SQL plus Hints and Helps:

Change the Width of the Output to Stop the Line Wrapping Problem:

- Use the Options menu from SQLplus to set the Environment variable '`linesize`' to a width that will stop the lines from wrapping. Try 200 as a starting point.
- You can also do this from the `SQL>` prompt by typing:

```
set linesize 200
```

A Column is Too Wide:

- Use the COLUMN command to set up the width of a column with FORMAT. To set the width of a column to 15 characters:

```
COLUMN column_name FORMAT a15
```

- You can also set the column to wrap on whole words by using:

```
COLUMN column_name word_wrapped
```

- To truncate a column to a certain width with no wrapping at all:

```
COLUMN column_name truncated
```

- The named column can either be a column from a table or an alias you intend to use. All `COLUMN` commands only last for a single session. So if you quit SQLplus and restart, the `COLUMN` commands will have to be reissued.

A Concatenated Column is Still Too Wide:

- Use the `RTRIM()` function to eliminate the trailing spaces from the columns that are being concatenated together.

```
SELLECT RTRIM(emp_lname)||', '||RTRIM(emp_fname) as Name
FROM employee;
```

To Format a Number Column:

- Use the `COLUMN` command to set the format to the desired size and numeric look. To set a numeric column to 9 characters wide with fixed two decimals and commas at the thousands location use:

```
COLUMN column_name FORMAT 99,999.00
```

- To put a dollar sign at the beginning use:

```
COLUMN column_name FORMAT $99,999.00
```

To Extract a Portion of a Value from a Column:

- Use the SUBSTR() function. To extract the first three characters use:

```
SELECT SUBSTR('ABCDEF',1,3) FROM DUAL;
```
Returns: `ABC`

- To extract a string from the middle the first argument is the name of the column, the second argument is the starting position, and the third argument is the number of characters. If a third argument is not given then the sub-string will start at the given position and encompass the rest of the column.

```
SELECT SUBSTR('ABCDEF',3) FROM DUAL;
```
Returns: `CDEF`

To Join Two or More Tables Together:

- You must supply a properly formulated `WHERE` clause that shows how the join is to be made:

```
SELECT sales_order.id, emp_fname, emp_lname
FROM sales_order, employee
WHERE sales_rep = emp_id
ORDER BY sales_order.id;
```

- When more than two tables are joined the `WHERE` clauses must use AND:

```
SELECT sales_order.id, emp_fname, emp_lname, company_name
FROM sales_order, employee, customer
WHERE sales_rep = emp_id AND
      cust_id = customer.id
ORDER BY sales_order.id;
```

To Create a View:

- A VIEW is basically a named query. Once created it can be used like any other table with the exception of some update operations. The VIEW is based on a SQL SELECT command. To create a VIEW that limits the amount of personal information available to an HR Clerk:

```
CREATE VIEW Current_Employee AS
SELECT emp_id, emp_fname, emp_lname, dept_id,
      street, city, state, zip_code, phone, birth_date
FROM employee
WHERE termination_date IS NULL;
```

- Once a view is created you use the name of the view just like you would any other table:

```
SELECT emp_id, emp_fname, dept_id
FROM Current_Employee;
```

To Obtain Single Rows That Represent a Group of Rows:

- Use the `GROUP BY` clause to cause the `SELECT` to produce one summary row for each group of rows. To obtain the average salary and number of employees in each department:

```
SELECT dept_id, AVG(salary), COUNT(emp_id)
FROM employee
GROUP BY dept_id
ORDER BY dept_id
```

- Use `WHERE` to form any needed joins, and to determine which rows will be selected into the groups. Use `HAVING` to control which groups are to be included. To create a limited list of sales order values:

```
SELECT soi.id, SUM(soi.quantity * unit_price) value
FROM sales_order_items AS soi, product
WHERE prod_id = product.id
GROUP BY soi.id
HAVING soi.id BETWEEN 2625 AND 2629
```

- Anytime you put other columns that are not constants, or group functions (`AVG`, `MIN`, `MAX`, etc.) in the `SELECT` clause those columns will also have to appear in the `GROUP BY` clause as well. To include the customer information with the value of the order:

```
SELECT soi.id as sales_order#,
      SUM(soi.quantity * unit_price) value, company_name
FROM sales_order_items AS soi, sales_order, product, customer
WHERE prod_id = product.id AND
      soi.id = sales_order.id AND
      cust_id = customer.id
GROUP BY soi.id, company_name
HAVING soi.id BETWEEN 2625 AND 2629
ORDER BY soi.id
```