```
CREATE DATABASE markbook;
[OK]
USE markbook;
ΓΟΚΊ
CREATE TABLE marks (name, mark, pass);
ГОКТ
INSERT INTO marks VALUES ('Simon', 65, TRUE);
ГОКТ
INSERT INTO marks VALUES ('Sion', 55, TRUE);
INSERT INTO marks VALUES ('Rob', 35, FALSE);
ΓΟΚΊ
INSERT INTO marks VALUES ('Chris', 20, FALSE);
ГОКТ
SELECT * FROM marks;
[OK]
id
      name mark
                  pass
      Simon 65
                  TRUE
1
2
      Sion 55
                  TRUE
3
      Rob
            35
                  FALSE
4
      Chris 20
                  FALSE
SELECT * FROM marks WHERE name != 'Sion';
[OK]
id
      name mark pass
1
      Simon 65
                  TRUE
3
            35
                  FALSE
      Rob
4
      Chris 20
                  FALSE
SELECT * FROM marks WHERE pass == TRUE;
[OK]
id
      name mark
                  pass
1
      Simon 65
                  TRUE
2
      Sion 55
                  TRUE
// Note: this is a comment for use in this transcript only (your server doesn't
need to be able parse them)
// Assuming there is a table called "coursework" in the database (and that table
has been filled with data)
SELECT * FROM coursework;
[OK]
id
      task submission
      0X0
1
            3
2
      DB
            1
3
      0X0
            4
      STAG 2
```

```
// For JOINs: discard the ids from the original tables
// discard the columns that the tables were matched on
// create a new unique id for each of row of the table produced
// attribute names are prepended with name of table from which they originated
JOIN coursework AND marks ON submission AND id;
[OK]
id
      coursework.task
                        marks.name marks.mark marks.pass
1
      000
                        Rob
                                     35
                                                 FALSE
                                                 TRUE
2
      DB
                        Simon
                                     65
                                     20
3
      0X0
                        Chris
                                                 FALSE
4
      STAG
                        Sion
                                     55
                                                 TRUE
UPDATE marks SET mark = 38 WHERE name == 'Chris';
[OK]
SELECT * FROM marks WHERE name == 'Chris';
ΓΟΚΊ
id
      name mark pass
      Chris 38
                  FALSE
DELETE FROM marks WHERE name == 'Sion';
[OK]
SELECT * FROM marks;
ГОКТ
id
      name mark
                  pass
      Simon 65
                  TRUE
1
3
      Rob
            35
                  FALSE
      Chris 38
                  FALSE
SELECT * FROM marks WHERE (pass == FALSE) AND (mark > 35);
[OK]
id
      name mark pass
      Chris 38
                  FALSE
SELECT * FROM marks WHERE name LIKE 'i';
ΓΟΚΊ
id
      name mark pass
1
      Simon 65
                  TRUE
      Chris 38
                  FALSE
SELECT id FROM marks WHERE pass == FALSE;
[OK]
id
3
4
SELECT name FROM marks WHERE mark>60;
[OK]
name
Simon
```

```
DELETE FROM marks WHERE mark<40;
ГОКТ
SELECT * FROM marks;
ГОКТ
id
      name mark pass
1
      Simon 65
                  TRUE
ALTER TABLE marks ADD age;
ΓΟΚΊ
SELECT * FROM marks:
ГОКТ
id
      name mark pass age
1
      Simon 65
                  TRUE
UPDATE marks SET age = 35 WHERE name == 'Simon';
\Gamma OK T
SELECT * FROM marks;
ГОКТ
id
      name mark pass age
      Simon 65
                  TRUE 35
1
ALTER TABLE marks DROP pass:
ГОКТ
SELECT * FROM marks;
[OK]
id
      name mark age
1
      Simon 65
                  35
SELECT * FROM marks
[ERROR]: Semi colon missing at end of line (or similar message !)
// Assuming there is NOT a table called "crew" in the database
SELECT * FROM crew;
[ERROR]: Table does not exist (or similar message !)
// Assuming there is NOT an attribute called "height" in the table
SELECT height FROM marks WHERE name == 'Chris';
[ERROR]: Attribute does not exist (or similar message !)
DROP TABLE marks;
[OK]
DROP DATABASE markbook;
[OK]
```

Note that this transcript is not intended to be complete and comprehensive... The aim is to provide some illustrative examples of typical queries that you might expect to see. When writing your test cases, you should cover all eventualities – including both valid and invalid queries.