció es troben explicades a l'apartat de disseny conceptual.

Documentació PAC 2

Com que el diagrama s'ha trobat gairebé en canvi constant fins al final del treball, aquesta secció de documentació ha anat variant conseqüentment.

Tot i així, en aquest període les entitats principals ja es trobaven consolidades i les entitats associatives que utilitzo per la majoria de consultes també. Els petits retocs tenen a veure amb les relacions entre aquestes entitats i els seus atributs.

6.3 PAC 3: Disseny Lògic i Físic (09/04/2019 – 06/05/2019)

Tot i que es va planificar el inici d'aquesta tasca el dia 09/04/2019, l'esquema del disseny conceptual ja el tenia acabat abans i vaig decidir que era millor

avançar i comprovar la solidesa d'aquest disseny i així corregir possibles obstacles amb major marge de temps.

Per aquesta raó, la seva data d'inici va ser, en realitat, al voltant del dia 26 de març de 2019.

Traducció a Taules i Columnes

L'objectiu d'aquesta subtasca és traduir el model conceptual al model de la tecnologia de gestió que havíem triat, en aquest cas, el model relacional.

Com s'ha comentat, es va triar dissenyar un diagrama relacional des de bon principi, així que la traducció a taules, columnes i claus va ser trivial.

Només m'havia d'assegurar d'escriure correctament les entitats, la seva relació entre aquestes i les claus en un pseudo-format SQL que facilita l'escriptura dels scripts.

Dins d'aquesta subtasca també puc incloure l'apartat de normalització. L'apartat de normalització va ser més llarg perquè havia d'anar taula per taula comprovant que complien cada forma normal. Gràcies a aquest apartat, vaig aconseguir refinar algunes de les relacions i també adonar-me d'altres faltants al diagrama conceptual.

Tria del SGBD

Vaig triar immediatament Oracle Database 11g com SGDB que utilitzaria per crear la base de dades. El principal motiu de la seva tria va ser el fet que n'he fet ús a totes les assignatures de bases de dades del grau.

He utilitzat d'altres SGBD relacionals però també volia facilitar la tasca de correcció i, en cas d'haver algun problema tècnic, poder-ho consultar.

Entre el diferent programari que optava es troben SQL Server i PostgreSQL, entre d'altres.

Scripts de Taules al SGBD

Aquesta subtasca no tenia gaire més que traduir els escrits del model lògic al llenguatge SQL de creació de taules amb les seves claus primàries, secundàries i restriccions (quins camps poden ser NULL o quins poden ser positius o negatius, entre d'altres).

Tot i així, no m'enrecordava que Oracle treballava amb espais de taules que eren sinònims a usuaris i que la versió express només acceptava una base de dades.

Tot aquest tema el vaig haver de tornar a revisar i comprendre amb més profunditat a l'hora de generar el usuari (o espai de taules). També vaig haver

de repassar què feia cada permís a donar-li ja que alguns eren necessaris per crear certes estructures o consultar-ne d'altres.

A més, hi ha hagut diversos cops que he hagut d'esborrar les taules (i les seves dependències amb la resta de taules) per tornar-les a generar o modificar certs camps directament per descuit previ o per canvis en el diagrama conceptual per una millor adequació al sistema.

Scripts CRUD al SGBD

Aquesta subtasca es trobava més orientada a l'escriptura de scripts per inicialitzar un petit joc de dades i proves amb les taules ja implementades.

Bàsicament volia fer proves directes d'inserció, modificació, esborrat i lectura i comprovar que me les permetia realitzar.

També en aquest apartat vaig fer una inserció directa d'un set petit de dades a les taules principals que no eren derivades d'entitats associatives. D'aquesta forma, amb aquest set de dades, ja podria generar de forma automatitzada la resta a les entitats que les referenciaven en forma de claus foranes (i primàries) més endavant.

Documentació PAC 3

Una gran part del temps l'he invertit en la seva documentació. He anat modificant les decisions de disseny a mesura que rebia respostes o les canviava per facilitar el treball i el seu manteniment a llarg plaç.

Per aquesta raó, la part de documentació d'aquesta tasca, com totes les altres, ha continuat fins el final del treball.

No obstant això, a la finalització d'aquesta entrega vaig aconseguir tenir preparats tots els scripts que es referien a les subtasques anteriors, els títols i subtítols que es composava aquesta tasca i gran part del text que ha acabat sent definitiu.

6.4 Lliurament Final: Optimització (07/05/2019 – 10/06/2019)

Com a la tasca anterior, aquesta tasca la vaig iniciar prematurament degut a què vaig considerar que seria la secció on la part més pesada del treball recauria.

El seu començament va ser aproximadament el 7 d'abril de 2019 i puc confirmar que la meva decisió es trobava correctament justificada.

El contingut d'aquesta tasca passava per la creació de procediments suficients per generar les consultes que es demanaven. El seu temps de disseny i programació ha sigut elevat degut a que havien de controlar errors, excepcions i la consistència de la base de dades.

No només això, sinó que aquest apartat també incorpora la generació dels scripts de consultes amb les diverses problemàtiques que s'han comentat durant el treball.

Per últim, també s'havien d'inicialitzar les dades, fer un conjunt de proves, documentar tots aquests procesos, revisar el treball i, finalment, crear la presentació del treball.

Clarament, aquest últim apartat era el més crucial de tots, ja que és on es pot comprovar la seva implementació correcta, encara que, com he comentat, una bona base inicial és el que et permet arribar a aquest punt.

Scripts de Inicialització de dades

En realitat, aquesta subtasca la vaig realitzar al final de tot el treball. La creació de les dades no es va completar fins al final de les proves ja que també servien per revisar tot el funcionament de la base de dades.

En un principi la vaig situar en aquesta posició perquè no vaig acabar de veure que els procediments que es demanaven serien cridats per l'aplicació i que serien els que crearien les dades de les taules de consulta.

Com a tal, creia que podria generar les dades mitjançant declaracions SQL tradicionals (INSERT, UPDATE, DELETE, etc.).

Després de consultar-ho, vaig veure que els scripts de inicialització de dades correspondrien als scripts de inserció massiva que he utilitzat per generar-les i per provar la robustesa del sistema.

Per altra banda, l'exportació de la base de dades ja conté els registres correctes creats d'aquesta forma.

Simplement ha resultat que el millor moment d'obtenir aquests scripts ha sigut amb els scripts de proves de generació de dades massives automàtiques i la còpia de la base de dades. Ambdós dels quals s'han obtingut correctament al final del treball.

Scripts de Procediments

El plantejament inicial era generar tota la lògica de creació i actualització de registres a través de triggers de taules, ja fossin de inserció, modificació, esborrat, abans o després de la confirmació de la transacció (Oracle deixa especificar-ho).

Quan vaig consultar quin era el rol dels procediments i em van confirmar que el seu objectiu era mantenir al dia les consultes vaig veure que dividir la feina entre triggers i els procediments no era gaire òptim.

Al final, vaig arribar a la conclusió que els procediments farien tota la feina de manteniment de dades tant a les taules a les que referenciaven directament

com a les taules de les entitats associatives que s'utilitzen per realitzar les consultes.

Problablement aquesta ha sigut una de les tasques més costoses degut a què un sol procediment pot modificar dades de diverses taules i ha de controlar tots els possibles errors, informar-los i gestionar les excepcions.

De fet, gran part de les proves es basen en la generació i modificació a partir d'aquests procediments que seran cridats per l'aplicació. D'aquesta forma, el refinament d'aquests procediments ha anat de la mà a les proves realitzades i al inrevés.

Vaig fer alguns petits retocs als procediments per facilitar la tasca de generació de log d'errors.

En un principi, anava a guardar diverses variables per tipus de paràmetre però vaig veure que augmentava molt la complexitat. Així que vaig decidir guardar en una sola variable de text la llista de paràmetres separats per coma i per ordre en què el procediment demanava.

Per altra banda, abans també tenia una variable on posava el nom del procediment manualment, però cercant, vaig trobar que hi havia una variable global que recollia el nom del procediment que s'estava executant $$PLSQL_UNIT.

Finalment, vaig aprofitar el moment quan vaig anar comentant cada línia de codi per unificar aquests formats.

Scripts de Consultes

Mentre que comprovava que la generació de dades de cada bloc de taules fos correcta, a la vegada també escrivia el codi de consulta per aquell bloc.

Per exemple, els procediments relacionats amb Stock modifiquen diverses taules de consulta. Per aquesta raó, en el moment d'escriure i comprovar el codi d'una taula de consulta, també s'aprofitava per fer l'automatització d'inserció de dades i escriure el codi de consulta i verificar el que el funcionament fos el desitjat.

Per una altra part, com he comentat a la memòria, aquesta versió de Oracle es troba limitat a l'hora de realitzar consultes que retornin els primers registres seleccionats. Per aquesta raó, aquestes han resultat més complexes del necessari quan en altres SGBD el seu disseny seria molt més simple.

En acabar les consultes, la meva intenció era generar vistes parametritzades amb els paràmetres específics necessaris de cada consulta com, per exemple, l'any que es vol consultar o el codi de producte, paràmetres necessaris per trobar el increment percentual de demanda anual.

Malauradament, i després de molta cerca, no vaig trobar que Oracle implementés aquesta funcionalitat de vistes parametritzades en l'edició que estic utilitzant ni tampoc en qualsevol edició posterior. Per tant, no he hagut simplificar la seva consulta amb una sola línia de consulta SQL que cridés a la hipotètica vista.

També vaig buscar alternatives però no en vaig trobar per aquest SGBD. Al haver treballat amb d'altres que sí tenen una funcionalitat a la que demano, vaig suposar que Oracle també ho tenia implementat.

No obstant tot això, les consultes responen a les indicacions de l'enunciat (o la interpretació que n'he fet) i funcionen segons el disseny que vaig plantejar.

Scripts de Índexs i altres

Al final del treball, vaig decidir desestimar aquesta tasca. L'objectiu de la creació dels índexs era que les consultes s'executessin òptimament però la seva prioritat en el desenvolupament va baixa dràsticament al realitzar procediments que mantenen les consultes actualitzades en cada moment.

Tot i així, es podria veure si les consultes definitives es poden optimitzar gaire més amb la introducció de índexs. Definitivament, i degut al temps invertit en revisar el treball amb la seva documentació i el temps de dedicació disponible a repartir amb altres assignatures i la jornada laboral amb la baixada de prioritat van ser factors claus per retirar aquesta subtasca.

Revisió i Presentació del Treball

Aquest apartat es podria considerar un annex al de la documentació del lliurament final.

Durant el treball s'ha invertit un temps considerable a la creació i testeig de les funcionalitats demanades i, un cop tot era correcte, s'ha procedit a la seva revisió i documentació.

Mentre es revisaven tots els apartats del treball (sobretot les de línies de codi) també es documentava a la memòria i es comentaven les línies de codi.

Documentació Lliurament Final

Probablement la part on he hagut de dedicar més temps. Gran part de les decisions que he pres em semblen naturals degut a l'experiència laboral que tinc en aquest camp i explicar gran part d'aquestes decisions de disseny i fonamentar-les ha resultat laboriós.

En aquesta subtasca he hagut de fer un procés reiteratiu de revisió de tot el treball, ampliar i explicitar apartats que resumia per simple hàbit.

Aquí també inclouria els comentaris del codi. Al ser un treball pràctic, té una quantitat considerable de codi i comentar pràcticament cada línia dels scripts ha correspost una inversió de temps important.

No només això, sinó que explicar pas per pas cada prova realitzada és molt més costós que realitzar-la, executar-la i veure el resultat de primera mà.

Encara que el joc de proves es pot ampliar de forma il·limitada, he considerat que en el seu estat actual és prou exhaustiu tenint en compte la limitació de temps del projecte.

Tribunal d'Avaluació

Aquesta subtasca es pot dividir en la creació de la presentació del treball que es troba inclòs en el seu desenvolupament i entrega juntament amb la memòria i la resposta a les preguntes plantejades pel tribunal que són posteriors a aquest lliurament.

En conseqüència, només puc parlar de la primera part que és el disseny i creació de la presentació.

7. Conclusions

Aquest treball m'ha servit per fonamentar els coneixements que he adquirit d'un gran rang d'assignatures del grau. A més, m'ha permés conèixer processos interns sobre les tecnologies utilitzades que desconeixia i ampliar el meu rang d'eines a l'hora de dissenyar projectes tecnològics.

També m'he adonat com gran part del temps ha anat dedicat a l'escriptura d'una bona documentació i com el seu manteniment és gairebé tant important com la pròpia implementació del codi del projecte.

Crec que puc dir que he assolit tots els objectius que es demanaven al treball. Com tot projecte, sempre es pot millorar o redissenyar a mesura que sorgeixen noves tecnologies i segons aquest és testejat i utilitzat per altres desenvolupadors o els seus propis usuaris.

Degut a la limitació de temps i de no disposar d'aquesta retroalimentació d'un usuari o client real, és possible que hagi omès certes casuístiques o hagi interpretat algun punt d'una forma diferent a la intencionada.

Tot i així, i gràcies a les respostes que m'ha procurat el consultor a les meves preguntes, considero que el treball s'ha adequat molt fidelment als requisits que es demanen.

Malgrat la meva poca experiència en la gestió de projectes, la planificació plantejada inicialment ha sigut prou acurada. És cert que vaig posar un pes important a l'última entrega del treball però, com que sabia que podria haver algun contratemps, vaig fer un esforç per avançar-me a la planificació.

Aquesta previsió em va permetre rectificar certs aspectes del disseny amb antelació i també em va donar flexibilitat a l'hora de millorar els diferents scripts i el seu disseny.

Com he comentat al seguiment, certes subtasques les vaig haver de moure d'ordre o les vaig anar perfeccionant durant el transcurs de l'entrega o de tot el treball però, en general, m'he ajustat molt a l'ordre de la planificació inicial i la major part del treball ha sortit com vaig plantejar.

En un futur, el projecte hauria de complementar-se amb l'aplicació que feia referència l'enunciat d'aquest. Implementar una aplicació per dispositius Android preferentment i també iOS a poder ser amb connexions xifrades, autenticació d'usuaris i control de temps d'empleats a partir de l'aplicació i de geolocalització.

En cas de centrar-nos només en el posterior desenvolupament de la base de dades, es podria trobar una forma per poder generar vistes parametritzades per realitzar les consultes només passant paràmetres com els anys o la data de consulta, el producte o els que necessitessin. La versió de Oracle Database 11g Express es troba limitada en aquest aspecte i vaig consultar que també la resta a data de realització d'aquest treball. Potser es podria mirar un altre

SGBD més adequat que ho permetés, els scripts SQL haurien de funcionar sense gaires modificacions.

També s'haurien de fer més proves per corregir possibles incidències amb desenvolupadors diferents o amb usuaris reals. Un sol desenvolupador té una visió limitada, sobretot quan es tracta d'un tòpic que desconeix.

Per últim, també es podrien crear procediments per altres taules que certs usuaris necessitessis gestionar. Amb aquesta idea en ment, també s'haurien de crear altres rols o usuaris amb diferents permisos per adequar el seu accés a la base de dades i evitar incidències que poguessin causar.

8. Glossari

ABM O alta, baixa i modificació són les sigles en la nostra llengua de CRUD sense tenir en compte la R de lectura.

CRUD O alta, lectura, modificació i baixa són les funcions bàsiques necessàries en qualsevol sistema d'emmagatzematge d'informació.

DDL O Llenguatge de Definició de Dades es tracta d'un seguit de comandes estandarditzades per crear, modificar i eliminar objectes de la base de dades com taules, índexos o usuaris.

ERD O diagrama d'Entitat-Relació o Entitat-Interrelació, és un diagrama que es basa en el model d'Entitat-Relació que es tracta d'un model conceptual de dades d'alt nivell i independent de la tecnologia.

ERP O Planificador de Recursos Empresarials, és un programari per obtenir, guardar, gestionar i interpretar dades obtingudes de les activitats generades per l'empresa.

IDE O Entorn Integrat de Desenvolupament és un programa que recull un conjunt d'utilitats per ajudar als desenvolupadors en les seves tasques.

PL/SQL O Llenguatge Procedimental/Llenguatge de Consultes Estructurat és una ampliació del llenguatge declaratiu SQL que permet programar un seguit d'ordres i utilitzar estructures pròpies de llenguatges procedimentals en una base de dades de Oracle.

SGBD O Sistema Gestor de Base de Dades, és un programari que permet definir, crear i mantenir l'accés i control a una base de dades.

SQL O Llenguatge de Consultes Estructurat és el llenguatge declaratiu estàndard utilitzat per fer consultes a una base de dades relacional.

UML O llenguatge unificat de modelització és un llenguatge de propòsit general per a modelitzar sistemes de programari.

9. Bibliografia

[1] Toby Teorey, Sam Lightstone, Tom Nadeau, H.V. Jagadish. (2011). Database modeling and design (5th Edition). ELSEVIER. Burlington.

[2] Rod Stephens. (2009). Beginning Database Design Solutions. Wiley. Indianapolis.

[3] Batini Ceri Navathe. (1992). Conceptual Database Design, An Entity-Relationship Approach. Benjamin Cummins. Redwood City.

[4] Craig S. Mullins. (2013). Database Administration, The Complete Guide to DBA Practices and Procedures (2nd Edition). Addison-Wesley. Crawfordsville.

[5] Jeffrey A. Hoffer, V. Ramesh, Heikki Topi. (2016). Modern Database Management (12th Edition). PEARSON. Boston.

[6] https://mariadb.com/kb/en/library/database-normalization/, data Consulta: 28/03/2019

[7] Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom. (2009). Database Systems, The Complete Book (2nd Edition). PEARSON. New Jersey.

[8] https://docs.oracle.com/cd/B28359_01/server.111/b28318/physical.htm#CNCPT1081 , data Consulta: 03/04/2019

[9] https://docs.oracle.com/database/121/ADMQS/GUID-3F47A659-71C8-4544-B3B6-736554805816.htm , data Consulta: 03/04/2019

[10] https://docs.oracle.com/cd/B19306_01/server.102/b14220/schema.htm#i22627, data Consulta: 03/04/2019

[11] David M. Kroenke, David J. Auer. (2012). Database Processing, Fundamentals, Design, and Implementation (12th Edition). PEARSON. Boston.

[12] https://docs.oracle.com/cd/A97630_01/appdev.920/a96624/07_errs.htm , data Consulta: 05/04/2019

[13] https://docs.oracle.com/cd/B19306_01/server.102/b14200/pseudocolumns009.htm, data Consulta: 09/04/2019

10. Annexos

Annex I: Scripts de preparació de l'entorn (entorn.sql)

```
/*
Crea un espai de taules anomenat TFG lligat a un fitxer de dades anomenat tfg.dat
amb una grandària inicial de 10 MB, si existeix el fitxer, el reutilitza
i s'ampliarà automàticament cada 10MB fins a un màxim de 500MB
*/
CREATE TABLESPACE TFG
   DATAFILE 'tfg.dat'
   SIZE 10M
   REUSE
   AUTOEXTEND ON NEXT 10M MAXSIZE 500M;

/*
Crea un usuari TFGAdmin
amb contrasenya passTFG
que treballarà a l'espai de taules permanent TFG
i el seu espai de taules temporal serà TEMP
*/
CREATE USER TFGAdmin
   IDENTIFIED BY passTFG
   DEFAULT TABLESPACE TFG
   TEMPORARY TABLESPACE TEMP;

/*Permet que es connecti a la base de dades*/
GRANT CREATE SESSION TO TFGAdmin IDENTIFIED BY passTFG;
/*Permet que creiï taules al seu esquema*/
GRANT CREATE TABLE TO TFGAdmin IDENTIFIED BY passTFG;
/*Permet que creiï procediments al seu esquema*/
GRANT CREATE PROCEDURE TO TFGAdmin IDENTIFIED BY passTFG;
/*Permet que creiï esdeveniments al seu esquema*/
GRANT CREATE TRIGGER TO TFGAdmin IDENTIFIED BY passTFG;
/*Permet que creiï vistes al seu esquema*/
GRANT CREATE VIEW TO TFGAdmin IDENTIFIED BY passTFG;
/*Permet que creiï vistes materialitzades al seu esquema*/
GRANT CREATE MATERIALIZED VIEW TO TFGAdmin IDENTIFIED by passTFG;
/*Permet que creiï sequències de nombres enters generats automàticament*/
GRANT CREATE SEQUENCE TO TFGAdmin IDENTIFIED by passTFG;
/*Permet que creiï rutines que gestionen índexos*/
GRANT CREATE INDEXTYPE TO TFGAdmin IDENTIFIED by passTFG;
/*Treu les seves limitacions d'espai en el seu espai de taules*/
GRANT UNLIMITED TABLESPACE TO TFGAdmin IDENTIFIED by passTFG;
/*Permet crear operadors que fan referències a indextypes, consultes SQL o sentències DML*/
GRANT CREATE OPERATOR TO TFGAdmin IDENTIFIED by passTFG;
```

Annex II: Scripts de creació de taules (taules.sql)

```
CREATE TABLE TFGAdmin.envas(
   idEnvas varchar(15) NOT NULL,
   descripcio varchar(200) NOT NULL,
   capacitat integer NOT NULL,
   CONSTRAINT envasPK PRIMARY KEY (idEnvas)
);

CREATE TABLE TFGAdmin.producte(
   idProducte varchar(15) NOT NULL,
   descripcio varchar(200) NOT NULL,
   idEnvas varchar(50) NOT NULL,
   CONSTRAINT productePK PRIMARY KEY (idProducte),
   CONSTRAINT producteEnvasFK FOREIGN KEY (idEnvas) REFERENCES
TFGAdmin.envas (idEnvas)
);

CREATE TABLE TFGAdmin.component(
   idComponent varchar(15) NOT NULL,
   descripcio varchar(200) NOT NULL,
   CONSTRAINT componentPK PRIMARY KEY (idComponent)
);

CREATE TABLE TFGAdmin.zonaDistribucio(
   idZona varchar(15) NOT NULL,
   nomZona varchar(80) NOT NULL,
   descripcio varchar(200) NOT NULL,
   CONSTRAINT zonaDistribucioPK PRIMARY KEY (idZona)
);

CREATE TABLE TFGAdmin.sistemaEmbalatge(
   idSistema varchar(15) NOT NULL,
   idProducte varchar(200) NOT NULL,
   descripcio varchar(200) NOT NULL,
   quantitat integer NOT NULL,
   CONSTRAINT sistemaEmbalatgePK PRIMARY KEY (idSistema),
   CONSTRAINT sistemaEmbalatgeProducte FOREIGN KEY (idProducte)
REFERENCES TFGAdmin.producte (idProducte)
);

CREATE TABLE TFGAdmin.tipusEmpresa(
   idTipus char(2) NOT NULL,
   descripcio varchar(200) NOT NULL,
   CONSTRAINT tipusEmpresaPK PRIMARY KEY (idTipus)
);

CREATE TABLE TFGAdmin.empresa(
   idEmpresa varchar(15) NOT NULL,
   CIF varchar(20) NOT NULL,
```

```
    nomComercial varchar(50) NOT NULL,
    nomFiscal varchar(50) NOT NULL,
    adreca varchar(100) NOT NULL,
    telefon varchar(15) NOT NULL,
    idTipus char(2) NOT NULL,
    idZona varchar(15) NULL,
    CONSTRAINT empresaPK PRIMARY KEY (idEmpresa),
    CONSTRAINT empresaTipusFK FOREIGN KEY (idTipus) REFERENCES
TFGAdmin.tipusEmpresa (idTipus),
    CONSTRAINT empresaZonaFK FOREIGN KEY (idZona) REFERENCES
TFGAdmin.zonaDistribucio (idZona),
    CONSTRAINT empresaCIFUnic UNIQUE (CIF)
);

CREATE TABLE TFGAdmin.empleat(
    idEmpleat varchar(15) NOT NULL,
    descripcio varchar(200) NOT NULL,
    CONSTRAINT empleatPK PRIMARY KEY (idEmpleat)
);

CREATE TABLE TFGAdmin.fabrica(
    idFabrica varchar(15) NOT NULL,
    descripcio varchar(200) NOT NULL,
    adreca varchar(100) NOT NULL,
    telefon varchar(15) NOT NULL,
    CONSTRAINT fabricaPK PRIMARY KEY (idFabrica)
);

CREATE TABLE TFGAdmin.liniaFabricacio(
    idLinia varchar(15) NOT NULL,
    descripcio varchar(200) NOT NULL,
    llargada number(10,2) NOT NULL,
    idResponsable varchar(15) NOT NULL,
    idFabrica varchar(15) NOT NULL,
    CONSTRAINT liniaFabricacioPK PRIMARY KEY (idLinia),
    CONSTRAINT liniaFabricacioResponsableFK FOREIGN KEY
(idResponsable) REFERENCES TFGAdmin.empleat (idEmpleat),
    CONSTRAINT liniaFabricacioFabricaFK FOREIGN KEY (idFabrica)
REFERENCES TFGAdmin.fabrica (idFabrica),
    CONSTRAINT llargadaPositiva CHECK (llargada>0)
);

CREATE TABLE TFGAdmin.neteja(
    idNeteja varchar(15) NOT NULL,
    descripcio varchar(200) NOT NULL,
    CONSTRAINT netejaPK PRIMARY KEY (idNeteja)
);

CREATE TABLE TFGAdmin.torn(
    idTorn varchar(15) NOT NULL,
```

```
    descripcio varchar(200) NOT NULL,
    idResponsable varchar(15) NOT NULL,
    horaInici varchar(8) NOT NULL,
    horaFi varchar(8) NOT NULL,
    capacitatPersones INTEGER NOT NULL,
    idLinia varchar(15) NOT NULL,
    CONSTRAINT tornPK PRIMARY KEY (idTorn),
    CONSTRAINT tornResponsableFK FOREIGN KEY (idResponsable)
REFERENCES TFGAdmin.empleat (idEmpleat),
    CONSTRAINT tornLiniaFK FOREIGN KEY (idLinia) REFERENCES
TFGAdmin.liniaFabricacio (idLinia)
);

CREATE TABLE transport(
    idTransport varchar(15) NOT NULL,
    tipus varchar(50) NOT NULL,
    descripcio varchar(200) NOT NULL,
    capacitat number(10,2) NOT NULL,
    disponibilitat varchar(50) NOT NULL,
    CONSTRAINT transportPK PRIMARY KEY (idTransport),
    CONSTRAINT transportCapacitatPositiva CHECK (capacitat>0)
);

CREATE TABLE magatzem(
    idMagatzem varchar(15) NOT NULL,
    descripcio varchar(200) NOT NULL,
    adreca varchar(100) NOT NULL,
    capacitat number(10,2) NOT NULL,
    CONSTRAINT magatzemPK PRIMARY KEY (idMagatzem),
    CONSTRAINT magatzemCapacitatPositiva CHECK (capacitat>0)
);

CREATE TABLE compost(
    idProducte varchar(15) NOT NULL,
    idComponent varchar(15) NOT NULL,
    quantitat integer NOT NULL,
    unitatMesura char(2) NOT NULL,
    CONSTRAINT compostPK PRIMARY KEY (idProducte,idComponent),
    CONSTRAINT compostProducteFK FOREIGN KEY (idProducte)
REFERENCES TFGAdmin.producte (idProducte),
    CONSTRAINT compostComponentFK FOREIGN KEY (idComponent)
REFERENCES TFGAdmin.component (idComponent)
);

CREATE TABLE demanda(
    idProducte varchar(15) NOT NULL,
    idZona varchar(15) NOT NULL,
    data DATE NOT NULL,
    quantitat integer NOT NULL,
    CONSTRAINT demandaPK PRIMARY KEY (idProducte,idZona,data),
```

```
   CONSTRAINT demandaProducteFK FOREIGN KEY (idProducte)
REFERENCES TFGAdmin.producte (idProducte),
   CONSTRAINT demandaZonaFK FOREIGN KEY (idZona) REFERENCES
TFGAdmin.zonaDistribucio (idZona),
   CONSTRAINT demandaQuantitatPositiva CHECK (quantitat>0)
);

CREATE TABLE acces(
   idMagatzem varchar(15) NOT NULL,
   idEmpresa varchar(15) NOT NULL,
   horari varchar(50) NOT NULL,
   CONSTRAINT accesPK PRIMARY KEY (idMagatzem,idEmpresa),
   CONSTRAINT accesMagatzemFK FOREIGN KEY (idMagatzem)
REFERENCES TFGAdmin.magatzem (idMagatzem),
   CONSTRAINT accesEmpresaFK FOREIGN KEY (idEmpresa)
REFERENCES TFGAdmin.empresa (idEmpresa)
);

CREATE TABLE produccio(
   idProducte varchar(15) NOT NULL,
   idLinia varchar(15) NOT NULL,
   dataInici DATE NOT NULL,
   dataFi DATE NOT NULL,
   quantitatFabricada integer,
   CONSTRAINT produccioPK PRIMARY KEY (idProducte,idLinia,dataInici),
   CONSTRAINT produccioProducteFK FOREIGN KEY (idProducte)
REFERENCES TFGAdmin.producte (idProducte),
   CONSTRAINT produccioLiniaFK FOREIGN KEY (idLinia) REFERENCES
TFGAdmin.liniaFabricacio (idLinia)
);

CREATE TABLE netejament(
   idNeteja varchar(15) NOT NULL,
   idLinia varchar(15) NOT NULL,
   dataInici DATE NOT NULL,
   dataFi DATE NOT NULL,
   CONSTRAINT netejamentPK PRIMARY KEY (idNeteja,idLinia,dataInici),
   CONSTRAINT netejamentLiniaFK FOREIGN KEY (idLinia) REFERENCES
TFGAdmin.liniaFabricacio (idLinia),
   CONSTRAINT netejamentNetejaFK FOREIGN KEY (idNeteja)
REFERENCES TFGAdmin.neteja (idNeteja)
);

CREATE TABLE assignacio(
   idEmpleat varchar(15) NOT NULL,
   idTorn varchar(15) NOT NULL,
   data DATE NOT NULL,
   CONSTRAINT assignacioPK PRIMARY KEY (idEmpleat,idTorn,data),
   CONSTRAINT assignacioEmpleatFK FOREIGN KEY (idEmpleat)
REFERENCES TFGAdmin.empleat (idEmpleat),
```

```
   CONSTRAINT assignacioTornFK FOREIGN KEY (idTorn) REFERENCES
TFGAdmin.torn (idTorn)
);

CREATE TABLE stock(
   idSistema varchar(15) NOT NULL,
   idMagatzem varchar(15) NOT NULL,
   data DATE NOT NULL,
   volum decimal(10,2) NOT NULL,
   quantitat integer NOT NULL,
   idTransport varchar(15) NOT NULL,
   idEmpresa varchar(15) NULL,
   CONSTRAINT stockPK PRIMARY KEY (idSistema,idMagatzem,data),
   CONSTRAINT stockSistemaFK FOREIGN KEY (idSistema) REFERENCES
TFGAdmin.sistemaEmbalatge (idSistema),
   CONSTRAINT stockMagatzemFK FOREIGN KEY (idMagatzem)
REFERENCES TFGAdmin.magatzem (idMagatzem),
   CONSTRAINT stockTransportFK FOREIGN KEY (idTransport)
REFERENCES TFGAdmin.transport (idTransport),
   CONSTRAINT stockEmpresaFK FOREIGN KEY (idEmpresa)
REFERENCES TFGAdmin.empresa (idEmpresa)
);
```

Annex III: Scripts de taules de consultes (taulesConsulta.sql)

```
CREATE TABLE producteProduit(
   any_ integer NOT NULL,
   idProducte varchar(15) NOT NULL,
   quantitat integer NOT NULL,
   CONSTRAINT proPK PRIMARY KEY (any_,idProducte),
   CONSTRAINT proProdFK FOREIGN KEY (idProducte) REFERENCES
producte (idProducte)
);

CREATE TABLE LiniaAturada(
   any_ integer NOT NULL,
   mes integer NOT NULL,
   idLinia varchar(15) NOT NULL,
   horesFuncionant float NOT NULL,
   CONSTRAINT liniaAtPK PRIMARY KEY (any_,mes,idLinia),
   CONSTRAINT liniaAtLinFK FOREIGN KEY (idLinia) REFERENCES
liniaFabricacio (idLinia),
   CONSTRAINT liniaAtMesC CHECK (mes>=1 and mes<=12)
);

CREATE TABLE capacitatMagatzem(
   idMagatzem varchar(15) NOT NULL,
   espaiOcupat number(10,2) NOT NULL,
   CONSTRAINT capMagPK PRIMARY KEY (idMagatzem),
   CONSTRAINT capMagMagFK FOREIGN KEY (idMagatzem) REFERENCES
magatzem(idMagatzem),
```

```
    CONSTRAINT capMagEspP CHECK (espaiOcupat>=0)
);

CREATE TABLE empleatsUtilitzats(
    any_ integer NOT NULL,
    idEmpleat varchar(15) NOT NULL,
    nombreTorns integer NOT NULL,
    CONSTRAINT empUtPK PRIMARY KEY (any_,idEmpleat),
    CONSTRAINT empUtEmFK FOREIGN KEY (idEmpleat) REFERENCES
empleat (idEmpleat)
);

CREATE TABLE demandaAnual(
    any_ integer NOT NULL,
    idProducte varchar(15) NOT NULL,
    quantitat integer NOT NULL,
    quantitatDies integer NOT NULL,
    CONSTRAINT demAnPK PRIMARY KEY(any_,idProducte),
    CONSTRAINT demAnProdFK FOREIGN KEY (idProducte) REFERENCES
producte(idProducte)
);

CREATE TABLE componentUtilitzat(
    idComponent varchar(15) NOT NULL,
    nombreComposts integer NOT NULL,
    CONSTRAINT comUtPK PRIMARY KEY(idComponent),
    CONSTRAINT comUtCompFK FOREIGN KEY (idComponent)
REFERENCES component(idComponent)
);

CREATE TABLE capacitatAnualMagatzems(
    any_ integer NOT NULL,
    espaiOcupat number(10,2) NOT NULL,
    espaiMaximOcupat number(10,2) NOT NULL,
    CONSTRAINT capAnMagPK PRIMARY KEY (any_),
    CONSTRAINT capAnMagEspP CHECK (espaiOcupat>=0)
);

CREATE TABLE begudaProduida(
    any_ integer NOT NULL,
    capacitatLitres decimal(10,3) NOT NULL,
    CONSTRAINT begProdPK PRIMARY KEY (any_)
);

CREATE TABLE netejaAnual(
    any_ integer NOT NULL,
    idLinia varchar(15) NOT NULL,
    hores decimal(10,2) NOT NULL,
    CONSTRAINT netAnPK PRIMARY KEY (any_,idLinia),
```

```
    CONSTRAINT netAnLinFK FOREIGN KEY (idLinia) REFERENCES
liniafabricacio (idLinia)
);

CREATE TABLE tornsAnuals(
    any_ integer NOT NULL,
    tornsTotals integer NOT NULL,
    tornsComplerts integer NOT NULL,
    CONSTRAINT torAnPK PRIMARY KEY (any_)
);

CREATE TABLE empresesDistribuides(
    idEmpresa varchar(15) NOT NULL,
    idProducte varchar(15) NOT NULL,
    volum decimal(10,3) NOT NULL,
    CONSTRAINT empDistrPK PRIMARY KEY (idEmpresa),
    CONSTRAINT empDistrEmpFK FOREIGN KEY (idEmpresa) REFERENCES
empresa (idEmpresa),
    CONSTRAINT empDistrProdFK FOREIGN KEY (idProducte) REFERENCES
producte (idProducte)
);
```

Annex IV: Scripts de procediments, funcions auxiliars i taula LOG (proc.sql)

```
CREATE TABLE LOG(
    idLog number(38),
    nomProc varchar(100) NOT NULL,
    dataExecucio date default sysdate NOT NULL,
    entrada varchar(500) NOT NULL,
    sortida varchar(500) NOT NULL,
    CONSTRAINT LOGPK PRIMARY KEY(idLog)
);

/*
Funció que torna el identificador únic de cada taula.
Demana el nom de la taula, el nom del camp, el primer caràcter del id (per
defecte buit) i la llargada (per defecte 5).
Retorna el identificador
*/
CREATE OR REPLACE FUNCTION tornaIdTaula(
    i_taula IN varchar,
    i_idCamp IN varchar,
    i_idCaracter IN varchar:='',
    i_llargada IN number:=5
) RETURN varchar
IS
    id varchar(15);
    caracters varchar(5);
    llargada number;
```

```
   ultimNombre varchar(10);
   seguentNombre number;
   lletra char(1);
   sql_stmt varchar(200);
BEGIN
   --Sentència de SQL dinàmic que monta la consulta depenent dels camps
entrats
   -- Bàsicament agafa l'últim id de la taula demanada.
   sql_stmt:=
   'SELECT '||i_idCamp||'
   FROM
      (SELECT '||i_idCamp||', rownum
      FROM TFGAdmin.' ||i_taula||'
      ORDER BY '||i_idCamp||' DESC)
   WHERE rownum=1';
   -- Executa la sentència de sql dinàmic i guarda el resultat a id
   EXECUTE IMMEDIATE sql_stmt INTO id;
   --Si la llargada per defecte (o indicada) és superior
   -- a la llargada de la clau actual, escull la de per defecte
   IF (i_llargada>LENGTH(id)) THEN
      llargada:=i_llargada;
   --Sinó, la llargada és la de la consulta (claus anteriors)
   ELSE
      llargada:=LENGTH(id);
   END IF;
   caracters:='';

   --Des de 1 a la llargada del id trobat
   FOR contador IN 1..llargada
   LOOP
      --Agafa lletra per lletra
      lletra:=SUBSTR(id,contador,1);
      -- Si el caràcter és alfabètic, el va afegint a la cadena caracters
      IF                                          LENGTH(TRIM(TRANSLATE(lletra,
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ',  '  '))) IS
NULL THEN
         caracters:=caracters || lletra;
      -- Si el caràcter és numèric, l'afegeix a la cadena ultimNombre
      ELSE
         ultimNombre:=ultimNombre || lletra;
      END IF;
   END LOOP;
   --Converteix l'últim nombre a numèric i li suma 1 (següent nombre)
   seguentNombre:=CAST(ultimNombre AS NUMBER);
   seguentNombre:=seguentNombre+1;

   --A la cadena caracters li suma 0 la resta de la llargada que no ocupa
seguentNombre
   -- Per exemple, 'F'+'00'+'11' per una llargada de 5.
```

```
   RETURN RPAD(caracters,llargada-LENGTH(seguentNombre),'0') ||
seguentNombre;
--Excepcions
EXCEPTION
--Quan no troba un id en el select
WHEN NO_DATA_FOUND THEN
   -- i no hi ha un caràcter de inici, torna 1
   IF i_idCaracter='' THEN
      RETURN '1';
   -- si hi ha caràcter de inici, el posa davant, omple de 0 la llargada-1 i l'últim
caràcter és 1
   ELSE
      RETURN RPAD(i_idCaracter,i_llargada-1,'0') || '1';
   END IF;
END;


/*
Procediment que genera un registre a la taula LOG
Demana el nom del procediment cridat, la cadena de variables d'entrada i la
cadena de variables de sortida.
*/
CREATE OR REPLACE PROCEDURE crearLOG(
   i_nomproc IN TFGAdmin.LOG.nomProc%type,
   i_entrada IN TFGAdmin.LOG.entrada%type,
   i_sortida IN TFGAdmin.LOG.sortida%type
)
IS
   id number;
BEGIN
   --Crida tornaIdTaula que retorna el següent id numèric (per aquest cas)
   id:=CAST(tornaIdTaula('LOG','idLog') AS NUMBER);
   --Insereix el registre de LOG amb el nou id i els paràmetres de entrada
   INSERT INTO TFGAdmin.LOG(idLog,nomProc,entrada,sortida)
   VALUES(id,i_nomProc,i_entrada,i_sortida);
END;

/*
Procediment que genera un registre de Component
Demana la descripció del component
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE crearComponent(
   i_descripcio IN TFGAdmin.component.descripcio%type,
   RSP OUT varchar
)
IS
   idComponent varchar(15);
   parametres varchar(500);
BEGIN
```

```
    --Per defecte, RSP és correcte
    RSP:='OK';
    --Llista de paràmetres
    parametres:=i_descripcio;
    --Crida tornaIdTaula per aconseguir el següent id
    idComponent:=tornaIdTaula('component','idComponent','C');
    --Inserta el registre
    INSERT INTO TFGAdmin.component VALUES(idComponent,i_descripcio);
    --Crea el log amb el nom del procediment, paràmetre i variable d'error
    crearLOG($$PLSQL_UNIT,parametres,RSP);
--Excepcions
EXCEPTION
--Per qualsevol excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
    RSP:='ERROR'|| ':' ||SQLERRM;
    --Crea el log amb el nom del procediment, paràmetre i variable d'error
    crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que modifica un registre de Component
Demana el id i descripció del component
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE modificarComponent(
    i_idComponent IN TFGAdmin.component.idComponent%type,
    i_descripcio IN TFGAdmin.component.descripcio%type,
    RSP OUT varchar
)
IS
    parametres varchar(500);
BEGIN
    --Per defecte, RSP és correcte
    RSP:='OK';
    --Llista de paràmetres
    parametres:=i_idComponent||','||i_descripcio;
    --Actualitza el registre indicat
    UPDATE TFGAdmin.component SET
    descripcio=i_descripcio
    WHERE
    idComponent=i_idComponent;
    --Crea el log amb el nom del procediment, paràmetre i variable d'error
    crearLOG($$PLSQL_UNIT,parametres,RSP);
--Excepcions
EXCEPTION
--Per qualsevol excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
    RSP:='ERROR'|| ':' ||SQLERRM;
```

```
    --Crea el log amb el nom del procediment, paràmetre i variable d'error
    crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que esborra un registre de Component
Demana el id del component
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE esborrarComponent(
    i_idComponent IN TFGAdmin.component.idComponent%type,
    RSP OUT varchar
)
IS
    parametres varchar(500);
BEGIN
    --Per defecte, RSP és correcte
    RSP:='OK';
    --Llista de paràmetres
    parametres:=i_idComponent;
    --Esborra el registre
    DELETE FROM TFGAdmin.component
    WHERE
    idComponent=i_idComponent;
    --Crea el log amb el nom del procediment, paràmetre i variable d'error
    crearLOG($$PLSQL_UNIT,parametres,RSP);
--Excepcions
EXCEPTION
--Per qualsevol excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
    RSP:='ERROR'|| ':' ||SQLERRM;
    --Crea el log amb el nom del procediment, paràmetre i variable d'error
    crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment auxiliar per inserir i modificar les hores que s'ha trobat en
funcionament una línia durant una data d'inici i una de fi
Demana la línia, la data d'inici de producció, la data d'aturada i si es tracta
d'una resta d'hores (2) o d'una suma (1)
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE actualitzaLiniaAturada(
    i_linia IN TFGAdmin.produccio.idLinia%type,
    i_dataInici IN TFGAdmin.produccio.dataInici%type,
    i_dataFi IN TFGAdmin.produccio.dataFi%type,
    i_SumaResta IN number,
    RSP IN OUT varchar
)
```

```
IS
   nombreLiniaAturada number(3);
   diferenciaMesos INTEGER;
   parametres varchar(500);
BEGIN
   --Per defecte RSP és correcte
   RSP:='OK';
   --Llista de paràmetres
   parametres:=i_linia||','||TO_CHAR(i_dataInici,'DD-MM-YYYY
HH24:MI:SS')||','||TO_CHAR(i_dataFi,'DD-MM-YYYY
HH24:MI:SS')||','||i_SumaResta;
   --Nombre enter de mesos entre la data d'inici i fi
   diferenciaMesos:=TRUNC(MONTHS_BETWEEN(i_dataFi,i_dataInici));
   --Si hi ha més d'1 mes de diferència
   IF diferenciaMesos>0 THEN
      --Recorre mes a mes
      FOR i in 0..diferenciaMesos
      LOOP
         --Si és el primer mes que recorre
         IF i=0 THEN
            --Guarda els registres existents d'aquell any, mes i línia de la data
inici (primer mes)
            SELECT COUNT(idLinia) INTO nombreLiniaAturada
            FROM LiniaAturada
            WHERE any_=EXTRACT(YEAR FROM i_dataInici) AND
mes=EXTRACT(MONTH FROM i_dataInici) AND idLinia=i_linia;
            --Si no hi ha registres i es tracta d'una suma
            IF nombreLiniaAturada=0 AND i_SumaResta=2 THEN
               --Insereix el registre amb l'any, mes, línia i nombre d'hores entre el
últim dia de mes i el de data Inici
               INSERT INTO LiniaAturada
               SELECT
               EXTRACT(YEAR FROM i_dataInici),EXTRACT(MONTH FROM
i_dataInici),i_linia,(TRUNC(LAST_DAY(i_dataInici))-i_dataInici)*24
               FROM DUAL;
            --Si hi ha registres o es tracta d'una resta (ha d'existir registre)
            ELSE
               --Actualitza la línia amb la suma (o resta) d'hores des del final de
mes fins la data d'inici
               UPDATE LiniaAturada SET
HoresFuncionant=HoresFuncionant+POWER(-
1,i_SumaResta)*(TRUNC(LAST_DAY(i_dataInici))-i_dataInici)*24 WHERE
any_=EXTRACT(YEAR FROM i_dataInici) AND mes=EXTRACT(MONTH
FROM i_dataInici) AND idLinia=i_linia;
            END IF;
         --Si és l'últim mes que recorre
         ELSIF i=diferenciaMesos THEN
            --Guarda els registres existents d'aquell any, mes i línia de la data fi
(últim mes)
            SELECT COUNT(idLinia) INTO nombreLiniaAturada
```

```
            FROM LiniaAturada
            WHERE any_=EXTRACT(YEAR FROM i_dataFi) AND
mes=EXTRACT(MONTH FROM i_dataFi) AND idLinia=i_linia;
            --Si no hi ha registres i es tracta d'una suma
            IF nombreLiniaAturada=0 AND i_SumaResta=2 THEN
               --Insereix el registre amb l'any, mes, línia i nombre d'hores entre el
últim dia de mes i el de data de Fi
               INSERT INTO LiniaAturada
               SELECT
               EXTRACT(YEAR FROM i_dataFi),EXTRACT(MONTH FROM
i_dataFi),i_linia,(i_dataFi-trunc(i_dataFi, 'MM'))*24
               FROM DUAL;
            --Si hi ha registres o es tracta d'una resta (ha d'existir registre)
            ELSE
               --Actualitza la línia amb la suma (o resta) d'hores des del final de
mes fins la data de fi
               UPDATE LiniaAturada SET
HoresFuncionant=HoresFuncionant+POWER(-1,i_SumaResta)*(i_dataFi-
trunc(i_dataFi, 'MM'))*24 WHERE any_=EXTRACT(YEAR FROM i_dataFi) AND
mes=EXTRACT(MONTH FROM i_dataFi) AND idLinia=i_linia;
            END IF;
         --Si és un mes d'entremig
         ELSE
            --Guarda els registres existents d'aquell any, mes i línia des de la
data d'inici (suma el nombre de mesos)
            SELECT COUNT(idLinia) INTO nombreLiniaAturada
            FROM LiniaAturada
            WHERE any_=EXTRACT(YEAR FROM ADD_MONTHS(i_dataInici,i))
AND mes=EXTRACT(MONTH FROM ADD_MONTHS(i_dataInici,i)) AND
idLinia=i_linia;
            --Si no hi ha registres i es tracta d'una suma
            IF nombreLiniaAturada=0 AND i_SumaResta=2 THEN
               --Insereix el registre amb l'any, mes, línia i nombre d'hores entre el
últim dia de mes i el primer dia del mes
               INSERT INTO LiniaAturada
               SELECT
               EXTRACT(YEAR FROM
ADD_MONTHS(i_dataInici,i)),EXTRACT(MONTH FROM
ADD_MONTHS(i_dataInici,i)),i_linia,(1+trunc(last_day(ADD_MONTHS(i_dataIni
ci,i)))-trunc(ADD_MONTHS(i_dataInici,i),'MM'))*24
               FROM DUAL;
            --Si hi ha registres o es tracta d'una resta (ha d'existir registre)
            ELSE
               --Actualitza la línia amb la suma (o resta) d'hores des del final de
mes fins al inici de mes
               UPDATE LiniaAturada SET
HoresFuncionant=HoresFuncionant+POWER(-
1,i_SumaResta)*(1+trunc(last_day(ADD_MONTHS(i_dataInici,i)))-
trunc(ADD_MONTHS(i_dataInici,i),'MM'))*24 WHERE any_=EXTRACT(YEAR
```

```
FROM ADD_MONTHS(i_dataInici,i)) AND mes=EXTRACT(MONTH FROM
ADD_MONTHS(i_dataInici,i)) AND idLinia=i_linia;
            END IF;
         END IF;
      END LOOP;
   --Si no hi ha més d'un mes de diferència
   ELSE
      --Guarda els registres existents d'aquell any, mes i línia de la data inici
(primer mes)
      SELECT COUNT(idLinia) INTO nombreLiniaAturada
      FROM LiniaAturada
      WHERE any_=EXTRACT(YEAR FROM i_dataInici) AND
mes=EXTRACT(MONTH FROM i_dataInici) AND idLinia=i_linia;
      --Si no hi ha registres i es tracta d'una suma
      IF nombreLiniaAturada=0 AND i_SumaResta=2 THEN
         --Insereix el registre amb l'any, mes, línia i nombre d'hores entre la data
de fi i la data d'inici
         INSERT INTO LiniaAturada
         SELECT
         EXTRACT(YEAR FROM i_dataInici),EXTRACT(MONTH FROM
i_dataInici),i_linia,(i_dataFi-i_dataInici)*24
         FROM DUAL;
      --Si hi ha registres o es tracta d'una resta (ha d'existir registre)
      ELSE
         --Actualitza la línia amb la suma (o resta) d'hores des de la data de fi
fins la data d'inici
         UPDATE LiniaAturada SET
HoresFuncionant=HoresFuncionant+POWER(-1,i_SumaResta)*(i_dataFi-
i_dataInici)*24 WHERE any_=EXTRACT(YEAR FROM i_dataInici) AND
mes=EXTRACT(MONTH FROM i_dataInici) AND idLinia=i_linia;
      END IF;
   END IF;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
--Excepcions
EXCEPTION
--Per qualsevol excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
   RSP:='ERROR'|| ':' ||SQLERRM;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que crea un compost (relaciona component amb producte)
Demana el producte, el component del producte, la quantitat i la unitat de
mesura que utilitza
Retorna RSP variable d'errors
*/
```

```
CREATE OR REPLACE PROCEDURE crearCompost(
   i_producte IN TFGAdmin.compost.idProducte%type,
   i_component IN TFGAdmin.compost.idComponent%type,
   i_quantitat IN TFGAdmin.compost.quantitat%type,
   i_unitatMesura IN TFGAdmin.compost.unitatMesura%type,
   RSP OUT varchar
)
IS
   nombreCompostComponent number(10);
   parametres varchar(500);
BEGIN
   --Per defecte RSP és correcte
   RSP:='OK';
   --Llista de paràmetres
   parametres:=i_producte||','||i_component||','||i_quantitat||','||i_unitatMesura;
   --Guarda el nombre de registres actuals que està sent utilitzat aquest
component
   SELECT COUNT(idComponent) INTO nombreCompostComponent
   FROM TFGAdmin.componentUtilitzat
   WHERE idComponent=i_component;
   --Insereix el registre a compost
   INSERT INTO TFGAdmin.compost
VALUES(i_producte,i_component,i_quantitat,i_unitatMesura);
   --Si no hi ha registres com a component utilitzat
   IF nombreCompostComponent=0 THEN
      --Insereix un nou registre com a component utilitzat
      INSERT INTO TFGAdmin.componentUtilitzat VALUES(i_component,1);
   --Si hi ha registres
   ELSE
      --Actualitza la taula sumant-li un al nombre total de productes que utilitzen
el component
      UPDATE TFGAdmin.componentUtilitzat SET
nombreComposts=nombreComposts+1 WHERE idComponent=i_component;
   END IF;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
--Excepcions
EXCEPTION
--Per qualsevol excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
   RSP:='ERROR'|| ':' ||SQLERRM;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que modifica un registre de Compost
Demana el producte, el component d'aquest producte, la quantitat i la unitat de
mesura de la quantitat
```

```
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE modificarCompost(
    i_producte IN TFGAdmin.compost.idProducte%type,
    i_component IN TFGAdmin.compost.idComponent%type,
    i_quantitat IN TFGAdmin.compost.quantitat%type,
    i_unitatMesura IN TFGAdmin.compost.unitatMesura%type,
    RSP OUT varchar
)
IS
    nombreCompostComponent number(10);
    parametres varchar(500);
BEGIN
    RSP:='OK';
    parametres:=i_producte||','||i_component||','||i_quantitat||','||i_unitatMesura;
    --Actualitza el registre indicat
    UPDATE TFGAdmin.compost SET
quantitat=i_quantitat,unitatMesura=i_unitatMesura WHERE
idProducte=i_producte AND idComponent=i_component;

    crearLOG($$PLSQL_UNIT,parametres,RSP);
--Excepcions
EXCEPTION
--Per qualsevol excepció
WHEN OTHERS THEN
     --Guarda el error a RSP
    RSP:='ERROR'|| ':' ||SQLERRM;
    --Crea el log amb el nom del procediment, paràmetre i variable d'error
    crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que esborra el registre de compost d'un producte
Demana el producte i el seu component
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE esborrarCompost(
    i_producte IN TFGAdmin.compost.idProducte%type,
    i_component IN TFGAdmin.compost.idComponent%type,
    RSP OUT varchar
)
IS
    nombreCompostComponent number(10);
    parametres varchar(500);
BEGIN
    RSP:='OK';
    parametres:=i_producte||','||i_component;
    --Esborra el registre de Compost
    DELETE TFGAdmin.compost WHERE idProducte=i_producte AND
idComponent=i_component;
```

```
    --Resta una unitat el nombre de composts que participa el component
    UPDATE TFGAdmin.componentUtilitzat SET
nombreComposts=nombreComposts-1 WHERE idComponent=i_component;
    --Elimina aquells registres que hagin quedat a 0
    DELETE TFGAdmin.componentUtilitzat WHERE nombreComposts=0;

    crearLOG($$PLSQL_UNIT,parametres,RSP);
--Excepcions
EXCEPTION
--Per qualsevol excepció
WHEN OTHERS THEN
     --Guarda el error a RSP
    RSP:='ERROR'|| ':' ||SQLERRM;
    --Crea el log amb el nom del procediment, paràmetre i variable d'error
    crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que crea registres a producció
Demana el producte produït, la línia on es produeix, la data d'inici i la de fi de la
producció i la quantitat fabricada
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE crearProduccio(
    i_producte IN TFGAdmin.produccio.idProducte%type,
    i_linia IN TFGAdmin.produccio.idLinia%type,
    i_dataInici IN TFGAdmin.produccio.dataInici%type,
    i_dataFi IN TFGAdmin.produccio.dataFi%type,
    i_quantitatFabricada IN TFGAdmin.produccio.quantitatFabricada%type,
    RSP OUT varchar
)
IS
    anyFi integer;
    mesFi integer;
    nombreProducteAny number(3);
    diferenciaMesos number(3);
    nombreLiniaAturada number(3);
    parametres varchar(500);
BEGIN
    RSP:='OK';
    parametres:=i_producte||','||i_linia||','||TO_CHAR(i_dataInici,'DD-MM-YYYY
HH24:MI:SS')||','||TO_CHAR(i_dataFi,'DD-MM-YYYY
HH24:MI:SS')||','||i_quantitatFabricada;
    --Any de fi de la producció
    anyFi:=EXTRACT(YEAR FROM i_dataFi);
    --Nombre de registres del producte en aquell any que s'han produït
    SELECT COUNT(idProducte) INTO nombreProducteAny
    FROM TFGAdmin.producteProduit
    WHERE any_=anyFi AND idProducte=i_producte;
    --Insereix el registre a produccio
```

```
   INSERT INTO TFGAdmin.produccio
VALUES(i_producte,i_linia,i_dataInici,i_dataFi,i_quantitatFabricada);

   /*Producte més produit*/
   --Si no hi ha registres
   IF nombreProducteAny=0 THEN
      --Insereix un registre amb l'any, producte i quantitat que s'han fabricat
      INSERT INTO TFGAdmin.producteProduit
VALUES(anyFi,i_producte,i_quantitatFabricada);
   --Si hi ha registres
   ELSE
      --Actualitza el registre sumant la nova quantitat produïda
      UPDATE TFGAdmin.producteProduit SET
quantitat=quantitat+i_quantitatFabricada WHERE any_=anyFi AND
idProducte=i_producte;
   END IF;

   /*Linia Aturada*/
   --Crea un registre amb les hores de la línia
   actualitzaLiniaAturada(i_linia,i_dataInici,i_dataFi,2,RSP);

   crearLOG($$PLSQL_UNIT,parametres,RSP);
--Excepcions
EXCEPTION
--Per qualsevol excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
   RSP:='ERROR'|| ':' ||SQLERRM;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que modifica un registre de produccio
Demana el producte produït, la línia on es produeix, la data d'inici i la de fi de la
producció i la quantitat fabricada
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE modificarProduccio(
   i_producte IN TFGAdmin.produccio.idProducte%type,
   i_linia IN TFGAdmin.produccio.idLinia%type,
   i_dataInici IN TFGAdmin.produccio.dataInici%type,
   i_dataFi IN TFGAdmin.produccio.dataFi%type,
   i_quantitatFabricada IN TFGAdmin.produccio.quantitatFabricada%type,
   RSP OUT varchar
)
IS
   anyFi integer;
   anyFiAnterior integer;
   nombreProducteAny number(3);
```

```
    quantitatAnterior TFGAdmin.produccio.quantitatFabricada%type;
    dataFiAnterior TFGAdmin.produccio.dataFi%type;
    parametres varchar(500);
BEGIN
    RSP:='OK';
    parametres:=i_producte||','||i_linia||','||TO_CHAR(i_dataInici,'DD-MM-YYYY
HH24:MI:SS')||','||TO_CHAR(i_dataFi,'DD-MM-YYYY
HH24:MI:SS')||','||i_quantitatFabricada;
    --Obté l'any de la data de fi
    anyFi:=EXTRACT(YEAR FROM i_dataFi);
    --Guarda l'anterior quantitat fabricada amb la seva data de fi
    SELECT quantitatFabricada,dataFi INTO quantitatAnterior,dataFiAnterior
    FROM TFGAdmin.produccio
    WHERE idProducte=i_producte AND idLinia=i_linia AND
dataInici=i_dataInici;
    --Obté l'any de la data de fi anterior
    anyFiAnterior:=EXTRACT(YEAR FROM dataFiAnterior);
    --Actualitza el registre de produccio
    UPDATE TFGAdmin.produccio SET
dataFi=i_dataFi,quantitatFabricada=i_quantitatFabricada WHERE
idProducte=i_producte AND idLinia=i_linia AND dataInici=i_dataInici;

    /*Producte més produit*/
    --Guarda els registres de producte produït pel nou any
    SELECT COUNT(idProducte) INTO nombreProducteAny
    FROM TFGAdmin.producteProduit
    WHERE any_=anyFi AND idProducte=i_producte;
    --Actualitza el registre anterior treient la quantitat que s'ha produït
    UPDATE TFGAdmin.producteProduit SET quantitat=quantitat-
quantitatAnterior WHERE any_=anyFiAnterior AND idProducte=i_producte;
    --Si no hi ha registres al nou any
    IF nombreProducteAny=0 THEN
        --Crea el nou registre
        INSERT INTO TFGAdmin.producteProduit
VALUES(anyFi,i_producte,i_quantitatFabricada);
    --Si hi ha registres al nou any
    ELSE
        --Li suma la nova quantitat
        UPDATE TFGAdmin.producteProduit SET
quantitat=quantitat+i_quantitatFabricada WHERE any_=anyFi AND
idProducte=i_producte;
    END IF;
    --Esborra aquells registres que hagin quedat a 0
    DELETE TFGAdmin.producteProduit WHERE quantitat=0;

    /*Linia aturada*/
    --Treu les hores de producció anterior
    actualitzaLiniaAturada(i_linia,i_dataInici,dataFiAnterior,1,RSP);
    --Suma les noves hores de producció
    actualitzaLiniaAturada(i_linia,i_dataInici,i_dataFi,2,RSP);
```

```
    --Esborra les línies sense hores de funcionament
    DELETE TFGAdmin.liniaAturada WHERE horesFuncionant=0;

    crearLOG($$PLSQL_UNIT,parametres,RSP);
--Excepcions
EXCEPTION
--Per qualsevol excepció
WHEN OTHERS THEN
     --Guarda el error a RSP
    RSP:='ERROR'|| ':' ||SQLERRM;
    --Crea el log amb el nom del procediment, paràmetre i variable d'error
    crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que esborra un registre de produccio
Demana el producte, la línia i la data d'inici (claus)
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE esborrarProduccio(
    i_producte IN TFGAdmin.produccio.idProducte%type,
    i_linia IN TFGAdmin.produccio.idLinia%type,
    i_dataInici IN TFGAdmin.produccio.dataInici%type,
    RSP OUT varchar
)
IS
    anyFi integer;
    quantitatAnterior TFGAdmin.produccio.quantitatFabricada%type;
    dataFi TFGAdmin.produccio.dataFi%type;
    parametres varchar(500);
BEGIN
    RSP:='OK';
    parametres:=i_producte||','||i_linia||','||TO_CHAR(i_dataInici,'DD-MM-YYYY
HH24:MI:SS');
    --Guarda la quantitat i data de fi anterior del registre de produccio
    SELECT quantitatFabricada,dataFi INTO quantitatAnterior,dataFi
    FROM TFGAdmin.produccio
    WHERE idProducte=i_producte AND idLinia=i_linia AND
dataInici=i_dataInici;
    --Obté l'any de la data de fi
    anyFi:=EXTRACT(YEAR FROM dataFi);
    --Esborra el registre de produccio
    DELETE TFGAdmin.produccio WHERE idProducte=i_producte AND
idLinia=i_linia AND dataInici=i_dataInici;

    /*Producte més produit*/
    --Resta la quantitat anterior al registre de producteProduit
    UPDATE TFGAdmin.producteProduit SET quantitat=quantitat-
quantitatAnterior WHERE any_=anyFi AND idProducte=i_producte;
    --Esborra els registres a 0
```

```
    DELETE TFGAdmin.producteProduit WHERE quantitat=0;

    /*Linia aturada*/
    --Resta les hores de producció a la línia corresponent
    actualitzaLiniaAturada(i_linia,i_dataInici,dataFi,1,RSP);
    --Elimina els registres a 0
    DELETE TFGAdmin.liniaAturada WHERE horesFuncionant=0;

    crearLOG($$PLSQL_UNIT,parametres,RSP);
--Excepcions
EXCEPTION
--Per qualsevol excepció
WHEN OTHERS THEN
     --Guarda el error a RSP
    RSP:='ERROR'|| ':' ||SQLERRM;
    --Crea el log amb el nom del procediment, paràmetre i variable d'error
    crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que crea un registre de Stock
Demana el sistema d'embalatge, el magatzem on es guarda, la data del registre,
el volum que ocupa, la quantitat del sistema, el transport utilitzat i l'empresa (si és que surt)
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE crearStock(
    i_idSistema TFGAdmin.stock.idSistema%type,
    i_idMagatzem TFGAdmin.stock.idMagatzem%type,
    i_data TFGAdmin.stock.data%type,
    i_volum TFGAdmin.stock.volum%type,
    i_quantitat TFGAdmin.stock.quantitat%type,
    i_idTransport TFGAdmin.stock.idTransport%type,
    i_idEmpresa TFGAdmin.stock.idEmpresa%type,
    RSP OUT varchar
)
IS
    producteSistema TFGAdmin.producte.idProducte%type;
    existeixEmpresaDistribuida number(1);
    capacitatEnvasML TFGAdmin.envas.capacitat%type;
    existeixBegudaAnual number(1);
    existeixCapacitatAnual number(1);
    espaiOcupat TFGAdmin.capacitatMagatzem.espaiOcupat%type;
    excesEspai EXCEPTION;
    espaiMagatzem TFGAdmin.magatzem.capacitat%type;
    registreMagatzem number(1);
    parametres varchar(500);
BEGIN
    RSP:='OK';
```

```
   parametres:=i_idSistema||','||i_idMagatzem||','||TO_CHAR(i_data,'DD-MM-
YYYY
HH24:MI:SS')||','||i_volum||','||i_quantitat||','||i_idTransport||','||i_idEmpresa;
   --Guarda la capacitat total del magatzem
   SELECT capacitat INTO espaiMagatzem
   FROM magatzem
   WHERE idMagatzem=i_idMagatzem;
   --Guarda els registres que hi ha en capacitatMagatzem
   SELECT COUNT(*) INTO registreMagatzem
   FROM capacitatMagatzem
   WHERE idMagatzem=i_idMagatzem;

   /*capacitat Magatzem*/
   --Si no hi ha registre
   IF registreMagatzem=0 THEN
      --Si el volum inserit sobrepassa al espai disponible
      IF i_volum>espaiMagatzem THEN
         --Crida l'excepció excesEspai
         RAISE excesEspai;
      END IF;
      --En cas contrari, insereix el registre a la capacitat magatzem
      INSERT INTO capacitatMagatzem VALUES(i_idMagatzem,i_volum);
   --Si hi ha registre
   ELSE
      --Guarda l'espai ocupat actual del magatzem
      SELECT espaiOcupat INTO espaiOcupat
      FROM capacitatMagatzem
      WHERE idMagatzem=i_idMagatzem;
      --Si afegint el nou espai ocupat supera el límit del magatzem
      IF (espaiOcupat+i_volum)>espaiMagatzem THEN
         --Crida l'excepció excesEspai
         RAISE excesEspai;
      END IF;
      --En cas contrari, suma el nou espai a l'espai ocupat del magatzem
      UPDATE capacitatMagatzem SET espaiOcupat=espaiOcupat+i_volum
WHERE idMagatzem=i_idMagatzem;
   END IF;

   /*Capacitat Anual Magatzem*/
   --Guarda el nombre de registres de la capacitatAnualMagatzems
   SELECT COUNT(*) INTO existeixCapacitatAnual
   FROM capacitatAnualMagatzems
   WHERE any_=EXTRACT(YEAR FROM i_data);
   --Si no existeix registre
   IF existeixCapacitatAnual=0 THEN
      --El insereix
      INSERT INTO capacitatAnualMagatzems VALUES(EXTRACT(YEAR
FROM i_data),i_volum,i_volum);
   --Si existeix el registre
   ELSE
```

```
      --Actualitza l'espai ocupat i el seu màxim si resulta que l'espaiOcupat més
el nou volum és superior al guardat
      UPDATE capacitatAnualMagatzems SET
espaiOcupat=espaiOcupat+i_volum,
      espaiMaximOcupat=(CASE WHEN
espaiOcupat+i_volum>espaiMaximOcupat THEN espaiOcupat+i_volum ELSE
espaiMaximOcupat END)
      WHERE any_=EXTRACT(YEAR FROM i_data);
   END IF;

   /*Beguda produida anualment*/
   --Guarda la capacitat de l'envàs i el producte del sistema
   SELECT Envas.capacitat,Producte.idProducte INTO
capacitatEnvasML,producteSistema
   FROM SistemaEmbalatge
   INNER JOIN Producte ON
      Producte.idProducte=SistemaEmbalatge.idProducte
   INNER JOIN Envas ON
      Producte.idEnvas=Envas.idEnvas
   WHERE SistemaEmbalatge.idSistema=i_idSistema;
   --Si el volum entrant és superior a 0
   IF i_volum>0 THEN
      --Guarda els registres de begudaProduida d'aquell any
      SELECT COUNT(*) INTO existeixBegudaAnual
      FROM begudaProduida
      WHERE any_=EXTRACT(YEAR FROM i_data);
      --Si no hi ha registre
      IF existeixBegudaAnual=0 THEN
         --Insereix el registre calculant els litres de beguda segons la quantitat i
capacitat de l'envàs (mL)
         INSERT INTO begudaProduida VALUES(EXTRACT(YEAR FROM
i_data),i_quantitat*capacitatEnvasML/1000);
      --Si hi ha registre
      ELSE
         --Suma a la capacitat produïda els litres calculats de la no capacitat
         UPDATE begudaProduida SET
capacitatLitres=capacitatLitres+i_quantitat*capacitatEnvasML/1000
         WHERE any_=EXTRACT(YEAR FROM i_data);
      END IF;
   END IF;

   /*Empreses Distribuides*/
   --Si el paràmetre d'empresa està ple
   IF i_idEmpresa IS NOT NULL THEN
      --Guarda els registres que hi ha d'aquella empresa i producte
      SELECT COUNT(idEmpresa) INTO existeixEmpresaDistribuida
      FROM empresesDistribuides
      WHERE idEmpresa=i_idEmpresa AND idProducte=producteSistema;
      --Si no existeix registre
      IF existeixEmpresaDistribuida=0 THEN
```

```
        --Insereix el registre posant la quantitat en positiu (la quantitat entrada
ha de ser negativa perquè surt del Stock)
        INSERT INTO empresesDistribuides
VALUES(i_idEmpresa,producteSistema,-i_quantitat*capacitatEnvasML/1000);
      --Si existeix registre
      ELSE
        --Suma al volum del registre el volum distribuit
        UPDATE empresesDistribuides SET volum=volum-
i_quantitat*capacitatEnvasML/1000
        WHERE idEmpresa=i_idEmpresa AND idProducte=producteSistema;
      END IF;
   END IF;

   /*Stock*/
   --Finalment, si tot ha funcionat, crea el registre a Stock
   INSERT INTO Stock
VALUES(i_idSistema,i_idMagatzem,i_data,i_volum,i_quantitat,i_idTransport,i_i
dEmpresa);

   crearLOG($$PLSQL_UNIT,parametres,RSP);
--Excepcions
EXCEPTION
--Excepció que es crida si es passa de l'espai del magatzem
WHEN excesEspai THEN
   --Error personalitzat
   RSP:='ERROR'|| ':' ||'espai insuficient al magatzem '||i_idMagatzem;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
--Per qualsevol altra excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
   RSP:='ERROR'|| ':' ||SQLERRM;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que modifica un registre de Stock
Demana el sistema d'embalatge, el magatzem on es guarda, la data del
registre,
el volum que ocupa, la quantitat del sistema, el transport utilitzat i l'empresa (si
és que surt)
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE modificarStock(
   i_idSistema TFGAdmin.stock.idSistema%type,
   i_idMagatzem TFGAdmin.stock.idMagatzem%type,
   i_data TFGAdmin.stock.data%type,
   i_volum TFGAdmin.stock.volum%type,
   i_quantitat TFGAdmin.stock.quantitat%type,
```

```
    i_idTransport TFGAdmin.stock.idTransport%type,
    i_idEmpresa TFGAdmin.stock.idEmpresa%type,
    RSP OUT varchar
)
IS
    producteSistema TFGAdmin.producte.idProducte%type;
    quantitatAnterior TFGAdmin.stock.quantitat%type;
    capacitatEnvasML TFGAdmin.envas.capacitat%type;
    volumAnterior TFGAdmin.stock.volum%type;
    espaiOcupat TFGAdmin.capacitatMagatzem.espaiOcupat%type;
    excesEspai EXCEPTION;
    espaiMagatzem TFGAdmin.magatzem.capacitat%type;
    parametres varchar(500);
BEGIN
    RSP:='OK';
    parametres:=i_idSistema||','||i_idMagatzem||','||TO_CHAR(i_data,'DD-MM-
YYYY
HH24:MI:SS')||','||i_volum||','||i_quantitat||','||i_idTransport||','||i_idEmpresa;
    --Guarda la capacitat màxima del magatzem implicat
    SELECT capacitat INTO espaiMagatzem
    FROM magatzem
    WHERE idMagatzem=i_idMagatzem;
    --Guarda el volum i quantitat anterior del registre modificat
    SELECT volum,quantitat INTO volumAnterior,quantitatAnterior
    FROM Stock
    WHERE idSistema=i_idSistema AND idMagatzem=i_idMagatzem AND
data=i_data;

    /*capacitat Magatzem*/
    --Guarda l'espai ocupat del registre anterior del magatzem
    SELECT espaiOcupat INTO espaiOcupat
    FROM capacitatMagatzem
    WHERE idMagatzem=i_idMagatzem;
    --Si l'espai anterior menys el volum anterior més el nou volum és major a
l'espai màxim
    IF (espaiOcupat-volumAnterior+i_volum)>espaiMagatzem THEN
        --Crida l'excepció excesEspai
        RAISE excesEspai;
    END IF;
    --Si passa, actualitza l'espai ocupat pel nou
    UPDATE capacitatMagatzem SET espaiOcupat=espaiOcupat-
volumAnterior+i_volum WHERE idMagatzem=i_idMagatzem;
    --Esborra els registres que quedin a 0
    DELETE capacitatMagatzem WHERE espaiOcupat=0;

    /*Capacitat Anual Magatzem*/
    --Actualitza l'espai ocupat anual i el seu màxim si dóna el cas que la
modificació fa que sigui superior
    UPDATE capacitatAnualMagatzems SET espaiOcupat=espaiOcupat-
volumAnterior+i_volum,
```

```
      espaiMaximOcupat=(CASE WHEN espaiOcupat-
volumAnterior+i_volum>espaiMaximOcupat THEN espaiOcupat-
volumAnterior+i_volum ELSE espaiMaximOcupat END)
   WHERE any_=EXTRACT(YEAR FROM i_data);
   --Esborra els registres a 0
   DELETE capacitatAnualMagatzems WHERE espaiMaximOcupat=0;

   /*Beguda produida anualment*/
   --Guarda la capacitat de l'envàs i el producte del sistema
   SELECT Envas.capacitat,Producte.idProducte INTO
capacitatEnvasML,producteSistema
   FROM SistemaEmbalatge
   INNER JOIN Producte ON
      Producte.idProducte=SistemaEmbalatge.idProducte
   INNER JOIN Envas ON
      Producte.idEnvas=Envas.idEnvas
   WHERE SistemaEmbalatge.idSistema=i_idSistema;
   --Si el volum és positiu
   IF i_volum>0 THEN
      --Actualitza els litres treient la quantitat anterior i reemplaçant-la per la
nova quantitat
      UPDATE begudaProduida SET
capacitatLitres=capacitatLitres+(i_quantitat-
quantitatAnterior)*capacitatEnvasML/1000
      WHERE any_=EXTRACT(YEAR FROM i_data);
      --Esborra els registres negatius o a 0 (tot i que no hi haurien d'haver de
negatius)
      DELETE begudaProduida WHERE capacitatLitres<=0;
   END IF;

   /*Empreses Distribuides*/
   --Si hi ha informada l'empresa
   IF i_idEmpresa IS NOT NULL THEN
      --Actualitza el registre amb el nou volum (li treu l'antic i suma el nou)
      UPDATE empresesDistribuides SET volum=volum-(i_quantitat-
quantitatAnterior)*capacitatEnvasML/1000
      WHERE idEmpresa=i_idEmpresa AND idProducte=producteSistema;
      --Esborra els registres a 0
      DELETE empresesDistribuides WHERE volum=0;
   END IF;

   /*Stock*/
   --Si tot ha anat bé, actualitza el registre de Stock amb les noves dades
   UPDATE Stock SET volum=i_volum,
quantitat=i_quantitat,idTransport=i_idTransport,idEmpresa=i_idEmpresa
   WHERE idSistema=i_idSistema AND idMagatzem=i_idMagatzem AND
data=i_data;

   crearLOG($$PLSQL_UNIT,parametres,RSP);
```

```
--Excepcions
EXCEPTION
--Excepció que es crida si es passa de l'espai del magatzem
WHEN excesEspai THEN
   --Error personalitzat
   RSP:='ERROR'|| ':' ||'espai insuficient al magatzem '||i_idMagatzem;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
--Per qualsevol altra excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
   RSP:='ERROR'|| ':' ||SQLERRM;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que esborra un registre de Stock
Demana el sistema d'embalatge, el magatzem on es guarda i la data del
registre (claus)
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE esborrarStock(
   i_idSistema TFGAdmin.stock.idSistema%type,
   i_idMagatzem TFGAdmin.stock.idMagatzem%type,
   i_data TFGAdmin.stock.data%type,
   RSP OUT varchar
)
IS
   empresaAnterior TFGAdmin.stock.idEmpresa%type;
   producteSistema TFGAdmin.producte.idProducte%type;
   quantitatAnterior TFGAdmin.stock.quantitat%type;
   capacitatEnvasML TFGAdmin.envas.capacitat%type;
   volumAnterior TFGAdmin.stock.volum%type;
   espaiOcupat TFGAdmin.capacitatMagatzem.espaiOcupat%type;
   parametres varchar(500);
BEGIN
   RSP:='OK';
   parametres:=i_idSistema||','||i_idMagatzem||','||TO_CHAR(i_data,'DD-MM-
YYYY HH24:MI:SS');
   --Guarda el volum del registre anterior
   SELECT volum,quantitat INTO volumAnterior,quantitatAnterior
   FROM Stock
   WHERE idSistema=i_idSistema AND idMagatzem=i_idMagatzem AND
data=i_data;

   /*capacitat Magatzem*/
   --Guarda l'espai ocupat del registre anterior
   SELECT espaiOcupat INTO espaiOcupat
   FROM capacitatMagatzem
```

```
    WHERE idMagatzem=i_idMagatzem;
    --Treu el volum anterior de l'espai ocupat del registre anterior
    UPDATE capacitatMagatzem SET espaiOcupat=espaiOcupat-volumAnterior
WHERE idMagatzem=i_idMagatzem;
    --Esborra els registres a 0
    DELETE capacitatMagatzem WHERE espaiOcupat=0;

    /*Capacitat Anual Magatzem*/
    --Elimina el volum anterior i li treu del màxim
    UPDATE capacitatAnualMagatzems SET espaiOcupat=espaiOcupat-
volumAnterior,espaiMaximOcupat=espaiMaximOcupat-volumAnterior
    WHERE any_=EXTRACT(YEAR FROM i_data);
    --Esborra els registres a 0
    DELETE capacitatAnualMagatzems WHERE espaiMaximOcupat=0;

    /*Beguda produida anualment*/
    --Guarda la capacitat i el producte del sistema
    SELECT Envas.capacitat,Producte.idProducte INTO
capacitatEnvasML,producteSistema
    FROM SistemaEmbalatge
    INNER JOIN Producte ON
        Producte.idProducte=SistemaEmbalatge.idProducte
    INNER JOIN Envas ON
        Producte.idEnvas=Envas.idEnvas
    WHERE SistemaEmbalatge.idSistema=i_idSistema;
    --En cas que el volum fos positiu
    IF volumAnterior>0 THEN
        --Li resta a la capacitat produïda d'aquell any
        UPDATE begudaProduida SET capacitatLitres=capacitatLitres-
quantitatAnterior*capacitatEnvasML/1000
        WHERE any_=EXTRACT(YEAR FROM i_data);
        --Esborra els registres a 0
        DELETE begudaProduida WHERE capacitatLitres<=0;
    END IF;

    /*Empreses Distribuides*/
    --Si tenia registre a empresa
    IF empresaAnterior IS NOT NULL THEN
        --Li resta la quantitat anterior que no ha distribuit (la quantitat anterior està
guardada com a negatiu)
        UPDATE empresesDistribuides SET
volum=volum+quantitatAnterior*capacitatEnvasML/1000
        WHERE idEmpresa=empresaAnterior AND idProducte=producteSistema;
        --Treu els registres a 0
        DELETE empresesDistribuides WHERE volum=0;
    END IF;

    /*Stock*/
    --Si tot ha anat bé, elimina el registre de Stock
    DELETE Stock
```

```
   WHERE idSistema=i_idSistema AND idMagatzem=i_idMagatzem AND
data=i_data;

   crearLOG($$PLSQL_UNIT,parametres,RSP);

EXCEPTION
--Per qualsevol altra excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
   RSP:='ERROR'|| ':' ||SQLERRM;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que crea un registre d'assignació entre empleat i torn
Demana l'empleat, el torn assignat i la data d'assignació
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE crearAssignacio(
   i_idEmpleat TFGAdmin.assignacio.idEmpleat%type,
   i_idTorn TFGAdmin.assignacio.idTorn%type,
   i_data TFGAdmin.assignacio.data%type,
   RSP OUT varchar
)
IS
   capacitatPersones TFGAdmin.torn.capacitatPersones%type;
   tornsActuals number(5);
   personesActuals TFGAdmin.torn.capacitatPersones%type;
   existeixTornAnual number(1);
   empleatUtilitzat number(1);
   parametres varchar(500);
   comptador number(1);
BEGIN
   RSP:='OK';
   parametres:=i_idEmpleat||','||i_idTorn||','||TO_CHAR(i_data,'DD-MM-YYYY
HH24:MI:SS');
   --Guarda el nombre de registres que participa l'empleat aquell any dels torns
que ha fet
   SELECT COUNT(*) INTO empleatUtilitzat
   FROM empleatsUtilitzats
   WHERE idEmpleat=i_idEmpleat AND any_=EXTRACT(YEAR FROM i_data);
   --Insereix el registre a assignacio
   INSERT INTO assignacio VALUES(i_idEmpleat,i_idTorn,i_data);

   /*empleats utilitzats*/
   --Si no hi ha registres
   IF empleatUtilitzat=0 THEN
      --Crea el registre amb un comptador de 1
```

```
      INSERT INTO empleatsUtilitzats VALUES(EXTRACT(YEAR FROM
i_data),i_idEmpleat,1);
   --Si existeix el registre
   ELSE
      --Li suma una unitat al comptador
      UPDATE empleatsUtilitzats SET nombreTorns=nombreTorns+1 WHERE
any_=EXTRACT(YEAR FROM i_data) AND idEmpleat=i_idEmpleat;
   END IF;

   /*torns anuals*/
   --Guarda el nombre màxim de persones del torn
   SELECT capacitatPersones INTO capacitatPersones
   FROM torn
   WHERE idTorn=i_idtorn;
   --Guarda el nombre de torns
   SELECT COUNT(idTorn) INTO tornsActuals
   FROM torn;
   --Guarda el nombre d'empleats assignats d'aquell torn
   SELECT COUNT(idEmpleat) INTO personesActuals
   FROM Assignacio
   WHERE idTorn=i_idTorn;
   --Guarda els registres de torns anuals
   SELECT COUNT(any_) INTO existeixTornAnual
   FROM tornsAnuals
   WHERE any_=EXTRACT(YEAR FROM i_data);
   --Si la capacitat màxima és la mateixa que les persones que hi han
assignades, el comptador és 1
   IF capacitatPersones=personesActuals THEN
      comptador:=1;
   --Sinó, el comptador és 0
   ELSE
      comptador:=0;
   END IF;

   --si no existeix registre
   IF existeixTornAnual=0 THEN
      --Crea el registre amb el comptador de torns complerts
      INSERT INTO tornsAnuals VALUES(EXTRACT(YEAR FROM
i_data),tornsActuals,comptador);
   --si existeix el registre
   ELSE
      --Li suma la quantitat del comptador als torns complerts
      UPDATE tornsAnuals SET tornsComplerts=tornsComplerts+comptador
WHERE any_=EXTRACT(YEAR FROM i_data);
   END IF;


   crearLOG($$PLSQL_UNIT,parametres,RSP);
EXCEPTION
--Per qualsevol altra excepció
```

```
WHEN OTHERS THEN
    --Guarda el error a RSP
   RSP:='ERROR'|| ':' ||SQLERRM;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que esborra un registre d'assignació de torn d'un empleat
Demana l'empleat, el torn assignat i la data d'assignació
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE esborrarAssignacio(
   i_idEmpleat TFGAdmin.assignacio.idEmpleat%type,
   i_idTorn TFGAdmin.assignacio.idTorn%type,
   i_data TFGAdmin.assignacio.data%type,
   RSP OUT varchar
)
IS
   capacitatPersones TFGAdmin.torn.capacitatPersones%type;
   personesAnteriors TFGAdmin.torn.capacitatPersones%type;
   parametres varchar(500);
BEGIN
   RSP:='OK';
   parametres:=i_idEmpleat||','||i_idTorn||','||TO_CHAR(i_data,'DD-MM-YYYY
HH24:MI:SS');
   --Guarda els registres d'assignació d'aquell torn abans de l'esborrat
   SELECT COUNT(idEmpleat) INTO personesAnteriors
   FROM Assignacio
   WHERE idTorn=i_idTorn;

   /*Assignacio*/
   --Esborra el registre d'assignació
   DELETE assignacio WHERE idEmpleat=i_idEmpleat AND idTorn=i_idTorn
AND data=i_data;

   /*empleats utilitzats*/
   --Treu un comptador al nombre de torns d'aquell empleat d'aquell any
   UPDATE empleatsUtilitzats SET nombreTorns=nombreTorns-1 WHERE
any_=EXTRACT(YEAR FROM i_data) AND idEmpleat=i_idEmpleat;
   --Esborra els registres a 0
   DELETE empleatsUtilitzats WHERE nombreTorns=0;

   /*torns anuals*/
   --Guarda el nombre màxim de persones per torn
   SELECT capacitatPersones INTO capacitatPersones
   FROM torn
   WHERE idTorn=i_idtorn;
   --Si la capacitat màxima és la mateixa que el nombre de persones d'abans
d'esborrar el registre
```

```
    IF capacitatPersones=personesAnteriors THEN
        --Treu un comptador al nombre de torns complerts d'aquell any
        UPDATE tornsAnuals SET tornsComplerts=tornsComplerts-1 WHERE
any_=EXTRACT(YEAR FROM i_data);
    END IF;
    --Treu els registres a 0
    DELETE tornsAnuals WHERE tornsComplerts=0;

    crearLOG($$PLSQL_UNIT,parametres,RSP);
EXCEPTION
--Per qualsevol altra excepció
WHEN OTHERS THEN
     --Guarda el error a RSP
    RSP:='ERROR'|| ':' ||SQLERRM;
    --Crea el log amb el nom del procediment, paràmetre i variable d'error
    crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que crea un registre de demanda
Demana el producte, la zona, la data i la quantitat de la demanda
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE crearDemanda(
    i_idProducte IN TFGAdmin.demanda.idProducte%type,
    i_idZona IN TFGAdmin.demanda.idZona%type,
    i_data IN TFGAdmin.demanda.data%type,
    i_quantitat IN TFGAdmin.demanda.quantitat%type,
    RSP OUT varchar
)
IS
    parametres varchar(500);
    registreDemandaAnual number(1);
BEGIN
    RSP:='OK';
    parametres:=i_idProducte||','||i_idZona||','||TO_CHAR(i_data,'DD-MM-YYYY
HH24:MI:SS')||','||i_quantitat;
    --Insereix el registre de demanda
    INSERT INTO Demanda VALUES(i_idProducte,i_idZona,i_data,i_quantitat);

    /*demanda anual*/
    --Guarda el nombre de registres de demanda anual
    SELECT COUNT(*) INTO registreDemandaAnual
    FROM demandaAnual
    WHERE any_=EXTRACT(YEAR FROM i_data) AND
idProducte=i_idProducte;
    --Si no hi ha registre
    IF registreDemandaAnual=0 THEN
        --Crea un registre amb un comptador a 1 als dies de demanda
```

```
      INSERT INTO demandaAnual VALUES(EXTRACT(YEAR FROM
i_data),i_idProducte,i_quantitat,1);
   --Si hi ha registre
   ELSE
      --Suma la nova quantitat i puja una unitat el comptador de dies
      UPDATE demandaAnual SET
quantitat=quantitat+i_quantitat,quantitatDies=quantitatDies+1
      WHERE any_=EXTRACT(YEAR FROM i_data) AND
idProducte=i_idProducte;
   END IF;

   crearLOG($$PLSQL_UNIT,parametres,RSP);
EXCEPTION
--Per qualsevol altra excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
   RSP:='ERROR'|| ':' ||SQLERRM;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que esborra un registre de demanda
Demana el producte, la zona i la data de la demanda (claus)
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE esborrarDemanda(
   i_idProducte IN TFGAdmin.demanda.idProducte%type,
   i_idZona IN TFGAdmin.demanda.idZona%type,
   i_data IN TFGAdmin.demanda.data%type,
   RSP OUT varchar
)
IS
   parametres varchar(500);
   quantitatAnterior TFGAdmin.demanda.quantitat%type;
BEGIN
   RSP:='OK';
   parametres:=i_idProducte||','||i_idZona||','||TO_CHAR(i_data,'DD-MM-YYYY
HH24:MI:SS');
   --Guarda la quantitat anterior de demanda del registre a esborrar
   SELECT quantitat INTO quantitatAnterior
   FROM demanda
   WHERE idProducte=i_idProducte AND idZona=i_idZona AND data=i_data;
   --Esborra el registre de demanda
   DELETE demanda WHERE idProducte=i_idProducte AND idZona=i_idZona
AND data=i_data;

   /*demanda anual*/
   --Treu la quantitat i resta una unitat al comptador de dies de demanda anual
```

```
   UPDATE demandaAnual SET quantitat=quantitat-
quantitatAnterior,quantitatDies=quantitatDies-1
   WHERE any_=EXTRACT(YEAR FROM i_data) AND
idProducte=i_idProducte;
   --Esborra els registres a 0
   DELETE demandaAnual WHERE quantitat=0;

   crearLOG($$PLSQL_UNIT,parametres,RSP);
EXCEPTION
--Per qualsevol altra excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
   RSP:='ERROR'|| ':' ||SQLERRM;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que modifica un registre de demanda
Demana el producte, la zona, la data i la quantitat de la demanda
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE modificarDemanda(
   i_idProducte IN TFGAdmin.demanda.idProducte%type,
   i_idZona IN TFGAdmin.demanda.idZona%type,
   i_data IN TFGAdmin.demanda.data%type,
   i_quantitat IN TFGAdmin.demanda.quantitat%type,
   RSP OUT varchar
)
IS
   parametres varchar(500);
   quantitatAnterior TFGAdmin.demanda.quantitat%type;
BEGIN
   RSP:='OK';
   parametres:=i_idProducte||','||i_idZona||','||TO_CHAR(i_data,'DD-MM-YYYY
HH24:MI:SS')||','||i_quantitat;
   --guarda la quantitat anterior del registre a modificar
   SELECT quantitat INTO quantitatAnterior
   FROM demanda
   WHERE idProducte=i_idProducte AND idZona=i_idZona AND data=i_data;
   --Modifica el registre de demanda
   UPDATE demanda SET quantitat=i_quantitat WHERE
idProducte=i_idProducte AND idZona=i_idZona AND data=i_data;

   /*demanda anual*/
   --Actualitza el registre de demanda anual treient l'anterior quantitat i sumant
la nova
   UPDATE demandaAnual SET quantitat=quantitat-
quantitatAnterior+i_quantitat
```

```
   WHERE any_=EXTRACT(YEAR FROM i_data) AND
idProducte=i_idProducte;
   --Esborra els registres a 0
   DELETE demandaAnual WHERE quantitat=0;

   crearLOG($$PLSQL_UNIT,parametres,RSP);
EXCEPTION
--Per qualsevol altra excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
   RSP:='ERROR'|| ':' ||SQLERRM;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que crea un registre de netejament
Demana el grup de neteja, la línia netejada, la data d'inici i la data de fi del
netejament
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE crearNetejament(
   i_idNeteja TFGAdmin.netejament.idNeteja%type,
   i_idLinia TFGAdmin.netejament.idLinia%type,
   i_dataInici TFGAdmin.netejament.dataInici%type,
   i_dataFi TFGAdmin.netejament.dataFi%type,
   RSP OUT varchar
)
IS
   parametres varchar(500);
   existeixNetejaAnual number(1);
BEGIN
   RSP:='OK';
   parametres:=i_idNeteja||','||i_idLinia||','||TO_CHAR(i_dataInici,'DD-MM-YYYY
HH24:MI:SS')||','||TO_CHAR(i_dataFi,'DD-MM-YYYY HH24:MI:SS');

   /*Neteja anual*/
   --Guarda els registres de neteja anual d'aquesta línia
   SELECT COUNT(any_) INTO existeixNetejaAnual
   FROM netejaAnual
   WHERE any_=EXTRACT(YEAR FROM i_dataFi) AND idLinia=i_idLinia;
   --Si no hi ha registre
   IF existeixNetejaAnual=0 THEN
      --Insereix el registre amb les hores de neteja
      INSERT INTO netejaAnual VALUES(EXTRACT(YEAR FROM
i_dataFi),i_idLinia,(i_dataFi-i_dataInici)*24);
   --Si hi ha registre
   ELSE
      --Suma les hores de neteja a la línia
```

```
      UPDATE netejaAnual SET hores=hores+(i_dataFi-i_dataInici)*24 WHERE
any_=EXTRACT(YEAR FROM i_dataFi) AND idLinia=i_idLinia;
   END IF;

   /*Netejament*/
   --Insereix el registre de netejament
   INSERT INTO netejament VALUES(i_idNeteja,i_idLinia,i_dataInici,i_dataFi);

   crearLOG($$PLSQL_UNIT,parametres,RSP);
EXCEPTION
--Per qualsevol altra excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
   RSP:='ERROR'|| ':' ||SQLERRM;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Procediment que modifica un registre de netejament
Demana el grup de neteja, la línia netejada, la data d'inici i la data de fi del
netejament
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE modificarNetejament(
   i_idNeteja TFGAdmin.netejament.idNeteja%type,
   i_idLinia TFGAdmin.netejament.idLinia%type,
   i_dataInici TFGAdmin.netejament.dataInici%type,
   i_dataFi TFGAdmin.netejament.dataFi%type,
   RSP OUT varchar
)
IS
   dataFiAnterior TFGAdmin.netejament.dataFi%type;
   parametres varchar(500);
BEGIN
   RSP:='OK';
   parametres:=i_idNeteja||','||i_idLinia||','||TO_CHAR(i_dataInici,'DD-MM-YYYY
HH24:MI:SS')||','||TO_CHAR(i_dataFi,'DD-MM-YYYY HH24:MI:SS');
   --Guarda l'anterior data de fi del registre a modificar
   SELECT dataFi INTO dataFiAnterior
   FROM netejament
   WHERE idNeteja=i_idNeteja AND idLinia=i_idLinia AND dataInici=i_dataInici;

   /*Neteja anual*/
   --Treu les hores anteriors i suma les noves hores (com que la data d'inici és
part de la clau, només poden modificar la data fi)
   UPDATE netejaAnual SET hores=hores-(dataFiAnterior-
i_dataInici)*24+(i_dataFi-i_dataInici)*24 WHERE any_=EXTRACT(YEAR
FROM i_dataFi) AND idLinia=i_idLinia;
   --Esborra els registres a 0
```

```
    DELETE netejaAnual WHERE hores=0;

    /*Netejament*/
    --Actualitza el registre de netejament
    UPDATE netejament SET dataFi=i_dataFi WHERE idNeteja=i_idNeteja AND
idLinia=i_idLinia AND dataInici=i_dataInici;

    crearLOG($$PLSQL_UNIT,parametres,RSP);
EXCEPTION
--Per qualsevol altra excepció
WHEN OTHERS THEN
     --Guarda el error a RSP
    RSP:='ERROR'|| ':' ||SQLERRM;
    --Crea el log amb el nom del procediment, paràmetre i variable d'error
    crearLOG($$PLSQL_UNIT,parametres,RSP);
END;

/*
Esborra un registre de netejament
Demana el grup de neteja, la línia netejada i la data d'inici (claus)
Retorna RSP variable d'errors
*/
CREATE OR REPLACE PROCEDURE esborrarNetejament(
    i_idNeteja TFGAdmin.netejament.idNeteja%type,
    i_idLinia TFGAdmin.netejament.idLinia%type,
    i_dataInici TFGAdmin.netejament.dataInici%type,
    RSP OUT varchar
)
IS
    dataFiAnterior TFGAdmin.netejament.dataFi%type;
    parametres varchar(500);
BEGIN
    RSP:='OK';
    parametres:=i_idNeteja||','||i_idLinia||','||TO_CHAR(i_dataInici,'DD-MM-YYYY
HH24:MI:SS');
    --Guarda la data fi anterior al registre a esborrar
    SELECT dataFi INTO dataFiAnterior
    FROM netejament
    WHERE idNeteja=i_idNeteja AND idLinia=i_idLinia AND dataInici=i_dataInici;

    /*Neteja anual*/
    --Resta les hores anteriors
    UPDATE netejaAnual SET hores=hores-(dataFiAnterior-i_dataInici)*24
WHERE any_=EXTRACT(YEAR FROM dataFiAnterior) AND idLinia=i_idLinia;
    --Esborra els registres a 0
    DELETE netejaAnual WHERE hores=0;

    /*Netejament*/
    --Esborra el registre de netejament
```

```
   DELETE netejament WHERE idNeteja=i_idNeteja AND idLinia=i_idLinia AND
dataInici=i_dataInici;

   crearLOG($$PLSQL_UNIT,parametres,RSP);
EXCEPTION
--Per qualsevol altra excepció
WHEN OTHERS THEN
    --Guarda el error a RSP
   RSP:='ERROR'|| ':' ||SQLERRM;
   --Crea el log amb el nom del procediment, paràmetre i variable d'error
   crearLOG($$PLSQL_UNIT,parametres,RSP);
END;
```

Annex V: Scripts de consultes (consultes.sql)

```
/*Component més utilitzat*/
--Troba el primer component més utilitzat i el nombre de compostos que
participa
SELECT t.idComponent, t.nombrecomposts
FROM
--Subconsulta que forma la taula de consulta
(
   --Ordena els components pel nombre de composts que participa
descendentment (major participació adalt)
   SELECT idComponent, nombrecomposts
   FROM ComponentUtilitzat
   ORDER BY NombreComposts DESC
) t
--Sentència que fa el TOP 1
WHERE rownum=1;

/*Producte més produit*/
--Troba el producte més produït amb la seva quantitat produïda per un any
concret
SELECT t.idProducte, t.quantitat
FROM
--Subconsulta que forma la taula de consulta
(
   --Ordena els productes descendentment en funció a la quantitat (els que
tenen més quantitat adalt)
   SELECT idProducte, quantitat
   FROM ProducteProduit
   --Limita el resultat a l'any indicat
   WHERE any_=2019
   ORDER BY quantitat DESC
) t
--Sentència que fa el TOP 1
WHERE rownum=1;

/*Línea més aturada*/
```

```
--Troba la primera línia amb més hores aturades per l'any i mes indicats
-- i mostra la línia, les hores que ha estat funcionant i les hores d'aturada respecte el total d'hores d'aquell mes
SELECT t.idLinia,t.horesFuncionant,t.horesAturada
FROM
--Subconsulta que forma la taula de consulta
(
   --Agafa la línia, l'any, mes, hores que es troba funcionant
   -- i fa el càlcul de totes les hores d'aquell mes i li resta les hores que ha funcionat per obtenir les hores d'aturada
   -- ho ordena segons aquestes hores descendentment (la que té més hores d'aturada adalt)
   SELECT idLinia,any_,mes,horesfuncionant, (1+trunc(last_day(to_date('01-'||LPAD(mes,2,'0')||'-'||any_,'DD-MM-YYYY')))-trunc(to_date('01-'||LPAD(mes,2,'0')||'-'||any_,'DD-MM-YYYY'),'MM'))*24-horesFuncionant horesAturada
   FROM LiniaAturada
   --Limita l'any i el mes
   WHERE any_=2019 AND mes=5
   ORDER BY horesAturada DESC
) t
--Sentència que fa el TOP 1
WHERE rownum=1;

/*Capacitat emmagatzematge actual magatzem*/
--Troba el magatzem, el seu espai ocupat, la seva capacitat màxima i l'espai disponible del magatzem indicat en el moment de consulta
SELECT capacitatMagatzem.idMagatzem,capacitatMagatzem.espaiOcupat, magatzem.capacitat,(magatzem.capacitat-capacitatMagatzem.espaiOcupat) espaiDisponible
FROM capacitatMagatzem
INNER JOIN magatzem ON
   magatzem.idMagatzem=capacitatMagatzem.idMagatzem
--Limita el magatzem
WHERE capacitatMagatzem.idMagatzem='M0001';

/*Empleats més utilitzats*/
--Troba els 10 empleats més utilitzats i el nombre de torns que ha realitzat cadascú per l'any indicat
SELECT t.idEmpleat,t.nombreTorns
FROM
--Subconsulta que forma la taula de consulta
(
   --Ordena segons el nombre de torns que han realitzat els empleats descendentment (qui ha treballat més torns adalt)
   SELECT idEmpleat,nombreTorns
   FROM empleatsUtilitzats
   --filtra per any
   WHERE any_=2019
   ORDER BY nombreTorns DESC
```

```
) t
--Sentència que fa el TOP 10
WHERE rownum<=10;

/*Increment percentual de demanda anual*/
--Troba la demanda mitjana l'any anterior (quantitat en relació als dies de demanda registrats) i el seu increment respecte
-- l'any anterior per un any i producte concret
SELECT t2.demandaMitjAnyAnterior,t.demandaMitjanaAnyActual,
   ((t.demandaMitjanaAnyActual-
t2.demandaMitjAnyAnterior)/(t2.demandaMitjAnyAnterior))*100
incrDecrPercentAnyActual
FROM
--Subconsulta que forma la primera taula de consulta
(
   --Troba la quantitat, els dies de registre de la demana i la demanda mitjana per l'any i producte demanats
   SELECT                    quantitat,quantitatdies,quantitat/quantitatdies
demandaMitjanaAnyActual
   FROM demandaAnual
   --Filtra per un producte i any concret
   WHERE idProducte='P0002' AND any_=2019
) t,
--Subconsulta que forma la segona taula de consulta
(
   --Troba la quantitat, els dies de registre de la demana i la demanda mitjana per l'any anterior i producte demanats
   SELECT                    quantitat,quantitatdies,quantitat/quantitatdies
demandaMitjAnyAnterior
   FROM demandaAnual
   --Filtra per un producte i any concret (ha de ser l'anterior al de abans)
   WHERE idProducte='P0002' AND any_=2018
) t2;

/*Any menor capacitat emmagatzematge*/
--Troba l'any i l'espai ocupat on el total de l'espai ocupat entre tots els magatzems s'ha trobat al mínim
SELECT t.any_,t.espaiOcupat
FROM
--Subconsulta que forma la taula de consulta
(
   --Ordena segons l'espai ocupat ascendentment (els registres amb menor espai ocupat adalt)
   SELECT any_,espaiOcupat
   FROM
   capacitatAnualMagatzems
   ORDER BY espaiOcupat ASC
) t
--Sentència que fa el TOP 1
WHERE rownum=1;
```

```
/*Increment percentual de beguda produida*/
--Troba la quantitat de litres produïts d'aquest any i de l'any anterior i percentatge d'increment (o decrement) entre aquestes quantitats
SELECT t2.capacitatLitresAnyAnterior,t.capacitatLitresAnyActual,
   ((t.capacitatLitresAnyActual-
t2.capacitatLitresAnyAnterior)/(t2.capacitatLitresAnyAnterior))*100
incrDecrPercentAnyActual
FROM
--Subconsulta que forma la primera taula de consulta
(
   --Troba la quantitat de litres produïts de l'any de consulta
   SELECT capacitatLitres capacitatLitresAnyActual
   FROM begudaProduida
   --Filtra per l'any de consulta
   WHERE any_=EXTRACT(YEAR FROM SYSDATE)
) t,
--Subconsulta que forma la segona taula de consulta
(
   --Troba la quantitat de litres produïts de l'any anterior de consulta
   SELECT capacitatLitres capacitatLitresAnyAnterior
   FROM begudaProduida
   --Filtra per l'any anterior de consulta
   WHERE any_=EXTRACT(YEAR FROM ADD_MONTHS( SYSDATE,-12))
) t2;

/*Any amb major temps de neteja*/
--Troba l'any, la línia i les hores dedicades de neteja on s'ha dedicat més hores per l'any indicat
SELECT t.idLinia,t.hores,any_
FROM
--Subconsulta que forma la taula de consulta
(
   --Ordena les línies segons les hores de neteja descendentment (les que han dedicat més temps adalt)
   SELECT idLinia,hores,any_
   FROM netejaAnual
   --Filtra per un any concret
   WHERE any_=2019
   ORDER BY hores DESC
) t
--Sentència que fa el TOP 1
WHERE rownum=1;

/*Percentatge de torns incomplerts*/
--Troba el nombre de torns totals realitzats, els d'aquests que es troben complerts i el percentatge de torns incomplerts
-- d'un any indicat
SELECT tornsTotals,tornsComplerts,((tornsTotals-
tornsComplerts)/tornsTotals)*100 percentatgeTornsIncomplerts
```

```
FROM tornsAnuals
--Filtra per un any indicat
WHERE any_=2019;

/*Primeres cinc empreses de distribució de producte*/
--Troba les primeres cinc empreses que més volum han distribuït i de quin producte ha sigut
SELECT t.idEmpresa,t.idProducte,t.volum
--Subconsulta que forma la taula de consulta
FROM (
   --Ordena les línies segons el volum descendentment (major volum distribuït adalt)
   SELECT idEmpresa,idProducte,volum
   FROM empresesDistribuides
   ORDER BY volum DESC
) t
--Sentència que fa el TOP 5
WHERE rownum<=5;
```

Annex VI: Scripts de proves i generació de dades

```
/*Component més utilitzat*/
--Troba el primer component més utilitzat i el nombre de compostos que participa
SELECT t.idComponent, t.nombrecomposts
FROM
--Subconsulta que forma la taula de consulta
(
   --Ordena els components pel nombre de composts que participa descendentment (major participació adalt)
   SELECT idComponent, nombrecomposts
   FROM ComponentUtilitzat
   ORDER BY NombreComposts DESC
) t
--Sentència que fa el TOP 1
WHERE rownum=1;

/*Producte més produit*/
--Troba el producte més produït amb la seva quantitat produïda per un any concret
SELECT t.idProducte, t.quantitat
FROM
--Subconsulta que forma la taula de consulta
(
   --Ordena els productes descendentment en funció a la quantitat (els que tenen més quantitat adalt)
   SELECT idProducte, quantitat
   FROM ProducteProduit
   --Limita el resultat a l'any indicat
   WHERE any_=2019
```

```
   ORDER BY quantitat DESC
) t
--Sentència que fa el TOP 1
WHERE rownum=1;

/*Línea més aturada*/
--Troba la primera línia amb més hores aturades per l'any i mes indicats
-- i mostra la línia, les hores que ha estat funcionant i les hores d'aturada respecte el total d'hores d'aquell mes
SELECT t.idLinia,t.horesFuncionant,t.horesAturada
FROM
--Subconsulta que forma la taula de consulta
(
   --Agafa la línia, l'any, mes, hores que es troba funcionant
   -- i fa el càlcul de totes les hores d'aquell mes i li resta les hores que ha funcionat per obtenir les hores d'aturada
   -- ho ordena segons aquestes hores descendentment (la que té més hores d'aturada adalt)
   SELECT idLinia,any_,mes,horesfuncionant, (1+trunc(last_day(to_date('01-'||LPAD(mes,2,'0')||'-'||any_,'DD-MM-YYYY')))-trunc(to_date('01-'||LPAD(mes,2,'0')||'-'||any_,'DD-MM-YYYY'),'MM'))*24-horesFuncionant horesAturada
   FROM LiniaAturada
   --Limita l'any i el mes
   WHERE any_=2019 AND mes=5
   ORDER BY horesAturada DESC
) t
--Sentència que fa el TOP 1
WHERE rownum=1;

/*Capacitat emmagatzematge actual magatzem*/
--Troba el magatzem, el seu espai ocupat, la seva capacitat màxima i l'espai disponible del magatzem indicat en el moment de consulta
SELECT capacitatMagatzem.idMagatzem,capacitatMagatzem.espaiOcupat, magatzem.capacitat,(magatzem.capacitat-capacitatMagatzem.espaiOcupat) espaiDisponible
FROM capacitatMagatzem
INNER JOIN magatzem ON
   magatzem.idMagatzem=capacitatMagatzem.idMagatzem
--Limita el magatzem
WHERE capacitatMagatzem.idMagatzem='M0001';

/*Empleats més utilitzats*/
--Troba els 10 empleats més utilitzats i el nombre de torns que ha realitzat cadascú per l'any indicat
SELECT t.idEmpleat,t.nombreTorns
FROM
--Subconsulta que forma la taula de consulta
(
```

```
   --Ordena segons el nombre de torns que han realitzat els empleats
descendentment (qui ha treballat més torns adalt)
   SELECT idEmpleat,nombreTorns
   FROM empleatsUtilitzats
   --filtra per any
   WHERE any_=2019
   ORDER BY nombreTorns DESC
) t
--Sentència que fa el TOP 10
WHERE rownum<=10;

/*Increment percentual de demanda anual*/
--Troba la demanda mitjana l'any anterior (quantitat en relació als dies de
demanda registrats) i el seu increment respecte
-- l'any anterior per un any i producte concret
SELECT t2.demandaMitjAnyAnterior,t.demandaMitjanaAnyActual,
   ((t.demandaMitjanaAnyActual-
t2.demandaMitjAnyAnterior)/(t2.demandaMitjAnyAnterior))*100
incrDecrPercentAnyActual
FROM
--Subconsulta que forma la primera taula de consulta
(
   --Troba la quantitat, els dies de registre de la demana i la demanda mitjana
per l'any i producte demanats
   SELECT                    quantitat,quantitatdies,quantitat/quantitatdies
demandaMitjanaAnyActual
   FROM demandaAnual
   --Filtra per un producte i any concret
   WHERE idProducte='P0002' AND any_=2019
) t,
--Subconsulta que forma la segona taula de consulta
(
   --Troba la quantitat, els dies de registre de la demana i la demanda mitjana
per l'any anterior i producte demanats
   SELECT                    quantitat,quantitatdies,quantitat/quantitatdies
demandaMitjAnyAnterior
   FROM demandaAnual
   --Filtra per un producte i any concret (ha de ser l'anterior al de abans)
   WHERE idProducte='P0002' AND any_=2018
) t2;

/*Any menor capacitat emmagatzematge*/
--Troba l'any i l'espai ocupat on el total de l'espai ocupat entre tots els
magatzems s'ha trobat al mínim
SELECT t.any_,t.espaiOcupat
FROM
--Subconsulta que forma la taula de consulta
(
   --Ordena segons l'espai ocupat ascendentment (els registres amb menor
espai ocupat adalt)
```

```
    SELECT any_,espaiOcupat
    FROM
    capacitatAnualMagatzems
    ORDER BY espaiOcupat ASC
) t
--Sentència que fa el TOP 1
WHERE rownum=1;

/*Increment percentual de beguda produida*/
--Troba la quantitat de litres produïts d'aquest any i de l'any anterior i percentatge d'increment (o decrement) entre aquestes quantitats
SELECT t2.capacitatLitresAnyAnterior,t.capacitatLitresAnyActual,
    ((t.capacitatLitresAnyActual-
t2.capacitatLitresAnyAnterior)/(t2.capacitatLitresAnyAnterior))*100
incrDecrPercentAnyActual
FROM
--Subconsulta que forma la primera taula de consulta
(
    --Troba la quantitat de litres produïts de l'any de consulta
    SELECT capacitatLitres capacitatLitresAnyActual
    FROM begudaProduida
    --Filtra per l'any de consulta
    WHERE any_=EXTRACT(YEAR FROM SYSDATE)
) t,
--Subconsulta que forma la segona taula de consulta
(
    --Troba la quantitat de litres produïts de l'any anterior de consulta
    SELECT capacitatLitres capacitatLitresAnyAnterior
    FROM begudaProduida
    --Filtra per l'any anterior de consulta
    WHERE any_=EXTRACT(YEAR FROM ADD_MONTHS( SYSDATE,-12))
) t2;

/*Any amb major temps de neteja*/
--Troba l'any, la línia i les hores dedicades de neteja on s'ha dedicat més hores per l'any indicat
SELECT t.idLinia,t.hores,any_
FROM
--Subconsulta que forma la taula de consulta
(
    --Ordena les línies segons les hores de neteja descendentment (les que han dedicat més temps adalt)
    SELECT idLinia,hores,any_
    FROM netejaAnual
    --Filtra per un any concret
    WHERE any_=2019
    ORDER BY hores DESC
) t
--Sentència que fa el TOP 1
WHERE rownum=1;
```

```
/*Percentatge de torns incomplerts*/
--Troba el nombre de torns totals realitzats, els d'aquests que es troben complerts i el percentatge de torns incomplerts
-- d'un any indicat
SELECT tornsTotals,tornsComplerts,((tornsTotals-tornsComplerts)/tornsTotals)*100 percentatgeTornsIncomplerts
FROM tornsAnuals
--Filtra per un any indicat
WHERE any_=2019;

/*Primeres cinc empreses de distribució de producte*/
--Troba les primeres cinc empreses que més volum han distribuït i de quin producte ha sigut
SELECT t.idEmpresa,t.idProducte,t.volum
--Subconsulta que forma la taula de consulta
FROM (
   --Ordena les línies segons el volum descendentment (major volum distribuït adalt)
   SELECT idEmpresa,idProducte,volum
   FROM empresesDistribuides
   ORDER BY volum DESC
) t
--Sentència que fa el TOP 5
WHERE rownum<=5;
```