

Zaawansowane systemy baz danych

Projekt – etap 2

Krzysztof Wyszyński

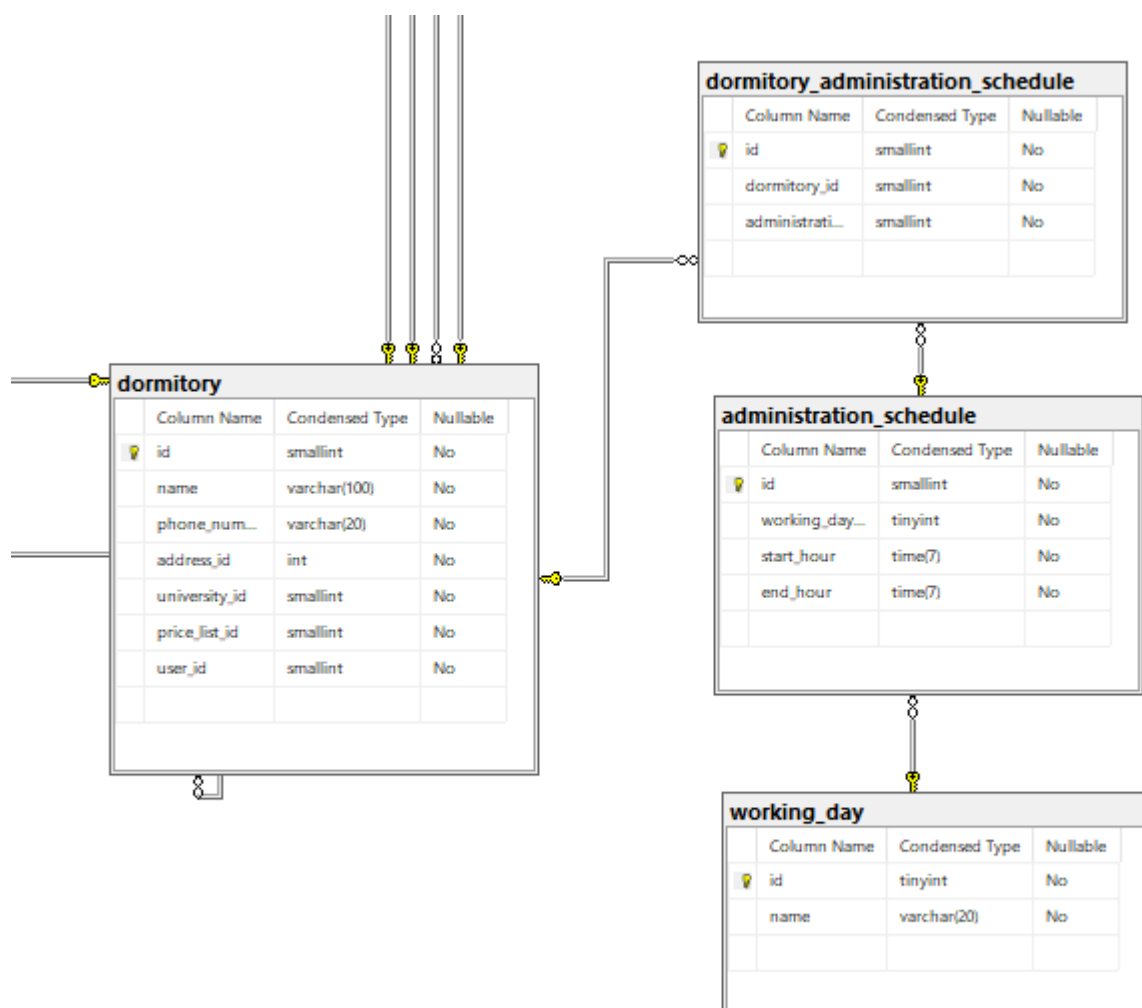
W etapie drugim baza danych uzupełniona została większą ilością danych – w każdym Domu Studenckim znajduje się 200 pomieszczeń. W każdym pomieszczeniu znajduje się jeden student. W związku z tym łączna liczba studentów w bazie wynosi 2600.

Podczas analizy etapu 1. wystąpiły błędy, które skorygowane zostały w etapie drugim:

- a) Relacja degree/department/university – wydział może mieć wiele kierunków, a uczelnia wiele wydziałów (dwie relacje one-to-many)
- b) Kolumna *number* w tabeli *room* nie jest już typu SMALLINT – typ VARCHAR(10) umożliwi wstawianie liter do numeru pokoju, np. 23A
- c) Typ CHAR(9) dla numeru telefonu nie jest dobrym rozwiązaniem – znacząco ogranicza zakres wprowadzanych wartości oraz nie uwzględnia numerów kierunkowych. Został zastąpiony przez VARCHAR(20)
- d) Usunięto flagę *is_working* z tabeli *doorkeeper*
- e) Zmodyfikowano typy danych w tabeli *price_list* – uwzględniono możliwość wprowadzania liczb zmiennoprzecinkowych

Pozostałe modyfikacje:

- a) W tabeli *dormitory* znajduje się kolumna *user_id*, dzięki której możliwe będzie zastosowanie RLS (Row Level Security) – ograniczenie dostępu do danych w tabeli dla poszczególnych użytkowników
- b) Utworzono tabele zawierające godziny pracy administracji dla poszczególnych Domów Studenckich:



1. Perspektywy

W celu udostępnienia użytkownikom czytelnej, łatwo dostępnej informacji wygenerowane zostały następujące widoki:

- a) *dormitories* – informacje o danych adresowych Domów Studenckich oraz godzinach pracy administracji w poszczególne dni i numerze kontaktowym. Może zostać wykorzystana przez dowolnego użytkownika, jako informacja powszechnie dostępna

	Dormitory name ▾	Phone number ▾	University name ▾	City ▾	Street ▾	ZIP ▾	Day ▾	Start hour ▾	End hour ▾
1	Dom Studencki Bratniak-Musze...	111222333	Politechnika Warszawska	Warszawa	Grójecka 39	02-031	Poniedziałek	10:00:00	16:00:00
2	Dom Studencki Bratniak-Musze...	111222333	Politechnika Warszawska	Warszawa	Grójecka 39	02-031	Wtorek	10:00:00	16:00:00
3	Dom Studencki Bratniak-Musze...	111222333	Politechnika Warszawska	Warszawa	Grójecka 39	02-031	Środa	10:00:00	16:00:00
4	Dom Studencki Bratniak-Musze...	111222333	Politechnika Warszawska	Warszawa	Grójecka 39	02-031	Czwartek	10:00:00	16:00:00
5	Dom Studencki Bratniak-Musze...	111222333	Politechnika Warszawska	Warszawa	Grójecka 39	02-031	Piątek	10:00:00	16:00:00
6	Dom Studencki Ustronie	222333444	Politechnika Warszawska	Warszawa	Księcia Janusza 39	01-452	Poniedziałek	10:00:00	16:00:00
7	Dom Studencki Ustronie	222333444	Politechnika Warszawska	Warszawa	Księcia Janusza 39	01-452	Wtorek	10:00:00	16:00:00
8	Dom Studencki Ustronie	222333444	Politechnika Warszawska	Warszawa	Księcia Janusza 39	01-452	Środa	10:00:00	16:00:00
9	Dom Studencki Ustronie	222333444	Politechnika Warszawska	Warszawa	Księcia Janusza 39	01-452	Czwartek	10:00:00	16:00:00
10	Dom Studencki Ustronie	222333444	Politechnika Warszawska	Warszawa	Księcia Janusza 39	01-452	Piątek	10:00:00	16:00:00
11	Dom Studencki Mikrus	333444555	Politechnika Warszawska	Warszawa	Ludwika Waryńskiego 10	00-631	Poniedziałek	10:00:00	16:00:00
12	Dom Studencki Mikrus	333444555	Politechnika Warszawska	Warszawa	Ludwika Waryńskiego 10	00-631	Wtorek	10:00:00	16:00:00
13	Dom Studencki Mikrus	333444555	Politechnika Warszawska	Warszawa	Ludwika Waryńskiego 10	00-631	Środa	10:00:00	16:00:00
14	Dom Studencki Mikrus	333444555	Politechnika Warszawska	Warszawa	Ludwika Waryńskiego 10	00-631	Czwartek	10:00:00	16:00:00
15	Dom Studencki Mikrus	333444555	Politechnika Warszawska	Warszawa	Ludwika Waryńskiego 10	00-631	Piątek	10:00:00	16:00:00
16	Dom Studencki Akademik	444555666	Politechnika Warszawska	Warszawa	Akademicka 5	02-038	Poniedziałek	08:00:00	15:00:00
17	Dom Studencki Akademik	444555666	Politechnika Warszawska	Warszawa	Akademicka 5	02-038	Wtorek	08:00:00	15:00:00
18	Dom Studencki Akademik	444555666	Politechnika Warszawska	Warszawa	Akademicka 5	02-038	Środa	08:00:00	15:00:00
19	Dom Studencki Akademik	444555666	Politechnika Warszawska	Warszawa	Akademicka 5	02-038	Czwartek	08:00:00	15:00:00
20	Dom Studencki Akademik	444555666	Politechnika Warszawska	Warszawa	Akademicka 5	02-038	Piątek	08:00:00	15:00:00
21	Dom Studencki nr 2	111111111	Uniwersytet Warszawski	Warszawa	Karolkowa 84	01-193	Poniedziałek	08:00:00	15:00:00
22	Dom Studencki nr 2	111111111	Uniwersytet Warszawski	Warszawa	Karolkowa 84	01-193	Wtorek	08:00:00	15:00:00
23	Dom Studencki nr 2	111111111	Uniwersytet Warszawski	Warszawa	Karolkowa 84	01-193	Środa	08:00:00	15:00:00
24	Dom Studencki nr 2	111111111	Uniwersytet Warszawski	Warszawa	Karolkowa 84	01-193	Czwartek	08:00:00	15:00:00
25	Dom Studencki nr 2	111111111	Uniwersytet Warszawski	Warszawa	Karolkowa 84	01-193	Piątek	08:00:00	15:00:00
26	Dom Studencki nr 1	228360361	Uniwersytet Warszawski	Warszawa	Batalionu AK Pięć 9	01-406	Poniedziałek	10:00:00	15:30:00

- b) *students* – informacje o studentach zamieszkujących Domy Studenckie. Może zostać wykorzystana przez pracowników administracji poszczególnych akademików

	First name	Last name	PESEL	Street	Flat number	ZIP	Country	Term	Department	University	Room number	Dormitory
1	Rosette	Bentley	95091961976	461-3731 Leo, _	NULL	73-278	Poland	2	Wydział Elektryczny	Politechnika Warszawska	100	Dom Studencki Bratniak-Mu...
2	Fabien	Pigden	95030429965	Ap #759-2681 S...	NULL	15-370	Poland	1	Wydział Elektryczny	Politechnika Warszawska	100	Dom Studencki Bratniak-Mu...
3	Jarad	Baudet	95083016338	P.O. Box 635, _	NULL	81-981	Poland	2	Wydział Elektryczny	Politechnika Warszawska	102	Dom Studencki Bratniak-Mu...
4	Stefano	Hofner	95060848756	245-4289 Tellu...	NULL	65-837	Poland	1	Wydział Elektryczny	Politechnika Warszawska	103	Dom Studencki Bratniak-Mu...
5	Gusti	MacCaughan	95032923214	Ap #293-3765 E...	NULL	54-581	Poland	3	Wydział Elektryczny	Politechnika Warszawska	104	Dom Studencki Bratniak-Mu...
6	Kaycee	Thyng	95071668949	830-6215 Sem R...	NULL	98-233	Poland	2	Wydział Elektronik...	Politechnika Warszawska	105	Dom Studencki Bratniak-Mu...
7	Sergeant	Handrek	95033035246	927-1812 Sapie...	NULL	52-088	Poland	1	Wydział Elektronik...	Politechnika Warszawska	106	Dom Studencki Bratniak-Mu...
8	Earvin	Alfvy	95121461281	227-3037 Enim _	NULL	12-004	Poland	3	Wydział Elektronik...	Politechnika Warszawska	107	Dom Studencki Bratniak-Mu...
9	Lauretta	Pipworth	95110752525	Ap #537-5491 P...	NULL	41-382	Poland	1	Wydział Elektryczny	Politechnika Warszawska	108	Dom Studencki Bratniak-Mu...
10	Sheffield	Smewing	95112237664	910 Quis Rd.	NULL	19-516	Poland	2	Wydział Elektryczny	Politechnika Warszawska	109	Dom Studencki Bratniak-Mu...
11	Austina	Millington	95121758316	8336 Consectet...	NULL	62-932	Poland	3	Wydział Elektronik...	Politechnika Warszawska	110	Dom Studencki Bratniak-Mu...
12	Brander	Hunnam	95072534122	6286 Natoque S...	NULL	24-986	Poland	3	Wydział Elektronik...	Politechnika Warszawska	111	Dom Studencki Bratniak-Mu...
13	Rayshell	Roobottom	95121153777	P.O. Box 355, _	NULL	83-366	Poland	2	Wydział Elektryczny	Politechnika Warszawska	112	Dom Studencki Bratniak-Mu...
14	Yolanthe	Gannaway	95081743328	992-1726 Donec...	NULL	03-839	Poland	2	Wydział Elektronik...	Politechnika Warszawska	113	Dom Studencki Bratniak-Mu...
15	Rozelle	Render	95032132111	622-7062 Tinci...	NULL	07-564	Poland	1	Wydział Elektryczny	Politechnika Warszawska	114	Dom Studencki Bratniak-Mu...
16	Godfree	MacGorrie	95121179724	374-5187 Ut Av.	NULL	43-115	Poland	2	Wydział Elektryczny	Politechnika Warszawska	115	Dom Studencki Bratniak-Mu...
17	Breena	Reavey	95070288955	Ap #418-979 Fa...	NULL	23-509	Poland	3	Wydział Elektryczny	Politechnika Warszawska	116	Dom Studencki Bratniak-Mu...
18	Norry	Dessaur	96022728389	Ap #253-4295 N...	NULL	58-017	Poland	3	Wydział Elektronik...	Politechnika Warszawska	117	Dom Studencki Bratniak-Mu...
19	Iolanthe	Kauffman	95122923612	577-6365 Gravi...	NULL	60-739	Poland	3	Wydział Elektronik...	Politechnika Warszawska	118	Dom Studencki Bratniak-Mu...
20	Haily	Satterthwai...	95102672415	9207 Pede Stre...	NULL	40-761	Poland	1	Wydział Elektryczny	Politechnika Warszawska	119	Dom Studencki Bratniak-Mu...
21	Eadie	Drissell	95072549656	5049 Augue Rd.	NULL	28-884	Poland	2	Wydział Elektryczny	Politechnika Warszawska	120	Dom Studencki Bratniak-Mu...
22	Ileane	Giacobilio	95082039796	385-1265 Metus...	NULL	52-021	Poland	2	Wydział Elektryczny	Politechnika Warszawska	121	Dom Studencki Bratniak-Mu...
23	Adolf	Aguirrezaba...	95101147745	Ap #898-2526 N...	NULL	18-898	Poland	1	Wydział Elektronik...	Politechnika Warszawska	122	Dom Studencki Bratniak-Mu...
24	Trudy	Beamond	95121384924	P.O. Box 320, _	NULL	25-456	Poland	1	Wydział Elektryczny	Politechnika Warszawska	123	Dom Studencki Bratniak-Mu...
25	Arlin	Ricciardiel...	95050859964	791-4770 Ultri...	NULL	72-111	Poland	3	Wydział Elektronik...	Politechnika Warszawska	124	Dom Studencki Bratniak-Mu...
26	Agustin	Saulter	95021445453	4059 Amet, Str...	NULL	14-621	Poland	3	Wydział Elektryczny	Politechnika Warszawska	125	Dom Studencki Bratniak-Mu...
27	Daryl	Sands-Allan	95060562465	857-875 Faucib...	NULL	83-644	Poland	2	Wydział Elektryczny	Politechnika Warszawska	126	Dom Studencki Bratniak-Mu...
28	Wilhelm	Cunniam	95102624979	Ap #427-3828 A...	NULL	40-068	Poland	2	Wydział Elektryczny	Politechnika Warszawska	127	Dom Studencki Bratniak-Mu...

c) *rooms* – informacje o pojemności pokoju, jego numerze, nazwie Domu Studenckiego do którego należy oraz aktualnej liczbie lokatorów

	number	capacity	name	Locators
1	100	1	Dom Studencki Akademik	1
2	101	2	Dom Studencki Akademik	1
3	102	3	Dom Studencki Akademik	1
4	103	4	Dom Studencki Akademik	1
5	104	5	Dom Studencki Akademik	1
6	105	1	Dom Studencki Akademik	1
7	106	2	Dom Studencki Akademik	1
8	107	3	Dom Studencki Akademik	1
9	108	4	Dom Studencki Akademik	1
10	109	5	Dom Studencki Akademik	1
11	110	1	Dom Studencki Akademik	1
12	111	2	Dom Studencki Akademik	1
13	112	3	Dom Studencki Akademik	1
14	113	4	Dom Studencki Akademik	1
15	114	5	Dom Studencki Akademik	1
16	115	1	Dom Studencki Akademik	1
17	116	2	Dom Studencki Akademik	1
18	117	3	Dom Studencki Akademik	1
19	118	4	Dom Studencki Akademik	1
20	119	5	Dom Studencki Akademik	1
21	120	1	Dom Studencki Akademik	1

d) *doorkeepers* – informacje o pracownikach portierni Domów Studenckich

	First name	Last name	PESEL	Street	Flat number	ZIP	Country	Dormitory
1	Bruis	Ramsey	82120225556	Ap #436-408 Cursus. St.	NULL	33-878	Poland	Dom Studencki Bratniak-Musze...
2	Sebastien	Fibbens	78022219319	Ap #869-6594 Leo. Ave	NULL	85-768	Poland	Dom Studencki Bratniak-Musze...
3	Evita	Tyrer	77091186625	518-5195 Ante, Rd.	NULL	44-258	Poland	Dom Studencki Ustronie
4	Dniren	Lazell	72020743232	823-379 Non Rd.	NULL	58-763	Poland	Dom Studencki Ustronie
5	Gina	Noot	77020459358	2649 Nunc Rd.	NULL	55-363	Poland	Dom Studencki Mikrus
6	Tracy	Brockington	88010945559	630 Eros Road	NULL	44-007	Poland	Dom Studencki Mikrus
7	Melania	Gabites	75032949119	219-4009 Ornare. Rd.	NULL	86-694	Poland	Dom Studencki Akademik
8	Shaun	Stokey	88022865519	426-9013 Suspendisse Av.	NULL	71-957	Poland	Dom Studencki Akademik
9	Rochette	Desorts	79103126173	452 Morbi Ave	NULL	62-731	Poland	Dom Studencki nr 2
10	Konstantin	Emblin	82020975977	8905 Sed Rd.	NULL	91-656	Poland	Dom Studencki nr 2
11	Samuele	Salvador	73111944136	174-6738 Quisque Street	NULL	50-159	Poland	Dom Studencki nr 1
12	Cecil	Bleiman	83102648884	109-6586 Lectus. St.	NULL	81-489	Poland	Dom Studencki nr 1
13	Paulina	Greengrass	78121448612	Ap #929-3829 Odio St.	NULL	18-197	Poland	Dom Studenta nr 6
14	Bettine	Pyett	88011623542	Ap #222-7398 Quis Rd.	NULL	16-166	Poland	Dom Studenta nr 6
15	Gunilla	Petris	72031046779	844-9402 Tincidunt, Rd.	NULL	57-774	Poland	Dom Studenta nr 5
16	Leodora	Pontefract	72111489766	P.O. Box 583, 6257 Dolo...	NULL	54-635	Poland	Dom Studenta nr 5
17	Vaughan	Perrington	82071743648	Ap #648-293 Id Rd.	NULL	57-456	Poland	Sabinki

e) *administration_workers* – informacje o pracownikach administracji Domów Studenckich

id	first_name	last_name	email	phone_number	dormitory
1	Lark	Lorek	administracja@bratniak.com	112222333	Dom Studencki Bratniak-Musze...
2	Berna	Dillway	administracja@ustronie.com	112222333	Dom Studencki Ustronie
3	Natalina	Twinbrow	administracja@mikrus.com	112222333	Dom Studencki Mikrus
4	Jeanine	Corpe	administracja@akademik.com	112222333	Dom Studencki Akademik
5	Angele	Glanville	administracja@uw1.com	112222333	Dom Studencki nr 2
6	Maribeth	Spendley	administracja@uw2.com	112222333	Dom Studencki nr 1
7	Pepe	Slocum	administracja@uw3.com	112222333	Dom Studenta nr 6
8	Stern	Stobbart	administracja@uw4.com	112222333	Dom Studenta nr 5
9	Hallie	Cicchitello	administracja@sabinki.com	112222333	Sabinki
10	Samuel	Tappington	administracja@dendryt.com	112222333	Grosik
11	Alaster	Braksper	administracja@grosik.com	112222333	Dom studenta Dendryt
12	Drucie	Abramovici	administracja@krokus.com	112222333	Krokus Dom Studenta
13	Angel	Pencot	administracja@oaza.com	112222333	Oaza Dom Studenta

f) *issues* – informacje o zgłoszonych skargach w Domach Studenckich, wraz z danymi osoby zgłaszającej oraz studenta, na którego została złożona skarga

	date	content	student first name	student last name	room number	doorkeeper first name	doorkeeper last name
1	2020-10-19 18:00:00	Student narozrabiał	Rosette	Bentley	100	Bruis	Ramsey
2	2021-10-12 14:00:00	Student narozrabiał	Fabien	Pigden	100	Sebastien	Fibbens
3	2020-10-19 18:00:00	Student narozrabiał	Jarad	Baudet	102	Bruis	Ramsey
4	2021-10-12 14:00:00	Student narozrabiał	Stefano	Hofner	103	Sebastien	Fibbens
5	2020-10-19 18:00:00	Student narozrabiał	Rosette	Bentley	100	Bruis	Ramsey
6	2021-10-12 14:00:00	Student narozrabiał	Fabien	Pigden	100	Sebastien	Fibbens
7	2021-10-16 12:00:00	Student narozrabiał	Gusti	MacCaughan	104	Evita	Tyrer
8	2021-10-18 18:00:00	Student narozrabiał	Kaycee	Thyng	105	Dniren	Lazell
9	2021-10-16 12:00:00	Student narozrabiał	Sergeant	Handrek	106	Evita	Tyrer
10	2021-10-18 18:00:00	Student narozrabiał	Earvin	Alfwy	107	Dniren	Lazell
11	2021-10-16 12:00:00	Student narozrabiał	Gusti	MacCaughan	104	Evita	Tyrer
12	2021-10-18 18:00:00	Student narozrabiał	Kaycee	Thyng	105	Dniren	Lazell
13	2021-10-16 12:00:00	Student narozrabiał	Sergeant	Handrek	106	Evita	Tyrer
14	2021-10-18 18:00:00	Student narozrabiał	Earvin	Alfwy	107	Dniren	Lazell
15	2021-10-12 14:00:00	Student narozrabiał	Lauretta	Pipworth	108	Gina	Noot
16	2021-10-16 12:00:00	Student narozrabiał	Sheffield	Smewing	109	Tracy	Brockington
17	2021-10-12 14:00:00	Student narozrabiał	Austina	Millington	110	Gina	Noot
18	2021-10-16 12:00:00	Student narozrabiał	Brander	Hunnam	111	Tracy	Brockington

g) *announcements* – informacje o ogłoszeniach administracji Domów Studenckich wraz z danymi kontaktowymi do autora ogłoszenia

	date	content	first_name	last_name	email	phone_number	name
1	2021-10-19 18:00:00	Pierwsze ważne ogłoszenie dL...	Lark	Lorek	administracja@bratniak.com	112222333	Dom Studencki Bratniak-Musze...
2	2021-10-19 16:00:00	Pierwsze ważne ogłoszenie dL...	Berna	Dillway	administracja@ustronie.com	112222333	Dom Studencki Ustronie
3	2021-10-19 14:00:00	Pierwsze ważne ogłoszenie dL...	Natalina	Twinbrow	administracja@mikrus.com	112222333	Dom Studencki Mikrus
4	2021-10-19 12:00:00	Pierwsze ważne ogłoszenie dL...	Jeanine	Corpe	administracja@akademik.com	112222333	Dom Studencki Akademik
5	2021-10-19 18:00:00	Pierwsze ważne ogłoszenie dL...	Angele	Glanville	administracja@uw1.com	112222333	Dom Studencki nr 2
6	2021-10-19 16:00:00	Pierwsze ważne ogłoszenie dL...	Maribeth	Spendley	administracja@uw2.com	112222333	Dom Studencki nr 1
7	2021-10-19 14:00:00	Pierwsze ważne ogłoszenie dL...	Pepe	Slocum	administracja@uw3.com	112222333	Dom Studenta nr 6
8	2021-10-19 12:00:00	Pierwsze ważne ogłoszenie dL...	Stern	Stobbart	administracja@uw4.com	112222333	Dom Studenta nr 5
9	2021-10-19 18:00:00	Pierwsze ważne ogłoszenie dL...	Hallie	Cicchitello	administracja@sabinki.com	112222333	Sabinki
10	2021-10-19 16:00:00	Pierwsze ważne ogłoszenie dL...	Samuel	Tappington	administracja@dendryt.com	112222333	Grosik
11	2021-10-19 14:00:00	Pierwsze ważne ogłoszenie dL...	Alaster	Braksper	administracja@grosik.com	112222333	Dom studenta Dendryt
12	2021-10-19 12:00:00	Pierwsze ważne ogłoszenie dL...	Drucie	Abramovici	administracja@krokus.com	112222333	Krokus Dom Studenta
13	2021-10-19 18:00:00	Pierwsze ważne ogłoszenie dL...	Angel	Pencot	administracja@oaza.com	112222333	Oaza Dom Studenta
14	2020-05-19 13:00:00	Drugie ważne ogłoszenie dla ...	Lark	Lorek	administracja@bratniak.com	111222333	Dom Studencki Bratniak-Musze...
15	2020-05-19 13:00:00	Drugie ważne ogłoszenie dla ...	Berna	Dillway	administracja@ustronie.com	112222333	Dom Studencki Ustronie
16	2020-05-19 13:00:00	Drugie ważne ogłoszenie dla ...	Natalina	Twinbrow	administracja@mikrus.com	112222333	Dom Studencki Mikrus
17	2020-05-19 13:00:00	Drugie ważne ogłoszenie dla ...	Jeanine	Corpe	administracja@akademik.com	112222333	Dom Studencki Akademik
18	2020-05-19 13:00:00	Drugie ważne ogłoszenie dla ...	Angele	Glanville	administracja@uw1.com	112222333	Dom Studencki nr 2

2. Indeksy

Podczas zajęć wykładowych zostaliśmy zapoznani z indeksami klastrowymi oraz nieklastrowymi. Pierwsze z nich służą do logicznego i fizycznego uporządkowania danych w tabeli. Pojedyncza tabela może posiadać tylko jeden indeks klastrowy, który może zostać nałożony na jedną bądź wiele kolumn. Indeks nieklastrowy to kopia wybranych kolumn z tabeli, pozwalająca na szybkie wyszukiwanie danych z wybranych kolumn – zamiast

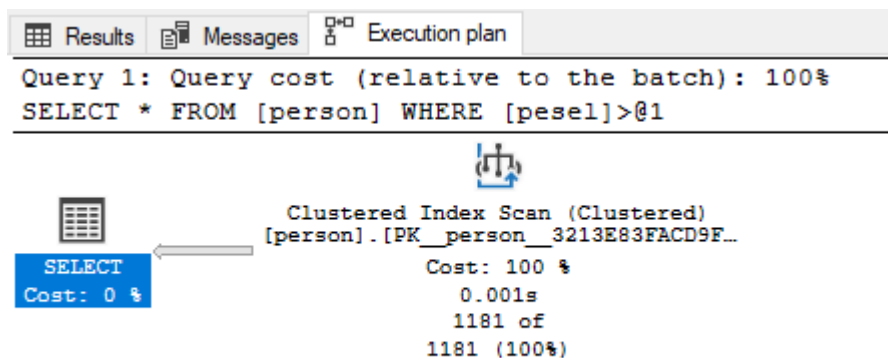
skanowania wykonywany jest *Seek*– system bazodanowy nie przemierza całej tabeli w poszukiwaniu danych.

2.1. Indeksy klastrowe

System bazodanowy Microsoft SQL Server Management Studio domyślnie nakłada klastrowe indeksy na klucze główne(PK) w tabelach. Niekoniecznie jest to działanie pożądane – klucz główny jest wartością unikatową w danej tabeli, natomiast indeksy klastrowe są skuteczne w przypadku danych często powtarzających się, często sortowanych, danych z pewnego pożądanego przedziału, mniejszych lub większych od określonej wartości.

Dobrym przykładem pozwalającym na wykorzystanie indeksu klastrowego w mojej bazie danych jest tabela *person*, przechowująca dane osobowe, m.in. numer PESEL. Uważam, że kolumna ta może być potencjalnie często wykorzystywana podczas zapytań. Poniżej przedstawiono rezultat wyszukiwania wszystkich osób o numerze PESEL większym niż 96, na tabeli z domyślnie założonym indeksem klastrowym na kluczu głównym:

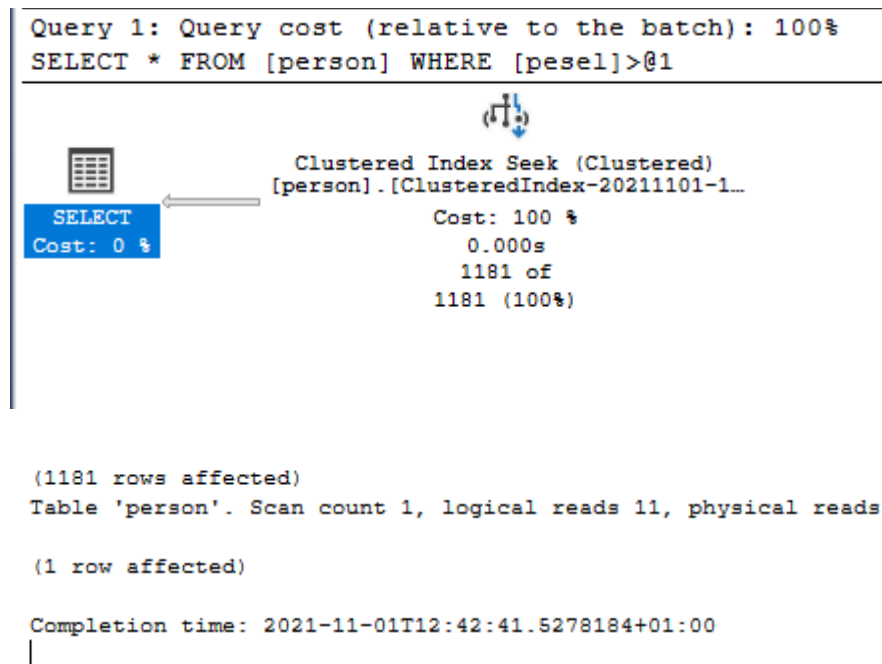
```
select * from person where pesel > '96'  
  
set statistics io on
```



(1181 rows affected)
Table 'person'. Scan count 1, logical reads 19, physical reads 0, page server reads 0,
(1 row affected)
Completion time: 2021-11-01T12:38:54.6557092+01:00
|

Jak przedstawiono powyżej, system bazodanowy przeskanował całą tabelę wykorzystując domyślny indeks klastrowy założony na kluczu głównym. Odczytano 19 stron bazodanowych.

Poniżej przedstawiono rezultat tego samego zapytania, po zdjęciu indeksu klastrowego z klucza głównego oraz nałożeniu go na kolumnę PESEL:

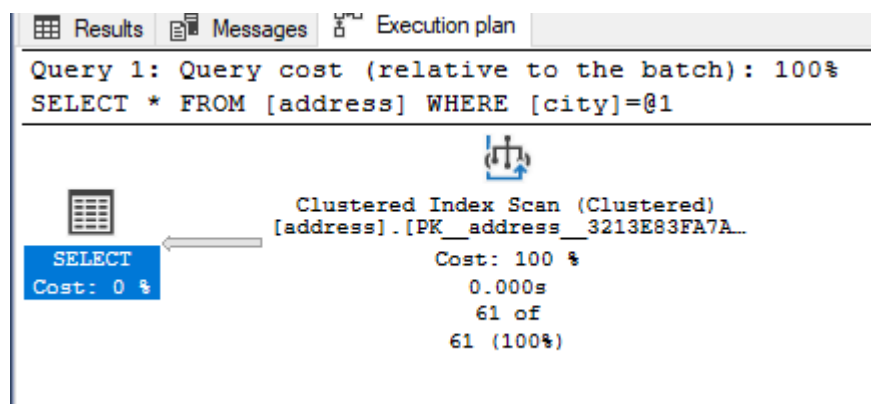


Jak można zauważyć, tym razem udało się uzyskać *Seek*, a ilość odczytanych stron wyniosła 11.

W przypadku tabeli *address* uznałem, że zapytania często odnosić mogą się do konkretnego miasta. Dodatkowo, jest to kolumna o danych powtarzających się, a zatem dobrym rozwiązaniem będzie założyć na nią indeks klastrowy. Tak jak w poprzednim przykładzie, poniżej przedstawiono rezultaty uzyskane w formie domyślnej:

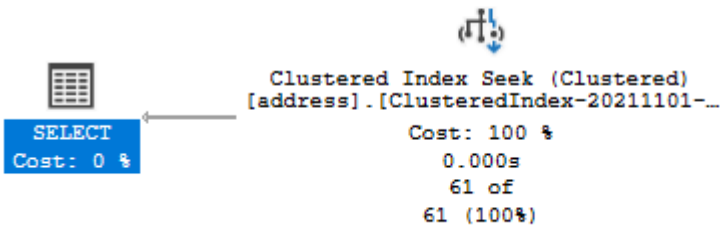
```
select * from address where city='Warszawa'
```

```
set statistics io on
```



Results	Messages	Execution plan
<pre> (61 rows affected) Table 'address'. Scan count 1, logical reads 25, physical reads 0, (1 row affected) Completion time: 2021-11-01T12:45:00.2262083+01:00 </pre>		

Oto wyniki po modyfikacji indeksu klastrowego:

Results	Messages	Execution plan
<pre> Query 1: Query cost (relative to the batch): 100% SELECT * FROM [address] WHERE [city]=@1 </pre>		
 <pre> (61 rows affected) Table 'address'. Scan count 1, logical reads 2, physical reads 0, (1 row affected) Completion time: 2021-11-01T12:50:28.3564163+01:00 </pre>		

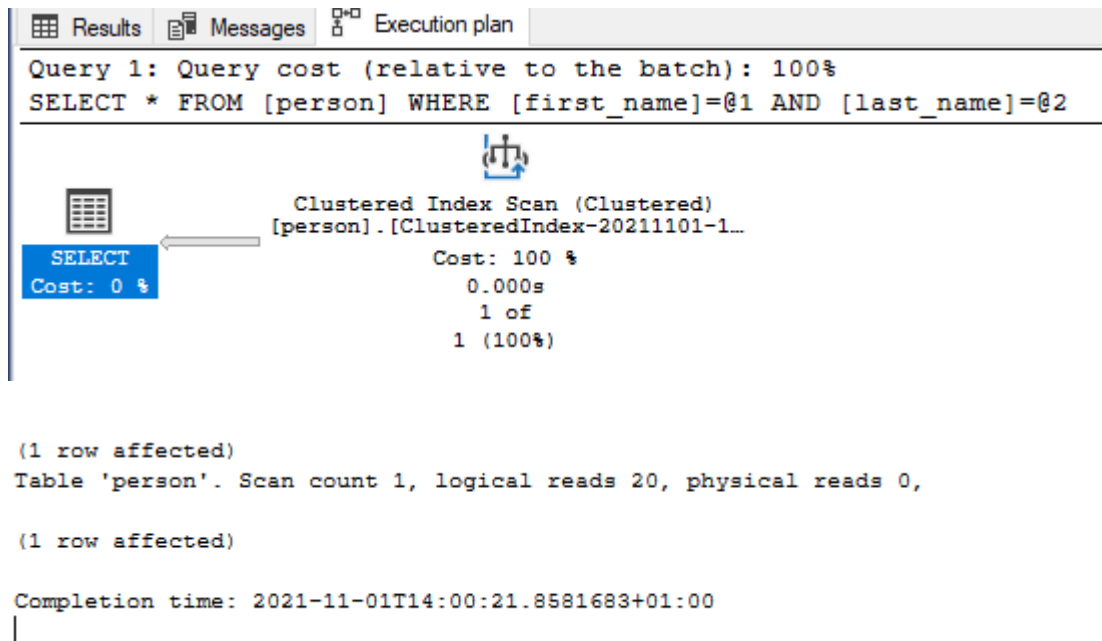
Jak można zauważyć, ilość odczytów została znacząco zmniejszona.

W przypadku tabeli *issue* oraz *announcement* indeksy klastrowe nałożyłem na kolumnę *date*, z porządkiem malejącym. W tabeli *room* ustawiłem indeks klastrowy na kolumny *capacity* oraz *dormitory_id*.

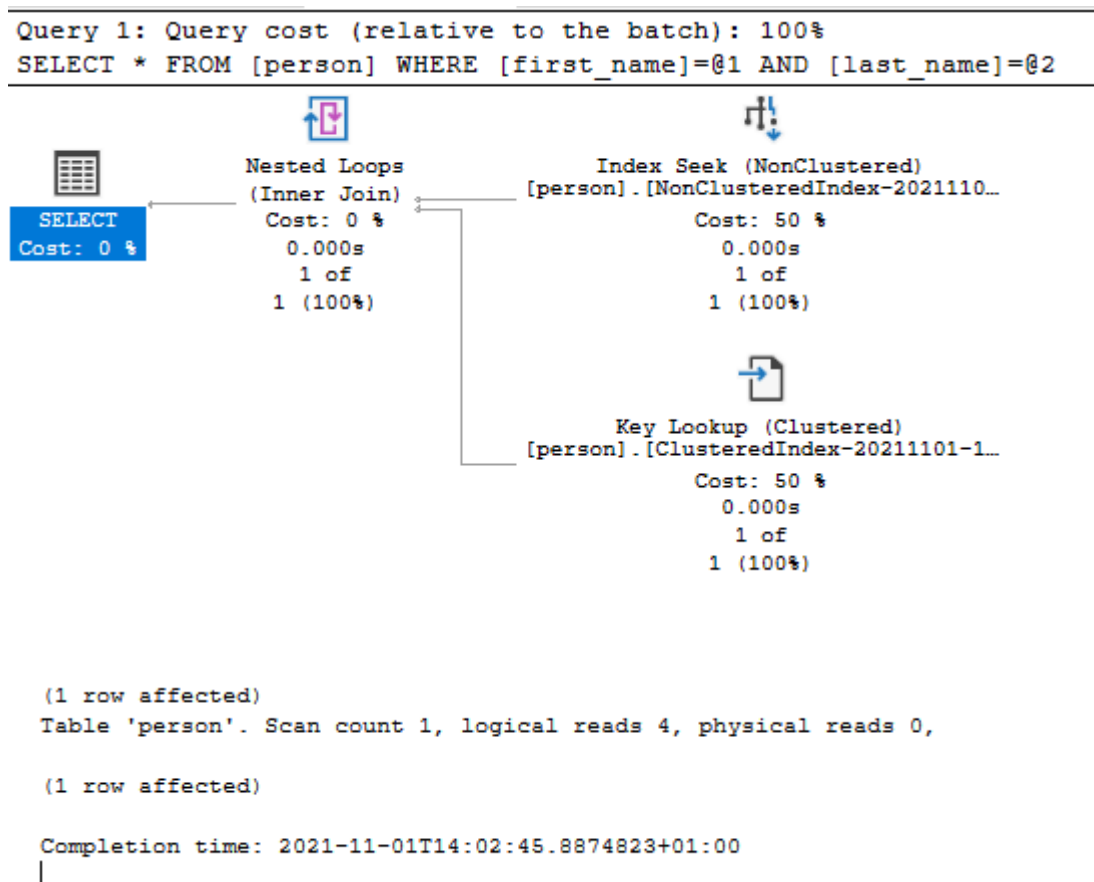
2.2. Indeksy nieklastrowe

W punkcie 2.1. dane w tabelach uporządkowane zostały fizycznie oraz logicznie. Aby dodatkowo przyspieszyć proces wyszukiwania rekordów po kolumnach, które mogą być często uwzględniane podczas wykonywania zapytań, można wykorzystać indeksy nieklastrowe. Zakładam, że w mojej bazie często następować będzie wyszukiwanie osób po imieniu oraz nazwisku. Poniżej przedstawiono rezultat przykładowego zapytania zawierającego podane kryteria:

```
select * from person where first_name='Arnold' and last_name='Eglise'
```



Jak można zauważyć, cała tabela została przeskanowana – odczytano 20 stron. Poniżej przedstawiono rezultat tego samego zapytania po nałożeniu indeksu nieklastrowego na kolumny *first_name* oraz *last_name*:



Tym razem przeszukany został indeks nieklastrowy, co przyniosło bardziej wydajny rezultat.

Wygenerowany indeks to indeks kompozytowy. Porównajmy jego działanie z sytuacją, gdy stworzone zostaną dwa indeksy proste dla kolumn *first_name* i *last_name*:

```
select * from person where first_name='Darya'
```

Query 1: Query cost (relative to the batch): 100%

```
SELECT * FROM [person] WHERE [first_name]=@1
```

Execution plan details:

- SELECT** (Cost: 0 %)
- Nested Loops (Inner Join)** (Cost: 0 %)
- Index Seek (NonClustered)** [person].[NonClusteredIndex-2021110...] (Cost: 46 %)
- Key Lookup (Clustered)** [person].[PK_person_3213E83F3982D...] (Cost: 54 %)

Results:

```
(1 row affected)
Table 'person'. Scan count 1, logical reads 4, physical reads 0,

(1 row affected)

Completion time: 2021-11-02T19:59:59.9559933+01:00
```

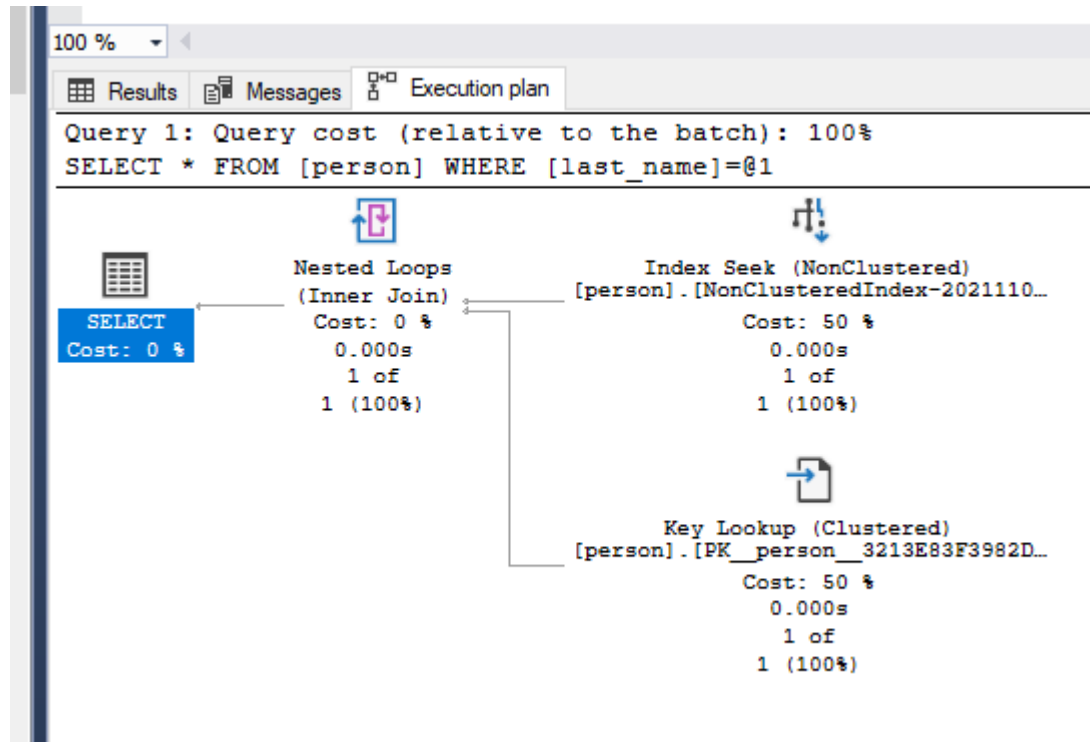
```
select * from person where last_name='Mathelin'
```

Results:

```
(1 row affected)
Table 'person'. Scan count 1, logical reads 4, physical reads 0,

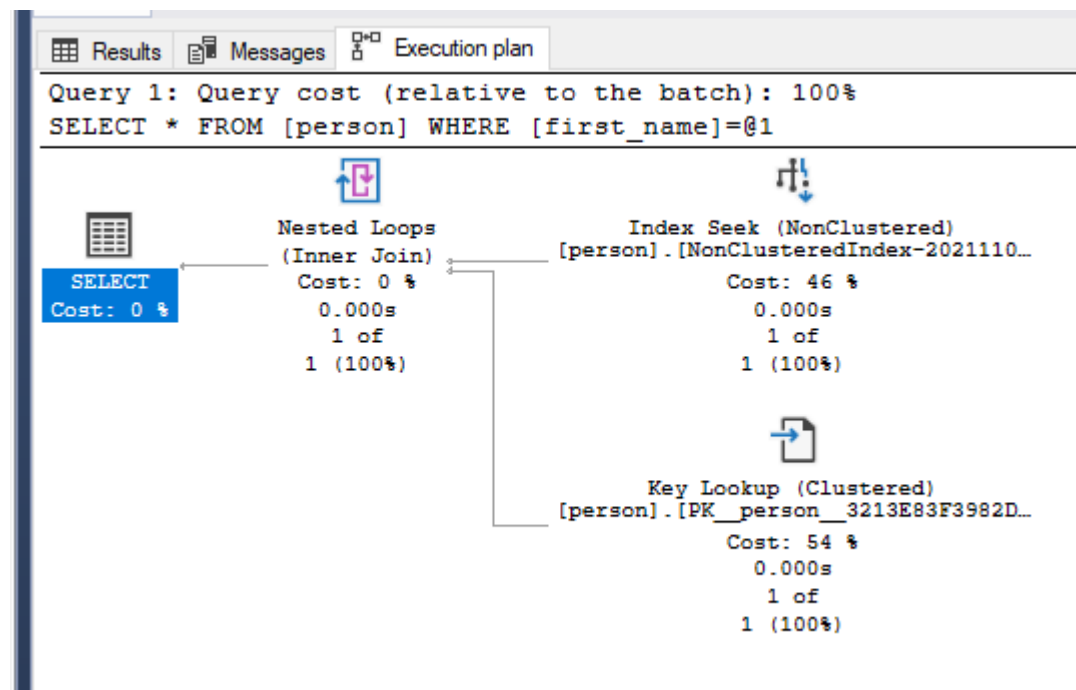
(1 row affected)

Completion time: 2021-11-02T20:01:02.7799211+01:00
```



Powtórzmy eksperyment przy użyciu indeksu kompozytowego:

```
select * from person where first_name='Darya'
```



Results Messages Execution plan

```

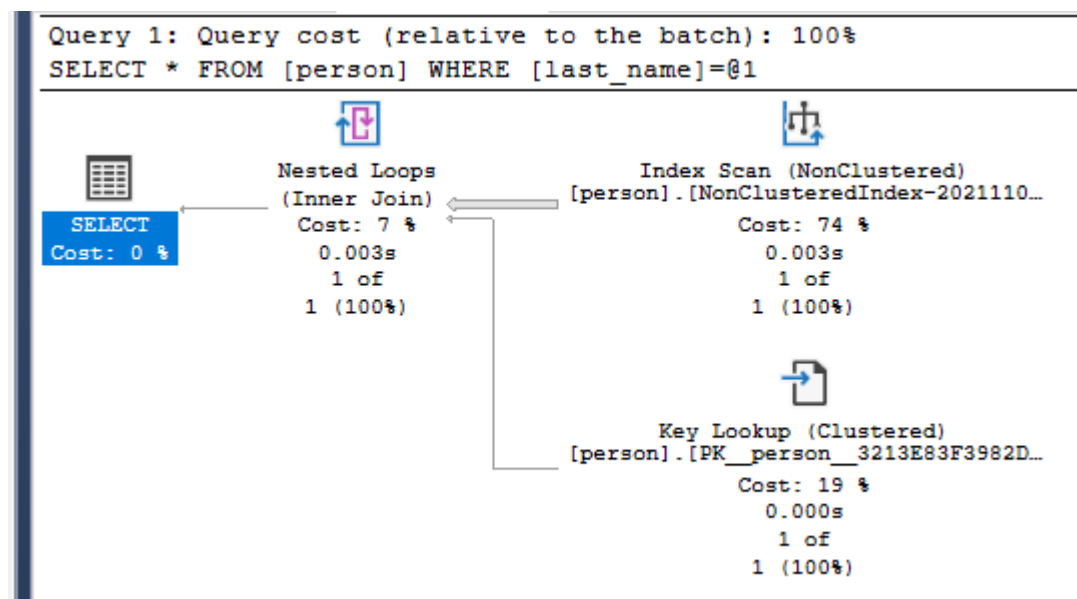
(1 row affected)
Table 'person'. Scan count 1, logical reads 4, physical reads 0,

(1 row affected)

Completion time: 2021-11-02T23:31:24.4890104+01:00
|

```

```
select * from person where last_name='Mathelin'
```



Results Messages Execution plan

```

(1 row affected)
Table 'person'. Scan count 1, logical reads 14, physical reads 0,

(1 row affected)

Completion time: 2021-11-02T23:32:49.4585769+01:00
|

```

Z powyższych zrzutów ekranu można wywnioskować, że w przypadku indeksu kompozytowego nastąpiło przeskanowanie całej struktury podczas filtrowania osoby po nazwisku. Wynika to z faktu, iż w indeksie kompozytowym istotna jest kolejność kolumn, które wchodzi w jego skład. Z tego powodu filtrowanie po imieniu przyniosło dobry rezultat – była to pierwsza kolumna w nim zadeklarowana.

3. Elementy programowalne

3.1. Funkcje

W celach zapoznawczych z elementami programowalnymi stworzone zostały funkcje: *StudentDataByPesel*, *NonFullRooms* oraz *StudentsSecurity*. Pierwsza z nich jako argument przyjmuje numer PESEL, dla którego zwraca dane osobowe studenta, pozyskane z utworzonej w punkcie 1. perspektywy.

```
CREATE FUNCTION StudentDataByPesel (@pesel CHAR(11))
RETURNS TABLE
AS
RETURN
(
SELECT * FROM students s WHERE s.PESEL=@pesel
)
GO
```

```
SELECT * FROM StudentDataByPesel('95050859964')
```

Results													Messages
	First name	Last name	PESEL	Street	Flat number	ZIP	Country	Term	Department	University	Room number	Dormitory	
1	Arlin	Ricciardiello	95050859964	791-4770 Ultrices. Rd.	NULL	72-111	Poland	3	Wydział Elektroniki i Technik Informacyjnych	Politechnika Warszawska	124	Dom Studencki Bratniak-Muszelka	

Funkcja *NonFullRooms* przyjmuje jako argument ciąg znaków odpowiadający nazwie Domu Studenckiego. Jako rezultat zwracane są wszystkie niezapełnione pokoje w wybranym akademiku, wraz z informacją o liczbie wolnych miejsc.

```
CREATE FUNCTION NonFullRooms(@dormitoryName VARCHAR(100))
RETURNS TABLE
AS
RETURN
(
SELECT *, (r.capacity - r.Locators) as 'Available spots'
FROM rooms r WHERE r.name LIKE '%' + @dormitoryName + '%' AND r.capacity > r.Locators
)
GO
```

```
SELECT * FROM NonFullRooms('Bratniak')
```

Results		Messages			
	number	capacity	name	Locators	Available spots
1	101	2	Dom Studencki Bratniak-Muszelka	1	1
2	102	3	Dom Studencki Bratniak-Muszelka	1	2
3	103	4	Dom Studencki Bratniak-Muszelka	1	3
4	104	5	Dom Studencki Bratniak-Muszelka	1	4
5	106	2	Dom Studencki Bratniak-Muszelka	1	1
6	107	3	Dom Studencki Bratniak-Muszelka	1	2
7	108	4	Dom Studencki Bratniak-Muszelka	1	3
8	109	5	Dom Studencki Bratniak-Muszelka	1	4
9	111	2	Dom Studencki Bratniak-Muszelka	1	1
10	112	3	Dom Studencki Bratniak-Muszelka	1	2
11	113	4	Dom Studencki Bratniak-Muszelka	1	3
12	114	5	Dom Studencki Bratniak-Muszelka	1	4
13	116	2	Dom Studencki Bratniak-Muszelka	1	1
14	117	3	Dom Studencki Bratniak-Muszelka	1	2
15	118	4	Dom Studencki Bratniak-Muszelka	1	3
16	119	5	Dom Studencki Bratniak-Muszelka	1	4

Zadaniem funkcji *StudentsSecurity* jest ograniczenie dostępu poszczególnych pracowników administracji Domów Studenckich do danych dotyczących placówki, w której pracują.

```

CREATE FUNCTION dbo.fn_StudentsSecurity(@user_id AS SMALLINT)
    RETURNS TABLE
    WITH SCHEMABINDING
    AS
    RETURN SELECT 1 AS fn_StudentsSecurity_Result
        WHERE @user_id=USER_ID() OR USER_ID()=1
GO

CREATE SECURITY POLICY StudentPrivacyPolicy
    ADD FILTER PREDICATE
        dbo.fn_StudentsSecurity(user_id) ON dbo.dormitory
    WITH (STATE = ON)

select * from dormitories

```

Poniżej przedstawiono rezultat pobrania studentów przez użytkownika będącego pracownikiem administracji Domu Studenckiego Bratniak-Muszelka:

	First name	Last name	PESEL	Street	Fiat number	ZIP	City	Country	Tem	Degree course	Department	Univer...	Room number	Dormitory
1	Darya	Mathelin	95081632112	Ap #647-23...	NULL	16-162	Kraków	Poland	5	Infomatyka	Wydział Elek...	Polite...	100	Dom Studencki Bratniak-Muszelka
2	Rosette	Bentley	95091961976	461-3731 L...	NULL	73-278	Częstochowa	Poland	2	Infomatyka	Wydział Elek...	Polite...	100	Dom Studencki Bratniak-Muszelka
3	Fabien	Pigden	95030429965	Ap #759-26...	NULL	15-370	Tomaszów Mazowiecki	Poland	1	Elektrotechnika	Wydział Elek...	Polite...	101	Dom Studencki Bratniak-Muszelka
4	Jarad	Baudet	95083016338	P.O. Box 63...	NULL	81-981	Gliwice	Poland	2	Infomatyka	Wydział Elek...	Polite...	102	Dom Studencki Bratniak-Muszelka
5	Stefano	Hofner	95060848756	245-4289 T...	NULL	65-837	Częstochowa	Poland	1	Infomatyka	Wydział Elek...	Polite...	103	Dom Studencki Bratniak-Muszelka
6	Gusti	MacCaughan	95032923214	Ap #293-37...	NULL	54-581	Piotrków Trybunalski	Poland	3	Infomatyka	Wydział Elek...	Polite...	104	Dom Studencki Bratniak-Muszelka
7	Kaycee	Thyng	95071668949	830-6215 S...	NULL	98-233	Starachowice	Poland	2	Elektronika	Wydział Elek...	Polite...	105	Dom Studencki Bratniak-Muszelka
8	Sergeant	Handrek	95033035246	927-1812 S...	NULL	52-088	Wrocław	Poland	1	Elektronika	Wydział Elek...	Polite...	106	Dom Studencki Bratniak-Muszelka
9	Earvin	Alfvy	95121461281	227-3037 E...	NULL	12-804	Chelm	Poland	3	Elektronika	Wydział Elek...	Polite...	107	Dom Studencki Bratniak-Muszelka
10	Lauretta	Pipworth	95110752525	Ap #537-54...	NULL	41-302	Jelenia Góra	Poland	1	Infomatyka	Wydział Elek...	Polite...	108	Dom Studencki Bratniak-Muszelka
11	Sheffield	Smewing	95112237664	910 Quis Rd.	NULL	19-516	Szczecin	Poland	2	Elektrotechnika	Wydział Elek...	Polite...	109	Dom Studencki Bratniak-Muszelka
12	Austina	Millington	95121758316	8336 Conse...	NULL	62-932	Olsztyn	Poland	3	Elektronika	Wydział Elek...	Polite...	110	Dom Studencki Bratniak-Muszelka
13	Brander	Hunnam	95072534122	6286 Natoq...	NULL	24-986	Toruń	Poland	3	Elektronika	Wydział Elek...	Polite...	111	Dom Studencki Bratniak-Muszelka
14	Rayshell	Roobottom	95121153777	P.O. Box 35...	NULL	83-366	Jelenia Góra	Poland	2	Infomatyka	Wydział Elek...	Polite...	112	Dom Studencki Bratniak-Muszelka
15	Yolanthe	Gannaway	95081743328	992-1726 D...	NULL	03-839	Chelm	Poland	2	Elektronika	Wydział Elek...	Polite...	113	Dom Studencki Bratniak-Muszelka
16	Rozelle	Render	95032132111	622-7062 Ti...	NULL	07-564	Gdańsk	Poland	1	Elektrotechnika	Wydział Elek...	Polite...	114	Dom Studencki Bratniak-Muszelka
17	Godfree	MacGonie	95121179724	374-5187 Ut...	NULL	43-115	Pabianice	Poland	2	Elektrotechnika	Wydział Elek...	Polite...	115	Dom Studencki Bratniak-Muszelka
18	Breena	Reavey	95070288955	Ap #418-97...	NULL	23-509	Legnica	Poland	3	Infomatyka	Wydział Elek...	Polite...	116	Dom Studencki Bratniak-Muszelka
19	Nomy	Dessaur	96022728389	Ap #253-42...	NULL	58-017	Kraków	Poland	3	Elektronika	Wydział Elek...	Polite...	117	Dom Studencki Bratniak-Muszelka
20	Iolanthe	Kauffman	95122923612	577-6365 Gr...	NULL	60-739	Kalisz	Poland	3	Elektronika	Wydział Elek...	Polite...	118	Dom Studencki Bratniak-Muszelka
21	Haily	Satterthwaite	95102672415	9207 Pede ...	NULL	40-761	Łódź	Poland	1	Elektrotechnika	Wydział Elek...	Polite...	119	Dom Studencki Bratniak-Muszelka
22	Eadie	Drissell	95072549656	5049 Augue...	NULL	28-884	Kraków	Poland	2	Elektrotechnika	Wydział Elek...	Polite...	120	Dom Studencki Bratniak-Muszelka
23	Ileane	Giacobilio	95082039796	385-1265 M...	NULL	52-021	Gorzów Wielkopolski	Poland	2	Elektrotechnika	Wydział Elek...	Polite...	121	Dom Studencki Bratniak-Muszelka

3.2. Procedury

Istotną różnicą między funkcją, a procedurą jest fakt, że procedura nie musi zwracać wartości. W związku z tym może zostać wykorzystana do wprowadzania danych. Poniżej przedstawiono przykładowe implementacje procedur:

- a) *addAnnouncement* - procedura przyjmująca jako argumenty id pracownika administracji oraz treść ogłoszenia. Po sprawdzeniu, czy identyfikator jest prawidłowy, umieszcza nowe ogłoszenie w bazie danych

```

CREATE PROCEDURE addAnnouncement(
    @content text = NULL,
    @id SMALLINT = NULL
)
AS
IF NOT EXISTS(SELECT aws.id FROM administration_workers aws WHERE aws.id=@id) THROW 51000, 'Please enter a valid id', 1;
INSERT INTO announcement(date, content, administration_worker_id) VALUES(
    GETDATE(), @content, @id
)
GO

EXEC addAnnouncement @content='N'Przypominamy o obowiązku noszenia maseczek na terenie
Domu Studenckiego', @id=1

```

30	2021-11-02 10:42:00	Przypominamy o obowiązku noszenia maseczek na terenie Domu Studenckiego	Lark	Lorek	administracja@bratniak.com	111222333	Dom Studencki Bratniak-Muszelka
----	---------------------	---	------	-------	----------------------------	-----------	---------------------------------

- b) *addIssue* – procedura przyjmująca jako argumenty treść skargi, id pracownika portierni oraz PESEL studenta, którego dotyczy skarga. Po sprawdzeniu, czy w bazie jest student o podanym numerze PESEL oraz pracownik portierni o podanym id, umieszcza nową skargę w bazie danych

```

CREATE PROCEDURE addIssue(
    @content text = NULL,
    @doorkeeperId SMALLINT = NULL,
    @pesel CHAR(11) = NULL
)
AS
IF NOT EXISTS(SELECT s.PESEL FROM students s WHERE s.PESEL=@pesel) THROW 51000, 'Invalid PESEL', 1;
IF NOT EXISTS(SELECT d.id FROM doorkeepers d WHERE d.id=@doorkeeperId) THROW 51000, 'Please enter a valid id', 1;
INSERT INTO issue(date, content, student_id, doorkeeper_id) VALUES (
    GETDATE(), @content, (SELECT s.id FROM students s WHERE s.pesel=@pesel), @doorkeeperId
)
GO

```

```

EXEC addIssue @content=N'Student zakłócał ciszę nocną', @doorkeeperId=1,
@pesel='95081632112'

```

47	2021-11-02 10:40:00	Student zakłócał ciszę nocną	Darya	Mathelin	100	Bruis	Ramsey	Dom Studencki Bratniak-Muszelka
----	---------------------	------------------------------	-------	----------	-----	-------	--------	---------------------------------

- c) *addNewStudent* – procedura dodająca nowego studenta do systemu. Jako argumenty przyjmuje dane osobowe i adresowe studenta oraz kierunek studiów, Dom Studencki, w którym student będzie mieszkał oraz numer pokoju

```

CREATE PROCEDURE addNewStudent
(
    @firstName varchar(25) = NULL,
    @secondName varchar(25) = NULL,
    @lastName varchar(50) = NULL,
    @pesel char(11) = NULL,
    @street varchar(70) = NULL,
    @flat tinyint = NULL,
    @zip varchar(10) = NULL,
    @city varchar(50) = NULL,
    @country varchar(50) = NULL,
    @term tinyint = NULL,
    @degreeCourse varchar(100) = NULL,
    @department varchar(100) = NULL,
    @university varchar(150) = NULL,
    @room varchar(10),
    @dormitory varchar(100) = NULL
)

```

Początkowym zadaniem procedury jest walidacja wprowadzonych pól. Następuje sprawdzenie, czy nazwy wydziału, kierunku studiów, uczelni oraz Domu Studenckiego są prawidłowe – czy występują w bazie danych.

```

BEGIN TRY
IF NOT EXISTS (SELECT id FROM university u WHERE u.name=@university) THROW 51000, 'Invalid university name.', 1;
IF NOT EXISTS (SELECT id FROM department d WHERE d.name=@department) THROW 51000, 'Invalid department name.', 1;
IF NOT EXISTS (SELECT id FROM degree_course dc WHERE dc.name=@degreeCourse) THROW 51000, 'Invalid degree course name.', 1;
IF NOT EXISTS (SELECT id FROM dormitory d WHERE d.name=@dormitory) THROW 51000, 'Invalid dormitory name.', 1;

```

Po wstępnej walidacji następuje rozpoczęcie transakcji. Na podstawie wprowadzonych danych adresowych procedura sprawdza, czy podany adres nie

istnieje już w bazie. Jeśli istnieje, zwraca jego identyfikator, w przeciwnym wypadku tworzy nowy adres.

```
DECLARE @duplicateAddressId AS INT
BEGIN TRAN
IF EXISTS (SELECT TOP 1 id FROM address ad WHERE ad.city=@city AND ad.country=@country AND ad.flat_number=@flat AND ad.postcode=@zip AND ad.street=@street)
SET @duplicateAddressId = (SELECT TOP 1 id FROM address ad WHERE ad.city=@city AND ad.country=@country AND ad.flat_number=@flat AND ad.postcode=@zip AND ad.street=@street)
ELSE
BEGIN
INSERT INTO address(street,postcode,city,country, flat_number) VALUES
(@street, @zip, @city, @country, @flat)
SET @duplicateAddressId = SCOPE_IDENTITY()
END
```

Kolejnym krokiem jest sprawdzenie, czy kierunek studiów należy do podanego wydziału oraz czy wydział należy do podanej uczelni.

```
DECLARE @degreeCourseId AS SMALLINT
SET @degreeCourseId = (SELECT dc.id FROM degree_course dc INNER JOIN department d ON d.id=dc.department_id WHERE d.name=@department AND dc.name=@degreeCourse)
IF @degreeCourseId = NULL THROW 51000, 'Degree course not found for provided department', 1;
DECLARE @departmentId AS SMALLINT
SET @departmentId = (SELECT d.id FROM department d INNER JOIN university u ON u.id=d.university_id WHERE u.name=@university AND d.name=@department)
IF @departmentId = NULL THROW 51000, 'Department not found for provided university', 1;
DECLARE @foundRoom AS TABLE(id INT, capacity TINYINT, locators INT)
```

Należy również upewnić się, że pokój, do którego rejestrowany jest student należy do podanego Domu Studenckiego oraz że nie jest on już zajęty.

```
DECLARE @foundRoom AS TABLE(id INT, capacity TINYINT, locators INT)
INSERT INTO @foundRoom select r.id, r.capacity, r.locators FROM rooms r WHERE r.number=@room AND r.dormitory=@dormitory
IF NOT EXISTS (SELECT id FROM @foundRoom) THROW 51000, 'No room with provided number found in chosen dormitory', 1;
IF EXISTS (SELECT * FROM @foundRoom WHERE capacity=locators) THROW 51000, 'This room is already full', 1;
```

Ostatnim krokiem jest zapisanie studenta do bazy danych oraz zakończenie transakcji.

```
INSERT INTO person(first_name, second_name, last_name, pesel, address_id) VALUES
(@firstName, @secondName, @lastName, @pesel, @duplicateAddressId)
SET @newPersonId = SCOPE_IDENTITY()
INSERT INTO student(term, degree_course_id, person_id, room_id) VALUES
(@term, @degreeCourseId, @newPersonId, (SELECT id FROM @foundRoom))
COMMIT
END TRY
BEGIN CATCH
THROW
ROLLBACK
END CATCH
```

W przypadku błędu podczas realizacji procedury, transakcja zostanie przerwana i zwrócony zostanie odpowiedni komunikat.

```
EXEC addNewStudent
@firstName='Adam',
@lastName='Adamiak',
@pesel='99034502199',
@street='Kwiatowa',
@flat=13,
@term=5,
@zip='02-031',
@city='Radom',
@country='Polska',
@university='Politechnika Warszawska',
@department=N'Wydział Elektryczny',
@degreeCourse='Informatyka',
@room=101,
@dormitory='Dom Studencki Bratniak-Muszelka'
```

(1 row affected)

Msg 51000, Level 16, State 1, Procedure addNewStudent, Line 47 [Batch Start Line 28]
This room is already full

Completion time: 2021-11-02T11:04:51.0429972+01:00

```
EXEC addNewStudent
@firstName='Adam',
@lastName='Adamiak',
@pesel='99034502199',
@street='Kwiatowa',
@flat=13,
@term=5,
@zip='02-031',
@city='Radom',
@country='Polska',
@university='Politechnika Warszawska',
@department=N'Wydział Elektryczny',
@degreeCourse='Informatyka',
@room=102,
@dormitory='Dom Studencki Jakikolwiek'
```

Msg 51000, Level 16, State 1, Procedure addNewStudent, Line 25 [Batch Start Line 28]
Invalid dormitory name.

Completion time: 2021-11-02T11:06:07.6286581+01:00

```
EXEC addNewStudent
@firstName='Adam',
@lastName='Adamiak',
@pesel='99034502199',
@street='Kwiatowa',
@flat=13,
@term=5,
@zip='02-031',
@city='Radom',
@country='Polska',
@university='Politechnika Warszawska',
@department=N'Wydział Elektryczny',
@degreeCourse='Informatyka',
@room=102,
@dormitory='Dom Studencki Bratniak-Muszelka'
```

100 %

Messages

(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)

Completion time: 2021-11-02T11:07:29.6734945+01:00

```
select * from students where pesel='99034502199'
```

0 %

Results Messages

	id	First name	Last name	PESEL	Street	Flat number	ZIP	City	Country	Term	Degree course	Department	University	Room number	Dormitory	
I	2605	Adam	Adamiak	99034502199	Kwiatowa	13	02-031	Radom	Polska	5	Informatyka	Wydział Elektryczny	Politechnika Warszawska	102	Dom Studencki Bratniak-Muszelka	

3.3. Wyzwalacze

Generalizacja pozwala na uniknięcie duplikacji kolumn w różnych tabelach poprzez umieszczenie wspólnych danych w jednej tabeli. Rozwiązanie to ma pewien mankament logiczny – w opracowywanej przeze mnie bazie danych może nastąpić sytuacja, w której zarówno student, jak i pracownik administracji będzie tą samą osobą. Aby rozwiązać ten problem można utworzyć trigger na wszystkich tabelach powiązanych z tabelą *person*. Każdy insert do tabel *student*, *administration_worker*, *doorkeeper* zostanie zwalidowany – jeśli dana osoba będzie już miała przypisaną rolę, system nie wykona zapytania oraz zwróci błąd. Poniżej przedstawiono implementację wyzwalacza w przypadku tabeli *student*:

```

CREATE TRIGGER trg_student
ON student
INSTEAD OF INSERT
AS
BEGIN TRY

IF EXISTS (SELECT aw.person_id FROM administration_worker aw WHERE aw.person_id = (SELECT person_id FROM inserted))
    THROW 51000, 'This person is already an administration worker', 1;
IF EXISTS (SELECT d.person_id FROM doorkeeper d WHERE d.person_id = (SELECT person_id FROM inserted))
    THROW 51000, 'This person is already a doorkeeper', 1;

INSERT INTO student(term, degree_course_id, person_id, room_id) SELECT term, degree_course_id, person_id, room_id FROM inserted
END TRY
BEGIN CATCH
    THROW;
END CATCH

```

```
INSERT INTO student VALUES (7, 1, 2688, 1)
```

```
Msg 51000, Level 16, State 1, Procedure trg_student, Line 10 [Batch Start Line 19]
This person is already an administration worker
```

```
Completion time: 2021-11-02T10:50:42.8846763+01:00
```

Analogicznie, próba dodania pracownika portierni lub administracji w niewłaściwy sposób spowoduje wywołanie błędu:

```
INSERT INTO administration_worker VALUES('test@mail', '123456788', 1, 300)
```

```
Msg 51000, Level 16, State 1, Procedure trg_administration_worker, Line 12 [Batch Start Line 22]
This person is already a student
```

```
Completion time: 2021-11-02T10:54:31.4818270+01:00
```

```
INSERT INTO doorkeeper VALUES(1, 2688)
```

```
Msg 51000, Level 16, State 1, Procedure trg_doorkeeper, Line 12 [Batch Start Line 26]
This person is already an administration worker
```

```
Completion time: 2021-11-02T10:55:01.2321267+01:00
```

Podczas zgłaszania skargi bądź rejestracji noclegu należy podać identyfikatory studenta oraz pracownika portierni. Aby upewnić się, że dany student i pracownik portierni należą do tego samego Domu Studenckiego można wykorzystać wyzwalacze:

```
CREATE TRIGGER trg_issue
ON issue
INSTEAD OF INSERT
AS
BEGIN TRY

DECLARE @dorm1 AS VARCHAR(100)
DECLARE @dorm2 AS VARCHAR(100)

SET @dorm1 = (SELECT s.[Dormitory] FROM students s WHERE s.id= (SELECT student_id FROM INSERTED))
SET @dorm2 = (SELECT d.[Dormitory] FROM doorkeepers d WHERE d.id=(SELECT doorkeeper_id FROM INSERTED))
IF @dorm1 != @dorm2 THROW 51000, 'There is an identity problem. Student and doorkeeper are from different dormitories', 1;

INSERT INTO issue(date, content, student_id, doorkeeper_id) SELECT date, content, student_id, doorkeeper_id FROM INSERTED

END TRY
BEGIN CATCH
THROW;
END CATCH
GO

CREATE TRIGGER trg_accomodation
ON accomodation
INSTEAD OF INSERT
AS
BEGIN TRY

DECLARE @dorm1 AS VARCHAR(100)
DECLARE @dorm2 AS VARCHAR(100)

SET @dorm1 = (SELECT s.[Dormitory] FROM students s WHERE s.id= (SELECT student_id FROM INSERTED))
SET @dorm2 = (SELECT d.[Dormitory] FROM doorkeepers d WHERE d.id=(SELECT doorkeeper_id FROM INSERTED))
IF @dorm1 != @dorm2 THROW 51000, 'There is an identity problem. Student and doorkeeper are from different dormitories', 1;

INSERT INTO accomodation(start_date, end_date, student_id, person_id, doorkeeper_id) SELECT start_date, end_date, student_id, person_id, doorkeeper_id FROM INSERTED

END TRY
BEGIN CATCH
THROW;
END CATCH
```

```
-- doorkeeper - Dom Studencki Bratniak-Muszelka
-- student - Dom Studencki nr 6

EXEC addIssue @content=N'Student zakłócał ciszę nocną', @doorkeeperId=1, @pesel='99053078685'
```

Msg 51000, Level 16, State 1, Procedure trg_issue, Line 13 [Batch Start Line 2]
There is an identity problem. Student and doorkeeper are from different dormitories

Completion time: 2021-11-02T10:44:03.7163844+01:00

4. Automatyzacja zadań

Za pomocą narzędzia Microsoft SQL Server Management Studio utworzony został Maintenance Plan, którego zadaniem jest generowanie backupu bazy danych codziennie o godzinie 21:00.

Subplan	Description	Schedule	Run as
Subplan_1		Occurs every day at 21:00:00. Sc...	SQL Server Agent service account

Back Up Database (Full)

Backup Database on Local server connection

Databases: dormitory

Type: Full

Append existing

Destination: Disk

Backup Compression (Default)

W celu przetestowania czy zadanie wykonuje się poprawnie, wywołano metodę *Execute*. W rezultacie otrzymano plik .bak w folderze, w którym powinien znaleźć się backup bazy danych.

The top screenshot shows the SQL Server Enterprise Manager interface. A right-click context menu is open over the 'BackupPl' job under 'Maintenance Plans'. The 'Execute' option is highlighted in yellow. Other menu items include 'New Maintenance Plan...', 'Maintenance Plan Wizard', 'View History', 'Modify', 'Reports', 'Rename', 'Delete', and 'Refresh'.

The bottom screenshot shows a Windows File Explorer window with the address bar set to 'Microsoft SQL Server > MSSQL15.MSSQLSERVER > MSSQL > Backup > dormitory'. The file list contains one item:

Nazwa	Data modyfikacji	Typ	Rozmiar
dormitory_backup_2021_11_02_142827_9...	02.11.2021 14:28	Plik BAK	9 313 KB

W celu zapoznania z możliwościami narzędzia SQL Server Agent dodano do bazy danych nową tabelę – *payment* oraz odpowiadający jej widok - *payments*. W tabeli przechowywane będą płatności studentów za pobyt w Domach Studenckich. Poniżej przedstawiono przykładowe dane odczytane z widoku *payments*:

	date	amount	id	First name	Last name	PESEL	Street	Fiat number	ZIP	City	Country	Term	Degree course	Department
1	2021-10-09 00:00:00	380.00	1	Rosette	Bentley	95091961976	461-3731 L...	NULL	73-278	Częstochowa	Poland	2	Informatyka	Wydział Elektryczny
2	2021-10-10 00:00:00	380.00	2	Fabien	Pigden	95030429965	Ap #759-2...	NULL	15-370	Tomaszów Mazowiecki	Poland	1	Elektrotechnika	Wydział Elektryczny
3	2021-10-01 00:00:00	380.00	3	Jarad	Baudet	95083016338	P.O. Box 6...	NULL	81-981	Gliwice	Poland	2	Informatyka	Wydział Elektryczny
4	2021-10-02 00:00:00	380.00	4	Stefano	Hofner	95060848756	245-4289 ...	NULL	65-837	Częstochowa	Poland	1	Informatyka	Wydział Elektryczny
5	2021-10-04 00:00:00	380.00	5	Gusti	MacCaughan	95032923214	Ap #293-3...	NULL	54-581	Piotrków Trybunalski	Poland	3	Informatyka	Wydział Elektryczny
6	2021-10-14 00:00:00	380.00	6	Kaycee	Thyng	95071668949	830-6215 ...	NULL	98-233	Starachowice	Poland	2	Elektronika	Wydział Elektroniki i Techn
7	2021-10-07 00:00:00	380.00	7	Sergeant	Handrek	95033035246	927-1812 ...	NULL	52-088	Wrocław	Poland	1	Elektronika	Wydział Elektroniki i Techn
8	2021-10-07 00:00:00	380.00	8	Earvin	Alfvy	95121461281	227-3037 ...	NULL	12-804	Chelm	Poland	3	Elektronika	Wydział Elektroniki i Techn
9	2021-10-13 00:00:00	380.00	9	Lauretta	Pipworth	95110752525	Ap #537-5...	NULL	41-302	Jelenia Góra	Poland	1	Informatyka	Wydział Elektryczny
10	2021-10-01 00:00:00	380.00	10	Sheffield	Smewing	95112237664	910 Quis Rd.	NULL	19-516	Szczecin	Poland	2	Elektrotechnika	Wydział Elektryczny
11	2021-10-10 00:00:00	380.00	11	Austina	Millington	95121758316	8336 Cons...	NULL	62-932	Olsztyn	Poland	3	Elektronika	Wydział Elektroniki i Techn
12	2021-10-03 00:00:00	380.00	12	Brander	Hunnam	95072534122	6286 Nato...	NULL	24-986	Toruń	Poland	3	Elektronika	Wydział Elektroniki i Techn
13	2021-10-09 00:00:00	380.00	13	Rayshell	Roobottom	95121153777	P.O. Box 3...	NULL	83-366	Jelenia Góra	Poland	2	Informatyka	Wydział Elektryczny
14	2021-10-07 00:00:00	380.00	14	Yolanthe	Gannaway	95081743328	992-1726 ...	NULL	03-839	Chelm	Poland	2	Elektronika	Wydział Elektroniki i Techn
15	2021-10-12 00:00:00	380.00	15	Rozelle	Render	95032132111	622-7062 ...	NULL	07-564	Gdańsk	Poland	1	Elektrotechnika	Wydział Elektryczny
16	2021-10-13 00:00:00	380.00	16	Godfree	MacGormie	95121179724	374-5187 ...	NULL	43-115	Pabianice	Poland	2	Elektrotechnika	Wydział Elektroniki i Techn

Biznesowym założeniem jest obowiązek uiszczenia opłaty przez studenta do 15. dnia każdego miesiąca. W celu skontrolowania terminów opłat w systemie utworzony został Job o nazwie *FindUnpaidBills*.

Name:

FindUnpaidBills

Owner:

DESKTOP-88S1D2V\optiplex

...

Category:

Full-Text

...

Description:

This job finds students who haven't paid for their dormitory each 13th day of the month

☒ Enabled

Każdego 13. dnia miesiąca o godzinie 00:00 do bazy danych automatycznie kierowane jest następujące zapytanie:

```
select * from students where id not in (select id from payments p where
month(p.date)=month(getdate()) and year(getdate())=year(p.date))
```

Rezultat zapytania zapisywany jest do pliku tekstowego. Zawarte są w nim dane wszystkich studentów, którzy nie umieścili jeszcze opłaty za pobyt w Domu Studenckim w danym miesiącu.

Schedule list:			
ID	Name	Enabled	Description
10	PaymentsCheckSchedule	Yes	Occurs every month on day 13 at 00:00:00. Schedule will be used starting on 02.11.2021.

Aby przetestować działanie zadania, chwilowo zmieniono datę wykonania zapytania na 02.11.2021 23:52.

Description:

Occurs every month on day 2 at 23:52:00. Schedule will be used starting on 02.11.2021.

Log file summary: No filter applied						
Date	Step ID	Server	Job Name	Step Name	Notifications	Message
02.11.2021 23:52:00		DESKTOP-88S1D2V	FindUnpaidBills			The job succeeded. The Job was invoked by Schedule 10 (PaymentsCheckSchedule).

W rezultacie otrzymano dane studentów, którzy nie umieścili opłaty, w pliku tekstowym.

Job 'FindUnpaidBills' : Step 1, 'FindUnpaidBills' : Began Executing 2021-11-02 23:52:00

id	First name	Last name	PESEL	Street	Flat number	ZIP	City
1	Rosette	Bentley	95091961976	461-3731 Leo, Road	(null)	73-278	Częstochowa
2	Fabien	Pigden	95030429965	Ap #759-2681 Sapien, Road	(null)	15-370	Tomaszów Mazowiecki
3	Jarad	Baudet	95083016338	P.O. Box 635, 4311 Mauris, Avenue	(null)	81-981	Gliwice
4	Stefano	Hofner	95060848756	245-4289 Tellus St.	(null)	65-837	Częstochowa
5	Gusti	MacCaughan	95032923214	Ap #293-3765 Est, St.	(null)	54-581	Piotrków Trybunalski
6	Kaycee	Thyng	95071668949	830-6215 Sem Road	(null)	98-233	Starachowice
7	Sergeant	Handrek	95033035246	927-1812 Sapien. Av.	(null)	52-088	Wrocław
8	Earvin	Alfvy	95121461281	227-3037 Enim Av.	(null)	12-804	Chełm
9	Lauretta	Pipworth	95110752525	Ap #537-5491 Purus. Avenue	(null)	41-302	Jelenia Góra
10	Sheffield	Smewing	95112237664	910 Quis Rd.	(null)	19-516	Szczecin
11	Austina	Millington	95121758316	8336 Consectetuer St.	(null)	62-932	Olsztyn
12	Brander	Hunnam	95072534122	6286 Natoque Street	(null)	24-986	Toruń
13	Rayshell	Roobottom	95121153777	P.O. Box 355, 720 Turpis St.	(null)	83-366	Jelenia Góra
14	Yolanthe	Gannaway	95081743328	992-1726 Donec Rd.	(null)	03-839	Chełm
15	Rozelle	Render	95032132111	622-7062 Tincidunt, Rd.	(null)	07-564	Gdańsk
16	Godfree	MacGorrie	95121179724	374-5187 Ut Av.	(null)	43-115	Pabianice

5. Podsumowanie

- 5.1. Za pomocą widoków można udostępniać użytkownikom bazodanowym informacje z tabel powiązanych relacjami w postaci czytelnych i łatwo dostępnych zbiorów danych.
- 5.2. Indeksy to struktury danych pozwalające na poprawę wydajności pozyskiwania danych z bazy. Należy jednak mieć na uwadze, że ich nieumiejętne wykorzystanie może spowodować więcej szkód niż dostarczyć korzyści.
- 5.3. Za pomocą funkcji, procedur oraz wyzwalaczy można wdrożyć logikę biznesową do bazy danych oraz uprościć proces wyszukiwania i wprowadzania informacji do tabel.
- 5.4. Microsoft SQL Server Management Studio umożliwia wdrożenie automatyzacji w bazie danych w prosty sposób, z wykorzystaniem narzędzia SQL Server Agent.