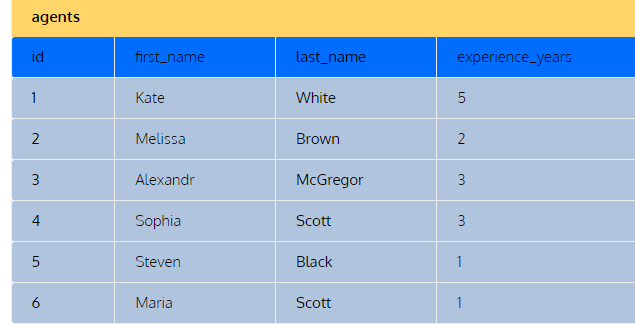
**What Are the Reasons for Duplicates in SQL JOINs?**

There are many possible reasons for getting duplicates in the result of your SQL JOIN query. I’ll go through the top 5 reasons; for each one, I’ll show a sample query with the problem and a corrected query to get a result without duplicates.

Let’s start by briefly reviewing the data to be used for our examples. Imagine we run a real estate agency that sells houses somewhere in the United States. We have tables with agents, customers, and sales. See below for what data is stored in each table.



CREATE OR REPLACE TABLE AGENTS

(

ID NUMBER(2,0),

FIRST\_NAME VARCHAR(25),

LAST\_NAME VARCHAR(30),

YEARS\_EXPERIENCE NUMBER(5,0)

);

INSERT INTO AGENTS VALUES

(1,'KATE','WHITE',5),

(2,'MELISSA','BROWN',2),

(3,'ALEXENDR','MCGREGOR',3),

(4,'SOFIA','SCOTT',3),

(5,'STEVEN','BLACK',1),

(6,'MARIA','SCOTT',1);

Table

Description automatically generated

create or replace table customers(

id int not null primary key,

first\_name varchar(20),

last\_name varchar(20),

email varchar(60) unique

);

insert into customers (id,first\_name,last\_name,email)

values (11,'Xaviera','Lopez','xaviera111111@gmail.com'),

(12,'Gabriel','Cumberly','gabrie111111@gmail.com'),

(13,'Elisabeth','Stevens','elisabeth111111@gmail.com'),

(14,'Oprah','Winfrey','oprah111111@gmail.com'),

(15,'Ivan','Lee','ivan111111@gmail.com');

Table

Description automatically generated

create table sales(id int,house\_id int,c\_date date,agent\_first\_name varchar(15),agent\_last\_name varchar(15),customer\_id int,price int);insert into sales values(101,1012,'2021-11-03','Kate','White',14,1200000),(102,2134,'2021-12-06','Sophia','Scott',12,950000),

(103,1015,'2021-12-10','Maria','Scott',13,800000),

(104,2013,'2021-12-12','Alexand','McGregor',15,1350000),

(105,2112,'2021-12-12','Alexand','McGregor',15,1450000),

(106,1010,'2022-01-10','Steven','Black',11,1500000);

Finally, unwanted duplicates in JOINs often result from an incorrect specification of joining conditions in SELF-JOINS– that is, when a table is joined with itself.

Let’s say we want our agents to form pairs for our next training. Obviously, we don’t want any agent to be paired with himself/herself. So, we might specify the ON condition a1.id <> a2.id:

|  |
| --- |
| SELECT      a1.first\_name as agent1\_first\_name,      a1.last\_name as agent1\_last\_name,      a1.experience\_years as agent1\_experience,      a2.first\_name as agent2\_first\_name,      a2.last\_name as agent2\_last\_name,      a2.experience\_years as agent2\_experience  FROM agents a1  JOIN agents a2  ON a1.id <> a2.id  ORDER BY a1.id; |

However, this query outputs each pair twice. For example, in the first row of the table below, Kate White is considered Agent 1, and Maria Scott is considered Agent 2. But closer to the end of the table, you get the same pair of agents but with Maria Scott as Agent 1 and Kate White as Agent 2.

Table

Description automatically generated

To solve this issue, you need to add an explicit condition to include each pair only once. One common solution is to specify the joining condition a1.id < a2.id. With this, you get the pair Kate White and Maria Scott but not vice versa. This is because Kate’s ID (1) is a lower number than Maria’s ID (6).

In practice, you may have some other conditions for pairing the agents. For instance, you may want to pair more experienced agents (3+ years) with less experienced ones (< 3 years). The corresponding filtering condition in WHERE solves the problem:

|  |
| --- |
| SELECT      a1.first\_name as agent1\_first\_name,      a1.last\_name as agent1\_last\_name,      a1.experience\_years as agent1\_experience,      a2.first\_name as agent2\_first\_name,      a2.last\_name as agent2\_last\_name,      a2.experience\_years as agent2\_experience  FROM agents a1  JOIN agents a2  ON a1.id <> a2.id  WHERE a1.experience\_years>=3 AND a2.experience\_years < 3  ORDER BY a1.id; |

Here’s the result:

Table

Description automatically generated