Com base no teste de 2017/2018 (não expectável que exista replicação de formato de teste, relativamente ao de 2019/2020), respondo a algumas questões com base em conteúdo pesquisado no Guia de Estudos, ou em diversos documentos de universidades, fóruns, etc. Deixo as respostas que considero corretas, sendo que não quero deduzir ninguém em erro, deverá existir uma pré validação dos conteúdos, e caso exista algum erro não detetado, peço uma discussão breve sobre o tópico.

# Complemento do Guia de Estudos

## Exercícios

Alexandre Coelho

# Índice

# Teste 2017/2018

Note-se que este teste, pode não ter semelhanças ao teste realizado no ano letivo 2019/2020, mas não existe uma variação significativa de conteúdos. As respostas são dadas através de leitura do **Guia de Estudos**, anteriormente fornecido, e também de leitura de slides e transcrições de áudio (gravações em aula) para texto. As respostas podem conter erros, pois partem da minha perceção de conteúdos, assume-se que pode existir uma margem de erro em qualquer resposta, pelo que peço, se detetarem algum erro, notifiquem o responsável pelo documento, com a questão e respetiva solução. As respostas assinaladas a **negrito**, representam a resposta escolhida.

1. **<u>Relativamente às alternativas de RAID:</u>**

**a. RAID5 é adequado a operações: Muita escrita e pouca leitura**

b. RAID0 suporta maior redundância

c. Respostas a. e b.

d. Nenhuma das anteriores

**<u>Complemento de Resposta:</u>**

Quanto ao ponto *a. (***o correto***)*, diz-se que o RAID5 em **pouca leitura** , **READ I/O (<u>SMALL OPS)</u>**, apresenta um bom performance, e quanto a **Muita Escrita**, **WRITE I/O (<u>LARGE OPS)</u>** apresenta um performance muito bom. (indicado na imagem seguinte (retirada do dispositivo 19 do documento "*2-Storage"*)).

| | Read I/O(small ops) | Read I/O (large ops) | Write I/O (small ops) | Write I/O (large ops) | Fault Tolerance |
|---|---|---|---|---|---|
| RAID 0 | Good | Very Good | Very Good | Very Good | No |
| RAID 5 | Good | Very Good | Poor | Very Good | Yes |
| RAID 10 | Very Good | Good | Good | Good | Yes |

Quanto ao ponto *b.* , relativamente ao RAID0, este, **não permite redundância de dados**, como também necessita de pelo menos 2 unidades de disco rígido.

*(Resposta e complemento, retirados dos documentos das teóricas ("2-Storage") e do Guia de Estudos (a partir da página 10 na secção de Storage).*

2. **<u>Registos de tamanho variável fisicamente (variable-lenght records) ocorrem por:</u>**

a. Conterem campos que permitem valores null

b. Conterem campos que cujo o tipo é de representação física variável

c. Participar em páginas multi-table clustering

**d. Todas as anteriores**

**<u>Complemento de Resposta:</u>**

Variable-Lenght Records, ou registos de tamanho variável fisicamente, com base na sua nomenclatura, é claramente correto afirmar que contem campos que cujo o tipo é de representação física variável (assume-se imediatamente que o ponto *b.*, se encontra correto), mas temos que avaliar os outros. Quando ao ponto *a.*, podemos assumir que pode ocorrer também, o facto desses registos conterem campos que permitam valores null, até porque existe uma secção do registo que indica quais os atributos que têm valor null, esta secção denomina-se de **<u>null bitmap</u>**. Sendo que já contemos duas respostas corretas, podemos afirmar que seria correto o ponto ***d.,*** mas temos que definir a ocorrência na participação em páginas multi-table clustering. Este clustering que por norma tem tipicamente uma estrutura simplificada em que os tuplos de uma relação são representados como **<u>fixed-lenght records</u>** (que pode deduzir em erro), resulta em **<u>variable-length records</u>**. **<u>Resultado este, que indica a participação de variable-length records</u>**.

*(Resposta e complemento, retirados dos documentos das teóricas ("2-Storage") ,do Guia de Estudos (a partir da página 12 na secção de Storage) e de um dispositivo fornecido online pelo Técnico (disponível em:* https://fenix.tecnico.ulisboa.pt/downloadFile/3779571247626/Aula18-Structure-ch11.ppt)

3. **<u>A performance do acesso aos dados armazenados por um SGBD é negativamente afetada de forma mais significativa por:</u>**

a. Necessidade de obter esses dados a partir da memória

b. Necessidade de obter esses dados a partir do disco

c. Utilização de ficheiros secundários dispersos por mais de um disco

**d. Respostas b. e c.**

**<u>Complemento de Resposta</u>**:

Esta pergunta foi colocada em aula, mas não foi devidamente respondida, pelo que assumo, a partir da gravação da aula, que a resposta correta é a do ponto *d.*

4. **<u>Um Filegroup no SQLServer:</u>**

a. Contém necessariamente apenas um ficheiro físico associado

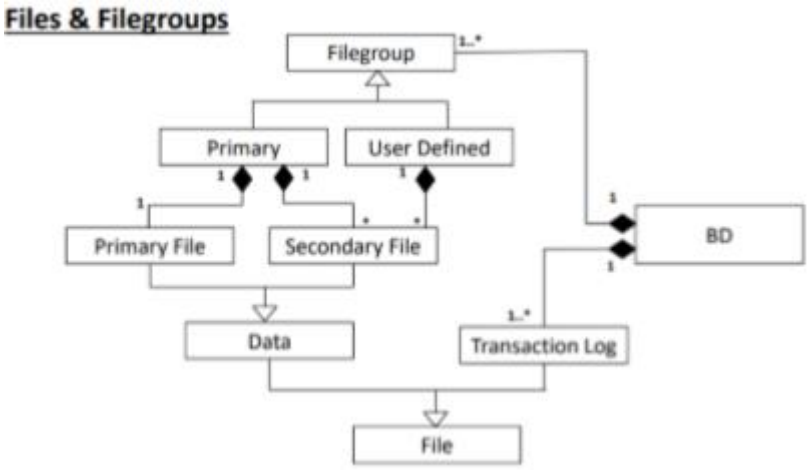
b. Contém ficheiros de dados de várias bases de dados

c. Contém somente ficheiros de dados

**d. Pode conter ficheiros de dados e metadados**

**<u>Complemento de Resposta:</u>**

Iniciando pelo ponto *a.*, um filegroup, pode-se dividir em diversos ficheiros físicos, sendo que apenas poderá conter um ficheiro primário, mas quanto a ficheiros secundários, até é benéfico conter múltiplos ficheiros secundários, de forma a dispersar os dados por vários discos (resumidamente, a resposta estaria correta se indicasse "*Contém necessariamente apenas um ficheiro primário associado*", o que não é o caso. Quanto ao ponto *b.*, um filegroup, apenas contem ficheiros de dados de apenas uma base de dados (uma relação de 1 para 1 entre Filegroup e BD, como representado na imagem seguinte)



Quanto ao ponto *c.* , podemos pegar no termo do Primary Filegroup, e dai, indicamos que este filegroup, contém o ficheiro primário e todas as tabelas de sistema (relativamente a metadados). Se esta explicação não é suficiente, o facto deste filegroup conter o ficheiro primário (Primary Data File), por si só já indica a presença de metadados, sendo que existe uma coexistência entre metadados e dados. (Esta indicação ao ponto *c.* , responde também ao ponto ***d.*** *(considerado o ponto correto)*.

*(Resposta e complemento, retirados unicamente do Guia de Estudos (a partir da página 16* (Files & Filegroups) *na secção de Storage).*

5. **Relativamente aos ficheiros: primário, secundário e de registo de transações (transaction log):**

a. São obrigatórios o Primário e Secundário e opcional o de registo de transações

b. É somente obrigatório o Primário sendo opcionais o Secundário e o de registo de transações

c. São todos obrigatórios

**d. Nenhuma das anteriores**

**Complemento de Resposta:**

Em relação a toda a resposta no geral, indicamos sempre que é obrigatório um ficheiro primário, embora apenas possa existir apenas um! ***("Existe apenas e obrigatoriamente um ficheiro primário por base de dados")***. Enquanto que os ficheiros secundários são definidos pelo utilizador, de maneira opcional. Quanto aos **transaction log files**, requer obrigatoriedade, com o mesmo número que o ficheiro primário, apenas um por base de dados. Estes files são compostos pela informação de "*logs"* que é utilizada na recuperação da base de dados. ("***The transaction log files hold the log information that is used to recover the database. There must be at least one log file for each database."*** *(SQL Server Documentation)*. Após estas observações, podemos concluir que nenhuma das duas primeiras opções (ponto *a.* e ponto *b.)*, estaria correta.

*(Resposta e complemento, retirados do Guia de Estudos (a partir da página 16* (Files & Filegroups) *na secção de Storage e da documentação do SQL Server (*https://docs.microsoft.com/en-us/sql/relational-databases/databases/database-files-and-filegroups?view=sql-server-ver15*)).*

6. **Sobre um índice clustered:**

a. Podem ser criados vários por tabela para otimizar consultas

**b. Apenas pode ser criado um por tabela**

c. Apenas pode ser criado um por tabela e somente na chave primária

d. Todas as anteriores

**Complemento de Resposta:**

Quanto ao ponto *c.*, apesar de indicar que pode ser criado um por tabela (o que é correto), indica também que é criado somente na chave primária, isto que contradiz o que é um índice clustered ("Índice Clustered é o índice sobre a chave de pesquisa que especifica a ordem sequencial do ficheiro de dados, **contudo esse campo não é chave primária**"). **Só é permitido existir um Índice Clustered por tabela**, porque as próprias linhas de dados, podem ser armazenadas numa ordem apenas.

*(Resposta e complemento, retirados do Guia de Estudos (a partir da página 20* (Índice Primário, Clustered e Secundário)*na secção de Índices e da documentação do SQL Server (*https://docs.microsoft.com/en-us/sql/relational-databases/indexes/clustered-and-nonclustered-indexes-described?view=sql-server-ver15*)).*

7. **O Filegroup atribuído a uma tabela:**

a. É sempre o Primário

**b. É sempre o Default**

**c. É sempre explicitamente definido pelo utilizador**

d. Nenhuma das anteriores

**Complemento de Resposta:**

Esta pergunta não possui uma composição correta, pois um filegroup, quando atribuído a uma tabela, pode ser explicitamente definido pelo utilizador, mas caso não exista essa definição, esse filegroup, será assumido como Default.

8. **<u>No ciclo de vida de execução de uma query a fase de "Parsing and Translation":</u>**

a. Realiza a validação sintática do SQL

b. É responsável pela tradução do SQL para um plano lógico de execução

**c. Respostas a. e b.**

d. Nenhuma das anteriores

**<u>Complemento da Resposta:</u>**

Quanto ao ciclo "Parsing and Translation" (representado por "Parser" no Guia de Estudos), aqui são realizadas duas tarefas principais. Estas tarefas são:

- A validação sintática (garantido que a query (SQL) está conformado com as relações da base de dados a que se refere – utilizando o catálogo);
- Traduz a *query* para uma árvore que representa as "*suboperações*" que terão de ser realizadas -> **Plano de Execução (ou avaliação).**

*(Resposta e complemento, retirados unicamente do Guia de Estudos (a partir da página 35* (Parser) *na secção de Processamento de Queries).*

9. **<u>Relativamente aos índices hash e B+-tree:</u>**

a. Os índices hash são necessariamente esparsos

b. Os índices B+-tree são esparsos

c. Os índices B+-tree são sempre primários

**d. Nenhuma das anteriores**

**<u>Complemento da Resposta</u>**:

Relativamente ao ponto *b.* , não está correto, pois índices B+-tree são sempre densos. Quanto ao ponto *c.* , os índices B+tree eliminam a redundância de armazenamento dos valores das chaves de pesquisa, o que contraria uma das definições do que é um índice primário ("a ordenação coincide com a ordenação do ficheiro de dados, pelo campo que é também chave primária"). E o facto de serem sempre densos, indica que por norma, estes índices são índices secundários (sendo que os índices secundários tem **sempre de ser densos**) Quanto ao ponto *a.* os índices hash não são necessariamente esparsos, devido a organizarem os valores das chaves de pesquisa com os respetivos apontadores, num ficheiro com uma estrutura de hash (esta estrutura de hash é composta por buckets, sendo que o acesso a esses buckets, é feito sequencialmente por ele próprio e não pela sequencia de chaves de pesquisa). Sendo que um índice esparso é aplicável quando os registos estão ordenados sequencialmente pela chave de pesquisa. Resposta correta, é a do ponto ***d.***

*(Resposta e complemento, retirados unicamente do Guia de Estudos (a partir da página 20 na secção de Índices (recomendo leitura da secção completa)).*

10. **No ciclo de vida de execução de uma query a fase de Otimização:**

a. Realiza a validação sintática do SQL

b. É responsável pela validação semântica do SQL

**c. Recorre a informação estatística sobre os acessos aos dados**

d. Todas as anteriores

**Complemento da Resposta:**

A fase de **Otimização** baseia-se nas estatísticas de distribuição, ao estimar os custos de recurso de método diferentes para extrair informações de uma tabela ou índice.

*(Resposta retirada com base em matéria expressa oralmente em aula, mas com maior base na documentação de SQL Server (https://docs.microsoft.com/pt-br/sql/relational-databases/query-processing-architecture-guide?view=sql-server-ver15)*

11. **Na comparação de um Heap File relativamente a um Hash File:**

a. Ambos têm desempenho semelhante em escrita

**b. O Heap File tem necessariamente melhor desempenho em leitura**

c. Respostas a. e b.

d. Nenhuma das anteriores

**Complementação da Resposta:**

Relativamente ao desempenho de leitura de um **Hash File**, neste, os registos são armazenados de forma aleatória, espalhados na memória. O que faz com que a memória não é usada de forma eficiente. Nesta organização de ficheiros, se tentarmos pesquisar por um intervalo de dados, não receberemos o endereço de memória correto, devido ao armazenamento aleatório, o que torna a procura desse intervalo ineficiente. Também, a procura de registos com o mesmo nome ou valor é ineficiente. Se o nome de um estudante começar por 'B', não vai ser eficiente devido a não retornar o nome exato desse estudante. Para esta organização, o hardware e o software necessários para a manutenção de memória são mais dispendiosos, e requerem a existência de programas complexos para tornar este método eficiente. Quanto a **Heap File**, este método de organização de ficheiros, é ineficiente apenas em bases de dados de grandes dimensões, devido ao tempo necessário para procurar/modificar o registo. Sendo que também é necessário uma devida manutenção da memória, para melhorar a performance, senão, existirá uma grande dimensão de blocos de memória não utilizados, enquanto o tamanho da memória acresce. A resposta correta neste caso é a resposta do ponto ***b.***

*(Resposta retirada com base em matéria expressa oralmente em aula, mas com maior base na documentação de TutorialCup (*https://www.tutorialcup.com/dbms/heap-file-organization.htm e para Hash File - https://www.tutorialcup.com/dbms/hash-file-organization.htm*).*

**12. <u>Considere o código das Figuras 1a e 1b:</u>**

a. O código de ambas produz o mesmo resultado

b. O código da Figura 1a indica as chaves primárias das respetivas tabelas

**c. O código da Figura 1b indica que tabelas têm chave primária**

d. Todas as anteriores

## Fig. 1a

```
select tt.TABLE_CATALOG+'.'+tt.TABLE_SCHEMA +'.' +
  tt.TABLE_NAME as rt,  tc.CONSTRAINT_NAME from
     INFORMATION_SCHEMA.TABLES tt Left join
   INFORMATION_SCHEMA.TABLE_CONSTRAINTS tc on
    tt.TABLE_SCHEMA=tc.CONSTRAINT_SCHEMA  AND
tc.CONSTRAINT_TYPE='PRIMARY KEY' WHERE tt.TABLE_TYPE='BASE
                  TABLE' order by rt
```

## Fig. 1b

```
   Select DB_Name()+'.'+OBJECT_SCHEMA_NAME(t.object_id)+'.'+
t.name as rt,  OBJECTPROPERTY(object_id,'TableHasPrimaryKey') as
                  'PK' From sys.tables t Where
  OBJECTPROPERTY(object_id,'TableHasPrimaryKey')=1 order by rt
```

*(Resposta provém de execução do código no MSSMS).*

# Exercícios de aula (respondidos oralmente no final de aula)

De acordo com a correção de exercícios de aula sobre processamento de queries e índices MS SQL, com recurso a uma transcrição de uma gravação voz-texto, obtivemos as seguintes respostas as seguintes perguntas:

**Processamento de Queries**

1. **Em que consiste a fase "Parsing & Translate"**

**Resposta**: Quanto a "Parsing" , este consiste na validação sintática, garantido que a query (ou SQL) está compatível com as relações a que a base de dados se refere. Quanto a "*Translate*" ou "*Translation*", consiste na tradução (como o nome indica) do SQL de linguagem de alto nível (ou uma *query*), para uma expressão de álgebra relacional (ou tradução de uma query para uma árvore).

**Transcrição direta voz-texto**: "Em relação ao parsing , tamos a falar de fazer uma validação sintática , portanto, se aquele SQL for submetido em compliant, portanto, se ta bem formado , se os objetos a que se refere (tabela, projeções dessa tabela, etc.) existem, se por exemplo, se existe um cláusula where , onde o domínio de valores é o daquele campo. Portanto, uma boa formação do SQL.A fase de translation, nesta fase , consiste na tradução deste SQL de linguagem de alto nível, para uma expressão de álgebra relacional. Essencialmente é um encadeamento sobre os operadores, que levam a cabo, as operações sobre os dados que permite devolver o que aquela query pressupõe do tempo de recolha de dados."

**Índices MS SQL**

1. **Quantos "Clustered Indexes" podem existir numa tabela?**

**Resposta**: De um modo breve, apenas pode existir 1 Clustered Index numa tabela.

**Transcrição direta voz-texto**: "Apenas 1. Clustered. Indexes supõem uma ordenação, e o ficheiro só estar ordenado pelos dados por um campo."

2. **Ter mais índices na tabela é sempre melhor? Porque?**

**Resposta:** Ter mais índices na tabela não é a melhor opção. Cada vez que é efetuada uma operação (inserção, atualização ou eliminação) sobre os registos de uma tabela, também é necessário existir uma atualização dos índices.

**Transcrição direta voz-texto:** "Não necessariamente. Porque cada vez que insiro, atualizo ou removo alguns registos da tabela, também é necessário propagar essas operações para uma atualização dos índices."

3. **Distinga composite index de cover index?**

**Resposta 1 (com base na transcrição):** O **Composite Index**, consiste numa indexação de tuplos, cujas entradas nos nós, tratam-se do tuplo de valores no índice, e esses indicam a ordenação (sem esquecer que a ordem interessa). Quanto ao **Cover Index**, este consiste em juntar, junto do valores indexado, outros valores do registo, com tudo o que não participe na definição da estrutura da indexação. Sendo este (**Composite Index**), vantajoso no facto de ter o índice em memoria, que permite ter junto dos valores indexados, outros valores de outros campos, que se forem esses os solicitados pela *query* que utiliza aquele índice, permite não ter que aceder aos dados, beneficiando uma existência e memória destes valores.

**Resposta 2 (com base no guia de estudos):** O Composite Index, resumidamente, indexa um **conjunto** de colunas (proveniente da nomenclatura "***Composite***", que indica composto). O **Cover Index** indica num índice non-clustered , informação adicional para ser armazenada juntamente no índice.

**Transcrição direta voz-texto:** "O cover index consiste em juntar, junto do valor indexado outros valores, outros campos, do registo, com tudo que não participe na definição da estrutura da indexação. Tem a vantagem de facto, tendo o índice em memória, consegue-se ter junto dos valores indexados, outros valores de outros campos, que se forem esses os solicitados pela query que utiliza aquele índice, permite não ter que aceder aos dados e, portanto, beneficiar de uma existência em memória destes valores. Para o composite index, o que estamos a dizer, é uma indexação de tuplos, como por exemplo, no caso concelhos e freguesias, não se trata de indexar concelhos numa árvore e indexar freguesias noutra árvore, neste caso, existe apenas uma árvore única, cujas entradas nos nós, tratam-se do tuplo de valores no índice, e esses indicam a ordenação, sem esquecer que a ordem interessa, pois ser um composite index concelho-freguesia, é diferente de freguesia-concelho."

## Potenciais perguntas extras:

1. O que é uma workload?

**Resposta:** Uma workload é um objeto de armazenamento que suporta uma aplicação. Sendo possível definir uma ou mais workloads (ou instâncias) por aplicação.

**Retirado de:**

https://mysupport.netapp.com/NOW/public/eseries/sam/index.html#page/GUID-8538272A-B802-49D9-9EA2-96C82DAD26A2/GUID-FFA076B1-0AFB-4490-967F-955070528236.html

2. Que ferramentas utilizo para gerar uma workload?

**Resposta:** Conforme discuto em aula, as ferramentas utilizadas para gerar uma workload, são o DTA (Data Tuning Advisor) e o SQL Profiler.