Cvičení 10: CASE a nejpoužívanější funkce

1) **S využitím jedné z textových funkcí přiřaďte k číslu, jménu a přijmení zákazníků z tabulky Customer také sloupec s počátečním písmenem jejich jména.** Poté k dotazu přidejte také sloupec s **počátečním písmenem příjmení** zákazníka.

2) **Napište dotaz, který vrátí všechny údaje k zákazníkům, jejichž jméno i příjmení začínají na stejné písmeno.**

3) **Napište dotaz, díky kterému zjistíte, na jaké písmeno začínají jména zákazníků nejčastěji.** Výsledkem dotazu tedy bude **seznam počátečních písmen jmen zákazníků** a k nim přiřazený **počet výskytů** v tabulce **Customer.**

4) **Napište dotaz, který vrátí seznam všech zákazníků s hodnotami ve formátu 'JMÉNO PŘÍJMENÍ, DATUM NAROZENÍ', tedy například 'PETR NOVÁK, 11.01.1991'. Výsledná tabulka tedy bude obsahovat pouze 1 sloupec s hodnotami v daném formátu.**

5) **Napište dotaz, který vrátí počet objednávek z tabulky Order na úrovni roku a stavu objednávky.**

6) **Napište dotaz, který vrátí počet objednávek z tabulky Order na úrovni roku a měsíce**. Bude nás zajímat pouze **TOP 5** záznamů s nejvyšším množstvím objednávek.

7) **Pro každého zákazníka z tabulky Order vytvořte shrnutí s datem jeho první objednávky, poslední objednávky a rozdílem mezi nimi (v měsících).**

8) **Za účelem kontroly plnění pole *Email* v tabulce Customer napište dotaz, který vrátí zákazníky, kteří mají jako hodnotu tohoto sloupce vyplněné číslo (a tedy ne text, jak by tomu správně být mělo).**

9) **Napište dotaz, který dle měsíce data objednávky definuje roční období, ve kterém byla objednávka provedena, a následně pro každé roční období vrátí celkový počet objednávek.** Pro zjednodušení roční období definujte dle celých 3 měsíců (jaro = březen + duben + květen, léto = červen + červenec + srpen, atd.)

10) **Pro účely segmentace zákazníků napište dotaz, ve kterém každého zákazníka zařadíte do určité skupiny dle doby, která uplynula od jeho registrace, a to následujícím způsobem:**

- ‘Nový zákazník – do 3 měsíců‘
- ‘Zákazníkem 3 až 12 měsíců‘
- ‘Zákazníkem 12 až 24 měsíců‘
- ‘Stálý zákazník – nad 24 měsíců‘
- ‘Neznámý‘

11) **Pro účely segmentace zákazníků napište dotaz, ve kterém každého zákazníka zařadíte do určité skupiny dle počtu jeho provedených objednávek, a to následujícím způsobem:**

- ‘0 objednávek‘
- ‘1 objednávka‘
- ‘2 – 50 objednávek‘
- ‘51 – 100 objednávek‘
- ‘100 + objednávek‘

12) **Napište dotaz, který z tabulky Product vrátí sloupce *Product_ID, Product_Category, Date_Start*, *Date_End* a nový sloupec *Product_Type*, jehož hodnoty budou následující:**

- ‘Ukončený produkt se známou kategorií‘
- ‘Ukončený produkt s neznámou kategorií‘
- ‘Aktivní produkt se známou kategorií‘
- ‘Aktivní produkt s neznámou kategorií‘
- ‘Neznámý‘

Pokuste se na základě daných hodnot definovat pravidla plnění tohoto sloupce.

13) **Naimportujte si do databáze data ze soubotu Functions_CSV a zkuste si i na nich fungování dalších funkcí, včetně těch ukázaných v lekci 28.**

Cvičení 10: CASE a nejpoužívanější funkce

1) **S využitím jedné z textových funkcí přiřaďte k číslu, jménu a přijmení zákazníků z tabulky Customer také sloupec s počátečním písmenem jejich jména.** Poté k dotazu přidejte také sloupec s **počátečním písmenem příjmení** zákazníka.

   Funkce **LEFT** vrací námi definovaný počet znaků zleva z určitého textového řetězce. Proto ji lze také využít k získání požadovaného prvního písmena jména a příjmení.

```
SELECT Customer_ID
      ,Customer_Name
      ,Customer_Surname
      ,LEFT(Customer_Name, 1) AS PrvníPísmenoJména
      ,LEFT(Customer_Surname, 1) AS PrvníPísmenoPříjmení
FROM [csv].[Customer]
```

2) **Napište dotaz, který vrátí všechny údaje k zákazníkům, jejichž jméno i příjmení začínají na stejné písmeno.**

   Zadání jinými slovy říká, že hodnoty sloupců ***PrvníPísmenoJména*** a ***PrvníPísmenoPříjmení*** z předchozí úlohy se ve vrácených záznamech mají shodovat. Použijeme tedy tuto podmínku v klauzuli **WHERE**.

```
SELECT *
FROM [csv].[Customer]
WHERE LEFT(Customer_Name, 1) = LEFT(Customer_Surname, 1)
```

	Customer_ID	Customer_Name	Customer_Surname	Gender	Date_Birth	Date_Registered	Cou
1	C100	Hana	Hrdinová	F	1992-05-18	2020-06-28 00:00:00.000	CZ
2	C102	Jindřiška	Jandová	F	1995-05-06	2020-04-08 00:00:00.000	CZ
3	C106	Juraj	Jirmásek	M	1968-03-25	2020-10-28 00:00:00.000	CZ
4	C130	Miroslava	Michalíčková	F	1999-08-04	2020-03-27 00:00:00.000	CZ
5	C132	Michaela	Makaturová	F	1998-07-09	2020-06-30 00:00:00.000	CZ
6	C137	Zuzana	Zachardová	F	1986-02-04	2020-05-17 00:00:00.000	CZ
7	C149	Stanislav	Sedláček	M	1983-06-16	2020-07-01 00:00:00.000	CZ
8	C173	Hana	Havlová	F	1999-01-23	2020-02-18 00:00:00.000	SV
9	C181	Petr	Potáček	M	1964-09-13	2020-04-26 00:00:00.000	DE
10	C185	Hana	Hrabovská	F	1986-08-12	2020-06-28 00:00:00.000	NU
11	C230	Barbora	Boturová	F	1982-02-11	2020-05-23 00:00:00.000	NU

3) **Napište dotaz, díky kterému zjistíte, na jaké písmeno začínají jména zákazníků nejčastěji.** Výsledkem dotazu tedy bude **seznam počátečních písmen jmen zákazníků** a k nim přiřazený **počet výskytů** v tabulce **Customer.**

Výrazy s SQL funkcemi lze využívat také v klauzuli **GROUP BY**. To můžeme vidět právě v dotazu níže, kde má požadovaný seznam obsahovat **množství záznamů** (to běžně získáváme pomocí agregační funkce **COUNT**) pro jednotlivá počáteční písmena, která získáváme právě funkcí **LEFT**. Seskupování dat v tabulce tedy proběhne podle výrazu **LEFT(Customer_Name, 1)**.

```
SELECT LEFT(Customer_Name, 1) AS PočátečníPísmeno
      ,COUNT(*)               AS PočetZákazníků
FROM [csv].[Customer]
GROUP BY LEFT(Customer_Name, 1)
ORDER BY PočetZákazníků DESC
```

	PočátečníPísmeno	PočetZákazníků
1	P	16
2	J	13
3	M	12
4	L	9
5	H	6
6	A	5

4) **Napište dotaz, který vrátí seznam všech zákazníků s hodnotami ve formátu 'JMÉNO PŘÍJMENÍ, DATUM NAROZENÍ', tedy například 'PETR NOVÁK, 11.01.1991'. Výsledná tabulka tedy bude obsahovat pouze 1 sloupec s hodnotami v daném formátu.**

Jelikož má být výsledná hodnota spojením několika textových řetězců, budeme používat funkci **CONCAT**. Začneme tím, že díky ní spojíme sloupec ***Customer_Name***, mezeru, ***Customer_Surname***, čárku a nakonec sloupec ***Date_Birth***. Protože použité sloupce budeme dále upravovat, je pro přehlednost každý argument funkce napsán na samostatný řádek. Mohly by ale být všechny na jednom.

```
SELECT CONCAT(Customer_Name
             ,' '
             ,Customer_Surname
             ,','
             ,Date_Birth
             ) AS Seznam
FROM [csv].[Customer]
```

	Seznam
1	Hana Hrdinová,1992-05-18
2	Hana Judová,1991-06-16
3	Jindřiška Jandová,1995-05-06
4	Petra Částková,1985-03-15
5	Věroslav Havránek,1967-07-27
6	Petr Vejvoda,1979-04-11
7	Juraj Jirmásek,1968-03-25
8	Iva Hrozková,1963-03-27
9	Martina Petrušelová,1994-07-04

Vidíme však, že hodnoty zatím neodpovídají zadání. Prvním důvodem je, že jméno a příjmení by měly být vráceny **velkými písmeny**. K tomu využijeme funkci **UPPER**.

```
SELECT CONCAT(UPPER(Customer_Name)
             ,' '
             ,UPPER(Customer_Surname)
             ,','
             ,Date_Birth
             ) AS Seznam
FROM [csv].[Customer]
```

	Seznam
1	HANA HRDINOVÁ,1992-05-18
2	HANA JUDOVÁ,1991-06-16
3	JINDŘIŠKA JANDOVÁ,1995-05-06
4	PETRA ČÁSTKOVÁ,1985-03-15
5	VĚROSLAV HAVRÁNEK,1967-07-27
6	PETR VEJVODA,1979-04-11
7	JURAJ JIRMÁSEK,1968-03-25
8	IVA HROZKOVÁ,1963-03-27
9	MARTINA PETRUŠELOVÁ,1994-07-04

Druhým, a posledním zbývajícím, důvodem je **datum narození**, které bychom rádi získali ve formátu **DD.MM.RRRR**. Tento formát je v SQL definován jako datumový styl 104 a lze na něj převést jakékoliv datum pomocí funkce **CONVERT**.

```
SELECT CONCAT(UPPER(Customer_Name)
              ,' '
              ,UPPER(Customer_Surname)
              ,','
              ,CONVERT (VARCHAR(10), Date_Birth, 104)
              ) AS Seznam
FROM [csv].[Customer]
```

	Seznam
1	HANA HRDINOVÁ,18.05.1992
2	HANA JUDOVÁ,16.06.1991
3	JINDŘIŠKA JANDOVÁ,06.05.1995
4	PETRA ČÁSTKOVÁ,15.03.1985
5	VĚROSLAV HAVRÁNEK,27.07.1967
6	PETR VEJVODA,11.04.1979
7	JURAJ JIRMÁSEK,25.03.1968
8	IVA HROZKOVÁ,27.03.1963
9	MARTINA PETRUŽELOVÁ,04.07.1994

5) Napište dotaz, který vrátí počet objednávek z tabulky Order na úrovni roku a stavu objednávky.

K tabulce **Order** nejdříve připojíme tabulku **OrderStatus**, abychom k objednávkám mohli přiřadit jejich (slovně vyjádřený) stav. Dále použijeme funkci **YEAR**, která z data objednávky vezme pouze **rok**. K tomuto roku a stavu objednávky si můžeme přidat všechny ostatní záznamy z tabulky **Order**, abychom si zkontrolovali, že funkce vrací opravdu údaje, které chceme.

```
SELECT YEAR(ord.Order_Date) AS Rok
      ,status.OrderStatus_Name AS Status
      ,ord.*
FROM [csv].[Order] ord
LEFT JOIN [dbo].[OrderStatus] status
ON ord.Order_Status = status.OrderStatus_ID
```

Pokud je vše v pořádku, výraz **ord.*** smažeme a začneme se **seskupováním dat**. To má probíhat na úrovni **roku** i **stavu objednávky**, a proto oba tyto sloupce zahrneme do klauzule **GROUP BY**.

Následně také do klauzule **SELECT** přidáme kalkulovaný sloupec (např. s názvem ***PočetObjednávek***), který určí počet záznamů spadající do konkrétní kategorie **rok – stav objednávky**. Nakonec můžeme pro přehlednost data seřadit dle **roku**.

```
SELECT YEAR(ord.Order_Date) AS Rok
      ,status.OrderStatus_Name AS Status
      ,COUNT(ord.Order_ID) AS PočetObjednávek
FROM [csv].[Order] ord
LEFT JOIN [dbo].[OrderStatus] status
ON ord.Order_Status = status.OrderStatus_ID
GROUP BY YEAR(ord.Order_Date), status.OrderStatus_Name
ORDER BY Rok
```

	Rok	Status	PočetObjednávek
1	2020	Zrušená	207
2	2020	Nezaplacená	860
3	2020	Zaplacená	3704
4	2021	Zrušená	61
5	2021	Nezaplacená	219
6	2021	Zaplacená	962

6) **Napište dotaz, který vrátí počet objednávek z tabulky Order na úrovni roku a měsíce.** Bude nás zajímat pouze **TOP 5** záznamů s nejvyšším množstvím objednávek.

Dotaz z předchozí úlohy upravíme tak, že odmažeme vazbu na tabulku **OrderStatus** a do klauzule **SELECT** přidáme sloupec ***Měsíc***, který bude obdobně jako ***Rok*** definován časovou funkcí, konkrétně **MONTH**. Oba sloupce ***Rok*** i ***Měsíc*** poté použijeme v klauzuli **GROUP BY**. Nakonec stačí přidat výraz **TOP 5** a klauzulí **ORDER BY** seřadit záznamy **sestupně** dle **počtu objednávek**.

```
SELECT TOP 5 YEAR(ord.Order_Date)    AS Rok
            ,MONTH(ord.Order_Date)   AS Měsíc
            ,COUNT(*)                AS PočetObjednávek
FROM [csv].[Order] ord
GROUP BY YEAR(ord.Order_Date), MONTH(ord.Order_Date)
ORDER BY PočetObjednávek DESC
```

	Rok	Měsíc	PočetObjednávek
1	2020	6	508
2	2020	11	482
3	2020	12	477
4	2020	7	476
5	2020	3	466

7) **Pro každého zákazníka z tabulky Order vytvořte shrnutí s datem jeho první objednávky, poslední objednávky a rozdílem mezi nimi (v měsících).**

Záznamy v tabulce seskupíme podle zákazníků, a k těm poté pomocí agregačních funkcí **MIN** a **MAX** nalezneme nejdřívější a nejpozdější datum. Pro výpočet rozdílu mezi těmito dvěma daty použijeme funkci **DATEDIFF**, ve které v argumentu uvedeme výraz **MONTH**, abychom výsledný rozdíl získali v **měsících**.

```
SELECT Customer_ID
      ,MIN(Order_Date)                                  AS PrvníObjednávka
      ,MAX(Order_Date)                                  AS PosledníObjednávka
      ,DATEDIFF(MONTH,MIN(Order_Date),MAX(Order_Date))  AS Rozdíl_Měsíce
FROM [csv].[Order] ord
GROUP BY Customer_ID
```

8) **Za účelem kontroly plnění pole *Email* v tabulce Customer napište dotaz, který vrátí zákazníky, kteří mají jako hodnotu tohoto sloupce vyplněné číslo (a tedy ne text, jak by tomu správně být mělo).**

Abychom našli záznamy s číselným údajem ve sloupci **Email**, použijeme funkci **ISNUMERIC**, která vrací hodnotu **1**, pokud záznam číslem je, a nebo **0**, pokud není.

```
SELECT Customer_ID
      ,Email
FROM [csv].[Customer]
WHERE ISNUMERIC(Email) = 1
```

	Customer_ID	Email
1	C111	717490491
2	C114	716932620
3	C117	712085579
4	C124	736617379
5	C184	711210623
6	C189	712774487

Dané záznamy vypadají, že pravděpodobně omylem obsahují telefonní číslo klienta místo emialu. Proto bychom hodnotu sloupce ***Email*** mohli vložit do sloupce ***Phone***, ale pouze v případech, kde je splněna **předchozí podmínka** a zároveň je sloupec ***Phone* prázdný**.

```
UPDATE [csv].[Customer]
SET Phone = Email
WHERE ISNUMERIC(Email) = 1 AND Phone IS NULL
```

Tím jsme doplnili telefon ke všem z těchto záznamů, kromě jednoho, který již vyplněné pole ***Phone*** měl.

	Customer_ID	Email	Phone
1	C111	717490491	717490491
2	C114	716932620	716932620
3	C117	712085579	712085579
4	C124	736617379	737589713
5	C184	711210623	711210623
6	C189	712774487	712774487

9) **Napište dotaz, který dle měsíce data objednávky definuje roční období, ve kterém byla objednávka provedena, a následně pro každé roční období vrátí celkový počet objednávek.** Pro zjednodušení roční období definujte dle celých 3 měsíců (jaro = březen + duben + květen, léto = červen + červenec + srpen, atd.)

Nejdříve si můžeme spustit dotaz, který vrátí seznam objednávek, jejich data a přiřazeného ročního období, které bude určeno na základě podmínek ve výrazu **CASE** s použitím funkce **MONTH** (vracející měsíc z určitého data).

```
SELECT Order_ID
      ,Order_Date
      ,CASE WHEN MONTH(Order_Date) IN (3, 4, 5)     THEN 'Jaro'
            WHEN MONTH(Order_Date) IN (6, 7, 8)     THEN 'Léto'
            WHEN MONTH(Order_Date) IN (9, 10, 11)   THEN 'Podzim'
            WHEN MONTH(Order_Date) IN (12, 1, 2)    THEN 'Zima'
       END AS RočníObdobí
FROM [csv].[Order]
```

	Order_ID	Order_Date	RočníObdobí
1	O1000	2020-06-05	Léto
2	O1001	2020-06-21	Léto
3	O1002	2020-10-05	Podzim
4	O1003	2020-07-05	Léto
5	O1004	2021-03-22	Jaro
6	O1005	2020-08-09	Léto

Vidíme, že výraz **CASE** vrací správné hodnoty, a můžeme ho tedy použít pro další krok. Tím je seskupení dat z tabulky dle ročního období, které jsme právě získali.

První možností je vložit celý výraz **CASE** do klauzule **GROUP BY** a následně v rámci klauzule **SELECT** nechat vrátit pouze napočítávané roční období spolu s počtem objednávek získaným pomocí agregační funkce **COUNT**.

```
SELECT CASE WHEN MONTH(Order_Date) IN (3, 4, 5)    THEN 'Jaro'
            WHEN MONTH(Order_Date) IN (6, 7, 8)    THEN 'Léto'
            WHEN MONTH(Order_Date) IN (9, 10, 11)  THEN 'Podzim'
            WHEN MONTH(Order_Date) IN (12, 1, 2)   THEN 'Zima'
       END        AS RočníObdobí
      ,COUNT(*)   AS PočetObjednávek
FROM [csv].[Order]
GROUP BY (CASE WHEN MONTH(Order_Date) IN (3, 4, 5)    THEN 'Jaro'
               WHEN MONTH(Order_Date) IN (6, 7, 8)    THEN 'Léto'
               WHEN MONTH(Order_Date) IN (9, 10, 11)  THEN 'Podzim'
               WHEN MONTH(Order_Date) IN (12, 1, 2)   THEN 'Zima'
               END)
```

	RočníObdobí	PočetObjednávek
1	Jaro	1741
2	Léto	1424
3	Podzim	1406
4	Zima	1442

Daný dotaz vrátí potřebné hodnoty a není zapotřebí používat složitější metody jako v případě druhé možnosti, kde se na původní dotaz budeme odkazovat pomocí **poddotazu**. Tato varianta vede ke stejnému výsledku, jen by byla **pomalejší**.

```
SELECT RočníObdobí
      ,COUNT(*) AS PočetObjednávek
FROM
(
SELECT Order_ID
      ,Order_Date
      ,CASE WHEN MONTH(Order_Date) IN (3, 4, 5)     THEN 'Jaro'
            WHEN MONTH(Order_Date) IN (6, 7, 8)     THEN 'Léto'
            WHEN MONTH(Order_Date) IN (9, 10, 11)   THEN 'Podzim'
            WHEN MONTH(Order_Date) IN (12, 1, 2)    THEN 'Zima'
       END AS RočníObdobí
FROM [csv].[Order]
) a
GROUP BY RočníObdobí
```

	RočníObdobí	PočetObjednávek
1	Jaro	1741
2	Léto	1424
3	Podzim	1406
4	Zima	1442

10) **Pro účely segmentace zákazníků napište dotaz, ve kterém každého zákazníka zařadíte do určité skupiny dle doby, která uplynula od jeho registrace, a to následujícím způsobem:**

- ‘Nový zákazník – do 3 měsíců‘
- ‘Zákazníkem 3 až 12 měsíců‘
- ‘Zákazníkem 12 až 24 měsíců‘
- ‘Stálý zákazník – nad 24 měsíců‘
- ‘Neznámé datum registrace‘

Aktuální datum získáme funkcí **GETDATE()**, kterou použijeme ve funkci **DATEDIFF** s argumentem **MONTH** k napočítání počtu měsíců od registrace daného zákazníka. Dle této hodnoty poté definujeme segmenty zákazníků ve výrazu **CASE**.

```
SELECT [Customer_ID]
      ,[Date_Registered]
      ,DATEDIFF(MONTH, [Date_Registered], GETDATE()) AS PočetMěsícůOdRegistrace
      ,CASE WHEN DATEDIFF(MONTH, [Date_Registered], GETDATE()) < 3 THEN 'Nový zákazník – do 3 měsíců'
            WHEN DATEDIFF(MONTH, [Date_Registered], GETDATE()) < 12 THEN 'Zákazníkem 3 až 12 měsíců'
            WHEN DATEDIFF(MONTH, [Date_Registered], GETDATE()) < 24 THEN 'Zákazníkem 12 až 24 měsíců'
            WHEN DATEDIFF(MONTH, [Date_Registered], GETDATE()) >= 24 THEN 'Stálý zákazník – nad 24 měsíců'
            ELSE 'Neznámé datum registrace'
            END AS SegmentDobaZakaznik
  FROM [csv].[Customer]
```

	Customer_ID	Date_Registered	PočetMěsícůOdRegistrace	SegmentDobaZakaznik
1	C100	2020-06-28 00:00:00.000	10	Zákazníkem 3 až 12 měsíců
2	C101	2020-07-18 00:00:00.000	9	Zákazníkem 3 až 12 měsíců
3	C102	2020-04-08 00:00:00.000	12	Zákazníkem 12 až 24 měsíců
4	C103	2020-10-31 00:00:00.000	6	Zákazníkem 3 až 12 měsíců
5	C104	2020-03-01 00:00:00.000	13	Zákazníkem 12 až 24 měsíců
6	C105	2020-12-06 00:00:00.000	4	Zákazníkem 3 až 12 měsíců

11) **Pro účely segmentace zákazníků napište dotaz, ve kterém každého zákazníka zařadíte do určité skupiny dle počtu jeho provedených objednávek, a to následujícím způsobem:**

- ‘0 objednávek‘
- ‘1 objednávka‘
- ‘2 – 50 objednávek‘
- ‘51 – 100 objednávek‘
- ‘100 + objednávek‘

V této úloze je ke všem zákazníkům z tabulky **Customer** potřeba připojit celkový počet jimi provedených objednávek. Ten je možné napočítat z tabulky **Order**, kterou pomocí **subselectu** (poddotazu) připojíme k tabulce zákazníků.

Poté již napočítané hodnoty počtu provedených objednávek použijeme ve výrazu **CASE**, abychom každého zákazníka zařadili do správného segmentu.

Nenajdeme-li pro nějaké ***Customer_ID*** žádnou objednávku v tabulce **Order**, budou mít všechny sloupce z připojovaného subselectu hodnotu **NULL**, a proto ji také použijeme ve výrazu **CASE** jako podmínku pro zákazníky kategorie **'0 objednávek'.**

```
SELECT cust.Customer_ID
      ,ord.PočetObjednávek
      ,CASE WHEN ord.PočetObjednávek IS NULL THEN '0 objednávek'
            WHEN ord.PočetObjednávek = 1 THEN '1 objednávka'
            WHEN ord.PočetObjednávek BETWEEN 2 AND 50 THEN '2 - 50 objednávek'
            WHEN ord.PočetObjednávek BETWEEN 51 AND 100 THEN '51 - 100 objednávek'
            WHEN ord.PočetObjednávek > 100 THEN '100 + objednávek'
            END AS SegmentObjednavkyZakaznik
FROM [csv].[Customer] cust
LEFT JOIN
( SELECT Customer_ID
        ,COUNT(*) AS PočetObjednávek
  FROM [csv].[Order]
  GROUP BY [Customer_ID] ) ord
ON cust.Customer_ID = ord.Customer_ID
```

	Customer_ID	PočetObjednávek	SegmentObjednavkyZakaznik
1	C100	64	51 - 100 objednávek
2	C101	63	51 - 100 objednávek
3	C102	48	2 - 50 objednávek
4	C103	57	51 - 100 objednávek
...			
96	C195	51	51 - 100 objednávek
97	C230	NULL	0 objednávek
98	C231	NULL	0 objednávek
99	C233	NULL	0 objednávek

12) Napište dotaz, který z tabulky Product vrátí sloupce *Product_ID, Product_Category, Date_Start*, *Date_End* a nový sloupec *Product_Type*, jehož hodnoty budou následující:

- 'Ukončený produkt se známou kategorií'
- 'Ukončený produkt s neznámou kategorií'
- 'Aktivní produkt se známou kategorií'
- 'Aktivní produkt s neznámou kategorií'
- 'Neznámý'

Pokuste se na základě daných hodnot definovat pravidla plnění tohoto sloupce.

Hodnoty sloupce ***Product_Type*** závisí na ukončení prodeje produktu (***Date_End***) a známé/ neznámé kategorii (***Product_Category***). Můžeme tedy využít výraz **CASE** pro definování pravidel tak, aby výsledné hodnoty odpovídaly zadání.

Pokud je tedy například pole ***Date_End*** vyplněné neprázdnou hodnotou (prodej produktu byl tedy k nějakému datu ukončen) a zároveň hodnota sloupce ***Product_Category*** **není 'Unknown'**, jedná se o **ukončený produkt** se **známou kategorií**.

```
SELECT Product_ID
      ,Product_Category
      ,Date_Start
      ,Date_End
      ,CASE WHEN Product_Category <> 'Unknown' AND Date_End IS NOT NULL
            THEN 'Ukončený produkt se známou kategorií'
            WHEN Product_Category = 'Unknown' AND Date_End IS NOT NULL
            THEN 'Ukončený produkt s neznámou kategorií'
            WHEN Product_Category <> 'Unknown' AND Date_End IS NULL
            THEN 'Aktivní produkt se známou kategorií'
            WHEN Product_Category = 'Unknown' AND Date_End IS NULL
            THEN 'Aktivní produkt s neznámou kategorií'
            ELSE 'Neznámý'
            END AS Product_Type
FROM [csv].[Product]
```

Pokud bychom chtěli zjistit, jaké unikátní hodnoty sloupce ***Product_Type*** se ve výsledné tabulce skutečně vysktují, lze použít klauzuli **SELECT DISTINCT** na hodnoty tohoto sloupce, i když je definovaný výrazem **CASE**.

Dotaz níže vrátí pouze dvě hodnoty, ze kterých vyplývá, že pokud je produkt **ukončený** (má vyplněné pole ***Date_End***), tak má zároveň **kategorii** nastavenou na **'Unknown'** – tedy ***Product_Type*** = **'Ukončený produkt s neznámou kategorií'**.

Není-li produkt ukončený, má vždy kategorii jinou než 'Unknown' – tedy ***Product_Type*** = **'Aktivní produkt se známou kategorií'**.

```
SELECT DISTINCT CASE WHEN Product_Category <> 'Unknown' AND Date_End IS NOT NULL
                     THEN 'Ukončený produkt se známou kategorií'
                     WHEN Product_Category = 'Unknown' AND Date_End IS NOT NULL
                     THEN 'Ukončený produkt s neznámou kategorií'
                     WHEN Product_Category <> 'Unknown' AND Date_End IS NULL
                     THEN 'Aktivní produkt se známou kategorií'
                     WHEN Product_Category = 'Unknown' AND Date_End IS NULL
                     THEN 'Aktivní produkt s neznámou kategorií'
                     ELSE 'Neznámý'
                     END AS Product_Type
FROM [csv].[Product]
```

	Product_Type
1	Aktivní produkt se známou kategorií
2	Ukončený produkt s neznámou kategorií