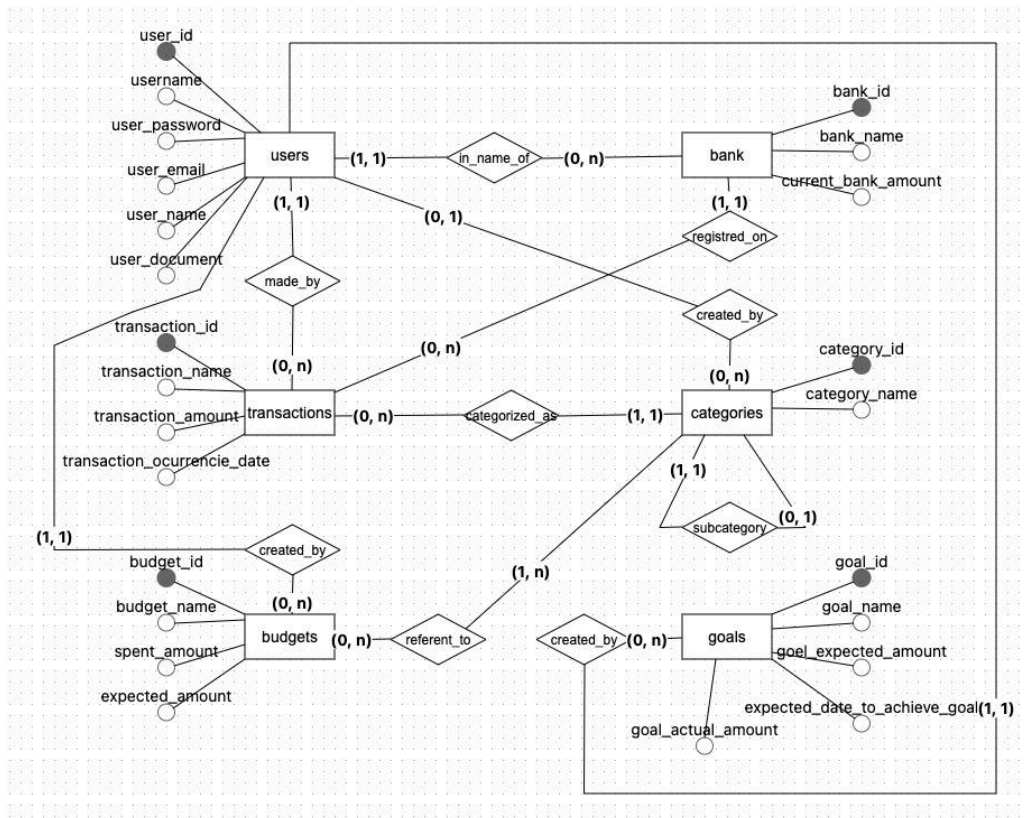
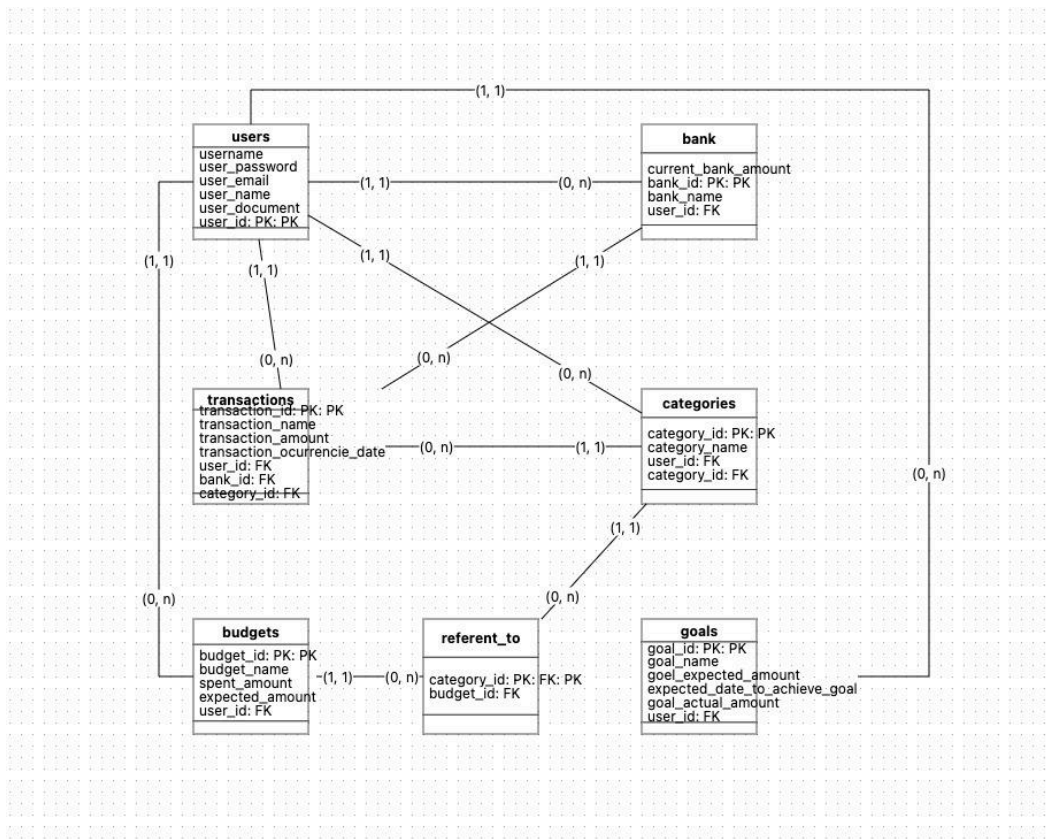


- Diagrama conceitual



- Diagrama lógico



- Tabelas preenchidas

- Banco

	bank_id	current_bank_amount	bank_name	user_id
▶	1	50	Banco Rico	1
	2	500	Banco Rico	2
	3	40	Banco Rico	3
	4	540	Banco Rico	4
	5	5033	Banco Rico	5
	6	501	Banco Rico	6
	7	650	Banco Rico	7
	8	750	Banco Rico	8
	9	90	Banco Rico	9
	10	100	Banco Rico	10
*	NULL	NULL	NULL	NULL

- Categoria

	category_id	category_name	user_id	sub_category_id
▶	1	Viajar	1	NULL
	2	Transporte	2	NULL
	3	Viajar	3	NULL
	4	Roupa	4	NULL
	5	Viajar	5	NULL
	6	Viajar	6	NULL
	7	Viajar	7	NULL
	8	Viajar	8	NULL
	9	Viajar	9	NULL
	10	Viajar	10	NULL
*	NULL	NULL	NULL	NULL

- Objetivo

	goal_id	goal_name	goal_expected_amount	expected_date_to_achieve_goal	goal_actual_amount	user_id
	1	Viajar Urubici	2000	2025-01-01	30	1
	2	Viajar Urubici	2000	2025-01-01	30	1
	3	Viajar Paris	20000	2027-01-01	30	3
	4	Viajar Urubici	2000	2025-01-01	30	3
	5	Comprar carro	25000	2028-01-01	30	2
	6	Tenis	300	2025-01-01	30	4
	7	Viajar Capão	2000	2025-01-01	30	5
	8	Viajar Gramado	2000	2025-01-01	30	6
	9	Viajar Canela	2000	2025-01-01	30	7
	10	Viajar São Francisco de Paula	2000	2025-01-01	30	8
	11	Viajar São Borja	2000	2025-01-01	30	9
*	NULL	NULL	NULL	NULL	NULL	NULL

- Orçamento

	budget_id	budget_name	spent_amount	expected_amount	user_id
▶	1	budget	50	100	1
	2	budget	50	100	1
	3	budget	50	100	2
	4	budget	50	100	3
	5	budget	50	100	4
	6	budget	50	100	5
	7	budget	50	100	6
	8	budget	50	100	7
	9	budget	50	100	8
	10	budget	50	100	9
	11	budget	50	100	10
*	NULL	NULL	NULL	NULL	NULL

○ Refere se

	category_id	budget_id
▶ 1	1	
2	2	
3	3	
4	4	
5	5	
6	6	
7	7	
8	8	
9	9	
10	10	
*	NULL	NULL

○ Transação

	transaction_id	transaction_name	transaction_amount	transaction_ocurrence_date	user_id	bank_id	category_id
▶ 1	1	viagem 1/10	50	2024-04-14 13:55:26	1	1	1
2	2	viagem 2/10	50	2024-06-04 20:33:21	1	1	1
3	3	viagem 3/10	50	2024-06-04 20:33:21	1	1	1
4	4	viagem 4/10	50	2024-06-04 20:33:21	1	1	1
5	5	viagem 5/10	50	2024-06-04 20:33:21	1	1	1
6	6	viagem 6/10	50	2024-06-04 20:33:21	1	1	1
7	7	viagem 7/10	50	2024-06-04 20:33:21	1	1	1
8	8	viagem 8/10	50	2024-06-04 20:33:21	1	1	1
9	9	viagem 9/10	50	2024-06-04 20:33:21	1	1	1
10	10	viagem 10/10	50	2024-06-04 20:33:21	1	1	1
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

○ Usuário

	user_id	username	user_password	user_email	user_name	user_document
▶ 1	1	tatitata	123	thais@mail.com	Thais Santos	1234567891
2	2	anabeatriz	124	anabeatriz@mail.com	Ana Beatriz	1234567892
3	3	gabrielmoraes	125	gabrielmoraes@mail.com	Gabriel Moraes	1234567893
4	4	crislinda	126	crisland@mail.com	Cristiane Santos	1234567894
5	5	felipefernando	127	felipe@mail.com	Felipe Santos	1234567895
6	6	pedrinho	128	pedro@mail.com	Pedro Antonio	1234567896
7	7	rafinha	129	rafael@mail.com	Rafael Santos	1234567897
8	8	nini	120	nidia@mail.com	Nidia Moraes	1234567898
9	9	ivan	121	ivan@mail.com	Ivan Freitas	1234567899
10	10	afonso	122	lucas@mail.com	Luccas Afonso	1234567890
*	NULL	NULL	NULL	NULL	NULL	NULL

- Querys
 - Order by
 - Ordena os usuários pelo nome alfabeticamente

85 • `select user_name from user order by user_name`

Result Grid | Filter Rows: | Export: | Wrap Cell

user_name
Ana Beatriz
Cristiane Santos
Felipe Santos
Gabriel Moraes
Ivan Freitas
Luccas Afonso
Nidia Moraes
Pedro Antonio
Rafael Santos
Thais Santos

- Duas tabelas relacionadas
 - Relaciona as transações com o usuário ao qual elas pertencem. No exemplo, o usuário Thais Santos foi o único a fazer transações.

86 • `select t.transaction_name, u.user_name from transaction t`
 87 `left join user u on u.user_id = t.user_id;`

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

transaction_name	user_name
viagem 1/10	Thais Santos
viagem 2/10	Thais Santos
viagem 3/10	Thais Santos
viagem 4/10	Thais Santos
viagem 5/10	Thais Santos
viagem 6/10	Thais Santos
viagem 7/10	Thais Santos
viagem 8/10	Thais Santos
viagem 9/10	Thais Santos
viagem 10/10	Thais Santos

- Três tabelas relacionadas
 - Relaciona as transações aos usuários aos quais elas pertencem, e a categoria a que as transações pertencem. No exemplo o usuário Thais Santos tem várias transações relacionadas a categoria viagem.

```

86 • select t.transaction_name, u.user_name, c.category_name from transaction t
87 left join user u on u.user_id = t.user_id
88 left join category c on c.category_id = t.category_id;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

transaction_name	user_name	category_name
viagem 1/10	Thais Santos	Viajar
viagem 2/10	Thais Santos	Viajar
viagem 3/10	Thais Santos	Viajar
viagem 4/10	Thais Santos	Viajar
viagem 5/10	Thais Santos	Viajar
viagem 6/10	Thais Santos	Viajar
viagem 7/10	Thais Santos	Viajar
viagem 8/10	Thais Santos	Viajar
viagem 9/10	Thais Santos	Viajar
viagem 10/10	Thais Santos	Viajar

- Between ou in
 - Seleciona todos os objetivos dos usuários 2 e 3.

```

89 • select * from goal where user_id in (2,3);

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

goal_id	goal_name	goal_expected_amount	expected_date_to_achieve_goal	goal_actual_amount	user_id
5	Comprar carro	25000	2028-01-01	30	2
3	Viajar Paris	20000	2027-01-01	30	3
4	Viajar Urubici	2000	2025-01-01	30	3
*	NULL	NULL	NULL	NULL	NULL

- Like
 - Seleciona todos os usuários em que o último nome é Santos.

```

90 • select * from user where user_name like '% Santos';

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

user_id	username	user_password	user_email	user_name	user_document
1	tatitata	123	thais@mail.com	Thais Santos	1234567891
4	crislinda	126	crisland@mail.com	Cristiane Santos	1234567894
5	felipefernando	127	felipe@mail.com	Felipe Santos	1234567895
7	rafinha	129	rafael@mail.com	Rafael Santos	1234567897
*	NULL	NULL	NULL	NULL	NULL

- Função de agregação
 - Soma o valor total das transações do usuário 1.

```
90 • select * from user where user_name like '%SANTOS';
```

```
91 • select sum(transaction_amount) as total from transaction where user_id = 1
```

Result Grid |  Filter Rows: | Export:  | Wrap Cell Content: 

total
500

- Left outer join
 - Relaciona as transações com o usuário ao qual elas pertencem.

```
86 • select t.transaction_name, u.user_name from transaction t  
87 left join user u on u.user_id = t.user_id;
```

Result Grid |  Filter Rows: | Export:  | Wrap Cell Content: 

transaction_name	user_name
viagem 1/10	Thais Santos
viagem 2/10	Thais Santos
viagem 3/10	Thais Santos
viagem 4/10	Thais Santos
viagem 5/10	Thais Santos
viagem 6/10	Thais Santos
viagem 7/10	Thais Santos
viagem 8/10	Thais Santos
viagem 9/10	Thais Santos
viagem 10/10	Thais Santos

- Right outer join
 - Relaciona usuário e transação, trazendo todos usuários com ou sem transações associadas

```

98 select u.user_name,t.transaction_name
99 from transaction t
100 right outer join user u on u.user_id = t.user_id

```

	user_name	transaction_name
▶	Thais Santos	viagem 10/10
	Thais Santos	viagem 9/10
	Thais Santos	viagem 8/10
	Thais Santos	viagem 7/10
	Thais Santos	viagem 6/10
	Thais Santos	viagem 5/10
	Thais Santos	viagem 4/10
	Thais Santos	viagem 3/10
	Thais Santos	viagem 2/10
	Thais Santos	viagem 1/10
	Ana Beatriz	NULL
	Gabriel Mor...	NULL
	Cristiane Sa...	NULL
	Felipe Santos	NULL
	Pedro Antonio	NULL
	Rafael Santos	NULL
	Nidia Moraes	NULL
	Ivan Freitas	NULL
	Luccas Afonso	NULL

- Distinct
 - Seleciona todos os usuários que já fizeram transações sem repetir os usuários.

```

92 select distinct user_id from transaction;






```

	user_id
▶	1

- Group by e having
 - Seleciona os valores totais de objetivo por usuário, apenas os com valor maior que 2000

```
93 • select user_id,sum(goel_expected_amount) as 'valor total objetivo'  
94 from goal  
95 group by user_id  
96 having sum(goel_expected_amount)>2000
```

<

Result Grid   Filter Rows: Export:  Wrap Cell Content:  

	user_id	valor total objetivo
▶	1	4000
	2	25000
	3	22000