

Pontifícia Universidade Católica do Rio Grande do Sul
Infraestrutura para Gestão de Dados
Engenharia de Software

Carolina Ferreira e Mateus Caçabuena

Trabalho Prático 2

Porto Alegre
2025

Sumário

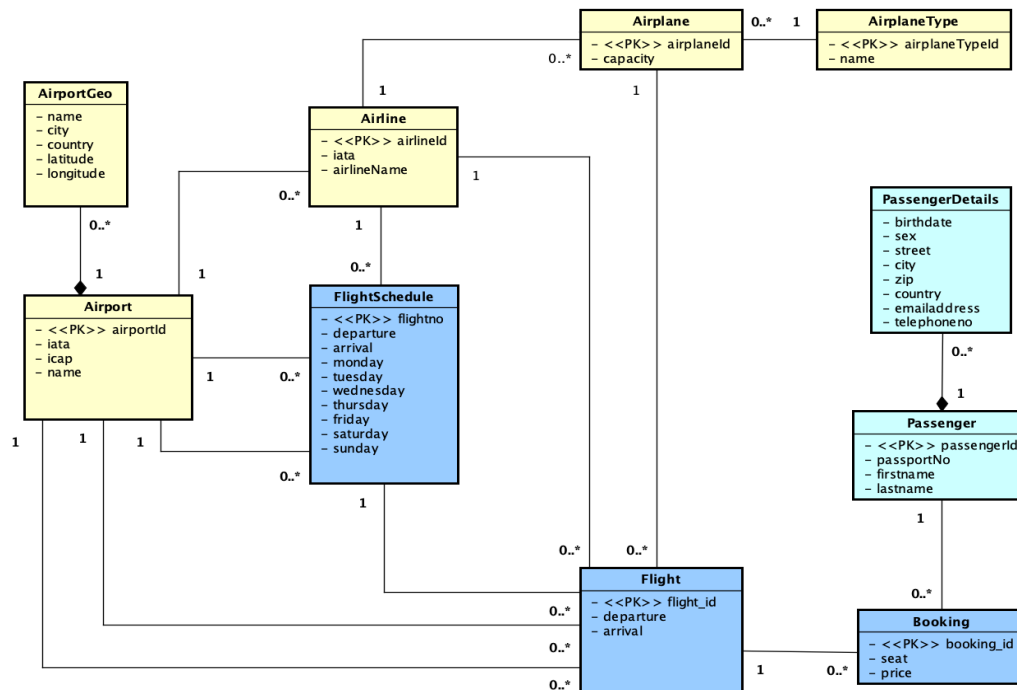
1. Introdução.....	3
2. Descrição das Consultas CQL	4
2.1. Consulta Q1 → Q2	4
2.2. Consulta Q3 → Q4	4
2.3. Consulta Q5 → Q6	4
3. Esquema Lógico	5
3.1. Diagrama Q1 → Q2	5
3.2. Diagrama Q3 → Q4	6
3.3. Diagrama Q5 → Q6	6
4. Comandos CQL DDL	7
4.1. CQL Q1 → Q2	7
4.2. CQL Q3 → Q4	8
4.3. CQL Q5 → Q6	9
5. Comandos CQL DML.....	10
6. Comandos CQL DQL	13

1. Introdução

Este relatório tem como objetivo documentar o processo de modelagem e implementação de um banco de dados não relacional utilizando o Sistema Gerenciador de Banco de Dados (SGBD) Apache Cassandra, a partir de um esquema relacional previamente existente na instância Oracle da PUC-RS.

Foram definidas três sequências de consultas ($Q1 \rightarrow Q2$, $Q3 \rightarrow Q4$ e $Q5 \rightarrow Q6$), cada uma voltada a uma necessidade distinta de acesso à informação. Para cada sequência, foi desenvolvido um esquema lógico específico, implementado em keyspaces independentes. A modelagem foi realizada utilizando a ferramenta Hackolade, voltada à criação de esquemas físicos para bancos NoSQL. Em seguida, os keyspaces e tabelas foram criados em uma instância do Cassandra, com posterior inserção de dados utilizando comandos CQL DML, extraídos a partir do banco relacional original. Por fim, foram implementadas e testadas as consultas definidas no início do processo.

Esquema conceitual de referência



2. Descrição das Consultas CQL

Nesta seção, são apresentadas as consultas CQL definidas para cada uma das três sequências estabelecidas: $Q1 \rightarrow Q2$, $Q3 \rightarrow Q4$ e $Q5 \rightarrow Q6$. Cada par de consultas foi projetado com base em necessidades específicas de acesso aos dados no contexto do domínio aeroportuário, respeitando o princípio de modelagem orientada a consultas adotado pelo Cassandra.

2.1. Consulta $Q1 \rightarrow Q2$

Q1: Encontrar passageiros pelo sobrenome.

Q2: Com esses sobrenomes, buscar o país e número de assento destes passageiros.

2.2. Consulta $Q3 \rightarrow Q4$

Q3: Buscar aeroportos pelo nome.

Q4: Com esses aeroportos, buscar as companhias aéreas que a cidade desse aeroporto possui.

2.3. Consulta $Q5 \rightarrow Q6$

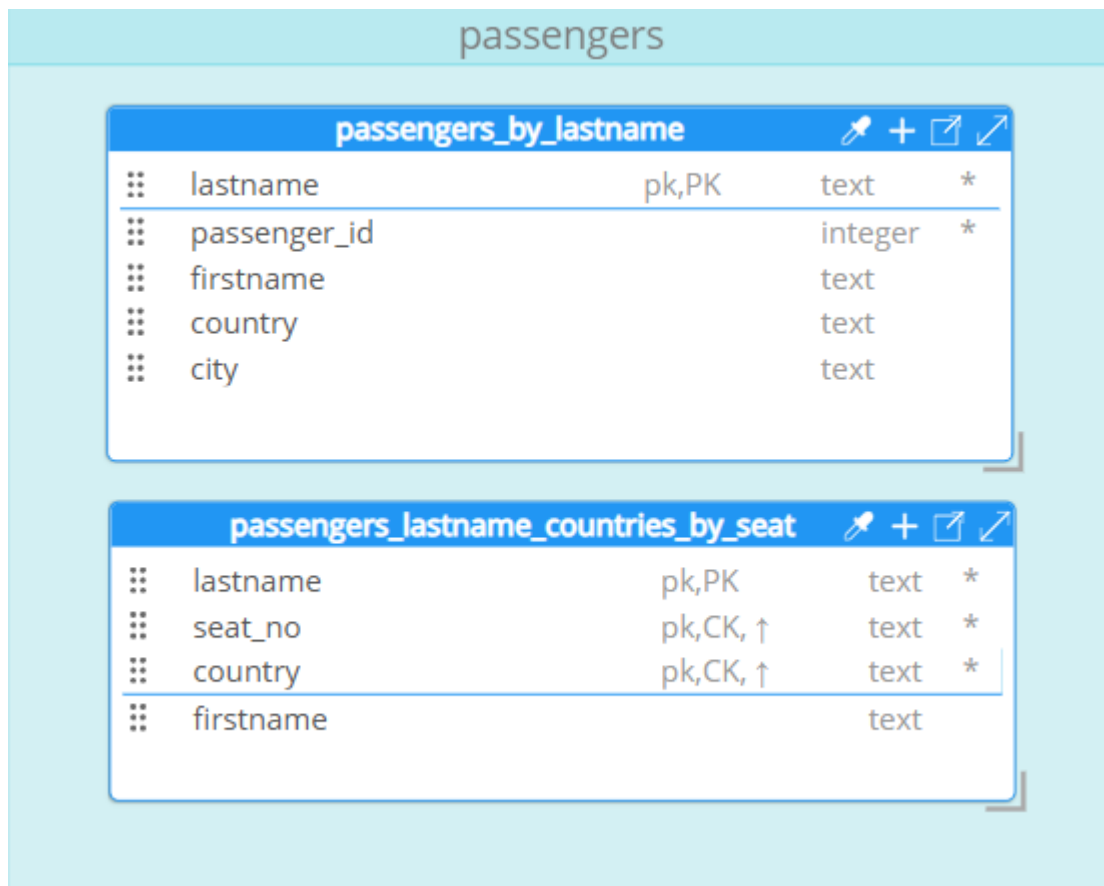
Q5: Buscar aviões pelo tipo.

Q6: Com o tipo destes aviões, descobrir os voos que este tipo possui.

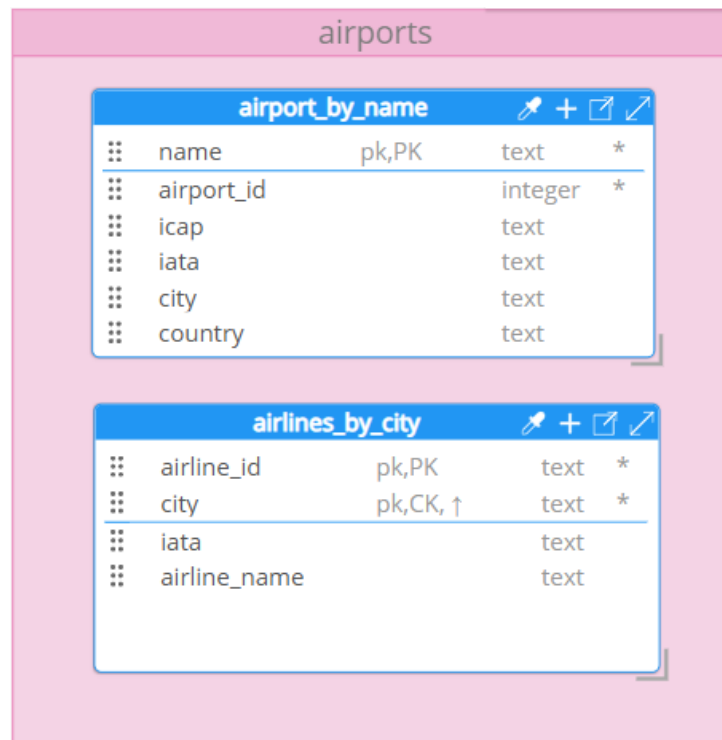
3. Esquema Lógico

Nesta seção, é apresentado o esquema lógico desenvolvido para o Apache Cassandra, com base nas sequências de consultas previamente definidas. Cada par de consultas foi implementado em um keyspace independente, conforme as boas práticas de modelagem orientada a consultas. O modelo foi construído utilizando a ferramenta Hackolade, detalhando a estrutura de cada tabela, suas chaves primárias (PK), chaves de clusterização (CK) e os atributos necessários para suportar as operações com eficiência.

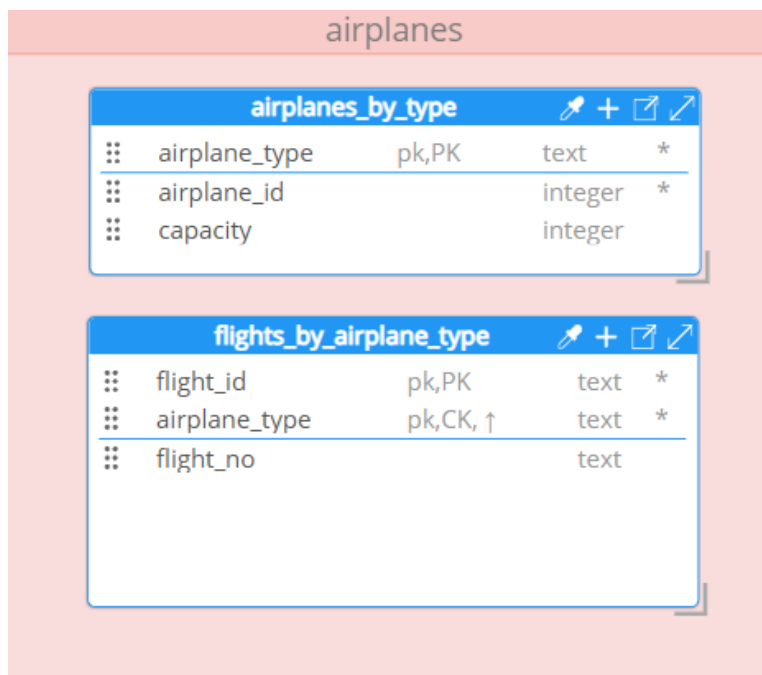
3.1. Diagrama Q1 → Q2



3.2. Diagrama Q3 → Q4



3.3. Diagrama Q5 → Q6



4. Comandos CQL DDL

Esta seção apresenta os **comandos CQL DDL (Data Definition Language)** utilizados para a criação dos keyspaces e tabelas no Cassandra. Os comandos foram elaborados com base no esquema lógico definido para cada sequência de consultas, especificando a estrutura das tabelas, tipos de dados, chaves primárias e de clusterização, de forma a garantir eficiência nas operações de leitura e conformidade com os requisitos do modelo distribuído do Cassandra.

4.1. CQL Q1 → Q2

```
1 CREATE KEYSPACE IF NOT EXISTS passengers
2 WITH REPLICATION = {
3     'class' : 'SimpleStrategy',
4     'replication_factor' : 1
5 }
6 AND DURABLE_WRITES = FALSE;
7 -- DROP KEYSPACE IF EXISTS passengers;
8
9 USE passengers;
10
11 CREATE TABLE passengers_by_lastname (
12     lastname TEXT,
13     passenger_id INT,
14     firstname TEXT,
15     country TEXT,
16     city TEXT,
17     PRIMARY KEY (lastname)
18 );
19
20 CREATE TABLE passengers_lastname_countries_by_seat (
21     lastname TEXT,
22     passenger_id INT,
23     country TEXT,
24     seat_no TEXT,
25     PRIMARY KEY ((lastname), seat_no, country)
26 );
```

4.2. CQL Q3 → Q4



```
1 CREATE KEYSPACE IF NOT EXISTS airports
2 WITH REPLICATION = {
3     'class' : 'SimpleStrategy',
4     'replication_factor' : 1
5 }
6 AND DURABLE_WRITES = FALSE;
7 -- DROP KEYSPACE IF EXISTS airports;
8
9 USE airports;
10
11 CREATE TABLE airports_by_name (
12     name TEXT,
13     airport_id INT,
14     icao TEXT,
15     iata TEXT,
16     city TEXT,
17     country TEXT,
18     PRIMARY KEY (name),
19 );
20
21 CREATE TABLE airlines_by_city (
22     airline_id INT,
23     city TEXT,
24     iata TEXT,
25     airline_name TEXT,
26     PRIMARY KEY ((airline_id), city)
27 );
```


4.3. CQL Q5 → Q6



```
1 CREATE KEYSPACE IF NOT EXISTS airplanes
2 WITH REPLICATION = {
3     'class' : 'SimpleStrategy',
4     'replication_factor' : 1
5 }
6 AND DURABLE_WRITES = FALSE;
7 -- DROP KEYSPACE IF EXISTS airplanes;
8
9 USE airplanes;
10
11 CREATE TABLE airplanes_by_type (
12     airplane_type TEXT,
13     airplane_id INT,
14     capacity INT,
15     PRIMARY KEY (airplane_type),
16 );
17
18 CREATE TABLE flights_by_airplane_type (
19     flight_id INT,
20     airplane_type TEXT,
21     flight_no TEXT,
22     PRIMARY KEY ((flight_id), airplane_type)
23 );
```

5. Comandos CQL DML

Nesta seção, são apresentados os comandos CQL DML (Data Manipulation Language) responsáveis pela inserção de dados nas tabelas criadas. Os dados foram extraídos do esquema relacional original, adaptados para o formato exigido pelo Cassandra, respeitando a estrutura de cada tabela e suas chaves definidas.

5.1. CQL Q1 → Q2

```
1 SELECT DISTINCT
2     'INSERT INTO passengers.passengers_by_lastname (lastname, passenger_id, firstname, country, city) VALUES (' ||
3     '"" || TRIM(passenger.lastname) || ', ' ||
4     passenger.passenger_id || ', ' ||
5     '"" || TRIM(passenger.firstname) || ', ' ||
6     '"" || TRIM(passengerDetails.country) || ', ' ||
7     '"" || TRIM(passengerDetails.city) || '');"
8 FROM
9     ACAMPOS.AIR_PASSENGERS passenger
10     JOIN ACAMPOS.AIR_PASSENGERS_DETAILS passengerDetails
11         ON passenger.passenger_id = passengerDetails.passenger_id
12 ;
13
14 SELECT DISTINCT
15     'INSERT INTO passengers.passengers_lastname_countries_by_seat (lastname, passenger_id, country, seat_no) VALUES (' ||
16     '"" || TRIM(passenger.lastname) || ', ' ||
17     passenger.passenger_id || ', ' ||
18     '"" || TRIM(passengerDetails.country) || ', ' ||
19     '"" || TRIM(booking.seat) || '');"
20 FROM
21     ACAMPOS.AIR_PASSENGERS passenger
22     JOIN ACAMPOS.AIR_PASSENGERS_DETAILS passengerDetails
23         ON passenger.passenger_id = passengerDetails.passenger_id
24     JOIN ACAMPOS.AIR_BOOKINGS booking
25         ON passenger.passenger_id = booking.passenger_id
26 ;
```

5.2. CQL Q3 → Q4

```
1  SELECT DISTINCT
2      'INSERT INTO airports.airports_by_name (name, airport_id, icao, iata, city, country) VALUES (' ||
3      '""" || TRIM(airport.name) || ""', ' ||
4      airport.airport_id || ', ' ||
5      '""" || TRIM(airport.icao) || ""', ' ||
6      '""" || TRIM(airport.iata) || ""', ' ||
7      '""" || TRIM(airportGeography.city) || ""', ' ||
8      '""" || TRIM(airportGeography.country) || ""');'
9  FROM
10     ACAMPOS.AIR_AIRPORTS airport
11     JOIN ACAMPOS.AIR_AIRPORTS_GEO airportGeography
12         ON airport.airport_id = airportGeography.airport_id
13 ;
14
15 SELECT DISTINCT
16     'INSERT INTO airports.airlines_by_city (airline_id, city, iata, airline_name) VALUES (' ||
17     airline.airline_id || ', ' ||
18     '""" || TRIM(airportGeography.city) || ""', ' ||
19     '""" || TRIM(airline.iata) || ""', ' ||
20     '""" || TRIM(airline.airline_name) || ""');'
21 FROM
22     ACAMPOS.AIR_AIRPORTS airport
23     JOIN ACAMPOS.AIR_AIRPORTS_GEO airportGeography
24         ON airport.airport_id = airportGeography.airport_id
25     JOIN ACAMPOS.AIR_AIRLINES airline
26         ON airport.airport_id = airline.base_airport_id
27 ;
```

5.3. CQL Q5 → Q6

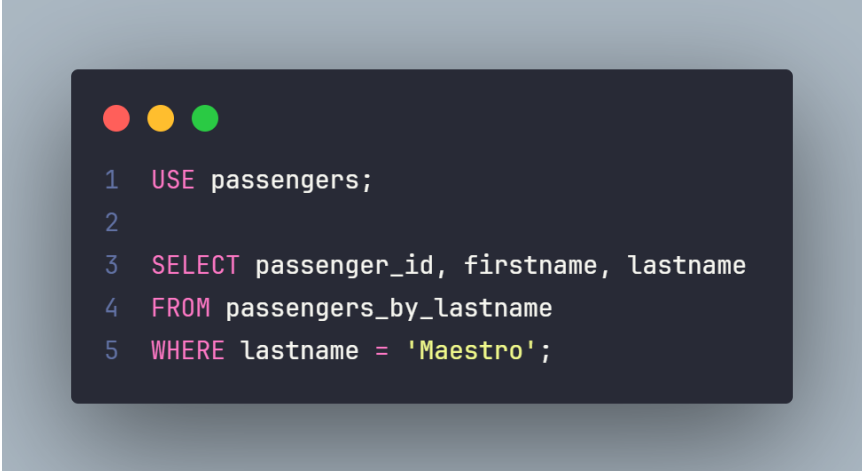
```
1  SELECT DISTINCT
2    'INSERT INTO airplanes.airplanes_by_type (airplane_type, airplane_id, capacity) VALUES (' ||
3    ''' || TRIM(airplaneType.name) || ''', ' ||
4    airplane.airplane_id || ', ' ||
5    airplane.capacity || ');'
6  FROM
7    ACAMPOS.AIR_AIRPLANES airplane
8    JOIN ACAMPOS.AIR_AIRPLANE_TYPES airplaneType
9      ON airplane.airplane_type_id = airplaneType.airplane_type_id
10 ;
11
12 SELECT DISTINCT
13   'INSERT INTO airplanes.flights_by_airplane_type (flight_id, airplane_type, flight_no) VALUES (' ||
14   flight.flight_id || ', ' ||
15   ''' || TRIM(airplaneType.name) || ''', ' ||
16   ''' || TRIM(flight.flightno) || ');'
17 FROM
18   ACAMPOS.AIR_FLIGHTS flight
19   JOIN ACAMPOS.AIR_AIRPLANES airplane
20     ON flight.airplane_id = airplane.airplane_id
21   JOIN ACAMPOS.AIR_AIRPLANE_TYPES airplaneType
22     ON airplane.airplane_type_id = airplaneType.airplane_type_id
23 ;
```

6. Comandos CQL DQL

Nesta seção, são descritos os **comandos CQL DQL (Data Query Language)** correspondentes às consultas $Q1 \rightarrow Q2$, $Q3 \rightarrow Q4$ e $Q5 \rightarrow Q6$. As consultas foram implementadas com base nos esquemas previamente modelados, visando garantir alta performance na recuperação dos dados. Cada comando reflete uma necessidade específica do negócio e foi projetado considerando as restrições e boas práticas da linguagem CQL no contexto do Cassandra.

6.1. $Q1 \rightarrow Q2$

6.1.1. Q1



```
1  USE passengers;
2
3  SELECT passenger_id, firstname, lastname
4  FROM passengers_by_lastname
5  WHERE lastname = 'Maestro';
```

passenger_id	firstname	lastname
1650	Johnny	Maestro

6.1.2. Q2



```
1 SELECT passenger_id, country, seat_no
2 FROM passengers_lastname_countries_by_seat
3 WHERE lastname = 'Maestro';
```

passenger_id	country	seat_no
1650	PAPUA NEW GUINEA	1F
1650	PAPUA NEW GUINEA	38B
1650	PAPUA NEW GUINEA	40A
1650	PAPUA NEW GUINEA	78E

6.2. Q3 → Q4

6.2.1. Q3



```
1 USE airports;
2
3 SELECT airport_id, name, icao, iata, city, country
4 FROM airports_by_name
5 WHERE city = 'TAIPEI' ALLOW FILTERING;
```

airport_id	name	icao	iata	city	country
2273	CHIANG KAI SHEK INTL	RCTP	TPE	TAIPEI	TAIWAN
11853	SUNGSHAN	RCSS	TSA	TAIPEI	TAIWAN

6.2.2. Q4



```
1 SELECT airline_id, airline_name
2 FROM airlines_by_city
3 WHERE city = 'TAIPEI' ALLOW FILTERING;
```

airline_id	airline_name
95	Taiwan Airlines

6.3. Q5 → Q6

6.3.1. Q5



```
1 USE airplanes;
2
3 SELECT airplane_id, capacity, airplane_type
4 FROM airplanes_by_type
5 WHERE airplane_type = 'Airbus A380';
```

airplane_id	capacity	airplane_type
1069	644	Airbus A380

6.3.2. Q6



```
1 SELECT flight_id, flight_no, airplane_type
2 FROM flights_by_airplane_type
3 WHERE airplane_type = 'Airbus A380' ALLOW FILTERING;
```

flight_id	flight_no	airplane_type
387294	KA5878	Airbus A380
642218	PE1747	Airbus A380
642358	PU1845	Airbus A380
639078	AR1842	Airbus A380
385099	AN1018	Airbus A380