# Introduction to Databases

(INFR10080)

## (Other Bits of SQL)

Instructor: Yang Cao

(Fall 2025)

THE UNIVERSITY *of* EDINBURGH

## Changelog

v25.0 Initial version

# Ordering

ORDER BY ⟨column$_1$⟩ [DESC], . . . , ⟨column$_n$⟩ [DESC]

Sorts the output rows according to the values of column$_1$

If two rows have the same value for column$_1$,
they are sorted by the values of column$_2$ and so on …

- Default ordering is **ascending** (can be specified with ASC)
- **Descending** ordering is specified by DESC



# Ordering example (1)

Account

| Number | Branch | CustID | Balance |
|---|---|---|---|
| 111 | London | 1 | 1330.00 |
| 222 | London | 2 | 1756.00 |
| 333 | Edinburgh | 1 | 450.00 |

```
SELECT   *
FROM     Account
ORDER BY custid ASC, balance DESC ;
```

| Number | Branch | CustID | Balance |
|---|---|---|---|
| 111 | London | 1 | 1330.00 |
| 333 | Edinburgh | 1 | 450.00 |
| 222 | London | 2 | 1756.00 |

# Ordering example (2)

Account

| Number | Branch | CustID | Balance |
|---|---|---|---|
| 111 | London | 1 | 1330.00 |
| 222 | London | 2 | 1756.00 |
| 333 | Edinburgh | 1 | 450.00 |

```
SELECT   *
FROM     Account
ORDER BY custid DESC, balance ASC ;
```

| Number | Branch | CustID | Balance |
|---|---|---|---|
| 222 | London | 2 | 1756.00 |
| 333 | Edinburgh | 1 | 450.00 |
| 111 | London | 1 | 1330.00 |



# Casting

CAST( term AS ⟨type⟩ )

## Rounding

CAST( 102.4675 AS NUMERIC(5,2) ) gives 102.47
Useful also to produce values in a specific format

## Aggregation

AVG( CAST( term AS NUMERIC($p$, $s$) ) )
avoids rounding errors in some systems

# Conditional expressions (1)

```
CASE WHEN ⟨bool-expr⟩
     THEN ⟨value-expr⟩
     ...
     WHEN ⟨bool-expr⟩
     THEN ⟨value-expr⟩
     ELSE ⟨value-expr⟩
END
```

- Each bool-expr is evaluated in order (from top to bottom)
- When bool-expr is true, then the value produced by the corresponding value-expr is returned and the CASE ends (subsequent WHEN are not evaluated)
- If no WHEN evaluates to true, the value produced by value-expr in ELSE is returned
- ELSE is optional (if missing, the default value is NULL)



# Conditional expressions (2)

Can be used in SELECT, WHERE and HAVING

Return the values of column *A* replacing NULL values with 0

```
SELECT CASE WHEN R.A IS NULL THEN 0
            ELSE R.A END
FROM R ;
```

Join *R* and *S* on column *A* (don't do this!)

```
SELECT *
FROM   R, S
WHERE  CASE WHEN R.A = S.A THEN TRUE
            ELSE FALSE END ;
```

# Pattern matching

**New comparison:** term LIKE pattern

where pattern is a string consisting of

characters (case-sensitive!)

_ (underscore) wildcard matching any one character

% (percent) wildcard matching any substring (including empty)

## Example

Customers with a name that **begins with 'K'**
and has **at least 5 characters**

```
SELECT *
FROM   Customer
WHERE  name LIKE 'K____%' ;
```