SQL JOIN TYPES SIMPLIFIED

CODING BUGS NOTES GALLERY

CREATE BY - ATUL KUMAR (LINKEDIN)


```sql
SELECT *
  FROM Table1;

SELECT *
  FROM Table2;
```

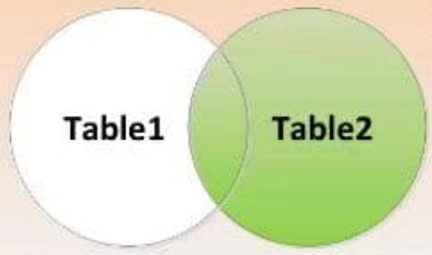
SELECT from two tables


```sql
SELECT *
  FROM Table1 t1
 INNER JOIN Table2 t2
    ON t1.fk = t2.id;
```

INNER JOIN


```sql
SELECT *
  FROM Table1 t1
 LEFT OUTER JOIN Table2 t2
    ON t1.fk = t2.id;
```

LEFT OUTER JOIN




```sql
SELECT *
  FROM Table1 t1
RIGHT OUTER JOIN Table2 t2
    ON t1.fk = t2.id;
```

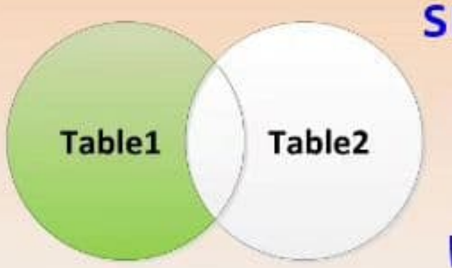
RIGHT OUTER JOIN


```sql
SELECT *
  FROM Table1 t1
 WHERE EXISTS (SELECT 1
                 FROM Table2 t2
                WHERE t1.fk = t2.id
               );
```

SEMI JOIN

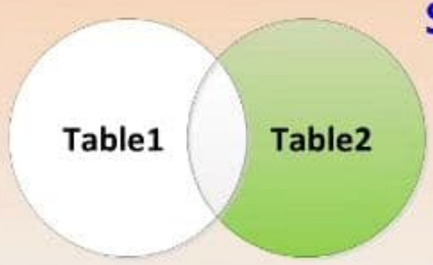

```sql
SELECT *
  FROM Table1 t1
 WHERE NOT EXISTS (SELECT 1
                     FROM Table2 t2
                    WHERE t1.fk = t2.id
                   );
```

ANTI SEMI JOIN



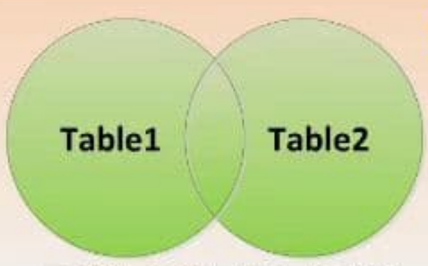

```sql
SELECT *
  FROM Table1 t1
  LEFT OUTER JOIN Table2 t2
    ON t1.fk = t2.id
 WHERE t2.id IS NULL;
```

LEFT OUTER JOIN with exclusion
– replacement for a NOT IN



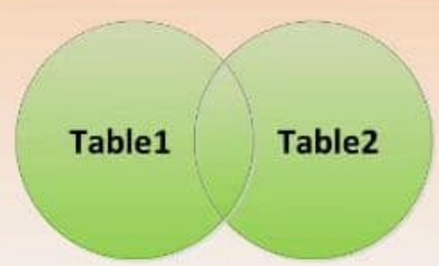

```sql
SELECT *
  FROM Table1 t1
  RIGHT OUTER JOIN Table2 t2
    ON t1.fk = t2.id
 WHERE t1.fk IS NULL;
```

RIGHT OUTER JOIN with exclusion
– replacement for a NOT IN



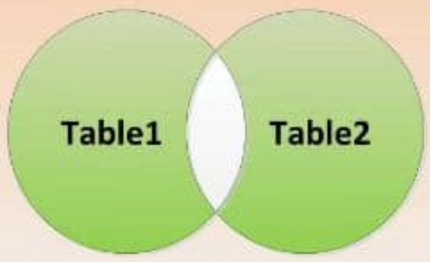

```sql
SELECT *
  FROM Table1 t1
  FULL OUTER JOIN Table2 t2
    ON t1.fk = t2.id;
```

FULL OUTER JOIN




```sql
SELECT *
  FROM Table1 t1
 CROSS JOIN Table2 t2;
```

CROSS JOIN, the Cartesian product




```sql
SELECT *
  FROM Table1 t1
  FULL OUTER JOIN Table2 t2
    ON t1.fk = t2.id
 WHERE t1.fk IS NULL
    OR t2.id IS NULL;
```

FULL OUTER JOIN with exclusion

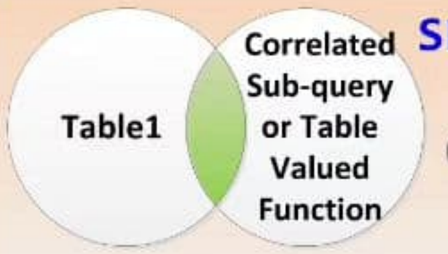

```sql
SELECT *
  FROM Table1 t1
 INNER JOIN Table2 t2
    ON t1.fk >= t2.id;
```

NON-EQUI INNER JOIN

SQL JOIN TYPES SIMPLIFIED

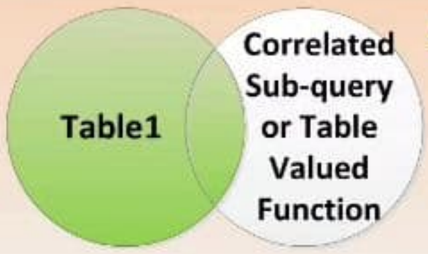
CODING BUGS NOTES GALLERY

CREATE BY - ATUL KUMAR (LINKEDIN)



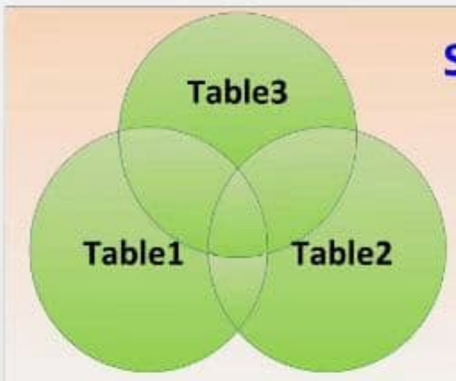

```
SELECT *
  FROM Table1 t1
CROSS APPLY
      [dbo].[someTVF](t1.fk)
      AS t;
```

CROSS APPLY



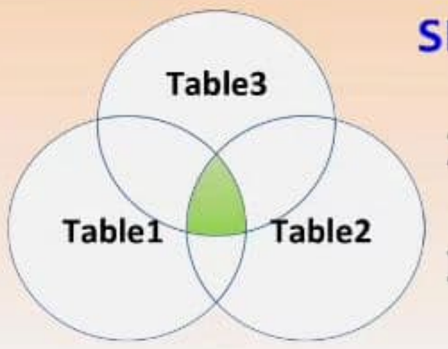

```
SELECT *
  FROM Table1 t1
OUTER APPLY
      [dbo].[someTVF](t1.fk)
       AS t;
```

OUTER APPLY



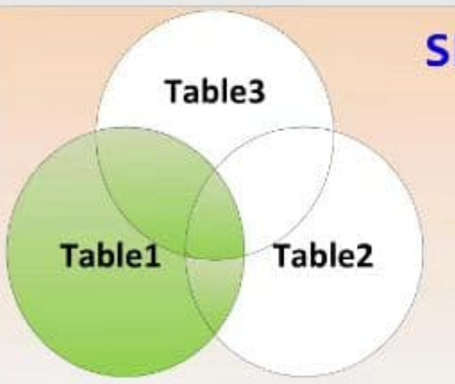

```
SELECT *
  FROM Table1 t1
  FULL OUTER JOIN Table2 t2
    ON t1.fk = t2.id
  FULL OUTER JOIN Table3 t3
    ON t1.fk_table3 = t3.id;
```

Two FULL OUTER JOINS



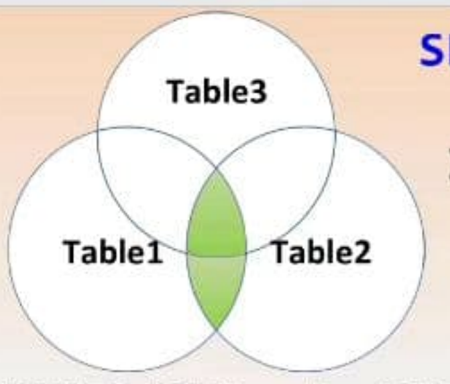

```
SELECT *
  FROM Table1 t1
INNER JOIN Table2 t2
   ON t1.fk = t2.id
INNER JOIN Table3 t3
   ON t1.fk_table3 = t3.id;
```

Two INNER JOINs




```
SELECT *
  FROM Table1 t1
LEFT OUTER JOIN Table2 t2
   ON t1.fk = t2.id
LEFT OUTER JOIN Table3 t3
   ON t1.fk_table3 = t3.id;
```

Two LEFT OUTER JOINS

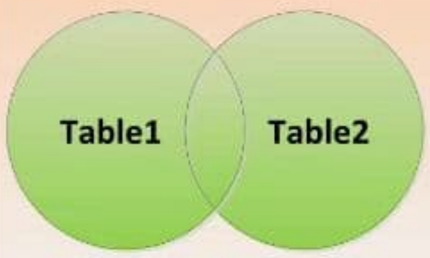



```
SELECT *
  FROM Table1 t1
INNER JOIN Table2 t2
   ON t1.fk = t2.id
LEFT OUTER JOIN Table3 t3
   ON t1.fk_table3 = t3.id;
```

INNER JOIN and a LEFT OUTER JOIN


```
SELECT fk as id
  FROM Table1
EXCEPT
SELECT ID
  FROM Table2;
```

EXCEPT




```
SELECT fk as id
  FROM Table1
UNION
SELECT ID
  FROM Table2;
```

UNION


```
SELECT fk as id
  FROM Table1
INTERSECT
SELECT ID
  FROM Table2;
```

INTERSECT

Sample Schema

Table 1 (People)

	id	Name	fk	fk_table3
1	1	Steve	1	NULL
2	2	Aaron	3	NULL
3	3	Mary	2	NULL
4	4	Fred	1	NULL
5	5	Anne	5	NULL
6	6	Beth	8	1
7	7	Johnny	NULL	1
8	8	Karen	NULL	2

Table 2 (Favorite Colors)

	id	FavoriteColor
1	1	red
2	2	green
3	3	blue
4	4	pink
5	5	purple
6	6	mauve
7	7	orange
8	8	yellow
9	1	indigo

Table 3 (Favorite Foods)

	id	dataValue
1	1	Pizza
2	2	Burger
3	3	Sushi

Note: Column names are very generic to simplify the sample queries.
Foreign keys are
Table1.fk -> Table2.id
Table2.fk_table3 -> Table3.id