PERTEMUAN 7
SELECT JOIN

A. INNER JOIN

Inner join pada dasarnya adalah menemukan persimpangan (intersection) antara dua buah tabel.

Sintaks *inner join* diperlihatkan sebagai berikut:

```
SELECT A1, A2, ..., An
FROM r1
  INNER JOIN r2
    ON r1.join_key = r2.join_key
```

Cara 1

SELECT A1, A2, ..., An
FROM r1
INNER JOIN r2
ON r1.join_key = r2.join_key

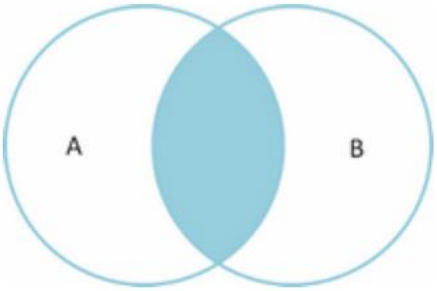
Inner join juga dapat direpresentasikan dalam bentuk implisit.

```
SELECT A1, A2, ..., An
FROM r1, r2
WHERE r1.key = r2.key
```

Cara 2

SELECT A1, A2, ..., An
FROM r1, r2
WHERE r1.key = r2.key

Misalkan terdapat tabel A dan B, maka hasil *inner join* dapat diperlihatkan—sebagai bidang terarsir—dalam diagram Venn seperti Gambar 1.



Gambar 1. Inner Join

B. OUTER JOIN

- **Left Outer Join**

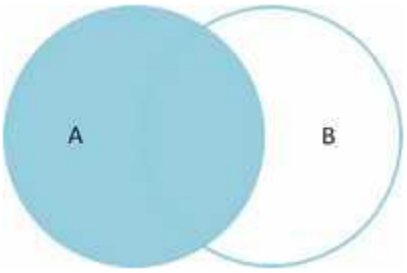
Left outer join (atau left join) mengembalikan semua nilai dari tabel kiri ditambah dengan nilai dari tabel kanan yang sesuai (atau NULL jika tidak ada nilai yang sesuai).

```
SELECT A1, A2, ..., An
FROM r1
  LEFT OUTER JOIN r2
    ON r1.join_key = r2.join_key
```

Syntax

SELECT A1, A2, ..., An
FROM r1
LEFT OUTER JOIN r2
ON r1.join_key = r2.join_key

Left outer join antara tabel A dan B dapat diilustrasikan dalam diagram Venn seperti Gambar 2.



Gambar 2. Left Outer Join

- **Right Outer Join**

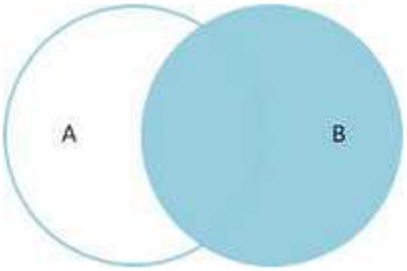
Right outer join (atau right join) pada dasarnya sama seperti left join, namun dalam bentuk terbalik—kanan dan kiri. Sintaks *right outer join* diperlihatkan sebagai berikut:

```
SELECT A1, A2, ..., An
FROM r1
  RIGHT OUTER JOIN r2
    ON r1.join_key = r2.join_key
```

Syntax

SELECT A1, A2, ..., An
FROM r1
RIGHT OUTER JOIN r2
ON r1.join_key = r2.join_key

Right outer join antara tabel A dan B dapat diilustrasikan dalam diagram Venn seperti Gambar 3.



Gambar 3. Right Outer Join

- **Full Outer Join**

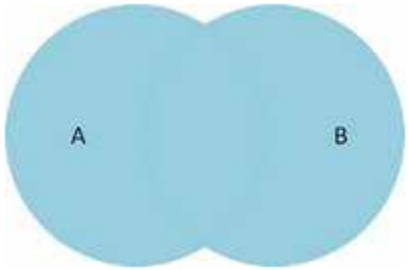
Full outer join (atau full join) pada hakekatnya merupakan kombinasi dari left dan right join. Sintaks *full outer join* diperlihatkan sebagai berikut:

```
SELECT A1, A2, ..., An
FROM r1
  FULL OUTER JOIN r2
    ON r1.join_key = r2.join_key
```

Syntax

SELECT A1, A2, ..., An
FROM r1
FULL OUTER JOIN r2
ON r1.join_key = r2.join_key

Bentuk visual dari *full outer join* dapat diperlihatkan menggunakan diagram Venn seperti Gambar 4.



Gambar 4. Full Outer Join

CONTOH 1

```
CREATE TABLE t1 (
    id INT PRIMARY KEY,
    pattern VARCHAR(50) NOT NULL
);

CREATE TABLE t2 (
    id VARCHAR(50) PRIMARY KEY,
    pattern VARCHAR(50) NOT NULL
);
INSERT INTO t1(id, pattern)
VALUES(1,'Divot'),
      (2,'Brick'),
      (3,'Grid');

INSERT INTO t2(id, pattern)
VALUES('A','Brick'),
      ('B','Grid'),
      ('C','Diamond');

SELECT * FROM T1;
SELECT * FROM T2;
```

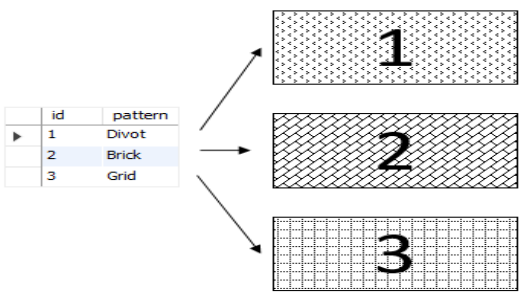
T1

id	pattern
1	Divot
2	Brick
3	Grid

T2

id	pattern
A	Brick
B	Grid
C	Diamond

Jika diilustrasikan akan menjadi seperti tampilan di bawah ini



t1

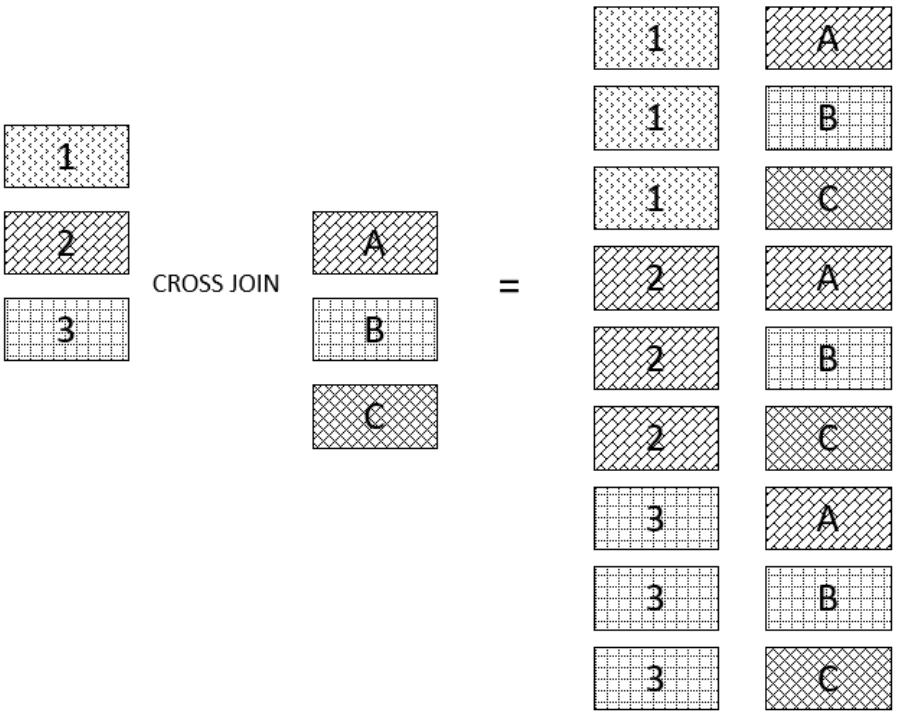
CROSS JOIN

```
--CROSS JOIN
SELECT
    t1.id, t2.id
FROM
    t1
CROSS JOIN t2;
```

Hasil

id	id
1	A
2	A
3	A
1	B
2	B
3	B
1	C
2	C
3	C

Ilustrasi



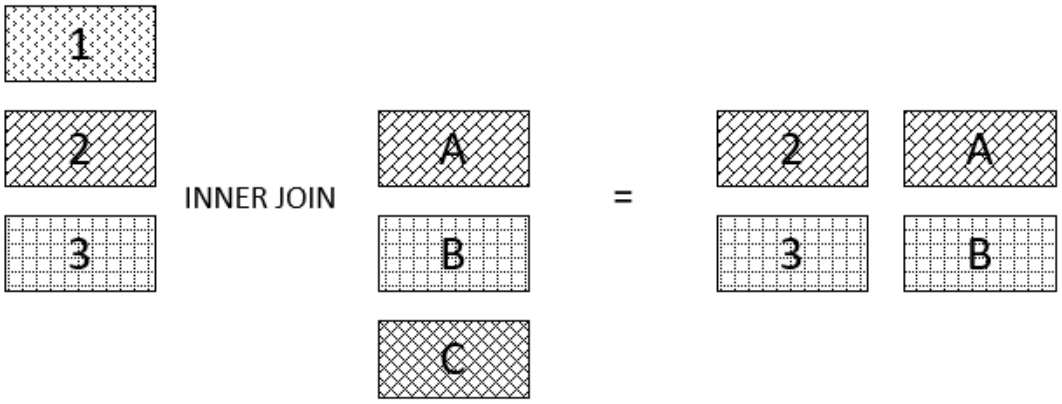
INNER JOIN

```
SELECT
    t1.id, t2.id
FROM
    t1
        INNER JOIN
    t2 ON t1.pattern = t2.pattern;
```

Hasil

id	id
2	A
3	B

Ilustrasi

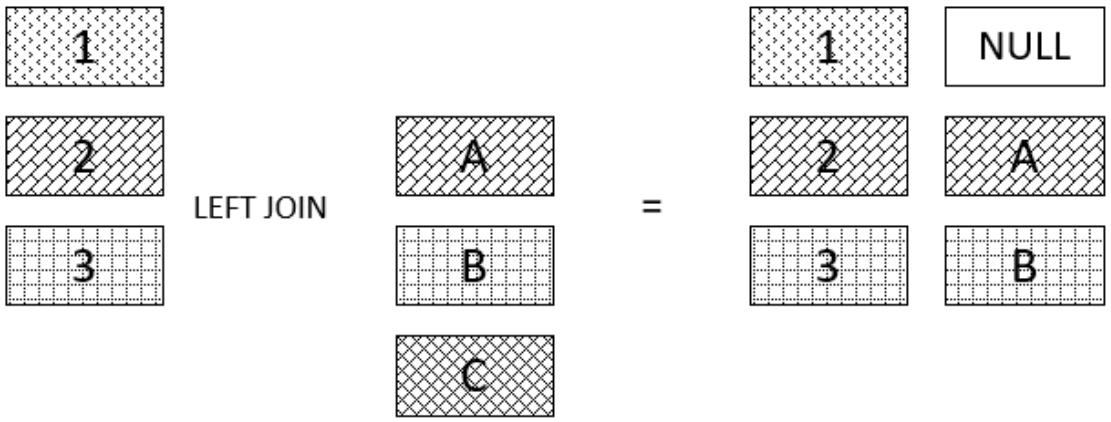


LEFT JOIN

```
--left join
SELECT
    t1.id, t2.id
FROM
    t1
        LEFT JOIN
    t2 ON t1.pattern = t2.pattern
ORDER BY t1.id;
```

id	id
1	*<NULL>*
2	A
3	B

Ilustrasi



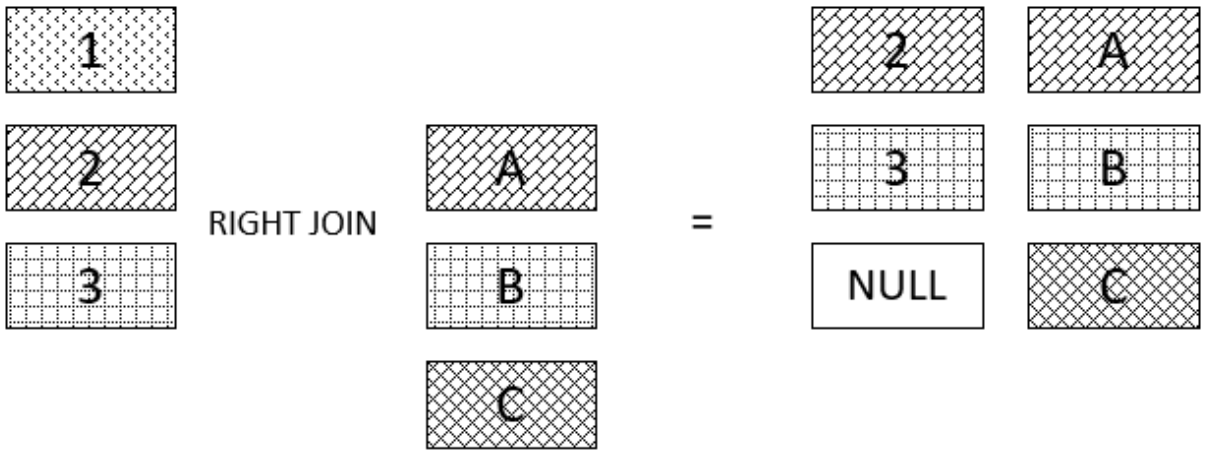
RIGHT JOIN

```
--RIGHT JOIN
SELECT
    t1.id, t2.id
FROM
    t1
        RIGHT JOIN
    t2 on t1.pattern = t2.pattern
ORDER BY t2.id;
```

Hasil

id	id
2	A
3	B
<NULL>	C

Ilustrasi



CONTOH 2

```
CREATE TABLE karyawan (
nama varchar(30) NOT NULL,
id_dep int(5) NOT NULL
) ;

CREATE TABLE departemen (
id_dep int(5) NOT NULL,
nama_dep varchar(30) NOT NULL,
PRIMARY KEY (id_dep)
) ;
```

```
INSERT INTO `departemen` (`id_dep`, `nama_dep`) VALUES
('10', 'PENELITIAN'),
('11', 'PEMASARAN'),
('12', 'SDM'),
('13', 'KEUANGAN') ;

INSERT INTO `karyawan` (`nama`, `id_dep`) VALUES
('AGUS', '10'),
('BUDI', '16'),
('CITRA', '12'),
('DANI', '17');

SELECT * FROM KARYAWAN;
SELECT * FROM DEPARTEMEN;
```

TABEL KARYAWAN

nama	id_dep
AGUS	10
BUDI	16
CITRA	12
DANI	17

TABEL DEPARTEMEN

id_dep	nama_dep
10	PENELITIAN
11	PEMASARAN
12	SDM
13	KEUANGAN

INNER JOIN

```
--1 INNER JOIN
SELECT * FROM KARYAWAN
INNER JOIN DEPARTEMEN
ON KARYAWAN.id_dep = DEPARTEMEN.id_dep;

SELECT * FROM KARYAWAN, DEPARTEMEN
WHERE KARYAWAN.id_dep = DEPARTEMEN.id_dep;
```

nama	id_dep	id_dep	nama_dep
AGUS	10	10	PENELITIAN
CITRA	12	12	SDM

LEFT OUTER JOIN

```
-- 2a. LEFT OUTER JOIN
SELECT * FROM KARYAWAN K
LEFT OUTER JOIN DEPARTEMEN D
ON K.id_dep = D.id_dep;

SELECT * FROM KARYAWAN K
LEFT OUTER JOIN DEPARTEMEN D
ON K.id_dep = D.id_dep
WHERE D.id_dep
IS NULL;
```

Max. rows: 100 Fetched Rows: 4 Matching Rows:

#	nama	id_dep	id_dep	nama_dep
1	AGUS	10	10	PENELITIAN
2	BUDI	16	*<NULL>*	*<NULL>*
3	CITRA	12	12	SDM
4	DANI	17	*<NULL>*	*<NULL>*

#	nama	id_dep	id_dep	nama_dep
1	BUDI	16	*<NULL>*	*<NULL>*
2	DANI	17	*<NULL>*	*<NULL>*

RIGHT OUTER JOIN

```
--2b. RIGHT OUTER JOIN
SELECT * FROM KARYAWAN K
RIGHT OUTER JOIN DEPARTEMEN D
ON K.id_dep = D.id_dep;
```

nama	id_dep	id_dep	nama_dep
AGUS	10	10	PENELITIAN
CITRA	12	12	SDM
<NULL>	<NULL>	11	PEMASARAN
<NULL>	<NULL>	13	KEUANGAN

FULL OUTER JOIN

```
--2c. FULL OUTER JOIN
SELECT * FROM KARYAWAN K LEFT OUTER JOIN DEPARTEMEN D ON K.id_dep = D.id_dep
UNION
SELECT * FROM KARYAWAN K RIGHT OUTER JOIN DEPARTEMEN D ON K.id_dep = D.id_dep;
```

nama	id_dep	id_dep	nama_dep
AGUS	10	10	PENELITIAN
CITRA	12	12	SDM
<NULL>	<NULL>	11	PEMASARAN
<NULL>	<NULL>	13	KEUANGAN

CROSS JOIN

```
--2d. CROSS JOIN
SELECT * FROM KARYAWAN
CROSS JOIN DEPARTEMEN;
```

nama	id_dep	id_dep	nama_dep
AGUS	10	10	PENELITIAN
BUDI	16	10	PENELITIAN
CITRA	12	10	PENELITIAN
DANI	17	10	PENELITIAN
AGUS	10	11	PEMASARAN
BUDI	16	11	PEMASARAN
CITRA	12	11	PEMASARAN
DANI	17	11	PEMASARAN
AGUS	10	12	SDM
BUDI	16	12	SDM
CITRA	12	12	SDM
DANI	17	12	SDM
AGUS	10	13	KEUANGAN
BUDI	16	13	KEUANGAN
CITRA	12	13	KEUANGAN
DANI	17	13	KEUANGAN

Perintah UNION dalam MySQL di gunakan untuk Menggabungkan/mengkombinasikan isi dari dua tabel menjadi satu. Sama seperti artinya INTERSECT ini di gunakan untuk mencari irisan pada dua atau lebih tabel. Kemudian perintah yang terakhir yaitu EXCEPT, EXCEPT ini di vgunakan untuk memunculkan isi tabel yang berada di luar irisan tabel. Berikut query dasar yang bisa di gunakan untuk menggunakan perintah tersebut.

UNION

SELECT * FROM namatabel1 UNION SELECT * FROM namatabel2 ;

```
--UNION
SELECT * FROM t1
UNION
SELECT * FROM t2 ;
```

id	pattern
1	Divot
2	Brick
3	Grid
A	Brick
B	Grid
C	Diamond

INTERSECT

Pada MySQL penggunaan operasi intersect dapat diterapkan dengan beberapa cara. Misalnya dengan menggunakan operasi INNER JOIN, SELECT IN, UNION, maupun EXIST. Contoh penerapan alternative penggunaan operasi intersect pada MySQL adalah sebagai berikut

```
CREATE TABLE `table_a` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `value` varchar(255),
  PRIMARY KEY (`id`)
) ;

CREATE TABLE `table_b` LIKE `table_a`;

INSERT INTO table_a VALUES (1, 'A'), (2, 'B'), (3, 'B');
INSERT INTO table_b VALUES (1, 'B');

SELECT * FROM table_a;
SELECT * FROM table_b;
```

Tabel A

id	value
1	A
2	B
3	B

Tabel B

id	value
1	B

```
---menggunakan inner join
SELECT value FROM table_a
INNER JOIN table_b
USING (value);

-- menggunakan SELECT IN
SELECT value FROM table_a
WHERE (value) IN
(SELECT value FROM table_b);

-- akan menghasilkan data terduplikasi sehingga gunakan DISTINCT
```

Hasil eksekusi SQL diatas

value
B
B

Hasil seleksi diatas akan menghasilkan data terduplikasi sehingga kita perlu menambahkan operator DISTINCT

```
SELECT DISTINCT value FROM table_a
INNER JOIN table_b
USING (value);

SELECT DISTINCT value FROM table_a
WHERE (value) IN
(SELECT value FROM table_b);
```

Hasil eksekusi SQL diatas

value
B

Alternatif lain bias menggunakan operator union dan exist

```
-- menggunakan UNION
SELECT t1.value from (
  (SELECT DISTINCT value FROM table_a)
  UNION ALL
  (SELECT DISTINCT value FROM table_b)
) AS t1 GROUP BY value HAVING count(*) >= 2;


-- menggunakan operator exist
SELECT DISTINCT t1.value
FROM table_a t1
WHERE EXISTS (
  SELECT t2.value
  FROM  table_b t2
  WHERE t1.`value`=t2.`value`
)
```

Untuk penggunaan operator INTERSECT dapat diterapkan di SQL Server yang nantinya akan dipelajari pada Matakuliah Basis Data selanjutnya.
Anda dapat membuka laman web sebagai berikut untuk mengenal secara lebih mendalam lagi mengenai penerapan operator INTERSECT pada SQL Server
https://blog.sqlauthority.com/2008/08/03/sql-server-2005-difference-between-intersect-and-inner-join-intersect-vs-inner-join/

TUGAS KELOMPOK TEORI

1. **Silahkan Diskusi Kelompok dengan topik materi yang diberikan**
2. **Silahkan terapkan operasi INNER JOIN, OUTER JOIN (LEFT,RIGHT, FULL), GROUP BY, HAVING, UNION DAN INTERSECTION. pada database tugas kelompok besar Anda. (1 Contoh untuk setiap perintah yang diminta)**

   Tuliskan SQL Shyntax nya dan capture hasil eksekusi (Format seperti laporan tugas kelompok Anda sebelumnya). Jawaban dikumpulkan di akhir perkuliahan pada thread yang telah disediakan di Edmodo.