Dr Simon D'Alfonso

INFO90002 Database Systems & Information Modelling

Week 03

SQL (1)

What is SQL

- SQL – or SEQUEL – is a language used to create, access and maintain relational databases
- Based on relational algebra and relational calculus
- SQL (DML) supports CRUD (Create, Read, Update, Delete)
  - Insert, Select, Update, Delete commands
- You can see the latest SQL 2011 standard at
  - http://www.jtc1sc32.org/doc/N2151-2200/32N2153T-text_for_ballot-FDIS_9075-1.pdf
- SQL is a widely used standard and there are resources online:
  - http://en.wikipedia.org/wiki/SQL
  - http://en.wikipedia.org/wiki/Category:SQL_keywords
  - https://dev.mysql.com/doc/refman/5.7/en/sql-syntax.html
  - https://www.w3schools.com/sql/

Brief history

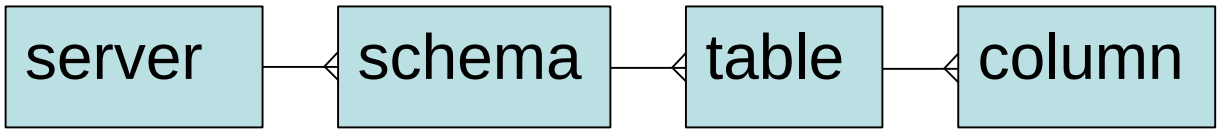
1974	IBM develops SEQUEL (renamed to SQL) based on Codd research
1979…	Oracle, IBM etc release RDBMS with SQL language
1986	1st SQL Standard (ANSI)
1989	2nd SQL Standard (ANSI) – includes referential integrity
1992	3rd SQL Standard (ISO) – most widely conformed to by vendors
1997…	dynamic websites enabled by SQL
1999	SQL-1999 – 4th SQL Standard (ISO) – Object support, recursion, procedures and flow control
2003	SQL-2003 – 5th SQL Standard (ISO) – XML support, auto number
2006	SQL-2006 – 6th SQL Standard (ISO) – Defines SQL use with XML
2008	SQL-2008 – 7th SQL Standard (ISO) – FETCH command added
2008	HTML 5 with SQLite built in
2011	SQL-2011 – 8th SQL Standard (ISO) – temporal databases

How we use SQL

- during Implementation of the database
  - Implement tables from physical design using DDL Create Table
- during Production
  - use Select commands to read the data from the tables
  - use DML Insert, Delete, Update commands to update data
  - use DDL Alter, Drop commands to update the database structure

MySQL

- We are using the MySQL implementation of SQL
  - If you are using other DBMS (such as ORACLE or SQLServer) you will need to check their implementation of SQL.
    - differences can range from valid keywords to data types
- The university's MySQL server = version 8
- You can get the latest version of MySQL (8.?) from
    - http://dev.mysql.com/downloads/
    - Community edition = FOSS
  - Get syntax help for MySQL SQL statements at
    - http://dev.mysql.com/doc/refman/5.7/en/sql-syntax.html
- Explore your server with these commands:
  - show schemas; (alternatively, 'show databases')
  - show tables;
  - describe table;



The SQL Language

- Consists of:
  - Data Definition Language (DDL)
    - to define and set up the database
    - CREATE, ALTER, DROP
      - also TRUNCATE, RENAME
  - Data Manipulation Language (DML)
    - to manipulate and read data in tables
    - SELECT, INSERT, DELETE, UPDATE
      - MySQL also provides others…. eg REPLACE
  - Data Control Language (DCL)
    - to control access to the database
      - GRANT, REVOKE
  - Other Commands
    - administer the database
    - transaction control

The SELECT Statement in detail

SELECT [ALL | DISTINCT] ***select_expr*** **[,** ***select_expr*** **...]**

List the columns (and expressions) that are returned from the query

[FROM ***table_references***

Indicate the table(s) or view(s) from where the data is obtained

[WHERE ***where_condition*****]**

Indicate the conditions on whether a particular row will be in the result

[GROUP BY {*col_name* **|** ***expr*** **} [ASC | DESC], ...]**

Indicate categorisation of results

[HAVING ***where_condition*****]**

Indicate the conditions under which a particular category (group) is included in the result

[ORDER BY {*col_name* **|** ***expr*** **|** ***position*****} [ASC | DESC], ...]**

Sort the result based on the criteria

[LIMIT {[*offset*,**] row_count | row_count OFFSET** ***offset*****}]**

Limit which rows are returned by their return order (ie 5 rows, 5 rows from row 2)

]

Select entire contents of table

Customer	
PK	**CustomerID**
	CustFirstName CustMiddleName **CustLastName** BusinessName **CustType**

```
SELECT * FROM Customer;
```

Edit: Export: Autosize:

CustomerID	CustFirstName	CustMiddleName	CustLastName	BusinessName	CustType
1	Peter	NULL	Smith	NULL	Personal
2	James	NULL	Jones	JJ Enterprises	Company
3	Akin	NULL	Smithies	Bay Wart	Company
4	Julie	Anne	Smythe	Konks	Company
5	Jen	NULL	Smart	BRU	Company
6	Lim	NULL	Lam	NULL	Personal
7	Kim	NULL	Unila	Saps	Company
8	James	Jay	Jones	JJ's	Company
9	Keith	NULL	Samson	NULL	Personal
NULL	NULL	NULL	NULL	NULL	NULL

Select specific columns

Customer	
PK	**CustomerID**
	CustFirstName CustMiddleName **CustLastName** BusinessName **CustType**

```
SELECT CustLastName, CustFirstName
    FROM Customer;
```

Export: Autosize:

CustLastName	CustFirstName
Smith	Peter
Jones	James
Smithies	Akin
Smythe	Julie
Smart	Jen
Lam	Lim
Unila	Kim
Jones	James
Samson	Keith

WHERE clause: select specific rows

Customer	
PK	**CustomerID**
	CustFirstName CustMiddleName **CustLastName** BusinessName **CustType**

```
SELECT CustLastName FROM Customer
    WHERE CustLastName = "Smith";
```

Export: Autosize

CustLastName
Smith

WHERE clause with LIKE

Customer	
PK	**CustomerID**
	CustFirstName CustMiddleName **CustLastName** BusinessName **CustType**

```
SELECT CustLastName FROM Customer
    WHERE CustLastName LIKE "Sm%";
```

Export Autosize

CustLastName
Smith
Smithies
Smythe
Smart

Select with ORDER BY

Customer	
PK	**CustomerID**
	CustFirstName CustMiddleName **CustLastName** BusinessName **CustType**

```
SELECT CustLastName, CustType
    FROM Customer
    ORDER BY CustLastName;
```

Export:

CustLastName	CustType
Jones	Company
Jones	Company
Lam	Personal
Samson	Personal
Smart	Company
Smith	Personal
Smithies	Company
Smythe	Company
Unila	Company

```
SELECT CustLastName, CustType
    FROM Customer
    ORDER BY CustLastName DESC;
```

Export:

CustLastName	CustType
Unila	Company
Smythe	Company
Smithies	Company
Smith	Personal
Smart	Company
Samson	Personal
Lam	Personal
Jones	Company
Jones	Company

Select with LIMIT

Customer	
PK	**CustomerID**
	CustFirstName CustMiddleName **CustLastName** BusinessName **CustType**

```
SELECT CustLastName, CustType
    FROM Customer
    ORDER BY CustLastName
    LIMIT 5;
```

Export:

CustLastName	CustType
Jones	Company
Jones	Company
Lam	Personal
Samson	Personal
Smart	Company

```
SELECT CustLastName, CustType
    FROM Customer
    ORDER BY CustLastName
    LIMIT 5
    OFFSET 3;
```

Export:

CustLastName	CustType
Samson	Personal
Smart	Company
Smith	Personal
Smithies	Company
Smythe	Company

CustLastName	CustType
Jones	Company
Jones	Company
Lam	Personal
Samson	Personal
Smart	Company
Smith	Personal
Smithies	Company
Smythe	Company
Unila	Company

(what happens if we Limit without Ordering?)

Select with GROUP BY

Customer	
PK	**CustomerID**
	CustFirstName CustMiddleName **CustLastName** BusinessName **CustType**

```
SELECT CustType, Count(CustomerID)
    FROM Customer
    GROUP BY CustType;
```

Export: Autosize:

CustType	Count(CustomerID)
Personal	3
Company	6

```
SELECT CustType, Count(CustomerID) AS Count
    FROM Customer
    GROUP BY CustType;
```

Export: Autosize:

CustType	Count
Personal	3
Company	6

Select with WHERE and GROUP BY

Customer	
PK	**CustomerID**
	CustFirstName CustMiddleName **CustLastName** BusinessName **CustType**

```
SELECT CustType, Count(CustomerID)
    FROM Customer
    WHERE CustLastName LIKE "Sm%"
    GROUP BY CustType;
```

Export: Autosize

CustType	Count(CustomerID)
Personal	1
Company	3

Select with GROUP BY and HAVING

Customer	
PK	**CustomerID**
	CustFirstName CustMiddleName **CustLastName** BusinessName **CustType**

```
1  select custtype, count(CustomerId) from customer
2  where custlastname like 'Sm%'
3  group by custtype
4  having count(CustomerId) = 3;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

custtype	count(CustomerId)
Company	3

"Having" works on groups
the way "Where" works on individual rows

Inner join, Natural join

- Data about an entity is spread across 2 tables – so join them
- Inner/Equi join - Join rows where FK value = PK value

```
SELECT * FROM Customer INNER JOIN Account
    ON Customer.CustomerID = Account.CustomerID;
```

Export: Autosize:

CustomerID	CustFirstName	CustMiddleName	CustLastName	BusinessName	CustType	AccountID	AccountName	OutstandingBalance	CustomerID
1	Peter	NULL	Smith	NULL	Personal	1	Peter Smith	245.25	1
2	James	NULL	Jones	JJ Enterprises	Company	2	JJ ENt.	552.39	2
2	James	NULL	Jones	JJ Enterprises	Company	3	JJ ENt. Mgr	10.25	2

- Natural Join gives the same result as Inner Join
  - requires PK and FK columns to have the same name

```
SELECT * FROM Customer NATURAL JOIN Account;
```

Export: Autosize:

CustomerID	CustFirstName	CustMiddleName	CustLastName	BusinessName	CustType	AccountID	AccountName	OutstandingBalance
1	Peter	NULL	Smith	NULL	Personal	1	Peter Smith	245.25
2	James	NULL	Jones	JJ Enterprises	Company	2	JJ ENt.	552.39
2	James	NULL	Jones	JJ Enterprises	Company	3	JJ ENt. Mgr	10.25

Outer join

- Outer join
  - Can be left or right (see difference below)
  - Includes records from left/right table that don't have a matching row

```
SELECT * FROM Customer LEFT OUTER JOIN Account
    ON Customer.CustomerID = Account.CustomerID;
```

Export: Autosize:

CustomerID	CustFirstName	CustMiddleNa	CustLastName	BusinessName	CustType	AccountID	AccountName	OutstandingBalance	CustomerID
1	Peter	NULL	Smith	NULL	Personal	1	Peter Smith	245.25	1
2	James	NULL	Jones	JJ Enterprises	Company	2	JJ ENt.	552.39	2
2	James	NULL	Jones	JJ Enterprises	Company	3	JJ ENt. Mgr	10.25	2
3	Akin	NULL	Smithies	Bay Wart	Company	NULL	NULL	NULL	NULL

```
SELECT * FROM Customer RIGHT OUTER JOIN Account
    ON Customer.CustomerID = Account.CustomerID;
```

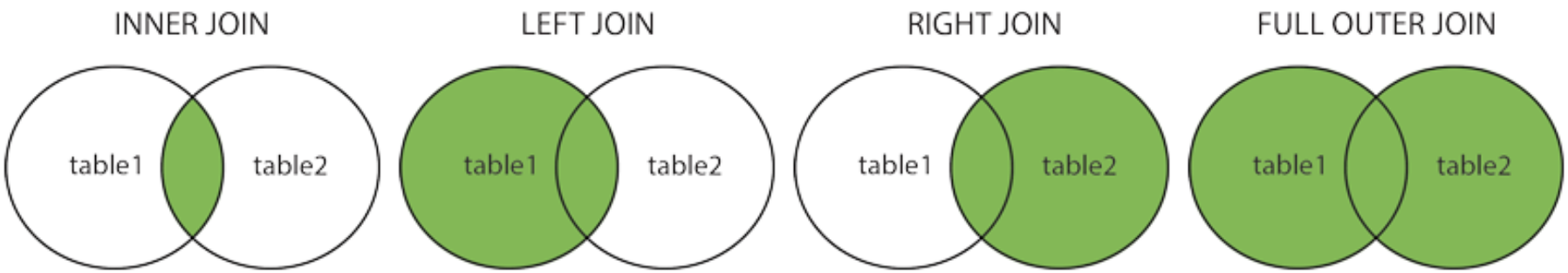
Export: Autosize:

CustomerID	CustFirstName	CustMiddleName	CustLastName	BusinessName	CustType	AccountID	AccountName	OutstandingBalance	CustomerID
1	Peter	NULL	Smith	NULL	Personal	1	Peter Smith	245.25	1
2	James	NULL	Jones	JJ Enterprises	Company	2	JJ ENt.	552.39	2
2	James	NULL	Jones	JJ Enterprises	Company	3	JJ ENt. Mgr	10.25	2

Outer Joins

- There is also a FULL OUTER JOIN, which gives us all records from both tables, whether or not they have a match in the other table.
- FULL OUTER JOIN is not supported in MySQL (but can be emulated). https://www.w3schools.com/sql/sql_join.asp:

- **(INNER) JOIN**: Returns records that have matching values in both tables
- **LEFT (OUTER) JOIN**: Return all records from the left table, and the matched records from the right table
- **RIGHT (OUTER) JOIN**: Return all records from the right table, and the matched records from the left table
- **FULL (OUTER) JOIN**: Return all records when there is a match in either left or right table



Cartesian join

- What if there is no join condition?

```
SELECT * FROM Customer, Account;
```

Export: Autosize:

CustomerID	CustFirstName	CustMiddleName	CustLastName	BusinessName	CustType	AccountID	AccountName	OutstandingBalance	CustomerID
1	Peter	NULL	Smith	NULL	Personal	1	Peter Smith	245.25	1
2	James	NULL	Jones	JJ Enterprises	Company	1	Peter Smith	245.25	1
3	Akin	NULL	Smithies	Bay Wart	Company	1	Peter Smith	245.25	1
1	Peter	NULL	Smith	NULL	Personal	2	JJ ENt.	552.39	2
2	James	NULL	Jones	JJ Enterprises	Company	2	JJ ENt.	552.39	2
3	Akin	NULL	Smithies	Bay Wart	Company	2	JJ ENt.	552.39	2
1	Peter	NULL	Smith	NULL	Personal	3	JJ ENt. Mgr	10.25	2
2	James	NULL	Jones	JJ Enterprises	Company	3	JJ ENt. Mgr	10.25	2
3	Akin	NULL	Smithies	Bay Wart	Company	3	JJ ENt. Mgr	10.25	2

NOT CORRECT: lack of join conditions -> Cartesian product

(every row in Customer combined with every record in Account)