Business Rules

Business Database Fundamentals

Business Rules

- Information systems contain and enforce rules about the businesses they support. So what is a business rule ?
- A rule is information about how the system should behave in particular situations
  - Each employee can belong to at most one union at a time
  - A student can only enrol in a course if he/she has successfully completed the prerequisites of the course
- Some rules can be shown on the ER diagram and relational model. Some rules must be documented separately.

Types Of Rules

- Data Rules
  - Cardinality rules
    - “An employee can belong to one union at a time”
  - Data validation/update rules
    - “An inventory location status can be “E”mpty or “A”llocated.
  - Data derivation rules
    - “The available space in an inventory location is the cubic area of the location minus the cubic area of the cases that are currently stored in it”
  - Referential integrity constraints

Types Of Rules

- Process Rules
  - Determine the processing that a system will do in particular circumstances.
    - “A 0.025 sales tax levy is to be applied to the cost price of each item sold prior to mark-up + sales tax to calculate the sale price”.

Business Rules

- What rules can be shown on the ER diagram and relational model:
  - Cardinality of relationships and attributes.
  - Optional nature of relationships and attributes
  - Referential integrity
  - Restrictions on relationships by sub-typing
  - Restrictions on attribute values by showing a relationship to a look-up table.
  - Data Type
  - Primary key uniqueness

Business Rules

- What rules cannot be shown on ER diagram and relational model easily:
  - Non discreet constraints on attribute values
    - “Sale price must be positive”
  - Attribute constraints dependent on the values of other attributes in the same instance
    - “End date must be later than start date”
  - Some cardinality/optionality constraints
  - Restrictions on updateability

Business Rules

- How do you document those rules that are not easily shown on ER Diagram or relational model ?
  - Use supplementary documentation including
    - decision tables/trees
    - data/flow diagrams
    - psuedo-code
    - plain language

Data Updates

- Data update rules may be comprised of:
  - Restrictions on inserting an instance of an entity (record).
  - Restrictions on the values that attribute may contain.
  - Restrictions on modifying values of existing instances (record).
  - Restrictions on removing instances of entity (record).

Data Updates

(referential constraints)

- What do we do when the primary key value that a foreign key refers to is updated ?
  - Disallow updates of primary keys
  - Cascade the update to all foreign keys
- What do we do when a record with a primary key value that a foreign key refers to is deleted ?
  - Disallow delete while there are references to the primary key from other tables
  - Null all foreign keys on delete
  - Cascade the delete to the foreign key table

Implementing Business Rules

- Means of implementing business rules
  - Database structure
  - Column definitions
  - Program logic
  - Database values
- Most appropriate method depends on the volatility of the rules
  - Recommendation: If a rule is volatile you do not enforce it at the database structure design level

Implementing Business Rules

- Can we build in some flexibility with volatile business rules ?
  - An employee can only have one super fund

    What happens if regulations change and an employee can now belong to multiple super funds?   What was once an attribute must now be link to another table.
  - A 1:many relationship that appears to be volatile may be implemented as a many:many at the database structure level and 1:many relationship may be enforced with a database trigger of program logic. This is easier to change than changing the database structure

Implementing Business Rules

- Hold rules in program logic.
  - All products attract a 0.025 sales tax levy on cost price.
    - Easily implemented at program level.
- The smart thing to do is implement it as a library function (CalcSalesTax) so there is one piece of code that needs to be changed in the event of a revision of the levy
  - Cost savings are created through modularisation
  - But what if the sales tax levy changes ?

Implementing Business Rules

- Store the sales tax levy in a separate database table
  - Write code to retrieve the value and to use it in calculation
- Code is more complex and may need maintenance screens to update it
- Application development and testing costs go up but production support costs go down
- Data supporting these rules needs to be protected from unauthorized access.

Integrity Rules

- Integrity rules/constraints are rules we specify that must be met before a row of data is inserted or a value is updated in a row.
- Rules attempt to keep the data in the tables valid, hence maintain integrity of database.
- Rules can be defined:
  - When creating a table using CREATE TABLE
  - After creating a table by using ALTER TABLE

Column Vs Table Constraints

- Column constraints are defined within the field definition column
  - Only usable if constraint involves one field
- Table constraints are defined at end of the table definition
  - Usable when constraint involves more than one field
- Some constraints only work in one form

NOT NULL

- By default all fields are defined as NULLable.
- Specifying NOT NULL means that the field is mandatory, data is required when inserting or updating a row.

Eg.

```
CREATE TABLE customer (
  CustomerID INT(10) NOT NULL,
  Surname VARCHAR(40),
  Given VARCHAR(40),
  DOB DATE,
  Sex CHAR(1) NOT NULL,
  …
  State VARCHAR(3),
  Postcode VARCHAR(5)  );
```

PRIMARY KEY – on one field

- A field that uniquely identifies each row of a table.

- Eg.

```
CREATE TABLE mobile (
  MobileID          INT(10) PRIMARY KEY,
  PhoneNumber       VARCHAR(20) NOT NULL,
  BrandName         VARCHAR(40),
  Joined            DATE,
  Cancelled         DATE,
  PlanName          VARCHAR(20),
  PhoneColour       VARCHAR(20),
  CustomerID        INT(10),
  StaffID           INT(10)
);
```

PRIMARY KEY – more than one

- A combination of one or more columns which uniquely identify each row of a table.

- Eg.

```
CREATE TABLE connect (
    TowerID INT(10),
    CallsID INT(10),
CONSTRAINT CONNECT_PK
   PRIMARY KEY (TowerID, CallsID)
);
```

Note: "CONSTRAINT CONNECT_PK" allows us to define a specific name for this constraint. Without it a name is automatically generated.

FOREIGN KEY

- A column whose value is based on the value of a primary key value of another table.

- Eg.

```
CREATE TABLE mobile (
  MobileID           INT(10) NOT NULL,
  PhoneNumber VARCHAR(20) NOT NULL,
  BrandName          VARCHAR(40),
  Joined             DATE,
  Cancelled          DATE,
  PlanName           VARCHAR(20),
  PhoneColour VARCHAR(20),
  CustomerID  INT(10)
         REFERENCES Customer(CustomerID),
  StaffID            INT(10)
         REFERENCES Staff(StaffID)
);
```

FOREIGN KEY – cascade delete

- deletes dependent row when you delete the corresponding row in the parent table

- Eg.

```
CREATE TABLE mobile (
  MobileID          INT(10) NOT NULL,
  PhoneNumber VARCHAR(20) NOT NULL,
  BrandName         VARCHAR(40),
  Joined            DATE,
  Cancelled         DATE,
  PlanName          VARCHAR(20),
  PhoneColour VARCHAR(20),
  CustomerID        INT(10),
  StaffID           INT(10),
FOREIGN KEY (CustomerID)
  REFERENCES Customer(CustomerID)
  ON DELETE CASCADE
);
```

Procedural Constraints

- More complex constraints can be done using triggers.

- Triggers, Stored Procedures and Functions usually covered in Advanced Database course

Information System: Components

- Components of IS Business Application
  - CRUD – Create, Read, Update and Delete
  - Reporting
  - Search and Navigating
  - Transactional processing
  - Auditing and logging
- What do all these components need?
- What do we use in order to access what is needed?

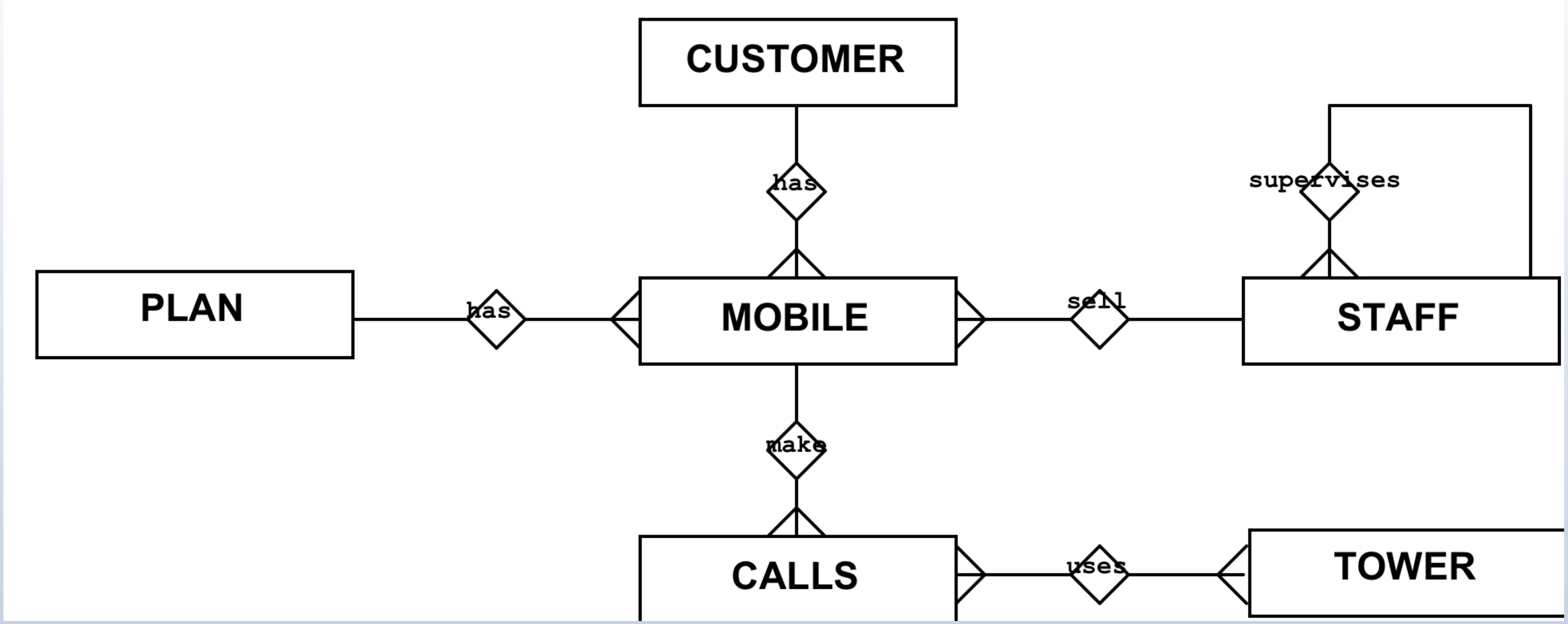
Information System: Data

- Data model for
  - Business data
  - Data about the data
  - Flexible business rules
  - Supporting process

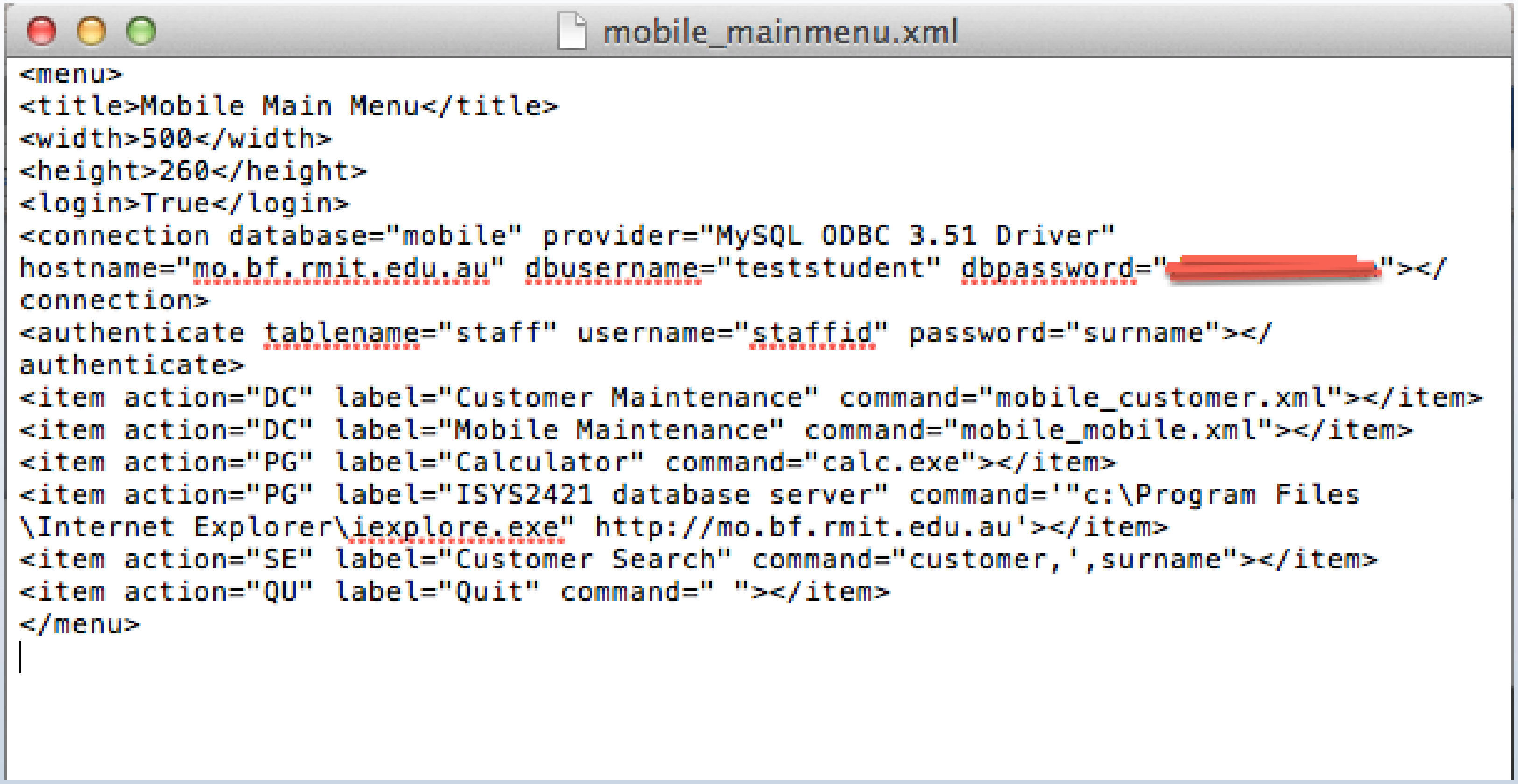
- Example:
  - Mobile phone database

Mobile Phone: Data

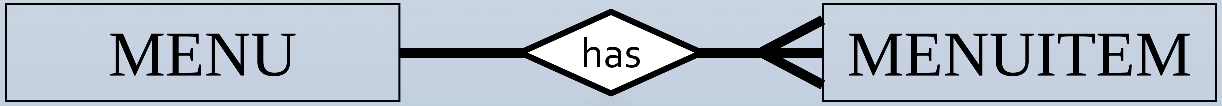
Entity-Relationship Diagram



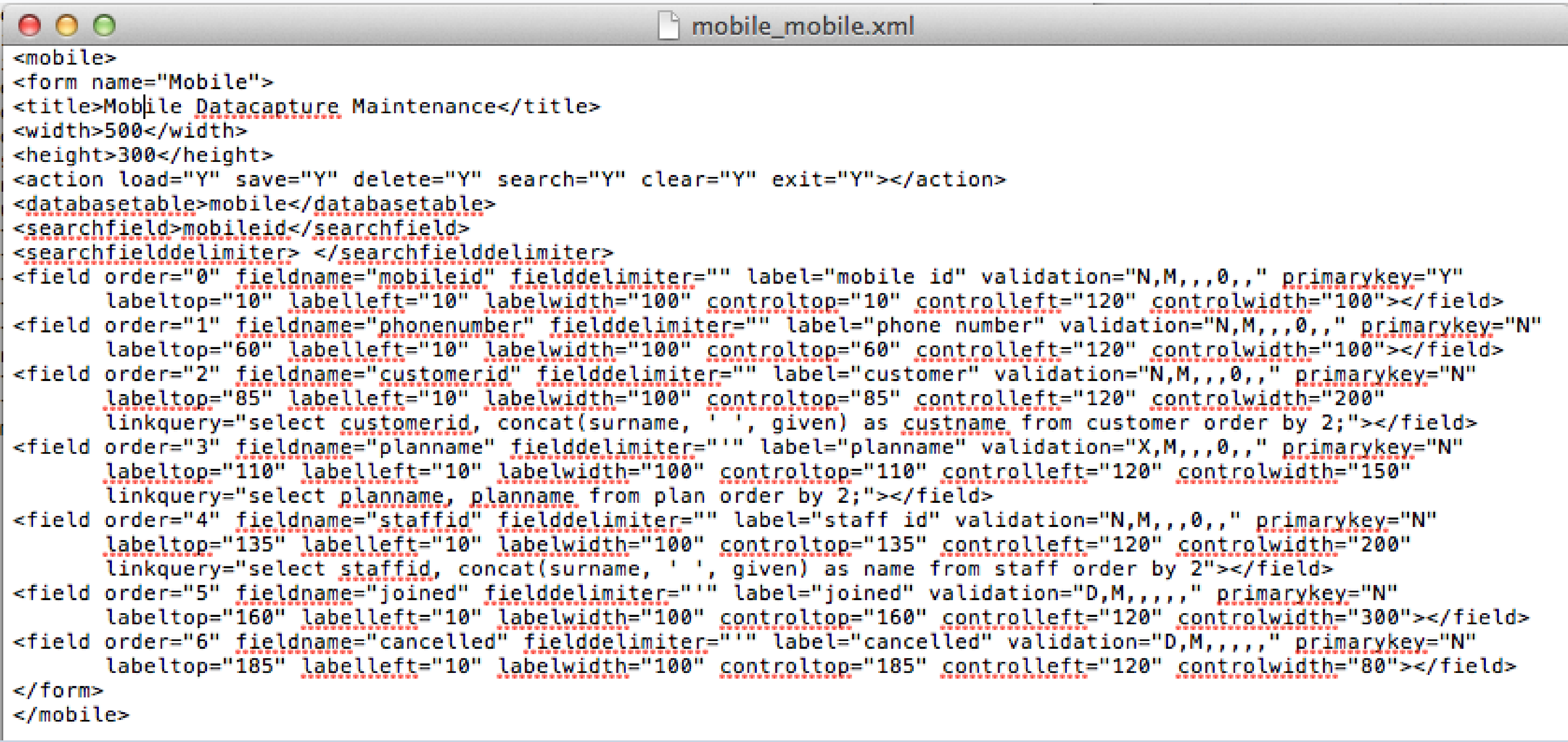
Mobile Phone: Menu



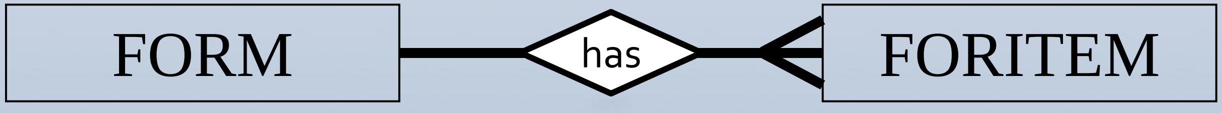
```
<menu>
<title>Mobile Main Menu</title>
<width>500</width>
<height>260</height>
<login>True</login>
<connection database="mobile" provider="MySQL ODBC 3.51 Driver"
hostname="mo.bf.rmit.edu.au" dbusername="teststudent" dbpassword="[redacted]"></
connection>
<authenticate tablename="staff" username="staffid" password="surname"></
authenticate>
<item action="DC" label="Customer Maintenance" command="mobile_customer.xml"></item>
<item action="DC" label="Mobile Maintenance" command="mobile_mobile.xml"></item>
<item action="PG" label="Calculator" command="calc.exe"></item>
<item action="PG" label="ISYS2421 database server" command='"c:\Program Files
\Internet Explorer\iexplore.exe" http://mo.bf.rmit.edu.au'></item>
<item action="SE" label="Customer Search" command="customer,',surname"></item>
<item action="QU" label="Quit" command=" "></item>
</menu>
```



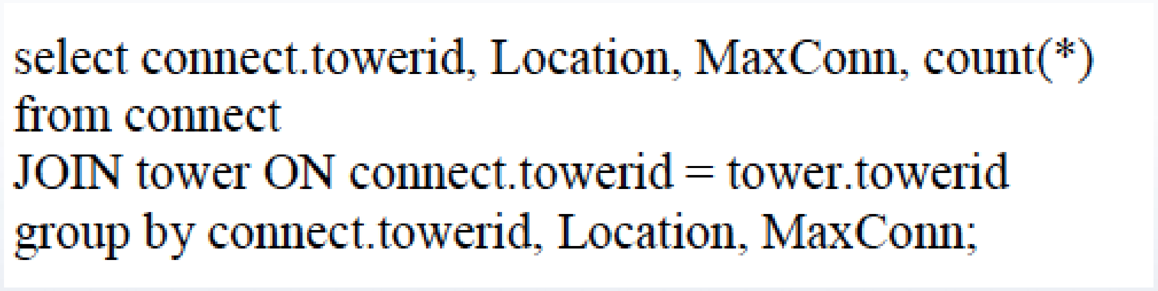
Mobile Phone: CRUD



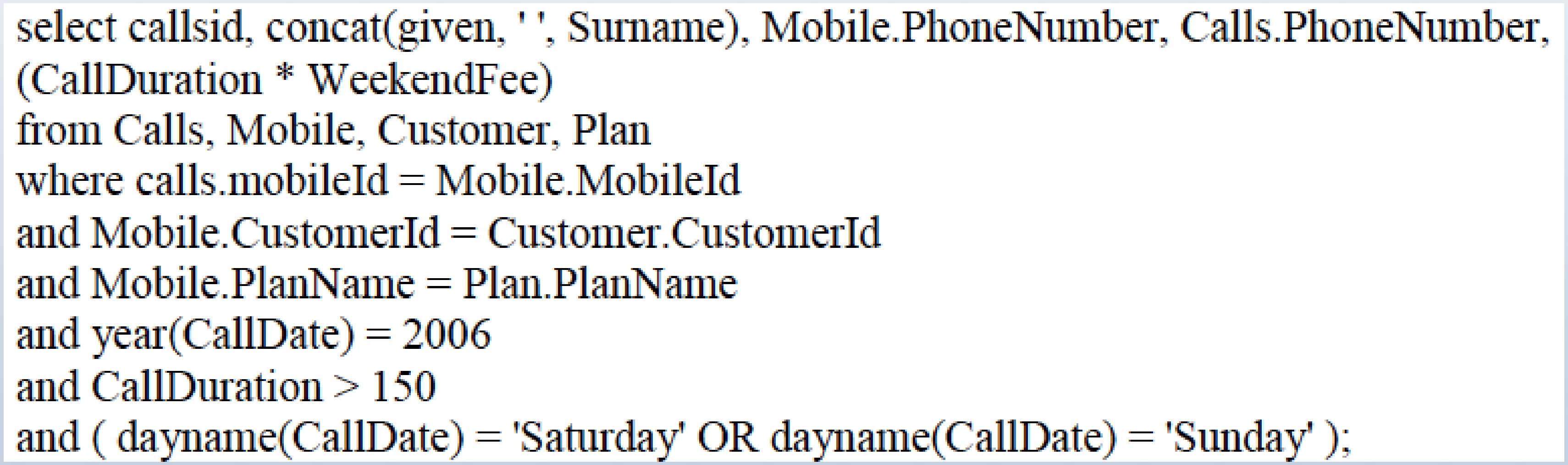
```
<mobile>
<form name="Mobile">
<title>Mobile Datacapture Maintenance</title>
<width>500</width>
<height>300</height>
<action load="Y" save="Y" delete="Y" search="Y" clear="Y" exit="Y"></action>
<databasetable>mobile</databasetable>
<searchfield>mobileid</searchfield>
<searchfielddelimiter> </searchfielddelimiter>
<field order="0" fieldname="mobileid" fielddelimiter="" label="mobile id" validation="N,M,,,0,," primarykey="Y"
       labeltop="10" labelleft="10" labelwidth="100" controltop="10" controlleft="120" controlwidth="100"></field>
<field order="1" fieldname="phonenumber" fielddelimiter="" label="phone number" validation="N,M,,,0,," primarykey="N"
       labeltop="60" labelleft="10" labelwidth="100" controltop="60" controlleft="120" controlwidth="100"></field>
<field order="2" fieldname="customerid" fielddelimiter="" label="customer" validation="N,M,,,0,," primarykey="N"
       labeltop="85" labelleft="10" labelwidth="100" controltop="85" controlleft="120" controlwidth="200"
       linkquery="select customerid, concat(surname, ' ', given) as custname from customer order by 2;"></field>
<field order="3" fieldname="planname" fielddelimiter="'" label="planname" validation="X,M,,,0,," primarykey="N"
       labeltop="110" labelleft="10" labelwidth="100" controltop="110" controlleft="120" controlwidth="150"
       linkquery="select planname, planname from plan order by 2;"></field>
<field order="4" fieldname="staffid" fielddelimiter="" label="staff id" validation="N,M,,,0,," primarykey="N"
       labeltop="135" labelleft="10" labelwidth="100" controltop="135" controlleft="120" controlwidth="200"
       linkquery="select staffid, concat(surname, ' ', given) as name from staff order by 2"></field>
<field order="5" fieldname="joined" fielddelimiter="'" label="joined" validation="D,M,,,,," primarykey="N"
       labeltop="160" labelleft="10" labelwidth="100" controltop="160" controlleft="120" controlwidth="300"></field>
<field order="6" fieldname="cancelled" fielddelimiter="'" label="cancelled" validation="D,M,,,,," primarykey="N"
       labeltop="185" labelleft="10" labelwidth="100" controltop="185" controlleft="120" controlwidth="80"></field>
</form>
</mobile>
```



Mobile Phone: Reports



```
select connect.towerid, Location, MaxConn, count(*)
from connect
JOIN tower ON connect.towerid = tower.towerid
group by connect.towerid, Location, MaxConn;
```



```
select callsid, concat(given, ' ', Surname), Mobile.PhoneNumber, Calls.PhoneNumber,
(CallDuration * WeekendFee)
from Calls, Mobile, Customer, Plan
where calls.mobileId = Mobile.MobileId
and Mobile.CustomerId = Customer.CustomerId
and Mobile.PlanName = Plan.PlanName
and year(CallDate) = 2006
and CallDuration > 150
and ( dayname(CallDate) = 'Saturday' OR dayname(CallDate) = 'Sunday' );
```