UNIVERSIDADE ESTADUAL DO CEARÁ
ESPECIALIZAÇÃO EM ENGEHARIA DE SOFTWARE COM ÊNFASE EM PADRÕES DE SOFTWARE

ERIVAN DE SENA RAMOS

PROJETO DE BANCO DE DADOS
SISTEMA DE CONTROLE DE TERMINAIS DE VENDAS

Fortaleza
2010

SUMÁRIO

1 APRESENTAÇÃO

Uma rede de supermercados deseja um sistema para controle de terminais de venda.

Cada supermercado possui diversos terminais de venda (TV); onde cada terminal é operado por um único caixa, em um dado período de tempo.

O caixa deve realizar o registro das compras de cada cliente, produto por produto, incluindo os pagamentos.

Os terminais são ligados a catálogos de produtos.

A gerência precisa de informações sobre: tempo de ociosidade de um terminal; faturamento mensal; quantidade de caixas que operaram o sistema; valor médio das vendas; produtos mais vendidos e faturamento total

2 LEVANTAMENTO DE REQUISITOS

3 MODELAGEM CONCEITUAL DOS DADOS

4 TRANSFORMAÇÃO DE ESQUEMA ER EM R

1.1.1 Lista de eventos

Evento	Tipo*	Estímulo	Ação	Resposta
Funcionário cadastra médico	F	dados-médico	Cadastrar médico	Médico cadastrado

Funcionário cadastra especialidade	F	dados-médico	Cadastrar especialidade	Especialidade cadastrada
Funcionário cadastra plano de saúde	F	dados-médico	Cadastrar plano saúde	Plano cadastrado
Funcionário cadastra pacientes	F	dados-paciente	Cadastrar Pacientes	Paciente cadastrado
Funcionário marca consultas	F	dados-consultas	Marcar consultas	Consulta marcada
Médico mantém observações	F	dados-consultas	Incluir e verificar obs.	Incluída/ verificada obs.
Direção solicita relatório das consultas realizadas todo mês	T	solicitação de relatório período	Gerar relatório de consultas	Relatório- cons. realizadas
Médico solicita Exames	F	dados-paciente	Marcar exames	Data do exame
Funcionário verifica autorização de consulta	F	dados-paciente	Receber autorização	Autorização realizada.
Médico prescreve Remédios	F	dados-consulta	Prescrever remédios	Remédio prescrito
Funcionário verifica as consultas do dia	F	dados-consulta	Visualizar consultas	Relação das consultas do dia
Funcionário verifica os exames com resultados	F	dados-exames	Visualizar exames com resultados	Relação dos exames com resultado
Funcionário consulta os telefones dos médicos	F	dados-médicos	Visualizar telefones dos médicos	Relação dos telefones dos médicos
Funcionário desvincula plano de saúde	C	dados-plano	Desvincular Plano de Saúde	Plano excluído / Cadastro paciente atualizado

***F – Evento orientado por fluxo**

T – Evento temporal

C – Evento de Controle

1.2 Levantamento de requisitos

1.2.1 Requisitos funcionais

RF_01 – Manter Médico

O sistema deve permitir realizar operações de consulta, inserção, atualização e exclusão dos dados (CRM, nome, endereço e telefone) dos médicos por meio de digitação nas aplicações de cadastro e consulta do sistema, realizada por usuário autenticado.

RF_02 – Manter Especialidade

O sistema deve permitir realizar operações de consulta, inserção, atualização e exclusão dos dados (código do CRM e nome) das especialidades, essa tarefa será realizada por usuário devidamente autorizado pela direção da clínica, por meio de digitação nas aplicações de cadastro e consulta do sistema.

RF_03 – Manter Plano de Saúde

O sistema deve permitir realizar operações de consulta, inserção, atualização e exclusão dos dados (CNPJ, nome, endereço e telefone) dos Planos de Saúde, ação desempenhada por usuário autorizado a realizar alterações nos cadastros de plano de saúde, por meio de digitação no sistema.

RF_04 – Manter Paciente

O sistema deve permitir realizar operações de consulta, inserção, atualização e exclusão dos dados (CPF, nome, endereço, profissão e telefone) dos pacientes, através de digitação nas aplicações de cadastro e consulta do sistema, realizada por usuário autenticado.

RF_05 – Manter Consulta

O sistema deve permitir realizar operações de verificação, inserção, atualização e exclusão dos dados das consultas: deve-se guardar a data e hora da consulta, essa tarefa será realizada por usuário devidamente autorizado, por meio de digitação nas aplicações de cadastro e consulta do sistema.

RF_06 – Manter Observações

O sistema deve permitir realizar operações de verificação, inserção, atualização e exclusão de observações das consultas, onde serão registrados textos sobre queixas do pacientes, resultados dos exames e respostas ao tratamento efetuado, essa tarefa será realizada pelo médico responsável pela consulta através de digitação nas aplicações de cadastro da consulta.

RF_07 – Relatório de Consultas

O sistema deve permitir emitir relatório das consultas por período, onde serão exibidos nomes dos pacientes, nomes dos médicos, datas e horários das consultas realizadas; essa tarefa será executada por funcionário autorizado pela Direção da clinica, por meio da aplicação que permitirá filtrar os dados de acordo com a necessidade da Direção.

RF_08 – Solicitar Exames

O sistema deve permitir realizar a solicitação de exames, onde o médico registra o nome do paciente e o nome no exame a ser realizado, através de digitação no sistema.

RF_09 – Prescrever Remédios

O sistema deve permitir realizar a prescrição de remédios de acordo com tipo de doença identificada para o paciente, nessa operação será registrado nome da doença e o nome do remédio a ser prescrito, por meio de digitação desses dados no cadastro do paciente.

RF_10 – Verificar Autorização

O sistema deve permitir realizar operações de verificação e confirmação da autorização da consulta para o paciente, onde será possível verificar a autenticidade do convênio do paciente, e posterior registro na aplicação de cadastro da consulta, essa tarefa será realizada por usuário autenticado através de digitação nas aplicações de cadastro da consulta.

RF_11 – Visualisar Exames com Resultados

O sistema deve permitir restrições para alguns usuários que somente poderão realizar operações de consulta ao verificar o cadastro dos exames, essa tarefa será realizada por usuário autenticado através da escolha de uma opção chamada “Exames com Resultados”. Onde serão relacionados apenas os exames que já possuem resultados, identificando o exame, a data, o resultado e nome do paciente.

RF_12 – Visualisar Consultas do Dia

O sistema deve permitir restrições para alguns usuários que somente poderão realizar operações de consulta ao verificar as consultas, essa tarefa será realizada por usuário autenticado através da escolha de uma opção chamada “Consultas Diárias”, onde estarão relacionadas apenas as consultas do dia com o nome do paciente, nome do médico, data e hora da consulta.

RF_13 – Visualisar Telefones dos Médicos

O sistema deve permitir restrições para alguns usuários que somente poderão realizar operações de consulta ao verificar o cadastro dos médicos, essa tarefa será realizada por usuário autenticado através da escolha de uma opção chamada “Contatos dos Médicos”, onde estarão relacionados somente CRM, especialidades, nomes e telefones dos médicos e de seus respectivos planos de saúde.

RF_14 – Desvincular Plano de Saúde

O sistema deve realizar controle dos cadastros pacientes nos quais os planos de saúde foram desvinculados da clinica. A operação de exclusão do plano de saúde do sistema deverá ser executada por usuário devidamente autenticado, e executará atualização nos cadastros dos pacientes pertencentes aos planos, recebendo uma observação que o plano do paciente foi desvinculado da clínica.

1.2.2 Requisitos não funcionais

RNF_01 – Padrão de Cadastros

Os cadastros a serem realizados no sistema deverão obedecer a um mesmo padrão de usabilidade. Permitindo:

a) Acessar diretamente ao registro pelo seu ID ou através de pesquisa avançada.

b) Operação de Inserir, Alterar, Excluir, Salvar, Cancelar o cadastro atual.

c) Operação de Fechar, permitindo a saída da tela de cadastro;

RNF_02 - Padronização dos Relatórios

Os relatórios a serem realizados no sistema deverão obedecer a um mesmo padrão, apresentando a logomarca da empresa, podendo ser fornecido em formato .xls, ou em .pdf .

RNF_03- Manual de Instalação

No ato da entrega do sistema, o mesmo deverá acompanhado com um manual de instalação em CD-R no formato .pdf e impresso.

RNF_04 - Manual de Usuário

No ato da entrega do sistema, o mesmo deverá acompanhado com um manual do usuário em CD-R no formato .pdf e impresso.

RNF_05 - Acesso

O banco de dados deverá armazenar os dados de acesso ao sistema. O acesso ao sistema deverá ser autenticado através do fornecimento de login e senhas criptografadas.

RNF_6 – Ambiente do sistema

O sistema deverá ser composto de 2 partes, com os seguintes requisitos de ambiente:

a) Aplicação Cliente: Windows Vista, com o Mozilla Firefox instalado;

b) Banco de Dados: Windows Vista, ou qualquer sistema operacional que suporte o PostGres.

RNF_7 – Hardware

Os requisitos para hardware são os seguintes:

a) Aplicação Cliente: Mínimo de 128mb de memória livre para a operação do sistema;

b) Banco de Dados: Os mesmos requisitos de hardware do PostGreSQL.

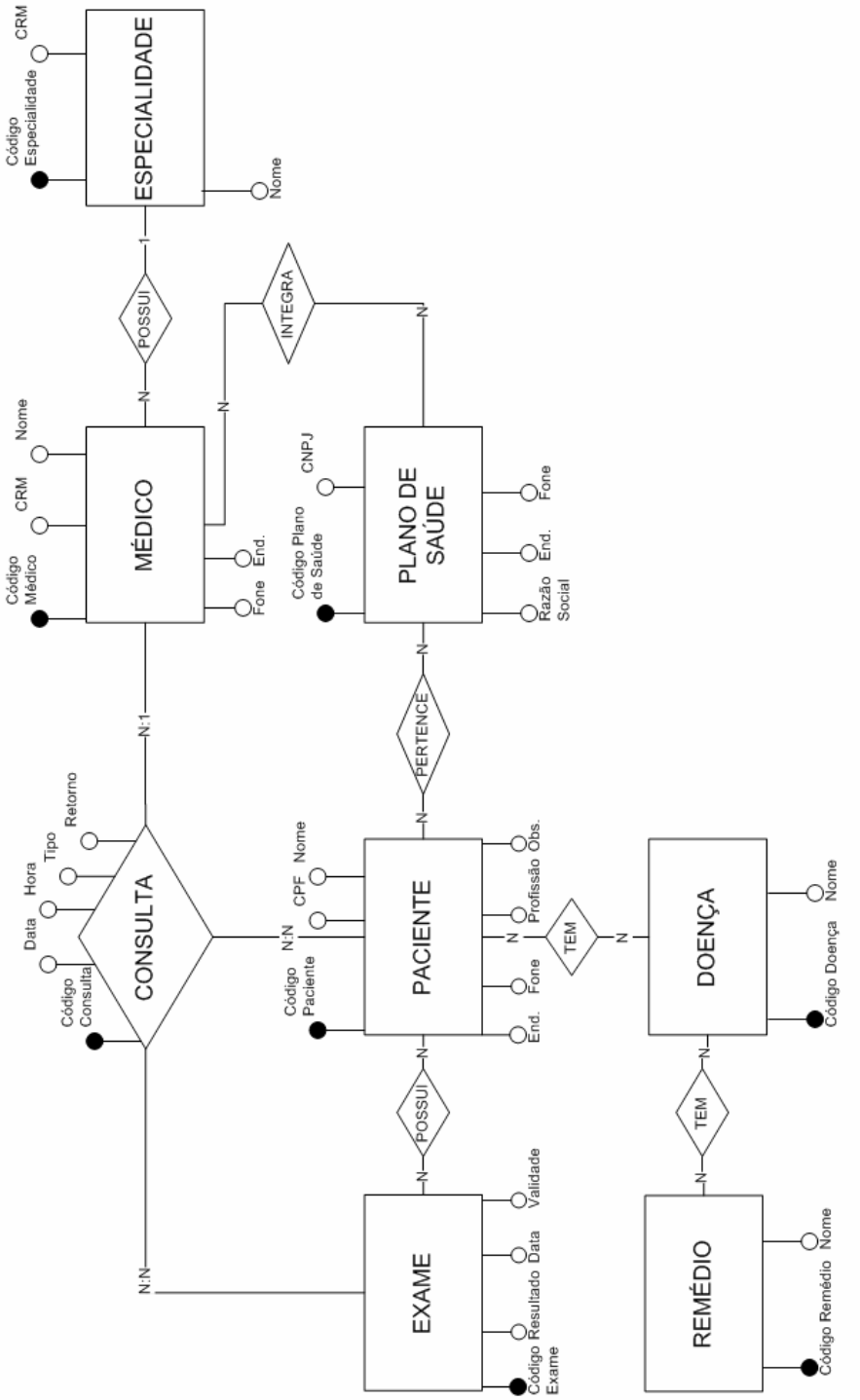
RNF_12 - Ferramentas de Desenvolvimento

O sistema será desenvolvido no Frameworking PHP/AJAX ScriptCase. Para banco de dados, será utilizado o PostGreSQL.

RNF_13 – Tamanho do Banco de Dados

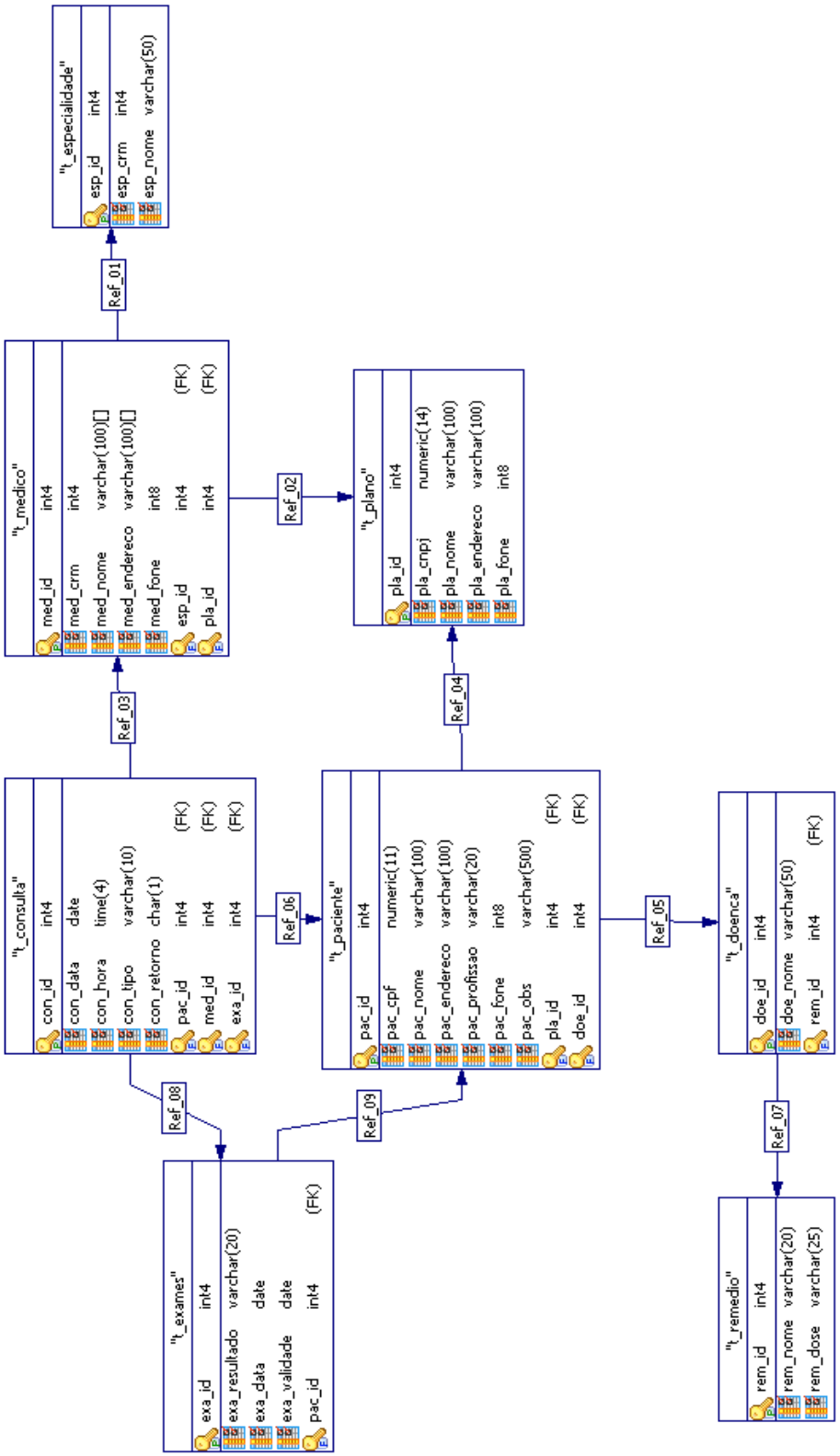
O Banco de Dados deverá possuir o tamanho estimado da ocupação de 16.018.530 kbytes, para o correto armazenamento dos dados inseridos no sistema.

1.3 Modelo Entidade-Relacionamento



2 PROJETO LÓGICO

2.1 Diagrama de Entidade-Relacionamento



2.2 Dicionário de Dados

Tabela t_especialidade

Atributo	Nulo?	Tipo	Tamanho
esp_id	Não	Inteiro	04
esp_crm	Não	Inteiro	04
esp_nome	Não	Varchar	50

Tabela t_medico

Atributo	Nulo?	Tipo	Tamanho
med_id	Não	Inteiro	04
med_crm	Não	Inteiro	04
med_nome	Não	Varchar	100
med_endereco	Não	Varchar	100
med_fone	Não	Inteiro	08

Tabela t_consulta

Atributo	Nulo?	Tipo	Tamanho
con_id	Não	Inteiro	04
con_data	Não	Date	08
con_hora	Não	Timetemp	04
con_tipo	Não	Varchar	10
con_retorno	Não	char	01

Tabela t_remedio

Atributo	Nulo?		Tamanho
rem_id	Não	Inteiro	04
rem_nome	Não	Varchar	20
rem_dose	Não	Varchar	25

Tabela t_doenca

Atributo	Nulo?	Tipo	Tamanho
doe_id	Não	Inteiro	04
doe_nome	Não	Varchar	50

Tabela t_paciente

Atributo	Nulo?	Tipo	Tamanho
pac_id	Não	Inteiro	04
pac_cpf	Não	Inteiro	11
pac_nome	Não	Varchar	100
pac_fone	Não	Inteiro	08
pac_endereco	Não	Varchar	100
pac_profissao	Sim	Varchar	20
pac_telefone	Não	Inteiro	20
pac_observacao	Sim	Varchar	500

Tabela t_plano

Atributo	Nulo?		Tamanho
pla_id	Não	Inteiro	04

pla_cnpj	Não	Inteiro	14
pla_nome	Não	Varchar	100
pla_endereco	Não	Varchar	100
pla_telefone	Não	Inteiro	08

Tabela t_exame

Atributo	Nulo?		Tamanho
exa_id	Não	Inteiro	04
exa_nome	Não	Varchar	100
exa_data	Sim	Date	08
exa_validade	Sim	Date	08

3 PROJETO FÍSICO

3.1 Estimativa de ocupação do Banco de Dados

Tabela t_especialidade

Atributo	Tamanho em kbytes
esp_id	04
esp_crm	04
esp_nome	50
Tamanho total atributos	58
Estimativa de registro	50
Tamanho total tabela	**2.900**

Tabela t_medico

Atributo	Tamanho em kbytes
med_id	04
med_crm	04
med_nome	100
med_endereco	100
med_fone	08
Tamanho total atributos	216
Estimativa de registro	100
Tamanho total tabela	**21.600**

Tabela t_consulta

Atributo	Tamanho em kbytes
con_id	04
con_data	08
con_hora	04
con_tipo	10
con_retorno	01
Tamanho total atributos	27
Estimativa de registro	5000
Tamanho total tabela	**135.000**

Tabela t_remedio

Atributo	Tamanho em kbytes
rem_id	04
rem_nome	20
rem_dose	25
Tamanho total atributos	49
Estimativa de registro	500
Tamanho total tabela	**24.500**

Tabela t_doenca

Atributo	Tamanho em kbytes
doe_id	04
doe_nome	50
Tamanho total atributos	54
Estimativa de registro	2000
Tamanho total tabela	**108.000**

Tabela t_paciente

Atributo	Tamanho em kbytes
pac_id	04
pac_cpf	11
pac_nome	100
pac_fone	08
pac_endereco	100
pac_profissao	20
pac_telefone	20
pac_observacao	500
Tamanho total atributos	753
Estimativa de registro	3.000
Tamanho total tabela	**2.259.000**

Tabela t_plano

Atributo	Tamanho em kbytes
pla_id	04

pla_cnpj	14
pla_nome	100
pla_endereco	100
pla_telefone	08
Tamanho total atributos	226
Estimativa de registro	50
Tamanho total tabela	**11.300**

Tabela t_exame

Atributo	Tamanho em kbytes
exa_id	04
exa_nome	100
exa_data	08
exa_validade	08
Tamanho total atributos	120
Estimativa de registro	10.000
Tamanho total tabela	**12.000.000**

Total Banco de Dados Clinica Médica

Tabela	*Tamanho em kbytes*
t_especialidade	2.900
t_medico	21.600
t_consulta	135.000
t_remedio	24.500
t_doenca	108.000
t_paciente	2.259.000
t_plano	11.300
t_exame	12.000.000
Tamanho total das tabelas	14.562.300
+10%	1.456.230
Estimativa da ocupação do Banco de Dados	**16.018.530**

3.2 Scripts

3.2.1 Criação das tabelas

```
-----------------------------------------
--CRIAÇÃO DO BANCO DE DADOS CLINICA MÉDICA--
-----------------------------------------

CREATE DATABASE clinica_medica;


-----------------------------------
--CRIAÇÃO DA TABELA ESPECIALIDADE--
```

```
-----------------------------------

CREATE TABLE "t_especialidade" (
      "esp_id" int4 NOT NULL,
      "esp_crm" int4 NOT NULL,
      "esp_nome" varchar(50) NOT NULL,
  PRIMARY KEY("esp_id")
);


-----------------------------------
--CRIAÇÃO DA TABELA PLANO DE SAÚDE--
-----------------------------------
CREATE TABLE "t_plano" (
      "pla_id" int4 NOT NULL,
      "pla_cnpj" numeric(14) NOT NULL,
      "pla_nome" varchar(100) NOT NULL,
      "pla_endereco" varchar(100) NOT NULL,
      "pla_fone" int8 NOT NULL,
  PRIMARY KEY("pla_id")
);









-----------------------------------
-----CRIAÇÃO DA TABELA MÉDICO------
-----------------------------------


CREATE TABLE "t_medico" (
      "med_id" int4 NOT NULL,
      "med_crm" int4 NOT NULL,
      "med_nome" varchar(100)[] NOT NULL,
      "med_endereco" varchar(100)[] NOT NULL,
      "med_fone" int8 NOT NULL,
      "esp_id" int4 NOT NULL,
      "pla_id" int4 NOT NULL,
  PRIMARY KEY("med_id"),
```

```
  CONSTRAINT "Ref_01" FOREIGN KEY ("esp_id")
    REFERENCES "t_especialidade"("esp_id")
      MATCH SIMPLE
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
      NOT DEFERRABLE,
  CONSTRAINT "Ref_02" FOREIGN KEY ("pla_id")
    REFERENCES "t_plano"("pla_id")
      MATCH SIMPLE
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
      NOT DEFERRABLE);
```

```
-----------------------------------
----CRIAÇÃO DA TABELA CONSULTA-----
-----------------------------------


CREATE TABLE "t_consulta" (
      "con_id" int4 NOT NULL,
      "con_data" date NOT NULL,
      "con_hora" time(4) NOT NULL,
      "con_tipo" varchar(10) NOT NULL,
      "con_retorno" char(1) NOT NULL,
      "pac_id" int4 NOT NULL,
      "med_id" int4 NOT NULL,
      "exa_id" int4 NOT NULL,
```

```
  PRIMARY KEY("con_id","con_data"),
  CONSTRAINT "Ref_06" FOREIGN KEY ("pac_id")
    REFERENCES "t_paciente"("pac_id")
      MATCH SIMPLE
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
      NOT DEFERRABLE,
  CONSTRAINT "Ref_03" FOREIGN KEY ("med_id")
    REFERENCES "t_medico"("med_id")
      MATCH SIMPLE
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
      NOT DEFERRABLE,
  CONSTRAINT "Ref_08" FOREIGN KEY ("exa_id")
    REFERENCES "t_exames"("exa_id")
      MATCH SIMPLE
      ON DELETE NO ACTION
      ON UPDATE NO ACTION NOT DEFERRABLE);
```

```
-----------------------------------
----CRIAÇÃO DA TABELA PACIENTE-----
-----------------------------------


CREATE TABLE "t_paciente" (
      "pac_id" int4 NOT NULL,
      "pac_cpf" numeric(11) NOT NULL,
      "pac_nome" varchar(100) NOT NULL,
      "pac_endereco" varchar(100) NOT NULL,
      "pac_profissao" varchar(20),
      "pac_fone" int8 NOT NULL,
      "pac_obs" varchar(500),
      "pla_id" int4 NOT NULL,
```

```
      "doe_id" int4 NOT NULL,
  PRIMARY KEY("pac_id"),
  CONSTRAINT "Ref_04" FOREIGN KEY ("pla_id")
    REFERENCES "t_plano"("pla_id")
      MATCH SIMPLE
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
      NOT DEFERRABLE,
  CONSTRAINT "Ref_05" FOREIGN KEY ("doe_id")
    REFERENCES "t_doenca"("doe_id")
      MATCH SIMPLE
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
      NOT DEFERRABLE
);
```

```
-----------------------------------
-----CRIAÇÃO DA TABELA EXAMES------
-----------------------------------
CREATE TABLE "t_exames" (
      "exa_id" int4 NOT NULL,
      "exa_resultado" varchar(20),
      "exa_data" date,
      "exa_validade" date,
      "pac_id" int4 NOT NULL,
  PRIMARY KEY("exa_id"),
  CONSTRAINT "Ref_09" FOREIGN KEY ("pac_id")
    REFERENCES "t_paciente"("pac_id")
      MATCH SIMPLE
      ON DELETE NO ACTION
```

```
      ON UPDATE NO ACTION
      NOT DEFERRABLE
);
-----------------------------------
-----CRIAÇÃO DA TABELA DOENÇA------
-----------------------------------
CREATE TABLE "t_doenca" (
      "doe_id" int4 NOT NULL,
      "doe_nome" varchar(50) NOT NULL,
      "rem_id" int4 NOT NULL,
  PRIMARY KEY("doe_id"),
  CONSTRAINT "Ref_07" FOREIGN KEY ("rem_id")
    REFERENCES "t_remedio"("rem_id")
      MATCH SIMPLE
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
      NOT DEFERRABLE
)
-----------------------------------
-----CRIAÇÃO DA TABELA REMÉDIO-----
-----------------------------------
CREATE TABLE "t_remedio" (
      "rem_id" int4 NOT NULL,
      "rem_nome" varchar(20) NOT NULL,
      "rem_dose" varchar(25),
  PRIMARY KEY("rem_id")
);
```

3.2.2 Alterações

3.2.2.1 Inclusão de tipos de dados

```
------------------------------
----INCLUSÃO DO TIPO CPF------
------------------------------
CREATE TYPE cpf AS
    ( pac_cpf numeric(11) );

------------------------------
---INCLUSÃO DO TIPO CNPJ------
------------------------------
CREATE TYPE cnpj AS
```

```
    ( pla_cnpj numeric(14) );
```

3.2.2.2 Inclusão de regras

```
-------------------------------------------------------------
----------REGRA AO CONFIRMAR SE A CONSULTA É RETORNO------------
-------------------------------------------------------------
ALTER TABLE "t_consulta" ADD CONSTRAINT "confirma_retorno" CHECK
("con_retorno"='S'OR "con_retorno"='N');

-----------------------------------------------------------------
REGRA AO CONFIRMAR SE A CONSULTA É CONVÊNIO OU PARTICULAR----
-------------------------------------------------------------
ALTER TABLE "t_consulta" ADD CONSTRAINT "tipo_consulta" CHECK
("con_tipo"='CONVÊNIO'OR "con_tipo"='PARTICULAR');
```

3.2.2.3 Valores *default*

```
-------------------------------------------------------------
--VALOR DEFAULT "CONVÊNIO" PARA O ATRIBUTO TIPO DE CONSULTA--
-------------------------------------------------------------
ALTER TABLE "t_consulta"
ALTER COLUMN "con_tipo" SET DEFAULT 'CONVÊNIO';

-----------------------------------------------------
-----VALOR DEFAULT "N" PARA O ATRIBUTO RETORNO------
-----------------------------------------------------
ALTER TABLE "t_consulta"
ALTER COLUMN "con_retorno" SET DEFAULT 'N';
```

3.2.2.4 Criação de triggers

3.2.2.4.1 Trigger para emissão de mensagem de erro

```
-------------------------------------------------------------
--TRIGGER QUE RETORNA MENSAGEM DE ERRO SE ULTIMA CONSULTA--
-------COM O MÉDICO TIVER SIDO A MENOS DE 30 DIAS----------
-------------------------------------------------------------
CREATE TRIGGER tr_consulta_medico
AFTER INSERT ON t_consulta
    FOR EACH ROW EXECUTE PROCEDURE tr_consulta_medico();

-------------------------------------------------------------
---------FUNÇÃO DA TRIGGER DE EMISSÃO DE MENSAGEM DE ERRO--
```

```
-------------------------------------------------------------
CREATE OR REPLACE FUNCTION tr_consulta_medico() RETURNS TRIGGER AS
$cons$
    DECLARE
      tr_con_data    date;
      tr_pac_id      integer;
      tr_med_id      integer;
    BEGIN
      tr_con_data    = NEW.con_data;
      tr_pac_id      = NEW.pac_id;
      tr_med_id      = NEW.med_id;
            BEGIN
            SELECT MAX(con_data)
            FROM t_consulta
            WHERE med_id = tr_med_id AND
                      pac_id = tr_pac_id;

                IF ((tr_con_data-con_data)< 30)THEN
                 RAISE EXCEPTION 'CONSULTA REALIAZA COM O MESMO
MÉDICO A MENOS DE 30 DIAS. INCLUSÃO CANCELADA';
            END IF;
            END;
        RETURN NEW;
    END;
$cons$ LANGUAGE plpgsql;
```

3.2.2.4.2 Triggers de inserção e atualização

```
-------------------------------------------------------------------
--TRIGGER QUE INSERE AUTOMATICAMENTE UMA NOVA CONSULTA DE RETORNO--
-------------------------------------------------------------------
CREATE TRIGGER tr_insere_retorno
AFTER INSERT ON t_consulta
    FOR EACH ROW EXECUTE PROCEDURE tr_insere_retorno();

-----------------------------------
----FUNÇÃO DA TRIGGER DE INSERÇÃO---
-----------------------------------
CREATE OR REPLACE FUNCTION tr_insere_retorno() RETURNS TRIGGER AS
$retorno$
    DECLARE
```

```
      tr_con_id      integer;
      tr_con_data    date;
      tr_con_hora    time(4);
      tr_con_tipo    varchar;
      tr_con_retorno char(1);
      tr_pac_id      integer;
      tr_med_id      integer;
      tr_exa_id      integer;
    BEGIN
      tr_con_id      = NEW.con_id+1;
      tr_con_data    = NEW.con_data+INTERVAL '30 DAYS';
      tr_con_hora    = NEW.con_hora;
      tr_con_tipo    = NEW.con_tipo;
      tr_con_retorno = 'S';
      tr_pac_id      = NEW.pac_id;
      tr_med_id      = NEW.med_id;
      tr_exa_id      = NEW.exa_id;
            BEGIN
                    INSERT INTO t_consulta(con_id,con_data, con_hora,
con_tipo, con_retorno, pac_id, med_id, exa_id)
                         VALUES (tr_con_id, tr_con_data, tr_con_hora,
tr_con_tipo, tr_con_retorno, _pac_id, tr_med_id, tr_exa_id);
           END;
        RETURN NULL;
    END;
$retorno$ LANGUAGE plpgsql;

-----------------------------------------------
--TRIGGER QUE ATUALIZA O CADASTRO DO CLIENTE--
--QUANDO SEU PLANO É DESVINCULADO DA CLÍNICA--
-----------------------------------------------

CREATE TRIGGER tr_atualiza_paciente
AFTER DELETE ON t_plano
    FOR EACH ROW EXECUTE PROCEDURE tr_atualiza_paciente();


----------------------------------------
----FUNÇÃO DA TRIGGER DE ATUALIZAÇÃO----
----------------------------------------
```

```
CREATE OR REPLACE FUNCTION tr_atualiza_paciente() RETURNS TRIGGER
AS $atualiza$
    DECLARE
        tr_pla_id integer;
    BEGIN
        tr_pla_id = OLD.pla_id;
             BEGIN
                UPDATE t_paciente
                      SET pac_obs ='PACIENTE COM PLANO DE SAÚDE
DESVINCULADO DA CLÍNICA'
                WHERE pla_id = tr_pla_id;
            END;
        RETURN NULL;
    END;
$atualiza$ LANGUAGE plpgsql;
```

3.2.2.5 Criação de *stored procedure*

```
-------------------------------------------------------
--PROCEDURE PARA CONSULTAR AS OBSERVAÇÕES DO PACIENTE--
-----PASSANDO COMO PARAMENTRO O ID DO PACIENTE---------
-------------------------------------------------------

CREATE OR REPLACE FUNCTION
procedure_consulta_obs(INT) RETURNS VARCHAR AS $obs$
DECLARE tmprow RECORD;
pac_id ALIAS FOR $1;
BEGIN
```

```
EXECUTE ('SELECT INTO tmprow * FROM t_paciente WHERE ' || ' pac_id = '
|| id || '');
END;
$obs$ LANGUAGE plpgsql;
```

3.2.2.6 Criação de visões

```
------------------------------
--VISÃO DAS CONSULTAS DO DIA--
------------------------------

CREATE VIEW consultas_do_dia AS
SELECT
      a.pac_nome AS NOME_PACIENTE,
      b.med_nome AS NOME_MEDICO,
      c.con_data AS DATA_CONSULTA,
      c.con_hora AS HORA_CONSULTA
FROM
      t_paciente a,
      t_medico   b,
      t_consulta c
WHERE
      c.med_id = b.med_id and
      a.pac_id = c.pac_id and
      con_data = current_date;


-----------------------------------------------
--VISÃO DOS EXAMES QUE JÁ POSSUEM RESULTADOS--
-----------------------------------------------
CREATE VIEW exames_com_resultado AS
SELECT
      a.exa_id         AS TIPO_EXAME,
      a.exa_data       AS DATA_EXAME,
      a.exa_resultado  AS RESULTADO_EXAME,
      b.pac_nome       AS NOME_PACIENTE
FROM
      t_exames a,
      t_paciente b
```

```
WHERE
      b.pac_id = a.pac_id and
      exa_resultado <> null;


------------------------------------------------------------
--VISÃO DOS TELEFONES DOS MÉDICOS QUE ATENDEM NA CLÍNICA--
------------------------------------------------------------


CREATE VIEW fone_medico AS
SELECT
      a.med_crm  AS CMR_MEDICO,
      a.med_nome AS NOME_MEDICO,
      a.med_fone AS FONE_MEDICO,
      b.pla_nome AS NOME_PLANO_SAUDE,
      b.pla_fone AS FONE_PLANO_SAUDE,
      c.esp_nome AS ESPECIALIDADE
FROM
      t_medico        a,
      t_plano         b,
      t_especialidade c
WHERE
      a.pla_id = b.pla_id and
      a.esp_id = c.esp_id
```

3.2.2.7 Criação de usuários

```
------------------------------------------------------------
--------------CRIAÇÃO DE USUÁRIO COM SENHA------------------
------------------------------------------------------------

CREATE USER lula WITH PASSWORD 'L1U2L3A4';


------------------------------------------------------------
--CRIAÇÃO DE USUÁRIO COM SENHA E DATA DE LIMITE DE CONEXÃO--
```

```
-------------------------------------------------------------

CREATE USER obama WITH PASSWORD 'O1B2A3M4A' VALID UNTIL '31/12/2009';
```

3.2.2.8 Permissão de acesso aos usuários

```
-------------------------------------------------------------
-----------PERMISSÃO DE ACESSO AO USUÁRIO LULA--------------
-------------------------------------------------------------

GRANT SELECT,UPDATE,INSERT ON t_paciente, t_consulta TO lula;


-------------------------------------------------------------
----------PERMISSÃO DE ACESSO AO USUÁRIO OBAMA--------------
-------------------------------------------------------------

GRANT SELECT ON t_consulta TO obama;
```

3.2.3 Consultas

3.2.3.1 Relatório para consultas com no mínino de três junções

```
-------------------------------------------------------------
--RELATÓRIO COM DADOS DO MÉDICO, PLANO, CONSULTA E PACIENTE--
-------------------------------------------------------------
SELECT
      a.med_crm  as CRM,
      a.med_nome as NOME_DO_MEDICO,
      b.pla_cnpj as CNPJ_DO_PLANO_DE_SAUDE,
      b.pla_nome as NOME_DO_PLANO_DE_SAUDE,
```

```
      c.con_data as DATA_DA_CONSULTA,
      c.con_hora as HORA_DA_CONSULTA,
      d.pac_nome as NOME_DO_PACIENTE
FROM
      t_medico a, t_plano b, t_consulta c, t_paciente d
WHERE
      a.pla_id = b.pla_id and
      a.med_id = c.med_id and
      c.pac_id = d.pac_id;
```

3.2.3.2 Relatório com sub-consulta

```
-----------------------------------------------------------------
--RELATÓRIO COM DADOS DO MÉDICO E QUANTIDADE DE CONSULTAS RETORNO---
-----------------------------------------------------------------
SELECT
      med_nome as NOME_DO_MEDICO,
        COUNT(*) as QUNTIDADE_DE_RETORNOS
FROM
      (SELECT
            a.med_id,
            b.med_nome,
            a.con_retorno
      FROM
                t_consulta a, t_medico b
      WHERE
            a.med_id = b.med_id and
                a.con_retorno='S') as consulta
GROUP BY med_nome;
```

3.2.3.3 Relatório com *inner-join*

```
-------------------------------------------------------
--RELATÓRIO COM DADOS DOS PACIENTES E PLANOS DE SAÚDE---
-------------------------------------------------------

SELECT
           a.pla_nome as NOME_DO_PLANO_DE_SAUDE,
           b.pac_nome as NOME_DO_PACIENTE
FROM
          t_plano a INNER JOIN t_paciente b
          ON (a.pla_id=b.pla_id)
```

```
ORDER BY
          a.pla_nome;
```

3.2.3.4 Relatório com agrupamento utilizando a cláusula *Having*

```
-------------------------------------------------
--RELATÓRIO DA QUANTIDADE DE EXAMES POR PACIENTE---
-------------------------------------------------

SELECT
      a.pac_nome AS NOME_DO_PACIENTE,
      COUNT(b.exa_id) AS QUANT_DE_EXAMES
FROM
      t_paciente a, t_exames b
WHERE
      a.pac_id = b.pac_id
GROUP BY
      (a.pac_nome)
HAVING
      COUNT(b.exa_id) > 0;
```