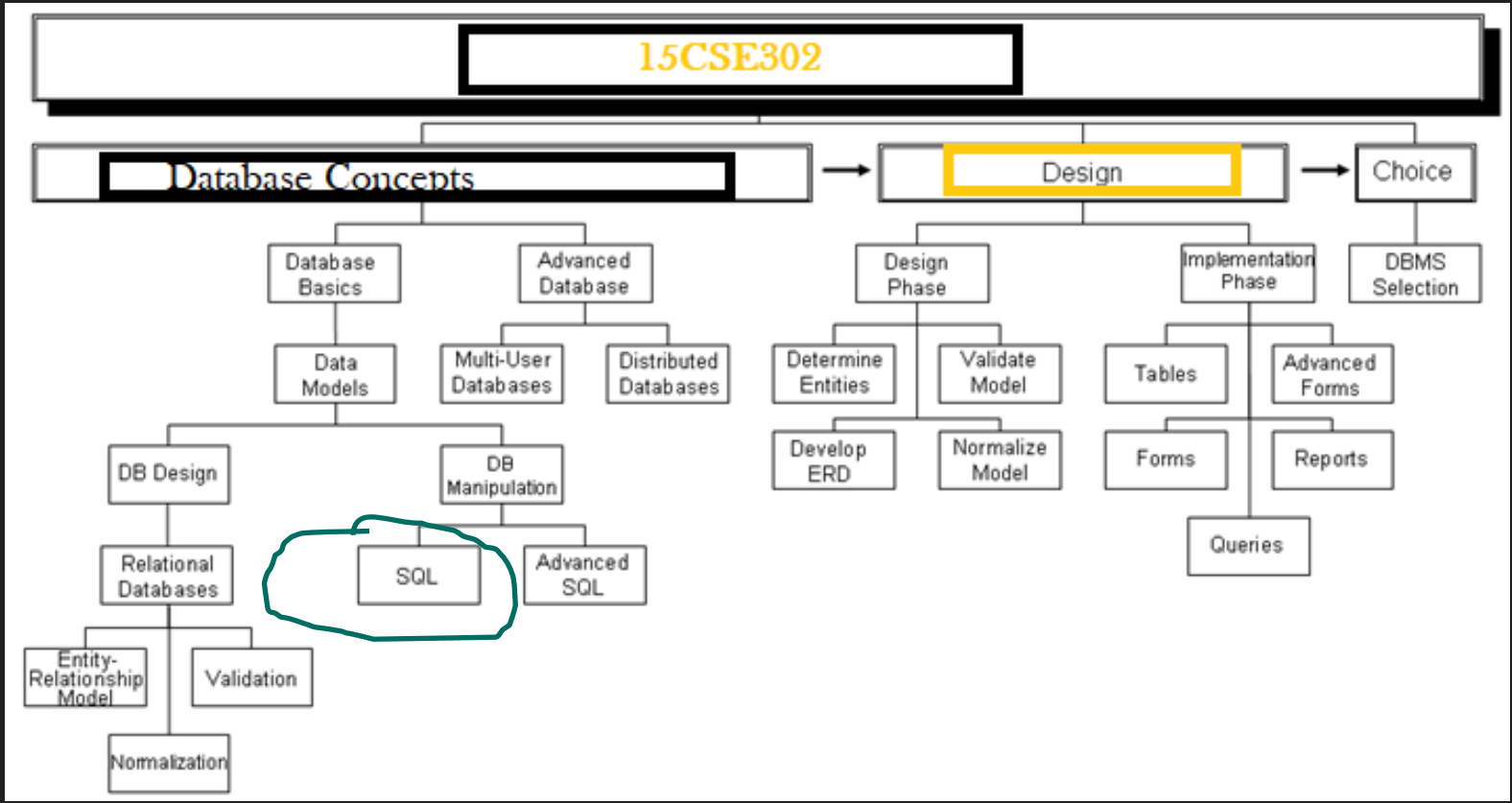
15CSE302 Database Management Systems

Lecture 8 SQL SET OPERATIONS and JOIN

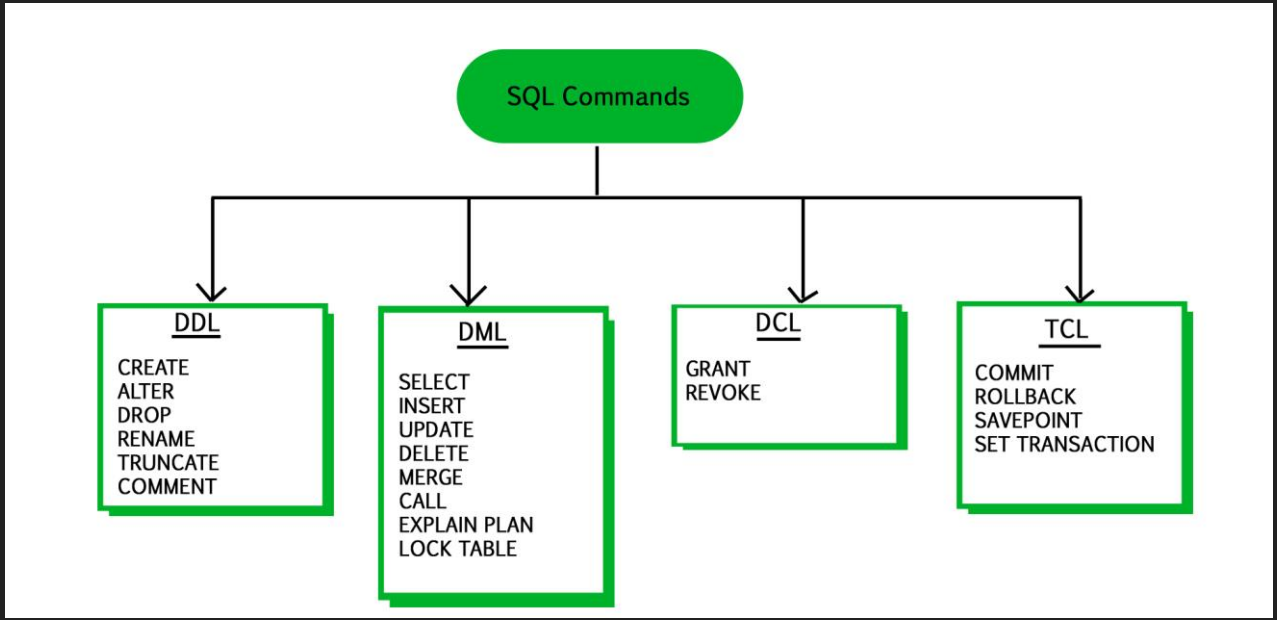
B.Tech /III Year CSE/V Semester L T P C 2 0 2 3

DBMS Team
Dr G Jeyakumar
Bindu K R
Dr Priyanka Kumar
R. Manjusha
Department of CSE
Amrita School of Engineering

Syllabus


15CSE302
Database Concepts
Design
Choice
Database Basics
Advanced Database
Design Phase
Implementation Phase
DBMS Selection
Data Models
Multi-User Databases
Distributed Databases
Determine Entities
Validate Model
Tables
Advanced Forms
DB Design
DB Manipulation
Develop ERD
Normalize Model
Forms
Reports
Queries
Relational Databases
SQL
Advanced SQL
Entity-Relationship Model
Validation
Normalization

SQL Structured Query Language



Brief Recap of Previous Lecture

- Order by
- Functions
  - Numerical Functions
    - Single value
    - Group value
    - List value
- Join

Today's lecture

- Aggregation -More examples
- JOIN –Types of Join
- SET Operations
  - Union
  - Intersection
  - Difference

Aggregation

SQL supports several aggregation operations:

SUM, MIN, MAX, AVG, COUNT

Find the average price for products from Toyota.

```
SELECT  Avg(price)
FROM    Product
WHERE   maker='Toyota'
```

Aggregation: Count

Find the number of products produced after 2010

```
SELECT  Count(*)
FROM    Product
WHERE   year > 2010
```

Except COUNT, all aggregations apply to a single attribute

Aggregation: Count

COUNT applies to duplicates, unless otherwise stated:

Find the number of products produced after 2010

```
SELECT  Category, Count(category)      /*  same as Count(*) */
FROM    Product
WHERE   year > 1995
```

Better:

```
SELECT  CATEGORY, Count(DISTINCT category)
FROM    Product
WHERE   year > 1995
```

Simple Aggregations

Purchase

Product	Date	Price	Quantity
Bagel	10/21	0.85	15
Banana	10/22	0.52	7
Banana	10/19	0.52	17
Bagel	10/20	0.85	20

Simple Aggregation

Purchase(product, date, price, quantity)

Example 1: Find total sales for the entire database

```
SELECT  Sum(price * quantity)
FROM    Purchase
```

Example 1': Find total sales of bagels

```
SELECT  Sum(price * quantity)
FROM    Purchase
WHERE   product = 'bagel'
```

Grouping and Aggregation

Usually, we want aggregations on certain parts of the relation.

Purchase(product, date, price, quantity)

Example 2: **Find total sales after 9/1 per product.**

```
SELECT    product, Sum(price*quantity) AS TotalSales
FROM      Purchase
WHERE     date > "9/1"
GROUP BY  product
```

Let's see what this means…

Grouping and Aggregation

1. Compute the FROM and WHERE clauses.
2. Group by the attributes in the GROUP BY
3. Select one tuple for every group (and apply aggregation)

SELECT can have (1) grouped attributes or (2) aggregates.

First compute the FROM-WHERE clauses (date > “9/1”) then GROUP BY product:

Product	Date	Price	Quantity
Bagel	10/21	0.85	15
Banana	10/22	0.52	7
Banana	10/19	0.52	17
Bagel	10/20	0.85	20

Then, aggregate

```
SELECT    product, Sum(price*quantity) AS TotalSales
FROM      Purchase
WHERE     date > "9/1"
GROUP BY  product
```

Another Example

For every product, what is the total sales and max quantity sold?

```
SELECT      product, Sum(price * quantity) AS SumSales
                     Max(quantity) AS MaxQuantity
FROM        Purchase
GROUP BY    product
```

HAVING Clause

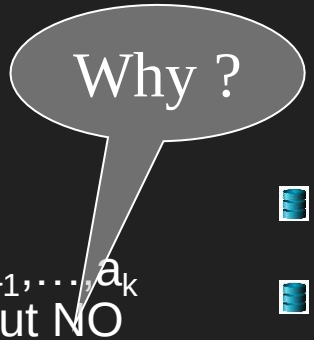
Same query, except that we consider only products that had at least 100 buyers.

```
SELECT    product, Sum(price * quantity)
FROM      Purchase
WHERE     date > "9/1"
GROUP BY  product
HAVING    Sum(quantity) > 30
```

HAVING clause contains conditions on aggregates.

General form of Grouping and Aggregation

SELECT S
FROM $R_1,\ldots,R_n$
WHERE C1
GROUP BY $a_1,\ldots,a_k$
HAVING C2



S = may contain attributes $a_1,\ldots,a_k$ and/or any aggregates but NO OTHER ATTRIBUTES
C1 = is any condition on the attributes in $R_1,\ldots,R_n$
C2 = is any condition on aggregate expressions

Evaluation steps:

- Compute the FROM-WHERE part, obtain a table with all attributes in $R_1,\ldots,R_n$
- Group by the attributes $a_1,\ldots,a_k$
- Compute the aggregates in C2 and keep only groups satisfying C2
- Compute aggregates in S and return the result

Aggregation -Example

Author(login,name)
Document(url, title)
Wrote(login,url)
Mentions(url,word)

- Find all authors who wrote at least 10 documents:

```
SELECT    Author.name
FROM      Author, Wrote
WHERE     Author.login=Wrote.login
GROUP BY  Author.name
HAVING    count(wrote.url) > 10
```

No need for DISTINCT: automatically from GROUP BY

Example

- Find all authors who have a vocabulary over 10000 words:

```
SELECT      Author.name
FROM        Author, Wrote, Mentions
WHERE       Author.login=Wrote.login AND Wrote.url=Mentions.url
GROUP BY    Author.name
HAVING      count(distinct Mentions.word) > 10000
```

SET Operations - Union, Intersection, Difference

Find the name of buyers who are from Seattle or buys from a particular store The Bon

```
(SELECT  name
 FROM    Person
 WHERE   City='Seattle')

   UNION

(SELECT  name
 FROM    Person, Purchase
 WHERE   buyer=name AND store='The Bon')
```

Similarly, you can use INTERSECT and EXCEPT. You must have the same attribute names (otherwise: rename).

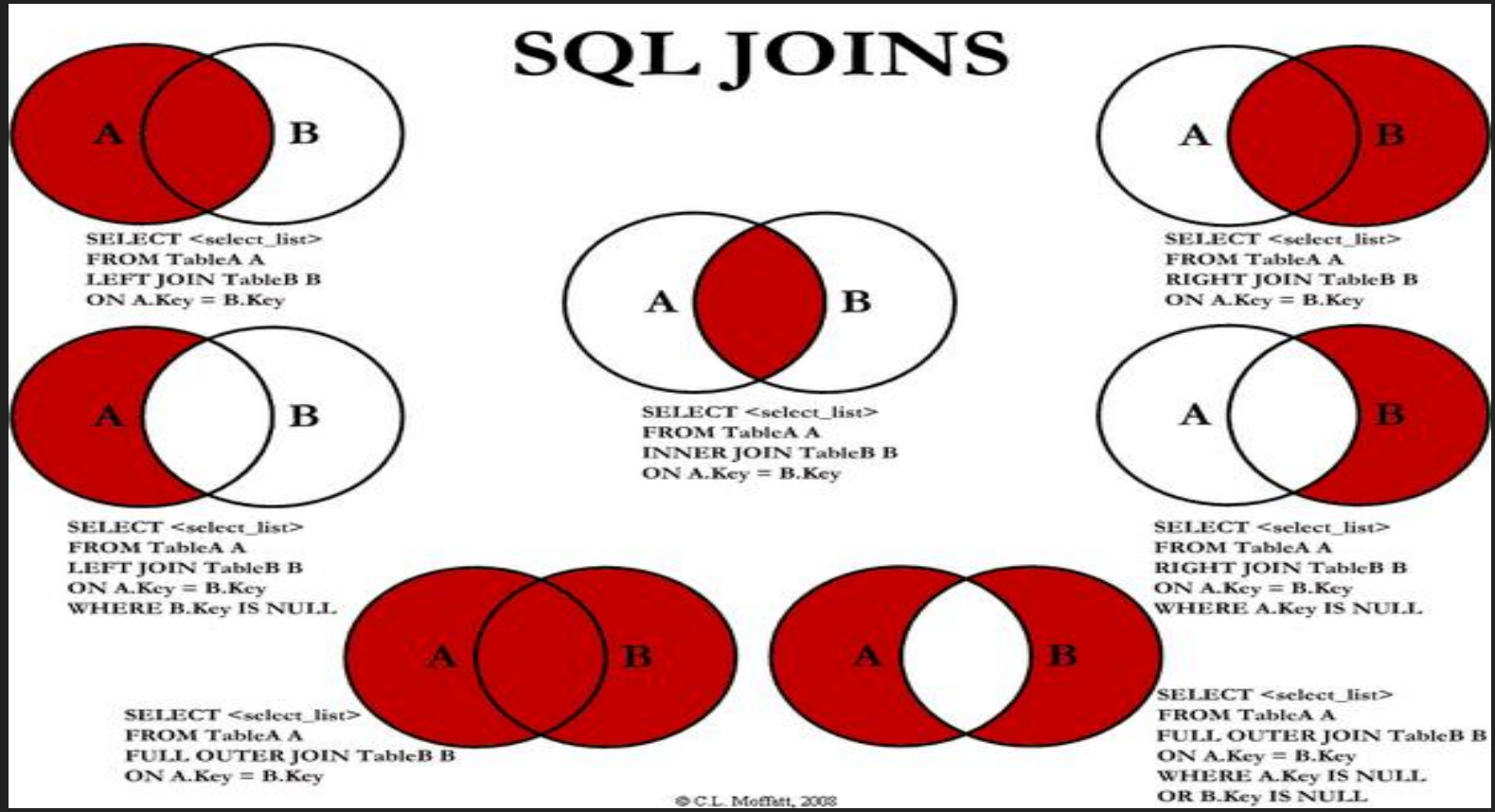
Conserving Duplicates

```
(SELECT  name
 FROM      Person
 WHERE    City="Seattle")

UNION  ALL

(SELECT  name
 FROM      Person, Purchase
 WHERE   buyer=name AND store='The Bon')
```

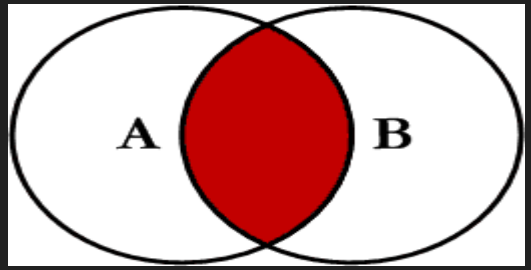
Types of Join

- INNER JOIN
- LEFT JOIN
- RIGHT JOIN
- OUTER JOIN
- LEFT JOIN EXCLUDING INNER JOIN
- RIGHT JOIN EXCLUDING INNER JOIN
- OUTER JOIN EXCLUDING INNER JOIN

Types of Join



Types of Join - Inner Join




```
SELECT <select_list>
FROM
Table_A A INNER JOIN Table_B B
ON A.Key = B.Key
```

INNER JOIN

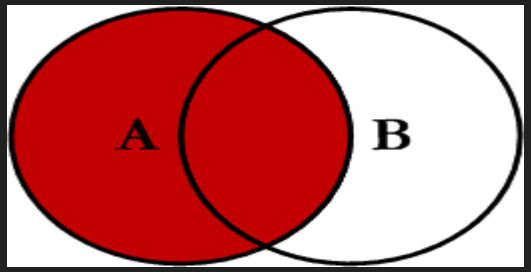
Table A	
PK	Value
1,	FOX
2	COP
3	TAXI
6	WASHINGTON
7	DELL
5	ARIZONA
4	LINCOLN
10	LUCENT

Table B	
PK	Value
1	TROT
2	CAR
3	CAB
6	MONUMENT
7	PC
8	MICROSOFT
9	APPLE
11	SCOTCH

INNER JOIN			
A_PK	A_Value	B_PK	B_Value
1	FOX	TROT	1
2	COP	CAR	2
3	TAXI	CAB	3
6	WASHINGTON	MONUMENT	6
7	DELL	PC	7

```
SELECT A.PK AS A_PK, A.Value AS
A_Value,
      B.Value AS B_Value, B.PK AS B_PK
FROM Table_A A INNER JOIN Table_B B
ON A.PK = B.PK
```

Types of Join - Left Join




```
SELECT <select_list>
FROM
Table_A A LEFT JOIN Table_B B
ON A.Key = B.Key
```

LEFT JOIN

Table A

PK	Value
1	FOX
2	COP
3	TAXI
6	WASHINGTON
7	DELL
5	ARIZONA
4	LINCOLN
10	LUCENT

Table B

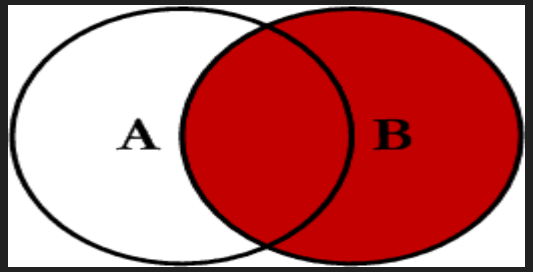
PK	Value
1	TROT
2	CAR
3	CAB
6	MONUMENT
7	PC
8	MICROSOFT
9	APPLE
11	SCOTCH

LEFT JOIN

A_PK	A_Value	B_PK	B_Value
1	FOX	TROT	1
2	COP	CAR	2
3	TAXI	CAB	3
6	WASHINGTON	MONUMENT	6
7	DELL	PC	7
5	ARIZONA	NULL	NULL
4	LINCOLN	NULL	NULL
10	LUCENT	NULL	NULL

```
SELECT A.PK AS A_PK, A.Value AS
A_Value,
      B.Value AS B_Value, B.PK AS B_PK
FROM Table_A A LEFT JOIN Table_B B
ON A.PK = B.PK
```

Types of Join - Right Join




```
SELECT <select_list>
FROM
Table_A A RIGHT JOIN Table_B B
ON A.Key = B.Key
```

RIGHT JOIN

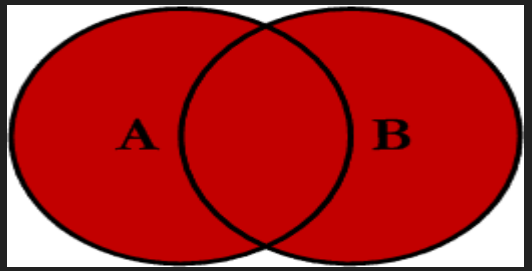
Table A	
PK	Value
1	FOX
2	COP
3	TAXI
6	WASHINGTON
7	DELL
5	ARIZONA
4	LINCOLN
10	LUCENT

Table B	
PK	Value
1	TROT
2	CAR
3	CAB
6	MONUMENT
7	PC
8	MICROSOFT
9	APPLE
11	SCOTCH

RIGHT JOIN			
A_PK	A_Value	B_PK	B_Value
1	FOX	TROT	1
2	COP	CAR	2
3	TAXI	CAB	3
6	WASHINGTON	MONUMENT	6
7	DELL	PC	7
NULL	NULL	MICROSOFT	8
NULL	NULL	APPLE	9
NULL	NULL	SCOTCH	11

```
SELECT A.PK AS A_PK, A.Value AS
A_Value,
      B.Value AS B_Value, B.PK AS B_PK
FROM Table_A A RIGHT JOIN Table_B B
ON A.PK = B.PK
```

Types of Join - FULL OUTER JOIN




```
SELECT <select_list>
FROM
Table_A A FULL OUTER JOIN Table_B B
ON A.Key = B.Key
```

FULL OUTER JOIN

Table A

PK	Value
1	FOX
2	COP
3	TAXI
6	WASHINGTON
7	DELL
5	ARIZONA
4	LINCOLN
10	LUCENT

Table B

PK	Value
1	TROT
2	CAR
3	CAB
6	MONUMENT
7	PC
8	MICROSOFT
9	APPLE
11	SCOTCH

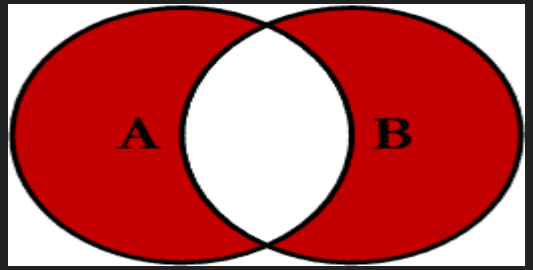
FULL OUTER JOIN

A_PK	A_Value	B_PK	B_Value
1	FOX	TROT	1
2	COP	CAR	2
3	TAXI	CAB	3
6	WASHINGTON	MONUMENT	6
7	DELL	PC	7
5	ARIZONA	NULL	NULL
4	LINCOLN	NULL	NULL
10	LUCENT	NULL	NULL
NULL	NULL	8	MICROSOFT
NULL	NULL	9	APPLE
NULL	NULL	11	SCOTCH

```
SELECT A.PK AS A_PK, A.Value AS A_Value,
       B.Value AS B_Value, B.PK AS B_PK
FROM Table_A A FULL OUTER JOIN Table_B B
ON A.PK = B.PK
```

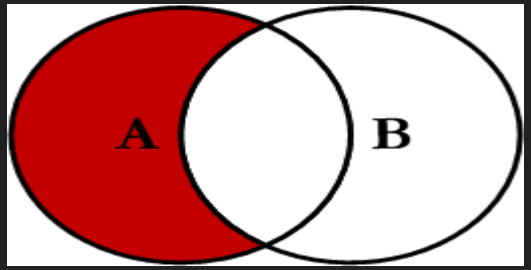
(11 row(s) affected)

Types of Join - Outer Excluding JOIN



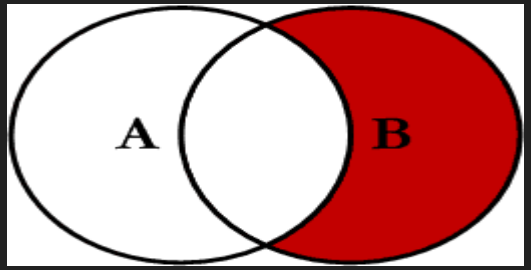

```
SELECT <select_list>
FROM
Table_A A FULL OUTER JOIN Table_B B
ON A.Key = B.Key
where A.Key is NULL OR B.Key is NULL
```

Types of Join - LEFT Excluding JOIN




```
SELECT <select_list>
FROM
Table_A A LEFT JOIN Table_B B
ON A.Key = B.Key
where A.Key is NULL
```

Types of Join - RIGHT Excluding JOIN




```
SELECT <select_list>
FROM
Table_A A RIGHT JOIN Table_B B
ON A.Key = B.Key
where B.Key is NULL
```

References

- https://docs.oracle.com/en/database/oracle/oracle-database/20/newft/new-features.html
- https://www.pda.org/scientific-and-regulatory-affairs/regulatory-resources/data-integrity
- https://www.digipay.guru/blog/all-you-need-to-know-about-agency-banking/
- https://md.ekstrandom.net/teaching/cs4332-f15.pdf
- https://https://bit.ly/31eE2Ar
- https://ipronline.com/oracle-the-pioneers-of-the-software-world/

Summary

- Aggregation -More examples
- SET Operations
  - Union
  - Intersection
  - Difference
- JOIN –Types of Join

Next Lecture

- Sub queries

Thank You

Happy to answer any questions ! ! !