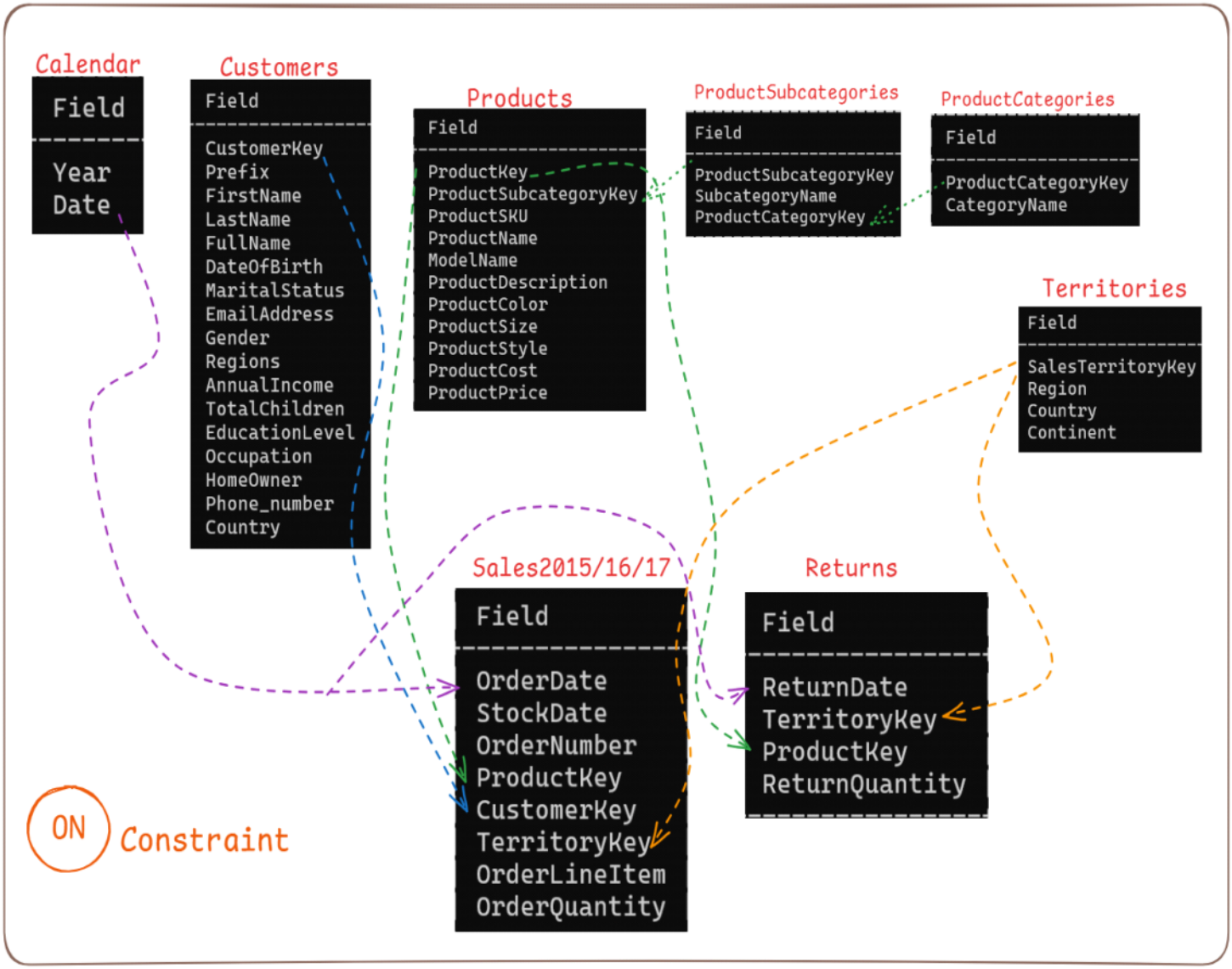
Advanced Concepts in SQL Joins

Session Objectives:

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

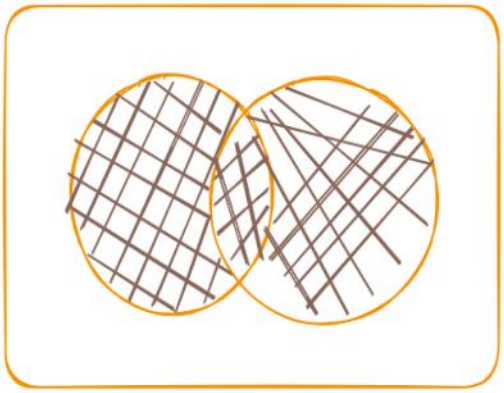


```sql
-- RIGHT JOIN
SELECT
    pc.CategoryName,
    SUM(r.ReturnQuantity) AS TotalReturns
FROM Returns r
RIGHT JOIN Products p
ON p.ProductKey = r.ProductKey
RIGHT JOIN ProductSubcategories ps
ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey
RIGHT JOIN ProductCategories pc
ON pc.ProductCategoryKey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

CategoryName	TotalReturns
Bikes	429
Components	NULL
Clothing	269
Accessories	1130

Joins

Full Join



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

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

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

Left JOIN Union {U} Right JOIN -> Full JOIN

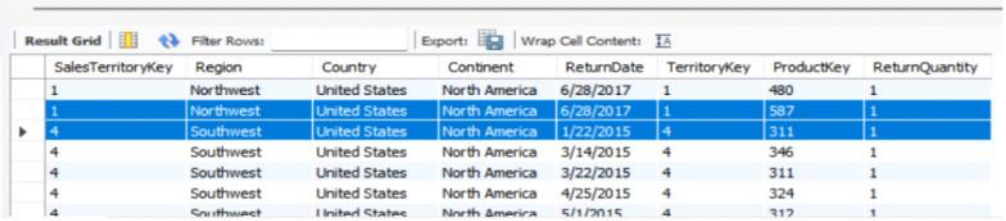
```sql
592   -- FULL JOIN
593 • SELECT t.* , r.*
594   FROM Territories t
595   LEFT JOIN Returns r
596   ON t.SalesTerritoryKey = r.TerritoryKey;
```

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

SalesTerritoryKey	Region	Country	Continent	ReturnDate	TerritoryKey	ProductKey	ReturnQuantity
2	Northeast	United States	North America	NULL	NULL	NULL	NULL
3	Central	United States	North America	NULL	NULL	NULL	NULL
4	Southwest	United States	North America	6/29/2017	4	587	1
4	Southwest	United States	North America	6/29/2017	4	356	1
4	Southwest	United States	North America	6/28/2017	4	528	2
4	Southwest	United States	North America	6/28/2017	4	387	1
4	Southwest	United States	North America	6/27/2017	4	537	1

UNION ALL -> FULL JOIN

```
598 •  SELECT t.* , r.*
599    FROM Territories t
600    RIGHT JOIN Returns r
601    ON t.SalesTerritoryKey = r.TerritoryKey;
```

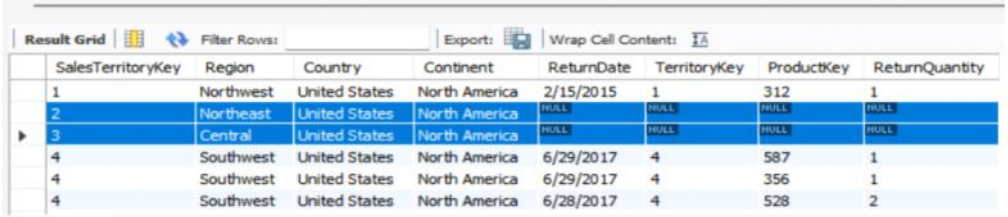


SalesTerritoryKey	Region	Country	Continent	ReturnDate	TerritoryKey	ProductKey	ReturnQuantity
1	Northwest	United States	North America	6/28/2017	1	480	1
1	Northwest	United States	North America	6/28/2017	1	587	1
4	Southwest	United States	North America	1/22/2015	4	311	1
4	Southwest	United States	North America	3/14/2015	4	346	1
4	Southwest	United States	North America	3/22/2015	4	311	1
4	Southwest	United States	North America	4/25/2015	4	324	1
4	Southwest	United States	North America	5/1/2015	4	312	1

```
592    -- FULL JOIN
593 •  SELECT t.* , r.*
594    FROM Territories t
595    LEFT JOIN Returns r
596    ON t.SalesTerritoryKey = r.TerritoryKey
597    UNION ALL -- 3620 row(s) returned
598    SELECT t.* , r.*
599    FROM Territories t
600    RIGHT JOIN Returns r
601    ON t.SalesTerritoryKey = r.TerritoryKey;
```

SalesTerritoryKey	Region	Country	Continent	ReturnDate	TerritoryKey	ProductKey	ReturnQuantity
1	Northwest	United States	North America	2/15/2015	1	312	1
2	Northeast	United States	North America	NULL	NULL	NULL	NULL
3	Central	United States	North America	NULL	NULL	NULL	NULL
4	Southwest	United States	North America	6/29/2017	4	587	1
4	Southwest	United States	North America	6/29/2017	4	356	1
4	Southwest	United States	North America	6/28/2017	4	528	2

```
604    -- FULL JOIN
605 •  SELECT t.* , r.*
606    FROM Territories t
607    LEFT JOIN Returns r
608    ON t.SalesTerritoryKey = r.TerritoryKey
609    UNION -- 1811 row(s) returned
610    SELECT t.* , r.*
611    FROM Territories t
612    RIGHT JOIN Returns r
613    ON t.SalesTerritoryKey = r.TerritoryKey;
```



SalesTerritoryKey	Region	Country	Continent	ReturnDate	TerritoryKey	ProductKey	ReturnQuantity
1	Northwest	United States	North America	2/15/2015	1	312	1
2	Northeast	United States	North America	NULL	NULL	NULL	NULL
3	Central	United States	North America	NULL	NULL	NULL	NULL
4	Southwest	United States	North America	6/29/2017	4	587	1
4	Southwest	United States	North America	6/29/2017	4	356	1
4	Southwest	United States	North America	6/28/2017	4	528	2

UNION VS UNION ALL → 'Sales2015' U 'Sales2016' U 'Sales2017'

```sql
-- UNION ALL VS UNION
-- CTE [Common Table Expression]

WITH AllSales AS(
    SELECT * FROM Sales2015
    UNION ALL
    SELECT * FROM Sales2016
    UNION ALL
    SELECT * FROM Sales2017
)
SELECT * FROM AllSales; -- 56046 row(s) returned
```

```sql
-- SUBQUERY -- UNION
SELECT *
FROM (
    SELECT * FROM Sales2015
    UNION
    SELECT * FROM Sales2016
    UNION
    SELECT * FROM Sales2017
) AS sales; -- 56046 row(s) returned
```

```sql
-- Find the name of the products bought by customers in all 3 years.
SELECT
    c.FullName AS CustomerName,
    p.ProductName,
    SUM(s.OrderQuantity) AS TotalOrders
FROM (
    SELECT * FROM Sales2015
    UNION
    SELECT * FROM Sales2016
    UNION
    SELECT * FROM Sales2017
) AS s
INNER JOIN Products p
ON p.ProductKey = s.ProductKey
INNER JOIN Customers c
ON c.CustomerKey = s.CustomerKey
GROUP BY 1,2
HAVING TotalOrders >= 15
ORDER BY 3 DESC;
```

CustomerName	ProductName	TotalOrders
APRIL SHAN	Touring Tire Tube	28
SAMANTHA JENKINS	Patch Kit/8 Patches	21
DALTON PEREZ	Patch Kit/8 Patches	21
JENNIFER SIMMONS	Road Tire Tube	19
HENRY GARCIA	Patch Kit/8 Patches	17
MASON ROBERTS	Patch Kit/8 Patches	17
JENNIFER SIMMONS	Patch Kit/8 Patches	17
ASHLEY HENDERSON	Patch Kit/8 Patches	16
APRIL SHAN	Patch Kit/8 Patches	15
CHARLES JACKSON	Road Tire Tube	15
ASHLEY HENDERSON	Road Tire Tube	15

```
WITH AllSales AS (
    SELECT * FROM Sales2015
    UNION
    SELECT * FROM Sales2016
    UNION
    SELECT * FROM Sales2017
)
SELECT
    c.FullName AS CustomerName,
    p.ProductName,
    SUM(s.OrderQuantity) AS TotalOrders
FROM AllSales s
INNER JOIN Products p
ON p.ProductKey = s.ProductKey
INNER JOIN Customers c
ON c.CustomerKey = s.CustomerKey
GROUP BY 1,2
HAVING TotalOrders >= 15
ORDER BY 3 DESC;
```

List ProductNames and ProductPrice that have never being returned.

```
SELECT
    ProductName,
    ProductPrice,
    r.ProductKey
FROM Products p
LEFT JOIN Returns r
ON p.ProductKey = r.ProductKey
WHERE r.ProductKey IS NULL; -- 169 row(s) returned
```

ProductName	ProductPrice	ProductKey
Mountain Bike Socks, M	9.5	NULL
Mountain Bike Socks, L	9.5	NULL
HL Road Frame - Red, 62	1263.4598	NULL
HL Road Frame - Red, 44	1263.4598	NULL
HL Road Frame - Red, 48	1263.4598	NULL
HL Road Frame - Red, 52	1263.4598	NULL
HL Road Frame - Red, 56	1263.4598	NULL
LL Road Frame - Black, 58	297.6346	NULL
LL Road Frame - Black, 60	297.6346	NULL
LL Road Frame - Black, 62	297.6346	NULL
LL Road Frame - Red, 44	306.5636	NULL
LL Road Frame - Red, 48	306.5636	NULL
LL Road Frame - Red, 52	306.5636	NULL
LL Road Frame - Red, 58	306.5636	NULL
LL Road Frame - Red, 60	306.5636	NULL
LL Road Frame - Red, 62	306.5636	NULL

List the products that are sold in all 3 years.
Ensuring the products appear in all 3 tables.

```
702 •  SELECT DISTINCT ProductKey FROM Sales2015;
```

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

ProductKey
332
312
350
338
310
314
345
313
351
344
326
348
311
324
342
340

```
IN (..... 2015)
AND IN (..... 2016)
AND IN (..... 2017)
```

```
SELECT
    p.ProductName
FROM Products p
WHERE p.ProductKey IN (SELECT DISTINCT ProductKey FROM Sales2015)
AND p.ProductKey IN (SELECT DISTINCT ProductKey FROM Sales2016)
AND p.ProductKey IN (SELECT DISTINCT ProductKey FROM Sales2017);
```

ProductName
Mountain-200 Silver, 38
Mountain-200 Silver, 42
Mountain-200 Silver, 46
Mountain-200 Black, 38
Mountain-200 Black, 42
Mountain-200 Black, 46
Road-250 Red, 58
Road-250 Black, 44
Road-250 Black, 48
Road-250 Black, 52
Road-250 Black, 58
Road-550-W Yellow, 38
Road-550-W Yellow, 40
Road-550-W Yellow, 42
Road-550-W Yellow, 44
Road-550-W Yellow, 48

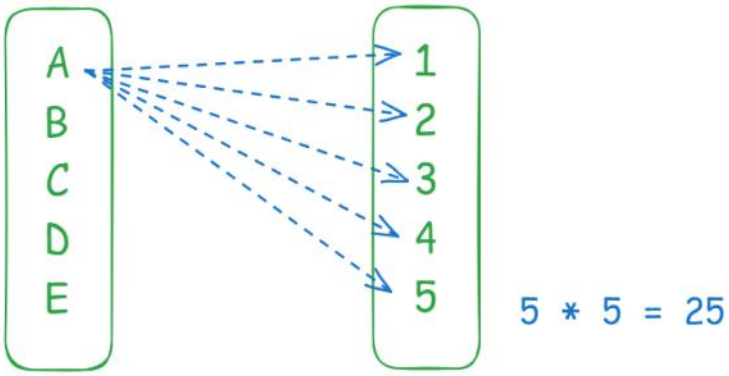
Find the Top 10 Products that are most Returned.

```
SELECT
    p.ProductName,
    SUM(r.ReturnQuantity) AS TotalReturnQuantity
FROM Products p
JOIN Returns r
ON p.ProductKey = r.ProductKey
GROUP BY 1
ORDER BY 2 DESC
LIMIT 10;
```

ProductName	TotalReturnQuantity
Water Bottle - 30 oz.	155
Patch Kit/8 Patches	95
Mountain Tire Tube	93
Mountain Bottle Cage	77
Sport-100 Helmet, Red	70
Road Tire Tube	67
Sport-100 Helmet, Blue	66
Road Bottle Cage	56
Fender Set - Mountain	54
Sport-100 Helmet, Black	52

```
SELECT
    ProductKey,
    ProductName,
    ProductCost,
    OrderQuantity
FROM Products p
JOIN Sales2017 s
ON p.ProductKey = s.ProductKey;
-- Error Code: 1052. Column 'ProductKey' in field list is ambiguous
```

Cross JOIN




```
-- CROSS JOIN
-- Product Categories [4] VS ProductSubcategories Table [37]

SELECT COUNT(*) FROM ProductSubcategories; -- 37
SELECT COUNT(*) FROM ProductCategories; -- 4

SELECT *
FROM ProductSubcategories
CROSS JOIN ProductCategories; -- 148 row(s) returned
```

ProductSubcategoryKey	SubcategoryName	ProductCategoryKey	ProductCategoryKey	CategoryName
1	Mountain Bikes	1	4	Accessories
1	Mountain Bikes	1	3	Clothing
1	Mountain Bikes	1	2	Components
1	Mountain Bikes	1	1	Bikes
2	Road Bikes	1	4	Accessories
2	Road Bikes	1	3	Clothing
2	Road Bikes	1	2	Components
2	Road Bikes	1	1	Bikes
3	Touring Bikes	1	4	Accessories
3	Touring Bikes	1	3	Clothing
3	Touring Bikes	1	2	Components
3	Touring Bikes	1	1	Bikes

SELF JOIN

Find the Customers with same Last Name

```
-- SELF JOIN
-- Find the Customers Pair who share the same last name
SELECT * FROM Customers;
```

```
SELECT
    c1.CustomerKey AS Customer1_Id,
    c1.FullName AS Customer1_Name,
    c2.CustomerKey AS Customer2_Id,
    c2.FullName AS Customer2_Name,
    c1.LastName
FROM Customers c1
JOIN Customers c2
ON c1.LastName = c2.LastName
AND c1.CustomerKey < c2.CustomerKey; -- 11481 row(s) returned
```

Customer1_Id	Customer1_Name	Customer2_Id	Customer2_Name	LastName
11000	JON YANG	11110	CURTIS YANG	YANG
11000	JON YANG	12001	SHANNON YANG	YANG
11000	JON YANG	12117	CAMERON YANG	YANG
11000	JON YANG	12139	FRANKLIN YANG	YANG
11000	JON YANG	12396	SUSAN YANG	YANG
11000	JON YANG	12564	MARSHALL YANG	YANG
11000	JON YANG	12566	OMAR YANG	YANG
11000	JON YANG	12786	JAMES YANG	YANG
11001	EUGENE HUANG	11220	ERICA HUANG	HUANG
11001	EUGENE HUANG	11464	ALEJANDRO HUANG	HUANG
11001	EUGENE HUANG	11515	SHANNON HUANG	HUANG
11001	EUGENE HUANG	11578	TRISHA HUANG	HUANG
11001	EUGENE HUANG	11943	DAWN HUANG	HUANG
11001	EUGENE HUANG	11945	DENNIS HUANG	HUANG
11001	EUGENE HUANG	11951	LACEY HUANG	HUANG

Find the Customers with same BirthDate

```
SELECT
    c1.CustomerKey AS Customer1_Id,
    c1.FullName AS Customer1_Name,
    c2.CustomerKey AS Customer2_Id,
    c2.FullName AS Customer2_Name,
    c1.DateOfBirth
FROM Customers c1
JOIN Customers c2
ON c1.DateOfBirth = c2.DateOfBirth
AND c1.CustomerKey < c2.CustomerKey;
```

231 row(s) returned

Customer1_Id	Customer1_Name	Customer2_Id	Customer2_Name	DateOfBirth
11004	ELIZABETH JOHNSON	11550	DEB TORRES	08/08/1968
11010	JACQUELYN SUAREZ	12002	ADRIANA LOPEZ	02/06/1964
11017	SHANNON WANG	12075	CAMERON GRIFFIN	26/06/1944
11018	CLARENCE RAI	12064	JOHN WHITE	10/09/1944
11031	THERESA RAMOS	12661	DEBORAH PAL	22/08/1947
11041	AMANDA CARTER	12029	DAMIEN HU	16/10/1977
11053	ANA PRICE	13039	TRISHA MA	20/08/1980
11054	DEANNA MUNOZ	12648	LORI DOMINGUEZ	03/10/1952
11055	GILBERT RAJE	11236	JEREMY BUTLER	03/05/1952
11056	MICHELE NATH	11312	SARA RICHARDSON	04/03/1953
11068	TIFFANY LIANG	12134	HECTOR ALONSO	23/09/1955
11068	TIFFANY LIANG	12770	CHARLES EVANS	23/09/1955
11075	FELICIA JIMENEZ	11529	AUSTIN BRYANT	16/11/1957
11081	SAVANNAH BAKER	11561	BRIANA DOMINGUEZ	24/07/1966
11083	ALYSSA COX	11572	BILLY MORENO	15/03/1966

IFNULL VS COALESCE()

IFNULL(exp1, exp2) -> String Column

COALESCE(exp1, exp2, exp3 ......) -> Numerical Column

```
SELECT
    CustomerKey,
    FullName,
    AnnualIncome,
    IFNULL(AnnualIncome , 'Not Available') AS IncomeCateogory
FROM Customers;
```

CustomerKey	FullName	AnnualIncome	IncomeCateogory
11000	JON YANG	90000	90000
11001	EUGENE HUANG	60000	60000
11002	RUBEN TORRES	60000	60000
11003	CHRISTY ZHU	NULL	Not Available
11004	ELIZABETH JOHNSON	80000	80000
11005	JULIO RUIZ	70000	70000
11007	MARCO MEHTA	60000	60000
11008	ROBIN VERHOFF	60000	60000
11009	SHANNON CARLSON	70000	70000
11010	JACQUELYN SUAREZ	70000	70000
11011	CURTIS LU	60000	60000
11012	LAUREN WALKER	100000	100000
11013	IAN JENKINS	100000	100000
11014	SYDNEY BENNETT	100000	100000
11015	CHLOE YOUNG	NULL	Not Available

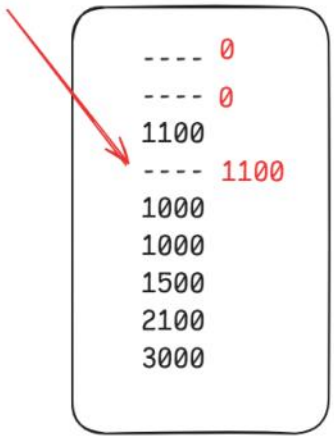
```
SELECT
    CustomerKey,
    FullName,
    Regions,
    IFNULL(Regions, 'No Region') AS Region,
    AnnualIncome,
    IFNULL(AnnualIncome , 'Not Available') AS IncomeCateogory
FROM Customers;
```

CustomerKey	FullName	Regions	Region	AnnualIncome	IncomeCateogory
11000	JON YANG	NULL	No Region	90000	90000
11001	EUGENE HUANG	NULL	No Region	60000	60000
11002	RUBEN TORRES	NULL	No Region	60000	60000
11003	CHRISTY ZHU	NULL	No Region	NULL	Not Available
11004	ELIZABETH JOHNSON	NULL	No Region	80000	80000
11005	JULIO RUIZ	NULL	No Region	70000	70000
11007	MARCO MEHTA	NULL	No Region	60000	60000
11008	ROBIN VERHOFF	NULL	No Region	60000	60000
11009	SHANNON CARLSON	NULL	No Region	70000	70000
11010	JACQUELYN SUAREZ	NULL	No Region	70000	70000
11011	CURTIS LU	NULL	No Region	60000	60000
11012	LAUREN WALKER	NULL	No Region	100000	100000
11013	IAN JENKINS	NULL	No Region	100000	100000
11014	SYDNEY BENNETT	NULL	No Region	100000	100000
11015	CHLOE YOUNG	NULL	No Region	NULL	Not Available

```
SELECT
    CustomerKey,
    FullName,
    IFNULL(Regions, 'No Region') AS Region,
    IFNULL(AnnualIncome , 'Not Available') AS IncomeCateogory
FROM Customers;
```

CustomerKey	FullName	Region	IncomeCateogory
11000	JON YANG	No Region	90000
11001	EUGENE HUANG	No Region	60000
11002	RUBEN TORRES	No Region	60000
11003	CHRISTY ZHU	No Region	Not Available
11004	ELIZABETH JOHNSON	No Region	80000
11005	JULIO RUIZ	No Region	70000
11007	MARCO MEHTA	No Region	60000
11008	ROBIN VERHOFF	No Region	60000
11009	SHANNON CARLSON	No Region	70000
11010	JACQUELYN SUAREZ	No Region	70000
11011	CURTIS LU	No Region	60000
11012	LAUREN WALKER	No Region	100000
11013	IAN JENKINS	No Region	100000
11014	SYDNEY BENNETT	No Region	100000
11015	CHLOE YOUNG	No Region	Not Available

```
-- COALESCE()
SELECT
    CustomerKey,
    FullName,
    COALESCE(AnnualIncome , 0) AS IncomeCateogory
FROM Customers;
```



COALESCE(AnnualIncome, PreviousAnnualIncome, 0)

CustomerKey	FullName	IncomeCateogory
11000	JON YANG	90000
11001	EUGENE HUANG	60000
11002	RUBEN TORRES	60000
11003	CHRISTY ZHU	0
11004	ELIZABETH JOHNSON	80000
11005	JULIO RUIZ	70000
11007	MARCO MEHTA	60000
11008	ROBIN VERHOFF	60000
11009	SHANNON CARLSON	70000
11010	JACQUELYN SUAREZ	70000
11011	CURTIS LU	60000
11012	LAUREN WALKER	100000
11013	IAN JENKINS	100000
11014	SYDNEY BENNETT	100000
11015	CHLOE YOUNG	0

Top 10 Products based on Total Revenue

```
WITH AllSales AS(
    SELECT * FROM Sales2015
    UNION
    SELECT * FROM Sales2016
    UNION
    SELECT * FROM Sales2017
) -- 56046 row(s) returned
SELECT
    p.ProductName,
    ROUND(SUM(s.OrderQuantity * p.ProductPrice),0) AS TotalRevenue
FROM AllSales s
JOIN Products p
ON p.ProductKey = s.ProductKey
GROUP BY 1
ORDER BY 2 DESC
LIMIT 10;
```

ProductName	TotalRevenue
Mountain-200 Black, 46	1241754
Mountain-200 Black, 42	1233557
Mountain-200 Silver, 38	1213852
Mountain-200 Silver, 46	1182781
Mountain-200 Black, 38	1165937
Mountain-200 Silver, 42	1133067
Road-250 Black, 52	689374
Road-250 Red, 58	661013
Road-250 Black, 48	641379
Road-150 Red, 48	640510

```
WITH AllSales AS(
    SELECT * FROM Sales2015
    UNION
    SELECT * FROM Sales2016
    UNION
    SELECT * FROM Sales2017
) -- 56046 row(s) returned
SELECT
    p.ProductName,
    ROUND(SUM(s.OrderQuantity * p.ProductPrice),0) AS TotalRevenue,
    ROUND(SUM(s.OrderQuantity * p.ProductCost),0) AS TotalExpenses
FROM AllSales s
JOIN Products p
ON p.ProductKey = s.ProductKey
GROUP BY 1
ORDER BY 2 DESC
LIMIT 10;
```

ProductName	TotalRevenue	TotalExpenses
Mountain-200 Black, 46	1241754	670121
Mountain-200 Black, 42	1233557	665698
Mountain-200 Silver, 38	1213852	655064
Mountain-200 Silver, 46	1182781	638296
Mountain-200 Black, 38	1165937	629206
Mountain-200 Silver, 42	1133067	611467
Road-250 Black, 52	689374	417336
Road-250 Red, 58	661013	400167
Road-250 Black, 48	641379	388281
Road-150 Red, 48	640510	388662