RELATIONAL DATABASES

SELECT

OBJECTIVES

- Anatomy of an SQL statement
- Arithmetic in the SELECT CLAUSE
- NULL values in arithmetic
- Column Aliases
- Concatenation
- Construct query to sort a result set in ascending or descending order
- Construct a query to order a result set using a column alias
- Construct a query to order a result set for single or multiple columns

ANATOMY OF A SQL STATEMENT

- SELECT is one of the most important, if not the most important keyword in SQL.
- SELECT allows you to search for specific data in a database
- The SELECT statement must contain a SELECT clause and a FROM clause
- The list of columns in a SELECT clause allows you to conduct projection i.e. columns in a table.
- The WHERE clause allows you to conduct selection i.e. rows in a table.
- SELECT * FROM tablename means that you want to see all columns from a table

ARITHMETIC IN SELECT CLAUSE

- You can construct a SELECT clause that contains arithmetic
- You may want to modify the way data is displayed, perform calculations etc
- We are not creating new columns for these calculations or changing the data in the database.
- The results appear only in the output

```
SELECT last_name, salary, salary + 300
FROM employee;
```

ARITHMETIC IN THE SELECT CLAUSE

- Precedence is the order in which the database management system evaluates the operators in the same expression.
- Oracle evaluates operators with higher precedence first * / + -
- Oracle evaluates operators with equal precedence from left to right within an expression.
- You can use parentheses to force the expression within the parentheses to be evaluated first.

```
SELECT last_name, salary, 12*salary + 100 FROM employees;

SELECT last_name, salary, 12*(salary +100) FROM employees;
```

NULL VALUES IN ARITHMETIC

- In SQL, NULL is not zero or space, In SQL, zero is a number and space is a character.
- If any column value in an arithmetic expression is null, the result is null.
- If you try to divide by a null value, the result is null.
- If you try to divide by zero you get an error.

```
SELECT last_name, job_id, salary, commission_pct,
      salary*commission_pct
FROM employees;
```

- Where there was a null value in commission_pct would result in a null value in the last column

COLUMN ALIASES

- An Alias is a way of renaming a column heading in the output.
- Without aliases, when the result of a SQL statement is displayed, the name of the columns displayed will be the same as the column names in the table or a name showing an arithmetic operation such as 12*(salary +100)
- You will want your output to display in a more user friendly way

COLUMN ALIASES

- A column alias:
  - Renames a column heading
  - Is useful with calculations
  - Immediately follows the column name in the SELECT clause
  - May have the optional AS keyword between the column name and alias
  - Requires double quotation marks if the alias contains spaces or special characters, or is case-sensitive

COLUMN ALIASES

```
SELECT * | column | expr [AS alias], …
FROM tablename;

SELECT last_name AS name,
       commission_pct AS commission
FROM employees;

SELECT last_name “Name”,
       commission_pct “Commission Percentage”
FROM employees;
```

CONCATENATION

- Concatenation means to connect or link together in a series.
- MySQL has a concat() function
- Values passed into the function are combined to make a single output column
- Syntax:

  concat(expr1,expr2, exprn..)

- Concatenation is used to produce readable data output

```
SELECT CONCAT(department_id,' ',department_name)
FROM departments;
```

CONCATENATION AND ALIASES

- Column aliases are useful when using the concatenation operator to ensure the heading is readable

```
SELECT CONCAT(department_id,‘ ‘,department_name) AS “Department Info”
FROM departments;
```

CONCATENATION AND LITERAL VALUES

- A literal value is a fixed data value such as a character, number or date.
- ‘dollars’ 1000 ‘January 1, 2009’ (number do not need quotes)
- You can create output that looks like a sentence or statement.

```
SELECT CONCAT(last_name,’ has a monthly salary of ‘,salary,’
dollars.’) AS Pay
FROM employees;
```

DISTINCT

- You will want to eliminate duplicate rows
- For example if you select all the department id's from the employees table it will output many rows that are the same department id
- If you want to just see one row for each unique department id then you use DISTINCT

```
SELECT DISTINCT department_id
FROM employees;
```

- DISTINCT affects all listed columns, returning rows that a unique across all columns. The keyword must appear first in SELECT clause

ORDER BY CLAUSE

- Information sorted ascending order is familiar to us.
- It's what makes looking up a number in a phone book, finding a word in a dictionary, or locating a house by its street address relatively easy.
- SQL uses the ORDER BY clause to order data.
- The ORDER BY clause can specify several ways in which to order rows returned in a query.

ORDER BY CLAUSE

- The default sort order is ascending.
- Numeric values are displayed lowest to highest.
- Date values are displayed with the earliest value first
- Character values are displayed in alphabetical order
- Null values are displayed last in ascending order and first in descending order
- NULLS FIRST specifies that NULL values should be returned before non-NULL values.
- NULLS LAST specifies that NULL values should be returned after non-NULL values.
- You can sort by more than one column (separate with commas).

ORDER BY CLAUSE

```
SELECT last_name, hire_date

FROM employees

ORDER BY hire_date;
```

LAST_NAME	HIRE_DATE
King	17/Jun/1987
Whalen	17/Sep/1987
Kochhar	21/Sep/1989
Hunold	03/Jan/1990
Ernst	21/May/1991
De Haan	13/Jan/1993
Gietz	07/Jun/1994
Higgins	07/Jun/1994
Rajs	17/Oct/1995
Hartstein	17/Feb/1996

ORDER BY CLAUSE

```
SELECT last_name, hire_date

FROM employees

ORDER BY hire_date DESC;
```

LAST_NAME	HIRE_DATE
Zlotkey	29/Jan/2000
Mourgos	16/Nov/1999
Grant	24/May/1999
Lorentz	07/Feb/1999
Vargas	09/Jul/1998
Taylor	24/Mar/1998
Matos	15/Mar/1998
Fay	17/Aug/1997
Davies	29/Jan/1997
Abel	11/May/1996

ORDER BY CLAUSE

```
SELECT last_name, hire_date AS “Date Started”
FROM employees
ORDER BY “Date Started”;
```

LAST_NAME	Date Started
King	17/Jun/1987
Whalen	17/Sep/1987
Kochhar	21/Sep/1989
Hunold	03/Jan/1990
Ernst	21/May/1991
De Haan	13/Jan/1993
Gietz	07/Jun/1994
Higgins	07/Jun/1994
Rajs	17/Oct/1995
Hartstein	17/Feb/1996

ORDER BY CLAUSE

```
SELECT employee_id, first_name
FROM employees
WHERE employee_id < 105
ORDER BY last_name;
```

EMPLOYEE_ID	FIRST_NAME
102	Lex
104	Bruce
103	Alexander
100	Steven
101	Neena

ORDER OF EXECUTION

- The order of execution of a SELECT statement is as follows:
  - FROM clause: locates the table that contains the data
  - WHERE clause: restricts the rows to be returned
  - SELECT clause: selects from the reduced data set the columns requested
  - ORDER BY clause: orders the result set

PRACTICE

- Write a SELECT statement that outputs the following:

Partner Name	Area of Expertise	Expense Amount
Jennifer cho	Weddings	
Jason Tsang		
Allison Plumb	Event Planning	30000

- The table is D_Partners with columns: first_name, last_name, expertise, and auth_expense_amt, order the output starting with the largest expense amount.