Capturing the output of your results.

The most crude approach is do a **print screen** and save that image, but you will notice that is is very cumbersome to do most of the time.

Another approach is to write the SQLs in a script and execute them as a batch and capture the output using the shell redirection operator.

Notice how below I have put all the SQLs I need to create my base project into one file (you can have multiple files if you want) and uses it to create everything in one shot. Saving the commands like this would facilitate you to work more efficiently.

```
$ cat  projectsetup.sql
DROP TABLE ourtable02;
DROP TABLE ourtable01;

CREATE TABLE ourtable01
(
   id INTEGER NOT NULL
  ,val VARCHAR(10)
  ,PRIMARY KEY(id)
);

CREATE TABLE ourtable02
(
   id2 INTEGER NOT NULL
  ,id INTEGER
  ,PRIMARY KEY(id2)
  ,FOREIGN KEY(id) REFERENCES ourtable01(id)
);

INSERT INTO ourtable01 VALUES(1, 'Meow');
INSERT INTO ourtable01 VALUES(2, 'Woof !');
INSERT INTO ourtable01 VALUES(3, 'ROAR');

INSERT INTO ourtable02 VALUES(2000, 2);
INSERT INTO ourtable02 VALUES(2003, 2);
INSERT INTO ourtable02 VALUES(2001, 1);

$
$ psql cs421 < projectsetup.sql > projectsetup.log 2>&1
Password:
```

```
$
$ cat projectsetup.log
DROP TABLE
DROP TABLE
CREATE TABLE
CREATE TABLE
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
$
```

You can use similar methods to redirect the output of the SQL into a log file and then capture them instead of having to do “screen shots”.
For example providing **`--echo-all`** to psql will cause it to echo the query into the output as well.
See example below.

```
$ cat selqry.sql
SELECT *
FROM ourtable01;
$
$ psql cs421 --echo-all < selqry.sql > selqry.log 2>&1
Password:
$ cat selqry.log
SELECT *
FROM ourtable01;
 id |  val
----+--------
  1 | Meow
  2 | Woof !
  3 | ROAR
(3 rows)
$
```

The selqry.log file can be used to provide as proof for your SQL execution.

For DB2, you can do a similar approach.

```
$ db2 connect to cs421

   Database Connection Information

 Database server        = DB2/LINUXX8664 10.5.3
 SQL authorization ID   = JDSILV2
 Local database alias   = CS421


$ db2 -t < projectsetup.sql > projectsetup.log 2>&1
```

That should help setup your database schema. (Notice how you can do the “connect” separately in the command line and not necessary to do as part of your sql script).

In db2 you can add the -v option to have the SQL echo back, so that you can capture both the SQL and the output in the log file when you redirect them.

```
$ db2 -t -v < selqry.sql > selqry.log 2>&1
$ cat selqry.log
(c) Copyright IBM Corporation 1993,2007
Command Line Processor for DB2 Client 10.5.3

You can issue database manager commands and SQL statements from the
command
prompt. For example:
    db2 => connect to sample
    db2 => bind sample.bnd

For general help, type: ?.
For command help, type: ? command, where command can be
the first few keywords of a database manager command. For example:
 ? CATALOG DATABASE for help on the CATALOG DATABASE command
 ? CATALOG          for help on all of the CATALOG commands.

To exit db2 interactive mode, type QUIT at the command prompt.
Outside
interactive mode, all commands must be prefixed with 'db2'.
To list the current command option settings, type LIST COMMAND
OPTIONS.

For more detailed help, refer to the Online Reference Manual.
```

```
db2 => db2 (cont.) => SELECT * FROM ourtable01

ID          VAL
----------- ----------
          1 Meow
          2 Woof !
          3 ROAR

  3 record(s) selected.


db2 =>
$
```