Introduction to Joins

🎯 Session Objectives:

- ✅ Understand various SQL constraints
- 🤝 Understand table relationships (ERDs)
- 🔗 Master different types of JOINs in SQL

```
USE weekend_analysis;

CREATE TABLE Locations (
    ID INT PRIMARY KEY AUTO_INCREMENT,
    PlaceInfo VARCHAR(100)
);

INSERT INTO Locations (PlaceInfo) VALUES
('Mumbai#Maharashtra@India'),
('Gorakhpur#UttarPradesh@India'),
('Bhopal#MadhyaPradesh@India'),
('Pune#Maharashtra@India'),
('Lucknow#UttarPradesh@India'),
('Jaipur#Rajsthan@India'),
('Kolkata#WestBengal@India'),
('Kashmir#J&K@India'),
('Gurugram#Haryana@India');

SELECT * FROM Locations;
```

```
SELECT
    PlaceInfo,
    SUBSTRING_INDEX(PlaceInfo, '#', 1) AS City,
    SUBSTR(PlaceInfo,1, INSTR(PlaceInfo, '#') - 1) AS City_Instr,

    SUBSTRING_INDEX(
                SUBSTRING_INDEX(PlaceInfo, '#', - 1),
        '@',1) AS State,
    SUBSTR(PlaceInfo,
        INSTR(PlaceInfo, '#') + 1,
        INSTR(PlaceInfo, '@') - INSTR(PlaceInfo, '#') - 1
        ) AS City_Instr,

    SUBSTRING_INDEX(PlaceInfo, '@', - 1) AS Country,
    SUBSTR(PlaceInfo,INSTR(PlaceInfo, '@') + 1, 5) AS Country_Instr
FROM Locations;
```

PlaceInfo	City	City_Instr	State	City_Instr	Country	Country_Instr
Mumbai#Maharashtra@India	Mumbai	Mumbai	Maharashtra	Maharashtra	India	India
Gorakhpur#UttarPradesh@India	Gorakhpur	Gorakhpur	UttarPradesh	UttarPradesh	India	India
Bhopal#MadhyaPradesh@India	Bhopal	Bhopal	MadhyaPradesh	MadhyaPradesh	India	India
Pune#Maharashtra@India	Pune	Pune	Maharashtra	Maharashtra	India	India
Lucknow#UttarPradesh@India	Lucknow	Lucknow	UttarPradesh	UttarPradesh	India	India
Jaipur#Rajsthan@India	Jaipur	Jaipur	Rajsthan	Rajsthan	India	India
Kolkata#WestBengal@India	Kolkata	Kolkata	WestBengal	WestBengal	India	India
Kashmir#J&K@India	Kashmir	Kashmir	J&K	J&K	India	India
Gurugram#Haryana@India	Gurugram	Gurugram	Haryana	Haryana	India	India

```sql
CREATE TABLE Users(
    ID INT NOT NULL,
    UserName VARCHAR(100) DEFAULT 'Unknown'
);

SHOW COLUMNS FROM Users;
```

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

Field	Type	Null	Key	Default	Extra
ID	int	NO		NULL	
UserName	varchar(100)	YES		Unknown	

```sql
INSERT INTO Users
VALUES('123','ShyamSundar');

SELECT * FROM Users;
```

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

ID	UserName
123	ShyamSundar

```sql
INSERT INTO Users
VALUES('k123','ShyamSundar');

-- Error Code: 1366. Incorrect integer value: 'k123' for column 'ID' at row 1
```

```sql
INSERT INTO Users(ID,UserName)
VALUES(121, 'Paramjeet Kaur');

INSERT INTO Users(ID)
VALUES(122);
```

ID	UserName
123	ShyamSundar
121	Paramjeet Kaur
122	Unknown
122	Unknown

```
ALTER TABLE Customers
MODIFY COLUMN CustomerKey INT NOT NULL;
```

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

Field	Type	Null	Key	Default	Extra
CustomerKey	int	NO		NULL	
Prefix	text	YES		NULL	
FirstName	varchar(50)	YES		NULL	
LastName	varchar(50)	YES		NULL	
FullName	varchar(100)	YES		NULL	
DateOfBirth	text	YES		NULL	
MaritalStatus	text	YES		NULL	
EmailAddress	varchar(100)	YES		NULL	
Gender	text	YES		NULL	
Regions	varchar(50)	YES		NULL	
AnnualIncome	int	YES		NULL	

```
ALTER TABLE Customers
MODIFY COLUMN CustomerKey INT NOT NULL;

ALTER TABLE Customers
ADD CONSTRAINT pk_cust PRIMARY KEY(CustomerKey);
```

Constraint

P.K

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

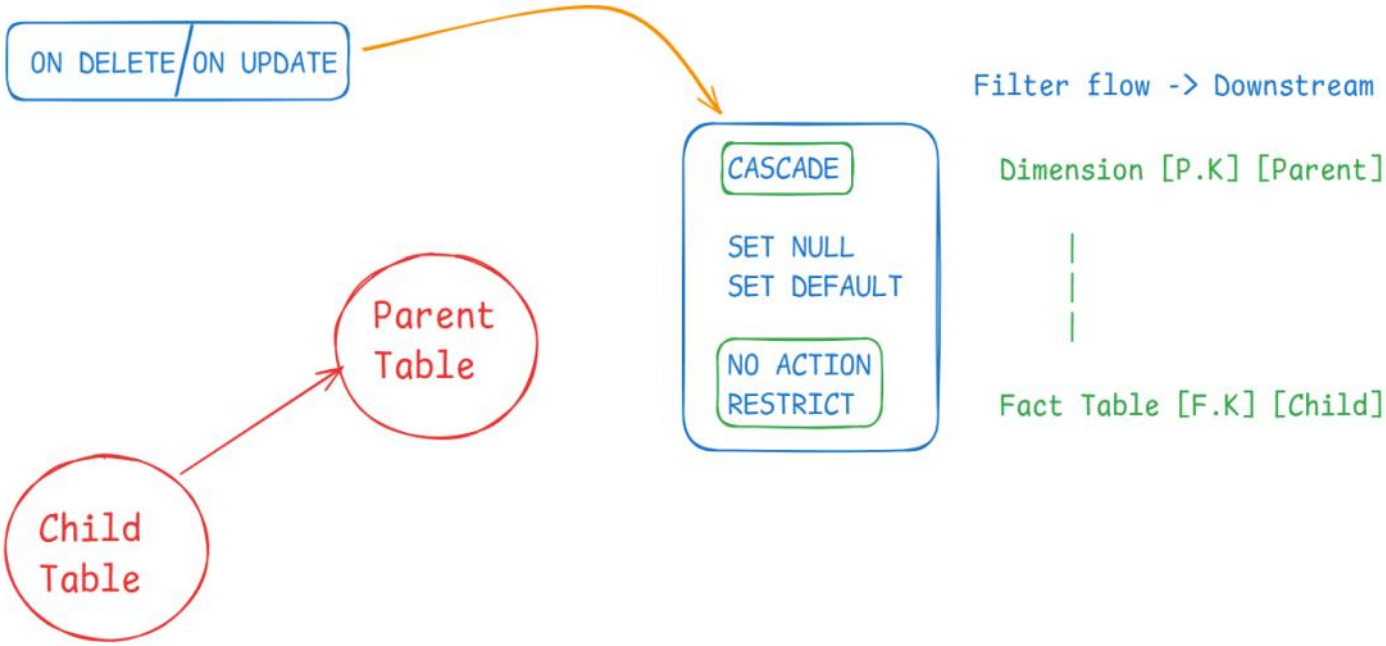
Field	Type	Null	Key	Default	Extra
CustomerKey	int	NO	PRI	NULL	
Prefix	text	YES		NULL	
FirstName	varchar(50)	YES		NULL	
LastName	varchar(50)	YES		NULL	
FullName	varchar(100)	YES		NULL	
DateOfBirth	text	YES		NULL	
MaritalStatus	text	YES		NULL	
EmailAddress	varchar(100)	YES		NULL	
Gender	text	YES		NULL	
Regions	varchar(50)	YES		NULL	
AnnualIncome	int	YES		NULL	

```
CREATE TABLE Employees(
    EmployeeID INT AUTO_INCREMENT,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    Email VARCHAR(100) UNIQUE NOT NULL,
    Salary DECIMAL(10,2), -- 99999999.99
    BirthDate DATE NOT NULL, -- YYYY-MM-DD
    Gender CHAR(1),
    IsActive BOOLEAN DEFAULT TRUE,
    PRIMARY KEY(EmployeeID)
);
```

Field	Type	Null	Key	Default	Extra
EmployeeID	int	NO	PRI	NULL	auto_increment
FirstName	varchar(50)	NO		NULL	
LastName	varchar(50)	NO		NULL	
Email	varchar(100)	NO	UNI	NULL	
Salary	decimal(10,2)	YES		NULL	
BirthDate	date	NO		NULL	
Gender	char(1)	YES		NULL	
IsActive	tinyint(1)	YES		1	

```
CREATE TABLE Orders(
    order_id INT PRIMARY KEY,
    customer_id INT,
    FOREIGN KEY (customer_id) REFERENCES Customers(customerKey)
);

DESC Orders;
```

Field	Type	Null	Key	Default	Extra
order_id	int	NO	PRI	NULL	
customer_id	int	YES	MUL	NULL	



```
-- CHECK CONSTRAINT
DESC Employees;

ALTER TABLE Employees
ADD COLUMN Age INT
AFTER BirthDate;
```

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

Field	Type	Null	Key	Default	Extra
EmployeeID	int	NO	PRI	NULL	auto_increment
FirstName	varchar(50)	NO		NULL	
LastName	varchar(50)	NO		NULL	
Email	varchar(100)	NO	UNI	NULL	
Salary	decimal(10,2)	YES		NULL	
BirthDate	date	NO		NULL	
Age	int	YES		NULL	
Gender	char(1)	YES		NULL	
IsActive	tinyint(1)	YES		1	

Check Constraint

```
-- CHECK CONSTRAINT
DESC Employees;

ALTER TABLE Employees
ADD COLUMN Age INT
AFTER BirthDate;

ALTER TABLE Employees
ADD CONSTRAINT check_age CHECK (age >= 21);

SELECT * FROM Employees;

INSERT INTO Employees(FirstName, LastName, Email, Salary , BirthDate, Age)
VALUES('Rajat','Thakur', 'rajat.thakur@gmail.com',1000000.00 , '1994-05-25', 31);

INSERT INTO Employees(FirstName, LastName, Email, Salary , BirthDate, Age)
VALUES('Akshay','Malik', 'akshay.malik@gmail.com',1000000.00 , '2005-05-25', 20);

-- Error Code: 3819. Check constraint 'check_age' is violated.
```

EmployeeID	FirstName	LastName	Email	Salary	BirthDate	Age	Gender	IsActive
1	Pragya	Chowdhury	pragya@gmail.com	1000000.00	1997-01-21	NULL	F	1
2	Soumya	Upadhyay	Soumya@gmail.com	1000000.00	1998-05-25	NULL	F	1
3	Akshay	malik	aksmalik@gmail.com	7000000.00	1997-04-19	NULL	M	1
4	Rajat	Thakur	rajat.thakur@gmail.com	1000000.00	1994-05-25	31	NULL	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
ALTER TABLE child_table
ADD [CONSTRAINT fk_name] FOREIGN KEY (child_col1, child_col2)
REFERENCES parent_table (parent_col1, parent_col2)
[ON DELETE reference_option]
[ON UPDATE reference_option];
```

```
DESC Products;

ALTER TABLE Products
ADD CONSTRAINT pk_productKey
PRIMARY KEY(ProductKey);
```

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

Field	Type	Null	Key	Default	Extra
ProductKey	int	NO	PRI	NULL	
ProductSubcategoryKey	int	YES		NULL	
ProductSKU	text	YES		NULL	
ProductName	text	YES		NULL	
ModelName	text	YES		NULL	
ProductDescription	text	YES		NULL	
ProductColor	text	YES		NULL	
ProductSize	text	YES		NULL	
ProductStyle	text	YES		NULL	
ProductCost	double	YES		NULL	

```
DESC Returns;

ALTER TABLE Returns
ADD CONSTRAINT fk_productkey
FOREIGN KEY (ProductKey)
REFERENCES Products(ProductKey);
```

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

Field	Type	Null	Key	Default	Extra
ReturnDate	text	YES		NULL	
TerritoryKey	int	YES		NULL	
ProductKey	int	YES	MUL	NULL	
ReturnQuantity	int	YES		NULL	

Check

```
-- CHECK CONSTRAINTS
-- Check Constraints on Products Ensure ProductCost > 0.

DESC Products;

ALTER TABLE Products
ADD CONSTRAINT chk_positive_cost
CHECK (ProductCost > 0);

INSERT INTO Products(ProductKey , ProductCost)
VALUES(12121 , -999);

-- Error Code: 3819. Check constraint 'chk_positive_cost' is violated.
```

```
-- Check Constraint [Making Sure ReturnQuantity is not negative or zero]

DESC Returns;

ALTER TABLE Returns
ADD CONSTRAINT chk_return_qty
CHECK (ReturnQuantity > 0);

INSERT INTO Returns(ReturnQuantity)
VALUES(-1);

-- Error Code: 3819. Check constraint 'chk_return_qty' is violated.
```

```
SET SQL_SAFE_UPDATES = 0;

UPDATE Customers
SET Gender = 'M'
WHERE Gender = 'NA';

-- 18 row(s) affected Rows matched: 18  Changed: 18  Warnings: 0
```

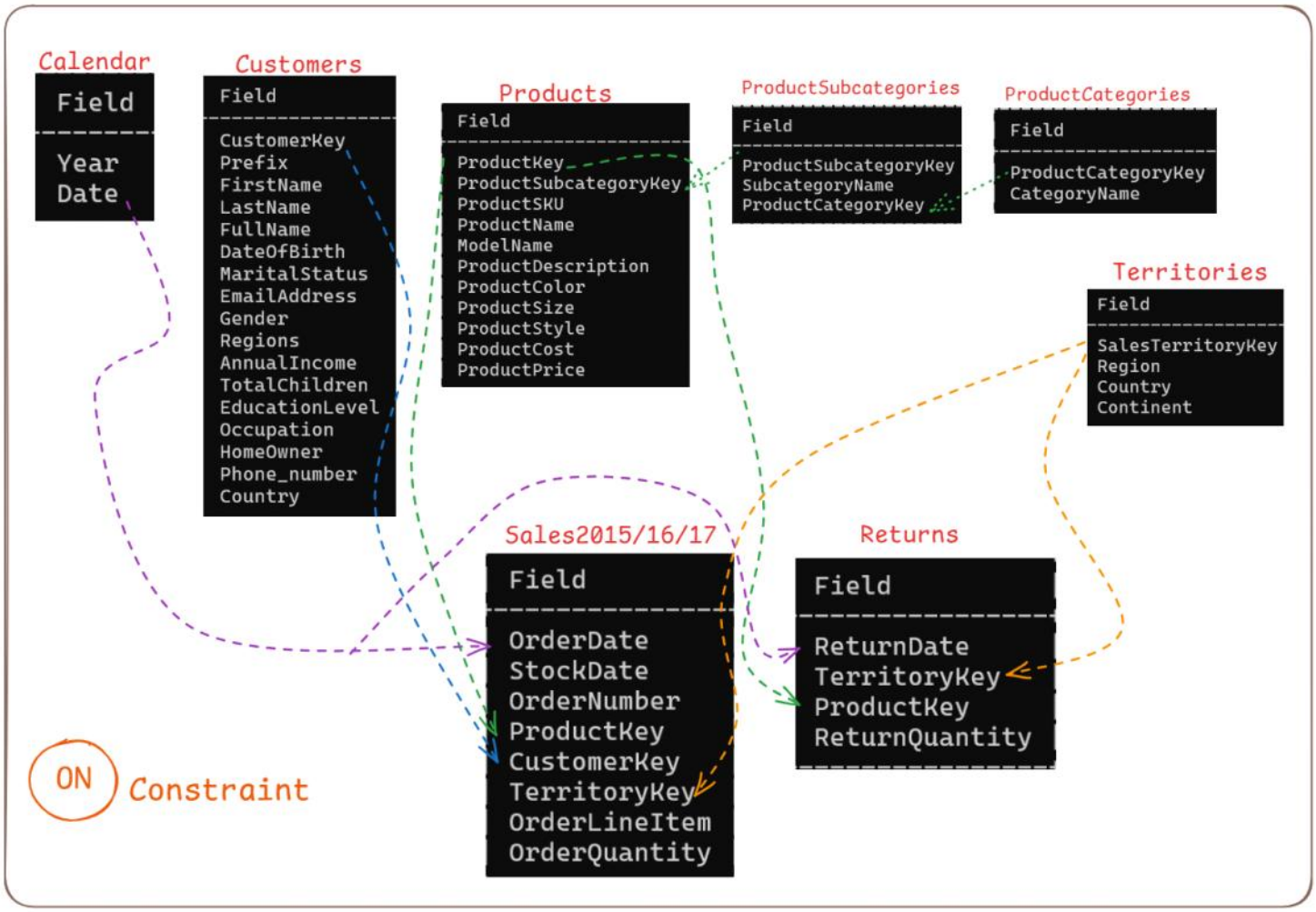
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Gender	Gender_Count
M	1039
F	1023

```sql
-- Applying a check constraints on Gender [Valid Input] ['M'/'F']
-- Goal : Limit Gender to 'M' or 'F'

DESC Customers;

ALTER TABLE Customers
ADD CONSTRAINT chk_gender
CHECK (Gender IN ('M' , 'F'));

SELECT DISTINCT Gender FROM Customers;

SELECT
    Gender,
    COUNT(*) AS Gender_Count
FROM Customers
GROUP BY 1;

SET SQL_SAFE_UPDATES = 0;

UPDATE Customers
SET Gender = 'M'
WHERE Gender = 'NA';

-- 18 row(s) affected Rows matched: 18  Changed: 18  Warnings: 0

INSERT INTO Customers(CustomerKey , Gender)
VALUES(12121, 'Male');
-- Error Code: 3819. Check constraint 'chk_gender' is violated.
```

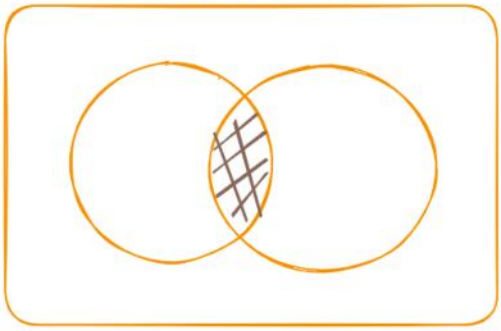
ERD [Entity-Relationship Diagram]

`product subcategories`



Joins

Inner Join



setA = {1,2,3,4,5,6,7,8,9}

setB = {2,4,6,8,7}

A ∩ B = {2,4,6,7,8}

```
486 •  SELECT * FROM Territories;
```

Result Grid | Filter Rows: | Export: | Wrap

SalesTerritoryKey	Region	Country	Continent
1	Northwest	United States	North America
2	Northeast	United States	North America
3	Central	United States	North America
4	Southwest	United States	North America
5	Southeast	United States	North America
6	Canada	Canada	North America
7	France	France	Europe
8	Germany	Germany	Europe
9	Australia	Australia	Pacific
10	United Kingdom	United Kingdom	Europe

```
SELECT
    *
FROM Territories
INNER JOIN Returns
ON Territories.SalesTerritoryKey = Returns.TerritoryKey;
```

1809 row(s) returned

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

SalesTerritoryKey	Region	Country	Continent	ReturnDate	TerritoryKey	ProductKey	ReturnQuantity
9	Australia	Australia	Pacific	1/18/2015	9	312	1
10	United Kingdom	United Kingdom	Europe	1/18/2015	10	310	1
8	Germany	Germany	Europe	1/21/2015	8	346	1
4	Southwest	United States	North America	1/22/2015	4	311	1
6	Canada	Canada	North America	2/2/2015	6	312	1
1	Northwest	United States	North America	2/15/2015	1	312	1
9	Australia	Australia	Pacific	2/19/2015	9	311	1
8	Germany	Germany	Europe	2/24/2015	8	314	1
8	Germany	Germany	Europe	3/8/2015	8	350	1
9	Australia	Australia	Pacific	3/13/2015	9	350	1
4	Southwest	United States	North America	3/14/2015	4	346	1

```
SELECT
    t.Continent,
    t.Country,
    t.Region,
    r.ReturnQuantity
FROM Territories AS t
INNER JOIN Returns AS r
ON t.SalesTerritoryKey = r.TerritoryKey;
```

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

Continent	Country	Region	ReturnQuantity
Pacific	Australia	Australia	1
Europe	United Kingdom	United Kingdom	1
Europe	Germany	Germany	1
North America	United States	Southwest	1
North America	Canada	Canada	1
North America	United States	Northwest	1
Pacific	Australia	Australia	1
Europe	Germany	Germany	1
Europe	Germany	Germany	1
Pacific	Australia	Australia	1
North America	United States	Southwest	1

```
SELECT
    t.Continent,
    t.Country,
    t.Region,
    SUM(r.ReturnQuantity) AS TotalReturns
FROM Territories AS t
INNER JOIN Returns AS r
ON t.SalesTerritoryKey = r.TerritoryKey
GROUP BY 1,2,3;
```

8 row(s) returned

Continent	Country	Region	TotalReturns
Pacific	Australia	Australia	404
Europe	United Kingdom	United Kingdom	204
Europe	Germany	Germany	163
North America	United States	Southwest	362
North America	Canada	Canada	238
North America	United States	Northwest	270
Europe	France	France	186
North America	United States	Southeast	1

```
SELECT
    t.SalesTerritoryKey,
    t.Continent,
    t.Country,
    t.Region,
    SUM(r.ReturnQuantity) AS TotalReturns
FROM Territories AS t
INNER JOIN Returns AS r
ON t.SalesTerritoryKey = r.TerritoryKey
GROUP BY 1,2,3,4;
```

8 records

SalesTerritoryKey	Continent	Country	Region	TotalReturns
1	North America	United States	Northwest	270
4	North America	United States	Southwest	362
5	North America	United States	Southeast	1
6	North America	Canada	Canada	238
7	Europe	France	France	186
8	Europe	Germany	Germany	163
9	Pacific	Australia	Australia	404
10	Europe	United Kingdom	United Kingdom	204

ChildTable [1309 Records]

```
498 •  SELECT * FROM Returns;
```

Result Grid | Filter Rows: | Export:

ReturnDate	TerritoryKey	ProductKey	ReturnQuantity
1/18/2015	9	312	1
1/18/2015	10	310	1
1/21/2015	8	346	1
1/22/2015	4	311	1
2/2/2015	6	312	1
2/15/2015	1	312	1
2/19/2015	9	311	1
2/24/2015	8	314	1
3/8/2015	8	350	1
3/13/2015	9	350	1
3/14/2015	4	346	1

Parent Table [10 records]

```
500 •  SELECT * FROM Territories;
```

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

SalesTerritoryKey	Region	Country	Continent
1	Northwest	United States	North America
2	Northeast	United States	North America
3	Central	United States	North America
4	Southwest	United States	North America
5	Southeast	United States	North America
6	Canada	Canada	North America
7	France	France	Europe
8	Germany	Germany	Europe
9	Australia	Australia	Pacific
10	United Kingdom	United Kingdom	Europe

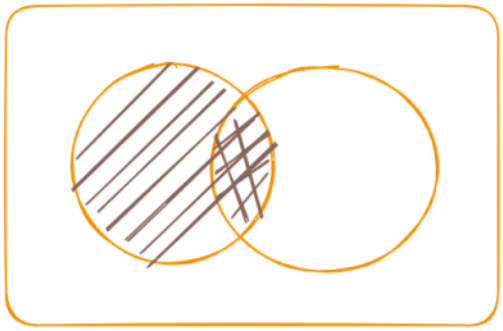
```
499 •  SELECT DISTINCT TerritoryKey FROM Returns;
```

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

TerritoryKey
1
4
5
6
7
8
9
10

Joins

Left Join



setA = {1,3,5,7,9}

setB = {2,4,6,8,7}

Left Join = {1,3,5,7,9}

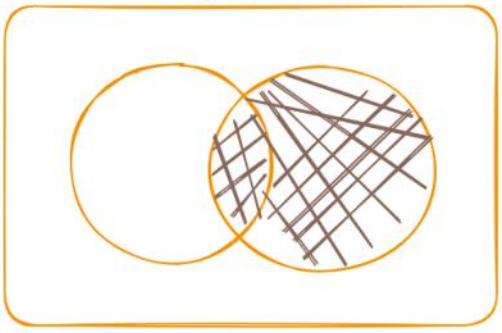
```sql
-- LEFT JOIN
SELECT
    t.SalesTerritoryKey,
    t.Continent,
    t.Country,
    t.Region,
    SUM(r.ReturnQuantity) AS TotalReturns
FROM Territories AS t
LEFT JOIN Returns AS r
ON t.SalesTerritoryKey = r.TerritoryKey
GROUP BY 1,2,3,4;
```

10 records

SalesTerritoryKey	Continent	Country	Region	TotalReturns
1	North America	United States	Northwest	270
2	North America	United States	Northeast	NULL
3	North America	United States	Central	NULL
4	North America	United States	Southwest	362
5	North America	United States	Southeast	1
6	North America	Canada	Canada	238
7	Europe	France	France	186
8	Europe	Germany	Germany	163
9	Pacific	Australia	Australia	404
10	Europe	United Kingdom	United Kingdom	204

Joins

Right Join



setA = {1,3,5,7,9}

setB = {2,4,6,8,7}

Right Join = {2,4,6,7,8}

```sql
-- RIGHT JOIN
SELECT
    t.SalesTerritoryKey,
    t.Continent,
    t.Country,
    t.Region,
    SUM(r.ReturnQuantity) AS TotalReturns
FROM Territories AS t
RIGHT JOIN Returns AS r
ON t.SalesTerritoryKey = r.TerritoryKey
GROUP BY 1,2,3,4;

-- In the above case right join will act as inner join
```

8 unique territoryKey

SalesTerritoryKey	Continent	Country	Region	TotalReturns
1	North America	United States	Northwest	270
4	North America	United States	Southwest	362
5	North America	United States	Southeast	1
6	North America	Canada	Canada	238
7	Europe	France	France	186
8	Europe	Germany	Germany	163
9	Pacific	Australia	Australia	404
10	Europe	United Kingdom	United Kingdom	204

```sql
-- RIGHT JOIN
SELECT
    t.SalesTerritoryKey,
    t.Continent,
    t.Country,
    t.Region,
    SUM(r.ReturnQuantity) AS TotalReturns
FROM Returns As r
RIGHT JOIN Territories AS t
ON t.SalesTerritoryKey = r.TerritoryKey
GROUP BY 1,2,3,4;

-- In the above case right join will act as Left join, just changing the order.
```

SalesTerritoryKey	Continent	Country	Region	TotalReturns
1	North America	United States	Northwest	270
2	North America	United States	Northeast	NULL
3	North America	United States	Central	NULL
4	North America	United States	Southwest	362
5	North America	United States	Southeast	1
6	North America	Canada	Canada	238
7	Europe	France	France	186
8	Europe	Germany	Germany	163
9	Pacific	Australia	Australia	404
10	Europe	United Kingdom	United Kingdom	204

Find the All Category Name and their Returns.

Total Returns

Bikes
Clothings
Accessories
Components

```sql
SELECT
    pc.CategoryName,
    SUM(r.ReturnQuantity) AS TotalReturns
FROM ProductCategories AS pc
JOIN ProductSubcategories As ps
ON pc.ProductCategoryKey = ps.ProductCategoryKey -- ps & pc
JOIN Products AS p
ON ps.ProductSubcategoryKey = p.ProductSubcategoryKey -- ps * pc * p
JOIN Returns AS r
ON p.productKey = r.ProductKey
GROUP BY 1;
```

CategoryName	TotalReturns
Bikes	429
Accessories	1130
Clothing	269

```
-- LEFT JOIN
SELECT
    pc.CategoryName,
    SUM(r.ReturnQuantity) AS TotalReturns
FROM ProductCategories AS pc
LEFT JOIN ProductSubcategories As ps
ON pc.ProductCategoryKey = ps.ProductCategoryKey -- ps & pc
LEFT JOIN Products AS p
ON ps.ProductSubcategoryKey = p.ProductSubcategoryKey -- ps * pc * p
LEFT JOIN Returns AS r
ON p.productKey = r.ProductKey
GROUP BY 1;
```

CategoryName	TotalReturns
Bikes	429
Components	NULL
Clothing	269
Accessories	1130

```
572 •  SELECT * FROM ProductCategories;
```

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

	ProductCategoryKey	CategoryName
▶	1	Bikes
	2	Components
	3	Clothing
	4	Accessories