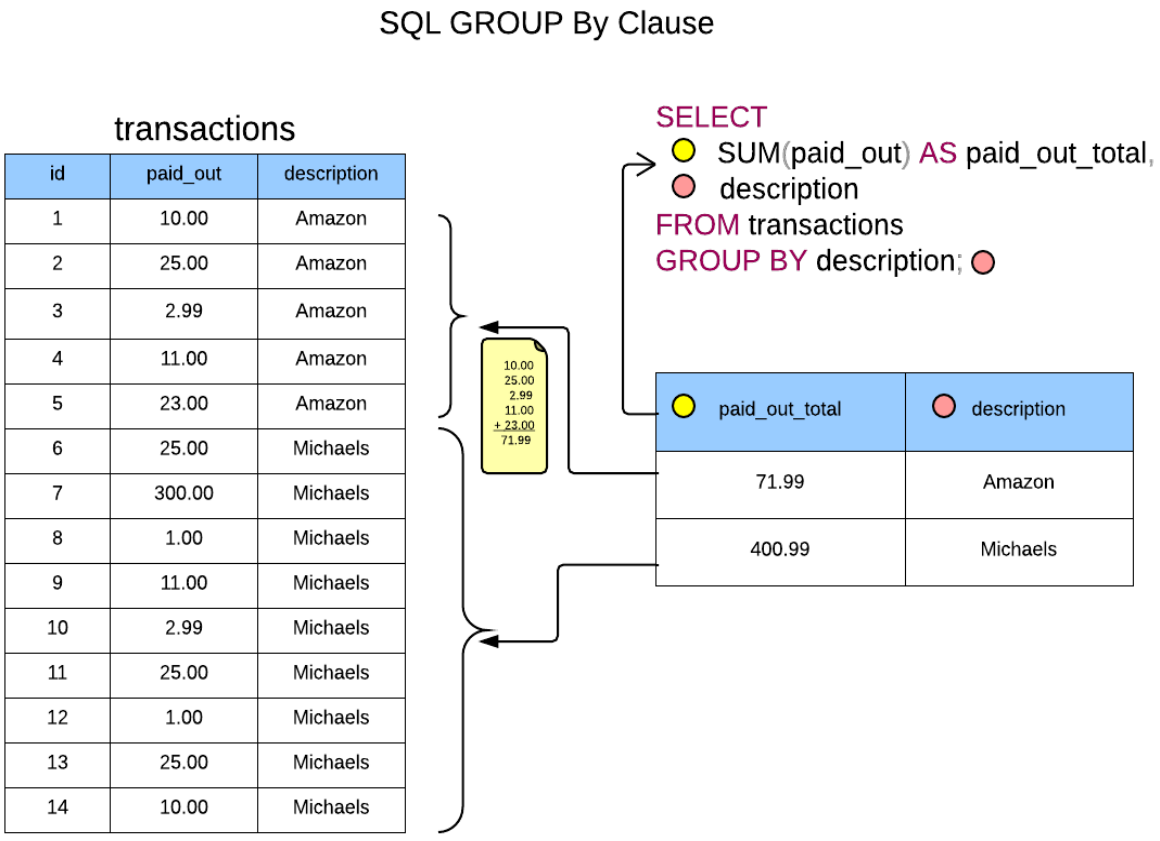
Learning Objectives

- Group By, Having, Order By
- JOINS
- SUBQUERIES
- VIEWS

Group By Clause: The SQL **GROUP BY** clause is used in collaboration with the SELECT statement to arrange identical data into groups. In most of the cases this GROUP BY clause follows the WHERE clause in a SELECT statement and precedes the ORDER BY clause.



id	paid_out	description
1	10.00	Amazon
2	25.00	Amazon
3	2.99	Amazon
4	11.00	Amazon
5	23.00	Amazon
6	25.00	Michaels
7	300.00	Michaels
8	1.00	Michaels
9	11.00	Michaels
10	2.99	Michaels
11	25.00	Michaels
12	1.00	Michaels
13	25.00	Michaels
14	10.00	Michaels

Example:

SELECT customer_id, sum(amount) as Total

Copyright by Xccelerate

```
FROM payment
GROUP BY customer_id
```

Order By Clause: Used to sort the output records.

From above example if needs to sort out the total value in descending order then we use Order By clause

Syntax:

```
SELECT sum(Paid_out) as paid_out_total, description
FROM transactions
GROUP BY description
ORDER BY paid_out_total
```

Example: Frequently visited customer

```
SELECT customer_id, sum(amount) as Total
FROM payment
GROUP BY customer_id
ORDER BY Total desc
```

Having Clause: To filter out the output result which are given by “group by” clause/aggregate functions.

Syntax:

```
SELECT sum(Paid_out) as paid_out_total, description
FROM transactions
GROUP BY description
Having sum(Paid_out) = 400.99
```

Result will be 1 row instead of 2 rows from group by clause example

Copyright by Xccelerate

** Having clause is similar to where clause but difference is where should be used before group by clause and having should be used to filter the result after group by clause.

Example: Customer details which has total amount more than 100

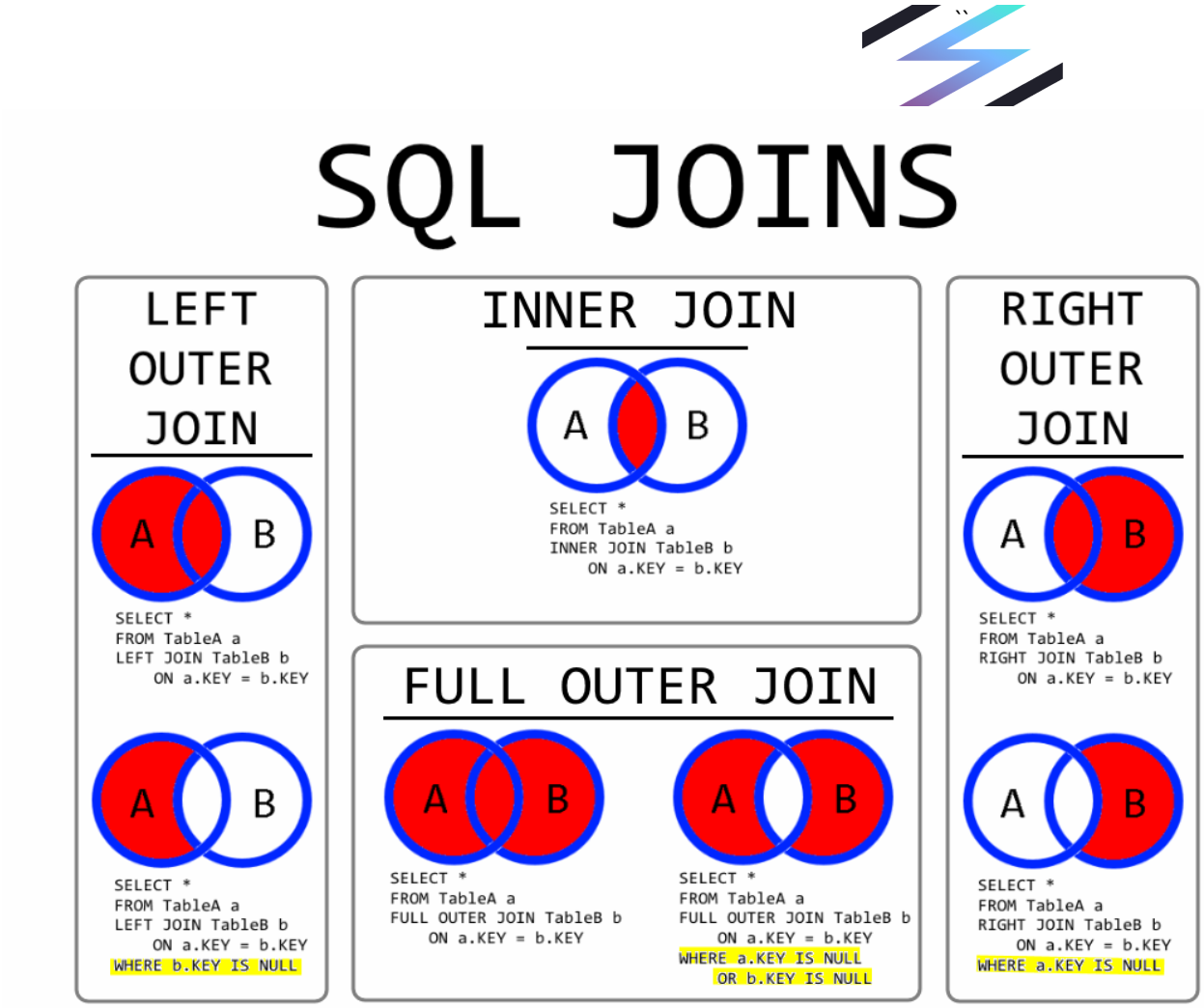
```
SELECT customer_id, sum(amount) as Total
FROM payment
GROUP BY customer_id
HAVING sum(amount) >=100
ORDER BY Total desc
```

We can not use below query as where can not apply on aggregate column

```
select customer_id, sum(amount) as Total from payment
where sum(amount) >=100
group by customer_id
order by Total desc
```

Joins: An SQL JOIN clause is used to combine rows from two or more tables, based on a common field between them.

Copyright by Xccelerate



Example:

```
SELECT *
FROM Products, Manufacturers
WHERE Products.Manufacturer = Manufacturers.Code;

SELECT * FROM Products INNER JOIN Manufacturers
ON Products.Manufacturer = Manufacturers.Code;
```

Sub Queries:

A sub query is a select query that is contained inside another query. The inner select query is usually used to determine the results of the outer select query.

Copyright by Xccelerate

For performance issues, when it comes to getting data from multiple tables, it is strongly recommended to use JOINs instead of sub queries. Sub queries should only be used with good reason.

Example:

```
SELECT Name, Price
FROM Products
WHERE Price = (SELECT MIN(Price) FROM Products);
```

Views: Views are virtual tables. Means the tables do not store any data of their own but display data stored in other tables.

Views improve security of the database by showing only intended data to authorized users. They hide sensitive data.

Syntax:

```
CREATE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

Example:

```
CREATE VIEW MINPRICEPRODUCT AS
SELECT Name, Price
FROM Products
WHERE Price = (SELECT MIN(Price) FROM Products);
```

How to query the view:

Copyright by Xccelerate

```
SELECT *
FROM MINPRICEPRODUCT
```

How to delete the view:

```
DROP VIEW MINPRICEPRODUCT
```

How to modify the existing view: This query also can used for creating new view

```
CREATE OR REPLACE VIEW MINPRICEPRODUCT AS
SELECT Name, Price
FROM Products
WHERE Price = (SELECT MIN(Price) FROM Products);
```

Copyright by Xccelerate