6 Functions, Aggregate and Group-set Functions

This chapter will show you how to create functions, manipulate and extract certain information from the database using more advanced SQL queries.

6.1 Functions

Functions fall into four categories:

- String functions
- Arithmetic functions
- Date functions
- Aggregate or group-set functions

Many string functions work on arithmetic expressions and dates, and then automatically convert them to a string format. Note that Microsoft products typically use operators and functions from Visual Basic rather than the SQL standard.

6.1.1 String Functions

String concatenation: string1 & string2

Joins string1 with string2. Note that the AS command is needed to produce a sensible column name. Different symbols are used for differing product: & (Microsoft) or || (SQL standard).

```
SELECT fname & „ „ & lname AS combined
FROM owner;
```

combined
Tina Murphy
Joe Keogh
Carol Farrel
Tony Shaw

Length: Length(string) or Len(string) (Microsoft)

Determines the length of a string e.g. this will tell you how many characters are in the lname – MURPHY has 6 letters in it.

```
SELECT lname, len(lname)
FROM owner;
```

lname	Expr1001
Murphy	6
Keogh	5
Farrel	6
Shaw	4

Substring: substr(string, x, y) or mid(string, x, y) (Microsoft)

Extracts a string of length y from the string starting at position x.

E.g.: mid("hello",2,3) would return "ell". This starts at letter 2 e.g. E and then takes 3 letters from there in total.

```
SELECT lname, Mid(lname,2,3)
FROM owner;
```

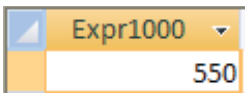
lname	Expr1001
Murphy	urp
Keogh	eog
Farrel	arr
Shaw	haw

6.1.2 Arithmetic Functions

`Greatest value`: greatest(columnname) or max(columnname) (Microsoft)

Returns the greatest value in a column.

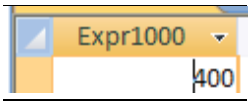
```
SELECT max(rent)
FROM property;
```



`Least value: least(columnname) or min(columnname) (Microsoft)`

Returns the least value in a column.

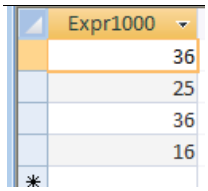
```
SELECT min(rent)
FROM property;
```



`Power: power(x,y) or x^y (Microsoft)`

Returns x to the power y. In this case it is to the power of 2. If we use MURPHY, it has 6 characters and to the power of 2 means times 6 × 6 and will return 36. If it was to the power of 3, it would be 6 × 6 × 6.

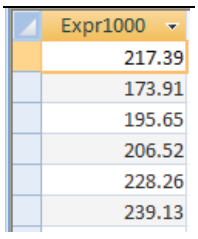
```
SELECT len(lname)^2
FROM owner;
```



`Round: round(x,y)`

Rounds the number x to y places (not available in Access 97). The below example will divide rent by 2.3 and return only 2 places after the decimal point.

```
SELECT round(rent/2.3, 2)
FROM property;
```



6.1.3 Date Functions

To increase or decrease a date: Use + or – a number of days.

```
SELECT bookingdate+20
FROM booking;
```

Expr1000
05/08/2007
29/07/2007
06/01/2008
23/09/2007
30/09/2007

Subtract one date from another: Use `date – date`.

6.2 Aggregate Functions

Aggregate functions operate on a number of rows (for example a whole table).

Calculate an average: `avg (x)`, where x is generally a column.

```
SELECT avg(rent)
FROM property;
```

Expr1000
483.333333

Calculate the **total** of numbers in a column: `sum (x)`, where x is generally a column.

```
SELECT sum(rent)
FROM property;
```

Expr1000
2900

Count a number of items: `count(x)`, where x is a column.

```
SELECT count(rent)
FROM property;
```

Expr1000
6

6.2.1 Grouping with Aggregate Functions

Aggregate functions may be applied to several subsets of the table. The `GROUP BY` clause divides (sorts) a table into groups of rows with matching values in the same column (or columns).

Example: Calculate the average rent for each different type of property

```
SELECT type, avg(rent)
FROM property
GROUP BY type;
```

type	Expr1001
Apartment	425
Studio	475
Villa	525

Example: Calculate the average rent for each different type of property

```
SELECT type, max(rent)
FROM property
WHERE type<>"Studio"
GROUP BY type;
```

type	Expr1001
Apartment	450
Villa	550

6.3 Nested Sub-queries

When one of the conditions of a `WHERE` clause is a query itself, this is called a nested sub-query, e.g.,

```
SELECT select-list
FROM table(s)
WHERE object operator (SELECT select-list
                       FROM table(s)
                       [WHERE condition]);
```

The sub-query must be enclosed in ( ) , as indeed must any levels of nested sub-query.

Example:

To find the owner number who charges the lowest rental figure, the query must first calculate the lowest rent and then find the person who has the lowest rental figure.

This is a consequence of not being able to place both non-aggregated columns and aggregate functions in the same select list.

```
SELECT propertyno, rent, ownerno
FROM property
WHERE rent=(select min(rent) from property);
```

propertyno	rent	ownerno
PG16	400	CO87

The nested query is executed first and the value retrieved is 'substituted' into the `WHERE` clause.

Note that sub-queries may be nested to any number of levels e.g.,

```
SELECT select-list
FROM table(s)
WHERE object operator (SELECT select-list
                       FROM table(s)
                       [WHERE object operator
                             (SELECT select-list
                                   FROM table(s)
                                   [WHERE condition]]);
```

6.4 Exercises

1. Display the details for the latest booking date

SQL:

clientno	propertyno	bookingdate	comment
CR56	PG4	17/12/2007	

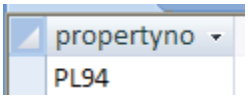
2. Display the lowest yearly income earned by properties in Barbados

SQL:

Expr1000
11075

3. Display the property number with the highest year income for a Villa

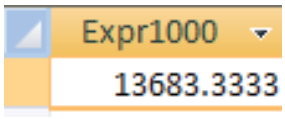
SQL:



propertyno
PL94

4. Find the average year income for Villas

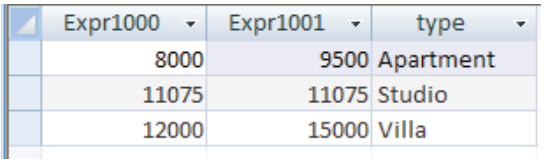
SQL:



Expr1000
13683.3333

5. List the minimum and maximum year incomes for each property type

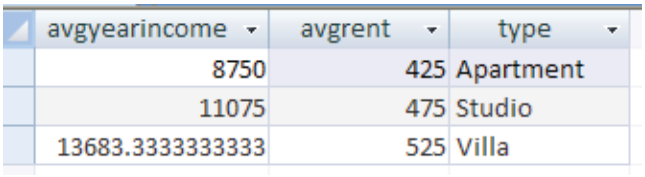
SQL:



Expr1000	Expr1001	type
8000	9500	Apartment
11075	11075	Studio
12000	15000	Villa

6. Find the average year income (name the new column `avgyearincome`) and average rent (name the new column `avgrent`) for each property type, grouped by property type

SQL:



avgyearincome	avgrent	type
8750	425	Apartment
11075	475	Studio
13683.3333333333	525	Villa

6.5 Summary

In this chapter you learnt how to create functions, manipulate and extract certain information from the database using more advanced SQL queries. Generally speaking functions used in SQL fall into four categories:

- String functions, such as `len(..), &, substr(..), mid(…) , etc.`
- Arithmetic functions, such as `round(..), sum(..),etc.`
- Date functions, such as + and – operations on dates
- Aggregate or group-set functions, such as `count(), avg(), etc.`

The `GROUP BY` clause divides (sorts) a table into groups of rows with matching values in the same column (or columns).