ITCS393 Database Systems Lab

Lab07: SET Operators

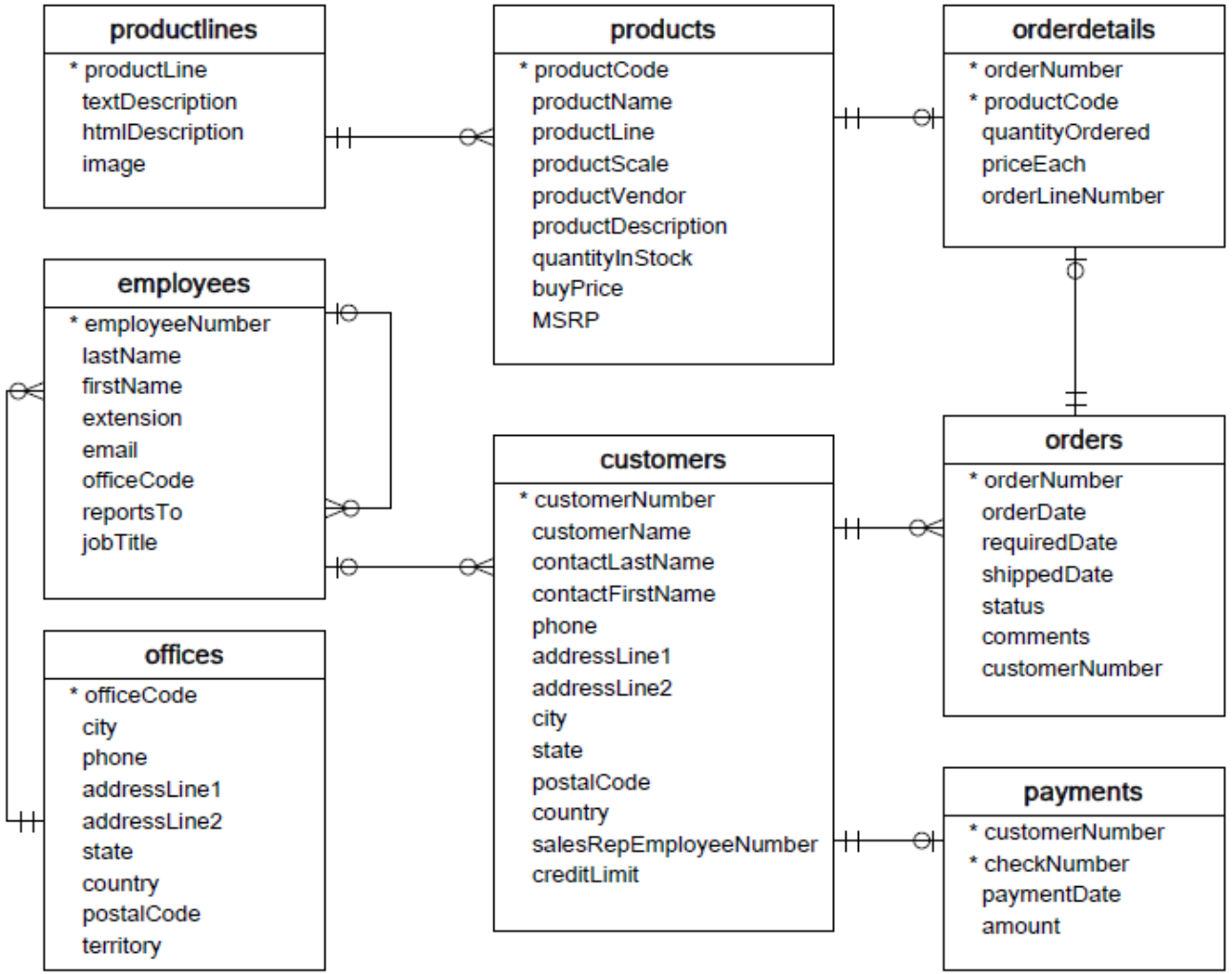
Dr. Petch Sajjacholapunt
Dr. Wudhichart Sawangphol
Dr. Jidapa Kraisangka

jidapa.kra@mahidol.edu

Class Objectives

- To learn on Set operators in SQL
- To learn how to emulate the Set operators in the limited circumstances
- To learn the individual query blocks with the Set operators
- To apply the Set operators to use in the given queries

Database: classicmodels



- **Customers**: stores customer's data.
- **Products**: stores a list of scale model cars.
- **ProductLines**: stores a list of product line categories.
- **Orders**: stores sales orders placed by customers.
- **OrderDetails**: stores sales order line items for each sales order.
- **Payments**: stores payments made by customers based on their accounts.
- **Employees**: stores all employee information as well as the organization structure such as who reports to whom.
- **Offices**: stores sales office data.

Original ERD and SQL from https://www.mysqltutorial.org/mysql-sample-database.aspx

SQL Set Operators

- SQL SET operators combines the results of two (`SELECT`) queries

```
SELECT the_first_select_query
SET_OPERATOR
SELECT another_select_query;
```

Union-compatible

- The number of columns needs to match between queries, and the data type of each column needs to be compatible.

Example:

```
-- Q: Employee's location
SELECT DISTINCT country FROM Customers
ORDER BY country;
```

country
Australia
Austria
Belgium
Canada
Denmark
Finland
France
Germany
Hong Kong
Ireland
Israel
Italy
Japan
Netherlands
New Zealand
Norway
Philippines
Poland
Portugal
Russia
Singapore
South Africa
Spain
Sweden
Switzerland
UK
USA

```
-- Q: Sales Rep's location
SELECT DISTINCT country
FROM Employees e
INNER JOIN Offices o ON e.officeCode = o.officeCode
WHERE e.jobTitle='Sales Rep'
```

country
Australia
France
Japan
UK
USA

Set Operators – UNION

UNION: Combines rows from two or more queries excluding duplicate rows

Example: What are the list of the countries of our customers and sales representative employee?

```
SELECT DISTINCT country
FROM Employees e
INNER JOIN Offices o ON e.officeCode = o.officeCode
WHERE e.jobTitle='Sales Rep'
UNION
SELECT DISTINCT country FROM Customers
ORDER BY country ASC;
```

5 countries

27 countries

The ORDER BY clause must be at the end of the last query for globally sorting

27 rows returned

country
Australia
Austria
Belgium
Canada
Denmark
Finland
France
Germany
Hong Kong
Ireland
Israel
Italy
Japan
Netherlands
New Zealand
Norway
Philippines
Poland
Portugal
Russia
Singapore
South Africa
Spain
Sweden
Switzerland
UK
USA

Set Operators – UNION ALL

UNION ALL: Combines rows from two or more queried including duplicate rows

Example: What are the list of the countries of our customers and sales representative employee?

```
SELECT DISTINCT country
FROM Employees e
INNER JOIN Offices o ON e.officeCode = o.officeCode
WHERE e.jobTitle='Sales Rep'
```

5 countries

```
UNION ALL
SELECT DISTINCT country FROM Customers
ORDER BY country ASC;
```

27 countries

32 rows returned

country
Australia
Australia
Austria
Belgium
Canada
Denmark
Finland
France
France
Germany
Hong Kong
Ireland
Israel
Italy
Japan
Japan
Netherlands
New Zealand
Norway
Philippines
Poland
Portugal
Russia
Singapore
South Africa
Spain
Sweden
Switzerland
UK
UK
USA
USA

Set Operators – INTERSECT

INTERSECT: Combines rows from two or more queries and return the rows that appear in both tables

Example:

```
SELECT DISTINCT country                    -- 5 countries
FROM Employees e
INNER JOIN Offices o ON e.officeCode = o.officeCode
WHERE e.jobTitle='Sales Rep'
INTERSECT
SELECT DISTINCT country                    -- 27 countries
FROM Customers
ORDER BY country ASC;
```

5 rows returned

country
Australia
France
Japan
UK
USA

Note: There is no "INTERSECT" operators in MySQL; emulate with the subquery

Set Operators – MINUS

MINUS: Returns the unique rows from the first table, which is not in the second table

Example:

```
SELECT DISTINCT country                    -- 27 countries
FROM Customers
MINUS
SELECT DISTINCT country                    --  5 countries
FROM Employees e
INNER JOIN Offices o ON e.officeCode = o.officeCode
WHERE e.jobTitle='Sales Rep'
ORDER BY country ASC;
```

22 rows returned

country
Austria
Belgium
Canada
Denmark
Finland
Germany
Hong Kong
Ireland
Israel
Italy
Netherlands
New Zealand
Norway
Philippines
Poland
Portugal
Russia
Singapore
South Africa
Spain
Sweden
Switzerland

Note: There is no "MINUS" operators in MySQL; emulate with the subquery

Set Operator Emulations

- Some DBMS, e.g., MySQL, do not support INTERSECT and MINUS operators
- Methods for Emulations
  - `INTERSECT` Operator using `DISTINCT` and `INNER JOIN` Clause
  - `INTERSECT` Operator using `IN` Operator
  - `MINUS` Operator using `LEFT OUTER JOIN` Clause
  - `MINUS` Operator using `NOT IN` Operator

RECAP: IN Operator

IN: Match multiple values in a WHERE clause (i.e., multiple `OR` conditions)

Example:

```
SELECT DISTINCT city, state FROM Customers
WHERE state NOT IN ('PA', 'CA', 'NY')
AND country = 'USA';
```

Match the values of the column with the multiple values

```
SELECT DISTINCT city, state,  country
FROM Customers
WHERE (city, state, country) IN
       (SELECT DISTINCT city, state, country
        FROM Offices
        WHERE state IS NOT NULL);
```

Match the set of the columns (using parenthesis to group the columns) with the multiple records from another queries.

INTERSECTION with INNER JOIN

Using INNER JOIN with the matched attribute condition

```
SELECT DISTINCT attr
FROM table1 INNER JOIN table2 c ON table1.attr = table2.attr;
```

Example:

```
SELECT DISTINCT o.country
FROM Employees e
INNER JOIN Offices o ON e.officeCode = o.officeCode -- table1
INNER JOIN
        (SELECT DISTINCT country FROM  Customers) c  -- table2
ON o.country = c.country
WHERE e.jobTitle='Sales Rep'
ORDER BY country ASC;
```

country
Australia
France
Japan
UK
USA

INTERSECTION with IN

Using IN to check the values in another table

```
SELECT DISTINCT attr
FROM table1
WHERE table1.attr IN
       (SELECT DISTINCT attr
        FROM table2);
```

Example:

```
SELECT DISTINCT o.country
FROM Employees e INNER JOIN Offices o
ON e.officeCode = o.officeCode -- table1
WHERE e.jobTitle='Sales Rep'
AND o.country IN
       (SELECT DISTINCT country FROM Customers) -- table2
ORDER BY country ASC;
```

country
Australia
France
Japan
UK
USA

MINUS with LEFT OUTER JOIN

Using LEFT OUTER JOIN with the matched attribute condition and IS NULL

```
SELECT attr
FROM table1 LEFT OUTER JOIN table2
ON table1.attr = table2.attr
WHERE table2.ID IS NULL;
```

Example:

```
SELECT DISTINCT c.country
FROM Customers c LEFT OUTER JOIN
       (SELECT o.country FROM Employees e
        INNER JOIN Offices o ON e.officeCode = o.officeCode
        WHERE e.jobTitle='Sales Rep') em
     ON c.country = em.country
WHERE em.country IS NULL
ORDER BY c.country ASC;
```

country
Austria
Belgium
Canada
Denmark
Finland
Germany
Hong Kong
Ireland
Israel
Italy
Netherlands
New Zealand
Norway
Philippines
Poland
Portugal
Russia
Singapore
South Africa
Spain
Sweden
Switzerland

MINUS with NOT IN

Using NOT IN to exclude the values from another table.

```
SELECT DISTINCT attr
FROM table1
WHERE table1.attr NOT IN
       (SELECT DISTINCT attr
        FROM table2);
```

```
SELECT DISTINCT country
FROM Customers
WHERE country NOT IN
       ( SELECT DISTINCT country
         FROM Employees e INNER JOIN Offices o
              ON e.officeCode = o.officeCode
         WHERE e.jobTitle='Sales Rep')
ORDER BY country ASC;
```

country
Austria
Belgium
Canada
Denmark
Finland
Germany
Hong Kong
Ireland
Israel
Italy
Netherlands
New Zealand
Norway
Philippines
Poland
Portugal
Russia
Singapore
South Africa
Spain
Sweden
Switzerland

Individual Query Blocks vs Entire Result

- Apply ( ) to create an individual query block used as part of a union, intersection, or other set operation.
- When applying `ORDER BY` or `LIMIT`, parenthesize the query block make differences

Example:

```
(SELECT Query Block 1)
Set_Operator
(SELECT Query Block 2)
[ORDER BY]    -- Sort the entire results
[LIMIT N] ;   -- Top N of the entire result
```

Example: Individual Query Blocks

Example: **UNION ALL**

```
SELECT attr FROM t1
UNION ALL
SELECT attr FROM t2
ORDER BY attr
LIMIT N;
```

```
(SELECT attr FROM t1)
UNION ALL
(SELECT attr FROM t2)
ORDER BY attr
LIMIT N;
```

```
(SELECT attr FROM t1)
UNION ALL
(SELECT attr FROM t2
ORDER BY attr)
LIMIT N;
```

```
(SELECT attr FROM t1
ORDER BY attr
LIMIT N)
UNION ALL
(SELECT attr FROM t2
ORDER BY attr
LIMIT N);
```

```
(SELECT attr FROM t1)
UNION ALL
(SELECT attr FROM t2
ORDER BY attr
LIMIT N);
```

Example: Individual Query Blocks

Example: **UNION ALL**

```
SELECT attr FROM t1
UNION ALL
SELECT attr FROM t2
ORDER BY attr
LIMIT N;
```

```
SELECT contactFirstname, contactLastname
FROM Customers   -- 1)
UNION ALL        -- 2)
SELECT firstName, lastName
FROM Employees   -- 1)
ORDER BY contactFirstName ASC -- 3)
LIMIT 5;         -- 4)
```

contactFirstname	contactLastname
Adrian	Huxley
Akiko	Shimamura
Alejandra	Camino
Alexander	Feuer
Alexander	Semenov

```
(SELECT attr FROM t1)
UNION ALL
(SELECT attr FROM t2)
ORDER BY attr
LIMIT N;
```

```
(SELECT contactFirstname, contactLastname
FROM Customers) -- 1)
UNION ALL       -- 2)
(SELECT firstName, lastName
FROM Employees  -- 1))
ORDER BY contactFirstName ASC -- 3)
LIMIT 5;        -- 4)
```

contactFirstname	contactLastname
Adrian	Huxley
Akiko	Shimamura
Alejandra	Camino
Alexander	Feuer
Alexander	Semenov

Example: Individual Query Blocks

Example: **UNION ALL**

```
(SELECT attr FROM t1)
UNION ALL
(SELECT attr FROM t2
ORDER BY attr)
LIMIT N;
```

```
(SELECT contactFirstname, contactLastname
FROM Customers -- 1) )
UNION ALL      -- 2)
(SELECT firstName, lastName
FROM Employees
ORDER BY firstName ASC -- 1) )
LIMIT 5;       -- 3)
```

contactFirstname	contactLastname
Carine	Schmitt
Jean	King
Peter	Ferguson
Janine	Labrune
Jonas	Bergulfsen

```
(SELECT attr FROM t1
ORDER BY attr
LIMIT N)
UNION ALL
(SELECT attr FROM t2
ORDER BY attr
LIMIT N);
```

```
(SELECT contactFirstname, contactLastname
FROM Customers
ORDER BY contactFirstname ASC
LIMIT 5)
UNION ALL
(SELECT firstName, lastName
FROM Employees
ORDER BY firstName ASC
LIMIT 5);
```

contactFirstname	contactLastname
Adrian	Huxley
Akiko	Shimamura
Alejandra	Camino
Alexander	Feuer
Alexander	Semenov
Andy	Fixter
Anthony	Bow
Barry	Jones
Diane	Murphy
Foon Yue	Tseng