Join Operation

- What if we also want to see the hours worked for each employee but also want to see their first names, last names, instead of the employee numbers?
- If we need to display columns from different tables, we must use a **Join Operation**.
- Two tables can be joined together to form a large table based on the matching of the primary key and the foreign key.
- E.g. Show the names of all the employees and the hours worked.

```
SELECT FirstName, LastName, HoursWorked
FROM EMPLOYEE, ASSIGNMENT
WHERE EMPLOYEE.EmployeeNumber = ASSIGNMENT.EmployeeNumber;
```
Join Operation

```sql
SELECT FirstName, LastName, HoursWorked
FROM EMPLOYEE, ASSIGNMENT
WHERE EMPLOYEE.EmployeeNumber = ASSIGNMENT.EmployeeNumber;
```
Join Operation

SELECT FirstName, LastName, HoursWorked
FROM EMPLOYEE, ASSIGNMENT
WHERE EMPLOYEE.EmployeeNumber = ASSIGNMENT.EmployeeNumber;

<table>
<thead>
<tr>
<th>firstname</th>
<th>lastname</th>
<th>hoursworked</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary</td>
<td>Jacobs</td>
<td>30.0</td>
</tr>
<tr>
<td>Mary</td>
<td>Jacobs</td>
<td>25.0</td>
</tr>
<tr>
<td>Mary</td>
<td>Jacobs</td>
<td>35.0</td>
</tr>
<tr>
<td>Rosalie</td>
<td>Jackson</td>
<td>20.0</td>
</tr>
<tr>
<td>Tom</td>
<td>Caruthers</td>
<td>40.0</td>
</tr>
<tr>
<td>Tom</td>
<td>Caruthers</td>
<td>45.0</td>
</tr>
<tr>
<td>Tom</td>
<td>Caruthers</td>
<td>15.0</td>
</tr>
<tr>
<td>Heather</td>
<td>Jones</td>
<td>40.0</td>
</tr>
<tr>
<td>Heather</td>
<td>Jones</td>
<td>10.0</td>
</tr>
<tr>
<td>Mary</td>
<td>Abernathy</td>
<td>45.0</td>
</tr>
<tr>
<td>Mary</td>
<td>Abernathy</td>
<td>27.5</td>
</tr>
<tr>
<td>Tom</td>
<td>Jackson</td>
<td>75.0</td>
</tr>
<tr>
<td>Tom</td>
<td>Jackson</td>
<td>80.0</td>
</tr>
<tr>
<td>Ken</td>
<td>Numoto</td>
<td>55.0</td>
</tr>
<tr>
<td>Ken</td>
<td>Numoto</td>
<td>50.0</td>
</tr>
</tbody>
</table>
Join Operation

- Join Operation –
  - Must list all the tables in the FROM clause.
  - Must indicate which field in what table is matched with which field in what table.
  - Duplicate field names from different tables must be prefixed by their table names using the dot notation.

- E.g.

```sql
SELECT EMPLOYEE.EmployeeNumber, FirstName, LastName, HoursWorked
FROM EMPLOYEE, ASSIGNMENT
WHERE EMPLOYEE.EmployeeNumber = ASSIGNMENT.EmployeeNumber;
```
<table>
<thead>
<tr>
<th>employeenumber</th>
<th>firstname</th>
<th>lastname</th>
<th>hoursworked</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mary</td>
<td>Jacobs</td>
<td>30.0</td>
</tr>
<tr>
<td>1</td>
<td>Mary</td>
<td>Jacobs</td>
<td>25.0</td>
</tr>
<tr>
<td>1</td>
<td>Mary</td>
<td>Jacobs</td>
<td>35.0</td>
</tr>
<tr>
<td>2</td>
<td>Rosalie</td>
<td>Jackson</td>
<td>20.0</td>
</tr>
<tr>
<td>4</td>
<td>Tom</td>
<td>Caruthers</td>
<td>40.0</td>
</tr>
<tr>
<td>4</td>
<td>Tom</td>
<td>Caruthers</td>
<td>45.0</td>
</tr>
<tr>
<td>4</td>
<td>Tom</td>
<td>Caruthers</td>
<td>15.0</td>
</tr>
<tr>
<td>5</td>
<td>Heather</td>
<td>Jones</td>
<td>40.0</td>
</tr>
<tr>
<td>5</td>
<td>Heather</td>
<td>Jones</td>
<td>10.0</td>
</tr>
<tr>
<td>6</td>
<td>Mary</td>
<td>Abernathy</td>
<td>45.0</td>
</tr>
<tr>
<td>6</td>
<td>Mary</td>
<td>Abernathy</td>
<td>27.5</td>
</tr>
<tr>
<td>8</td>
<td>Tom</td>
<td>Jackson</td>
<td>75.0</td>
</tr>
<tr>
<td>8</td>
<td>Tom</td>
<td>Jackson</td>
<td>80.0</td>
</tr>
<tr>
<td>10</td>
<td>Ken</td>
<td>Numoto</td>
<td>55.0</td>
</tr>
<tr>
<td>10</td>
<td>Ken</td>
<td>Numoto</td>
<td>50.0</td>
</tr>
</tbody>
</table>
Multiple Conditions in WHERE Clause

- Notice that the records that don’t match in two tables won’t show in the result.
- Can apply other WHERE conditions.
- E.g. Show the names of the employees and their hours worked from the Accounting department.

```sql
SELECT EMPLOYEE.EmployeeNumber, FirstName, LastName, HoursWorked, Department
FROM EMPLOYEE, ASSIGNMENT
WHERE EMPLOYEE.EmployeeNumber = ASSIGNMENT.EmployeeNumber AND Department = 'Accounting';
```
Multiple Conditions in WHERE Clause

SELECT EMPLOYEE.EmployeeNumber, FirstName, LastName, HoursWorked, Department
FROM EMPLOYEE, ASSIGNMENT
WHERE EMPLOYEE.EmployeeNumber = ASSIGNMENT.EmployeeNumber AND Department = 'Accounting';
Multiple Conditions in WHERE Clause

- E.g. Show the names of the employees and their hours worked but only interested in hours worked more than 45.

```sql
SELECT EmployeeNumber, FirstName, LastName, HoursWorked
FROM EMPLOYEE, ASSIGNMENT
WHERE EMPLOYEE.EmployeeNumber = ASSIGNMENT.EmployeeNumber AND HoursWorked > 45;
```
Table Aliases

- Using table aliases to avoid repeatedly typing long table names – in the FROM clause, use AS XX after a table name with XX as the new shortened name you give for that table.
- E.g. Show the names of the employees and their hours worked from the Accounting department.

```sql
SELECT E.EmployeeNumber, FirstName, LastName, HoursWorked, Department
FROM EMPLOYEE AS E, ASSIGNMENT AS A
WHERE E.EmployeeNumber = A.EmployeeNumber
AND Department = 'Accounting';
```
Join More Than Two Tables

- Can join more than two tables.
- E.g. Show the names of the employees and their hours worked on each project, including the project name.

```sql
SELECT P.ProjectName, FirstName, LastName, HoursWorked
FROM EMPLOYEE AS E, ASSIGNMENT AS A, PROJECT AS P
WHERE E.EmployeeNumber = A.EmployeeNumber
AND P.ProjectID = A.ProjectID
```
INNER JOIN ... ON

- Using the JOIN ... ON key words
- Another way to apply a JOIN operation
- ACCESS syntax: must use INNER JOIN ... ON
- E.g. Show the names of all the employees and the hours worked.

Without using the INNERJOIN key word:
```
SELECT FirstName, LastName, HoursWorked
FROM EMPLOYEE AS E, ASSIGNMENT AS A
WHERE E.EmployeeNumber = A.EmployeeNumber;
```

Using the INNER JOIN key word:
```
SELECT FirstName, LastName, HoursWorked
FROM EMPLOYEE AS E INNER JOIN ASSIGNMENT AS A ON
  E.EmployeeNumber = A.EmployeeNumber;
```
INNER JOIN ...

- Using the JOIN ... ON key words
- Another way to apply a JOIN operation
- ACCESS syntax: must use INNER JOIN ...
- E.g. Show the names of all the employees and the hours worked.

Without using the INNERJOIN key word:

```sql
SELECT FirstName, LastName, HoursWorked
FROM EMPLOYEE AS E, ASSIGNMENT AS A
WHERE E.EmployeeNumber = A.EmployeeNumber;
```

Using the INNER JOIN key word:

```sql
SELECT FirstName, LastName, HoursWorked
FROM EMPLOYEE AS E INNER JOIN ASSIGNMENT AS A ON
E.EmployeeNumber = A.EmployeeNumber;
```
INNER JOIN ... ON

- Using the JOIN ... ON key words
- Another way to apply a JOIN operation
- ACCESS syntax: must use INNER JOIN ... ON
- E.g. Show the names of all the employees and the hours worked.

Without using the INNERJOIN key word:

```sql
SELECT FirstName, LastName, HoursWorked
FROM EMPLOYEE AS E, ASSIGNMENT AS A
WHERE E.EmployeeNumber = A.EmployeeNumber;
```

Using the INNER JOIN key word:

```sql
SELECT FirstName, LastName, HoursWorked
FROM EMPLOYEE AS E INNER JOIN ASSIGNMENT AS A ON
E.EmployeeNumber = A.EmployeeNumber;
```
INNER JOIN ... ON

- Using the JOIN ... ON key words
- Another way to apply a JOIN operation
- ACCESS syntax: must use INNER JOIN ... ON
- E.g. Show the names of all the employees and the hours worked.

Without using the INNERJOIN key word:
```
SELECT FirstName, LastName, HoursWorked
FROM EMPLOYEE AS E, ASSIGNMENT AS A
WHERE E.EmployeeNumber = A.EmployeeNumber;
```

Using the INNER JOIN key word:
```
SELECT FirstName, LastName, HoursWorked
FROM EMPLOYEE AS E INNER JOIN ASSIGNMENT AS A ON
  E.EmployeeNumber = A.EmployeeNumber;
```