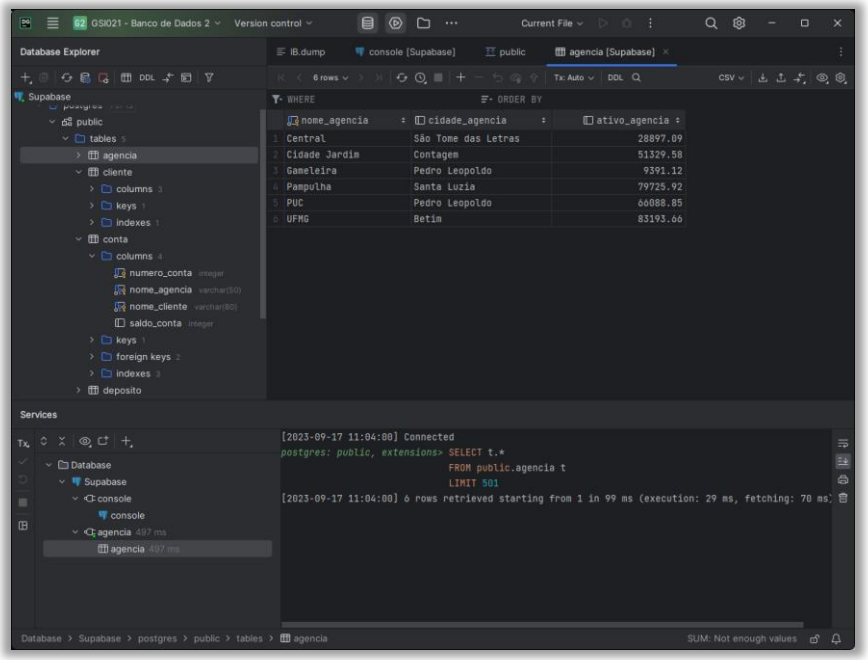
T02 – Instalação Postgres, to Join or not to Join e funções

Igor Augusto Reis Gomes – 12011BSI290 – igor.augusto@ufu.br

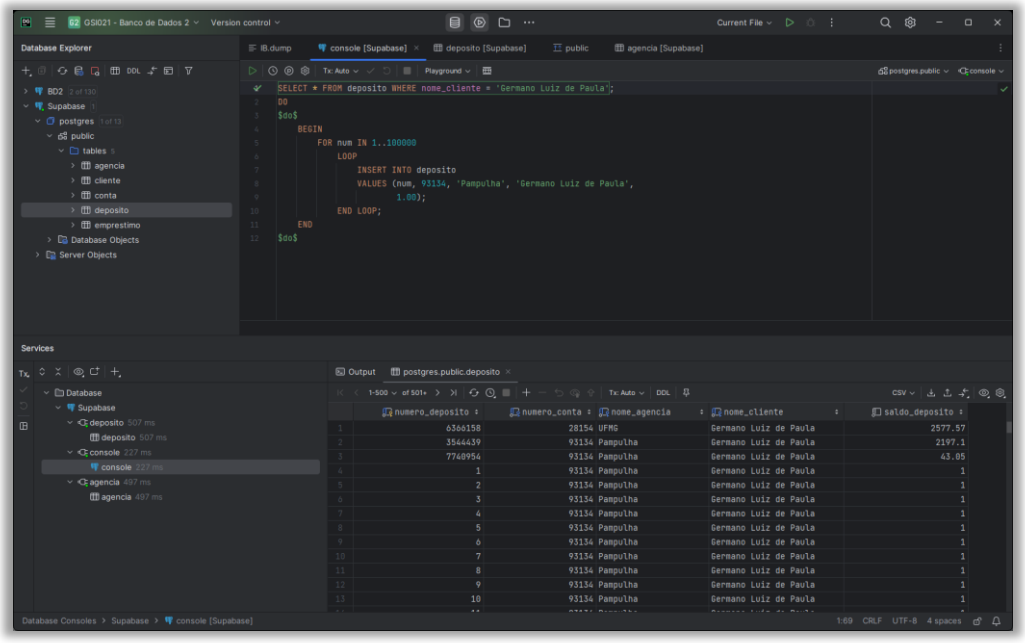
Heitor Guimarães Da Fonseca Filho – 12011BSI203 – heitor.filho@ufu.br

*Este relatório contêm prints dos resultados das operações realizadas, o código em está localizado em um arquivo separado.

A seguir, um print do banco de dados conectado e populado no DataGrip (uma alternativa mais moderna ao pgadmin), sendo da mesma empresa responsável pela conhecida IDE de Java, o IntelliJ:

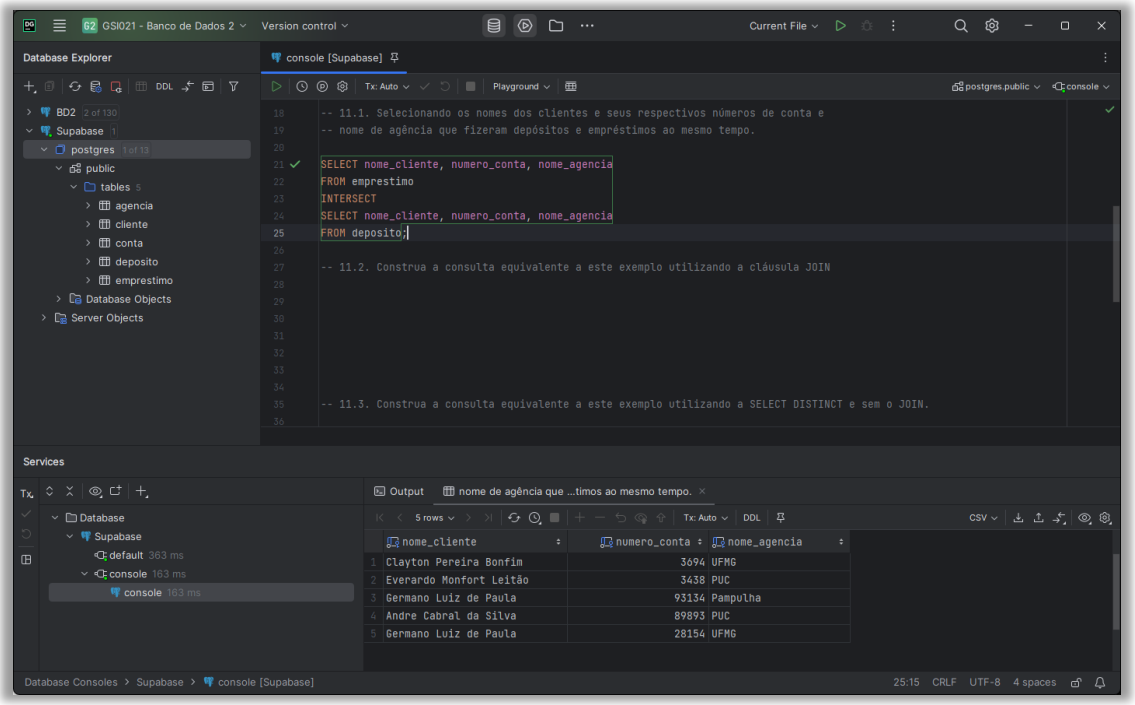


9. Acrescente, por exemplo, 100.000 depósitos de R$ 1,00 (Um Real) na conta do cliente 'Germano Luiz de Paula', na agência 'Pampulha', na conta 93134. Você deve executar o código abaixo em uma janela de comandos do PostgreSQL, quando o banco de dados selecionado para consultas for o nosso banco IB.



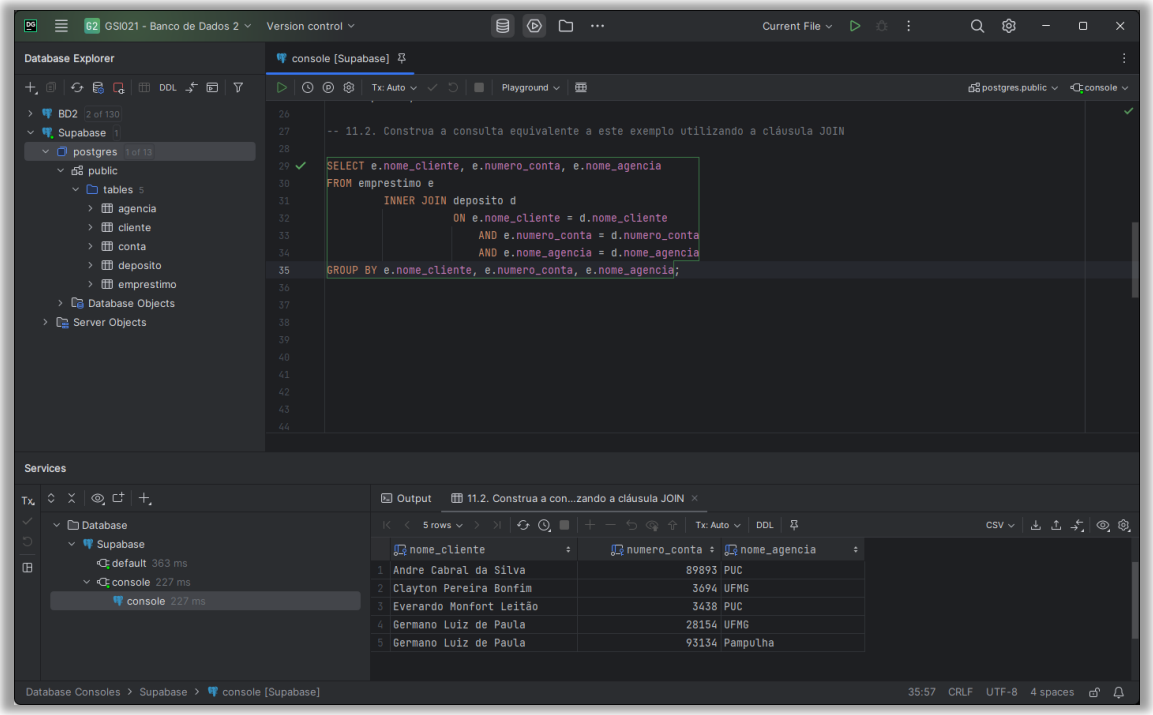
11. Agora que temos o banco IB com uma tabela de tamanho significativamente maior, podemos testar a hipótese levantada no texto introdutório: o uso da cláusula JOIN é cômodo, mas custoso.

11.1. Selecione os nomes dos clientes e seus respectivos números de conta e nome de agência que fizeram depósitos e empréstimos ao mesmo tempo.

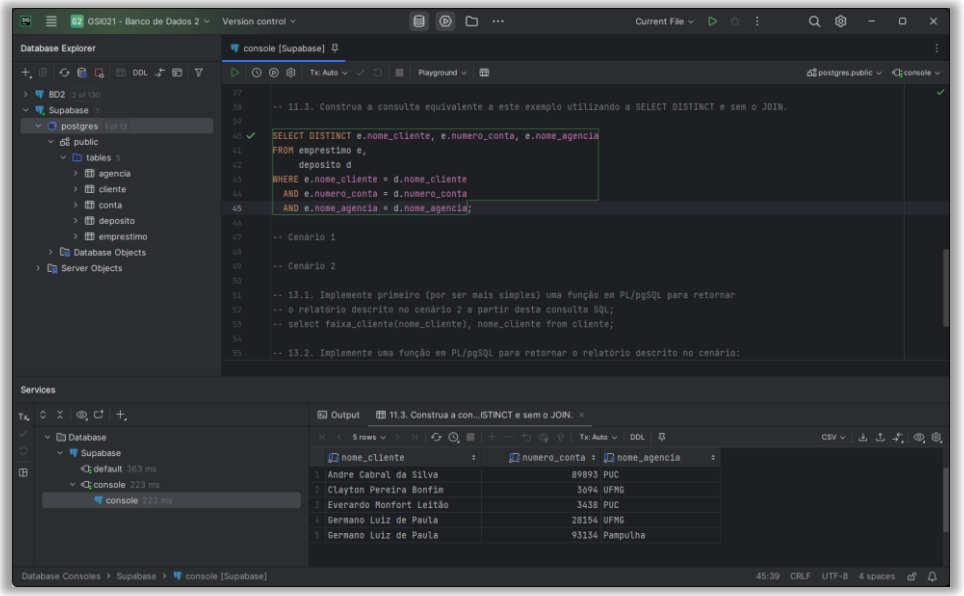


Agora vem sua tarefa:

11.2. Construa a consulta equivalente a este exemplo utilizando a cláusula JOIN



11.3. Construa a consulta equivalente a este exemplo utilizando a SELECT DISTINCT e sem o JOIN.



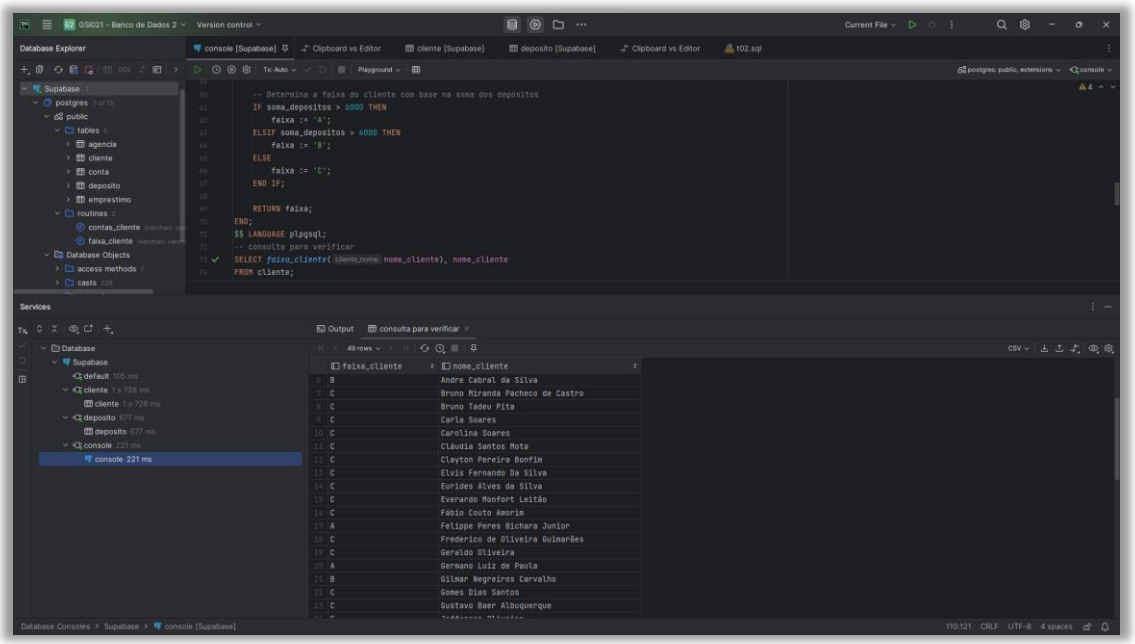
12. Construa uma tabela em uma planilha (Programa Calc do Libre Office) com três colunas: intersect, distinct e join. Execute as três consultas pelo menos 30 vezes e registre na planilha o tempo de execução em milissegundos para a conclusão de cada uma das três versões da consulta. O tempo de execução de cada consulta é exibido no canto inferior direito da tela que executou uma consulta. Ao final, tire a média de cada coluna e conclua qual versão da consulta foi mais rápida.

	INTERSECT	JOIN	DISTINCT
	101	130	135
	91	133	132
	78	133	132
	99	131	129
	106	132	128
	87	126	137
	89	141	130
	96	125	144
	91	137	155
	106	138	155
	97	133	172
	102	128	161
	95	131	148
	107	148	147
	99	147	139
	98	133	125
	102	132	131
	100	136	142
	86	135	136
	91	142	146
	90	146	132
	94	130	129
	99	132	128
	101	133	132
	103	141	136
	100	133	128
	99	134	132
	113	136	123
	89	133	132
	96	130	126
MÉDIA	96,83	134,6	137,4

13. Uma das principais vantagens obtidas pela utilização da SQL está no uso de uma linguagem não procedural, ou seja, é necessário dizer apenas “o que” é desejado que seja retornado em uma consulta sem se preocupar com os detalhes de “como” a consulta será executada. Entretanto, em determinadas situações nos encontramos enfrentando problemas nos quais uma simples consulta feita na SQL não nos atende. Vejamos alguns exemplos:

13.1. Implemente primeiro (por ser mais simples) uma função em PL/pgSQL para retornar o relatório descrito no cenário 2 a partir desta consulta SQL;

```
select faixa_cliente(nome_cliente), nome_cliente from cliente;
```



13.2. Implemente uma função em PL/pgSQL para retornar o relatório descrito no cenário 1

```
select nome_cliente, contas_cliente(nome_cliente), cidade_cliente from cliente;
```