## Grupo I – 2º teste 2020/21

I-1) Devolve os elementos cujos pais têm no máximo 2 filhos elementos

Para o documento dado obtemos:

```
<Item codItem="E1" nome="Croquetes" tipo="carne" preço="1.2">Croquetes especiais</Item>
<Item codItem="E2" nome="Azeitonas" tipo="vegan" preço="0.95">Sortido de azeitonas</Item>
```

I-2a)

```
//Item[@tipo='vegan' and not(@alérgenos)]
```

I-2b)

```
/Menu/Combinado[sum(id(@itens)/@preço)*(100 - @desconto) div 100 > 9]/@nome
```

OU

```
/Menu/Combinado[sum(id(@itens)/@preço)*(1 – 0.01 * @desconto) > 9]/@nome
```

I-3)

```
<Menu restaurante="{/Menu/@restaurante}">
{ for $x in (//Entradas | //Pratos | //Bebidas  | //Sobremesas ) return $x }
{ for $x in //Combinado return
   <Combinado nome="{$x/@nome}" desconto="{$x/@desconto}">
      { for $y in id($x/@itens) return $y}
   </Combinado>
}
</Menu>
```

ou

```
<Menu restaurante="{/Menu/@restaurante}">
   { /Menu/Entradas }
   { /Menu/Pratos }
   { /Menu/Sobremesas }
   { /Menu/Bebidas }
   { for $x in //Combinado return
      <Combinado nome="{$x/@nome}" desconto="{$x/@desconto}">
         { for $y in id($x/@itens) return
            <Item nome="{$y/@nome}" tipo="{$y/@tipo}" preço="{$y/@preço}">
                  {$y/@alérgenos}
            </Item>
          }
      </Combinado>
}
</Menu>
```

**Grupo II – 2º teste 2020/21**

II-1a)

```
SELECT DISTINCT artigos.nome
FROM serviços INNER JOIN Pedidos USING (numS)
                    INNER JOIN Artigos USING (codA)
                    INNER JOIN Empregaos USING (codE)
WHERE empregados.nome = 'Francisco' AND
          Serviços.dataS BETWEEN '2021-05-23' AND ‘2021-05-30'
ORDER BY artigos.nome
```

II-1b)

```
CREATE VIEW MesasServiços AS
SELECT *
FROM Serviços INNER JOIN AtribuiçõesMesas USING (numS)
                    LEFT OUTER JOIN MesasReservadas USING (codRs,numM)
```

II-1c)

```
CREATE VIEW MesasDisponíveis AS
SELECT numM, dataH as dataD, horaH as horaD, lugaresM as lugaresD
FROM Mesas INNER JOIN HoráriosAbertura USING (nomeR)
WHERE NOT EXISTS (
   SELECT * FROM Reservas INNER JOIN MesasReservadas USING (codRs)
   WHERE Mesas.numM = MesasReservadas.numM AND
              dataH = dataRS AND
              horaRS > horaH - 2 AND
              horaRS < horaH + 2
)
```

II-1d)

```
WITH faturação_servico(nomeR,numS,dataS,valor) AS
(SELECT nomeR, numS, dataS, SUM(qtd*preço) AS valor
  FROM serviços INNER JOIN pedidos USING (numS)
                      INNER JOIN artigos USING (codA)
  WHERE concluído = 1
  GROUP BY nomeR, numS, dataS
)
SELECT nomeR,dataS
FROM faturação_serviço
GROUP BY nomeR, dataS
HAVING AVG(valor) >= 100
```

II-1e)

```
nunca_vendidos(CodE,NomeE,CodA,NomeA) :-
   empregados(CodE,NomeE,_,_),
   artigo(CodA,NomeA,_,_),
   not vendidos(CodE,CodA).

vendidos(CodE,CodA) :-
   pedidos(_,_,_,CodE,CodA,_,_).
```

## Grupo III – 2º teste 2020/21

III-1a)

```
CREATE ASSERTION mesmo_restaurante
CHECK NOT EXISTS (
    SELECT *
    FROM Reservas INNER JOIN MesasReservadas USING (codRS)
                        INNER JOIN Mesas USING (numM)
    WHERE Reservas.nomeR <> Mesas.nomeR
)
```

III-1b)

```
ALTER TABLE reservas ADD CONSTRAINT horas_válidas
FOREIGN KEY (nomeR,dataRs,horaRS)
REFERENCES HorárioAbertura
```

III – 2 (em sintaxe oracle)

```
CREATE OR REPLACE TRIGGER RESERVA_MESAS
INSTEAD OF INSERT ON RESERVACLIENTES
REFERENCING OLD AS OROW NEW AS NROW
FOR EACH ROW
DECLARE m NUMBER;
BEGIN
 IF :NROW.CODRS IS NULL THEN

   SELECT MIN(numM) INTO m
   FROM Mesas NATURAL INNER JOIN MesasDisponiveis
   WHERE mesas.nomeR = :NROW.nomeR AND
       dataD = :NROW.dataRS AND
       to_char(horaD,'HH24:MI') = to_char(:NROW.horaRS,'HH24:MI') AND
       lugaresD >= :NROW.num_pessoas;

   IF m IS NOT NULL THEN
       INSERT INTO Reservas VALUES
(cod_reservas.NEXTVAL,:NROW.nomeR,:NROW.nomeC, :NROW.telC,
:NROW.dataRs,:NROW.horaRS);
       INSERT INTO MesasReservadas VALUES
(cod_reservas.CURRVAL,m,:NROW.num_pessoas);
   ELSE
       RAISE_APPLICATION_ERROR(-20000,'Não há mesas disponíveis!');
   END IF;
 END IF;
END;
```

III -3

O escalonamento indicado em modo de snapshot isolation imprime o valor x=6 e z=11. Ora, esta combinação de valores é impossível de gerar por uma das serializações das transações T1, T2 e T3:

T1, T2, T3: x = 6, z = 17
T1, T3, T2: x = 6, z = 13
T2, T1, T3: x = 10, z = 17
T2, T3, T1: x = 17, z = 11
T3, T1, T2: x = 13, z = 7
T3, T2, T1: x = 17, z = 7