DBS311: Advanced Database Services

Lecture 1

DBS311

- Oracle accounts supplied
- Some examples presented with DB2
- Your work is done with Oracle

Number of Companies worldwide using DB2

with revenue above 1 billion	–	6,585
with revenue 500m - 1 billion	–	1,915
with revenue 50m – 500m	–	7,228
with revenue 1m – 50m	–	23,382
with revenue below 1m	–	2,680

- https://discovery.hgdata.com/product/ibm-db2

DBS311 is a Pre Requisite for

- DBT544 – Professional Option
- DB2
- A lot more
- Embedded SQL in programs
- Host variables, parameter markers
- Dynamic & Static SQL
- Commitment Control in a program

Agenda

1. DBS311 Classroom Policies
2. Review
3. Single Row Functions
4. First Lab

Tests, Labs & Assignments

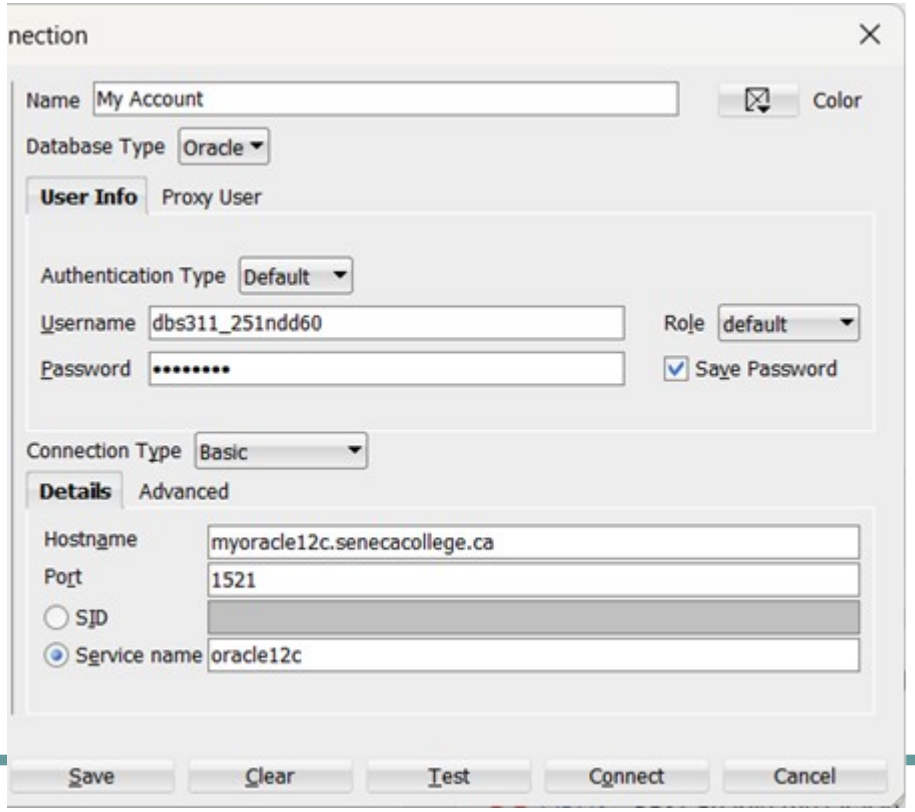
- 2 Tests
- Lab Demonstrations
- 10 labs
- Read first document in the Labs Folder
- Assignments
- 2 assignments A1 – 5%, A2 – 15%
- Groups for assignment

Lectures & Office Hours

- Teams is used for the lecture period
- We will use zoom for the three virtual office hours
- Monday at 8:55 AM & 10:45 AM, Thursday at 10:45 AM.
- Read the Office Hours content on Blackboard
- Use email if you don’t need a live interaction office hour

Setting up for Oracle Connection

- Oracle accounts and passwords are available in Blackboard



SQL

- SQL: Structured Query Language
- Designed specifically for communicating with databases
- SQL statements fit into three broad categories:
- DDL
- Data Definition Language
- DML
- Data Manipulation Language
- TCL
- Transaction Control Language

SQL

- Data definition language
  - SQL includes commands to create
    - Database objects such as tables
    - Commands to define access rights to those database objects
- Data manipulation language
  - Includes commands to insert, update, delete, and retrieve data within the database tables objects
- Transaction control language
  - Includes commands to ensure the integrity of the database. (Commit or Rollback)

SQL

- American National Standards Institute (ANSI) prescribes a standard SQL
- Several SQL dialects exist
  - DB2, Oracle, MySQL, Access etc

Joins

- PATIENT

PATIENTNO	FIRSTNAME	LASTNAME	PINSURNUM
123	Karen	Wong	555
456	Bill	Trimble	666
789	Fred	Hamer	888
246	John	Howard	0

- INSURANCE

INSNUM	INSNAME	INSPHONE
555	Manulife	7,056,663,344
666	Royal Insurance	4,167,774,444
888	Cut Rate Insurers	9,058,883,333
444	SureHealth Ins	6,132,225,555

Joins

- JOIN Example 1

PATIENTNO	FIRSTNAME	LASTNAME	PINSURNUM	INSNAME	INSPHONE
123	Karen	Wong	555	Manulife	7,056,663,344
456	Bill	Trimble	666	Royal Insurance	4,167,774,444
789	Fred	Hamer	888	Cut Rate Insurers	9,058,883,333

- JOIN Example 2

PATIENTNO	FIRSTNAME	LASTNAME	PINSURNUM	INSNAME	INSPHONE
123	Karen	Wong	555	Manulife	7,056,663,344
456	Bill	Trimble	666	Royal Insurance	4,167,774,444
789	Fred	Hamer	888	Cut Rate Insurers	9,058,883,333
-	-	-	-	SureHealth Ins	6,132,225,555

Joins

- JOIN Example 3

PATIENTNO	FIRSTNAME	LASTNAME	PINSURNUM	INSNAME	INSPHONE
123	Karen	Wong	555	Manulife	7,056,663,344
456	Bill	Trimble	666	Royal Insurance	4,167,774,444
789	Fred	Hamer	888	Cut Rate Insurers	9,058,883,333
246	John	Howard	0	Not Allocated Yet	-

- JOIN Example 4

PATIENTNO	FIRSTNAME	LASTNAME	PINSURNUM	INSNAME	INSPHONE
123	Karen	Wong	555	Manulife	7,056,663,344
456	Bill	Trimble	666	Royal Insurance	4,167,774,444
789	Fred	Hamer	888	Cut Rate Insurers	9,058,883,333
246	John	Howard	0	-	-
-	-	-	-	SureHealth Ins	6,132,225,555

Joins

- JOIN Example 5

PATIENTNO	FIRSTNAME	LASTNAME	PINSURNUM	INSNAME	INSPHONE
246	John	Howard	0	-	

- Which patients do not have insurance
- JOIN Example 6

PATIENTNO	FIRSTNAME	LASTNAME	PINSURNUM	INSNAME	INSPHONE
-	-	-	-	SureHealth Ins	6,132,225,555

- What insurance company does not insure any patients at our hospital

Referential Integrity

- No constraints have been included with the PATIENT and INSURANCE table creation
- WHAT HAS TO HAPPEN FOR REFERENTIAL INTEGRITY TO BE ENFORCED?
- Remove the patient row that refers to a non existent insurance company
- What is the second step on the path to enforcing referential integrity?
- Make the insurance number the primary key for the insurance table
- What is the third step?
- Create a foreign key for insurance number in the PATIENT table

Single Row Functions & Variables

- Character Functions
- Numeric Functions
- Datetime Functions
- Conversion Functions
- General Comparison Functions
- Assigning Variables

SQL Functions

- There are two types of functions:
- Single-row
- Multiple-row
- A single-row function returns one result for each row. These functions operate on single rows only and return one result for every row acted on.
- A multiple-row function returns one result per set of rows. Functions can manipulate groups of rows to give one result per group of rows. These functions are also called group functions.

SQL Functions

- Single-row functions return a single result for each row in the result set.
- Single-row functions can be used in
  - SELECT
  - WHERE
  - HAVING
  - ORDER BY
- Functions can be used to
  - Perform calculations on data
  - Modify individual data items
  - Manipulate output for groups of rows
  - Format dates and numbers for display
  - Convert column data types
- SQL functions may take arguments and always return a value.

Single Row Functions

- These functions manipulate data items
- Used in one or more arguments and return a single value for each row that is retrieved by the query
- An argument can be one of the following:
  - User supplied constant
  - Variable value
  - Column name
  - Expression

Single Row Functions

- The actions of a single row function include:
- Acts on each row that is returned by the query
- Returns one result per row
- May possibly return a different data type than the one that is referenced
- The function expects one or more arguments
- Let us look at Character Functions

Character Functions

- These functions accept character type arguments and return character or numeric values
- Character Functions
- LOWER
- UPPER
- INITCAP
- Character manipulation
- SUBSTR
- CONCAT
- LENGTH
- INSTR
- TRIM
- REPLACE

Character Manipulation Functions

Function	Returning Result
LOWER('DATABASE systems')	database systems
UPPER('DATABASE systems')	DATABASE SYSTEMS
INITCAP('DATABASE systems')	Database Systems

Character Manipulation Functions

- LOWER()
  - returns the character argument with all lower case letters.
- UPPER()
  - returns the character argument with all upper case letters.
- INITCAP()
  - returns each word with the first letter capital and other letters lower case.
  - Words are delimited by white space or characters that are not alphanumeric.

Character Manipulation Functions

Function	Returning Result
CONCAT('Database', 'Systems')	DatabaseSystems
SUBSTR ('DatabaseSystems',1,4)	Data
LENGTH('DatabaseSystems')	15
INSTR('DatabaseSystems', 'b')	5
LPAD('Tommy', 10, '*')	*****Tommy
RPAD('Tommy', 10, '*')	Tommy*****
REPLACE('Jack and Jue', 'J', 'Bl')	Black and Blue
TRIM('D' FROM 'Database')	**atabase**
NVL(<column>,<replacement>	Replaces NULL with <rep> in <col>

SUBSTR

PATIENTNO	FIRSTNAME	LASTNAME	PINSURNUM
123	Karen	Wong	555
456	Bill	Trimble	666
789	Fred	Hamer	888
246	John	Howard	0

```
SELECT PATIENTNO,
       SUBSTR(FIRSTNAME,1,2) FIRSTPART,
       SUBSTR(FIRSTNAME,3,4) SECONDPART,
       LASTNAME
FROM   PATIENT
```

PATIENTNO	FIRSTPART	SECONDPART	LASTNAME
123	Ka	ren	Wong
456	Bi	ll	Trimble
789	Fr	ed	Hamer
246	Jo	hn	Howard

- DB2 handles INITCAP

```
SELECT UPPER(SUBSTR(FIRSTNAME,1,1))   ||
       LOWER(SUBSTR(FIRSTNAME,2)) FIRSTNAME
FROM   PATIENT
```

```
FIRSTNAME
Karen
Bill
Fred
John
```

- But, only works on a single word

LENGTH

```
INSNUM   INSNAME                    INSPHONE
  555    Manulife                7,056,663,344
  666    Royal Insurance         4,167,774,444
  888    Cut Rate Insurers       9,058,883,333
  444    SureHealth Ins          6,132,225,555

SELECT  INSNUM,
        LENGTH(INSNAME)  NAMELENGTH,
        LENGTH(INSPHONE)  PHONELENGTH
FROM    INSURANCE

   INSNUM      NAMELENGTH      PHONELENGTH
     555              20                6
     666              20                6
     888              20                6
     444              20                6
```

WHY 6 FOR PHONE ?

- EBCDIC representation
- NUMERIC / ZONED
- F1 F1 F1 F1 F1 F1 F1 F1 F1 F1
- F1 broken down into bits is 1111 0001
- DECIMAL / PACKED
- 01 11 11 11 11 1F
- F1 displays as a 1 in all interfaces
- 11 doesn’t display very well in some interfaces

WHY 6 FOR PHONE ?

- EBCDIC representation

```
SELECT  INSNAME,
        HEX(INSPHONE) DECIMAL,
        HEX(INSPHONE2) ZONED
FROM    INSURANCE2
```

INSNAME	DECIMAL	ZONED
Manulife	07056663344F	F7F0F5F6F6F6F3F3F4F4
TD Insurance	04167774444F	F4F1F6F7F7F7F4F4F4F4
Cut Rate Insurers	09058883333F	F9F0F5F8F8F8F3F3F3F3
SureHealth Ins	06132225555F	F6F1F3F2F2F2F5F5F5F5

INSTR

- The INSTR function returns the starting position of a string (called the *search-string*) within another string (called the *source-string*). If the *search-string* is not found and neither argument is null, the result is zero.
- If the *search-string* is found, the result is a number from 1 to the actual length of the *source-string*.

```
PATIENTNO  FIRSTNAME        LASTNAME                    AN_A   PINSURNUM
   123     Karen            Wong                           0        555
   456     Bill             Trimble                        0        666
   789     Tom              Seaver                         3        888
   246     John             Howard                         4          0
********  End of data  ********
    SELECT PATIENTNO,
           FIRSTNAME,
           LASTNAME,
           INSTR(LASTNAME,'a') AN_a,
           PINSURNUM
    FROM   PATIENT
```

INSTR in a different clause

- We can move the function from the Select clause to a different clause
- Patients who have an a in their last name

```
SELECT PATIENTNO,
       FIRSTNAME,
       LASTNAME,
       PINSURNUM
FROM   PATIENT
WHERE INSTR(LASTNAME,'a') > 0
```

```
PATIENTNO  FIRSTNAME        LASTNAME          PINSURNUM
   789     Tom              Seaver               888
   246     John             Howard                 0
********  End of data  ********
```

LPAD

- The LPAD function returns a string that is padded on the left with a designated character.
- The LPAD function treats leading or trailing blanks in the column or *expression* supplied as significant. Padding will only occur if the actual length of *expression* is less than *length*, and *pad* is not an empty string.

PATIENTNO	LASTNAME	PINSURNUM	LPAD
123	Wong	555	*555
456	Trimble	666	*666
789	Seaver	888	*888
246	Howard	0	***0

```
SELECT PATIENTNO,
       LASTNAME,
       PINSURNUM,
       LPAD(PINSURNUM,4,'*')
FROM   PATIENT
```

RPAD

- Right pad up to 30 characters with * all the last names without an ‘a’ anywhere in the name

```
SELECT PATIENTNO,
       RPAD(LASTNAME,30,'*'),
       PINSURNUM
FROM   PATIENT
where instr(lastname,'a') = 0
```

```
PATIENTNO  RPAD                           PINSURNUM
   123     Wong           ***************     555
   456     Trimble        ***************     666
```

Fixing our RPAD

- You really wanted this:

```
PATIENTNO  RPAD                                PINSURNUM
   123     Wong****************************       555
   456     Trimble*************************       666
********  End of data  ********
```

- How can you achieve this?
- Use the Trim character manipulation function

```
SELECT PATIENTNO,
       RPAD(trim(LASTNAME),30,'*'),
       PINSURNUM
FROM   PATIENT
```

- The TRIM function removes blanks or another specified character from the end, from the beginning, or from both of a string expression

TRIM

- The first argument, if specified, indicates whether characters are removed from the end or beginning of the string. If the first argument is not specified, then the characters are removed from both the end and the beginning of the string.
- TRIM (<BOTH><LEADING><TRAILING> strip character FROM argument

```
SELECT PATIENTNO,
         trim(leading 'W' from LASTNAME),
       PINSURNUM
FROM   PATIENT
```

PATIENTNO	TRIM function	PINSURNUM
123	ong	555
456	Trimble	666
789	Seaver	888
246	Howard	0

REPLACE

- The REPLACE function replaces all occurrences of search-string in source-string with replace-string. If search-string is not found in source-string, source-string is returned unchanged.

```
SELECT PATIENTNO,
       replace(firstname,'o','?'),
       replace( lastname,'n','$#'),
       PINSURNUM
FROM   PATIENT
```

PATIENTNO	REPLACE	REPLACE	PINSURNUM
123	Karen	Wo$#g	555
456	Bill	Trimble	666
789	T?m	Seaver	888
246	J?hn	Howard	0

Numeric Functions

Function	Returning Result
ROUND(5.678, 2)	5.68
TRUNC(5.678, 2)	5.67
MOD(10, 3)	1

ROUND(v, n)

It receives two arguments

v is a value of any numeric type

n is an integer value

returns the argument value v rounded to n places to the right of the decimal point.
If you use 0 or no value for the second argument, n is rounded to 0 decimal places
ROUND(22.96) → 23

ROUND()

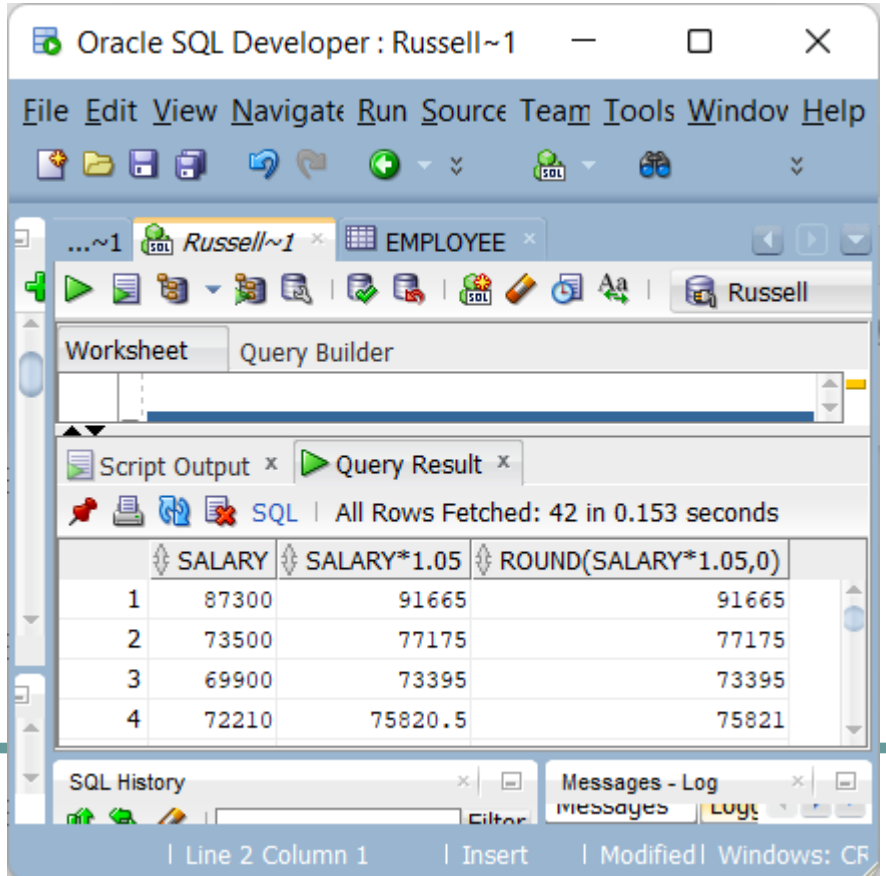
- DB2

```
SELECT SALARY,
       SALARY * 1.05,
       ROUND(SALARY * 1.05,1)
FROM EMPLOYEE
```

SALARY	SALARY * 1.05	ROUND
72,750.00	76,387.5000	76,387.5000
61,250.00	64,312.5000	64,312.5000
58,250.00	61,162.5000	61,162.5000
60,175.00	63,183.7500	63,183.8000

ROUND (ORACLE)

```
SELECT SALARY,
       SALARY * 1.05,
       ROUND(SALARY * 1.05,0)
FROM EMPLOYEE
```



TRUNC()

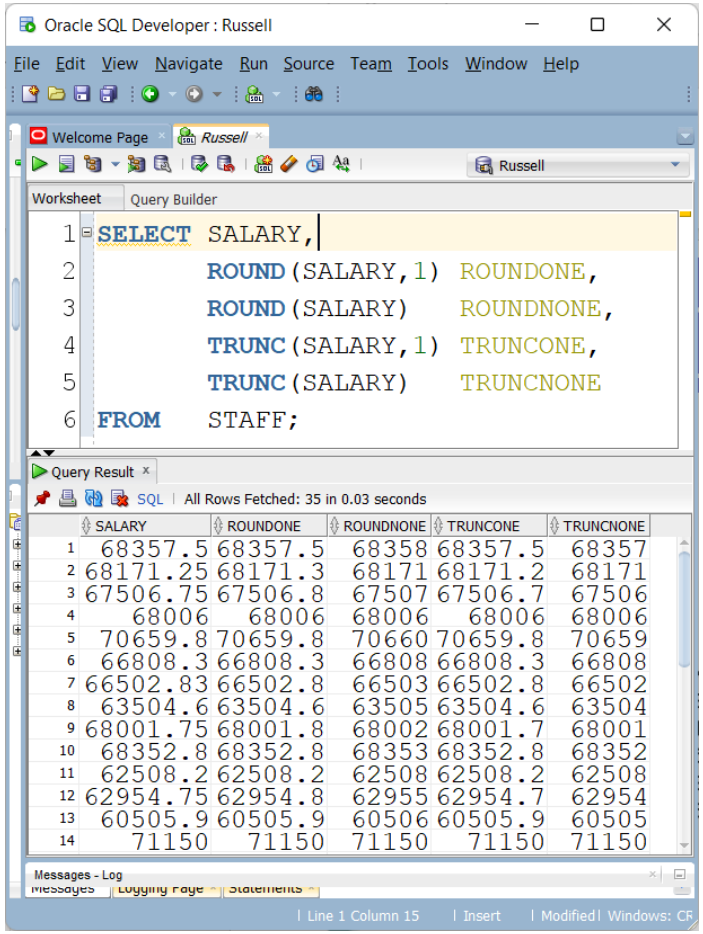
- TRUNC(v,n)
- truncates a number v to n decimal places
- TRUNC(15.193, 2) → 15.19
- TRUNC(15.193, 3) → 15.193
- TRUNC(15.193, 1) →15.1
- TRUNC(n)
- truncate a number n to zero decimal places.
- TRUNC(15.193) → 15

Number Functions

```
select salary,
       round(salary,1) roundone,
       round(salary) roundnone,
       trunc(salary,1) truncone,
       trunc(salary) truncnone
from staff
```

SALARY	ROUNDONE	ROUNDNONE	TRUNCONE	TRUNCNONE
68,357.50	68,357.50	68,358.00	68,357.50	68,357.00
68,171.25	68,171.30	68,171.00	68,171.20	68,171.00
67,506.75	67,506.80	67,507.00	67,506.70	67,506.00
68,006.00	68,006.00	68,006.00	68,006.00	68,006.00
70,659.80	70,659.80	70,660.00	70,659.80	70,659.00

Using Oracle



MOD()

- The MOD function divides the first argument by the second argument and returns the remainder
- MOD(121,14) → 9
- MOD(25, 7) → 4
- The MOD() function is used to determine if a number is odd or even

MOD()

```
SELECT MOD(156,7) REMAINDER
FROM   SYSIBM/SYSDUMMY1

  REMAINDER
          2

SELECT 156/7
FROM   SYSIBM/SYSDUMMY1


   156 / 7
        22
```

Datetime Functions

- DATES
- The default display and input format for any date is DD-MON-RR. Valid Oracle dates are between January 1, 4712 B.C., and December 31, 9999 A.D. • In the example in the slide, the HIRE_DATE column output is displayed in the default format DD-MON-RR. However, dates are not stored in the database in this format. All the components of the date and time are stored. So, although a HIRE_DATE such as 17-JUN-87 is displayed as day, month, and year, there is also time and century information associated with the date. The complete data might be June 17, 1987, 5:10:43 p.m.
- CENTURY YEAR MONTH DAY HOUR MINUTE SECOND
- 19 87 06 17 17 10 43 • Note: century or year stored as 4 digits even if displayed as 2

ORACLE USES SYSDATE

- SYSDATE returns current
  - Date
  - Time
- SELECT SYSDATE
- FROM DUAL
  - 22-09-11
  - 11-SEP-22
- Since Oracle database stores dates as numbers, arithmetic operations such as addition or subtraction can be performed on date values. You can add or subtract both numbers and dates to or from date values.
  - Date + Number
  - Date – Number
  - Date – Date

DB2

- SELECT CURRENT DATE FROM STAFF

```
CURRENT DATE
  09/01/25
  09/01/25
  09/01/25
  09/01/25
```

- SELECT CURRENT TIMESTAMP FROM STAFF

```
CURRENT TIMESTAMP
2025-09-01-18.30.57.915706
2025-09-01-18.30.57.915706
2025-09-01-18.30.57.915706
2025-09-01-18.30.57.915706
2025-09-01-18.30.57.915706
```

- FETCH FIRST ROW ONLY

```
CURRENT TIMESTAMP
2025-09-01-18.31.36.159594
```

Date Arithmetic

- DB2

```
SELECT CURRENT DATE,
       CURRENT DATE + 30 DAYS  DATEPLUS30
FROM   SYSIBM/SYSDUMMY1

CURRENT DATE  DATEPLUS30
  09/01/25      10/01/25
```

- ORACLE

```
SELECT SYSDATE, SYSDATE + 30
FROM   DUAL
```

Date Arithmetic

- DB2 – how many years has an employee been at company

```
  CURRENT DATE   HIREDATE   YEARSOFSERVICE
    09/01/25     01/01/95                30
    09/01/25     10/10/03                22
    09/01/25     04/05/05                20
    09/01/25     08/17/79                46
    09/01/25     09/14/03                22
    09/01/25     09/30/10                15
```

```
SELECT CURRENT DATE,
       HIREDATE,
       YEAR(CURRENT DATE) - YEAR(HIREDATE) YEARSOFSERVICE
FROM   DBS311W25.EMPLOYEE
```

Date Arithmetic and DayName

- DB2 – How old is an employee

```
CURRENT DATE  BIRTHDATE              AGE   DAYOFBIRTH
  09/01/25    08/24/63                62    born on a Saturday
  09/01/25    02/02/78                47    born on a Thursday
  09/01/25    05/11/71                54    born on a Tuesday
  09/01/25    09/15/55                70    born on a Thursday
  09/01/25    07/07/75                50    born on a Monday
  09/01/25    05/26/83                42    born on a Thursday
  09/01/25    05/15/71                54    born on a Saturday
```

```
SELECT CURRENT DATE,
       BIRTHDATE,
       YEAR(CURRENT DATE) - YEAR(BIRTHDATE) AGE,
       ' born on a ' || DAYNAME(BIRTHDATE) DAYOFBIRTH
FROM   DBS311W25.EMPLOYEE
```

Lab 1

- Lab 1 and Powerpoint already uploaded.