# Views and Joins

Intro to Views:

Sometimes in practice the user of a database may want to reference data regularly from different tables and see this data on screen at the same time. This can be achieved using the select statement but if the same data is being accessed regularly then why not create a table with that data that the user wants as columns in the table? The problem with this is that each time data is added or updated to the original tables the new table must be checked to see if this data should be added or updated in the new table. This degrades performance. Another problem is that if another user wants to see that data in a different format another table must be created and maintained, adding redundancy and making the problem worse.

To provide this functionality SQL allows you to create virtual tables, called views, that can be queried as regular tables but not stored as such. The contents of these views change when the contents of the regular tables change.

Views are a mechanism provided by SQL that allow the creation of virtual tables which can be queried as regular tables but are not stored as such. Their contents change when the contents of the regular tables change. This allows a user to access frequently queried data in one table.

Advantages:

* Saves user typing long queries
* Views can be used in other statements
* Allow different users to focus in on particular data
* Provide security by hiding sensitive parts of the DB, access privileges can be granted
* If the base tables are changed the view can be re-composed altering the SELECT statement use to create the view
* Simplify database access
* Simplifies querying by drawing data from several tables

CREATE VIEW inserviceplanes (inser\_name, inser\_model, inser\_flight\_no) AS

SELECT aircraft\_name, model, flight\_no FROM aircraft, aircraft\_flight WHERE

aircraft.call\_sign = aircraft\_flight.call\_sign;

* Creates a view called inserviceplanes which takes its values from the aircraft and aircraft\_flight tables, using the field call\_sign in both tables.

Intro to Joins:

We call the combining of information from different tables a JOIN. Tables can be joined by adding a condition to the WHERE clause. You can use any of the relational operators to express the relationships between the join columns i.e. =, <, >, <=, >=, <>.

Columns from any tables may be named in the SELECT clause.

Columns which have the same name in the multiple tables named in the FROM clause must uniquely identified by specifying tablename.columnname. If the columns are uniquely named across the tables, it is not necessary to specify the table.

When and = is used as the join operator it is called and Equi-Join.

Example of Equi-Join:

SELECT select\_list FROM table1, table2, table3 WHERE table1.column = table2.cloumn AND table2.column = table3.column;