COMM1822

Term 2 2022

Introduction to Databases for Business Analytics

Week 3: SQL 1

Lecturer-in-Charge: Kam-Fung (Henry) Cheung
Email: kf.cheung@unsw.edu.au
Tutors: Theresa Tran (Tutor-in-Charge)
Liam Li Chen
Kathy Xu
PASS Leader: Srilekha Chandrashekara Kolaki

UNSW SYDNEY

Copyright

- There are some file-sharing websites that specialise in buying and selling academic work to and from university students.

- If you upload your original work to these websites, and if another student downloads and presents it as their own either wholly or partially, **you might be found guilty of collusion — even years after graduation**.

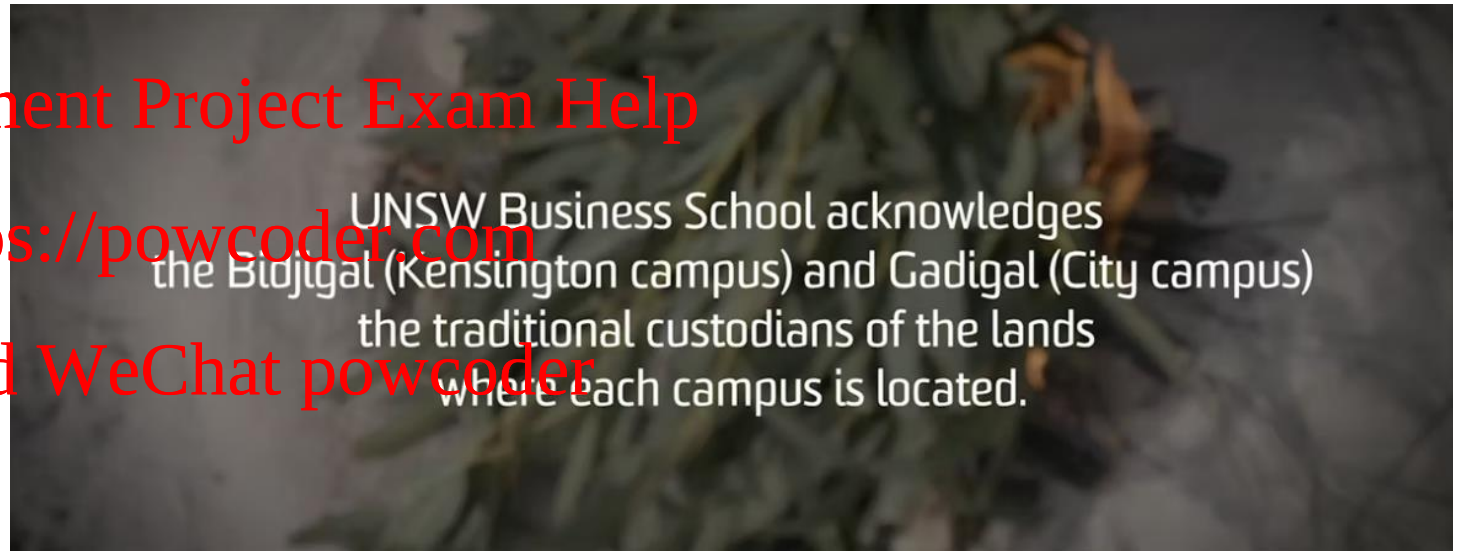
- These file-sharing websites may also accept purchase of course materials, **such as copies of lecture slides and tutorial handouts. By law, the copyright on course materials, developed by UNSW staff in the course of their employment, belongs to UNSW. It constitutes copyright infringement, if not academic misconduct, to trade these materials.**

Acknowledgement of Country

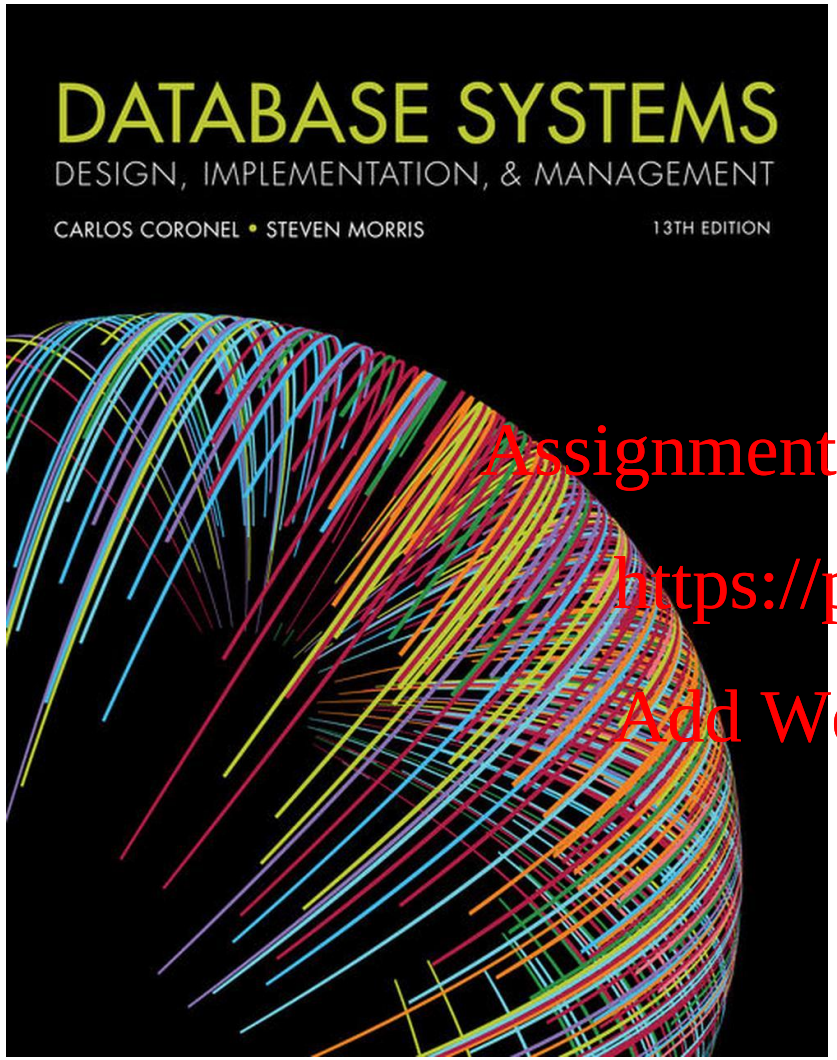
UNSW Business School acknowledges the Bidjigal (Kensington campus) and Gadigal (City campus) the traditional custodians of the lands where each campus is located.

We acknowledge all Aboriginal and Torres Strait Islander Elders, past and present and their communities who have shared and practiced their teachings over thousands of years including business practices.

We recognise Aboriginal and Torres Strait Islander people's ongoing leadership and contributions, including to business, education and industry.



UNSW Business School. (2022, May 7). *Acknowledgement of Country* [online video]. Retrieved from https://vimeo.com/369229957/d995d8087f



Chapter 7
Introduction to Structured Query Language (SQL)

SeQueL

UNSW SYDNEY

Relational Languages

Relational DBMS's query languages (e.g., SQL in Oracle) contain three components:

- **Data Definition Language (DDL)**: used to specify/modify the database schema.

- **Data Control Language (DCL)**: used to control the DB (e.g., user rights).

- **Data Manipulation Language (DML):** used to retrieve/manipulate data.

SQL

- **SQL = Structured Query Language = Sequel**
- SQL is the **first standard database language**.
- Originally **developed by D. Chamberlin and R. Boyce at IBM**.
- The most common SQL standard is the **ANSI/ISO SQL**. Originally defined in 1988, **SQL-86**, it has been undergone major revisions in 1992, **SQL-92**, and 1999, **SQL-99**. The latest version is **SQL-2011**.
- Microsoft, Oracle, and other vendors have introduced deviations from ANSI SQL.
- As a relational language, SQL has **three main components**:
  - Data Definition Language (DDL)
  - Data Manipulation Language (DML)
  - Data Control Language (DCL)

SQL DLL (Data Definition Language)

- To create the database structure:

CREATE SCHEMA AUTHORIZATION Creator
Example: CREATE SCHEMA AUTHORIZATION Chris

CREATE DATABASE Database_Name
Example: CREATE DATABASE Student

- To create tables:

CREATE TABLE Table_Name (column_name data_type [NULL|NOT NULL] [,…])

SQL DDL

- Example of table creation:

```
CREATE TABLE COURSE (
        COURSE_CODE       CHAR (8)        NOT NULL,
        COURSE_NAME       VARCHAR (18)    NOT NULL,
        PRGRAM_CODE       CHAR (4)        NOT NULL,
        SEMESTER          CHAR (1),
        PRIMARY KEY               (COURSE_CODE),
        FOREIGN KEY               (PROGRAM_CODE)
);
```


Another table, may call PROGRAM, exists with PK as PROGRAM_CODE

(SQL example from DBMS Microsoft Access)

SQL DDL

- Example of table creation:


```
CREATE TABLE COURSE_1 (
   COURSE_CODE       VarCHAR (8),
   COURSE_NAME       VarCHAR (8)
   CONSTRAINT COURSE_CODE_PK      PRIMARY KEY
(COURSE_CODE)
);
```

SQL DML (Data Manipulation Language)

- ANSI/ISO SQL standard use the terms "**tables**," "**columns**" and "**rows**" (not relations, attributes, and tuples)
- The **principal SQL DML statements** are:
  - SELECT
  - INSERT
  - UPDATE
  - DELETE
- Complete SQL statements consists of **reserved words** and **user-defined** words:
  - The **reserved words** are fixed **part of the language**.
  - The **user-defined words** represent the meaning of the data to the user (e.g., "users", "bookings").

SQL Query Structures

- The **SELECT** statement is used to retrieve and display data from one or more tables.
- Relation algebra's **selection, projection and join statements** can be performed with **one single SELECT statement**.

- "**SELECT FROM WHERE**"
  - **SELECT** clause tells which attributes [columns] of the tuples [rows] matching the condition are produced as part of the answer.
  - **FROM** clause gives the names of relation(s) [table(s)].
  - **WHERE** clause is a condition that tuples [rows] must satisfy in order to match the query.

SQL Query Structures

SELECT [DISTINCT | ALL] {| [column_expression AS new_name] [, …]}
FROM table_name [alias] [, …]
[**WHERE** condition]
[GROUP BY column_list]
[HAVING condition]
[ORDER BY column_list];

[] : indicates optional elements.
{} : indicates that the element may or may not appear.
| : indicates “or.”
; : indicates the end of the statement.

SQL Query Structures

EMPLOYEE (**Employee_ID,** Employee_FName, Employee_LName, Employee_HireDate, Employee_Title)

CERTIFIED (**Employee_ID, Skill_ID**, Certified_Date)

SKILL (**Skill_ID,** Skill_Name, Skill_Description)

FIGURE 1.5 EMPLOYEE SKILL CERTIFICATIONS IN A GOOD DESIGN

Database name: Ch01_Text

Table name: EMPLOYEE

Employee_ID	Employee_FName	Employee_LName	Employee_HireDate	Employee_Title
02345	Johnny	Jones	2/14/1995	DBA
03373	Franklin	Johnson	3/15/2002	Purchasing Agent
04893	Patricia	Richards	6/1/2004	DBA
06234	[illegible]	[illegible]	8/10/2005	Programmer
08273	Marco	Bienz	7/28/2006	Analyst
09002	Ben	Joiner	5/20/2010	Clerk
09283	Juan	Chavez	7/4/2010	Clerk
09382	Jessica	Johnson	8/2/2010	Database Programmer
10282	Amanda	Richardson	4/11/2011	Clerk
12383	Raymond	Matthew	3/12/2012	Programmer
13567	Robert	Arnold	9/30/2012	Analyst
13932	Megan	Lee	9/29/2013	Programmer
14311	Lee	Duong	9/1/2014	Programmer

Table name: CERTIFIED

Employee_ID	Skill_ID	Certified_Date
02345	100	2/14/2002
02345	110	8/9/2003
02345	180	2/14/2005
03373	120	6/20/2011
04893	180	6/11/2006
04893	220	9/20/2012
06234	110	8/10/2007
06234	200	8/10/2007
06234	210	1/29/2012
08273	110	3/8/2009
08273	190	8/19/2012
09002	110	5/16/2013
09002	120	5/16/2013
09382	140	8/2/2012
09382	210	8/2/2012
09382	220	5/1/2013
13383	170	3/12/2014
13567	130	9/30/2014
13567	140	5/23/2015
14311	110	9/1/2016

Table name: SKILL

Skill_ID	Skill_Name	Skill_Description
100	Basic Database Management	Create and manage database user accounts.
110	Basic Web Design	Create and maintain HTML and CSS documents.
120	Advanced Spreadsheets	Use of advanced functions, user-defined functions, and macroing.
130	Basic Process Modeling	Create core business process models using standard libraries.
140	Basic Database Design	Create simple data models.
150	Master Database Programming	Create integrated trigger and procedure packages for a distributed environment.
160	Basic Spreadsheets	Create single tab worksheets with basic formulas
170	Basic C# Programming	Create single-tier data aware modules.
180	Advanced Database Management	Manage Database Server Clusters.
190	Advance Process Modeling	Evaluate and Redesign cross-functional internal and external business processes.
200	Advanced C# Programming	Create multi-tier applications using multi-threading
210	Basic Database Manipulation	Create simple data retrieval and manipulation statements in SQL.
220	Advanced Database Manipulation	Use of advanced data manipulation methods for multi-table inserts, set operations, and correlated subqueries.

SQL Query Structures

SQL allows us to use the keyword ALL to specify that all tuples are to be selected.

SELECT ALL
FROM EMPLOYEE;

or

SELECT *
FROM EMPLOYEE;

* : is a “wild card.”

SQL Query Structures

- The SQL syntax is basically:

SELECT <columns>
FROM <table>;

- List all Skill Name and Skill Description:

SELECT Skill_Name, Skill_Description
FROM SKILL;

- SQL supports the elimination of duplicates by using the keyword **DISTINCT**.

SELECT DISTINCT Employee_ID
FROM CERTIFIED;

WHERE Clause Options

- Selecting rows with conditional restrictions
  - WHERE clause is used to add conditional restrictions to the SELECT statement that limit the rows returned by the query
  - Syntax:

```
SELECT      columnlist
FROM        tablelist
[WHERE      conditionlist]
[ORDER BY   columnlist [ASC | DESC] ];
```


- Using comparison operators on character attributes
  - May be used to place restrictions on character-based attributes
- Using comparison operators on dates
  - Date procedures are often more software-specific than other SQL procedures

SQL Query Structures

For instance, in the previous example, we only interest in “Basic Database Manipulation”, we can put a condition in the WHERE clause:


```
SELECT Skill_Name, Skill_Description
FROM SKILL
WHERE Skill_Name = “Basic Database Manipulation”;
```

Mathematical Operators for SQL

Mathematical operators that can be used in a **WHERE clause** for comparison:

= equal to
< less than
<= less than or equal to
> greater than
>= greater than or equal to
<> not equal to

Mathematical Operators for SQL

- Create a list of product description, product in-date and product price for products sold by vendor that are not coded “21344”.


```
SELECT P_Description, P_Indate, P_Price, V_Code
FROM PRODUCT
WHERE V_Code <> 21344;
```


- Create a list of product description, product on hand, product minimum, and product price for products with product code less than “1558-QWI”.

```
SELECT P_Description, P_Onhand, P_Min, P_Price
FROM PRODUCT
WHERE P_Code < ‘1558-QWI’ ;
```

ASCII Codes in SQL

- All characters/signs are assigned an **ASCII** (American Standard Code for Information Interchange) code by the computer.
- See manual or online for more information on ASCII codes.
- The comparisons of strings are made from left to right. This is useful when comparing names. However, it also may create problems:
  - "2" is sorted as if greater than "11" (because "2" > "1").
  - "01/01/2020" is sorted before "12/31/2015" (because "0" < "1").
  - Recommendation: use the date/number format instead of string.

Character	ASCII Code
A	65
a	97
B	66
*	42
0	48
1	49
2	50

Logical (Boolean) Operators in SQL

Boolean Operators:

- ❑ OR
- ❑ AND
- ❑ NOT

- ❑ List products where the vendor code is '21344' or '24288':


```
SELECT P_Description, P_Indate, P_Price, V_Code
FROM PRODUCT
WHERE V_Code = 21344 OR V_Code = 24288;
```


- ❑ List products where either the product in-date is after July 15, 2015 and the product price is less than 50.00 – or the vendor code is 24288.

```
SELECT P_Description, P_Indate, P_Price, V_Code
FROM PRODUCT
WHERE (P_Price < 50 AND P_Indate > '07/15/15') OR V_Code = 24288;
```

Special Operators in SQL

- **BETWEEN** is used to define range limits.
- **IS NULL** is used to check whether an attribute value is null.
- **LIKE** is used to check for similar character strings.
- **IN** is used to check whether an attribute value matches a value contains within a subset of listed values.
- **EXISTS** is used to check whether an attribute has a value.

Special Operators in SQL

- **BETWEEN** is used to define range limits.

Examples:

- List the products with prices between 50 and 100.


```
SELECT *
FROM PRODUCT
WHERE P_Price BETWEEN 50.00 AND 100.00;
```


or

```
SELECT *
FROM PRODUCT
WHERE P_Price >= 50.00 AND P_Price <= 100.00;
```

Special Operators in SQL

- **LIKE** is used to check for similar character strings.

- List the details of all vendors whose last name begins with “Smith”.


```
SELECT    V_Name, V_Contact, V_AreaCode, V_Phone
FROM      VENDOR
WHERE     V_Contact LIKE ‘Smith%’;
```


% : wild card

Special Operators in SQL

- **IN** is used to check whether an attribute value matches a value contains within a subset of listed values.

- List the contents of the product table where the product price is $ 50 or $ 100.


```
SELECT *
FROM   PRODUCT
WHERE P_Price = 50.00 OR P_Price = 100.00;
```

or

```
SELECT *
FROM   PRODUCT
WHERE P_Price IN (50.00, 100.00);
```

Special Operators in SQL

- **IS NULL** is used to check whether an attribute value is null.
- **EXISTS** is used to check whether an attribute has a value.
- List the details of products with existing (not-NULL) vendor codes.


```
SELECT  *
FROM    PRODUCT
WHERE   V_Code EXISTS;
```

or

```
SELECT  *
FROM    PRODUCT
WHERE   NOT ISNULL (V_Code);
```

Ordering SQL Results

- **ORDER BY** <columns> : produces a list in ascending order
- **ORDER BY** <columns> [**DESC**] : produces a list in descending order
- List the details of product table listed by product price in ascending order:

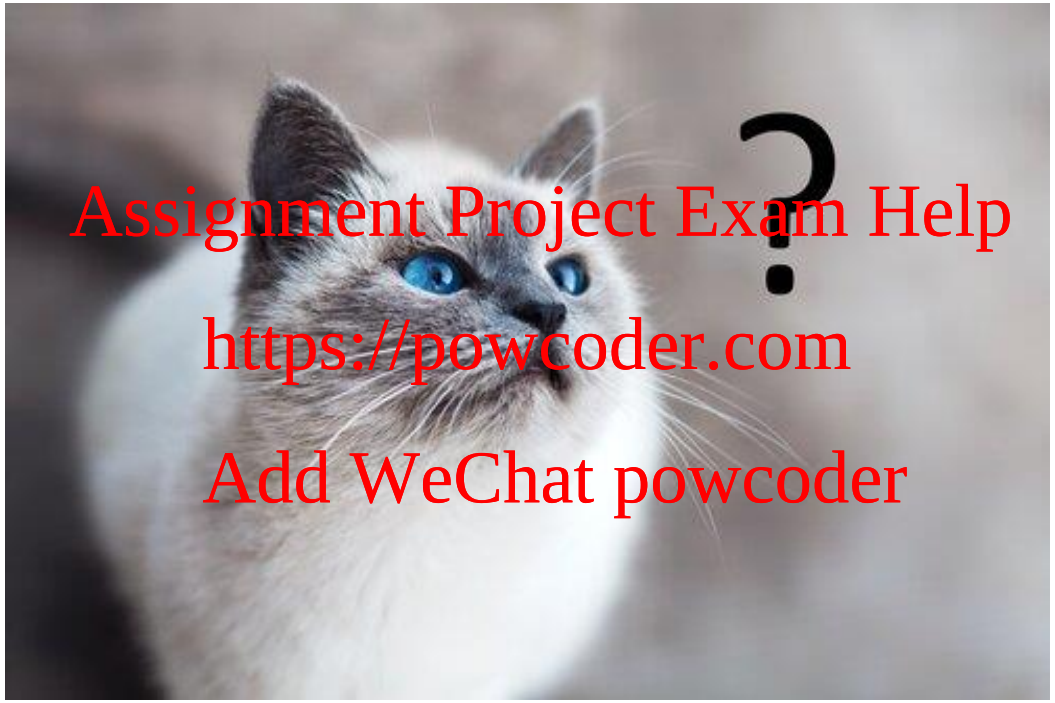

```
SELECT      P_Description, P_Indate, P_Price, V_Code
FROM        PRODUCT
ORDER BY    P_Price;
```

Ordering SQL Results

- List the details of of products with an in-date before 15 September 1999 and a price less than A$ 50.
- Put the results in ascending order of vendor code and descending order of price.


```
SELECT      P_Description, P_Indate, P_Price, V_Code
FROM        PRODUCT
WHERE       P_Indate < '9/15/99' AND P_Price <= 50.00
ORDER BY    V_Code, P_Price DESC;
```

Questions



Source: ealt.ca

Please email your question(s) to kf.cheung@unsw.edu.au!