Universitat Oberta de Catalunya

Projecte Final de Carrera

Enginyeria Tècnica Informàtica

Curs 2005/06 - Quadrimestre 1

Disseny i implementació d'un sistema de control d'un consultori mèdic

Memòria del Projecte

Autor: **Jordi Aguilar Martínez (jaguilarm@uoc.edu)**

Director: **Alexandre Cornet Arnaez**

9 de gener de 2006

Índex

1 Resum....4

2 Introducció 5
2.1 Descripció del projecte.... 5
2.2 Estudi previ 5
2.2.1 Disseny de la base de dades 6
2.2.2 Controls d'integritat.... 7
2.2.3 Tractament d'esborrat de dades.... 7
2.2.4 Tractament d'excepcions.... 7
2.2.5 Procediments emmagatzemats 8
2.2.6 Sistema de registre d'activitat (o 'log')..... 8
2.2.7 Funcions que haurem de donar a la nostra interfície 8
2.3 Objectius generals 9
2.4 Entregables del projecte 9
2.5 Anàlisi de riscos 9

3 Planificació del Projecte....10
3.1 Relació de tasques 10
3.1.1 Definició del projecte 10
3.1.2 Preparació de l'entorn de treball.... 10
3.1.3 Formació.... 11
3.1.4 Anàlisi i disseny 11
3.1.5 Implementació 12
3.1.6 Proves del sistema 13
3.1.7 Finalització documentació 13
3.1.8 Preparació presentació PPT.... 13
3.1.9 Debat TFC amb el tribunal 13
3.2 Fites 13
3.3 Calendari de Treball.... 14

4 Anàlisis i Disseny....15
4.1 Estudi de les dades que tractarem al nostre sistema.... 15
4.1.1 Dades sobre persones 15
4.1.2 Dades sobre els horaris dels metges i del centre 15
4.1.3 Casos, visites i documentació 17
4.1.4 Medicaments i al·lèrgies 17
4.1.5 Altres dades del sistema 18
4.2 Disseny conceptual (model E-R) 19
4.2.1 Identificació d'entitats del nostre sistema.... 19
4.2.2 Diagrama Entitat-Relació.... 20
4.2.3 Comentaris al diagrama E-R : 21
4.3 Disseny lògic (model relacional) 22
4.3.1 Definició de les taules i camps 22
4.3.2 Esquema de les taules de la Base de Dades.... 27
4.3.3 Controls d'integritat.... 28
4.3.4 Índexs 29
4.4 Estudi de les funcions que s'han de poder fer amb les dades del sistema.... 30
4.4.1 Comportament comú dels procediments emmagatzemats de la interfície 30
4.4.1.1 Paràmetre de sortida RSP 30
4.4.1.2 Paràmetre d'entrada USER 30
4.4.1.3 Comprovar que no existeixi una dada abans d'introduir-la 31
4.4.1.4 Comprovar que existeixin tots els registres als que s'està fent referència... 31
4.4.1.5 Data d'alta : data del sistema.... 31
4.4.1.6 Comprovar que existeixi allò que es vol eliminar i que es pugui eliminar 31
4.4.1.7 Tractament d'excepcions 31
4.4.2 Definició dels procediments.... 31
4.5 Disseny del sistema de registre d'activitat (Log) 35

5 Implementació 36
5.1 Preparació del sistema (instal·lació de la plataforma de treball) 36
5.1.1 Instal·lació Oracle 36
5.1.2 Requisits del sistema 36
5.1.3 Preparació del sistema operatiu per l'instal·lació 36
5.1.4 Instal·lació de l'SGBD 36
5.1.5 Iniciar/Detenir serveis d'oracle 39
5.1.6 Contrasenyes i usuaris 40
5.2 Instal·lació PL-SQL Developer 41
5.2.1 Instal·lació de PL-SQL Developer 41
5.2.2 Paquets d'idioma 43
5.2.3 Execució d'scripts SQL des de PL-SQL Developer 44
5.3 Accés a la Base de Dades des de Microsoft Access 45
5.3.1 Creació d'un DSN 45
5.3.2 Vincular taules des de Microsoft Access 47
5.3.3 Error consultant les dades des d'Access 50
5.4 Aspectes generals de la implementació 50
5.4.1 Codi estructurat i comentat 50
5.4.2 Tipus de les variables : %type 50
5.4.3 Paràmetres als procediments 51
5.4.4 Scripts 52
5.4.5 Camps autonumèrics : Seqüències 53
5.4.6 Sistema de gestió d'hores 54
5.4.6.1 Solapaments de franges horàries a l'horari d'un metge 54
5.4.6.2 Programació de visites, comprovació d'hores disponibles 55
5.4.6.3 Format dels camps d'hores de visites i durades de visites 56
5.4.7 Es demana un camp usuari per executar cada procediment 57
5.4.8 Funcions per donar d'alta usuaris 57
5.4.9 Package i funcions 58
5.4.10 Sistema pel control d'excepcions 58
5.4.11 Valors de resultats de sortida dels procediments (RSP) 59
5.4.12 Sistema de LOG 60
5.4.13 Comparacions : Case Insensitive 60
5.5 Scripts de creació de taules 61
5.5.1 Tipus de dades fets servir pels camps de les taules 61
5.5.2 Llistat de les taules del sistema 61
5.6 Implementació dels controls d'integritat 62
5.6.1 Implementació de Triggers 63
5.7 Scripts de creació d'índexs 63
5.8 Scripts de creació de Seqüències 64
5.9 Implementació dels procediments 64
5.9.1 Codi d'exemple del procediment AltaMedicament 65
5.10 Variacions entre Anàlisis i Implementació 66
6 Proves 67
7 Conclusions 68

1 Resum

L'objectiu d'aquest projecte ha estat crear un sistema de control de les dades que necessita emmagatzemar un consultori mèdic. No es tracta d'un cas real, sinó d'un exercici plantejat pel Director del projecte. El Director del projecte ha estat qui ha plantejat les necessitats, i en cas de dubtes, ha estat ell qui ha actuat com a client per a prendre les decisions necessàries.

Tot el desenvolupament del projecte s'ha fet en Oracle, un entorn desconegut per a mi abans de començar aquest projecte.

La base de dades que desenvoluparem no és només un conjunt de taules i consultes, sinó que crearem tot un sistema per administrar les dades que guardarem a les taules de la base de dades. Aquest sistema no serà un sistema final per un usuari, sinó que serà una capa intermèdia del sistema final, la capa de la base de dades.

Aquesta capa és independent del programari que la utilitzi. L'aplicatiu (o els aplicatius) que en un futur es puguin dissenyar per treballar contra aquesta base de dades, podran accedir a les taules del sistema en mode lectura, però no podran fer modificacions de les dades directament sobre les taules. Per a les insercions, modificacions i supressions, hauran de fer servir el conjunt de procediments emmagatzemats que facilitem en el nostre sistema.

La capa de la base de dades, llavors, queda com una capa robusta i independent. Aquest sistema s'encarrega de fer totes les comprovacions abans de fer qualsevol modificació sobre les dades de les taules per assegurar la integritat de les dades del sistema.

Tres avantatges d'aquest sistema :

- Fiabilitat de les dades

  Aquest sistema de treball incrementa molt la fiabilitat de les dades, ja que es limita el codi que permet fer modificacions sobre les taules. En cas d'haver algun error d'inconsistència de dades, el codi que s'hauria de revisar seria només el dels procediments emmagatzemats que han permès la modificació de dades que ha donat lloc a aquesta inconsistència, mai el de totes les aplicacions que treballin contra el nostre sistema.

- Possibilitat de tenir un Log d'activitat detallat

  Degut a que totes les modificacions a les dades passaran pels nostres procediments, amb la crida a cada procediment s'inclourà una línia a la taula del registre d'activitat, per tenir controlat en tot moment : quines dades s'han inserit, en quin ordre, en quina data, quin usuari les ha inserit, quins paràmetres ha passat al procediment, etc, etc. En cas de tenir un problema algun cop amb el sistema, aquesta taula ens serà de gran ajuda per poder reconstruir i entendre l'escenari de l'error.

- Facilitat d'ús per a tercers

  Els programadors que hagin de desenvolupar aplicacions per treballar amb el nostre sistema no hauran de conèixer amb detall totes les restriccions que han de mantenir les dades. Bé, és clar que les hauran de conèixer per no fer funcions que fallin, però podran tenir la tranquil·litat que les funcions que executin, si els donen resultat correcte, estaran afegint dades al sistema que no trencaran la consistència.

2 Introducció

2.1 Descripció del projecte

En aquest projecte es desenvoluparà una part d'un sistema informàtic per gestionar les necessitats d'un consultori mèdic.

Es desenvoluparà la base de dades que farà servir posteriorment un aplicatiu dissenyat i implementat per un altre enginyer o equip de treball.

Com a desenvolupament de la base de dades entenem el disseny i desenvolupament de tot el que hagi de contemplar el nostre Sistema Gestor de Bases de Dades : Disseny i creació de les taules, vistes, creació dels índexs que es creguin convenients en funció de les funcions que preveiem que més es faran, creació de triggers, implementació de procediments emmagatzemats.

L'aplicació que treballi posteriorment contra el nostre sistema tindrà accés restringit a les taules i vistes (només en mode lectura per fer consultes). Per qualsevol modificació a les dades del nostre sistema (insercions, modificacions i eliminacions), s'haurà d'accedir a les dades mitjançant una interfície que donarem també amb el nostre sistema, basada en procediments emmagatzemats. Haurem de fer tots els procediments que considerem necessaris per poder aconseguir totes les funcions que hagi de donar el sistema a l'aplicatiu que hagi de treballar contra nosaltres.

Aquest sistema d'accés a les nostres dades aporta tres grans beneficis :

> 1 - Integritat de les dades : Només els nostres procediments podran modificar les dades, per tant, podrem assegurar la integritat de les dades, definint uns procediments que assegurin que no es viola cap regla d'integritat que haguem fixat nosaltres en el disseny de la base de dades.
>
> 2 – Facilitat d'ús del nostre sistema : Els programadors que treballin amb el nostre sistema des de les seves aplicacions no hauran de conèixer amb exactitud totes les regles d'integritat que s'han de complir per què les dades siguin consistents. Només hauran de conèixer la signatura dels procediments de la interfície que els hi oferirem.
>
> 3 – Facilitat de tenir un sistema de control d'activitat o *log* : Com que qualsevol canvi a les dades es produirà des dels nostres procediments, serà fàcil portar un control d'activitat ('*log*') de tot els canvis/insercions/modificacions que s'han fet a la base de dades.

El desenvolupament d'aquest sistema es farà amb el Sistema Gestor de Bases de Dades (SGBD) Oracle 9, un SGBD complert, que permetrà totes les funcionalitats que necessitem (és transaccional, permet creació de triggers, índexs, stored procedures amb llenguatge PL/SQL), etc. Oracle és un entorn robust i amb gran quota de mercat actualment.

2.2 Estudi previ

A continuació s'entrega l'estudi previ que es va fer per determinar les diferents característiques que tindrà el nostre sistema.

2.2.1 Disseny de la base de dades

El primer pas serà la realització del diagrama Entitat-Relació, identificant i descrivint els atributs de cada entitat. Anotarem juntament amb l'esquema totes les restriccions (camps que només puguin prendre certs valors, ...)
Un cop haguem dissenyat el diagrama E/R passarem al disseny equivalent lògic-relacional per poder crear les taules en el nostre sistema.

Aquesta és la informació que emmagatzemarem :

- Metges
  Haurem d'emmagatzemar les dades dels diferents metges que treballin al centre.

- Calendari metges
  S'haurà de gestionar el calendari i l'horari de treball dels metges, tenint en compte que l'horari i els dies de vacances de cadascun dels metges pot ser independent dels calendaris de la resta de metges. Aquesta informació serà necessària de cara a poder assignar hores de visita als pacients.

- Pacients
  Haurem de guardar informació sobre els nostres pacients per poder fer un seguiment dels seus expedients.

- Al·lèrgies
  Part de la informació que es guardarà sobre els pacients serà les al·lèrgies que té (si és que en té), de cara a saber quins medicaments no són compatibles pels seus tractaments.

- Visites
  Emmagatzemarem tota la informació sobre la visita d'un pacient amb un metge, anotant les notes que hagi volgut fer el metge en aquella visita, el llistat de medicaments receptats si és que n'hi ha, i el diagnòstic, o els diagnòstics, si és que en la visita s'ha detectat més d'un cas.

- Medicaments
  Es codificaran els medicaments i s'emmagatzemaran, de manera que es pugui disposar amb rapidesa de la informació més important del medicament (com per exemple, amb quins medicaments és incompatible, o si no és adequat per pacients amb una determinada al·lèrgia), i que es pugui guardar a la fitxa d'una visita d'un pacient, quins medicaments se li han receptat.

- Diagnòstics
  Els diagnòstics es poden tenir codificats, de manera que es pugui guardar informació sobre cada diagnòstic, i es puguin fer cerques d'altres pacients amb el mateix diagnòstic, per si es considera important fer alguna consulta, o per poder treure estadístiques dels diagnòstics més freqüents en segons quina època del any, o en pacients d'una determinada condició, ...

- Usuaris del sistema
  Emmagatzemarem informació sobre els usuaris que poden accedir a l'aplicació. Nosaltres no farem control d'autentificació en el nostra sistema, però el programa final si que podria fer-ho, i hauria de fer servir la nostre informació sobre usuaris del sistema. D'aquesta manera, podrem emmagatzemar en el registre de l'aplicació quin usuari ha executat cadascuna de les accions que s'han executat a la base de dades, per facilitar possibles seguiments.

- Paràmetres de configuració del sistema

  Emmagatzemarem a una taula possibles variables de configuració del sistema que ens hagin de fer falta per fer un desenvolupament flexible (per emmagatzemar valors per defecte de determinades accions, que es puguin modificar sense haver de retocar codi).

2.2.2 Controls d'integritat

En el disseny de la base de dades haurem indicat tots els controls d'integritat que volem respectar. Tant els que es puguin controlar mitjançant claus foranes (per exemple, que no deixi esborrar de les taules un medicament que estigui receptat a un pacient i ens quedin les dades inconsistents), com els que no es poden controlar amb aquests mecanismes (com per exemple, que el sistema no permeti receptar un medicament amb penicil·lina a un pacient al·lèrgic a aquest component).

2.2.3 Tractament d'esborrat de dades

Haurem de definir quines dades es poden esborrar i quines dades no, i com gestionarem l'esborrat de dades que no s'han d'esborrar. Per exemple : Si es vol esborrar un medicament perquè es deixa de fabricar. Si aquest medicament ha estat receptat a algun pacient, no podem permetre que s'esborri el registre del medicament (les dades serien inconsistents ja que a la visita s'indicaria que es va receptar un medicament amb un codi que no existeix).

Les dades que s'hagin introduït i no estiguin relacionades amb cap altre registre (per exemple, un medicament que mai s'ha receptat), si que es podran eliminar (quedarà constància a la taula de registre, ó '*log*'). Les dades que no es puguin eliminar pel motiu que hem explicat, s'hauran de marcar com "donades de baixa", però sense esborrar el seu registre, ja que està relacionat amb altres dades.

Seguint amb l'exemple dels medicaments, si hem donat de baixa un medicament, no es podrà receptar a un pacient.

2.2.4 Tractament d'excepcions

Tots els errors que pugui donar l'execució dels nostres procediments han de ser errors tractats mitjançant el nostre control d'errors, de manera que puguem donar un missatge d'error controlat i personalitzat, i puguem deixar les dades en un estat correcte.

El primer que farem en un procediment serà comprovar que els paràmetres que ens està passant l'aplicació compleixen certes normes (si ens indiquen un codi de pacient per paràmetre, comprovarem que aquest pacient existeix, per exemple). Farem totes les comprovacions que creiem que poden donar lloc a un error al sistema (per duplicat de claus primàries, per eliminar claus foranes, etc) o que poden donar un error a les nostres dades (s'ha receptat un medicament a un pacient que no pot prendre'l perquè és al·lèrgic a ell).

Si les comprovacions ens indiquen que l'operació no es pot dur a terme, no farem cap canvi a les dades de la nostre base de dades, introduirem un missatge al registre d'activitat i retornarem un missatge d'error a l'aplicació.

Si les comprovacions en indiquen que la operació es pot dur a terme, ho farem. Si tot i així es produís un error (per exemple, s'ha exhaurit l'espai al disc dur), haurem de capturar l'excepció generada, cancel·lar qualsevol modificació que haguem fet ja a la base de dades des d'aquell procediment (treballarem amb transaccions), indicarem un missatge d'error al client i registrarem l'error a la taula de registre. En aquest cas, indicarem al missatge d'error de la taula de registre la traça al procediment, és a dir : Al procediment marcarem punts de control (numerats, per exemple), i indicarem en quin

punt de control ha fallat l'execució, per poder estudiar el cas que ha fet que el sistema fallés, i en quin punt exacte ha fallat.

2.2.5 Procediments emmagatzemats

Els procediments PL/SQL que implementarem hauran de seguir tots unes característiques comuns (algunes d'elles les hem comentat ja en aquest document) :

- Comprovació dels paràmetres

- Es tractaran les excepcions no previstes

- Com a mínim hi haurà un paràmetre de sortida a tots els procediments anomenat RSP, de tipus *string* que indicarà si l'execució ha finalitzat amb èxit (valor 'OK') o si ha fracassat (valor 'ERROR+TIPUS D'ERROR)

- Tot el codi anirà indentat i comentat convenientment

- S'emmagatzemaran a la taula de *log* totes les crides als procediments (crides exitoses o que han provocat fallida), i indicarem els paràmetres d'entrada del procediment i els paràmetres de sortida.

2.2.6 Sistema de registre d'activitat (o 'log')

Com ja hem anat introduint en aquest document, el nostre sistema tindrà una taula de registre d'activitat, on introduirem un registre nou cada vegada que es faci una crida a un dels nostres procediments que han de modificar les dades.

Aquest sistema de log ens ajudarà a saber totes les activitats que han executat els usuaris de l'aplicació juntament amb els errors que s'hagin pogut produir.

A cada registre s'introduirà com a mínim aquesta informació :

- data i hora
- identificador del procediment (nom o codi)
- paràmetres d'entrada
- paràmetres de sortida
- codi d'error (null si el procediment s'ha executat correctament)
- indicador de traça (punt on s'ha produït l'error dins el procediment)
- usuari que ha executat el procediment

2.2.7 Funcions que haurem de donar a la nostra interfície

Com es pot veure més endavant a la planificació (veure detall de tasques a la Organització del Projecte), estudiarem amb detall els procediments que ha de tenir la nostra interfície, els llistarem, detallarem els paràmetres d'entrada i sortida, i definirem el comportament que han de tenir, però de moment ja podem fer el següent estimació de procediments que segur haurem d'implementar :

- Altes, baixes i modificacions de tots els elements de les taules que tinguem en el nostre sistema.

- Procediments necessaris per a :
  - Gestió de pacients
  - Control d'hores de visita

- Gestió de medicaments
- Manteniment d'historials mèdics

- Procediments per a altres necessitats que es puguin detectar a la fase d'anàlisi / disseny

2.3 Objectius generals

Els objectius generals d'aquest projecte son :

- Realitzar el disseny i la implementació del sistema de base de dades per una aplicació per gestionar les necessitats d'un consultori mèdic.

- Consolidar els coneixements adquirits al llarg dels estudis pel que fa a assignatura de l'àrea de bases de dades.

- Aprendre a treballar amb un sistema gestor de bases de dades amb gran quota de mercat : Oracle

- Aprendre a treballar amb el llenguatge PL/SQL i SQL Dinàmic

- Portar a la pràctica els coneixements adquirits sobre planificació de projectes

2.4 Entregables del projecte

Els entregables finals seran :

- Memòria del treball
- Annexes a la documentació si es consideren necessaris
- Presentació Power Point
- Scripts de creació de la base de dades
- Procediments PL/SQL
- Jocs de proves

Hi haurà altres entregables durant l'execució del projecte, que es podran veure detallats al calendari de treball.

2.5 Anàlisi de riscos

Abans de començar el desenvolupament del projecte es detecten aquests riscos :

- Data d'entrega fixada i inamovible : El projecte s'ha d'entregar el proper 9 de gener de 2006. Qualsevol desviació que es produeixi a la planificació inicial haurà de ser corregida en les posteriors tasques de manera que es pugui mantenir la data fixada d'entrega.

- No es coneix el sistema de bases de dades amb el que es treballarà (Oracle), així que a la planificació s'haurà de tenir en compte un temps d'estudi d'aquesta plataforma. Aquest fet s'haurà de tenir en compte també al fixar el temps que es deixarà per proves i solució d'incidències que es trobin al sistema.

3 Planificació del Projecte

3.1 Relació de tasques

Aquestes son les tasques en les que he dividit el projecte :

3.1.1 Definició del projecte

Amb aquestes tasques ja tindrem definit el que ha de fer el nostre projecte (sense totes les especificacions tècniques encara, això ho tindrem després de la fase d'anàlisi i disseny). Haurem definit en quines tasques volem dividir l'execució del projecte i com s'han planificat per poder-les dur a terme en el temps previst.

- **Estudi enunciat**
  Lectura i comprensió de l'enunciat proporcionat pel consultor de l'assignatura

- **Trobada de presentació**
  Primer contacte amb el "client", on es detallarà més el que es vol d'aquest projecte, sobre la manera de treballar i sobre els dubtes que ens hagin pogut sortir de la lectura de l'enunciat

- **Pla de treball**
  Elaboració d'aquest document, en el que s'especifica com es durà a terme l'execució del projecte, i on es defineixen les bases del que ha de ser el nostre projecte

- **Correcció del pla de treball**
  Pla de treball amb les correccions que ens ha demanat que hi fem el consultor del projecte

3.1.2 Preparació de l'entorn de treball

Aquesta tasca es subdivideix en dos tasques, i l'objectiu que assolirem quan l'haguem finalitzat és el de tenir la instal·lació de l'entorn necessari pel desenvolupament del projecte al nostre ordinador

- **Instal·lació ORACLE**
  Instal·lació dels CD's que ens va proporcionar la UOC amb el material d'aquest quadrimestre segons el manual d'instal·lació d'oracle de l'assignatura SGBD que ens van deixar al Fòrum de l'assignatura

- **Instal·lació entorn programació**
  Instal·lació del programari per fer compilar i provar el codi PL/SQL dels procediments emmagatzemats que implementem. El software recomanat pel consultor és : PL/SQL Developer
  (http://www.allroundautomations.com/plsqldev.html)

3.1.3 Formació

Amb aquestes tasques plasmem el temps d'autoformació necessari que he previst que hauré de dedicar.

- **Formació entorn Oracle**
  Formació per conèixer com funciona el SGBD, de quina manera es creen taules, on s'executen scripts, entorn de configuració dels paràmetres del gestor, etc.

- **Formació PL-SQL**
  Formació específica en el llenguatge de programació que farem servir pels procediments emmagatzemats (llenguatge propi d'Oracle)

- **Formació gestió d'excepcions Oracle**
  Formació sobre com es gestionen les excepcions causades per un intent d'execució d'una ordre que no es pot portar a terme per algun motiu. Hauré d'aprendre com capturar aquests missatges d'error per poder tractar-los i que l'usuari final rebi un error generat per mi.

3.1.4 Anàlisi i disseny

Amb aquestes tasques definirem amb detall totes les dades que emmagatzemarà el nostre sistema i de quina manera ho farà (estructura taules), així com tots els procediments que creiem necessari desenvolupar per donar totes les funcionalitats necessàries. Es definirà l'estructura i el comportament de tot el codi que desenvoluparem a la fase d'implementació.

- **Estudi de les dades que volem tractar**
  Farem un estudi detallat de les dades que ha de guardar el nostre sistema. Aquest estudi s'ha començat en el punt "estudi previ" d'aquest mateix document.

- **Disseny diagrama E-R**
  Creació del diagrama entitat-relació per obtenir l'esquema de les dades del sistema i de la relació entre elles.

- **Disseny lògic-relacional**
  Transformació del diagrama E-R en l'estructura de taules, identificant quins camps son clau primària, clau forana, etc.. (pas previ a la creació dels scripts SQL de creació de taules).

- **Disseny de regles d'integritat de les nostres dades**
  Farem un recull de totes les regles d'integritat que vulguem tenir en les nostres dades, tant les que solucionarem mitjançant eines del SGBD (les que es puguin controlar amb claus foranes, per exemple), com les que haguem de controlar nosaltres amb altres tipus d'eines (per exemple : incompatibilitat entre medicaments receptats a un pacient).

- **Disseny dels triggers / índexs que necessitem**
  S'estudiarà si serà necessari crear índexs o triggers per la gestió eficient de les dades.

- **Llistat de tots els procediments que formaran la interfície**
  Es farà un llistat detallat de tots els procediments que s'han de donar a l'usuari per tal de tenir una interfície complerta que permeti fer totes les operacions que s'hagin de fer al nostre sistema.

- **Disseny dels procediments d'altes/baixes/modif de totes les entitats**
  Els primers procediments que dissenyarem seran els que podríem anomenar "bàsics", que permetran inserir/eliminar/modificar les dades d'una entitat. Indicarem la signatura dels procediments i descriurem el seu funcionament.

- **Disseny de la resta de procediments (funcionalitats)**
  Un cop haguem fet el pas anterior, crearem els procediments que donaran funcions combinades (com per exemple : buscar hora lliure d'un metge per assignar-li una visita). Indicarem la signatura dels procediments i descriurem el seu funcionament.

- **LOG - Especificacions generals del sistema**
  Dissenyarem la taula que haurà d'emmagatzemar el registre d'activitat. Quins camps tindrà, de quin tipus seran, ...

- **LOG - Detall del registre que guardarà cada procediment**
  Indicarem quins valors podrà gravar a la taula de registre cadascun dels procediments que haguem dissenyat.

- **Documentació anàlisi i disseny**
  Revisió i agrupació en un sol document (amb el format que hagi de tenir el document de la memòria del projecte) de la documentació de tots els punts anteriors.

3.1.5 Implementació

Amb aquestes tasques aconseguirem la implementació de tot el sistema dissenyat en el punt anterior. Hem subdividit aquesta tasca en les tasques següents :

- **Scripts creació de taules**
  Crearem els fitxers amb les comandes SQL per tal de crear les taules tal com les haguem dissenyat en el procés de disseny.

- **Scripts de creació dels triggers**
  Crearem els fitxers amb les comandes SQL per tal de crear els triggers que haguem de posar a les taules si és que s'ha considerat necessari a l'etapa de disseny.

- **Scripts de creació dels índexs**
  Crearem els fitxers amb les comandes SQL per tal de crear els índexs que s'hagin de crear als camps de les nostres taules.

- **Implementació dels procediments d'altes/baixes/modificacions de cada entitat**
  Implementarem els procediments que hem descrit en l'etapa de disseny. En primer lloc implementarem els procediments bàsics per fer altes/baixes/modificacions de totes les entitats.

- **Implementació de la resta de procediments (funcionalitats)**
  Implementarem els procediments que hem descrit en l'etapa de disseny. Un cop implementats els procediments bàsics, crearem les funcionalitats, que podran fer servir els procediments bàsics quan sigui necessari.

- **Revisió del codi / comentaris**
  Abans d'entregar el codi es revisarà que tot segueixi un estil igual, que els noms dels procediments i de les variables segueixin tots un mateix estil, que el codi estigui convenientment indentat, i que hi hagi tots els comentaris que hi hagi d'haver al codi perquè aquest sigui el més entenedor possible.

- **Documentació implementació**
  Es finalitzarà la tasca d'implementació quan s'hagi finalitzat aquesta última subtasca : Revisió i agrupació en un sol document (amb el format que hagi de tenir el document de la memòria del projecte) de la documentació de tots els punts anteriors.

3.1.6 Proves del sistema

Temps de marge per preparar jocs de proves i testejar el sistema en tot el seu conjunt.

3.1.7 Finalització documentació

Unió de tota la documentació que hem fet en els diversos entregables, revisions, finalització de la memòria i preparació dels annexos si es creu convenient quan arribi el moment.

3.1.8 Preparació presentació PPT

Preparació de la presentació virtual amb Power Point.

3.1.9 Debat TFC amb el tribunal

Setmana de debat amb el tribunal del projecte, foro de debat amb preguntes i respostes.

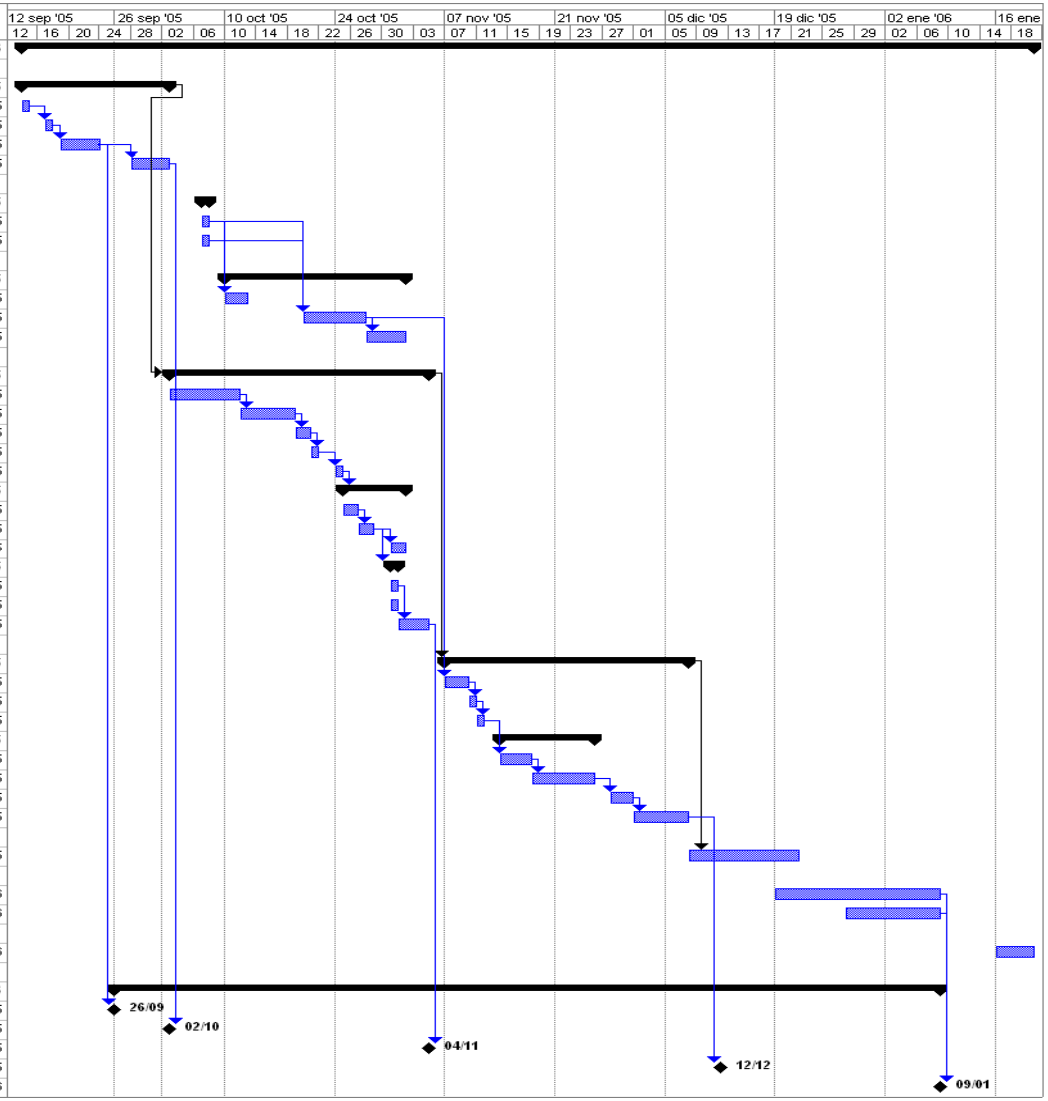
3.2 Fites

DATA	Entrega	Contingut
26/09/05	Pla de Treball	Planificació del projecte
01/10/05	Pla de Treball	Planificació del projecte amb correccions proposades pel consultor
02/11/05	PAC2	Anàlisi i disseny
12/12/05	PAC3	Implementació (documentació + codi)
09/01/06	Lliurament TFC	Memòria TFC + presentació PPT

3.3 Calendari de Treball

Aquest és el diagrama de Gantt de planificació del projecte

Id	Nombre de tarea	Duración	Predece	Comienzo	Fin
1	**TFC - Consultori Mèdic**	**94 días**		**mié 14/09/05**	**vie 20/01/06**
2					
3	**Definició del Projecte**	**14 días**		**mié 14/09/05**	**dom 02/10/05**
4	Estudi enunciat	1 día		mié 14/09/05	mié 14/09/05
5	Trobada de presentació	1 día	4	sáb 17/09/05	sáb 17/09/05
6	Pla de treball	5 días	5	lun 19/09/05	vie 23/09/05
7	Correccinos del pla de treball	3 días	6	mié 28/09/05	dom 02/10/05
8					
9	**Preparació de l'entorn de treball**	**1 día**		**vie 07/10/05**	**vie 07/10/05**
10	Instal.lació ORACLE	1 día		vie 07/10/05	vie 07/10/05
11	Instal.lació entorn programació	1 día		vie 07/10/05	vie 07/10/05
12					
13	**Formació**	**17 días**		**lun 10/10/05**	**mar 01/11/05**
14	Formació entorn Oracle	3 días	10	lun 10/10/05	mié 12/10/05
15	Formació PL-SQL	6 días	10;11	jue 20/10/05	jue 27/10/05
16	Formació gestió d'excepcions Oracle	3 días	15	vie 28/10/05	mar 01/11/05
17					
18	**Anàlisi i disseny**	**25 días**	**3**	**lun 03/10/05**	**vie 04/11/05**
19	Estudi de les dades que volem tractar	7 días		lun 03/10/05	mar 11/10/05
20	Disseny diagrama E-R	5 días	19	mié 12/10/05	mar 18/10/05
21	Disseny lògic-relacional	2 días	20	mié 19/10/05	jue 20/10/05
22	Disseny de regles d'integritat de les nostres dades	1 día	21	vie 21/10/05	vie 21/10/05
23	Disseny dels triggers / indexos que necessitem	1 día	22	lun 24/10/05	lun 24/10/05
24	**Disseny interfície (PROCEDIMENTS)**	**6 días**	**23**	**mar 25/10/05**	**mar 01/11/05**
25	Llistat de tots els procediments que formaran la interfici	2 días		mar 25/10/05	mié 26/10/05
26	Disseny dels procediments d'altes/baixes/modif de totes	2 días	25	jue 27/10/05	vie 28/10/05
27	Disseny de la resta de procediments (funcionalitats)	2 días	26	lun 31/10/05	mar 01/11/05
28	**Disseny sistema LOG**	**1 día**	**26**	**lun 31/10/05**	**lun 31/10/05**
29	Especificacions generals del sistema	1 día		lun 31/10/05	lun 31/10/05
30	Detall del registre que guardara cada procediment	1 día		lun 31/10/05	lun 31/10/05
31	Documentació anàlisi i disseny	4 días	29	mar 01/11/05	vie 04/11/05
32					
33	**Implementació**	**23 días**	**18**	**lun 07/11/05**	**mié 07/12/05**
34	Scripts creació taules	3 días	15	lun 07/11/05	mié 09/11/05
35	Scripts de creació dels triggers	1 día	34	jue 10/11/05	jue 10/11/05
36	Scripts de creació dels indexos	1 día	35	vie 11/11/05	vie 11/11/05
37	**Implementació dels procediments**	**10 días**		**lun 14/11/05**	**vie 25/11/05**
38	Implementació dels procediments d'altes/baixes/modif de	4 días	36	lun 14/11/05	jue 17/11/05
39	Implementació de la resta de procediments (funcionalita	6 días	38	vie 18/11/05	vie 25/11/05
40	Revissió del codi / comentaris	3 días	39	lun 28/11/05	mié 30/11/05
41	Documentació implementació	5 días	40	jue 01/12/05	mié 07/12/05
42					
43	Proves del sistema	10 días	33	jue 08/12/05	mié 21/12/05
44					
45	Finalització documentació	15 días		lun 19/12/05	dom 08/01/06
46	Preparació presentació PPT	8 días		mié 28/12/05	dom 08/01/06
47					
48	Debat TFC amb el tribunal	5 días		lun 16/01/06	vie 20/01/06
49					
50	**Fites i entregables**	**75 días**		**lun 26/09/05**	**lun 09/01/06**
51	PAC1 - Pla de projecte	0 días	6	lun 26/09/05	lun 26/09/05
52	PAC1 - Pla de projecte corregit	0 días	7	dom 02/10/05	dom 02/10/05
53	PAC2	0 días	31	vie 04/11/05	vie 04/11/05
54	PAC3	0 días	41	lun 12/12/05	lun 12/12/05
55	Entrega TFC	0 días	45;46	lun 09/01/06	lun 09/01/06



4 Anàlisis i Disseny

4.1 Estudi de les dades que tractarem al nostre sistema

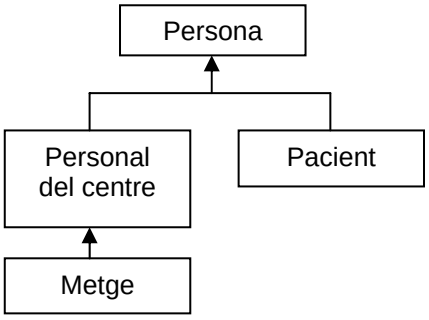
Segons les indicacions fetes al Pla de Projecte, a continuació estudiarem les dades que emmagatzemarem al nostre sistema, de cara a començar el disseny de la Base de Dades que necessitem.

4.1.1 Dades sobre persones

En primer lloc, pensarem en les persones que intervindran al sistema : Pacients, Metges, i Personal del Centre. Aquest últim grup de Personal del Centre seran els usuaris que han de tenir accés a les dades mitjançant l'aplicació. Per exemple : el personal de la Secretaria del centre, per donar hores de visita als pacients, ajudants dels metges, etc. És interessant tenir informació sobre aquest últim grup de cara a poder emmagatzemar al nostre sistema quin usuari ha fet cadascuna de les accions que provoquen una modificació de dades.

Tots tres grups (Pacients, Metges, Personal del Centre) tindran moltes dades en comú (DNI, Nom, Cognoms, Adreça, Telèfon, ..., ...), i després, cada col·lectiu tindrà unes dades específiques (per exemple, per als metges haurem de guardar el número de col·legiat, pels usuaris de l'aplicació haurem de guardar les dades sobre la última connexió que van fer al sistema, ...).

Per tant, ja tenim :



Com es veu a l'esquema, els metges, a més de metges, son Personal del Centre (per aquest fet, tindran un usuari definit per treballar al sistema).

Aquest esquema ens permetrà que un metge pugui ser també tractat com a pacient dintre del nostre sistema, o que una persona que treballi al nostre centre també pugui ser visitada per algun dels nostres metges.

Sobre els metges, haurem d'emmagatzemar dades com el seu número de col·legiat, la seva especialitat, la data d'alta com a metge al nostre centre, la data de baixa si toca, etc. Cada metge tindrà capacitat per treballar en una especialitat (Traumatologia, Dermatologia, Otorinolaringologia, Oculista, Dentista, etc.).

4.1.2 Dades sobre els horaris dels metges i del centre

Haurem d'emmagatzemar la informació necessària per saber si un dia a una determinada hora podrem trobar a un metge determinat (per programar una visita, per exemple). Per aconseguir

això, haurem d'emmagatzemar : l'horari habitual de visites de cada metge, els dies de vacances del nostre centre (en els quals no treballarà cap metge), els dies de vacances de cada metge, i possibles excepcions d'horaris (per exemple, per poder reflectir que un metge treballarà un dia concret fora del seu horari perquè està de guàrdia o perquè està substituint un company).

Hem de tenir en compte que : Els horaris de cada metge son independents i a més, un metge, en un mateix dia pot tenir la seva jornada dividida en vàries franges horàries (Per exemple : els dilluns de 9 a 12, de 14 a 15 i de 17 a 21h, els dimarts altres franges, ...). La forma més clara d'explicar com s'emmagatzemarà és donant un exemple de com serà la taula que guardi aquesta informació :

Taula per guardar horari normal de visites dels metges :

Metge	DiaSet	H.Inici	H.Fi	Data Alta	Data Baixa
Id1	Dilluns	9.00	14.00	01/01/2005	
Id1	Dilluns	17.30	19.00	01/03/2005	
Id1	Dimarts	10.00	15.00	01/01/2005	
Id2	Dilluns	9.30	13.30	01/01/2005	
Id1	Divendres	12.00	16.00	01/01/2005	01/03/2005

(*les dades son només una mostra.. per guardar els dies es codificarà segons el número de dia de la setmana, ..., ...*)

On es pot observar que el metge Id1 actualment treballa :
Dilluns de 9.00 a 14.00 i Dimarts de 10.00 a 15.00 (el dia 1 de Març va deixar de treballar els divendres per la tarda)

I el metge 2 treballa els Dilluns de 9.30 a 13.30

També hem de tenir en compte que en determinades dates els metges poden estar de vacances, de baixa, o per algun altre motiu, podrien haver canviat el seu horari. També hem de contemplar la possibilitat de que el centre tanqui durant alguns dies de vacances, i cap metge treballi aquells dies (per Nadal per exemple).

Per contemplar aquestes últimes excepcions tindrem una altre entitat, que guardarà totes les excepcions, tant si son perquè els metges fan hores de més, com hores de menys.

Aquesta informació la podem guardar amb una taula d'aquest estil :

Metge	DiaSet	H.Inici	H.Fi	DataInici	DataFi
NULL	NULL	NULL	NULL	24/12/2005	24/12/2005
Id2	NULL	NULL	NULL	01/01/2005	15/01/2005
Id1	Dilluns	NULL	NULL	01/01/2005	01/01/2005
Id1	Dimarts	9.00	12.00	01/03/2005	31/03/2005

Com veiem, el fet de deixar en blanc (Valor NULL) alguns valors té sentits diferents, en funció del camp que es deixi amb aquest valor, per exemple :

El camp Metge amb valor NULL significarà que és un canvi que afecta a TOTS els metges (als exemples de la taula, el primer registre indica que el dia 24/12/2005 tots els metges tenen festa).

El camp dia de la setmana amb valor NULL indica que el canvi afecta a tots els dies de la setmana (als exemples de la taula, el 3er registre indica que des de el dia 01/01/2005 fins el 15/01/2005 el metge Id2 no treballarà.

Els camps Hora Inici i Hora Fi a NULL indicaran que aquell dia el metge indicat no treballa (registre 3 de la taula d'exemple).

L'últim registre de la taula ens està indicant que el metge Id1 ha canviat el seu horari durant els dimarts del mes de març

La taula d'excepcions sempre substituirà d'informació de la taula d'horaris generals, justament és aquesta la seva finalitat, marcar les excepcions d'horaris que no segueixen la norma general. Per tant, sempre que hi hagi una excepció indicada a la taula d'excepcions, s'ignorarà el valor de la taula d'horaris normals per aquella data que s'està mirant.

4.1.3 Casos, visites i documentació

Cada pacient tindrà un expedient, on s'emmagatzemarà tota la informació d'aquell pacient al nostre centre i de totes les visites que faci. La manera de classificar la informació serà mitjançant casos. Un expedient pot estar format per molts casos, i cada cas pot estar format per una o vàries visites.

Al parlar d'un cas, ens referirem a un problema d'un usuari que s'ha tractat o que s'està tractant (per exemple : una grip, o un trencament d'un braç, o un problema d'artrosis). Per tractar un cas és possible que es necessiti una o vàries visites. I en una mateixa visita és possible que un mateix metge tracti dos casos del mateix pacient (sempre que el metge tingui competència per treballar en l'especialitat que requereixi aquell cas).

Un cas tindrà un camp que indicarà el diagnòstic que s'ha fet per a aquell cas.. Aquest diagnòstic podria variar entre una visita i una altra, si el metge creu que el diagnòstic inicial no és del tot precís i considera canviar-lo. També es podrà deixar el diagnòstic sense especificar en cas de no poder-lo determinar en les primeres visites, o per quan es programi la primera visita a la secretaria del centre (la persona que atengui al pacient per telèfon no identificarà el diagnòstic, només indicarà el metge pel qual s'està demanant hora (de l'especialitat que toqui) i deixarà el diagnòstic en blanc). Els diagnòstics estaran codificats, de manera que es puguin fer estadístiques, i es pugui comprovar, per el mateix diagnòstic, quins usuaris l'han tingut i així puguem consultar, per exemple, quin tractament els varem donar.

Cada cas tindrà assignat un metge de l'especialitat que es necessiti per aquell pacient, que serà l'encarregat de fer el seguiment del cas. Puntualment, un altre metge de la mateixa especialitat podrà fer una visita per tractar un cas d'un pacient encara que no sigui el metge encarregat d'aquell cas. Per suposat, sempre es deixarà constància de quin metge ha fet cadascuna de les visites que formen un cas.

Per als pacients, a més de l'expedient de casos i visites, emmagatzemarem arxius que anomenarem documentació (que seran totes les proves que s'han fet a l'usuari : analítiques, radiografies, escàners, i qualsevol altre prova que se li hagi fet a l'usuari). A la documentació, s'indicarà a quin cas corresponen, però per accedir a ells no serà necessari entrar a aquell cas concret. Accedint a un usuari podrem llistar tota la documentació que tenim d'ell, de cara a poder aprofitar documentació entre casos (si a un pacient ja se li ha fet una determinada analítica, pot ser útil per altres casos que tingui aquell pacient). Per guardar aquesta informació, emmagatzemarem fitxers digitals (fotografies per les radiografies, documents digitals, o escanejats per les analítiques, ... ,...).

A cada visita, si es recepta algun medicament pel pacient, s'indicarà el medicament receptat, la data quan ha de començar a prendre'l, la data en la que l'ha de deixar de prendre, d'interval entre dosis i altres indicacions.

4.1.4 Medicaments i al·lèrgies

Hem de pensar que cal poder guardar a la fitxa del pacient si pren algun tipus de medicament habitualment (per exemple, pastilles per regular la tensió arterial, medicament que potser no correspon a cap cas que haguem tractat a la nostre clínica). També guardarem, a la fitxa del

pacient, si té al·lèrgies a algun medicament (per evitar receptar-li medicaments que no pugui prendre).

També guardarem codificats els medicaments que receptem, indicant incompatibilitats entre ells (per no donar una medicació a un pacient que actualment està prenent una altre incompatible amb aquesta) i indicant incompatibilitats amb determinades al·lèrgies, per saber si un medicament no es pot subministrar a alguns pacients (per exemple, medicaments amb penicil·lina).

4.1.5 Altres dades del sistema

Tal com es va indicar a l'enunciat, el nostre sistema de Base de Dades inclourà una taula a la qual s'enregistrarà tota l'activitat que ha tingut la nostre BD, de cara a :

- Poder fer un seguiment en cas d'error del programa
- Poder comprovar quin usuari i en quin moment ha fet una determinada inserció/modificació/supressió de dades
- Poder fer una reconstrucció de la base de dades

En aquesta taula, per cada acció que realitzi el nostre programa, guardarem un registre amb la següent informació : data, hora, usuari que efectua l'operació, paràmetres d'entrada de la funció cridada, etc.

El sistema de log es detalla en el punt 4.5 d'aquesta memòria.

4.2 Disseny conceptual (model E-R)

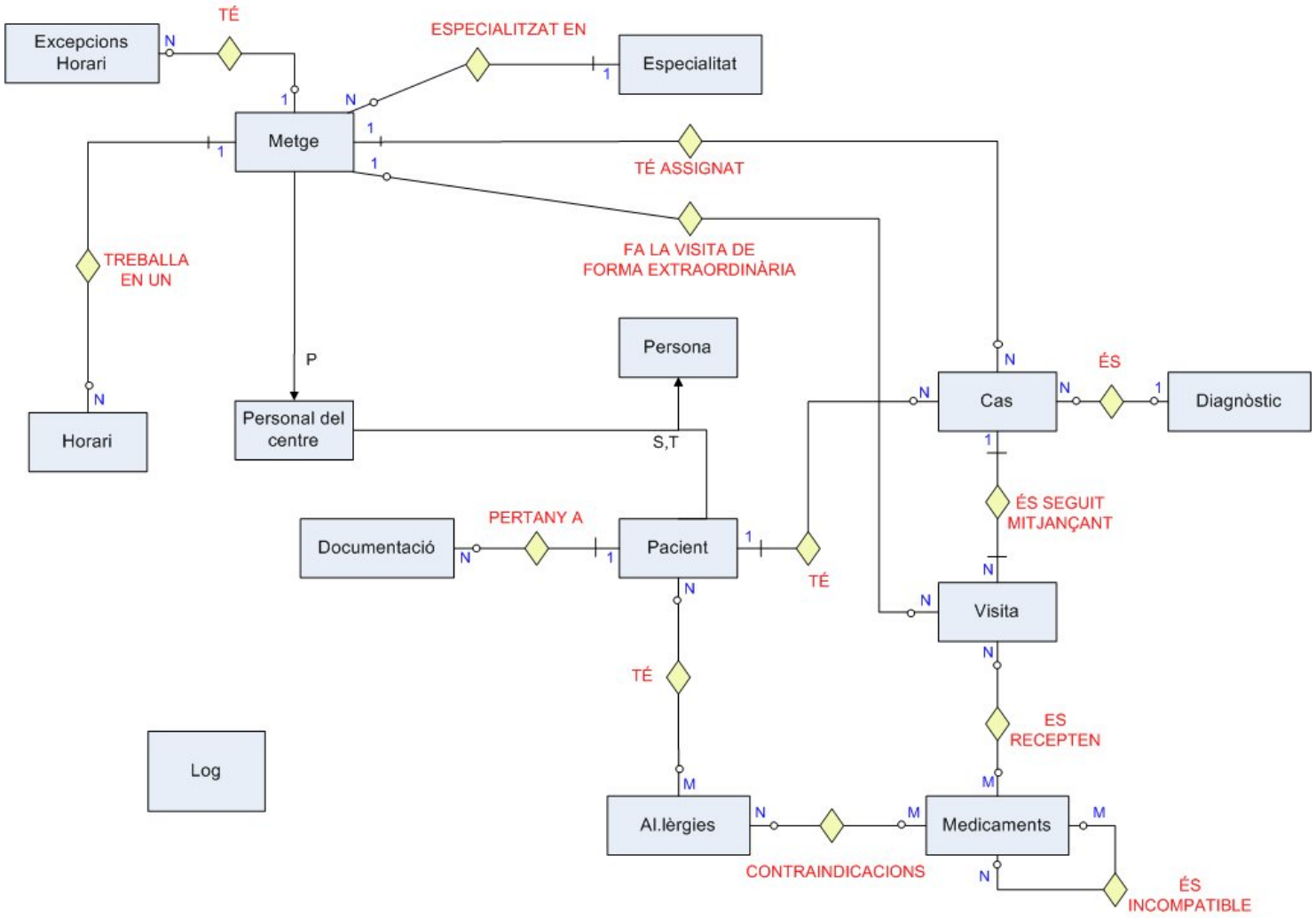
4.2.1 Identificació d'entitats del nostre sistema

Segons la descripció de les dades feta al punt anterior, podem identificar les següents entitats :

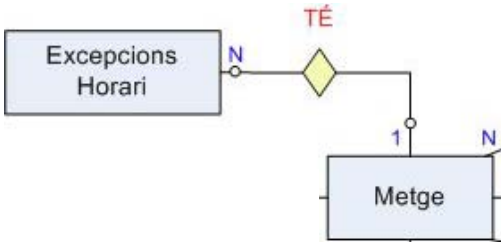
Sobre cada entitat indico la descripció i, a mode informatiu, quins camps tindrà aquesta entitat, només per ampliar l'informació de la descripció i donar una idea més clara sobre l'entitat. En aquest mateix document, en el punt 3.1 (Definició de les taules i camps) s'indicarà amb detall els camps definitius de cada entitat, el seu tipus, etc.

Persona	Dades bàsiques que guardarem al sistema de qualsevol persona (nom, cognoms, DNI, data naixement, adreça, ...)
Pacient	Dades sobre els pacients del centre no contemplades a l'entitat persones
Personal del centre	Dades sobre l'afiliació de les persones al centre (data alta, data baixa, nom d'usuari i contrasenya al nostre sistema, ...)
Metge	Dades sobre els metges que visiten o que han visitat al nostre centre, indicant dades com el número de col·legiat. Cada metge, a més, indicarà quin temps mínim vol per cada visita que se li programi.
Cas	Dades sobre problemes mèdics que han tingut els nostres pacients. Es farà el seguiment d'un cas d'un pacient mitjançant tantes visites com siguin necessàries. Un cas tindrà un metge associat a aquell cas (i a partir d'aquest metge, es podrà esbrinar l'especialitat en la que es
Visita	Dades sobre una visita d'un dia concret d'un pacient a un metge. Una visita sempre formarà part d'un cas. S'indicarà el dia de la visita, la hora, el pacient, el metge, els comentaris que anoti el metge, els medicaments receptats, ... La visita també es programarà introduint-la aquí, mitjançant un camp s'indicarà si la visita està programada, cancel·lada, aplaçada o ja realitzada.
Documentació	Emmagatzemament de tot tipus de proves que es facin a l'usuari (radiografies, escàners, analítiques, etc) en format digital.
Horari	Horari de visites que tenen els metges habitualment
Excepcions Horari	Aquesta entitat ens servirà per indicar quines excepcions hem de tenir en compte amb els horaris dels metges, per exemple : vacances, períodes on poden estar de baixa els metges, o també guàrdies que podem programar perquè metges treballin fora del seu horari habitual, o torns que puguin cobrir uns metges a uns altres, ...
Medicaments	Informació sobre els medicaments (nom, posologia, contraindicacions, incompatibilitat amb altres medicaments, incompatibilitat amb al·lèrgies de pacients, ...)
Al·lèrgies	Codificació d'al·lèrgies conegudes
Especialitat	Codificació de les diferents especialitats que tenen els nostres metges
Diagnòstic	Codificació dels diferents diagnòstics que hem tractat al nostre centre
Log	Entitat pel registre de qualsevol tipus d'activitat a les nostres taules de dades

4.2.2 Diagrama Entitat-Relació

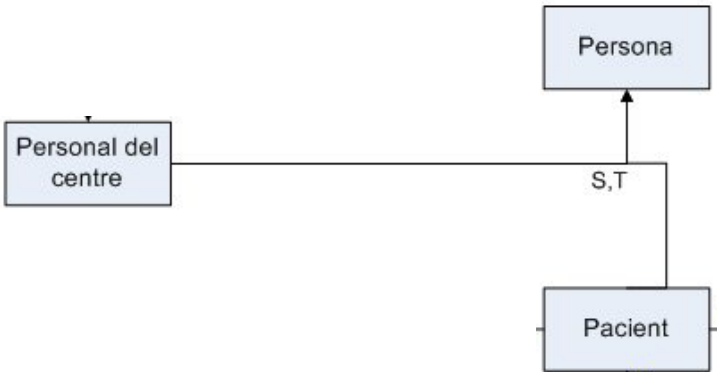


4.2.3 Comentaris al diagrama E-R :

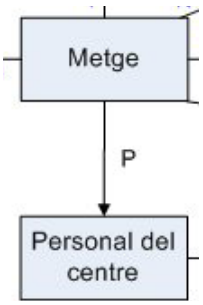


Sobre la relació d'excepcions Horari i Metge : Hi ha un cas que no queda del tot clar al diagrama E-R. El funcionament d'aquest sistema d'excepcions està explicat en aquest mateix document, al punt 1.2 (Dades sobre els horaris dels metges i del centre), tot i així fem el següent aclariment : Es una relació 1:N.. però es permet l'opció de marcar un registre on metge sigui NULL (per això el cercle a l'extrem de l'entitat Metge de la relació). Aquest cas (NULL) però, vol dir precisament que està afectant a TOTS els metges. (Sense aquesta explicació, només amb el diagrama E-R es pot entendre que hi poden haver Excepcions d'horari que no afectin a cap metge, la qual cosa seria un concepte erroni).

Els dos casos d'especialització del diagrama :



Especialització S,T : Encavalcada (S) ja que una mateixa ocurrència podria aparèixer a les dos entitats (una mateixa persona pot ser Pacient i Personal del centre al mateix temps). I Total (T) : Totes les persones apareixen a alguna de les dos subentitats (no tindrem Persones que no siguin ni Pacients ni Personal del Centre).



En canvi, l'especialització de Personal del Centre a Metge es tracta d'una especialització Parcial (P), ja que no tot el personal del centre té perquè ser metge.

4.3 Disseny lògic (model relacional)

Un cop definit el model conceptual (estudi de les entitats i diagrama E-R), estem en condicions de poder passar al model relacional

4.3.1 Definició de les taules i camps

TAULA Persones

Descripció : Dades comuns a totes les persones que tindrem emmagatzemades al nostre sistema, metges, pacients i personal del centre inclosos.

- ***dni*** : [Clau Primària] ; Camp text 8 dígits; DNI en format text de 8 dígits (només acceptarem valors numèrics en aquest camp). En cas de persones amb passaport i sense DNI, farem una transformació del seu número de passaport al format demanat (substituint lletres per zeros i omplint amb zeros per l'esquerra fins arribar a 8 caràcters, o bé truncant a 8 caràcters en cas de ser més llarga la cadena).
- ***lletranif*** : lletra del nif corresponent al dni de l'usuari
- ***passaport*** : Camp text ; Per usuaris amb passaport, farem servir el camp DNI com a clau primària, però ens interessarà emmagatzemar el seu passaport original (per generar qualsevol informe). Per usuaris amb DNI el camp tindrà valor NULL.
- ***nom*** : Camp text ; Nom de la persona
- ***cognoms*** : Camp text ; Cognoms de la persona
- ***data_naixement*** : Camp data ; Camp Obligatori ; Data de naixement de l'usuari.
- ***sexe*** : Camp caràcter ; Camp Obligatori ; H indica Home, D indica Dona
- ***adreca*** : Camp de text ; Adreça de l'usuari
- ***població*** : Camp de text ; Població del domicili de l'usuari
- ***provincia*** : Camp de text ; Província del domicili de l'usuari
- ***codi postal*** : Camp de text ; Codi postal del domicili de l'usuari
- ***foto*** : Camp per adjuntar fitxer ; Camp per emmagatzemar una fotografia de l'usuari
- ***telf_fixe*** : Camp de text ; Telèfon fixe de l'usuari
- ***telf_mòbil*** : Camp de text ; Telèfon mòbil de l'usuari
- ***fax*** : Camp de text; Fax de l'usuari

TAULA Pacients

Descripció : Dades que emmagatzemarem dels nostres pacients

- ***dni*** : [Clau Primària] ; [Clau Forana, taula Persones]
- ***tipus_asseguranca*** : Camp de text ; Indicarem si l'usuari és usuari de la Seguretat Social o si pertany a alguna mútua privada (indicarem el nom de la mútua)
- ***num_asseguranca*** : Camp de text ; Indicarem el número / codi d'assegurança o del registre de la Seguretat Social
- ***data_alta*** : Camp tipus data ; Camp obligatori. Indiquem data d'alta del pacient al nostre centre
- ***data_baixa*** : Camp tipus data ; Camp que indicarà en quina data un usuari s'ha donat de baixa com a pacient. En cas de tenir valor Null significa que el pacient no s'ha donat de baixa.
- ***motiu_baixa*** : Camp de text ; En cas de donar-se de baixa, indicar el motiu (descontent, canvi població, defunció, ...)

TAULA PersonalCentre

Descripció : Dades addicionals (no incloses a la taula Persones) que emmagatzemarem dels nostres treballadors

- ***dni*** : [Clau Primària] ; [Clau Forana, taula Persones]
- ***compte_bancari*** : Camp de text ; Emmagatzemarem el número de compte bancari del nostre treballador per ingressar la nòmina
- ***username*** : Camp de text ; Nom d'usuari al sistema (*)
- ***password*** : Camp de text ; Contrasenya de l'usuari (*)
- ***data_alta*** : Camp tipus data ; Camp obligatori. Indiquem data d'alta del treballador al nostre centre
- ***data_baixa*** : Camp tipus data ; Camp que indicarà en quina data un treballador s'ha donat de baixa. En cas de tenir valor Null significa que el treballador està en estat actiu.
- ***motiu_baixa*** : Camp de text ; En cas de donar-se de baixa, indicar el motiu (descontent, canvi població, canvi de feina, defunció, ...)

(*) Proporcionem aquests camps a les aplicacions que treballin contra la nostre base de dades, però nosaltres no ens encarregarem d'aquest nivell d'autentificació a les aplicacions. Es recomana que les aplicacions no guardin aquí el valor literal de la contrasenya, sinó que facin servir algun tipus d'encriptació per emmagatzemar-lo.

TAULA Metges

Descripció : Dades addicionals (no incloses entre les taules Persones i PersonalCentre) que emmagatzemarem dels nostres metges

- ***dni*** : [Clau Primària] ; [Clau Forana, taula PersonalCentre]
- ***numero_colegiat*** : Camp de text ; Camp obligatori ; Número de col·legiat del metge
- ***id_especialitat*** : [Clau Forana, taula Especialitats] ; Camp numèric ; Camp obligatori ; Identifica l'especialització del metge
- ***durada_visita*** : Camp numèric ; Camp obligatori ; Durada (en minuts) que vol el metge per les seves consultes (aquest és el valor per defecte que tindran les visites programades per ell)

TAULA Especialitats

Descripció : Especialitats donades d'alta al sistema, que ens serviran per agrupar els nostres metges.

- ***id_especialitat*** : [Clau Primària] ; Camp numèric
- ***nom_especialitat*** : Camp de text ; Camp obligatori ; Descripció de l'especialitat (Medicina General, Òptica, Dermatologia, Traumatologia, etc...)

TAULA Horari

Descripció : Horari de visita general que tenen els nostres metges a la nostre clínica

- ***dni*** : [Clau Primària composta] ; [Clau Forana, taula Metges] ; Dni del metge al que afecta l'horari
- ***dia_setmana*** : [Clau Primària composta] ; Camp numèric ; Codificació del dia de la setmana segons aquests valors : (L = 1 ; M = 2 ; X = 3 ; J = 4 ; V = 5 ; S = 6 ; D = 7).
- ***hora_inici*** : [Clau Primària composta] ; Camp text en format hh:mm ; Indica l'hora en la que es comença a treballar en la franja horària que s'està definint
- ***hora_fi*** : Camp text en format hh:mm ; Camp obligatori. Indica l'hora en la que es termina la franja horària que s'està definint

- ***data_alta*** : [Clau Primària composta] ; Camp tipus data ; Data en la que comença a donar servei aquest metge en aquesta franja
- ***data_baixa*** : Camp tipus data ; Data en la que el metge deixa de treballar aquesta franja horària. Valor Null indica que la franja horària és vigent.

TAULA ExcepcionsHorari

Descripció : Recull totes les excepcions que es produeixen sobre els horaris de visita dels metges, com les vacances, períodes en els que estiguin de baixa, o qualsevol canvi d'horari que facin en una data puntual o durant un període. Aquesta taula no té clau primària (els camps candidats poden ser Nuls). Per millorar el temps d'accés a les dades crearem els índexs necessaris (veure apartat 3.4 - Índexs d'aquest document).

- ***dni*** : [Clau Forana, taula Metges] ; dni del metge al que afecta l'excepció. Valor Null indica que afecta a tots els metges.
- ***dia_setmana*** : Camp numèric ; Codificació del dia de la setmana segons aquests valors : (L = 1 ; M = 2 ; X = 3 ; J = 4 ; V = 5 ; S = 6 ; D = 7). Valor Null indica que l'excepció afecta a tots els dies de la setmana.
- ***hora_inici*** : Camp text en format hh:mm ; Indica l'hora en la que es comença a treballar en la franja horària que s'està definint. Valor Null indica que l'excepció es refereix a tot el dia.
- ***hora_fi*** : Camp text en format hh:mm ; Indica l'hora en la que es termina la franja horària que s'està definint. Valor Null indica que l'excepció es refereix a tot el dia.
- ***data_alta*** : Camp tipus data ; Camp obligatori ; Data en la que comença a donar servei aquest metge en aquesta franja
- ***data_baixa*** : Camp tipus data ; Data en la que el metge deixa de treballar aquesta franja horària. Valor Null indica que la franja horària és vigent.

TAULA Al·lèrgies

Descripció : Dades que emmagatzemarem sobre les al·lèrgies que tenim codificades

- ***id_alergia*** : [Clau Primària] ; Camp numèric ; Codi identificador de l'al·lèrgia
- ***nom_alergia*** : Camp text ; Camp obligatori ; Indica el nom de l'al·lèrgia
- ***descripcio*** : Camp text llarg; Descripció dels símptomes de l'al·lèrgia o d'altres dades d'utilitat pel metge
- ***data_alta*** : Camp tipus data ; Camp obligatori ; Indica la data en la que hem donat d'alta aquesta al·lèrgia en el nostre sistema

TAULA AlergiesPacients

Descripció : Taula per emmagatzemar les possibles relacions entre les taules d'al·lèrgies i de pacients.

- ***dni_pacient*** : [Clau primària composta] [Clau forana, camp dni taula Pacients]
- ***id_alergia*** : [Clau primària composta] [Clau forana, taula Al·lèrgies]
- ***data_deteccio*** : Camp tipus data; Camp obligatori ; Indica la data en la que s'ha introduït aquesta relació al sistema.

TAULA Medicaments

Descripció : Dades que emmagatzemarem sobre medicaments

- ***id_medicament*** : [Clau Primària] ; Camp numèric ; Identificador del medicament
- ***nom*** : Camp de text ; Camp obligatori ; Nom del producte
- ***prospecte*** : Camp text llarg; Descripció detallada del producte, on s'indicarà la posologia, les contraindicacions, i d'altre informació útil pels metges.
- ***data_alta*** : Camp tipus data ; Camp obligatori ; Data en la que introduïm el medicament al nostre sistema per poder començar a receptar-lo
- ***data_baixa*** : Camp tipus data ; Data en la que donem de baixa un medicament del sistema, per si no volem tornar a receptar-lo

TAULA Contraindicacions

Descripció : Relacions entre les taules d'al·lèrgies i Medicaments, que indicaran que un medicament no és vàlid per pacients amb una determinada al·lèrgia.

- ***id_medicament*** : [Clau primària composta] [Clau Forana, taula Medicaments]
- ***id_alergia*** : [Clau primària composta] [Clau Forana, taula Al·lèrgies]
- ***data_deteccio*** : Camp tipus data; Camp obligatori ; Indica la data en la que s'ha introduït aquesta relació al sistema.

TAULA MedicamentsIncompatibles

Descripció : Relacions d'incompatibilitats entre medicaments

- ***id_medicament1*** : [Clau primària composta] [Clau Forana, camp id_medicament taula Medicaments]
- ***id_medicament2***: [Clau primària composta] [Clau Forana, camp id_medicament taula Medicaments]
- ***data_deteccio*** : Camp tipus data; Camp obligatori ; Indica la data en la que s'ha introduït aquesta relació al sistema.

TAULA Casos

Descripció : Dades sobre cada nou problema mèdic que un pacient ve a consultar al nostre centre.

- ***id_cas*** : [Clau Primària] ; Camp numèric ; Identificador únic del cas
- ***dni_pacient*** : [Clau forana, camp dni taula Pacients] ; Camp obligatori
- ***dni_metge*** : [Clau forana, camp dni taula Metges] ; Camp obligatori
- ***id_diagnòstic*** : [Clau forana, taula Diagnòstics]. Es permet el valor Null pel cas en el que no s'hagi trobat encara el diagnòstic
- ***data_inici*** : Camp tipus data ; Camp Obligatori; Data en la que s'obre el cas
- ***data_fi*** : Camp tipus data ; Data en la que es tanca el cas
- ***estat*** : Camp text valors {O,T} (Obert / Tancat) : Camp Obligatori; Indica l'estat del cas (no es fa camp booleà per permetre ampliar a més estats en el futur)
- ***comentari*** : Camp text llarg ; Permet fer un comentari general sobre el cas, independent dels comentaris que afegim associat a cadascuna de les visites que formen el cas

TAULA Visites

Descripció : Dades que anotarem per cada visita que un pacient faci al centre. Tota visita forma part d'un cas

- ***id_visita*** : [Clau Primària] ; Camp numèric ; Identificador únic de la visita
- ***id_cas***: [Clau forana, taula Casos]; Camp obligatori ; Identifica al cas al que fa referència aquesta visita.
- ***dni_metge_extra*** : [Clau forana, camp dni taula Metges] ; Valor Null indica que el metge que atén és el metge associat al cas. Altrament, indica el metge que ha atès al pacient de manera extraordinària.
- ***data_visita_programada*** : Camp tipus data ; Data i hora en la que està programada la visita
- ***durada_programada*** : Camp numèric ; Duració estimada de la visita (en minuts)
- ***data_visita*** : Camp tipus data ; Data i hora en la que es comença la visita
- ***durada*** : Camp numèric ; Duració real que ha tingut la visita (en minuts)
- ***estat_visita*** : Camp caràcter valors {P,A,R} ; Camp Obligatori ; Estat de la visita (Programada;Anul·lada;Realitzada}
- ***preu*** : Camp numèric 2 decimals ; Preu de la visita
- ***comentari*** : Camp text llarg; Comentari que el metge guarda amb els comentaris relatius a la visita.

TAULA Receptes

Descripció : Medicaments receptats durant les visites

- ***id_visita*** : : [Clau Primària composta] [Clau Forana, taula Visites]
- ***id_medicament*** : : [Clau Primària composta] [Clau Forana, taula Medicaments]
- ***indicacions :*** Camp text llarg ; Indicacions que es fan a l'usuari sobre la medicació (posologia, dosis, contraindicacions,...)
- ***data_fi :*** Camp tipus text que indica quan un usuari ha de deixar de prendre la medicació

TAULA Diagnòstics

Descripció : Diagnòstics que tenim emmagatzemats al nostre sistema

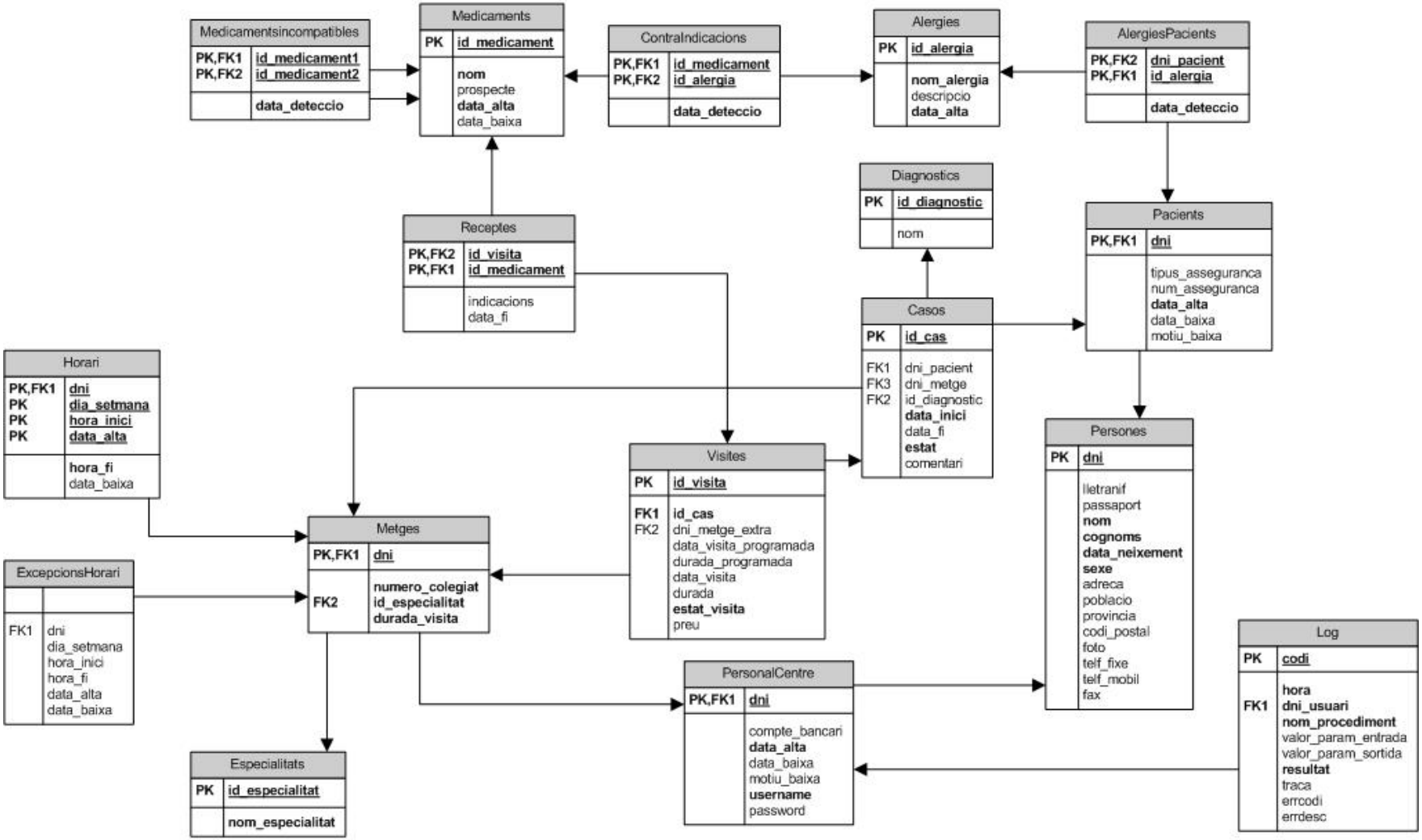
- ***id_diagnòstic*** : [Clau Primària] ; Camp numèric ; Identificador numèric del diagnòstic
- ***nom*** : Camp text ; Nom del diagnòstic

TAULA LOG

Descripció : Taula amb el registre d'activitat del nostre sistema

- ***codi*** : [Clau Primària] ; Camp autonumèric ; Identificador numèric de l'entrada del registre
- ***hora*** : Camp tipus data; Data i hora d'execució; Camp obligatori;
- ***dni_usuari*** : [Clau forana, camp dni taula Personal Centre] ; Camp obligatori ; Usuari que ha fet la crida al procediment
- ***nom_procediment*** : Camp text ; Nom del procediment cridat ; Camp obligatori
- ***valor_param_entrada*** : Camp text llarg ; Camp on emmagatzemarem el valor de tots els paràmetres passats a la funció en el moment d'entrar
- ***valor_param_sortida*** : Camp text llarg ; Valors dels paràmetres de sortida a la sortida de la funció
- ***resultat*** : Camp text ; Camp Obligatori ; Valor RSP retornat per la funció que indica si el resultat de l'execució ha estat correcte o no
- ***traca*** : Camp numèric ; Valor que donarà informació sobre el punt en el que s'ha produït una excepció que ha interromput l'execució del procediment
- ***errcodi*** : Camp numèric : Codi d'error Oracle sobre l'excepció que s'ha produït
- ***errdesc*** : Camp text llarg : Descripció Oracle sobre l'error que s'ha produït

4.3.2 Esquema de les taules de la Base de Dades



4.3.3 Controls d'integritat

Al punt anterior (3.1 Definició de les taules i camps) s'han indicat ja alguns controls d'integritat, en aquells camps que s'han marcat com a Claus Primàries i Claus Foranes.

El fet de que un camp sigui Clau Primària ja suposa que :
No acceptarà valors Nuls
No permetrà duplicats d'aquell camp

El fet de que un camp sigui Clau Forana ja suposa que :
No pot existir si no existeix a la taula a la que fa referència la clau forana

Amb aquests dos mecanismes ja estem assegurant (automàticament, ja que ho controlarà el SGBD) part de la integritat de les nostres dades (el sistema no ens està permetent, per exemple, assignar un metge inexistent a un cas, o eliminar pacient si té un cas obert).

També hem indicat a alguns camps que son obligatoris (és a dir, que no poden prendre el valor Null). L'SGBD també controlarà automàticament que això sigui així.

Hi ha, però, altres restriccions que no queden cobertes amb aquests mecanismes, i que haurem de comprovar amb comandes CHECK al definir les taules al SGBD (per comprovar que un camp es mantingui entre un rang de valors, per exemple), Triggers, o comprovacions que farem al codi dels nostres procediments.

Si s'intenta saltar alguna regla d'integritat (passant paràmetres amb dades errònies a algun dels nostres procediments), haurem de controlar-ho comprovant els paràmetres nosaltres, sempre que sigui possible. I en cas de no haver detectat l'error abans d'executar el procediment, haurem de capturar qualsevol Excepció que ens doni el nostre SGBD per registrar l'error i per informar a l'usuari que no es pot executar el procediment amb els paràmetres que demana.

Les restriccions que haurem de complir per tal de mantenir la coherència en les nostres dades son :

Taula Persones
dni: només caràcters amb valor 0-9
lletra_NIF : només caràcters amb valor A-Z
sexe : només valors M ó H

Taula Pacients
data alta <= data baixa

Taula PersonalCentre
data alta <= data baixa

Taula Metges
durada visita > 0

Taula Horari
dia_setmana : valors limitats entre 1 i 7
hora_inici < hora_fi
data_alta <= data_baixa (a no ser que data_baixa sigui Null)
comprovar que no s'estiguin definint pel mateix metge dos franges horàries que se solapin

Taula ExcepcionsHorari
dia_setmana : valors limitats entre 1 i 7 o Null
hora_inici < hora_fi (a no ser que hora_inici sigui Null)
data_alta <= data_baixa (a no ser que data_baixa sigui Null)

comprovar que no s'estiguin definint pel mateix metge dos franges horàries que se solapin

Taula Medicaments

data_baixa >= data_alta

Taula Casos

data_fi >= data_inici
estat : valors tancats per uns valors

Taula Visites

estat_visita : valors tancats per uns valors

4.3.4 Índexs

Ens interessarà tenir índexs a tots els camps que fem servir per relacionar taules entre si, de manera que totes les consultes relacionades siguin el més ràpides possible.

Després, haurem de valorar, per la resta de camps, quins volem indexar i quins no. Com sabem per les assignatures de Bases de dades I i II : Indexar un camp ens dóna com a principal avantatge el fet de que les consultes per aquell camp seran més ràpides (sobre tot consultes d'ordenació per aquell camp), en canvi, en contra presenta : les insercions / supressions / modificacions son més lentes i ocupa espai addicional en disc. És per això que indexarem només els camps que considerem necessaris perquè es faran cerques en funció d'aquest camp, o es trauran informes ordenats per el camp en qüestió.

Els camps triats per ser indexats son :

Tots els camps que son Clau Primària per defecte ja estan indexats

Indexarem tots els camps clau Forana, ja que seran els que farem servir per relacionar taules quan haguem de fer consultes.

Aquesta és la relació de camps a indexar per aquest motiu :

excepcionshorari.dni
metges.id_especialitat
visites.id_cas
visites.dni_metge_extra
casos.dni_pacient
casos.dni_metge
casos.id_diagnòstic

També indexarem els camps de la taula ExcepcionsHorari que haguéssim triat com a Clau Primària, però que no ho hem pogut fer degut a que es dona l'opció de que continguin valors Nuls :

excepcionshorari.dni (també és clau forana com hem indicat al llistat anterior)
excepcionshorari.dia_setmana
excepcionshorari.hora_inici
excepcionshorari.data_alta

Altres camps que es faran servir bastant en consultes seran aquests :

personalcentre.username
medicaments.nom
diagnostics.nom
persones.nom
persones.cognoms

4.4 Estudi de les funcions que s'han de poder fer amb les dades del sistema

El nostre sistema només donarà accés en mode lectura a les taules de dades per al programari que treballi amb les nostres dades, per tant, hem de preparar tots els procediments emmagatzemats necessaris perquè el programari pugui treballar amb les nostres dades, encarregant-nos nosaltres sempre des de la part de la Base de Dades de mantenir la integritat de les dades (la consistència).

El més important és proporcionar els procediments bàsics amb un codi robust de control d'errors. Les aplicacions podran preparar procediments més complexes partint dels nostres procediments per modificació i inserció i fent les consultes que considerin necessàries a les taules de la base de dades.

S'hauran de preparar tots els procediments necessaris per insercions / modificacions / supressions.

4.4.1 Comportament comú dels procediments emmagatzemats de la interfície

Resumirem una mica l'explicació indicant els criteris comuns que se seguiran a tots els procediments, i després detallarem les particularitats de cadascun d'ells.

4.4.1.1 Paràmetre de sortida RSP

Tots els procediments tindran un paràmetre de sortida anomenat RSP (serà el primer paràmetre de totes les funcions) i ens servirà per identificar si l'execució del procediment ha estat correcta o no. Hi haurà tres possibles grups de valors de retorn :

- **CORRECTE** : El funcionament del procediment ha estat correcte (Codi retorn 'OK')
- **AVÍS (Warning)** : La inserció de dades ha estat correcta, i les dades continuen sent consistents al sistema, però s'ha detectat algun possible problema del que caldria avisar al metge o al pacient afectat (per exemple, si s'introdueix una nova contraindicació que no coneixíem entre una al·lèrgia i un medicament, i temps enrere s'havia medicat amb aquest fàrmac a algun pacient amb aquesta al·lèrgia). (Codi retorn 'WARNING+TipusWarning')
- **ERROR** : El procediment no s'ha pogut executar perquè s'intentava violar alguna condició que posava en perill la consistència de les dades, o perquè el sistema ens ha generat alguna excepció. (Codi retorn 'ERROR + TipusError').

4.4.1.2 Paràmetre d'entrada USER

Els aplicatius que facin crides als procediments hauran d'indicar sempre, com a segon paràmetre, l'usuari que està executant aquella funció. La manera d'identificar a l'usuari serà mitjançant el seu DNI (és la dada que nosaltres tenim com a Clau Primària per identificar usuaris a les nostres taules), i aquest usuari haurà de ser una persona donada d'alta com a Personal del Centre per poder executar el procediment). No identifiquem l'usuari mitjançant el camp username perquè és un camp que es pot modificar, el camp Username l'hem inclòs a les taules només com a suport per les aplicacions, per si el necessiten fer servir, però pel nostre sistema el camp és innecessari.

4.4.1.3 Comprovar que no existeixi una dada abans d'introduir-la

Per a tots els procediments en els que haguem d'afegir/modificar alguna dada, comprovarem abans d'inserir el nou valor, que no existís ja a la base de dades. Per exemple, abans d'introduir un medicament comprovarem que no hi hagi cap ja donat d'alta amb el mateix nom. Si existeix, retornarem codi d'error.

4.4.1.4 Comprovar que existeixin tots els registres als que s'està fent referència

Sempre que introduïm un nou registre a una taula, si les dades que estem introduint fan referència a dades d'altres taules, es comprovarà que les dades existeixin a les altres taules. És a dir, si receptem a un pacient un cert medicament, aquest medicament haurà d'existir. La majoria d'aquestes comprovacions es porten a terme automàticament mitjançant les claus foranes, però haurem de controlar les possibles excepcions que poden generar aquests casos i tractar-les degudament.

4.4.1.5 Data d'alta : data del sistema

Per tots els procediments d'alta, com s'ha pogut veure, no es demana el valor de "data d'alta", encara que a les taules estigui representat. En tots els casos, com a "data d'alta" s'entendrà la data del sistema en el moment d'introduir la dada que s'estigui introduint.

4.4.1.6 Comprovar que existeixi allò que es vol eliminar i que es pugui eliminar

Per tots els procediments d'eliminació/baixa, el primer de tot que farem serà comprovar que allò que s'intenta comprovar existeixi, i, en cas d'existir, que la seva eliminació no provoqui cap excepció perquè s'estiguin fent servir en una relació amb alguna altre taula (caldrà comprovar que cap taula tingui els seus camps com Clau Forana).

4.4.1.7 Tractament d'excepcions

Capturarem qualsevol altre excepció que es produeixi durant l'execució del nostre procediment i que impedeixi l'execució. En tots els procediments inclourem un sistema de traça : Tindrem una variable per indicar fins a quin punt del procediment ha arribat l'execució abans de fallar, per tenir tota la informació possible a la taula de registre (log) de cara a saber en quines condicions i en quin punt s'ha produït el possible error no controlat.

En aquests casos, anul·larem la transacció que hem començat amb el procediment i emmagatzemarem a la taula de registre tota la informació possible de l'error (número identificador de l'excepció, descripció, etc).

4.4.2 Definició dels procediments

A continuació llistarem tots els procediments que proporcionarem com a interfície al nostre sistema. Com hem comentat al punt 4.1, els punts de comprovació comuns a tots els procediments ja no els repetirem a la descripció de cadascun.

Aquest llistat mostra el nom dels procediments, els paràmetres de cada procediment, i les particularitats que cal tenir en compte de cara a la seva implementació (casos en els que hem de pensar, restriccions, ...).

Com que hem d'indicar particularitats de 40 procediments, i la majoria es podran resumir en poques línies, m'ha semblat millor posar només un breu comentari indicant el que s'ha de tenir en compte de tractar a cada procediment, en comptes de fer un diagrama per a cada funció.

Especificació del comportament que ha de tenir cadascun dels procediments :

AltaMedicament (RSP, USER, nom, prospecte, id_medicament)
Afegir medicament
Retornar el id_medicament del nou medicament creat

AltaAlergia (RSP, USER, nom_alergia, descripcio, id_alergia)
Afegir medicament
Retornar el id_alergia de la nova al·lèrgia creada

AltaDiagnostic (RSP, USER, nom, id_diagnòstic)
Afegir diagnòstic
Retornar el id_diagnòstic del nou diagnòstic creat

AltaMedicamentIncompatible (RSP, USER, id_medicament1, id_medicament2)
Crear incompatibilitat
Si algú ha pres aquests dos medicaments → avisar (Warning)

AltaContraindicacio (RSP, USER, id_medicament, id_alergia)
Crear contraindicació
Si algú té aquesta al·lèrgia i ha pres aquest medicament → avisar (Warning)

AltaAlergiaPacient (RSP, USER, dni_pacient, id_alergia)
Si està prenent medicaments que tinguin contraindicacions per aquesta al·lèrgia,
→ avisar (Warning)

AltaEspecialitat (RSP, USER, nom_especialitat, id_especialitat)
Crear especialitat
Retornar el id_especialitat de la nova especialitat creada

AltaCas (RSP, USER, dni_pacient, dni_metge, id_diagnòstic, estat, comentari, id_cas)
Crear cas
Retornar el id_cas del cas creat

AltaVisita (RSP, USER, id_cas, dni_metge_extra, data_visita_programada, durada_programada, id_visita)
En cas d'indicar metge_extra : comprovar que sigui de la mateixa especialitat que el metge que porta el cas. En cas contrari → Error
Comprovar que el metge assignat (el metge del cas o el metge_extra en el seu cas) tinguin hora de visita a la data i hora demanades i no tinguin ja cap visita programada per aquella franja horària. En cas contrari → Error
Crear visita en estat P (programada)
Retornar el id_visita de la visita creada.

AltaRecepta (RSP, USER, id_visita, id_medicament, indicacions, data_fi)
Comprovar que es pugui receptar aquest medicament
Comprovar si el pacient és al·lèrgic a aquest medicament. Si ho és → Error
Comprovar que el pacient no estigui prenent algun medicament incompatible amb el nou medicament que se li està receptant

AltaHorari (RSP, USER, dni, dia_setmana, hora_inici, data_alta, hora_fi, data_baixa)
Comprovar que no hi hagi dos franges solapades del mateix metge → Error
hora inici > hora fi
data_alta <= data_baixa

AltaExcepcioHorari (RSP, USER, dni, dia_setmana, hora_inici, data_alta, hora_fi, data_baixa)

Comprovar que no hi hagi dos franges d'excepció solapades del mateix metge → Error
hora inici > hora fi
data_alta <= data_baixa

AltaPacient (RSP, USER, dni, lletranif, passaport, data_naixement, sexe, adreca, poblacio, provincia, codi_postal, foto, telf_fixe, telf_mobil, fax, tipus_asseguranca, nom_asseguranca)
Comprovar si ja està donat d'alta a la taula persones perquè sigui personal del centre
Si ja està → Actualitzar dades d la taula persones
Sinó → Introduir dades a la taula persones
Introduir dades a la taula pacients

AltaPersonal (RSP, USER, dni, lletranif, passaport, data_naixement, sexe, adreca, poblacio, provincia, codi_postal, foto, telf_fixe, telf_mobil, fax, compte_bancari, username, password)
Comprovar si ja està donat d'alta a la taula persones perquè sigui pacient
Si ja està → Actualitzar dades d la taula persones
Sinó → Introduir dades a la taula persones
Introduir dades a la personal

AltaMetge (RSP, USER, dni, lletranif, passaport, data_naixement, sexe, adreca, poblacio, provincia, codi_postal, foto, telf_fixe, telf_mobil, fax, compte_bancari, username, password, numero_colegiat, id_especialitat, durada_visita)
Comprovar si ja està donat d'alta a la taula persones perquè sigui personal o pacient
Si ja està → Actualitzar dades d la taula persones
Sinó → Introduir dades a la taula persones
Introduir dades a la taula personal i metge

ModificaAlergia (RSP, USER, id_alergia, nom_alergia, descripcio)
Sense particularitats. Es permet modificar el nom i la descripció.

ModificaDiagnostic (RSP, USER, id_diagnòstic, nom)
Sense particularitats. Es permet modificar el nom.

ModificaMedicament (RSP, USER, id_medicament, nom, prospecte)
Sense particularitats. Es permet modificar el nom i el prospecte.

ModificaRecepta (RSP, USER, id_visita, id_medicament, indicacions, data_fi)
Sense particularitats. Es permeten modificar les indicacions i la data fí.

ModificaCas (RSP, USER, id_cas, dni_metge, id_diagnòstic, data_fi, estat, comentari)
Si es modifica el metge assignat al cas :
→ Comprovar que el nou metge sigui de la mateixa especialitat
→ Comprovar que l'antic metge no tingui visites programades per aquest cas, si n'hi ha → ERROR : S'han d'anular primer les visites programades

Si el metge que portava el cas és el metge A i ara ho serà el metge B :
En totes les visites en les que teníem metge_extra a Null, perquè les havia fet el metge A, indicar que ara el metge_extra és el A (perquè el metge que portarà el cas és el B).

Si en algun registre de visita teníem al metge B com metge_extra, deixar aquests registres a Null (ja que ara B és el metge del cas)

ModificaVisita (RSP, USER, id_visita, dni_metge_extra, data_visita_programada, durada_programada, data_visita, durada, estat_visita, preu)
Si es modifica metge_extra :
→ Comprovar que el nou metge sigui de la mateixa especialitat
→ Comprovar que el nou metge tingui disponibilitat a aquesta hora

Si es canvia l'hora de visita_programada, comprovar que l'horari del metge ho permeti

ModificaHorari (RSP, USER, dni, dia_setmana, hora_inici, data_alta, hora_fi, data_baixa)
Es permet modificar només el camp hora_fi i data_baixa

ModificaExcepcioHorari (RSP, USER, dni, dia_setmana, hora_inici, data_alta, data_baixa)
Es permet modificar només el camp data_baixa

ModificaPacient (RSP, USER, dni, lletranif, passaport, nom, cognoms, data_naixement, sexe, adreca, poblacio, provincia, codi_postal, foto, telf_fixe, telf_mobil, fax, tipus_asseguranca, nom_asseguranca, data_baixa, motiu_baixa)
Si es modifica qualsevol dada que afecta a la taula Persones → modificar
Si s'introdueix data_baixa, comprovar que el pacient no tingui visites programades.
Si en té → Error : Indicar que primer s'han de cancel·lar les visites

ModificaPersonal (RSP, USER, dni, lletranif, passaport, nom, cognoms, data_naixement, sexe, adreca, poblacio, provincia, codi_postal, foto, telf_fixe, telf_mobil, fax, compte_bancari, username, password, data_baixa, motiu_baixa)
Si s'intenta modificar un metge mitjançant aquesta funció → Error

ModificaMetge (RSP, USER, dni, lletranif, passaport, nom, cognoms, data_naixement, sexe, adreca, poblacio, provincia, codi_postal, foto, telf_fixe, telf_mobil, fax, compte_bancari, username, password, data_baixa, motiu_baixa, numero_colegiat, id_especialitat, durada_visita)
Si s'intenta donar de baixa un metge : Comprovar que no tingui casos oberts o visites programades com a metge_extra. Si en té → Error
Comprovar també que no tingui franges horàries obertes → Error (s'han de donar de baixa abans)

Per a totes les funcions d'eliminar, la comprovació que hem indicat que faríem és que el registre no s'estigui fent servir en alguna altre taula en alguna relació. Si s'està fent servir (és clau forana des d'una altre taula i es fa servir el registre que es vol esborrar), no es permetrà eliminar el registre. També comprovem que existeixi allò que es vol eliminar, si no existeix, es retornarà un codi d'error.

EliminaMedicament (RSP, USER, id_medicament)

EliminaMedicamentIncompatible (RSP, USER, id_medicament1, id_medicament2)

EliminaContraIndicacio (RSP, USER, id_medicament, id_alergia)

EliminaAlergia (RSP, USER, id_alergia)

EliminaAlergiaPacient (RSP, USER, id_pacient, id_alergia)

EliminaDiagnostic (RSP, USER, id_diagnòstic)

EliminaPacient (RSP, USER, dni)

EliminaPersonalCentre (RSP, USER, dni)

EliminaMetge (RSP, USER, dni)

EliminaCas (RSP, USER, id_cas)

EliminaVisita (RSP, USER, id_visita)

EliminaEspecialitat (RSP, USER, id_especialitat)

EliminaRecepta (RSP, USER, id_visita, id_medicament)

EliminaHorari (RSP, USER, dni, dia_setmana, hora_inici, data_alta)
Comprovar que no hi hagi cap visita programada en aquesta franja

EliminaExcepcioHorari (RSP, USER, dni, dia_setmana, hora_inici, hora_fi, data_alta, data_baixa)

4.5 Disseny del sistema de registre d'activitat (Log)

El nostre entorn tindrà una taula destinada a guardar un registre de les crides que s'han fet als nostres procediments. Ja que serà la única manera mitjançant la qual es podran modificar les dades, aquesta taula ens servirà com a diari de la base de dades. Amb ella hauríem de ser capaços de detectar qualsevol problema que s'hagi produït, i hauríem de ser capaços de reproduir el mateix error, doncs tenim un històric de totes les crides que s'han fet, i podríem reproduir la mateixa situació.

Per a cada crida que facin a un dels nostres procediments guardarem un registre indicant :

- hora de l'execució
- usuari que executa el procediment (el programa que cridi al nostre procediment ens ho haurà indicat mitjançant un paràmetre)
- nom del procediment emmagatzemat
- valor dels paràmetres a l'entrada del procediment
- valor dels paràmetres a la sortida del procediment
- resultat de l'execució (valor RSP tornat per la funció)

- En cas d'excepcions que no haguéssim controlat
    - Traça (posició on s'ha produït l'error)
    - Codi d'error Oracle
    - Descripció d'Oracle de l'error

S'haurà de controlar periòdicament el volum d'aquesta taula, ja que és una taula amb molta informació a cada registre, i és una taula a la que constantment s'afegiran registres. La informació del registre l'hem d'emmagatzemar, però no és necessari tenir sempre a la taula tota la informació de tota l'activitat des del inici de posada en marxa de l'aplicació. Proporcionarem una funció per esborrar el contingut d'aquesta taula, de manera que des dels aplicatius que facin servir el nostre sistema, es puguin copiar les dades d'aquesta taula (per emmagatzemar tot l'històric en fitxers periòdicament a altres sistemes, per exemple).

5 Implementació

5.1 Preparació del sistema (instal·lació de la plataforma de treball)

5.1.1 Instal·lació Oracle

La base del nostre sistema serà el Sistema Gestor de Bases de Dades (en endavant, SGBD) triat per aquest projecte : Oracle 9i Enterprise Edition. Aquest SGBD permetrà totes les funcionalitats que necessitem (és transaccional, permet creació de triggers, índexs, stored procedures amb llenguatge PL/SQL), etc. Oracle és un entorn robust i amb gran quota de mercat actualment.

5.1.2 Requisits del sistema

Els requisits mínim per instal·lar Oracle 9i son :

Microsoft Windows NT/2000/XP
Pentium166 (recomanat 233)
128 Mb de memòria RAM (recomanat 256 Mb)
Monitor VGA de 256 colors

Per treballar en aquest projecte, l'equip de desenvolupament té unes característiques molt superiors a les mínimes sol·licitades :

Microsoft Windows XP Professional
Intel Pentium Centrino 1.3 GHz
512 Mb de memòria RAM
Monitor XGA

5.1.3 Preparació del sistema operatiu per l'instal·lació

Seguint els passos del document "Procediment bàsic d'instal·lació d'oracle 9i", el primer que hem de fer és comprovar que a la variable d'entorn PATH no es faci referència a més d'una màquina virtual de Java. Això pot ser degut a haver instal·lat varies versions de la màquina virtual de Java al mateix equip. Per a fer aquesta comprovació, seguirem aquests passos :

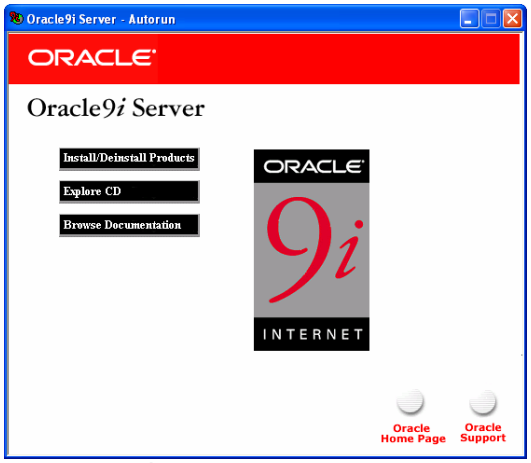
1. Icona Mi Pc -> Botó dret -> Propietats
2. Pestanya Opciones Avanzadas -> Botó Variables de entorno
3. A la secció de Variables del Sistema, buscarem PATH
4. Seleccionem Modificar, i revisem el contingut de la variable

En el nostre sistema, es fa referència a dos versions diferents : "j2sdk1.4.1_01" i "jdk1.3.1_01". Revisant els directoris d'instal·lació, veiem que només hi ha instal·lada una de les dos versions al sistema, així que eliminem les entrades de l'altre versió que hem trobat al PATH.

Segon pas : Comprovar que a la variable CLASSPATH no es faci referència a cap arxiu d'oracle. Per comprovar aquesta variable, anem al mateix diàleg on hem consultat la variable PATH. En el nostre sistema, el contingut de la variable és correcte.

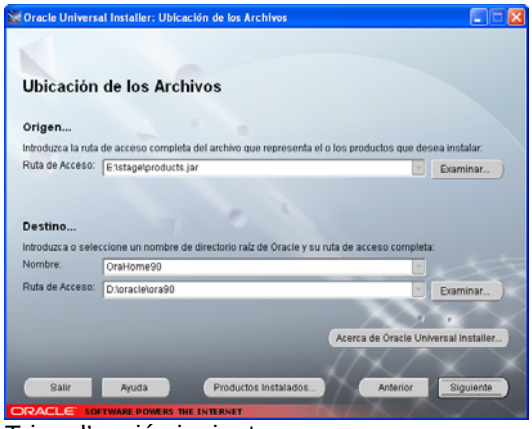
5.1.4 Instal·lació de l'SGBD

Aquest és el procés d'instal·lació que s'ha dut a terme, s'inclouen les captures de pantalla amb les dades que s'han especificat per l'instal·lació.



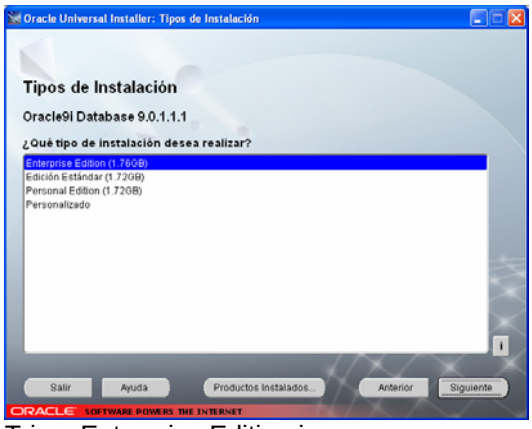
Triem l'opció Install Products

Triem l'opció Siguiente

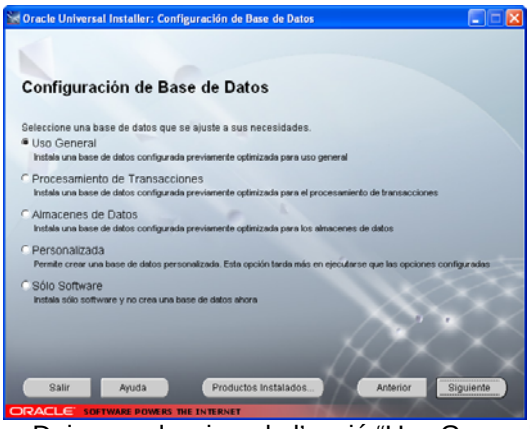


Triem l'opció siguiente
(deixant valors que venien per defecte)

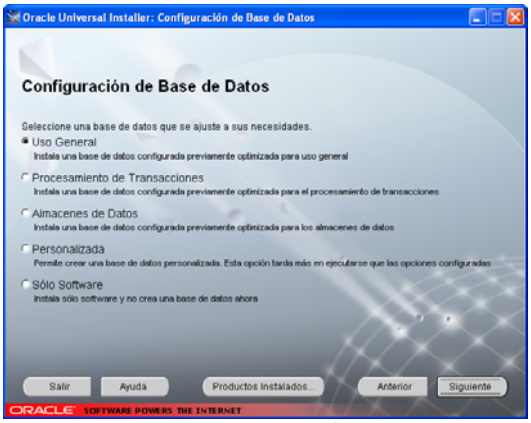
Triem Oracle9i Database 9.0.1.1.1
premem el botó Siguiente



Triem Enterprise Edition i
premem el botó Siguiente



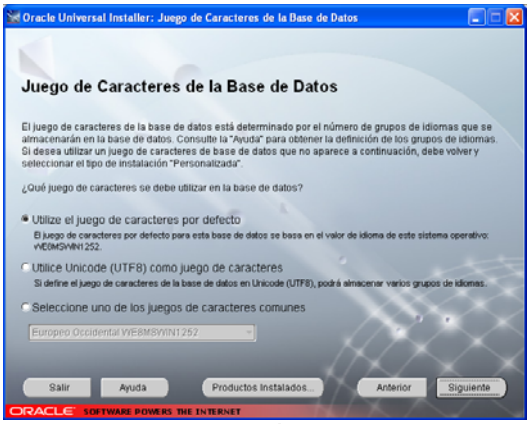
Deixem seleccionada l'opció "Uso General"
(ens instal·larà una base de dades
d'exemple amb la qual podrem fer proves
al sistema), i prenem Siguiente



Deixem seleccionada l'opció "Uso General" (ens instal·larà una base de dades d'exemple amb la qual podrem fer proves al sistema), i prenem Siguiente

Com a nom de la base de dades global, triem "uoc" i premem Siguiente

En aquesta finestra ens està informant del directori on s'emmagatzemaran els fitxers de la nostre base de dades. Deixem el directori que ens proposa per defecte al nostre segon disc dur, i anotem el path on es guardaran els fitxers. Premem siguiente



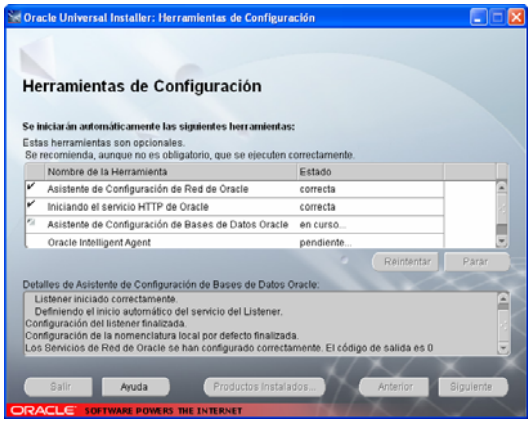
Deixem el joc de caràcters per defecte Premem Siguiente

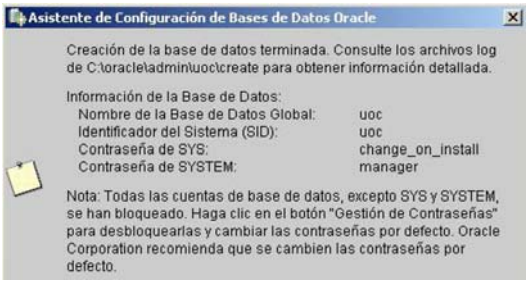
… … …

Ens presenta un resum de tots els productes que s'instal·laran. Premem Instal·lar

La instal·lació és un procés lent Se'ns demanarà anar canviant els CD's d'instal·lació.

En equips amb WindowsXP amb SP2 instal·lat i amb el Firewall de Windows activat, se'ns demanarà obrir els ports que han d'obrir algunes aplicacions que acabem d'instal·lar, com xsagent o Apache.





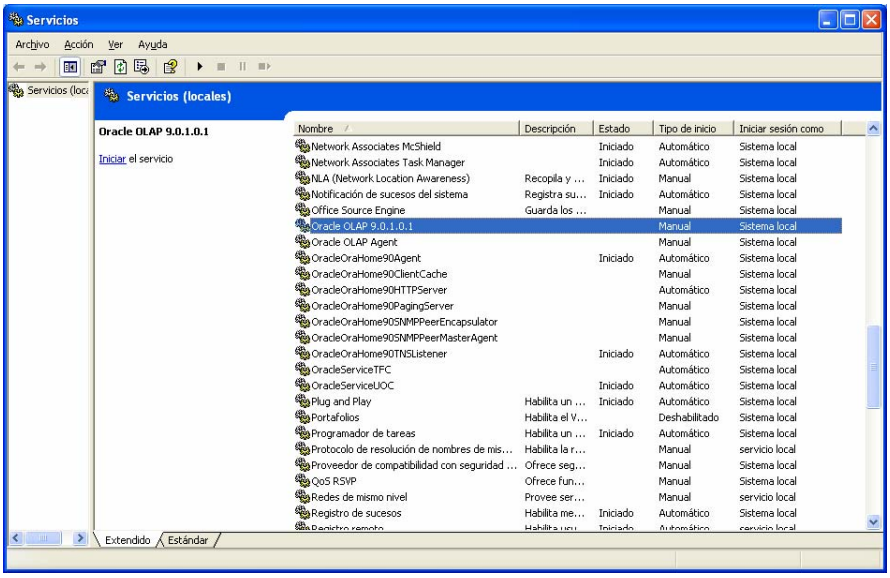
5.1.5 Iniciar/Detenir serveis d'oracle

Per defecte, a l'instal·lació, alguns dels serveis d'oracle queden configurats per iniciar-se automàticament. Per a la màquina en la que el servei estigui en explotació, deixarem aquesta configuració així. Per a la màquina de desenvolupament, voldrem tenir els serveis funcionant només quan necessitem posar-los en prova, la resta del temps, alliberarem els recursos que fa servir Oracle.

Anirem a la finestra de configuració de Serveis de Windows :

→ MiPc → Botó dret → Administrar → Servicios y Aplicaciones → Servicios

O bé : Panel de control → Herramientas Administrativas → Servicios



Els serveis que necessitem funcionant per treballar amb Oracle son :

OracleOraHome90Agent
OracleOraHome90HTTPServer
OracleOraHome90TNSListener
OracleServiceUOC (aquesta és la BD que hem creat)

Per defecte, el tipus d'inici és Automàtic, el canviarem a Manual.

Quan vulguem arrencar aquests serveis, o bé ho farem des de aquesta mateixa finestra, amb botó dret sobre el servei que volem arrencar i triant l'opció Iniciar, o bé mitjançant una comanda des de la línia de comandes (ens podem preparar un petit script bat que arrenqui/aturi els quatre serveis), amb les comandes : net start nomservei / net stop nomservei. On substituirem "nomservei" pels noms dels serveis que hem indicat anteriorment.

5.1.6 Contrasenyes i usuaris

Els usuaris que s'han creat durant el procés d'instal·lació son :

Información de la Base de Datos:	
Nombre de la Base de Datos Global:	uoc
Identificador del Sistema (SID):	uoc
Contraseña de SYS:	change_on_install
Contraseña de SYSTEM:	manager

Aquests usuaris, (SYS i SYSTEM), son usuaris que farem servir només per a feines d'administració de la base de dades. Des de les nostres aplicacions ens interessarà tenir usuaris amb un nivell de permisos menor.

Crearem dos usuaris : Un usuari amb privilegis totals sobre l'espai UOC, i un altre al que només donarem permisos de lectura.

Usuari amb privilegis totals :

Aquest serà l'usuari que farem servir des de el nostre codi PL-SQL per treballar contra la base de dades.

Nom d'usuari : AdmConsultori
Password : pasw2222

Nom d'usuari amb permisos de lectura :

Aquest serà l'usuari que donarem als futurs usuaris de la base de dades (Aplicacions que treballin contra el nostre sistema), els quals han de poder accedir a les taules directament només en mode lectura.
Nom d'usuari : Consultori
Password : pasw1111

Per crear el primer Usuari (AdmConsultori), ens connectarem a PL-SQL Developer (ens identificarem amb les dades de l'usuari SYSTEM), i obrirem un nou "Command-Window", on introduirem aquest Script :

```
GRANT CONNECT TO AdmConsultori IDENTIFIED BY "pasw2222";
GRANT CONNECT TO Consultori IDENTIFIED BY "pasw1111";

GRANT ALL PRIVILEGES TO AdmConsultori;
```

Per a crear el segon usuari (Consultori), amb només permisos de lectura, ens connectarem amb les credencials de l'usuari AdmConsultori i executarem un script d'aquest estil, indicant totes les taules i procediments sobre els quals volem donar permisos al nou usuari :

```
GRANT CONNECT TO Consultori IDENTIFIED BY "pasw1111";

-- Permisos de lectura sobre les taules :
GRANT SELECT ON ADMCONSULTORI.ALERGIES TO CONSULTORI;
GRANT SELECT ON ADMCONSULTORI.ALERGIESPACIENTS TO CONSULTORI;
...
-- Permisos d'execució per als procediments :
GRANT EXECUTE ON ADMCONSULTORI.AltaAlergiaPacient TO CONSULTORI;
GRANT EXECUTE ON ADMCONSULTORI.AltaContraIndicacio TO CONSULTORI;
...
```

Als Scripts de creació del sistema (Veure punt 5.4.4 de la memòria) es donen més detalls sobre aquests scripts.

5.2 Instal·lació PL-SQL Developer

Aquest és el software que farem servir per editar el codi del nostre projecte i per fer totes les proves d'execució contra Oracle.

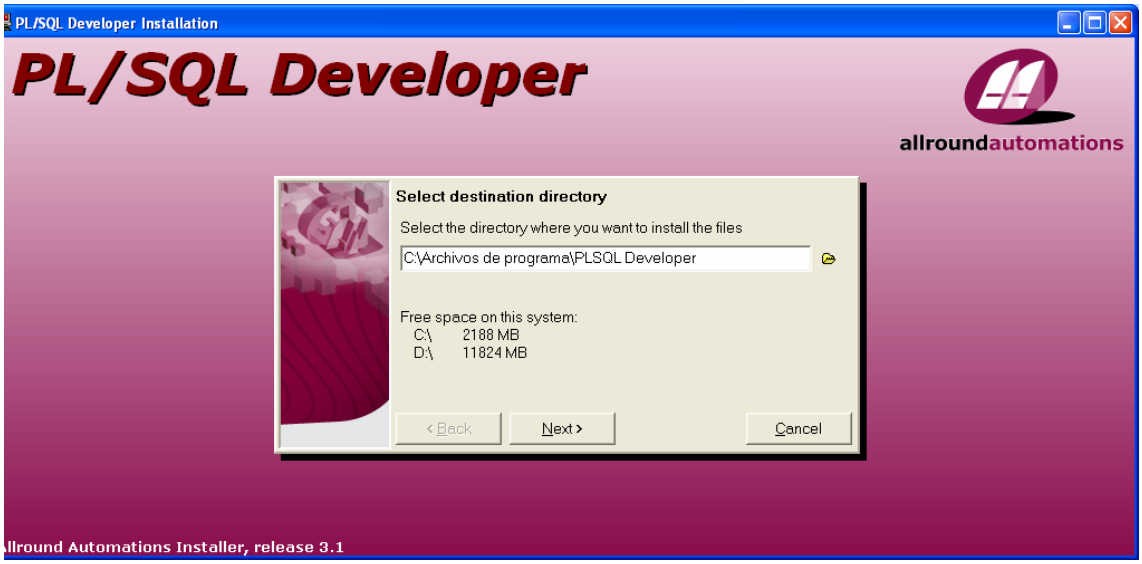
Ens permet editar i executar scripts PL/SQL, executar consultes SQL per fer proves, veure les taules i objectes definits al nostre sistema Oracle, i totes les funcionalitats que necessitem pel desenvolupament del nostre projecte.

Hem aconseguit una versió de prova per 30 dies que ens ha facilitat el consultor. Amb aquests dies tindrem suficient pel desenvolupament

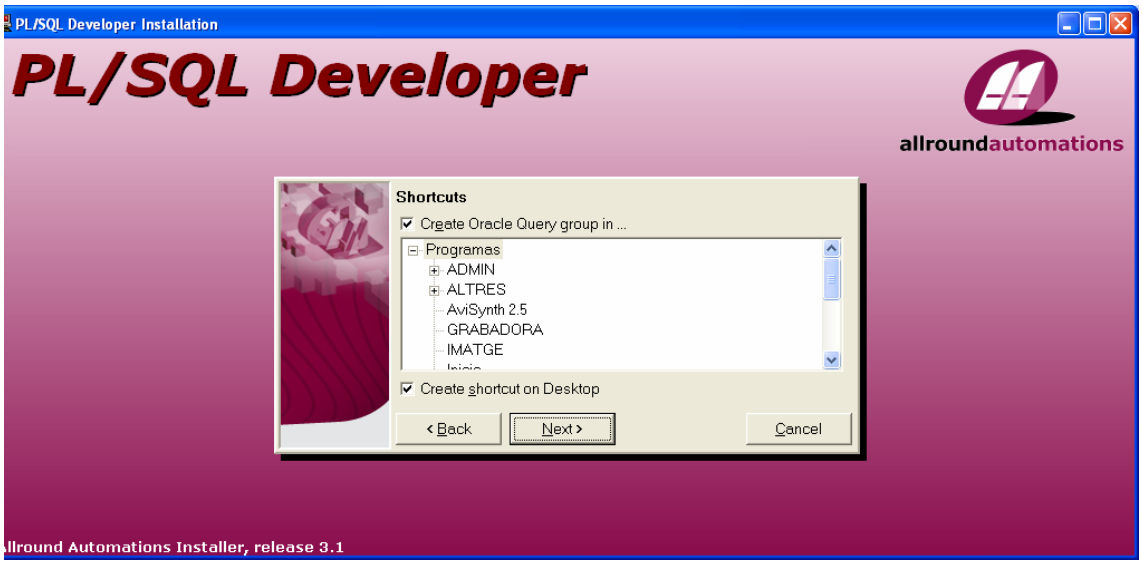
5.2.1 Instal·lació de PL-SQL Developer

Per descarregar aquest software podem accedir a la URL d'allroundautomations : http://www.allroundautomations.com , on podem aconseguir una versió de demostració per avaluar-la en un període de 30 dies.

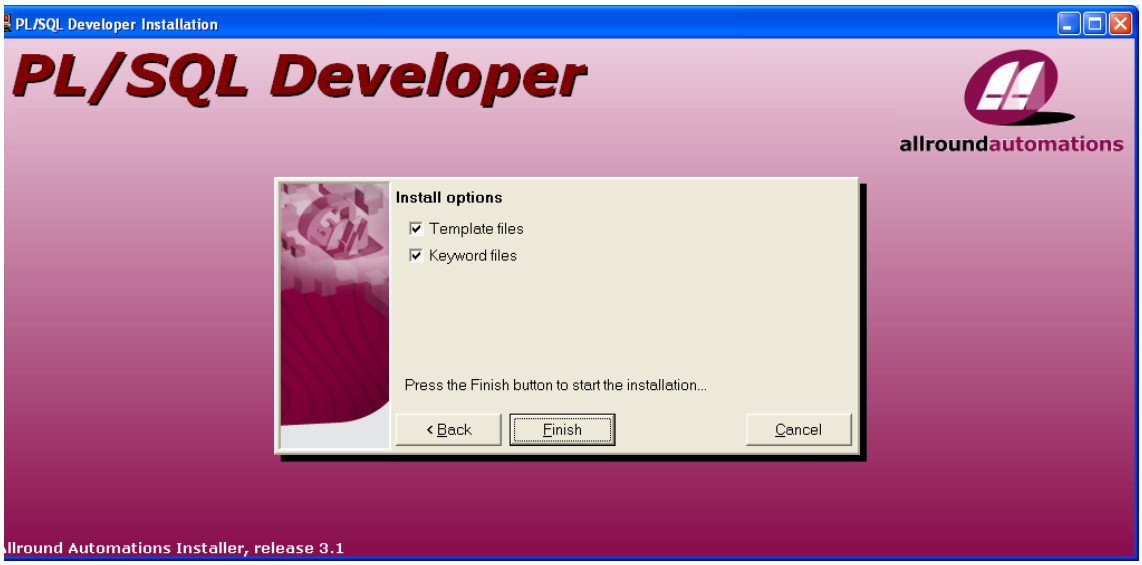
I Agree



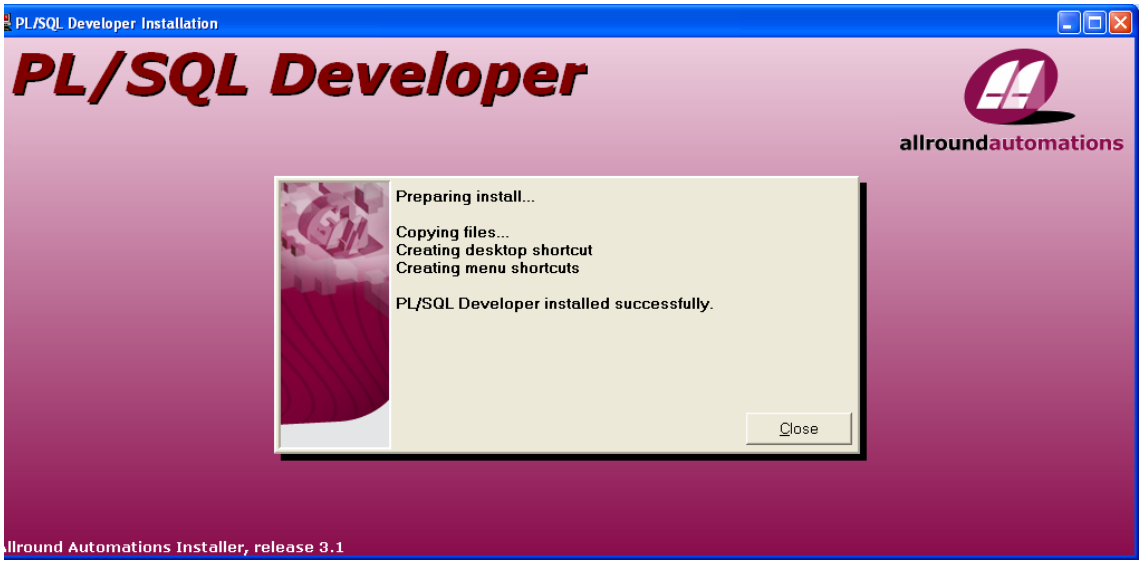
Next



Next



Finish



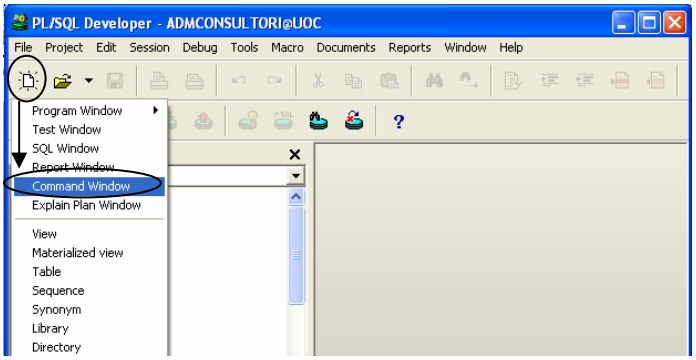
Close

5.2.2 Paquets d'idioma

Si volem disposar de la interfície de PL-SQL Developer en algun llenguatge diferent al que tenim a la instal·lació per defecte (Anglès), podem aconseguir fitxers amb paquets d'idioma a la URL : http://www.allroundautomations.com/plsqldevlang/60/index.html . (El paquet per idioma Castellà està disponible).

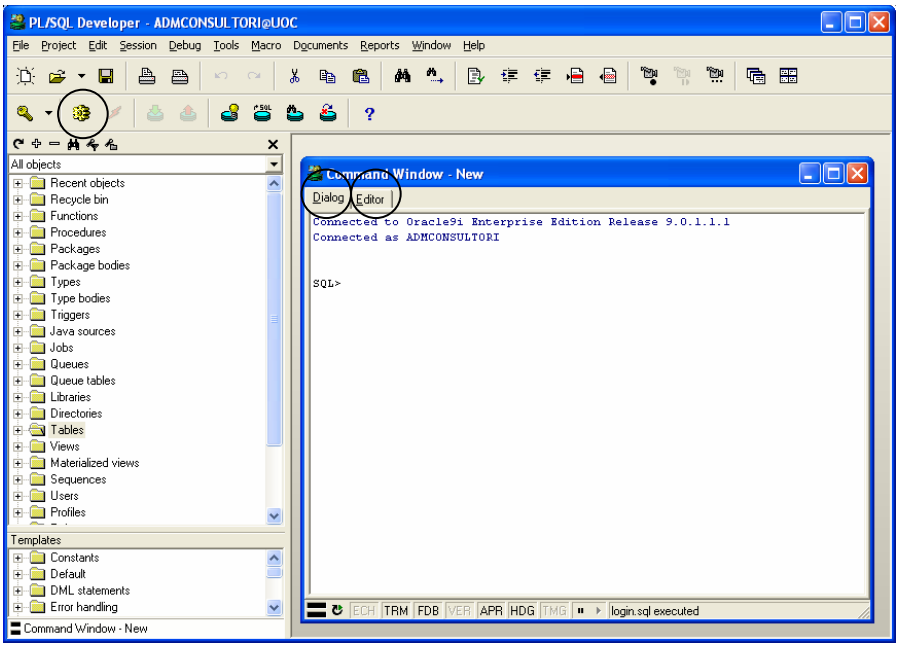
5.2.3 Execució d'scripts SQL des de PL-SQL Developer

Per executar scripts com el de l'apartat anterior (creació usuaris), obrirem un nou "Command window" :



Des de la finestra que se'ns obrirà, a la pestanya Editor podem escriure directament l'script, enganxar-lo des de el clipboard de Windows, o fer botó dret, opció Load, i carregar el nostre fitxer d'script.

Per executar les comandes, o bé premerem el botó de l'engranatge ( ) de la barra d'eines, o bé premerem la tecla F8 (utilitzada a molts entorns per la funció Executar).



NOTA : Per treballar amb més comoditat, al combo de selecció que apareix a sobre del menú de l'esquerra de l'aplicació (menú amb forma d'arbre amb totes les opcions), en comptes de deixar seleccionat "All Objects", seleccionarem "My Objects" per veure només els objectes creats per nosaltres (i no veure tots els objectes que hi ha a la base de dades creats pel sistema)

5.3 Accés a la Base de Dades des de Microsoft Access

Per visualitzar les dades de les taules que creem al nostre SGBD Oracle, farem servir el programa Access de Microsoft, ja que les vistes que presenta aquest sistema per accedir a les dades de les taules son molt còmodes.

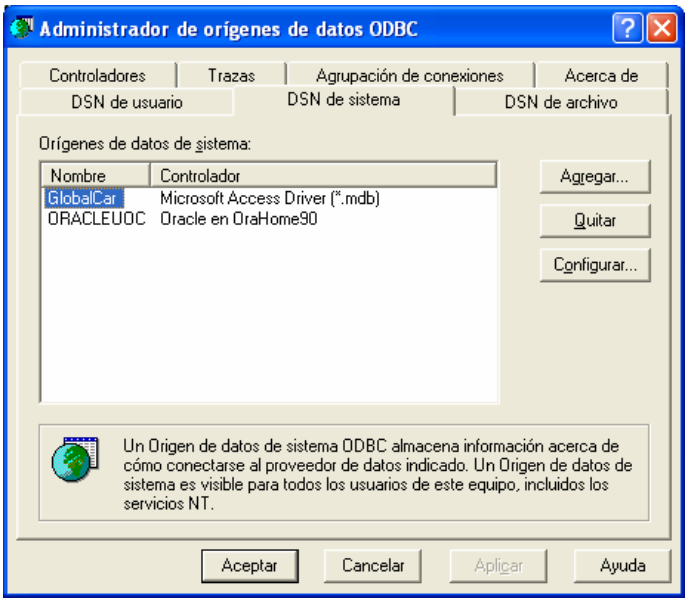
El que farem serà vincular les taules del nostre sistema des d'una base de dades d'Access.

Per connectar-nos, farem servir la capa intermèdia de connexió per a Bases de Dades de Microsoft : ODBC (Open Database Connectivity). Ens crearem un DSN (Data Source Name) amb el Driver d'oracle (que s'ha instal·lat durant el procés d'instal·lació d'oracle) que "*apunti*" a la nostre Base de Dades Oracle (podem entendre aquest DSN que crearem com un "*Accés directe*" cap a la Base de dades del servidor Oracle).

5.3.1 Creació d'un DSN

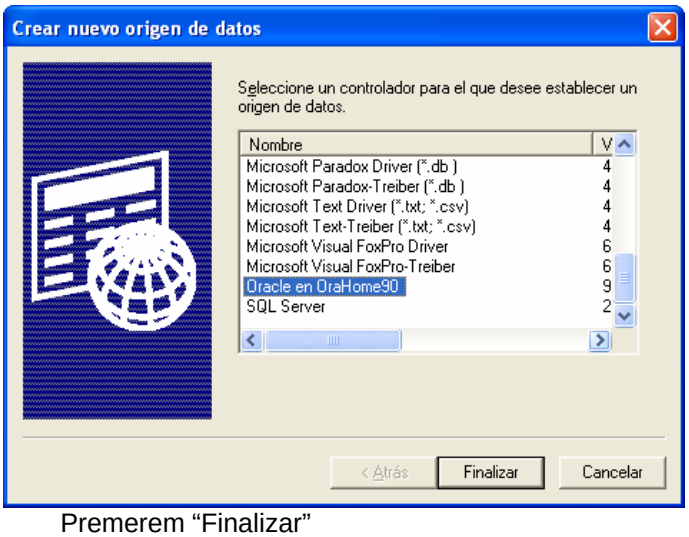
Per crear un DSN amb el Driver d'oracle, anirem a :

Menú Inició → Panel de control → Herramientas Administrativas → Orígenes de datos (ODBC) (pestanya DSN de sistema)



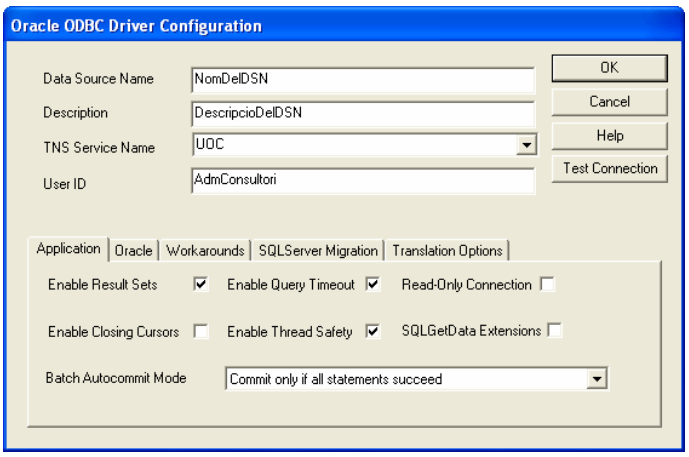
Premerem "Agregar"

Seleccionem el Driver que volem fer servir. En el nostre cas, com volem fer un "apuntador" a una Base de Dades Oracle, seleccionarem el Driver "Oracle en OraHome90" :

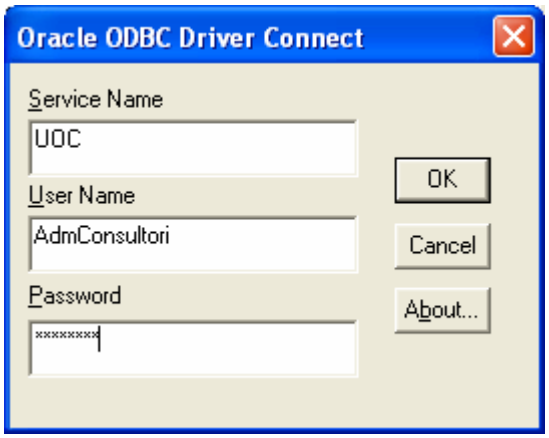


Premerem “Finalizar”

A la següent finestra indicarem : El nom que volem donar a aquest “apuntador”, la descripció, TNS Service Name (a quina base de dades volem connectar-nos : nosaltres triarem UOC), i UserID : usuari amb el que volem connectar-nos a la BD (triarem AdmConsultori).



Per provar que aquest “apuntador” funciona correctament, polsarem el botó “Test Connection”, i ens demanarà la contrasenya per aquest usuari que hem indicat :



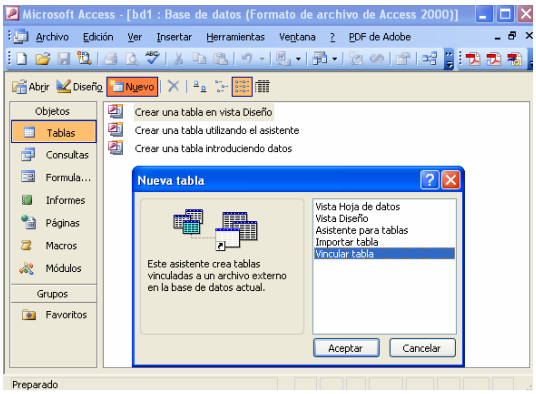
Si tot és correcte, ens avisarà amb un diàleg on indicarà : “Connection successful”

5.3.2 Vincular taules des de Microsoft Access

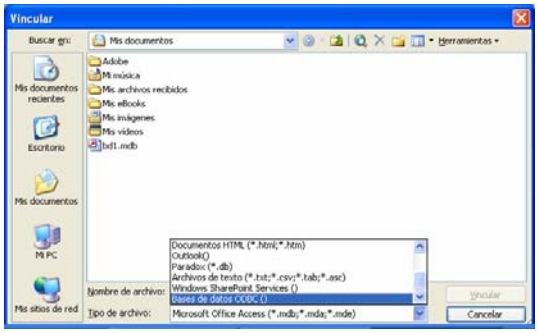
Ara, el que farem, serà obrir MSAccess i vincular des d'allà les taules del nostre sistema Oracle :

Primer de tot, crearem una Base de Dades en blanc a Microsoft Access.

Des de la pestanya Taules, polsarem el botó "Nueva" i triarem "Vincular tabla".

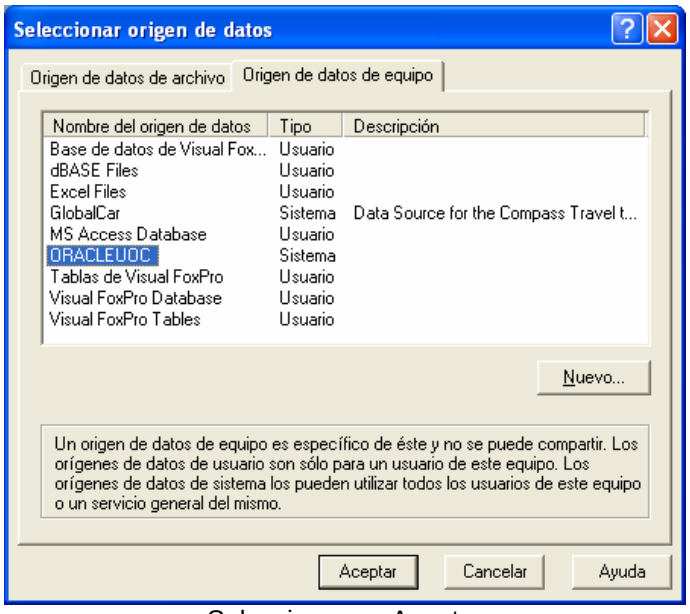


Seleccionarem "Aceptar"



A "Tipo de archivo" seleccionarem Bases de datos ODBC

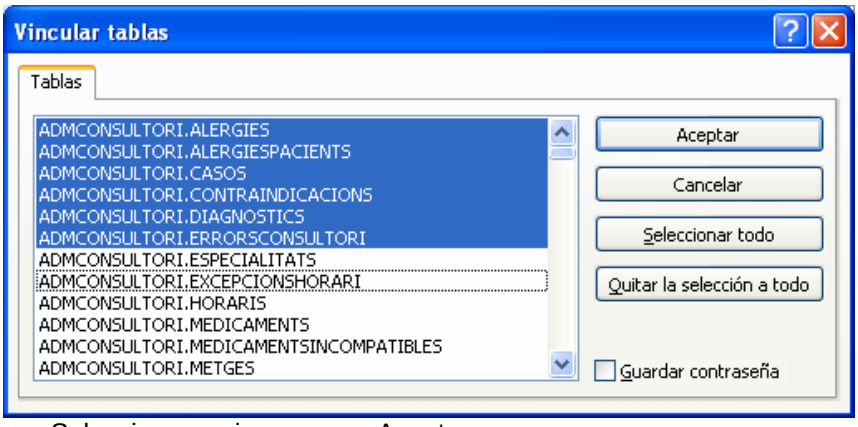
A la pestanya "Origenes de datos de Equipo", triarem el nostre DSN
(en el meu cas, l'he anomenat ORACLEUOC)



Seleccionarem Aceptar

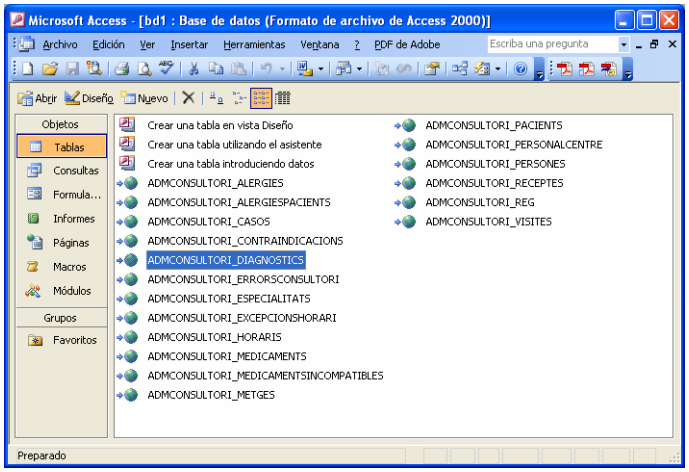
Ens demanarà la contrasenya :

Seleccionarem per vincular totes les taules que comencin amb el nom d'usuari "admconsultori" (la resta son taules del sistema que no ens interessen) :

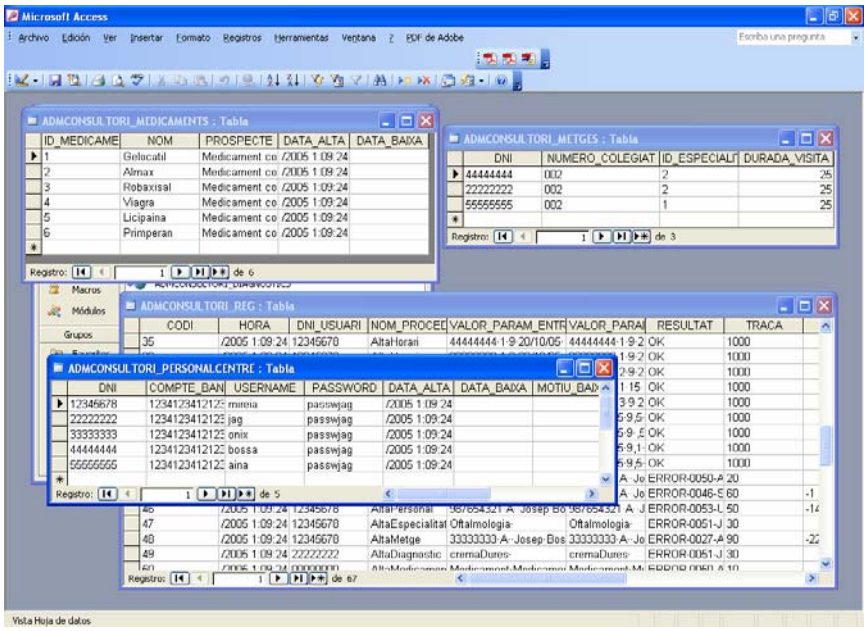


Seleccionarem i premerem Aceptar

Ara, la nostre Base de dades tindrà Vinculades les taules de la Base de Dades d'oracle :



Com l'usuari amb el que ens estem connectant té permisos d'escriptura sobre aquestes taules, podrem consultar, modificar o esborrar dades de les taules (sempre que complim les restriccions de claus primàries, camps no nuls, claus foranes, ..., ...)


Microsoft Access
Archivo Edición Ver Insertar Formato Registros Herramientas Ventana ? PDF de Adobe
ADMCONSULTORI_MEDICAMENTS : Tabla
ID_MEDICAME NOM PROSPECTE DATA_ALTA DATA_BAIXA
1 Gelocatil Medicament co /2005 1:09:24
2 Almax Medicament co /2005 1:09:24
3 Robaxisal Medicament co /2005 1:09:24
4 Viagra Medicament co /2005 1:09:24
5 Licipaina Medicament co /2005 1:09:24
6 Primperan Medicament co /2005 1:09:24
ADMCONSULTORI_METGES : Tabla
DNI NUMERO_COLEGIAT ID_ESPECIALI DURADA_VISITA
44444444 002 2 25
22222222 002 2 25
55555555 002 1 25
ADMCONSULTORI_REG : Tabla
CODI HORA DNI_USUARI NOM_PROCED VALOR_PARAM_ENTR VALOR_PARA RESULTAT TRACA
ADMCONSULTORI_PERSONALCENTRE : Tabla
DNI COMPTE_BAN USERNAME PASSWORD DATA_ALTA DATA_BAIXA MOTIU_BAI
12345678 mirela passwjag /2005 1:09:24
22222222 jag passwjag /2005 1:09:24
33333333 onix passwjag /2005 1:09:24
44444444 bossa passwjag /2005 1:09:24
55555555 aina passwjag /2005 1:09:24
Macros
Módulos
Grupos
Registro:
Vista Hoja de datos

5.3.3 Error consultant les dades des d'Access

Hem comprovat que Access presenta algun error al visualitzar les dades de determinats tipus de camps. Per exemple, al visualitzar camps que hem definit en Oracle com NUMBER(n,d) (Número de n dígits, dels quals d son decimals), Access no mostra aquests valors correctament (ignora el punt decimal i mostra el valor sencer).
Per tant, hem de fer servir aquestes vistes des d'Access per fer comprovacions de si les dades s'estan inserint o no correctament a les taules, però hem de tenir en compte que algun valor es podria estar visualitzant de manera incorrecta per conversions de tipus d'aquest estil.

En cas de dubte, si volem assegurar-nos de quines dades tenim realment a la Base de Dades d'oracle, ens connectarem des de l'entorn PL-SQL Developer, i des d'aquesta aplicació, obrirem una "SQL Window" per fer les consultes de les dades que vulguem comprovar.

5.4 Aspectes generals de la implementació

En aquest apartat s'explicaran criteris que s'han seguit en tota la implementació. Alguns d'ells son funcions específiques que ofereix Oracle i que hem fet servir, i alguns altres son maneres de tractar un tipus de problema, i que s'han repetit a diversos procediments.

També es donarà una visió sobre aspectes generals del codi (com s'ha estructurat, comentat) com s'ha organitzat (en scripts), com es tracten els paràmetres a tots els procediments, etc.

5.4.1 Codi estructurat i comentat

A tota la implementació s'ha donat molta importància al fet de fer un codi que sigui fàcilment llegible, de cara a que si un altre usuari l'ha de consultar, modificar, o mantenir en el futur, s'hi pugui familiaritzar ràpidament.

Buscant aconseguir aquest objectiu :

- S'ha indentat tot el codi

- S'ha respectat un codi de majúscules/minúscules (per exemple, totes les funcions i paraules clau SQL estan escrites en majúscules)

- S'han posat tots els comentaris que han semblat necessaris per facilitar la comprensió de la implementació. Per a cada bloc de codi destinat a fer una funció concreta, es comenta abans quina és la funció que farà aquell codi

- Les consultes SQL s'han separat en vàries línies, separant les clàusules SELECT, FROM, WHERE (pel cas de les consultes), una en cada línia.

5.4.2 Tipus de les variables : %type

A l'hora de definir variables als procediments, s'ha fet servir una característica d'oracle que ens permet assignar a una variable el mateix tipus de variable que tingui una determinada variable.

És a dir, si necessitem crear-nos una variable per emmagatzemar el Dni d'un metge (valor que obtindrem potser executant una consulta) podem definir una variable amb nom DNI, i dir al sistema que aquesta nova variable que estem definint tingui el mateix tipus que té el camp Dni de la taula Metges.

Això ho faríem d'aquesta manera :

```
dni Metges.DNI%type;
```

Les avantatges de treballar així son clares :

- No hem de recordar el tipus exacte de cada camp de les taules de la nostre base de dades mentre estem programant i necessitem definir una variable.
- Si per algun motiu hem de modificar el tipus de variable d'un camp d'una taula, automàticament canviarà el tipus de les variables dels procediments que hem creat d'aquesta manera, i evitarem haver de repassar tot el codi per canviar el tipus a totes les variables.

És una llàstima, però aquesta tècnica no es pot fer servir a l'hora de definir els tipus dels camps de les taules de dades. Aquesta funcionalitat seria molt útil per definir els tipus dels camps que son clau forana d'alguna altre taula (que evidentment han de ser del mateix tipus a les dos taules).

5.4.3 Paràmetres als procediments

Tots els noms dels paràmetres de les funcions, per tal de distingir-los de la resta de variables a dintre del procediment, s'han anomenat : començant-los amb una lletra p minúscula, la següent lletra en majúscula, i la resta del camp en minúscula.

En els camps que son de sortida s'ha indicat el modificador OUT per poder fer variar el seu valor (l'equivalent a passar una variable per referència en llenguatge C), i en els que no s'ha indicat, es per què per defecte el tipus d'un paràmetre és IN (passat per valor).

Els tipus dels paràmetres s'han definit, com hem explicat al punt anterior, fent servir la característica %type dels camps als que faran referència a les taules.

Els dos primers paràmetres als procediments sempre son els mateixos :

pRsp : Resultat de Sortida del Procediment, que indicarà el resultat de l'execució segons uns codis de resultats que comentarem més endavant.

pUser : Dni de l'usuari que vol executar aquella funció (el metge, o la persona del centre que vol intentar afegir aquell medicament, o pacient, o visita, o ...). D'aquesta manera, a la taula de registre d'activitat sempre tindrem constància de qui ha executat cada operació executada.

5.4.4 Scripts

Tot el codi que prepararem el guardarem a fitxers de text que podrem copiar després a l'editor PL-SQL Developer (o a qualsevol altre editor que ens permeti connectar-nos a la Base de Dades per executar un Script). Per veure amb detall com executar aquests scripts amb PL-SQL Developer consultar el punt 5.2.3 d'aquesta memòria.

Entregarem quatre scripts a la part dels Annexes de la memòria :

SCRIPT 1 : Creació de l'usuari Administrador

Aquest script l'haurem d'executar com a usuari SYSTEM, i ens crearà l'usuari que necessitem per treballar amb la nostre base de dades UOC.

SCRIPT 2 : Creació de tots els elements que formen el projecte

Aquest script l'haurem d'executar identificats al sistema com l'usuari AdmConsultori (qui te permisos totals sobre la base de dades UOC.

Aquest script farà les següents funcions :

- Creació de taules
- Creació dels triggers
- Creació d'índexs
- Creació de seqüències
- Inserció dels registres de la taula d'errors
- Creació del package amb les funcions internes que ens hem creat
- Creació dels procediments
- Creació de l'usuari Consultori (per a consultes i execució de procediments)
- Commit !

La creació de les taules s'haurà de fer en ordre, pensant en que primer cal crear unes taules, i després aquelles altres taules que tinguin claus foranes referenciant a taules que ja haguem creat.

SCRIPT 3 : Joc de proves

Aquest script contindrà les crides als procediments que hem creat.
Totes les primeres crides seran amb paràmetres correctes, de cara a omplir la base de dades amb dades per poder-hi treballar.
Un cop tinguem dades introduïdes correctes, farem un segon joc de proves (seguit, en el mateix script), comprovant que les funcions fallin de la manera que s'espera quan passem paràmetres incorrectes que fan saltar les restriccions que imposa el nostre sistema per assegurar la integritat i la consistència de les dades.
Aquest script escriurà per la sortida standard (mitjançant la funció `dbms_output.put_line (VARCHAR2)`) : una línia de comentari indicant quan comença el joc de proves amb dades correctes, una línia de comentari indicant quan comença el joc de proves amb dades que han de donar error, i una línia per cada execució, indicant la funció que estem executant i el resultat de sortida del procediment (RSP).

Entreguem un fitxer (resultat_execucio_jocproves.txt) que indica la sortida de l'execució de l'script, en el que també s'ha copiat el resultat d'una consulta SELECT sobre la taula de registre d'activitat per mostrar els canvis produits per aquest joc de proves sobre el sistema (En aquest resultat, per que les dades siguin més llegibles, s'han copiat només 4 dels 10 camps de la taula de registre).

Aquesta és una mostra del format del resultat que presenta l'script del joc de proves al ser executat :

```
** Provant casos que haurien de funcionar (si partim de la BD buida)
Alta Personal           : OK
Alta Pacient            : OK
Alta Especialitat       : OK
   ... etc etc etc ...

** Provant casos que NO haurien de funcionar
Alta Personal           : ERROR-0050-Aquest usuari no té permís per fer aquesta funció
Alta Personal           : ERROR-0053-Un dels camps excedeix la longitud permesa
Alta Especialitat       : ERROR-0051-Ja existeix un registre amb el mateix contingut (nom)
Alta Metge              : ERROR-0027-Aquesta especialitat no està donada d'alta
Alta Cas                : ERROR-0018-Aquest metge no esta donat d'alta
Alta Cas                : ERROR-0052-S'ha deixat en blanc un camp obligatori
Alta Horari             : ERROR-0023-Aquest metge no esta donat d'alta
   ... etc etc etc ...
```

SCRIPT 4 : Eliminació de tots els elements de la base de dades

Aquest script eliminarà tots els elements creats amb l'script 3. De manera que si hem estat fent proves inserint valors a la base de dades, i volem tornar al principi, amb tota la base de dades buida, només caldrà eliminar tots els elements (executar aquest script 4) i tornar-los a crear (executant l'script 2).

En aquest cas, l'ordre per eliminar les taules també ha de tenir en compte que primer cal esborrar aquelles taules que no son referenciades per cap clau forana des d'altres taules. El més còmode és prendre com a ordre d'esborrat l'ordre invers de creació de les taules, així assegurem que no hi haurà problema.

5.4.5 Camps autonumèrics : Seqüències

A algunes taules, el camp creat per a ser clau primària, és només un camp que volem que sigui un identificador numèric (sense cap més sentit que el de numerar inequívocament cadascun dels registres de la taula).

Per a aquesta finalitat, altres sistemes gestors de bases de dades (Informix, MySql, SQLServer, 'Access' (aquest últim no és considerat un SGBD)) proporcionen un tipus de dades anomenat 'Autonumèric'.

Aquests camps 'Autonumèric' son camps que a la inserció d'un nou registre, van generant un nou número, consecutiu a l'últim que van generar normalment (hi ha sistemes que permeten definir que els números incrementin de n en n, per exemple), de manera que sempre generen números diferents.

Oracle no incorpora aquest tipus de dades, però incorpora un objecte anomenat SEQÜÈNCIA. Podem crear tants objectes Seqüència com ens sigui necessari, i indicar el valor inicial i l'increment per a cada nou número generat en aquesta seqüència.

Podem consultar sempre l'últim número de la seqüència generat o bé executar una instrucció perquè la seqüència generi i ens entregui el següent valor.

5.4.6 Sistema de gestió d'hores

Per a les funcions relacionades amb la programació de visites (aquelles que insereixin valors a la taula d'horaris de Metges o que insereixin visites), s'hauran de fer comprovacions de la consistència de les dades.

5.4.6.1 Solapaments de franges horàries a l'horari d'un metge

Per exemple, **si estem introduint l'horari en el qual treballa un metge**, no podrem permetre que introdueixi dos franges horàries solapades pel mateix dia. Recordarem ràpidament com es plasma a les taules l'horari de treball d'un metge :

Taula Horaris (els formats dels camps no son aquests, s'expliquen després) :

Metge	DiaSet	H.Inici	H.Fi	Data Alta	Data Baixa
Id1	Dilluns	9.00	14.00	01/01/2005	
Id1	Dilluns	17.30	19.00	01/01/2005	
Id1	Dimarts	10.00	15.00	01/01/2005	
Id1	Divendres	12.00	16.00	01/01/2005	01/03/2006

On es pot observar que el metge Id1 treballa (des de el dia 01/01/2005) :

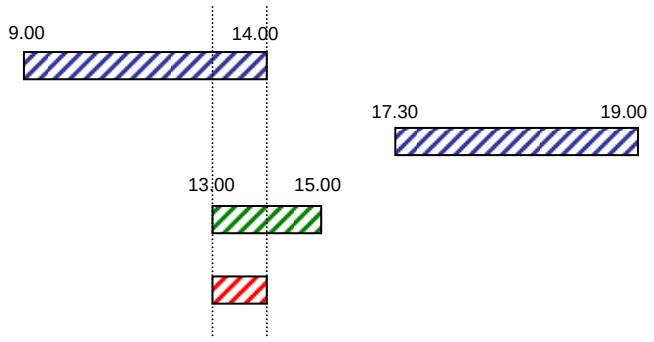
Dilluns de 9.00 a 14.00 i de 17.30 a 19.00
Dimarts de 10.00 a 15.00
Divendres de 12.00 a 16.00 entre el 1/1/2005 i el 1/3/2006

Per tant, intentar introduir que el metge Id1 treballa els Dilluns de 13.00 a 15.00 (en un període que comprengui dates superiors a l'1/1/2005, que és quan és vàlid el període definit al primer registre) el sistema no ens ho ha de permetre (ja que aquesta franja se solaparia amb una altre que ja té introduïda).

Per fer aquesta comprovació, hem de comprovar que no se solapin dos entrades de registres d'horaris de : Mateix metge, mateix dia de la setmana i que no se solapin en DATA i en HORA.

Comprovarem, primer si se solapa per HORA, la franja que volem introduir amb totes les franges que hi hagi al sistema per aquell metge i dia de la setmana :

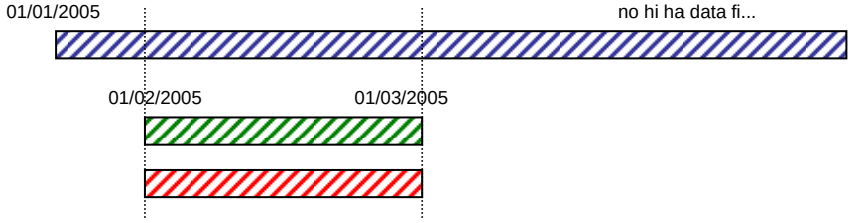
A l'esquema marquem amb trama blava les franges que corresponen a registres ja introduïts a la base de dades, i en verd la franja que s'intenta introduir.



Com es veu, es produeix una zona de solapament (indicador vermell), i per tant, en principi no es possible introduir la nova franja horària, perquè se solapa amb una d'existent. S'ha de veure, però, que se solapi també en DATA, ja que igual la franja amb la que estem comparant finalitza al Març i intentem afegir aquesta franja per l'abril (en aquest cas, hauria de permetre l'inserció).

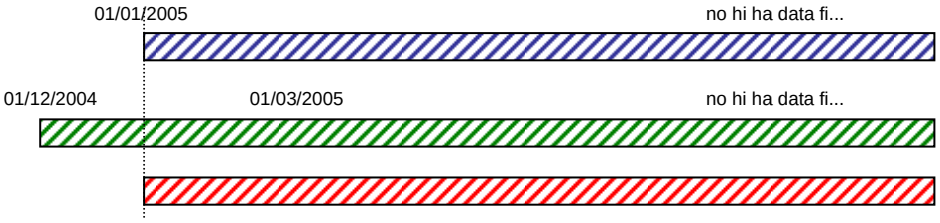
Es comprova, per tant, si el solapament, a més de ser en HORA, també ho és en DATA, i en aquest cas, hem de tenir en compte que potser tenim la data de finalització d'una franja definida, o potser tenim aquesta data de finalització amb valor NULL (el que vol dir que aquella franja és vàlida per qualsevol data a partir de la data_inici). Fent un esquema similar a l'anterior, pel cas de les dates, tindríem el següent : Al nou esquema, només mostrem les franges que han donat problema de solapament per hora, la resta ja no ens preocuparan.

Cas 1 : Intentem afegir la nova franja a partir del 01/02/2005 i fins el 01/03/2005



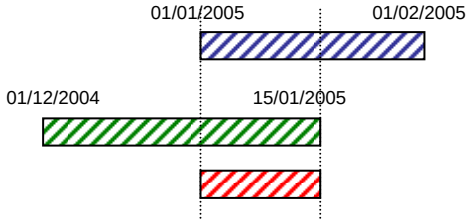
➔ Hi ha solapament

Cas 2 : Intentem afegir la nova franja a partir del 01/12/2004 sense data fi



➔ Hi ha solapament

Cas 3 : Suposem que el registre que teníem introduït finalitza el 01/02/2005
Intentem afegir la nova franja a partir del 01/12/2004 i fins el 15/01/2005
(és el mateix cas que si intentem afegir una franja sense data final)



➔ Hi ha solapament

Per tant : Per decidir que una franja horària es pot introduir, s'ha de comprovar la no coincidència en DATA i en HORA simultàniament. Si només coincideix en un dels dos aspectes, no hi ha solapament real (perquè és en dates diferents, o en hores diferents).

5.4.6.2 Programació de visites, comprovació d'hores disponibles

Ara bé, pel **cas d'introduir una visita** d'un pacient amb un determinat metge, la comprovació que s'ha de fer és en dos sentits :

1 : Comprovar que la franja horària de la visita (*) que es vol introduir estigui dintre de les franges horàries de treball d'aquell metge. En cas de produir-se aquest error, no

podrem programar la visita, i el procediment tornarà un error indicant que el metge no treballa en les hores en les que es tracta de programar aquesta visita.

(*) Per calcular la franja horària de la visita :
- Tenim la hora d'inici de la visita
- Ens poden indicar la durada programada per aquella visita
- Si no indiquen la durada programada, haurem de comprovar quina és la durada per defecte que tenen les visites d'aquell metge
- Obtindrem la hora fi de la visita sumant la hora inici + durada

2 : Comprovar que aquell metge no tingui ja assignada cap visita en aquella franja horària. En el cas de produir-se aquest error, no podrem programar la visita, i el procediment tornarà un error indicant que el metge ja té una visita assignada en aquest període horari.

5.4.6.3 Format dels camps d'hores de visites i durades de visites

El sistema més eficient que hem trobat per gestionar aquests càlculs de solapaments, és treballar amb les hores en format numèric decimal i no amb el sistema sexagesimal.

És a dir, per passar d'una hora en format sexagesimal a format decimal, el que farem és :

hores_en_decimal = hores_en_sexagesimal;
minuts_en_decimal = minuts_en_sexagesimal / 60;
hora_format_decimal = hores_en_ decimal +minuts_en_decimal;

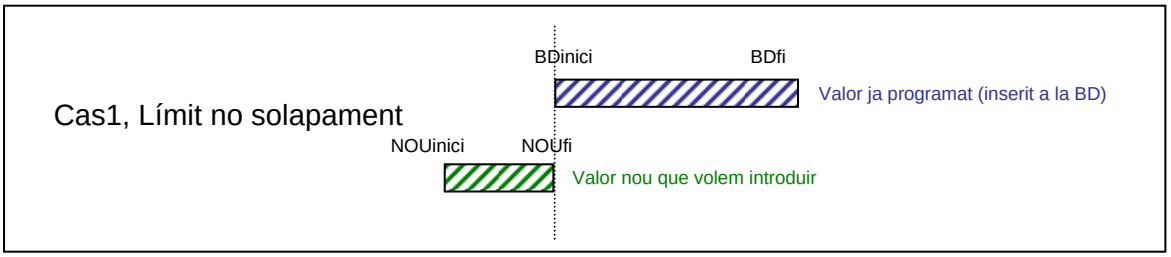
Si tenim 19:30, la conversió ens donarà 19,5
Si tenim 19:15, la conversió ens donarà 19,25
Si tenim 19:45, la conversió ens donarà 19,75
(evidentment, la resta de conversions, proporcionalment)

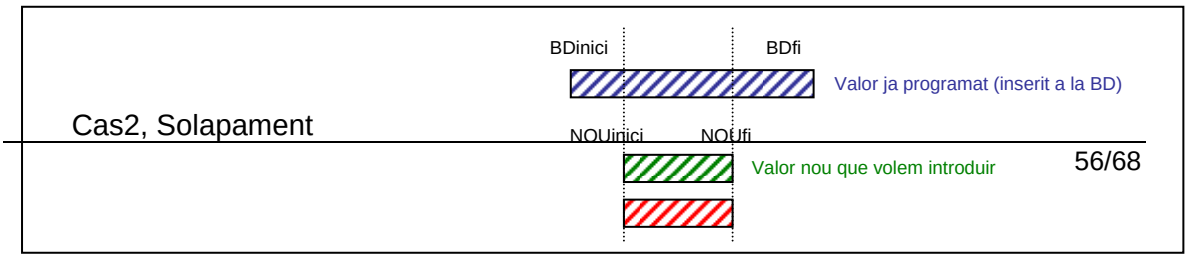
La conversió del format decimal al sexagesimal és la funció inversa (* 60).

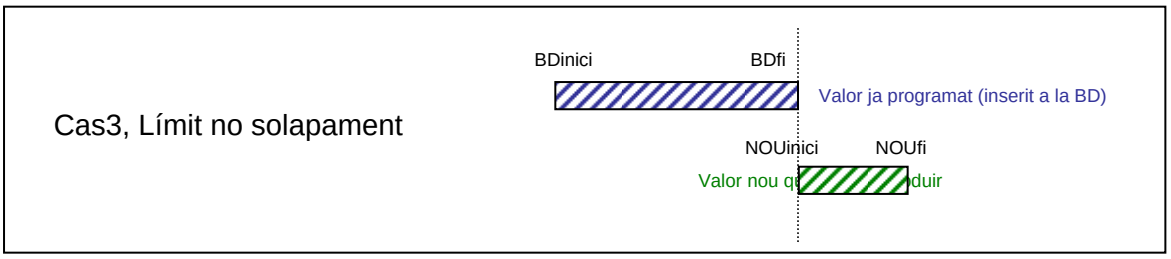
La principal avantatge de treballar així és que amb aquest format decimal podem sumar hores amb hores, restar hores d'hores, sumar minuts a hores (per calcular la hora_fi a partir de l'hora_inici + duració com hem comentat abans), sense haver-nos de preocupar en fer el càlcul en sexagesimal (i quan el valor dels segons sigui igual o superior a 60, fer operacions de divisió entre 60, mòdul, etc, etc).

Les dades es guardaran en format decimal a la base de dades, i treballarem en tots els procediments en aquest format. Serà responsabilitat de l'aplicació que accedeixi a aquestes dades transformar-les a format sexagesimal quan consulti les dades per presentar-les a l'usuari (si les consulta per fer càlculs, podrà aprofitar aquest sistema).

D'aquesta manera, simplifiquem el càlcul de solapaments.. hem de pensar en aquests casos (tant per dates com per hores) :







Del que es dedueix que tenim solapament en aquest cas :

Quan $(NOU_{FI} > BD_{INI})$ AND $(NOU_{INI} < BD_{FI})$

5.4.7 Es demana un camp usuari per executar cada procediment

A tots els procediments, com s'ha comentat, el segon paràmetre que es demana és l'identificador de l'usuari que està executant aquest procediment, per poder deixar constància a la taula de Registre d'activitat quin usuari ha executat aquell procediment.

Hem posat la restricció que només els usuaris que son personal del centre puguin executar els nostres procediments (no tindria sentit que un usuari pacient fes l'execució d'un d'aquests procediments). Els metges, a més de metges, son personal del centre, així que ells també podran executar els procediments (per exemple, per afegir medicaments, diagnòstics, casos, etc, etc).

Aquest sistema presenta un problema : Quan el sistema està acabat d'inicialitzar, no tenim cap usuari a la taula de Personal del centre, llavors, seguint aquesta política, no es podria executar mai cap procediment (ni tant sols el de donar d'alta a una persona del centre, perquè aquest procediment només el pot executar una persona JA donada d'alta com a personal del centre).

La solució per la que hem optat ha estat la de fer una excepció a aquesta regla que havíem establert :

> A la funció AltaPersonalCentre, si quan s'executa no hi ha CAP registre a la taula PersonalCentre, es permetrà fer l'inserció, sense fer cap comprovació sobre l'identificador de la persona que està executant aquesta funció.

Aquesta excepció es fa només en aquesta funció, de manera que la única funció que es podrà executar amb un usuari que no sigui personal del centre, serà, per una sola vegada, al inicialitzar el sistema, la de donar d'alta al primer usuari Personal del Centre.

5.4.8 Funcions per donar d'alta usuaris

Per tractar les altes d'usuaris tenim tres funcions :

> AltaPacient ; AltaPersonalCentre ; AltaMetge

Aquestes tres funcions, si s'executen per primera vegada per a un determinat usuari, donen d'alta a l'usuari a totes les taules que l'hagin de donar d'alta, és a dir, si executem AltaMetge, aquesta funció donarà d'alta a l'usuari a la taula Persones, a la taula PersonesCentre i també a la taula Metges.

En canvi, imaginem que tenim un usuari amb Dni X, que es dona d'alta com a pacient. Amb aquesta primera inserció, se'l donarà d'alta a les taules Persones i Pacients. Si ara introduïm aquest mateix usuari com a Metge, executarem la funció AltaMetge, però ara ja no se l'ha de donar d'alta a la taula Persones, ja que ja està donat d'alta de quan va ser donat d'alta com a pacient. En aquest cas, el que faria la funció seria : Actualitzar les dades de la taula Persones (ja que entenem que les dades que ens donin ara son més actuals), se'l dona d'alta a la taula PersonalCentre, i finalment se'l dona d'alta a la taula Metges.

5.4.9 Package i funcions

Hem creat una llibreria de funcions (un Package), per funcions que fèiem servir repetidament als procediments emmagatzemats. Aquesta llibreria s'anomena libtfc.

A continuació mostraré la signatura de cadascuna d'aquestes tres funcions i comentaré breument la seva funció :

```
function esPersonalCentre (pDni PersonalCentre.dni%type) return boolean;
```

Indiquem un DNI i ens retorna un booleà indicant si l'usuari és PersonalCentre o no.

```
function retornaRSP (pSqlErr varchar2) return varchar2;
```

Indiquem un identificador d'error per paràmetre i ens retorna el missatge d'error que ha de retornar el nostre Procedure. (aquesta funció s'explica amb més detall al següent punt d'aquesta memòria, amb el control d'excepcions).

```
procedure registrelog (pDni Reg.dni_usuari%type, pNom_procediment
Reg.nom_procediment%type, pValor_param_entrada Reg.valor_param_entrada%type,
pValor_param_sortida Reg.valor_param_sortida%type, pResultat Reg.resultat%type, pTraca
Reg.traca%type, pErrcodi Reg.errcodi%type, pErrdesc Reg.errdesc%type);
```

Aquesta funció insereix a la taula de registre (taula Reg) una entrada amb els valors dels paràmetres passats (veure apartat Sistema de LOG d'aquesta memòria per comprendre els paràmetres).

El codi d'aquestes funcions s'entrega juntament amb el codi de la resta del projecte a l'script de creació de tots els elements de la Base de Dades.

5.4.10 Sistema pel control d'excepcions

A cada procediment que creem, capturarem diferents tipus d'excepcions, les primeres seran les que generem nosaltres mateixos quan detectem un error a les dades que ens passa l'aplicació per paràmetre, i la resta, seran excepcions que genera el sistema (Oracle) per casos no controlats per nosaltres.

En el meu cas, tots els errors de restriccions indicades a la creació de les taules ("Constraints" (Restriccions) per Clau Primària, Clau Forana, Check formats, etc), deixo que saltin (és a dir, intento fer l'inserció encara que els paràmetres siguin incorrectes), i al final de la funció detecto quina excepció s'ha generat, i identifico la restricció que s'ha violat (ja que la variable SqlErrm conté el nom que hem assignat nosaltres a aquesta restricció).

He afegit una nova taula al sistema, anomenada ErrorsConsultori, on registro tots els errors que tinc previst que poden succeir. En aquesta taula tenim tres camps : Codi ; Identificador ; Descripció. El primer és un codi numèric, el segon és un text clau que identifica l'error, i el tercer és un text comprensible per l'usuari, per indicar-li quin error ha generat la execució que ha demanat al nostre procediment.

Els Identificadors poden ser d'aquests tipus :

PK_Nom → Restricció de clau primària
FK_Nom → Restricció de clau forana
CHK_Nom → Restricció per comprovació de format (Check)
INT_Nom → Comprovació interna de dades a un procediment
WRN_Nom → Warning (mirar valors de sortida RSP)
ORA-Codi → Excepció d'oracle

Quan es captura una excepció, el que fem és : cridar a la funció retornaRSP (del package libtfc) passant-li com a paràmetre la variable SqlErrm, que conté el text que Oracle ens retorna com explicació d'aquest error. A continuació mostrem tres missatges d'error que ens pot retornar Oracle :

```
ORA-00001: restricción única (ADMCONSULTORI.PK_PERSONALCENTRE) violada
ORA-01400: no se puede realizar una inserción NULL en ("ADMCONSULTORI"."PERSONES"."NOM")
ORA-01401: valor introducido demasiado largo para columna
```

Ens interessa poder personalitzar aquests missatges d'error, i per exemple, en el cas del primer, donar un missatge personalitzat, com :

"S'intenta donar d''alta a una persona al centre que ja existeix"

Es comprova si els identificadors d'error que tenim a la nostre taula ErrorsConsultori estan inclosos en aquest missatge d'error que ens indica Oracle (els identificadors de les restriccions creades per nosaltres sempre apareixen).

Amb aquest sistema, també podem personalitzar errors d'oracle (identificant-los per ORA-#####). Inserim aquests valors a la nostre taula, i introduïm la descripció que volem donar als usuaris en aquests casos, per exemple :

ORA-01400 → Descripció : S'ha deixat en blanc un camp obligatori
ORA-01401 → Descripció : Un dels camps excedeix la longitud permesa

Per casos que nosaltres no haguem controlat, es retornarà el missatge d'error que ha donat Oracle sense fer-li cap modificació.

En cas de detectar un error, s'executa la comanda "rollback" i s'elimina qualsevol canvi que s'hagi fet durant el procediment que s'acaba d'executar.

5.4.11 Valors de resultats de sortida dels procediments (RSP)

Tots els procediments tenen com a primer paràmetre un paràmetre anomenat pRSP, amb el modificador OUT, indicant que es tracta d'un paràmetre de només sortida.

Aquest paràmetre té com a finalitat informar a l'usuari de l'èxit o el fracàs de l'execució del procediment que ha cridat.

El valor que retornarem mitjançant aquest paràmetre començarà sempre per un d'aquests tres prefixes :

OK ; WARNING ; ERROR

OK → Es retorna només OK, l'execució ha estat correcta

ERROR-CODI-Descripció →

S'ha produït algun error i no s'ha fet cap canvi a les dades de la base de dades.

CODI és un codi de 4 dígits que indica el codi d'error, si és que és un error controlat a la nostra taula d'errors. Si és un error que nosaltres no hem controlat, el codi retornat és 0000

Descripció : en cas de ser un error controlat per la nostre taula d'errors, es tornarà la descripció que haguem assignat a l'error. En cas de ser un error no controlat per nosaltres, es tornarà (després del codi 0000-) el valor sense modificar de l'error que Oracle ens ha indicat a la variable SqlErrm.

WARNING-CODI-Descripcio →

L'execució s'ha portat a terme, les dades han estat introduïdes a la base de dades, però s'ha introduït una dada que es desconeixia abans, i s'ha de tenir en compte que això ha pogut afectar a casos que s'haurien de revisar.

Per exemple, tenim un pacient amb una al·lèrgia determinada A, i alguna vegada li hem administrat un medicament M. Si més endavant introduïm que el medicament M no es pot subministrat a pacients amb l'al·lèrgia A (taula ContraIndicacions), haurem d'avisar que s'ha de tenir en compte que en el passat s'ha subministrat aquest medicament que no es devia. A partir d'haver introduït la contraindicació, el sistema ja no deixarà subministrar aquell medicament a aquell pacient.

5.4.12 Sistema de LOG

Tal com es va explicar al document de Disseny, s'ha implementat una taula de registre d'activitat (taula Reg), que emmagatzema un registre per cada crida que es fa a un procediment.

La idea d'aquesta taula és la de guardar tota l'activitat que s'ha produït a les nostres taules (ja que totes les insercions/modificacions/eliminacions) que es faran a les taules de dades es faran a través dels nostres procediments.

Amb aquesta taula hem de ser capaços de poder reproduir les condicions que han portat al nostre sistema a un estat determinat, per poder analitzar, en cas d'algun error concret, que és el que ha passat per que el sistema hagi quedat en aquell estat, o hagi donat aquell error.

En aquesta taula emmagatzemarem :

Codi	→ Identificador de l'entrada del registre (Autonumèric)
Hora	→ Data i hora del moment de finalització de l'execució
DNI_Usuari	→ Dni de l'usuari que ha executat la funció
Nom_Procediment	→ Nom del procediment que s'ha cridat
Valor_Param_In	→ Valors de tots els paràmetres en el moment d'entrada (les 2 primeres variables, pRSP i pUser excloses)
Valor_Param_Out	→ Valors de tots els paràmetres en el moment de sortida (les 2 primeres variables, pRSP i pUser excloses)
Resultat	→ Valor del paràmetre pRSP al sortir del procediment
Traca	→ Traça de la funció, número que ens ajudarà a seguir el punt exacte on ha fallat la funció en cas d'error
ErrCodi	→ En cas d'error d'oracle, codi d'error
ErrDesc	→ En cas d'error d'oracle, descripció Oracle de l'error

5.4.13 Comparacions : Case Insensitive

Totes les comparacions que es fan entre paràmetres rebuts als procediments i valors de les taules de la base de dades, es fan sense tenir en compte majúscules i minúscules. Per aconseguir això, tenim les funcions Upper (o Lower).. el que fem, per comparar dos valors, és comparar-los d'aquesta manera :

Upper(valor1) = Upper(valor2)

De manera que, encara que valor1 i valor2 tinguin lletres en majúscules o minúscules diferents en els dos casos, si les cadenes son iguals (sense tenir en compte les majúscules/minúscules), la funció retorni que son iguals.

5.5 Scripts de creació de taules

El primer punt de la implementació és la creació de les taules. Seguirem el document de Disseny, l'apartat Definició de les taules i camps, on vam definir cada taula amb tots els seus camps, indicant nom, tipus i comentaris dels camps.

També repassarem el punt sobre Controls d'integritat, per implementar en aquest pas de creació de taules, tots els controls que corresponguin (com restriccions per valors de camps).

Els scripts de creació de taules estan inclosos a l'script 2 comentat al punt 5.4.4 d'aquesta memòria.

5.5.1 Tipus de dades fets servir pels camps de les taules

A continuació presentem una taula resum on s'indica el tipus de dades que s'ha fet servir pels camps que més es repeteixen a les taules del sistema :

Concepte	Tipus de Dades
dnis	CHAR(8)
codi_postal	CHAR(5)
telèfons	VARCHAR2(15)
dates	DATE
identificadors	INTEGER
temps comptats en minuts	NUMBER(3)
horaris (per indicar franges horàries)	NUMBER(4,2) → 4 dígits dels quals 2 decimals
preu (Euros)	NUMBER(7,2) → 7 dígits dels quals 2 decimals
foto	BLOB
texts	VARCHAR2(x)
texts llargs	VARCHAR2(2000) (ó 4000)

5.5.2 Llistat de les taules del sistema

A continuació presentem una llista de les taules que formen el nostre sistema. Els detalls sobre els camps es poden veure a l'apartat de Disseny o bé als scripts de creació de taules :

Alergies, AlergiesPacients, Casos, Contraindicacions, Diagnòstics, ErrorsConsultori, Especialitats, ExcepcionsHorari, Horaris, Medicaments, MedicamentsIncompatibles, Metges, Pacients, PersonalCentre, Persones, Receptes, Reg, Visites

Presentem una mostra del codi per crear una taula (la taula de Visites), on es poden veure les comandes per crear la Clau Primària, Claus foranes i restriccions :

```
CREATE TABLE Visites (
        id_visita INTEGER,
        id_cas INTEGER NOT NULL,
        dni_metge_extra CHAR(8),
        data_visita_programada DATE,
        durada_programada NUMBER(4,2),
        hora_inici_programada NUMBER(4,2),
        data_visita DATE,
        hora_inici NUMBER(4,2),
        durada NUMBER(4,2),
        estat_visita CHAR(1) NOT NULL,
        preu NUMBER(7,2),
        comentari VARCHAR2(2000),
        CONSTRAINT PK_Visites PRIMARY KEY (id_visita),
        CONSTRAINT FK_Visites_id_cas FOREIGN KEY (id_cas) REFERENCES Casos(id_cas),
        CONSTRAINT FK_Visites_dni_metge_extra FOREIGN KEY (dni_metge_extra) REFERENCES Metges(dni),
        CONSTRAINT CHK_Visites_estat_visita CHECK (estat_visita IN ('P','A','R'))
);
```

5.6 Implementació dels controls d'integritat

A l'apartat de disseny de la memòria es van indicar els punts que s'havien de tenir en compte en quant a la integritat de dades. Hi ha aspectes que ja els hem controlat als scripts de creació de les taules (aquells que es podien controlar mitjançant claus primàries, claus foranes o controls CHECK).

A continuació farem un breu repàs dels punts de control que vam indicar que s'havien de tenir en compte, i indicarem si ja han estat resolts a l'apartat de la creació de les taules, o si estan pendents de ser controlats, i com ho farem.

Taula Persones
dni: només caràcters amb valor 0-9 → **Pendent (Trigger)**
lletra_NIF : només caràcters amb valor A-Z → Fet (Check)
sexe : només valors M ó H → Fet (Check)

Taula Pacients
data alta <= data baixa → Fet (Check)

Taula PersonalCentre
data alta <= data baixa Fet (Check)

Taula Metges
durada visita > 0 → Fet (Check)

Taula Horaris
dia_setmana : valors limitats entre 1 i 7 → Fet (Check)
hora_inici < hora_fi → Fet (Check)
data_alta <= data_baixa → Fet (Check)
comprovar que no s'estiguin definint pel mateix metge dos franges horàries que se solapin
→ Comprovat mitjançant codi als Stored Procedures

Taula Medicaments
data_baixa >= data_alta → Fet (Check)

Taula Casos
data_fi >= data_inici
estat : valors tancats per uns valors → Fet (Check)

Taula Visites
estat_visita : valors tancats per uns valors → Fet (Check)

Taula Errors
id_error : comprovar que cap identificador d'error formi part del nom d'identificador d'un altre error → **Pendent (Trigger)**

5.6.1 Implementació de Triggers

Com hem repassat en el punt anterior, ens resten unes comprovacions d'integritat que haurem d'implementar mitjançant una eina que ofereixen alguns SGBD (Oracle els ofereix).

Les tres comprovacions que hem de fer son les que indiquem a continuació. Igual que en el cas anterior, amb els fitxers annexes a la memòria s'entrega un fitxer anomenat CreacioTriggers.sql

Taula Persones
dni: només caràcters amb valor 0-9

Taula Errors
id_error : comprovar que cap identificador d'error formi part del nom d'identificador d'un altre error

Els Triggers s'entreguen també als Annexes, a l'script 2 (script de creació de tota l'estructura del sistema). Aquest és el codi d'un dels dos Triggers (el primer, sobre la taula Perones, que comprova que els caràcters del dni siguin només caràcters numèrics) :

```
CREATE OR REPLACE TRIGGER TR_Persones
    -- Aquest Trigger comprovarà que al camp DNI només s'introduiexin caracters numèrics
    BEFORE INSERT OR UPDATE
    ON Persones
    REFERENCING NEW AS newRow
    FOR EACH ROW
    BEGIN
         IF (TO_NUMBER(:newRow.dni)<0) THEN
            RAISE_APPLICATION_ERROR (-20000, 'TR_Persones - Només s''accepten c numèrics per al camp DNI');
         END IF;

       EXCEPTION
          WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR (-20000, 'TR_Persones - Només s''accepten c numèrics per al camp DNI');
    END;
/
```

5.7 Scripts de creació d'índexs

A l'etapa de disseny es van proposar 16 camps per a ser indexats. Aquests camps son les Claus Foranes (ja que farem servir aquests camps quan relacionem taules per a fer consultes), i altres camps que es va considerar que es farien servir sovint al fer consultes, per filtrar valors o bé per ordenar els resultats.

Aquest és el codi que hem d'executar per tal de crear un índex a un camp d'una taula :

```
CREATE INDEX C1 ON CASOS (dni_pacient);
```

Aquests son els camps que s'han indexat :

VISITES.id_cas
METGES.id_especialitat
CASOS.dni_metge
EXCEPCIONSHORARI.dni
EXCEPCIONSHORARI.hora_inici
PERSONALCENTRE.username
DIAGNOSTICS.nom
PERSONES.cognoms
VISITES.dni_metge_extra
CASOS.dni_pacient
CASOS.id_diagnòstic
EXCEPCIONSHORARI.dia_setmana
EXCEPCIONSHORARI.data_alta
MEDICAMENTS.nom
PERSONES.nom
ErrorsConsultori.id_error

5.8 Scripts de creació de Seqüències

Aquestes son les 7 seqüències que hem hagut de crear per emular el funcionament dels camps autonumèrics :

sEspecialitats, sAlergies, sMedicaments, sDiagnostics, sCasos, sVisites, sReg

El codi per crear una seqüència en Oracle és el següent :

```
CREATE SEQUENCE sEspecialitats
INCREMENT BY 1 START WITH 1
MAXVALUE 99999999 MINVALUE 1
CYCLE NOCACHE;
```

Després, per obtenir el valor següent de la seqüència des del codi PL/SQL, als Stored Procedures, haurem de fer servir la següent consulta :

```
SELECT nomseqüència.NEXTVAL
INTO variable
FROM DUAL;
```

5.9 Implementació dels procediments

No inclourem el codi dels procediments en la memòria degut a la seva extensió. Inclourem una mostra amb el codi d'un procediment d'exemple i, en l'apartat següent, la signatura de tots els procediments.

Els procediments es troben al fitxer adjunt que s'entrega amb la memòria : Script 2. A la memòria incloem només el codi del procediment AltaMedicament a mode d'exemple. Als annexes es troben els Scripts Complerts amb el codi de totes les funcions que s'han desenvolupat.

5.9.1 Codi d'exemple del procediment AltaMedicament

```
create or replace procedure AltaMedicament
         (pRsp OUT VARCHAR2, pUser PERSONALCENTRE.DNI%type, pNom MEDICAMENTS.Nom%type,
          pPro MEDICAMENTS.Prospecte%type, pId OUT MEDICAMENTS.Id_Medicament%type) is

        parametresin REG.valor_param_entrada%type;
        parametresout REG.valor_param_sortida%type;
        vtrace INTEGER;
        exc_user EXCEPTION;
        exc_jaexisteix EXCEPTION;
        comptador INTEGER;
        nomprocediment VARCHAR(30);

begin
  vtrace:=0;
  nomprocediment:='AltaMedicament';

  parametresin:=pNom||'Þ'||pPro||'Þ'||pId;

  -- Comprovo que l'usuari pUsuari sigui personal del centre i no estigui donat de baixa
  IF (NOT libtfc.esPersonalCentre(pUser)) THEN
     vtrace:=10;
     -- L'usuari no té permís per inserir aquest registre
     RAISE exc_user;
  END IF;
  vtrace:=20;

  -- Comprovo que no hi hagi ja un registre amb aquest mateix nom
  SELECT COUNT(*)
  INTO comptador
  FROM Medicaments
  WHERE Upper(nom)=Upper(pNom);

  IF (comptador <> 0) THEN
     vtrace:=30;
     -- Ja existeix un registre amb aquest nom
     raise exc_jaexisteix;
  ELSE
      vtrace:=40;
      SELECT sMedicaments.NEXTVAL
      INTO comptador
      FROM DUAL;

      vtrace:=50;
      INSERT INTO Medicaments (id_medicament, nom, prospecte, data_alta)
                       VALUES (comptador, pNom, pPro, sysdate);

      pId:= comptador;
      -- Ok, l'insertem a la BD
  END IF;

  vtrace:=1000;
  pRsp:='OK';
  parametresout:=pNom||'Þ'||pPro||'Þ'||pId;
  libtfc.registrelog (pUser, nomprocediment, parametresin, parametresout, pRsp,
                          vtrace, null, null);

  commit;

  EXCEPTION
     WHEN exc_user THEN
          rollback;
          pRsp:=libtfc.retornaRSP('INT_User');
          parametresout:=pNom||'Þ'||pPro||'Þ'||pId;
          libtfc.registrelog (pUser, nomprocediment, parametresin, parametresout,
                                  pRsp, vtrace, null, null);
          commit;

     WHEN exc_jaexisteix THEN
          rollback;
          pRsp:=libtfc.retornaRSP('INT_JaExisteix');
          parametresout:=pNom||'Þ'||pPro||'Þ'||pId;
          libtfc.registrelog (pUser, nomprocediment, parametresin, parametresout,
                                  pRsp, vtrace, null, null);
          commit;

     WHEN OTHERS THEN
          rollback;
          pRsp:=libtfc.retornaRSP(SqlErrm);
          parametresout:=pNom||'Þ'||pPro||'Þ'||pId;
          libtfc.registrelog (pUser, nomprocediment, parametresin, parametresout,
                                  pRsp, vtrace, SqlCode, SqlErrm);
          commit;


end AltaMedicament;
```

5.10 Variacions entre Anàlisis i Implementació

S'han produït alguns canvis entre el que es va preveure a l'etapa d'anàlisis i el que finalment s'ha acabat implementant a l'etapa de implementació.

Hi ha algunes variacions per diversos motius, com per exemple : s'han hagut de canviar alguns noms de taules o paràmetres de procediments, ja que en l'etapa d'anàlisis es van posar noms que son paraules restringides d'Oracle, com per exemple, la taula de registre d'activitat. S'havia indicat que es diria Log, i finalment s'ha hagut de canviar el nom i l'he anomenat Reg.

Hi ha hagut altres canvis, com canvis en tipus de dades, ja que al fer la implementació he vist que per algunes dades anava millor un altre tipus de camp, o bé perquè per que ha canviat d'alguna manera la forma en la que pensàvem emmagatzemar les dades (com per exemple, als camps d'hores de visita i duracions de la visita, que finalment es va decidir canviar el tipus i es va fer servir el format decimal en comptes del sexagesimal per tractar les dades d'horaris.

A alguna taula també varem oblidar algun camp important, com per exemple, a la taula visites no indicàvem l'hora d'inici de la visita (l'hora d'inici prevista i l'hora d'inici real, igual que sí que fem amb les dates).

Degut a canvis com aquest últim que hem comentat, o a d'altres, també s'han hagut de modificar lleugerament els paràmetres d'algun dels procediments.

Excepcions Horaris : A l'etapa d'anàlisis es va dissenyar un sistema de tractament de les excepcions d'horaris molt sofisticat. Era un sistema flexible que permetia qualsevol tipus d'entrada d'horaris i de modificacions d'horaris. La implementació d'aquesta funcionalitat al complert complicava molt el codi de la pràctica, i la implementació del sistema com s'havia dissenyat hauria fet impossible acabar aquest projecte amb les hores que requeria i que s'havia planificat, així que vaig comentar a l'entrega de la implementació, que aquesta part no estaria a l'entrega final. Es poden planificar hores de visita amb la taula d'horaris, però no es pot treballar amb les funcionalitats d'excepcions d'horaris que s'havien comentat a l'etapa de disseny.

6 Proves

Amb el codi que s'entrega, tal i com s'ha explicat al punt 5.4.4. Scripts, s'inclou un joc de proves de totes les funcions que s'han implementat. Aquest joc de proves està separat en tres fases (s'executen totes seguides, una a continuació de l'altre).

El joc de proves està pensat per ser executat amb la base de dades buida (ja que se suposa que les seqüències començaran a comptar des de 1, i això es fa servir per introduir dades tenint en compte que els valors que s'assignaran a alguns identificadors son aquests).

Les tres fases son les següents :

a) OK : Funcions que han de donar resultat correcte
Insercions, modificacions i eliminacions de dades correctes per omplir la base de dades

b) WARNING : Funcions que han de donar com a resultat un Warning
Funcions que s'executaran, però que donaran com a resultat (RSP) un missatge d'avís (per exemple, si s'introdueix una incompatibilitat entre medicaments i s'adverteix que Aquests dos medicaments han estat receptats amb anterioritat a algun pacient)

c) ERROR : Funcions que han de donar error com a resultat
Aquestes funcions no s'executaran (es fa un rollback abans de sortir del procediment)
Es posen a prova les funcions amb els casos erronis que han de detectar

Entreguem un fitxer (resultat_execucio_jocproves.txt) que indica la sortida de l'execució de l'script, en el que també s'ha copiat el resultat d'una consulta SELECT sobre la taula de registre d'activitat per mostrar els canvis produits per aquest joc de proves sobre el sistema (En aquest resultat, per que les dades siguin més llegibles, s'han copiat només 4 dels 10 camps de la taula de registre).

Si ens interessa omplir la base de dades amb dades correctes per tal de fer consultes de prova, o analitzar altres casos, es recomana comentar les línies del Joc de Proves corresponents a eliminacions i modificacions (des de la última part de la fase de proves OK del joc de proves, on es comença amb les modificacions, fins el final de l'script).

7 Conclusions

El primer que comentarem és que els objectius plantejats a l'inici del projecte han estat complerts amb èxit : S'ha aconseguit desenvolupar un sistema per controlar les dades d'un consultori mèdic.

S'han treballat a fons tots els conceptes de bases de dades que he après a les assignatures de Bases de Dades I i II (taules, consultes, índexs, triggers, ..., ...), i aspectes generals de programació apresos a altres assignatures, i he hagut de documentar-me sobre molts altres temes per poder desenvolupar aquest projecte, com : programació amb el llenguatge PL/SQL, gestió d'excepcions, procediments emmagatzemats, llibreries de funcions en Oracle, entorn de programació PL/SQL Developer, etc.

Un aspecte molt positiu per a mi d'haver fet aquest projecte ha estat el de conèixer aquest sistema de treball, separant en una capa completament independent del programari final tot el relacionat amb la base de dades. El consultor també ens va remarcar molt la importància que tenia el fet de treballar amb un log detallat de tota l'activitat, i fent el desenvolupament que hem fet, i fent les proves que he anat fent durant tot el projecte, si que he vist que comptar amb l'ajuda d'un registre on s'especifiqui absolutament tot el que està passant al nostre sistema és de gran ajuda per solucionar errors que anem trobant. I, evidentment, de molta més ajuda quan un sistema com aquest es posi en explotació i les dades que s'introdueixin al sistema ja no són les que pensem nosaltres, sinó totes les que introdueixin els usuaris dia a dia.

A la part de la implementació s'han hagut de fer algunes variacions amb respecte al que s'havia planificat a l'etapa de disseny. Alguns d'irrellevants en quant al nivell de feina total, com canvis en algunes taules, en tipus de variables, en noms d'algunes variables, etc, i canvis que sí que han implicat una variació en el volum de feina, com ja s'ha comentat en el punt 5.10 de la memòria.

S'ha respectat el més fidelment possible el pla de treball que es va proposar a la primera entrega del projecte (Diagrama de Gantt).

El volum de treball d'aquest projecte és superior al que hauria d'haver estat, ja que una assignatura de 7,5 crèdits hauria de correspondre a unes 75 hores de feina, i les hores dedicades a aquest projecte, entre formació, implementació, i redactat de la memòria, han estat moltes més. Potser els objectius en el moment de preparar el pla de treball van ser massa ambiciosos.

La meva conclusió final és que el desenvolupament d'aquest projecte ha estat molt positiu per a mi, ja que amb ell he conegut una eina tant coneguda com és Oracle.