Select Single Column Ordering
List salaries for all staff, arranged in descending order of salary.

```
SELECT staffNo, fName, lName, salary
FROM Staff
ORDER BY salary DESC;
```

Select Multiple Column Ordering
Produce abbreviated list of properties in order of property type.

```
SELECT propertyNo, type, rooms, rent
FROM PropertyForRent
ORDER BY type;
```

To arrange in order of rent, specify minor order:

```
SELECT propertyNo, type, rooms, rent
FROM PropertyForRent
ORDER BY type, rent DESC;
```

SELECT Statement - Aggregates
•ISO standard defines five aggregate functions:
COUNT returns number of values in specified column.
SUM returns sum of values in specified column.
AVG returns average of values in specified column.
MIN returns smallest value in specified column.
MAX returns largest value in specified column.
•Each operates on a single column of a table and returns a single value.
•COUNT, MIN, and MAX apply to numeric and non-numeric fields, but SUM and AVG may be used on numeric fields only.
•Apart from COUNT(*), each function eliminates nulls first and operates only on remaining non-null values.
•COUNT(*) counts all rows of a table, regardless of whether nulls or duplicate values occur.
•Can use DISTINCT before column name to eliminate duplicates.
•DISTINCT has no effect with MIN/MAX, but may have with SUM/AVG.
•Aggregate functions can be used only in SELECT list and in HAVING clause.
•If SELECT list includes an aggregate function and there is no GROUP BY clause, SELECT list cannot reference a column out with an aggregate function. For example, the following is illegal

```
SELECT staffNo, COUNT(salary)
 FROM Staff;
```

Select - Use of COUNT(*)
How many properties cost more than £350 per month to rent?

```
SELECT COUNT(*) AS myCount
FROM PropertyForRent
WHERE rent > 350;
```

Select - Use of COUNT(DISTINCT)
How many different properties viewed in May ‘13?

```
SELECT COUNT(DISTINCT propertyNo) AS myCount
FROM Viewing
WHERE viewDate BETWEEN ‘1-May-13’
          AND ‘31-May-13’;
```

Select - Use of COUNT and SUM

Find number of Managers and sum of their salaries.

```
  SELECT COUNT(staffNo) AS myCount, SUM(salary) AS mySum
  FROM Staff
  WHERE position = ‘Manager’;
```

Select - Use of MIN, MAX, AVG

Find minimum, maximum, and average staff salary.

```
      SELECT MIN(salary) AS myMin,
      MAX(salary) AS myMax,
      AVG(salary) AS myAvg
      FROM Staff;
```

SELECT Statement - Grouping

•Use GROUP BY clause to get sub-totals.
•SELECT and GROUP BY closely integrated: each item in SELECT list must be *single-valued per group*, and SELECT clause may only contain:
•column names
•aggregate functions
•constants
•expression involving combinations of the above.
•All column names in SELECT list must appear in GROUP BY clause unless name is used only in an aggregate function.
•If WHERE is used with GROUP BY, WHERE is applied first, then groups are formed from remaining rows satisfying predicate.
•ISO considers two nulls to be equal for purposes of GROUP BY.

Find number of staff in each branch and their total salaries.

```
 SELECT branchNo,
 COUNT(staffNo) AS myCount,
 SUM(salary) AS mySum
 FROM Staff
 GROUP BY branchNo
 ORDER BY branchNo;
```

Restricted Groupings – HAVING clause

•HAVING clause is designed for use with GROUP BY to restrict groups that appear in final result table.
•Similar to WHERE, but WHERE filters individual rows whereas HAVING filters groups.
•Column names in HAVING clause must also appear in the GROUP BY list or be contained within an aggregate function.
For each branch with more than 1 member of staff, find number of staff in each branch and sum of their salaries.

```
  SELECT branchNo,
               COUNT(staffNo) AS myCount,
     SUM(salary) AS mySum
FROM Staff
GROUP BY branchNo
HAVING COUNT(staffNo) > 1
```

ORDER BY branchNo;

Subqueries
•Some SQL statements can have a SELECT embedded within them.
•A subselect can be used in WHERE and HAVING clauses of an outer SELECT, where it is called a *subquery* or *nested query*.
•Subselects may also appear in INSERT, UPDATE, and DELETE statements.

Select - Subquery with Equality
List staff who work in branch at '163 Main St'.

```
  SELECT staffNo, fName, lName, position
  FROM Staff
  WHERE branchNo =
  (SELECT branchNo
   FROM Branch
   WHERE street = '163 Main St');
```

•Inner SELECT finds branch number for branch at '163 Main St' ('B003').
•Outer SELECT then retrieves details of all staff who work at this branch.
•Outer SELECT then becomes:

```
  SELECT staffNo, fName, lName, position
FROM Staff
WHERE branchNo = 'B003';
```

Select - Subquery with Aggregate
List all staff whose salary is greater than the average salary, and show by how much.

```
SELECT staffNo, fName, lName, position,
  salary – (SELECT AVG(salary) FROM Staff) As SalDiff
FROM Staff
WHERE salary >
  (SELECT AVG(salary)
   FROM Staff);
```

•Cannot write 'WHERE salary > AVG(salary)'
•Instead, use subquery to find average salary (17000), and then use outer SELECT to find those staff with salary greater than this:

```
SELECT staffNo, fName, lName, position,
     salary – 17000 As salDiff
FROM Staff
WHERE salary > 17000;
```

Subquery Rules
•ORDER BY clause may not be used in a subquery (although it may be used in outermost SELECT).
•Subquery SELECT list must consist of a single column name or expression, except for subqueries that use EXISTS.
•By default, column names refer to table name in FROM clause of subquery. Can refer to a table in FROM using an *alias*.
•When subquery is an operand in a comparison, subquery must appear on right-hand side.
•A subquery may not be used as an operand in an expression.

Select - Nested subquery: use of IN
List properties handled by staff at '163 Main St'.

```
SELECT propertyNo, street, city, postcode, type, rooms, rent
FROM PropertyForRent
WHERE staffNo IN
(SELECT staffNo
 FROM Staff
 WHERE branchNo =
  (SELECT branchNo
   FROM Branch
   WHERE street = '163 Main St'));
```

ANY and ALL

•ANY and ALL may be used with subqueries that produce a single column of numbers.
•With ALL, condition will only be true if it is satisfied by *all* values produced by subquery.
•With ANY, condition will be true if it is satisfied by *any* values produced by subquery.
•If subquery is empty, ALL returns true, ANY returns false.
•SOME may be used in place of ANY.

Select - Use of ANY/SOME

Find staff whose salary is larger than salary of at least one member of staff at branch B003.

```
     SELECT staffNo, fName, lName, position, salary
  FROM Staff
  WHERE salary > SOME
  (SELECT salary
   FROM Staff
   WHERE branchNo = 'B003');
```

Use of ALL

Find staff whose salary is larger than salary of every member of staff at branch B003.

```
         SELECT staffNo, fName, lName, position, salary
  FROM Staff
  WHERE salary > ALL
  (SELECT salary
   FROM Staff
   WHERE branchNo = 'B003');
```