



Universidade do Porto

Faculdade de Engenharia

FEUP

Bases de Dados, MIEIC

2010/2011

João Mendes Moreira, Válter Neto da Rocha

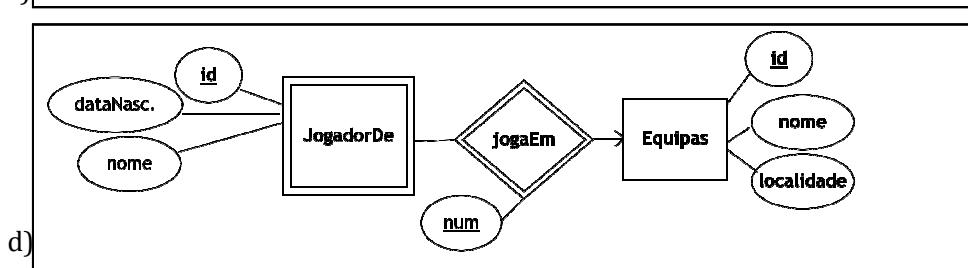
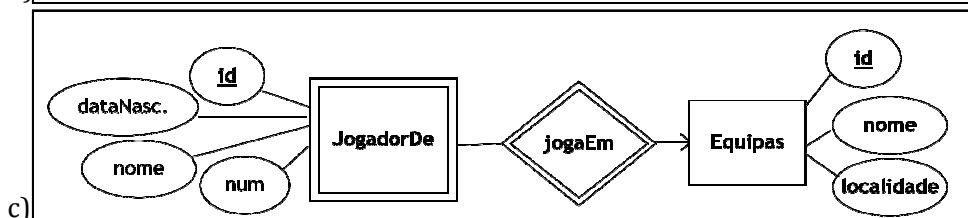
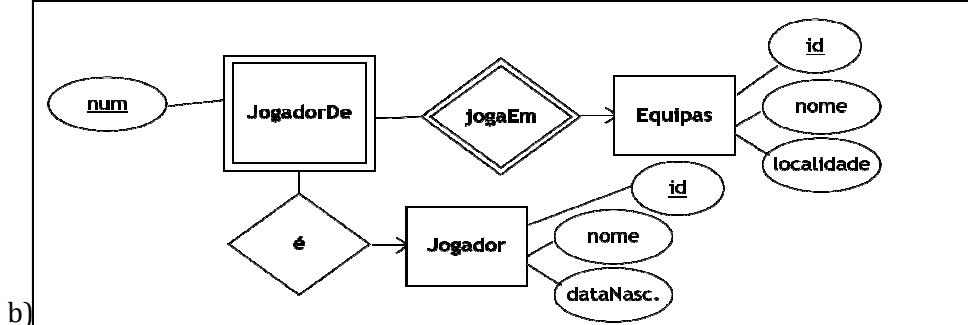
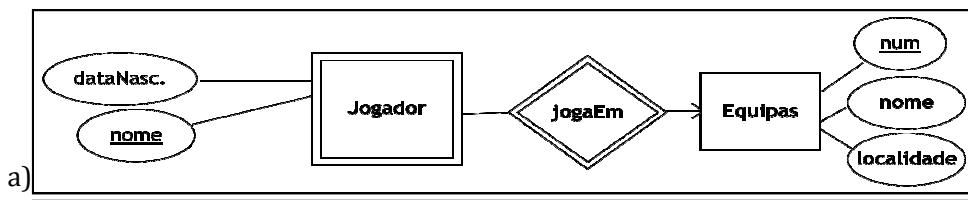
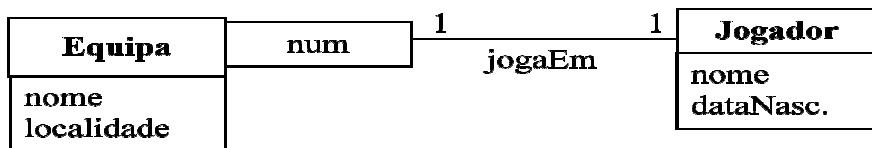
21 de Janeiro de 2011

Exame época normal: correcção

1. Selecione uma e uma só das 4 respostas possíveis. Só há uma resposta certa por alínea. Uma resposta certa vale 0,5 valores; errada vale -0,1 valores; e omissa vale 0 valores.

i) Indique qual o modelo Entidade-Associação equivalente ao seguinte diagrama UML:

b



ii) Diga em que forma normal se encontra o seguinte esquema relacional:
R1(a,b), R2(b,c,d) com as DFs: a->b; b->c; c->d.

b

- a) 1^a FN
- b) 2^a FN
- c) 3^a FN
- d) 3^a FNBC

iii) Em SQL, uma sequência serve para:

a

- a) Gerar sequências de números.
- b) Ordenar os campos devolvidos numa consulta por um dado critério.
- c) Ordenar os tuplos resultantes de uma consulta por um dado critério.
- d) Nenhuma das respostas anteriores.

iv) Diga qual das seguintes afirmações sobre a regra Datalog “ $R(x) \leftarrow S(x) \text{ AND } x > y$ ” descreve melhor a segurança dessa regra:

a

- a) A regra não é segura porque y não está limitada a um conjunto finito.
- b) A regra não é segura porque x não tem limite superior.
- c) A regra é segura.
- d) As duas primeiras alíneas são verdadeiras.

v) Diga qual das seguintes frases se aplica a vistas materializadas:

c

- a) Servem para fazer *backups* da base de dados.
- b) Servem para facilitar a escrita de instruções SQL.
- c) Permitem a criação de imagens parciais físicas da base de dados.
- d) Permitem a criação de imagens parciais lógicas da base de dados.

vi) Considere a instrução SQL “SELECT Encomenda.data FROM Cliente, Encomenda WHERE Cliente.idCliente=Encomenda.idCliente ORDER BY Encomenda.data DESC;”. Sabendo que as chaves primárias das tabelas Cliente e Encomenda são, respectivamente, idCliente e idEncomenda, diga que índices deviam ser criados para tornar esta consulta mais rápida.

d

- a) Cliente.idCliente, Encomenda.data.
- b) Encomenda.idCliente, Cliente.idCliente, Encomenda.data.
- c) Encomenda.data.
- d) Encomenda.idCliente, Encomenda.data.

vii) A instrução “EXEC SQL EXECUTE IMMEDIATE <consulta>;” permite:

d

- a) Executar uma instrução SQL de forma mais célere.
- b) Construir uma instrução SQL de forma dinâmica.
- c) Executar uma instrução SQL de forma dinâmica.
- d) Construir e executar uma instrução SQL de forma dinâmica.

viii) Durante o processo de submissão de propostas científicas, uma conhecida instituição espera receber muitas propostas através da página de submissão. Para tornar o processo mais eficiente optou pela utilização de bloqueio em duas fases não estrito, com os bloqueios pedidos quando necessários e libertados imediatamente a seguir, uma vez que não são mais necessários. Não é tomada nenhuma medida adicional. Qual o problema que pode ocorrer com esta estratégia?

d

- a) Bloqueio activo (*livelock*).
- b) Encravamento (*deadlock*).
- c) Cancelamento em cascata.
- d) Qualquer um dos três referidos.

ix) Porque é que o método que utiliza marcas temporais como forma de garantir que um escalonamento alternado é serializável, é pouco eficiente quando há muitas transacções concorrentes?

b

- a) Porque proporciona muitos encravamentos.
- b) Porque proporciona cancelamentos em cascata frequentes.
- c) Porque proporciona tempos de espera elevados.
- d) Qualquer um dos três referidos.

x) Nos sistemas de armazenamento de dados, qual a frase que melhor descreve o esquema em estrela por comparação com o esquema em floco de neve?

c

- a) Consultas mais rápidas e menos espaço em disco.
- b) Consultas mais lentas e menos espaço em disco.
- c) Consultas mais rápidas e mais espaço em disco.
- d) Consultas mais lentas e mais espaço em disco.

Bases de Dados, MIEIC

2010/2011

João Mendes Moreira, Válter Neto da Rocha

21 de Janeiro de 2011

Duração: 2h30m

1. Uma associação de pombos-correios quer informatizar toda a informação de que dispõe.

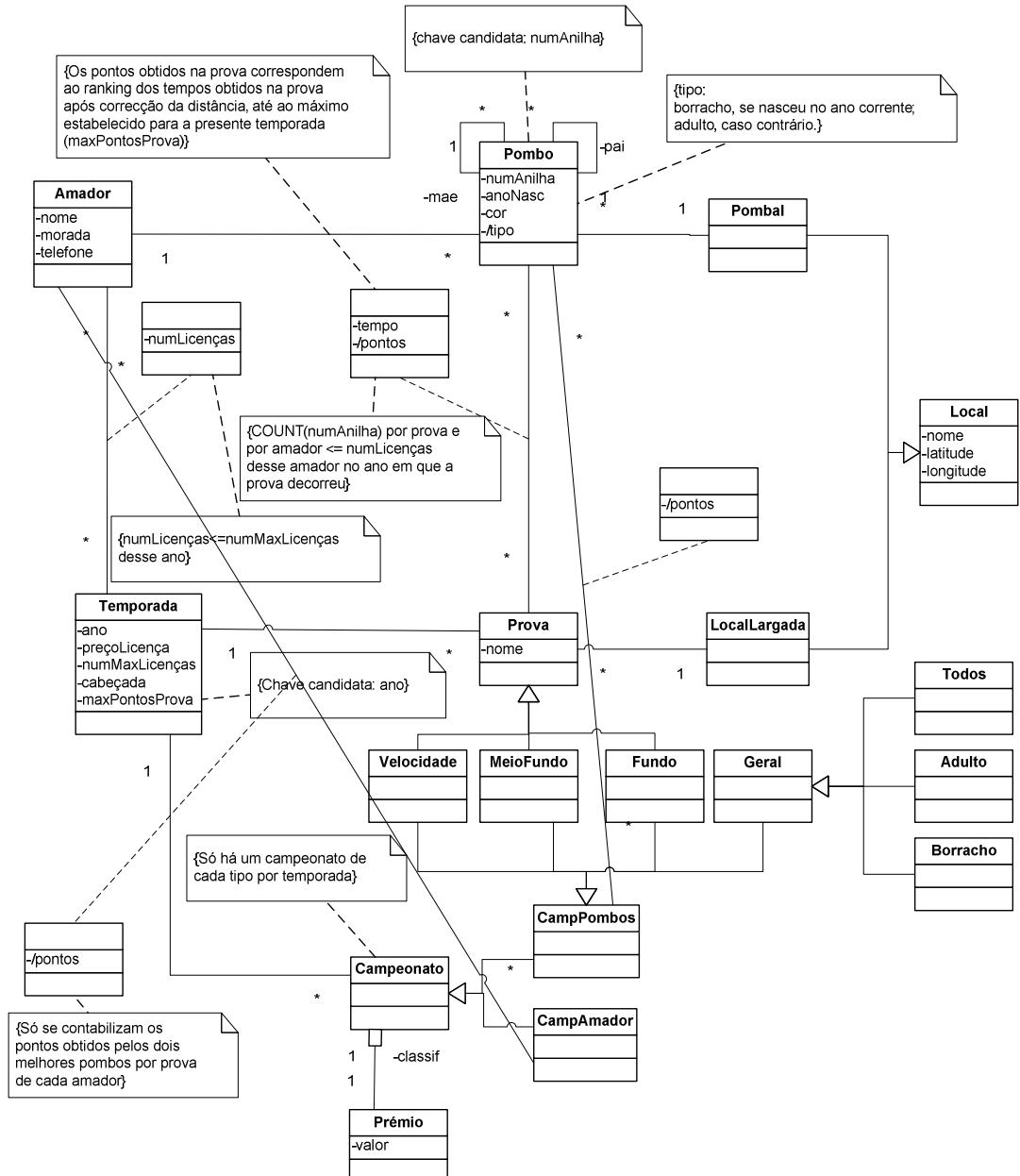
A associação é composta por vários membros (designados por amadores) dos quais convém guardar os dados pessoais (nome, morada, etc.). Cada amador pode ter vários pombos sendo identificados pelo número da anilha – anel metálico numerado que se coloca na pata do pombo. Associado a cada anilha existe um cartão de identidade do pombo onde constam os números das anilhas quer do pai quer da mãe, a cor e o ano de nascimento do pombo: se o pombo tiver nascido no ano corrente é considerado borracho, caso contrário é adulto. No início de cada ano a associação estabelece o preço por licença, assim como o número máximo de licenças por amador. Cada amador paga um certo número de licenças (necessariamente inferior ou igual ao máximo permitido) correspondentes ao número máximo de pombos que o amador pode utilizar numa mesma prova.

A associação organiza três tipos de provas: velocidade, meio fundo e fundo. Das provas sabe-se, para além do tipo de prova, o seu nome e o local de largada. Os pombos utilizados por cada amador podem variar de prova para prova. A pontuação dada é de 1 ponto para o 1º classificado, 2 para o 2º e assim sucessivamente até um máximo pré-definido no início do ano. Todos os pombos que cheguem para além desse limite ou que não participem na prova são penalizados com uma pontuação superior (designada por cabeçada), que também é estabelecida no início do ano. Para estabelecer a ordem de chegada é necessário fazer correções do tempo de chegada em função da localização do pombal de cada amador (pois a distância entre o ponto de largada dos pombos e o de chegada depende da localização do pombal). Para isso, é necessário definir a latitude e a longitude do pombal (assume-se que cada amador tem um e um só pombal) e do local de largada.

Com base nas pontuações obtidas nas diversas provas são estabelecidas as classificações de diferentes campeonatos: velocidade, meio fundo, fundo e geral (neste último, todas as provas são contabilizadas) para todos os pombos e ainda o campeonato geral em separado para pombos borrachos e adultos. É, também, feito um campeonato por amador em que são considerados os dois pombos melhores classificados de cada amador, em cada uma das provas promovidas pela associação. Ganha cada um dos campeonatos o pombo ou amador que obtiver menos pontos.

Os prémios monetários dados em cada competição são definidos no início de cada ano em função da competição e da classificação obtida.

- a) **(5 valores)** Defina o diagrama de classes UML para a informação acima descrita. Indique todas as restrições que possam ser úteis para a construção da base de dados.



2. Considere a relação $R(A,B,C,D,E)$ com as dependências funcionais $F=\{A \rightarrow BC, AC \rightarrow D, BD \rightarrow C, C \rightarrow E\}$.
- a) **(1 valor)** Sabendo que esta relação foi decomposta nas relações $R1(A,B)$, $R2(B,C,D)$ e $R3(C,E)$, diga, justificando, quais os problemas, se os houver, de que esta decomposição sofre.

Há dois problemas possíveis: (1) não preservação das dependências; (2) decomposição com perdas.

Quanto ao primeiro problema, a forma minimal das DFs apresentadas é: $A \rightarrow B$; $A \rightarrow C$; $AC \rightarrow D$; $BD \rightarrow C$; $C \rightarrow E$.

$AC \rightarrow D$ é equivalente a $A \rightarrow D$, pois $A \rightarrow C$; $A \rightarrow C$ é redundante pois $A \rightarrow BD$ e $BD \rightarrow C$.

Logo: $A \rightarrow B$; $A \rightarrow D$; $BD \rightarrow C$ e $C \rightarrow E$ é a forma minimal das DFs dadas. A DF $A \rightarrow D$ não é preservada por motivos óbvios: estes dois atributos não aparecem em simultâneo em nenhuma das relações decompostas.

Para verificar se há junção sem perdas, utiliza-se o algoritmo da caça.

A	B	C	D	E	
a	b	c1	d1	e1	R1(A,B)
a2	b	c	d	e2	R2(B,C,D)
a3	b3	c	d3	e	R3(C,E)

Uma vez que as linhas 2 e 3 têm só letras nos atributos do lado esquerdo da DF $C \rightarrow E$, vem que

A	B	C	D	E	
a	b	c1	d1	e1	R1(A,B)
a2	b	c	d	e	R2(B,C,D)
a3	b3	c	d3	e	R3(C,E)

Uma vez que não se consegue obter todos os atributos a partir dos dados de uma relação pode-se concluir que esta decomposição não garante junções sem perdas. Notar que não existe nenhuma linha só com algarismos.

Observação: para ver se há preservação das DFs, é relevante obter antes a forma minimal. Para ver se há junção sem perdas o resultado obtido seria o mesmo com forma minimal e com as DFs originais. Porque será?

- b) **(1 valor)** Decomponha a relação R na 3^a forma normal.

Uma vez que já se obtiveram as DFs na forma minimal, a obtenção das tabelas na 3^a forma normal é directa:

$R1(\underline{A},B,D)$; $R2(\underline{B},\underline{D},C)$; $R3(\underline{C},E)$. De notar que (A) seria a chave da relação original R . Como tal, não é necessário acrescentar mais nenhuma relação porque já existe uma tendo como chave primária (A).

3. **(FAZER ESTE GRUPO NUMA FOLHA DE EXAME SEPARADA)** Um laboratório de análises pretende um sistema de informação que deverá automatizar o processo de detecção de problemas dos seus clientes assim como guardar os seus dados, os limites clínicos para cada paciente acima ou abaixo dos quais é necessário gerar um alerta, os valores das análises e os alertas que forem gerados sempre que um limite seja ultrapassado. Verifique o seguinte modelo relacional:

MODELO RELACIONAL

Paciente(numUtente, nome, dataNascimento, sexo, morada, telf, notas, codPostal)

Analise_Tipo(tipoAnalise, nome)

Analise (idAnalise, tipoAnalise → Analise_Tipo, numUtente → Paciente, dataHora, valor, notas, bool_alerta) – O bool_alerta é 0 se FALSO ou 1 se VERDADEIRO.

Paciente_Limite(numUtente → Paciente, tipoAnalise → Analise_Tipo, limInf, limSup)

- a) **(1 valor)** Indique as instruções de SQL que permitam construir as tabelas do modelo relacional acima. Note que os atributos tipoAnalise e idAnalise são auto-incrementáveis.

CREATE TABLE "PACIENTE"

```
(  "NUMUTENTE" NUMBER NOT NULL,
  "NOME" VARCHAR2(100),
  "DATANASCIMENTO" DATE,
  "SEXO" VARCHAR2(1),
  "MORADA" VARCHAR2(200),
  "TELF" VARCHAR2(9),
  "NOTAS" VARCHAR2(500),
  "CODPOSTAL" VARCHAR2(8),
  CONSTRAINT "PACIENTE_PK" PRIMARY KEY ("NUMUTENTE")
)
```

CREATE TABLE "ANALISE_TIPO"

```
(  "TIPOANALISE" NUMBER NOT NULL,
  "NOME" VARCHAR2(4000),
  CONSTRAINT "ANALISE_TIPO_PK" PRIMARY KEY ("TIPOANALISE")
);
```

CREATE OR REPLACE TRIGGER "AI_ANALISE_TIPO"

```
BEFORE INSERT ON "ANALISE_TIPO"
FOR EACH ROW
```

```
BEGIN
    SELECT "ANALISE_TIPO_SEQ".NEXTVAL INTO :NEW.TIPOANALISE FROM DUAL;
END;
```

CREATE TABLE "ANALISE"

```
(  "IDANALISE" NUMBER NOT NULL,
  "TIPOANALISE" NUMBER NUMBER NOT NULL,
  "NUMUTENTE" NUMBER NUMBER NOT NULL,
  "DATAHORA" DATE,
  "VALOR" NUMBER,
  "NOTAS" VARCHAR2(100),
  "BOOL_ALERTA" NUMBER,
  CONSTRAINT "ANALISE_PK" PRIMARY KEY ("IDANALISE"),
  CONSTRAINT "ANALISE_FK1" FOREIGN KEY ("TIPOANALISE") REFERENCES
  "ANALISE TIPO" ("TIPOANALISE"),
  CONSTRAINT "ANALISE_FK2" FOREIGN KEY ("NUMUTENTE") REFERENCES "PACIENTE"
  ("NUMUTENTE")
);
```

CREATE OR REPLACE TRIGGER "AI_ANALISE"

```
BEFORE INSERT ON "ANALISE"
FOR EACH ROW
BEGIN
    SELECT "ANALISE_SEQ".NEXTVAL INTO :NEW.IDANALISE FROM DUAL;
END;
```

CREATE TABLE "PACIENTE_LIMITE"

```
(  "NUMUTENTE" NUMBER,
  "TIPOANALISE" NUMBER,
  "LIMINF" NUMBER,
  "LIMSUP" NUMBER,
  CONSTRAINT "PACIENTE_LIMITE_PK" PRIMARY KEY ("NUMUTENTE", "TIPOANALISE"),
  CONSTRAINT "PACIENTE_LIMITE_FK" FOREIGN KEY ("NUMUTENTE") REFERENCES
  "PACIENTE" ("NUMUTENTE") ENABLE,
  CONSTRAINT "PACIENTE_LIMITE_FK2" FOREIGN KEY ("TIPOANALISE") REFERENCES
  "ANALISE TIPO" ("TIPOANALISE"));
```

- b) **(1 valor)** Construa a função PL/SQL *PacientesOK* que, ao receber o tipoAnalise e o numUtente, devolve um valor Booleano a indicar se esse paciente tem o último valor para esse tipo de análise dentro dos seus limites. Obs.: em PL/SQL existe o tipo de dados *Boolean*.

```

CREATE OR REPLACE FUNCTION "PACIENTESOK" (TA IN NUMBER, NU IN NUMBER) RETURN
BOOLEAN IS
    VAL NUMBER;
    LS NUMBER;
    LI NUMBER;
    BEGIN

        SELECT LIMINF, LIMSUP INTO LI, LS FROM PACIENTE_LIMITE WHERE NUMUTENTE
        = NU AND TIPOANALISE = TA;
        SELECT VALOR INTO VAL FROM ANALISE NATURAL JOIN (SELECT NUMUTENTE,
        TIPOANALISE, MAX(DATAHORA) AS DATAHORA FROM ANALISE WHERE
        NUMUTENTE = NU AND TIPOANALISE = TA GROUP BY NUMUTENTE, TIPOANALISE);

        IF ((VAL < LI) OR (VAL > LS))
            THEN RETURN(TRUE);
        ELSE RETURN (FALSE);
        END IF;

    END;

```

- c) **(1 valor)** Construa uma Vista PacientesHemoglobinaNotOk que tire partido da função *PacientesOK* e retorne uma lista com o numUtente, nome, telefone, valor da análise e o limite que foi transgredido de todos os pacientes cuja Hemoglobina (*Analise_Tipo.nome*) não esteja dentro dos valores normais.

```

CREATE OR REPLACE FORCE VIEW "PACIENTESHEMOGLOBINANOTOK" ("NUMUTENTE",
"NOME", "TELF", "NOMEANALISE", "VALOR", "LIMINF", "LIMSUP") AS
SELECT NUMUTENTE, NOME, TELF, NOMEANALISE, VALOR, LIMINF, LIMSUP FROM (
    SELECT NUMUTENTE, TIPOANALISE, NOME AS NOMEANALISE, DATAHORA, PNOK FROM (
        SELECT NUMUTENTE, TIPOANALISE, MAX(DATAHORA) AS DATAHORA,
        PACIENTESOK(TIPOANALISE, NUMUTENTE) AS PNOK FROM ANALISE GROUP BY
        NUMUTENTE, TIPOANALISE) NATURAL JOIN ANALISE_TIPO
        WHERE NOME = 'HEMOGLOBINA' AND PNOK = 1)
NATURAL JOIN PACIENTE_LIMITE NATURAL JOIN PACIENTE
NATURAL JOIN (SELECT NUMUTENTE, TIPOANALISE, DATAHORA, VALOR FROM ANALISE)

```

- d) **(1 valor)** Crie um *Trigger* que actualize o atributo bool_alerta sempre que ao inserir dados na tabela análise estes estejam fora dos limites impostos para o paciente e para o tipo de análise em questão.

```
CREATE OR REPLACE TRIGGER "ANALISE_T1" BEFORE INSERT ON "ANALISE"
FOR EACH ROW
DECLARE
    RESULT BOOLEAN;
BEGIN
    RESULT := PACIENTESOK(:NEW.TIPOANALISE, :NEW.NUMUTENTE);
    IF RESULT = TRUE
        THEN :NEW.BOOL_ALERTA := 1;
        ELSE :NEW.BOOL_ALERTA := 0;
    END IF;
END;
```

- e) **(1 valor)** Usando a LMD SQL diga quanto tipos de análises diferentes foram realizadas por cada paciente.

```
SELECT NUMUTENTE, COUNT(DISTINCT TIPOANALISE) FROM ANALISE GROUP BY NUMUTENTE
```

- f) **(1 valor)** Usando a LMD SQL diga quais os resultados das análises mais recentes para cada um dos tipos de análise realizadas pelo paciente 'Alberto Amaral'.

```
SELECT TIPOANALISE, VALOR FROM ANALISE NATURAL JOIN (
    SELECT NUMUTENTE, TIPOANALISE, MAX(DATAHORA) AS DATAHORA FROM ANALISE
    NATURAL JOIN( SELECT NUMUTENTE, NOME FROM PACIENTE WHERE NOME = 'ALBERTO
    AMARAL')
    GROUP BY NUMUTENTE, TIPOANALISE)
```

- g) **(1 valor)** Usando a LMD SQL diga, para todos os pacientes, quais os tipos de análise que viram os valores aumentados entre a penúltima e a última vez que foram recolhidas.

```
SELECT * FROM (
    SELECT NUMUTENTE, TIPOANALISE, MAX(DATAHORA) AS PENDATAHORA FROM (
        SELECT NUMUTENTE, TIPOANALISE, DATAHORA FROM ANALISE
        MINUS (
            SELECT NUMUTENTE, TIPOANALISE, MAX(DATAHORA) FROM ANALISE GROUP BY
            NUMUTENTE, TIPOANALISE)
        GROUP BY NUMUTENTE, TIPOANALISE
    ) NATURAL JOIN (
        SELECT NUMUTENTE, TIPOANALISE, MAX(DATAHORA) AS ULTADATAHORA FROM ANALISE
        GROUP BY NUMUTENTE, TIPOANALISE)
    NATURAL JOIN (
        SELECT NUMUTENTE, TIPOANALISE, DATAHORA AS PENDATAHORA, VALOR AS PENVALOR
        FROM ANALISE) NATURAL JOIN (
        SELECT NUMUTENTE, TIPOANALISE, DATAHORA AS ULTADATAHORA, VALOR AS ULTVALOR
        FROM ANALISE)
    WHERE ULTVALOR > PENVALOR
```

- h) **(1 valor)** Usando a LMD SQL diga qual o nome dos três pacientes com mais alertas activos (bool_alerta=1).

```
SELECT NOME FROM PACIENTE NATURAL JOIN (
    SELECT NUMUTENTE, COUNT(BOOL_ALERTA) CNT FROM ANALISE WHERE BOOL_ALERTA =
    1 GROUP BY NUMUTENTE ORDER BY CNT DESC) WHERE ROWNUM<=3
```