2001

A Web-Enabled Temporal Database Human Resources Application

Joseph A. Brooke III
University of North Florida

A Web-Enabled Temporal Database Human Resources Application

by

Joseph A. Brooke, III

A Professional Option Project submitted to the Department of Computer and Information Sciences in partial fulfillment of the requirements for the degree of

Candidate for Master of Science, Computer and Information Sciences

University of North Florida
Department of Computer and Information Sciences

April, 2001

The Professional Option Project "A Web-Enabled Temporal Database Human Resources Application" submitted by Joseph A. Brooke in partial fulfillment of the requirements for the degree of Master of Science in Computer and Information Sciences has been

Approved by:	Date
Signature Deleted Dr. Jae Y. Lee Project Director	4-30-01
Signature Deleted Dr. Charles N. Winton Graduate Director	5-2-2001
Signature Deleted Dr. Judith L. Solano Department Chair	5/2/01

ACKNOWLEDGEMENT

This work represents the culmination of over five years of effort in the pursuit of higher education. No effort of this magnitude can be successfully undertaken by a single person. My wife Elizabeth and son Joe share equally in this accomplishment, as they made sacrifices of at least equal measure to any I may have made. I would like to thank them for their unending support and seemingly constant words of encouragement. When I was ready to give up they were always there with a kind word.

CONTENTS

ABSTRACT

Despite the inclusion of a variety of time-related or 'temporal' datatypes in the SQL-92 standard, vendors of commercially-available Relational Database Management Systems (RDBMS) have universally elected to not fully comply with the standard. Perhaps even more frustrating is the fact that each vendor has chosen to include a different subset of temporal datatypes than their competitors, with most vendors adding a proprietary twist to their datatypes not even contained within the standard.

This lack of conformity has left users of these database products faced with a difficult choice: either avoid temporal functionality within their database applications or develop and manage complex and often convoluted code to insert, maintain, and query this important data. Industry is replete with examples of both choices; not all of which have happy endings.

This project demonstrates some of the options available to RDBMS users who choose to employ temporal functionality in their applications. Alternatives are presented using a conceptual human resources application deployed via internet. Data is stored in a commercial RDBMS product. Solutions to temporal storage and query issues are presented.

Chapter 1

Introduction

In 1997, sixteen cases of E. Coli bacteria were reported in the U.S., all of which were subsequently traced to a meat packing facility in Columbus, Nebraska. Ironically this facility, owned by Hudson Foods Corp., had consistently received good marks from governmental inspectors for its cleanliness and adherence to federal standards. The outbreak had not originated at the processing plant however, but at one of the slaughterhouses which supplied carcasses to the plant.

Because the E. Coli bacteria is contagious in cows, the outbreak could have been contained relatively easily by knowing which cows had occupied a specific pen at the slaughterhouse concurrently with other cows. Unfortunately however, no such data was kept by the processing plant or the slaughterhouses, and so all of the slaughterhouses became suspect. This eventually led to the recall of approximately 25 million pounds of frozen hamburger, about one-fifth the plant's total annual output. In short, the lack of a temporal or 'time-sensitive' database cost Hudson Foods more than $20 million[8[.

The above case is but one of several examples which illustrate the ongoing need for temporally-enabled databases and database applications in commercial and industrial environments. As this need becomes more prevalent, it becomes acutely apparent that current relational systems do not adequately support these requirements. Although current RDBMS systems can store date-related data using 'timestamp' datatypes the granularities of these datatypes are sometimes quite limited, and often present issues when users attempt to obtain valid query results.

The goal for this project is to illustrate some of the various temporal aspects of a hypothetical business environment, and to exemplify the lengths to which users of currently available database systems must go to achieve a viable degree of temporal functionality within those systems. Toward that end, a database application will be developed for a theoretical human resources division of a large company as a vehicle to demonstrate existing fundamental temporal requirements of business and a possible (if not somewhat convoluted) method of dealing with those requirements.

To achieve this example, the project will incorporate a working relational database model with a schema designed to facilitate the storage and query of temporally-related data. It will also involve a graphical web-based 'front-end' to allow users to insert, update, and most importantly, query the database for date-related information. In addition, the user interface will include functionality which demonstrates the management of temporal data from an administrative (as opposed to end-user) perspective.

Chapter 2

Issues in Temporal Data Management

2.1 Support for Temporal Data in the SQL-92 Standard

The current standard for the implementation of SQL is based upon ISO Standard 9075, commonly referred to as 'SQL-92'. This standard called for the inclusion of several datatypes and features relating to temporal aspects of data. They are each described below:

- DATE: Specified as a datatype, the DATE attribute was specified to store date values in the form YYYY-MM-DD, with valid values of January 1st, 0001 to December 31st, 9999 AD.

- TIME: Also a datatype, TIME is defined in the standard to store time values in the form HH:MM:SS.999, where '999' depicts milliseconds. Any valid time in a 24-hour day is considered a valid value.

- TIMESTAMP: A combination of the DATE and TIME datatypes.

- INTERVAL: The INTERVAL datatype allows users to specify a starting point in time using either the DATE or TIME datatype, along with an offset to indicate a fixed span of time.

- TIMEZONE: This datatype acts as an offset or displacement to a TIME attribute. It enables users to record data in a TIME attribute using Universal Coordinated Time (also known as Greenwich Mean Time) instead of local time, and then add another attribute of type TIMEZONE to convert that TIME value into the correct local time in whatever timezone the user is in.

- CURRENT: The CURRENT feature was designed to be used as a function in the SQL language for the purpose of entering a value into a TIMESTAMP datatype attribute mentioned above. This function takes the time from the computer's operating system and inserts it into a TIMESTAMP field for the purpose of identifying when a certain insert or update was made to a tuple.

While these temporal attributes provide the framework necessary to capture and manage time-specific data, a true set of temporal predicates and semantic structures were not included in the original standard. What is most limiting however is the fact that these temporal datatypes are not completely supported by any commercial RDBMS product on the market today. In addition to not being in full conformance with the SQL-92 standard, each vendor has developed their own proprietary SQL 'enhancements', thus serving to not only further distance themselves from the standard but making code portability between database products practically impossible. Although these enhancements have ultimately added several supplemental tools to each vendor's version of SQL, they have increased the slope of the learning curve for a developer or DBA trying to transition from one platform to another.

	Informix	SQL Server	Oracle
DATE	✓	**P**	**P**
TIME	**P**	**P**	**P**
TIMESTAMP	**P**	✓	✓
INTERVAL	✓	**P**	**P**
TIMEZONE			
'Current'	✓	✓	✓

The chart above illustrates the extent to which the SQL-92 standard is supported by three major relational database products. A check mark (✓) indicates full adherence to the standard, while a capital 'P' indicates partial compliance in some form. Again, it is important to note here that while many of these datatypes were at least partially supported by commercial RDBMS systems as shown, the existence of the temporal predicates and other aspects of semantic language are not present, even in the proprietary language extensions noted earlier.

In the vendors' defense it should be noted that while the 'P' values represented in the chart denote only partial compliance, an astute programmer could indeed manipulate the partially-compliant datatypes into a reasonable facsimile of the SQL-92 standard. For example, the Oracle DATE datatype is actually a composite 7-byte field which includes year, month, day, hour, minute, second, and millisecond. While it is possible to derive only the date value (or any other component) from this datatype by parsing it from the value of a DATE field, producing these values separately requires extra programming work as well as processing time.

2.2 The Representation of 'Now' in a Temporal Database

While the insertion of a static date value into a temporally-enabled table is in and of itself not difficult, one concept that designers of temporal databases often find complex to manage is the representation of 'Now' within the database. This issue manifests itself frequently in a valid-time or bitemporal table where the table records information for intervals of time that have not yet expired, and/or for which we do not yet know the interval's endpoint. For this discussion we will use the example of a relation named EMPLOYEE, defined as follows:

NAME	POSITION	FROM_DATE	END_DATE
Bob	Assistant	01-JUN-85	?

In this example we have an employee named Bob, who is employed as an Assistant beginning on June 1st, 1985; Bob holds that position currently and indefinitely.

Conceptual models of temporal database systems have unique representations of this value, including 'NOW', '@', '∞', '-', and 'until-changed' 1. Obviously these values cannot be used in a commercial RDBMS, since the attempted insertion of a text string into a field defined as a DATE datatype would constitute a data integrity violation. In the scope of valid values, we have several alternatives. Our first alternative is to insert the current date. Had this option been selected on the date Bob was hired, we would have inserted '01-JUN-85' into the END_DATE field. A simple query on Bob's tenure performed on June 1st, 1985 would have reflected that he had been employed for one day thusfar. On June 2nd however (and each succeeding day) it would have been necessary to update the END_DATE field with the current date, a most labor-intensive project. Perhaps the most serious disadvantage of this option however is the concept that entering the current date into the END_DATE implies that Bob's employment has ceased as of that date. In other words, this option leads to an overly pessimistic interpretation of this tuple when in fact it is unknown how long Bob will be employed.

The next alternative is to use the maximum value permitted by the RDBMS. Most prominent vendors' products conform to the maximum date specified by the SQL-92 standard, December 31st, 9999. While this option avoids the maintenance problems associated with using the current date, it has other disadvantages. First, it is optimistic in that, when applied to our example, it implies that Bob will be employed as an Assistant for another 8,000 years. In addition, it is predictive in that it implies a termination date which at the current time cannot be proved or disproved. In the temporal world this is tantamount to predicting a future state of the database.

A third option would be to leave the END_DATE field blank, i.e., NULL. Unfortunately however, a NULL value is often ignored by a query, which would render any query results inaccurate. Ideally then, the solution to this problem would be the use of a value which would:

- eliminate the requirement for manual, regular updates, and
- incorporate a value which would be meaningful and accurate in query situations.

To satisfy these criteria, temporal database designers have devised the notion of a 'bind operator', a fixed, ground value inserted at query time to provide accurate query results. At the conceptual level, END_DATE attributes would be initially populated with a value or symbol having a recognized meaning of 'until changed'. When a query is executed involving a tuple with an attribute such as this, the query preprocessor substitutes a valid date value corresponding to 'NOW'. After the query is completed, this current-date value would then be changed back to its original value.

At the practical level, the bind operator concept can be implemented using a database trigger. Although the timing of the trigger differs from that of the preprocessor in the conceptual model, then end result is similar. In a commercial RDBMS, the trigger would be created so as to fire prior to the execution of any query involving a table with temporal attributes. The trigger would be written to search through specific temporal attributes of a table, replacing any values of '12-DEC-9999' (or other platform-specific value) with a current timestamp. Following the printing of the query results, another trigger would replace these current timestamp values with the original maximum value which was present prior to the query. This satisfies the necessary criteria for low maintenance and accurate reporting, while avoiding the pitfalls of pessimism or optimism inherent to the use of variables noted earlier.

2.3 Classifications of Temporal Tables

Certainly the most common type of temporal data stored is that which provides a view of when a condition existed in real time; in other words, a modeled reality. A relation capable of storing data of this type is known as a 'valid-time state table'. In addition to storing whatever non-temporal data is relevant, a valid-time state table would include attributes to designate when the information in that particular tuple started being valid and ceased to become valid. Using an example of an EMPLOYEE table indicating employee name, position name, and the dates that employee held this particular position, the schema for such a table could appear as follows:

E_NAME	POSITION	START_DATE	END_DATE

A valid-time state table can be surveyed using any of three types of queries:

- Current: This type of query returns data that is valid at a given time, and is also known as a 'current time-slice query'.
- Sequenced: Returns multiple rows of valid-time data as a history of criteria specified in the predicate of the query.
- Non-sequenced: A query that does not consider temporal attributes as determinate criteria for inclusion in the return. This type of query considers data that was valid either over a span of periods ('past and present') or merely at some point in time which is not relevant to the query itself.

Another type of table, known as a 'transaction-time state table', records the time in which data is entered or modified in the table itself. Capturing data in this state has several functions. First, it captures a history of the changing state of the table. This permits database administrators to answer the question 'What was the state of the database at "X" time?'. Also, it allows a logical rollback of the table to a previous state, which can be particularly useful in the case of a database which cannot be shut down and restored to a prior state. The schema for a transaction-time state table would appear similar to a valid-time state table, with the START_DATE field representing the times in which the data was entered into the database itself and the END_DATE field denoting the point at which that data was modified to another value or logically deleted.

Inclusion of transaction time has a price however. While tracking of valid time data involves the update of temporal values within a tuple, the incorporation of transaction time means that a new row must be added to the table whenever data is added or modified. The implications are obvious: transaction-time state tables become larger at a much faster rate than valid-time state tables, resulting in increased storage requirements and reduced query response time. Like valid-time tables, transaction-time tables can also be examined using different types of queries, as follows:

- Reconstruction: Also known as a 'current' or 'time-slice' query, this type of query simply returns the state or instance of the table at the date given as input in the query.
- Sequenced: Returns a history of when the data was recorded into the table.
- Non-sequenced: Typically used for auditing to determine when a particular change was made to the table. The changes appear in the table as two periods of distinct value that meet at a given point (the point of change).

The most versatile type of temporal table however and the type most popular with designers of temporal databases is called the 'bitemporal table'. This includes attributes for both valid-time and transaction-time data. It's schema appears below.

E_NAME	POSITION	VT_START	VT_END	TT_START	TT_END

The bitemporal architecture offers users the ability not only to work with valid-time data in the present time, but also to perform valid-time queries at some instant in historical time.

Chapter 3

Implementation

To demonstrate the management of temporal data in a hypothetical production environment, a software application was designed which might theoretically be employed by the human resources department of a medium-sized corporation. While by no means intended to be fully-functional, the application allows users to insert, update, and query various types of temporal and non-temporal data relating to the area of human resources management. It also includes security features which allow only authorized individuals to access the system.

3.1 Interface Architecture

The application's architecture is a two-tier internet-deliverable design. The user interface or 'front end' is comprised of a collection of active-server web pages written using the web application development suite Cold Fusion, produced by Allaire Corp. Cold Fusion installs on a Windows NT or UNIX-based server platform, and runs as a service/daemon of the operating system. Cold Fusion works in conjunction with a web server such as Internet Information Server or Apache, which must be running on the server as well. As with other active-server architectures, web page code is stored on the server in text files. When an HTML page is requested by a remote user, the Cold Fusion service/daemon parses the appropriate active-server text file into HTML code and submits it to the web server for delivery to the remote user. This architecture allows the values of dynamic variables to be hard-coded into the delivered HTML code by the parser, enabling 'custom' web pages to be delivered to each remote user during each individual session. All processing of the web pages is done on the server side, thus requiring the server hardware to be rather robust but facilitating the use of the application by users with minimally-configured workstations (only current-generation web browsing software is required, along with an internet connection). See Appendix C for a hierarchy of application web pages and a complete listing of the source code for each active-server page text file.

For this implementation a Compaq server running Microsoft Windows NT version 4 Server operating system was used to host the web application. The web server used for this project is Internet Information Server version 4, which is packaged with the Windows operating system.

3.2 Database Design

Data for the application is maintained in an Oracle database running on a Windows NT-based server. The database schema consists of several tables connected relationally by foreign key constraints. The tables created for the application and their functions are as follows:

- DEPARTMENTS: Lists department names along with a primary key department number field.
- EMPLOYEES: Includes preliminary information for each employee including first and last names, their current job number, hire and termination date, and employee ID, which is the primary key for the table.
- EMPLOYEES_DTL: For this application, this table includes employee ID (as a foreign-key constraint on the EMPLOYEES table) and a column named LOGIN containing the user's system password. There is also a column which denotes whether the user is a system administrator, FL_ADMIN. This is the table queried by the application during the login process. In a true production environment this table would most likely also include additional detail information relative to each employee, such as address, family contact, etc.
- EMP_JOBS: This table incorporates columns for employee ID and job number as foreign keys from of the EMPLOYEES and JOBS table, respectively. The table also includes the columns EMP_LST_UPDT and DT_LST_UPDT. These columns are transparently populated by the application with the employee ID and date that the record was added to the database for auditing purposes.
- EMP_JOB_SALS: This table includes columns for employee ID, job number (as foreign keys on their respective tables) as well as annual salary, employee last update, and date last update for auditing purposes. During the course of an employee's tenure in a single job, this table may include multiple tuples reflecting salary changes within that job.

- JOBS: Each job within the company is listed as a tuple in this table, with a primary key of job number. Additionally, columns for job name, job description, job level and supervisory position are included. This table also includes columns for reflect on what date each job was created and when it was last updated.

In addition to the non-temporal columns listed above, the EMP_JOBS and EMP_JOBS_SALS tables each contain two additional columns: DT_START and DT_END. Each of these columns are of DATE datatype, and contain the date on which the data contained in the non-temporal columns of each row became valid and ceased to be valid, respectively. Therefore each of the tables containing these two columns can be considered a *valid-time table.* [Snodgrass 00]

Each table is created with a primary key. In each valid-time table all of the columns excepting the audit columns are taken together to form a composite primary key. In the Oracle RDBMS architecture, every table created with a primary key has a unique index on the primary key column which is automatically created along with the table. A SQL script to view information regarding these indices and its resulting output is contained in Appendix A, as is an entity-relationship diagram and the SQL code used to create the tables referenced above.

3.3 Business Rules of the Application

The application attempts to enforce certain business rules to ensure uniformity with regard to data. These rules are implied in the functionality of the system and are for the most part transparent to the user. They are as follows:

- No employee may hold more than one job at any given point in time. An employee will be assigned to a job on their date of hire, and will be assigned to a job until his date of separation from the company. The employee's job may change from time to time, however there may be no overlap in periods from one job to the next.
- Jobs are maintained within the application as non-unique, therefore multiple employees may be assigned to the same job at a given point in time.
- An employee may not be listed as holding a job as of a date prior to their date of employment or after their date of separation.
- Employees will hold jobs and/or salary levels in intervals of one or more days. The minimum granularity of the application is one day.

Chapter 4

Application Functionality

4.1 Login

To begin using the application, the user directs their web browser to the login page of the application (Appendix A, LOGIN.CFM). This page features text input boxes for the user's employee ID and password as well as a button which, when clicked by the user, submits the employee ID and password to the application. Upon clicking the button, the Cold Fusion service submits a SQL query to the database using the two variables submitted by the user. Employee ID and password values are contained in the EMPLOYEES_DTL table of the database, and only if the query returns a row from that table with corresponding data will the application allow the user to proceed to the next page. Users who attempt to bypass the login page and enter the application directly will receive an error message. In the event of an unsuccessful login attempt the user will be directed to a page advising them of their unsuccessful attempt and providing a hyperlink to return to the login page for another attempt.

Users who successfully log in to the application are presented with a menu-style listing of hyperlinks which lead to a variety of insert, update, and query sub-menus. Each page includes the user's name at the top of the page, which is queried from the EMPLOYEES table at login and coded into each HTML page the user accesses. The user's name is stored as a variable cached in memory on the web server along with the IP address of the user's workstation. Each time the user requests a new page from the web server, the Cold Fusion service copies the IP address from the TCP packet and compares it with entries in the cache. This architecture enables the user to have a 'personalized' session despite the connectionless nature of TCP internet protocols. The user's connection information is purged from the server's variable cache after thirty (30) minutes of inactivity by the user. Continued use of the application after this idle period requires a new login.

4.2 Insert and Update

Several HTML pages are devoted to insert and update of both temporal and non-temporal data. Users are prompted with text boxes for the input screens, and can select records for update from dynamically-populated drop-down boxes. While the non-temporal insert and update screens do not directly contribute to the temporal research aspects of the project, they allow users to manipulate data directly in the application which supports those temporal aspects. They were therefore considered a necessary feature of the application and are functional for all tables.

For this application, temporal data is contained mainly in two tables: EMP_JOBS and EMP_JOBS_SALS. Temporal data is entered and viewed by the user in the format MM/DD/YYYY rather than the Oracle NLS default DD-MMM-YY format. Translation is handled by the active server page code using the Oracle TO_DATE and TO_CHAR functions.

Since enforcement of the business rules listed above include the avoidance of overlapping intervals for related data within temporal tables, logic within the application attempts to prevent this wherever possible. This particular functionality was originally planned to be incorporated at the database level using multiple triggers which passed parameters to stored procedures. This design however would have resulted in an error resulting from attempted operations on a mutating table, which occurs whenever a trigger attempts to manipulate data in a table which is in the process of undergoing an insert, update, or delete operation (i.e., a 'mutation'). Instead, these features exist within the application code, beginning with the 'Employee Position Management' link on the application's main menu. Clicking this link displays the screen EMPJOB_ADD.CFM which prompts the user to select an employee and job, each from a dynamically-populated dropdown box. A 'Submit' button on this screen sends the user's session to the succeeding screen, EMPJOB_ADD2.CFM. This screen accepts parameters for employee ID and job number and queries the database for current and proposed job names. These values are displayed along with the employee's original hire date. This screen also displays text input boxes which prompt the user to enter the starting date of the proposed job in MM/DD/YYYY format as well as an annual salary.

There are three distinct possibilities with regard to the date which the user may enter. The first possibility is that the date entered will be earlier than the employee's hire date. Although displaying this hire date on the screen should serve to minimize the occurrence of this error, error-handling code within the application should eliminate it altogether. In the event the user enters a date prior to the employee's hire date, the application will return an error message indicating this event as well as listing the user-supplied date and the hire date on a new screen. A button labeled 'Return' prompts the user to return to the EMPJOB_ADD.CFM screen to re-enter the data.

The second possibility is that the user may enter a date which is later than the start date of that employee's current job. An entry of this nature would result in an overlap of the time intervals for which an employee held a job within the company. This is contradictory to the business rules of the application, which are based upon the premise that an employee may only hold one job at any given instant in time. In the event the user enters a date less than the start date of the current job the application will again present the user with an error message indicating the date entered and the start date of the employee's current job, as well as a button labeled 'Return'. Clicking the button will direct the user back to the EMPJOB_ADD.CFM screen to re-enter the data.

The final possibility is that the job start date entered by the user is greater than the start date of the employee's current job, which results in no overlap in interval and is in keeping with the application's business rules. In this event the application inserts a tuple into the EMP_JOBS table containing employee ID, job number, the start date of the new job, and the date value '12/31/9999' for the ending date of the employee's new job. This value indicates that the employee has this job into the foreseeable future or, 'until changed'. In addition, the application also locates the tuple in the EMP_JOBS table corresponding to the employee's current job and replaces the ending date (which was stored as 12/31/9999) with the value of the new job starting date minus one day. Lastly, a similar insert/update operation is made to the EMP_JOB_SALS table, where a tuple is inserted reflecting the salary of the new job as well as employee ID, job number, and start date. Like its related row in the EMP_JOBS table, this newly-inserted tuple in the EMP_JOB_SALS table includes an ending date of '12/31/9999'. In the event of a successful insert using an acceptable date the user is presented with a confirmation screen indicating the values inserted into the database and a prompt to return to the main menu.

4.3 Temporal Querying

Example 1: Employee Salary History

Temporal data management is illustrated in the form of three examples in the user interface. The first example allows the user to view an employee's salary history over the course of their employment. This is accomplished through the use of two interface screens. The first screen (web page EMP_QUERY.CFM) presents the user with a dropdown box containing the names of all employees, sorted by last name. This dropdown is dynamically populated by the application each time a call is made for this page. The query that returns the population set for the dropdown is written into the web page file as active server code. It selects employee ID as well as first and last names, and orders the return set by last name. Each row in the dropdown contains each of these values. When the user highlights the employee he wishes to view and clicks the 'Submit' button the application passes the employee ID value of the selected row to the next screen as a memory variable.

The second screen of this function (EMP_QUERY2.CFM) takes the employee ID value passed to it from the first screen (EMP_QUERY.CFM) and uses it to query the JOBS and EMP_JOBS_SALS tables to obtain job name, job start date, and job end date for that employee. The return set from the query is displayed in a HTML table with column headings for Job Title, Salary, Start Date, and End Date. Since Oracle stores dates in the format 'YYYY-MM-DD-HH-MI:SS', the start-date and end-date columns must be translated into a readable format. This is done using the Oracle TO_CHAR function, which takes a date value as input and converts it into a more readable format of the programmer's choosing. TO_CHAR has the ability to format date values in a variety of ways, including month names or abbreviations, days of the week, and 12 or 24-hour time. Since the granularity of this application is one day, the TO_CHAR function has been used here to express the employee's job start and end dates in MM/DD/YYYY format. A four-digit year value is used due to the proximity of the century change at this time.

If the user has queried the database for an active employee (one which is currently employed by the company), the query will return a tuple from the EMP_JOB_SALS table which has a value of 'December 31st, 9999' as the end date for the employee's current job. This is the value entered into the ending date field by the application anytime a user places an employee into a new job within the company thereby, among other operations, inserting a row into the EMP_JOB_SALS table. This value is used to indicate the employee's current job and is meant to have the literal meaning 'until changed'. While this date seems to serve this purpose rather well, it could be confusing to a user. To provide a more meaningful value within the application, Oracle's 'DECODE' function is used.

DECODE performs much like a case statement in programming languages, providing IF-THEN-ELSE logic within a query. In this case DECODE is used to replace the 'until changed' value with the literal string 'Current', denoting to the user that the row with an end date of 'Current' is the employee's present job.

Example 2: Departmental Salaries

While the previous example projected data over valid history, this example provides the user with data at a single point in time or 'snapshot'. The first screen of this function (DEPT_SALARIES.CFM) displays a dropdown box which lists all of the departments listed in the DEPARTMENTS table. This dropdown is dynamically populated by a query in the active server code which selects department name and number, and orders the return set by department name. The user selects the department and clicks a button labeled 'Submit'. When the button is clicked, a parameter containing the department number value is passed to the next screen (DEPT_SALARIES2.CFM). This active server file contains a query which takes the value of the passed parameter and queries the EMPLOYEES, EMP_JOB_SALS, and JOBS tables for employee names, job names, and annual salary. The return set for the query is displayed in a dynamically-built HTML table with the appropriate column headings. The annual salary figures are formatted as currency values with a leading dollar sign.

Example Three: Temporal Organizational Chart

This third example allows the user to view an organizational chart of a selected department at a user-specified point in time. The initial screen for this tool (TEMPORAL_ORG_CHART1.CFM) presents the user with a dynamically-populated dropdown box using functionality similar to the earlier examples. This dropdown contains department name and number, ordered in the query by department name. This screen also features a text input box which prompts the user to enter a date representing the point in time of the desired organizational chart.

To use this tool, the user selects the department from the dropdown, and then enters a date in the text input box. Navigation between these two controls can be accomplished by either mouse or tab key. When both pieces of information are entered, the user clicks a button labeled 'Submit'. This click initiates the passing of department number and the user-supplied date to the second screen (TEMPORAL_ORG_CHART2.CFM).

Two queries are then executed by active server code in the file which creates this second screen. First, since only the department number (not the department name) is passed in as a variable the DEPARTMENTS table is queried for the department name to be used in the display. Also, another query is issued on the EMPLOYEES, JOBS, DEPARTMENTS, EMP_JOBS and EMP_JOBS_SALS tables to collect the data for the organizational chart. Columns returned for this query include employee ID, employee name, salary and job number. Tuples from the return set of the query are presented to the user in a HTML table with column headings.

4.3 System Administration

The application includes separate functionality for use by system administrators. The administration menu is navigated to via a hyperlink appearing on the main menu screen. This link however is visible only to employees listed in the database as system administrators. This is accomplished through the querying of the FL_ADMIN column in the EMPLOYEES_DTL table during the login process. Records representing users with system administrator privileges are indicated accordingly in this column; this value is recorded as a session variable at login. Conditional logic in the MAIN.CFM page determines the visibility of the administration links. On the administration screen, authorized users are presented with two links. The first allows the changing of passwords for application users in the EMPLOYEES_DTL table.

The second option is a utility which allows users to identify and condense records stored in temporal tuples which do not comply with the application's business rules. Although the functionality of the user interface might prevent these tuples from being entered, it is conceivable that not all of the data in the database would have been entered through the user interface. Bulk data transfer tools are typically used to input large quantities of data, usually during off-hours or maintenance windows. Oracle's 'Sqlloader' is an example of one such tool. These utilities do not share the interface's ability to enforce business rules on row by row basis.

It is therefore possible that tuples may appear in the database where the non-temporal columns contain identical data and the periods of validity overlap. This is known as a 'sequenced duplicate'. [Snodgrass 00]

Removing these duplicates is important in order to maintain accurate results in temporal queries. The process of combining tuples with identical non-temporal data and overlapping validity periods is known as 'coalescing'. In the context of this project coalescing involves querying the temporal tables for temporally-overlapping tuples and displaying them for the system administrator. The administrator then has the option of coalescing them at that time or simply noting their presence for disposition at a later time.

Chapter 5

Conclusion

This project has demonstrated some of the difficulties which accompany temporal data management when using current-generation relational database products.

In the database system chosen for this project (Oracle), the vendor has chosen to incorporate only one (1) temporal datatype into the product. In fact, despite the extensive use of this datatype in nearly every table in the application, nowhere was the data in any date column used directly as it is stored in the database. Modifications to some aspect of this data was made by the application , either when being inserted into the database, or retrieved from it, or in some cases, both.

In many mainstream business software applications, the database plays an integral role in the enforcement of business rules and data integrity. Objects such as triggers, constraints, and stored procedures help to ensure that data is meaningful to the task for which it is being stored. In this project however these objects were of little value. Indeed, nearly all of the business rules incorporated into the application were enforced in the application code rather than at the database level.

There appears to be hope for temporal support in the next generation of RDBMS releases on two fronts. First, parts of the standard for SQL3, which has been in development for several years, is nearing the implementation stage. This new standard is composed of ten separate sections. Although the section pertaining to temporal data (section 7) will not be voted upon by the SQL committees until sometime in the year 2001, two of the sections have already been accepted as standards. SQL3 was originally estimated to become a full standard in the mid to late 90's; at least part of the delay is due to its size: the ten sections comprise over 2,000 pages of documentation, while the SQL-92 standard is a mere 580 pages.

Secondly, vendors of commercial RDBMS systems and other software are now becoming keenly aware of the need for inclusion of temporal functionality in their products. While research in the temporal data management field has been ongoing for the past 20 years and industry has demonstrated a clear business need for the use of temporal data, vendors have not devoted substantive energy to incorporating these features other than the datatypes mentioned earlier. While database vendors may elect to wait for complete finalization of the SQL3 standard prior to full-scale inclusion, several middleware products are already starting to appear on the market. These software applications either interface between application programs and the RDBMS database or act as a front end application themselves, providing support for many of the datatypes and semantics proposed in the SQL3 standard. Some of these products are briefly described below:

- TimeDB: This product is marketed by a company known as TimeConsult. It is a middleware application which includes many of the proposed SQL3 constructs. There are two versions, one written in Prolog, and another written in Java.

- Tiger: Tiger is a front-end application which runs with Oracle. Like TimeDB, Tiger is written in Prolog, and also supports SQL3-specified temporal datatypes and semantics.
- Synchrony: Although not directly based on the proposed SQL3 standard, Synchrony uses a graphical query language that generates SQL for use with an RDBMS system. It is designed for data warehousing applications.

References

[Bohlen 96]
Bohlen, Michael H., Richard T. Snodgrass and Michael D. Soo, "Coalescing in Temporal Databases", Proceedings of the 22nd VLDB Conference, Mumbai, India, 1996.

[Clifford 97]
Clifford, James, Curtis Dyreson, Christian S. Jensen and Richard T. Snodgrass, "On the Semantics of 'Now" in Databases", ACM Transactions on Database Systems, 22, 3 (June, 1997), pp. 171-214.

[Date 97]
Date, C.J., and Hugh Darwen, A Guide to The SQL Standard, Addison-Wesley Longman, Inc., Reading, Massachusetts, 1997.

[Feuerstein 97]
Feuerstein, Steven and Bill Pribyl, Oracle PL/SQL Programming, O'Reilly and Associates, Inc., Sebastopol, California, 1997.

[Forta 98]
Forta, Ben, Cold Fusion Web Application Construction Kit, Que, Indianapolis, Indiana, 1998.

[Koch 97]
Koch, George and Kevin Loney, ORACLE 8, The Complete Reference, Osborne McGraw-Hill Publishing, Inc., Berkeley California, 1997.

[Loney 98]
Loney, Kevin, ORACLE 8 DBA Handbook, Osborne McGraw-Hill Publishing, Inc., Berkeley, California, 1998.

[Snodgrass 00]
Snodgrass, Richard T., Developing Time-Oriented Database Applications in SQL, Morgan Kaufmann Publishers, San Francisco, California, 2000.

[Snodgrass 98]
Snodgrass, Richard T., "Querying Valid-Time State Tables", Database Programming & Design, July, 1998, pp. 60-65.

[Taylor 98]
Taylor, Dave and James C. Armstrong, Jr., Teach Yourself UNIX in 24 Hours, Sams Publishing, Indianapolis, Indiana, 1998.

Appendix A

Web Page Source Code

DEPT_ADD.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
        <title>HR-Web Department Insert Screen</title>
</head>
<body>
<H2><CENTER>HR-Web<br>
        Add New Department</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CENTER>Use this screen to insert information about a new department.<br> Then
        enter information in each textbox and click 'Add Department' to insert
        the record.
</CENTER>
<br><br>
<FORM ACTION="dept_add2.cfm" METHOD-"POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="40%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Department Name:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="nm_dept"
                SIZE="20" MAXLENGTH="20"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Department Number:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="num_dept"
                SIZE="20" MAXLENGTH="20"></INPUT></td>
 </tr>
 <tr>
        <td width="50%" align="center"><INPUT TYPE="SUBMIT"
                VALUE="Add Department"></td>
        <td width="50%" align="center"><INPUT TYPE="RESET"
                VALUE="  Clear  "></td>
 </tr>
 </TABLE>
 <br><br>
 <TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
<tr>
 <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
</tr>
</TABLE>
</form>
<br><br><br>
</body>
</html>
```

DEPT_ADD2.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<!---This CFQUERY tag takes the user-supplied variables from the prior screen
(emp_add.cfm) and performs an INSERT into the EMPLOYEES table) --->
<CFQUERY datasource="vega">
INSERT INTO DEPARTMENTS(nm_dept, num_dept)
 VALUES ('#nm_dept#', '#num_dept#')
</CFQUERY>

<html>
<CFOUTPUT>
<title>HR-Web #nm_dept# Department Inserted</title>
</cfoutput>
<body>
<H2><CENTER>HR-Web<br>
        Department Insert Confirmation</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<P><CENTER>
<CFOUTPUT> Department <B>#nm_dept#</B> has been added.
</cfoutput></CENTER>
</P>
<br>
Click below to insert or update another department, or use the link below
to return to the main menu.
<FORM ACTION="dept_add.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Insert
        a new department</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT SIZE=30
        VALUE=" Insert "></INPUT></td>
 </tr>
</TABLE>
</FORM>
<FORM ACTION="dept_upd.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Update an existing
        department</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT
VALUE="Update"></INPUT></td>
  </tr>
</TABLE>
</FORM>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
<br><br>
</body>
</html>
```

DEPT_DATA.CFM

```
<HTML>
<TITLE>HR-Web Department Management</TITLE>
<BODY>
<H2><CENTER>HR-Web<br>
        Department Management</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CENTER>Use this screen to manage departments. Select
an option below to insert or update a specific department and/or its
pertinent information.</CENTER>
<br><br>
<FORM ACTION="dept_add.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Insert
         a new department</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT SIZE=30
         VALUE=" Insert "></INPUT></td>
 </tr>
</TABLE>
</FORM>
<FORM ACTION="dept_upd.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Update an existing
         department</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT
VALUE="Update"></INPUT></td>
 </tr>
</TABLE>
</FORM>
<br>
<br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</BODY>
```

```
</HTML>
```

DEPT_SALARIES.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfquery DATASOURCE="VEGA" NAME="qry_dept1">
SELECT num_dept, nm_dept from DEPARTMENTS
        ORDER BY 2;
</cfquery>
<html>
        <title>HR-Web Departmental Salary Query</title>
<body>
<H2><CENTER>HR-Web<br>
        Departmental Roster Query</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CENTER>Use this form to select a department for which to view current
salary information for each employee. Select a department from the dropdown
list below and click the 'Submit' button.
</CENTER>
<CFFORM ACTION=dept_salaries2.cfm METHOD="post">
<select name="DeptQuery_Dropdown" size="7">
        <option value=" "></option>
<cfloop QUERY="qry_dept1">
        <cfoutput>
        <option value="#qry_dept1.num_dept#">
                #nm_dept#, #num_dept#</option>
        </cfoutput>
</cfloop>
</select>
<P>
<INPUT TYPE="submit" VALUE="Select">
</CFFORM>
<br><br>
 <TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
<br>
</body>
</html>
```

DEPT_SALARIES2.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfset v_QueryNum_Dept = #form.DeptQuery_Dropdown#>
<CFQUERY DATASOURCE="VEGA" NAME="qry_dept_name">
  SELECT nm_dept from departments
        WHERE num_dept = #v_QueryNum_Dept#;
</CFQUERY>
<CFQUERY DATASOURCE="VEGA" NAME="qry_curr_salaries">
SELECT a.nm_emp_first, a.nm_emp_last, substr(d.nm_job, 1,20) nm_job, b.ann_salary
from employees a, emp_job_sals b, jobs d
WHERE b.dt_end = to_date('12/31/9999', 'MM/DD/YYYY')
AND d.num_dept = '#v_QueryNum_Dept#'
AND b.num_job = d.num_job
AND a.emplid = b.emplid
ORDER BY 2;
</CFQUERY>
<html>
<title>HR-Web Departmental Salary Query Result</title>
<body>
<H2><CENTER>HR-Web<br>
        Departmental Roster Query</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<TABLE border="1" cellpadding="1" cellspacing="1" width="80%" align="center">
 <tr bgcolor="CCCC99">
   <cfoutput query="qry_dept_name">
   <td width="100%" align="center" colspan="3">Current Roster and Salaries
       for <FONT COLOR="BLUE">#nm_dept#</FONT> Department</td>
         </cfoutput>
</tr>
<tr>
   <td width="33%" align="center">Name</td>
   <td width="34%" align="center">Job Name</td>
         <td width="33%" align="center">Salary</td>
 </tr>
 <cfoutput query="qry_curr_salaries">
<tr bgcolor="#IIf(CurrentRow mod 2 is 1, DE("CCCC99"), DE("White"))#">
```

```
  <td align="CENTER"><font size="-1">#nm_emp_last#, #nm_emp_first#</font></td>
  <td align="CENTER"><font size="-1">#nm_job#</font></td>
  <td align="CENTER"><font size="-1">$#numberformat(ann_salary)#</font></td>
 </tr>
</cfoutput>
</TABLE>
<br><br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</body>
</html>
```

DEPT_UPD.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfquery DATASOURCE="VEGA" NAME="deptupdate1">
SELECT nm_dept, num_dept from DEPARTMENTS
        ORDER BY 1</cfquery>
<html>
<head>
        <title>HR Web Department Update Query</title>
</head>
<body>
<H2><CENTER>HR-Web<br>
        Update Departments</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
Select a department from the box below to update, and click the Submit button.
<br>
<CFFORM ACTION=dept_upd2.cfm METHOD="post">
<select name="DeptUpdate_Dropdown" size="7">
<cfloop QUERY="deptupdate1">
        <cfoutput>
        <option value="#deptupdate1.nm_dept#">
                #nm_dept#</option>
        </cfoutput>
</cfloop>
</select>
<P>
<INPUT TYPE="submit" VALUE="Submit">
</CFFORM>
</body>
</html>
```

DEPT_UPD2.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfset v_updateDeptNm = #form.DeptUpdate_dropdown#>
<CFQUERY DATASOURCE="VEGA" NAME="DEPTUPDATE_QUERY">
        SELECT nm_dept, num_dept
         FROM departments
         WHERE nm_dept = '#v_updateDeptNm#'</cfquery>
<html>
<head>
        <title>HR Web Department Update Screen</title>
</head>
<body>
<H2><CENTER>HR-Web<br>
        Update Departments</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CFOUTPUT QUERY="DEPTUPDATE_QUERY">
<FORM ACTION="dept_upd3.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="40%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Department Name:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="nm_dept"
                VALUE="#Trim(nm_dept)#" "SIZE="40"
MAXLENGTH="40"></INPUT></td>
 </tr>
 <tr>
        <td width="50%" align="center"><INPUT TYPE="SUBMIT"
                VALUE="Update Department"></td>
        <td width="50%" align="center"><INPUT TYPE="RESET"
                VALUE="  Clear  "></td>
 </tr>
 </TABLE>
 <INPUT TYPE="HIDDEN" NAME="num_dept" VALUE="#num_dept#">
 <br><br>
 <TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</form>
</cfoutput>
```

```
</body>
</html>
```

DEPT_UPD3.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<CFQUERY DATASOURCE="VEGA">
UPDATE jobs
        SET nm_dept='#nm_dept#'
        WHERE num_dept='#num_dept#'
</CFQUERY>
<html>
<CFOUTPUT>
        <title>HR-Web #nm_dept# Department Updated</title>
</cfoutput>
<body>
<H2><CENTER>HR-Web<br>
        Update Departments</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CFOUTPUT>The <B>#nm_dept#</B> department has been updated.
</cfoutput></CENTER>
</P>
<br>
Click below to insert or update another department, or use the link below
to return to the main menu.

<FORM ACTION="dept_add.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Insert
        a new department</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT SIZE=30
        VALUE=" Insert "></INPUT></td>
 </tr>
</TABLE>
</FORM>
<FORM ACTION="dept_upd.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Update an existing
        department</td>
```

```
    <td width="50%" align="center"><INPUT TYPE=SUBMIT
VALUE="Update"></INPUT></td>
  </tr>
</TABLE>
</FORM>
<br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
   <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</body>
</html>
```

EMP_ADD.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<title>HR-Web Employee Insert Screen</title>
</head>
<body>
<H2><CENTER>HR-Web<br>
        Add New Employee</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CENTER>Use this screen to insert information about a new employee.<br> Then
        enter information in each textbox and click 'Add Employee' to insert
        the record.
</CENTER>
<br><br>
<FORM ACTION="emp_add2.cfm" METHOD-"POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="40%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">First Name:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="nm_emp_first"
                SIZE="20" MAXLENGTH="20"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Last Name:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="nm_emp_last"
                SIZE="20" MAXLENGTH="20"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Employee ID:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="emplid"
                SIZE="20" MAXLENGTH="12"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Date
                of Hire (MM/DD/YYYY):</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="dt_hire"
                SIZE="20" MAXLENGTH="20"></INPUT></td>
 </tr>
 <tr>
        <td width="50%" align="center"><INPUT TYPE="SUBMIT"
```

```
                  VALUE="Add Employee"></td>
          <td width="50%" align="center"><INPUT TYPE="RESET"
                  VALUE="  Clear  "></td>
  </tr>
  </TABLE>
  <br><br>
  <TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</form>
<br><br><br>
</body>
</html>
```

EMP_ADD2.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
        <title>HR-Web Employee Insert Screen</title>
</head>
<body>
<H2><CENTER>HR-Web<br>
        Add New Employee</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CENTER>Use this screen to insert information about a new employee.<br> Then
        enter information in each textbox and click 'Add Employee' to insert
        the record.
</CENTER>
<br><br>
<FORM ACTION="emp_add2.cfm" METHOD-"POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="40%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">First Name:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="nm_emp_first"
                SIZE="20" MAXLENGTH="20"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Last Name:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="nm_emp_last"
                SIZE="20" MAXLENGTH="20"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Employee ID:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="emplid"
                SIZE="20" MAXLENGTH="12"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Date
                of Hire (MM/DD/YYYY):</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="dt_hire"
                SIZE="20" MAXLENGTH="20"></INPUT></td>
 </tr>
 <tr>
        <td width="50%" align="center"><INPUT TYPE="SUBMIT"
                VALUE="Add Employee"></td>
```

```
          <td width="50%" align="center"><INPUT TYPE="RESET"
                   VALUE="  Clear  "></td>
  </tr>
  </TABLE>
  <br><br>
  <TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</form>
<br><br><br>
</body>
</html>
```

EMP_DATA.CFM

```
<HTML>
<TITLE>HR-Web Employee Data Management</TITLE>
<BODY>
<H2><CENTER>HR-Web<br>
        Employee Data Management</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CENTER>Use this screen to manage employees and their personal data. Select
an option below to insert or update a specific employee and/or their
pertinent information.</CENTER>
<br><br>
<FORM ACTION="emp_add.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Insert
        a new employee</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT SIZE=30
        VALUE=" Insert "></INPUT></td>
 </tr>
</TABLE>
</FORM>
<FORM ACTION="emp_upd.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Update an existing
        employee</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT
VALUE="Update"></INPUT></td>
 </tr>
</TABLE>
</FORM>
<br>
<br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</BODY>
</HTML>
```

EMP_QUERY.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfquery DATASOURCE="VEGA" NAME="emp_query1">
SELECT nm_emp_last, nm_emp_first, emplid from EMPLOYEES
        ORDER BY 1</cfquery>
<html>
        <title>HR Web - Employee Salary History Query</title>
<body>
<H2><CENTER>HR-Web<br>
        Employee Salary Query</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
Select an employee from the box below to update, and click the Select button.
<br>

<CFFORM ACTION=emp_query2.cfm METHOD="post">
<select name="EmpQuery_Dropdown" size="7">
<cfloop QUERY="emp_query1">
        <cfoutput>
        <option value="#emp_query1.emplid#">
                #nm_emp_last#, #nm_emp_first# #emplid#</option>
        </cfoutput>
</cfloop>
</select>
<P>
<INPUT TYPE="submit" VALUE="Select">
</CFFORM>
<br><br>
 <TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
<br>
</body>
</html>
```

EMP_QUERY2.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfset v_QueryEMPLID = #form.EmpQuery_Dropdown#>
<CFQUERY DATASOURCE="VEGA" NAME="EMPSALQUERY">
 SELECT a.ann_salary, to_char(a.dt_start, 'MM/DD/YYYY') "start",
    decode(to_char(a.dt_end,'MM/DD/YYYY'), '12/31/9999', 'Current',
           to_char(a.dt_end,'MM/DD/YYYY')) "end", b.nm_job
        FROM emp_job_sals a, jobs b
        WHERE emplid = '#v_QueryEMPLID#'
         AND a.num_job = b.num_job
        ORDER BY a.dt_start
</CFQUERY>
<CFQUERY DATASOURCE="VEGA" NAME="EMPNAMEQUERY">
 SELECT nm_emp_first, nm_emp_last
  FROM employees
        WHERE emplid = '#v_QueryEMPLID#'
</CFQUERY>
<html>
        <title>HR Web - Employee Job History Query Result</title>
<body>
<H2><CENTER>HR-Web<br>
        Employee Salary Query</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<TABLE border="1" cellpadding="1" cellspacing="1" width="80%" align="center">
 <tr bgcolor="CCCC99">
   <cfoutput query="EMPNAMEQUERY">
   <td width="100%" align="center" colspan="4">Job History for Employee
       <FONT COLOR="BLUE">#nm_emp_first# #nm_emp_last#</FONT></td>
        </cfoutput>
 </tr>
 <tr>
   <td width="30%" align="center">Job Title</td>
   <td width="20%" align="center">Salary</td>
   <td width="25%" align="center">Start Date</td>
   <td width="25%" align="center">End Date</td>
 </tr>
<cfoutput query="EMPSALQUERY">
 <tr bgcolor="#IIf(CurrentRow mod 2 is 1, DE("CCCC99"), DE("White"))#">
  <td align="CENTER"><font size="-1">#nm_job#</font></td>
  <td align="CENTER"><font size="-1">$#numberformat(ann_salary)#</font></td>
  <td align="CENTER"><font size="-1">#start#</font></td>
  <td align="CENTER"><font size="-1">#end#</font></td>
  </tr>
</cfoutput>
</TABLE>
<br><br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</body>
</html>
```

EMP_UPD.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfquery DATASOURCE="VEGA" NAME="update1">
SELECT nm_emp_last, nm_emp_first, emplid from EMPLOYEES
        ORDER BY 1</cfquery>
<html>
<head>
        <title>HR-Web Employee Update Query</title>
</head>
<body>
<H2><CENTER>HR-Web<br>
        Update Employees</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
Select an employee from the box below to update, and click the Select button.
<br>
<CFFORM ACTION=emp_upd2.cfm METHOD="post">
<select name="Update_Dropdown" size="7">
        <option value=" "></option>
<cfloop QUERY="update1">
        <cfoutput>
        <option value="#update1.emplid#">
                #nm_emp_last#, #nm_emp_first# #emplid#</option>
        </cfoutput>
</cfloop>
</select>
<P>
<INPUT TYPE="submit" VALUE="Select">
</CFFORM>
</body>
</html>
```

EMP_UPD2.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfset v_updateEMPLID = #form.Update_dropdown#>
<CFQUERY DATASOURCE="VEGA" NAME="UPDATE_QUERY">
        SELECT emplid, nm_emp_last, nm_emp_first,
         to_char(dt_hire, 'MM/DD/YYYY') dt_hire,
         to_char(dt_term, 'MM/DD/YYYY') dt_term
         FROM employees
         WHERE emplid = #v_updateEMPLID#</cfquery>
<html>
<head>
        <title>HR-Web Employee Update Screen</title>
</head>
<body>
<H2><CENTER>HR-Web<br>
        Update Employees</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CFOUTPUT QUERY="UPDATE_QUERY">
<FORM ACTION="emp_upd3.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="40%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">First Name:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="nm_emp_first"
              VALUE="#Trim(nm_emp_first)#" "SIZE="20"
MAXLENGTH="20"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Last Name:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="nm_emp_last"
         VALUE="#Trim(nm_emp_last)#" SIZE="20" MAXLENGTH="20"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Hire Date:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="dt_hire"
              VALUE="#Trim(dt_hire)#" SIZE="20" MAXLENGTH="20"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Term Date:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="dt_term"
              VALUE="#Trim(dt_term)#" SIZE="20" MAXLENGTH="20"></INPUT></td>
 </tr>
 <tr>
         <td width="50%" align="center"><INPUT TYPE="SUBMIT"
                 VALUE="Update Employee"></td>
         <td width="50%" align="center"><INPUT TYPE="RESET"
                 VALUE="  Clear  "></td>
 </tr>
 </TABLE>
 <INPUT TYPE="HIDDEN" NAME="emplid" VALUE="#Trim(emplid)#">
 <br><br>
 <TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</form>
</cfoutput>
</body>
</html>
```

EMP_UPD3.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<CFQUERY DATASOURCE="VEGA">
UPDATE employees
        SET nm_emp_first='#nm_emp_first#',
          nm_emp_last='#nm_emp_last#',
                dt_hire=to_date('#dt_hire#','MM/DD/YYYY'),
                dt_term=to_date('#dt_term#','MM/DD/YYYY')
        WHERE emplid='#emplid#'
</CFQUERY>

<html>
<CFOUTPUT>
        <title>HR-Web #nm_emp_first# #nm_emp_last# Updated</title>
</cfoutput>
<body>
<H2><CENTER>HR-Web<br>
        Update Employees</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CFOUTPUT> Employee <B>#nm_emp_first# #nm_emp_last#</B> has been updated.
</cfoutput></CENTER>
</P>
<br>
Click below to insert or update another employee, or use the link below
to return to the main menu.

<FORM ACTION="emp_add.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Insert
        a new employee</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT SIZE=30
        VALUE=" Insert "></INPUT></td>
 </tr>
</TABLE>
</FORM>
<FORM ACTION="emp_upd.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Update an existing
```

```
          employee</td>
    <td width="50%" align="center"><INPUT TYPE=SUBMIT
VALUE="Update"></INPUT></td>
  </tr>
</TABLE>
</FORM>
<br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</body>
</html>
```

EMPJOB_ADD.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfquery DATASOURCE="VEGA" NAME="empjob_query1">
SELECT nm_emp_last, nm_emp_first, emplid from EMPLOYEES
        ORDER BY 1</cfquery>
<cfquery DATASOURCE="VEGA" NAME="empjob_query2">
SELECT a.nm_job, a.descr, a.num_job, b.nm_dept
   FROM JOBS A, DEPARTMENTS B
        WHERE a.num_dept = b.num_dept
        ORDER BY 1</cfquery>
<html>
        <title>HR-Web Employee Position Management</title>
<body>
<H2><CENTER>HR-Web<br>
        Employee Position Management</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CENTER>Use this screen to assign a position to an existing employee.
Choose an employee and job from the lists below, then click the
'Submit' button to proceed to the next step.
</CENTER>
<CFFORM ACTION=empjob_add2.cfm METHOD="post">
<TABLE width="50%" align="center">
<tr bgcolor="3366CC">
 <td width="100%" colspan="2" align="center"><FONT color="white">
  <B>Select an employee and a job to assign to that employee.</B></font>
 </td>
</tr>
<tr bgcolor="CCCC99">
 <td width="100%" align="center">Employees
 </td>
 <td width="100%" align="center">Job / Department
 </td>
</tr>
<tr>
 <td align="left">
<select name="EmpJob_Dropdown1" size="7">
<cfloop QUERY="empjob_query1">
        <cfoutput>
        <option value="#empjob_query1.emplid#">
                #nm_emp_last#, #nm_emp_first# / #emplid#</option>
```

```
          </cfoutput>
</cfloop>
</select>
 </td>
 <td align="right">
 <select name="EmpJob_Dropdown2" size="7">
<cfloop QUERY="empjob_query2">
          <cfoutput>
          <option value="#empjob_query2.num_job#">
                    #nm_job# / #nm_dept#</option>
          </cfoutput>
</cfloop>
</select>
 </td>
</tr>
<tr bgcolor="ffffff"><td><br></td></tr>
<tr bgcolor="3366CC">
 <td width="100%" colspan="2" align="center"><FONT color="white">
 <B>Click here to continue.</B></FONT></td>
  </tr>
<tr>
<tr>
 <td colspan="2" align="center"><INPUT TYPE="submit" VALUE="Submit">
 </td>
</tr>
</TABLE>
</CFFORM>
</body>
</html>
```

EMPJOB_ADD2.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfset v_empjobEMPLID = #form.EmpJob_Dropdown1#>
<cfset v_empjobNUMJOB = #form.EmpJob_Dropdown2#>
<cfquery DATASOURCE="VEGA" NAME="qry_empjob0">
  SELECT a.ann_salary "ann_salary", to_char(a.dt_start, 'MM/DD/YYYY') "start",
     to_char(a.dt_end,'MM/DD/YYYY') "end", b.nm_job "jobname"
          FROM emp_job_sals a, jobs b
          WHERE emplid = '#v_empjobEMPLID#'
           AND dt_end = to_date('12/31/9999','MM/DD/YYYY')
           AND a.num_job = b.num_job;
</cfquery>
<cfquery DATASOURCE="VEGA" NAME="qry_empjob2">
  SELECT nm_emp_first, nm_emp_last,
           to_char(dt_hire, 'MM/DD/YYYY') dt_hire
     FROM employees
           WHERE emplid = '#v_empjobEMPLID#';
</cfquery>
<cfquery DATASOURCE="VEGA" NAME="qry_jobname">
 SELECT nm_job, num_job FROM jobs WHERE num_job = '#v_empjobNUMJOB#';
</cfquery>
<html>
<title>HR-Web Employee Position Management</title>
<body>
<H2><CENTER>HR-Web<br>
        Employee Position Management</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<TABLE border="1" cellpadding="1" cellspacing="1" width="80%" align="center">
 <tr bgcolor="CCCC99">
  <cfoutput QUERY="qry_empjob2">
  <td width="100%" align="center" colspan="4">Relevant Position and Salary
        for employee <FONT COLOR="Blue">#nm_emp_first# #nm_emp_last#,
        </font>hire date: <FONT COLOR="Blue">#dt_hire#</font>
       </cfoutput></td>
 </tr>
 <tr>
       <td align="center">Current Job</td>
       <td align="center">Current Salary</td>
       <td align="center">Job Start Date</td>
       <td align="center">Proposed Job</td>
  </tr>
  <tr>
   <td align="center"><cfif #qry_empjob0.recordcount# LT 1>
              No Current Job
        <cfelse>
         <cfoutput>#qry_empjob0.jobname#</cfoutput>
        </cfif></td>
        <td align="center"><cfif #qry_empjob0.recordcount# LT 1>
              N/A
        <cfelse>
              <cfoutput>$#numberformat(qry_empjob0.ann_salary)#</cfoutput></td>
        </cfif></td>
        <td align="center"><cfif #qry_empjob0.recordcount# LT 1>
              N/A
        <cfelse>
              <cfoutput>#qry_empjob0.start#</cfoutput>
        </cfif></td>
        <td align="center"><CFOUTPUT>#qry_jobname.nm_job#</cfoutput></td>
  </tr>
</TABLE>
<br><br>
<CFFORM ACTION=empjob_add3.cfm METHOD="post">
<TABLE width="50%" align="center">
<tr bgcolor="3366CC">
<tr bgcolor="3366CC">
 <td width="100%" colspan="2" align="center"><FONT color="white">
 <B>If the proposed job listed above is correct, enter the date
 on which this job assignment becomes effective and the new
 salary below, and click on "Submit".</B></FONT></td>
  </tr>
<tr>
 <td width="50%" bgcolor="CCCC99" align="center">Effective
              Date (MM/DD/YYYY):</td>
 <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="dt_start"
              SIZE="20" MAXLENGTH="20"></INPUT></td>
</tr>
<tr>
 <td width="50%" bgcolor="CCCC99" align="center">Salary:</td>
 <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="ann_salary"
              SIZE="20" MAXLENGTH="20"></INPUT></td>
</tr>
<tr></tr>
<tr>
 <td colspan="2" align="center"><INPUT TYPE="submit" VALUE="Submit">
 </td>
</tr>
</TABLE>
<cfoutput query="qry_jobname">
        <INPUT TYPE="hidden" name="nm_job" VALUE="#nm_job#">
        <INPUT TYPE="hidden" name="num_job" VALUE="#num_job#">
</cfoutput>
<cfoutput><INPUT TYPE="hidden" name="emplid" VALUE=#v_empjobEMPLID#>
</cfoutput>
</CFFORM>
<br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
<br><br>
</body>
</html>
```

EMPJOB_ADD3.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfset v_dt_start = '#dt_start#'>
<cfset v_ann_salary = #ann_salary#>
<cfquery DATASOURCE="VEGA" NAME="qry_empjob1">
  SELECT dt_hire, to_char(dt_hire, 'MM/DD/YYYY') char_dt_hire
  FROM employees
                WHERE emplid = #emplid#;
</cfquery>
<cfquery DATASOURCE="VEGA" NAME="qry_empjob2">
  SELECT dt_start from emp_jobs
where dt_end = to_date('12/31/9999', 'MM/DD/YYYY')
AND emplid = #emplid#
AND dt_start > to_date('#dt_start#', 'MM/DD/YYYY');

</cfquery>
<cfquery DATASOURCE="VEGA" NAME="qry_empjob3">
        SELECT dt_start, to_char(dt_start, 'MM/DD/YYYY') char_dt_start
          from emp_jobs
                where dt_end = to_date('12/31/9999', 'MM/DD/YYYY');
</cfquery>
<html>
        <title>HR-Web Employee Position Management</title>
        <body>
<H2><CENTER>HR-Web<br>
        Employee Position Management</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>

<CFIF #dt_start# LT #qry_empjob1.dt_hire#>
 <cfoutput>
  <TABLE border="1" cellpadding="1" cellspacing="1" width="80%" align="center">
  <tr bgcolor="CCCC99">
   <td width="100%" align="center">
  You have entered <FONT COLOR="Blue">#dt_start#</FONT> as the start
  date for this employee's job, which is prior to their hire date of
  <FONT COLOR="Blue">#qry_empjob1.char_dt_hire#</FONT>.
   </td>
        </tr>
        </TABLE>
 </cfoutput>
```

```
<CFFORM ACTION=empjob_add.cfm METHOD="post">
<TABLE width="50%" align="center">
<tr bgcolor="3366CC">
<tr bgcolor="3366CC">
 <td width="100%" colspan="2" align="center"><FONT color="white">
 <B>Click the button below to return to the Employee Job Management
   screen and re-enter your data, or click the link below to return
          to the main menu.</B></FONT></td>
 </tr>
<tr></tr>
<tr>
 <td colspan="2" align="center"><INPUT TYPE="submit" VALUE="Return">
 </td>
</tr>
</TABLE>
</CFFORM>
<br><br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
   <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
<br><br>
<CFELSEIF #qry_empjob2.recordcount# GT 0>
 <cfoutput>
  <TABLE border="1" cellpadding="1" cellspacing="1" width="80%" align="center">
  <tr bgcolor="CCCC99">
   <td width="100%" align="center">
  You have entered <FONT COLOR="Blue">#dt_start#</FONT> as the start
  date for this employee's job, which is prior to the start date for
  their current job, which is
  <FONT COLOR="Blue">#qry_empjob3.char_dt_start#</FONT>. The start
  date for the employee's new job must be later than the start date
  of their current job.
   </td>
         </tr>
         </TABLE>
 </cfoutput>
<CFFORM ACTION=empjob_add.cfm METHOD="post">
<TABLE width="50%" align="center">
 <tr bgcolor="3366CC">
 <tr bgcolor="3366CC">
  <td width="100%" colspan="2" align="center"><FONT color="white">
  <B>Click the button below to return to the Employee Job Management
  screen and re-enter your data, or click the link below to return
         to the main menu.</B></FONT></td>
 </tr>
 <tr></tr>
 <tr>
```

```
   <td colspan="2" align="center"><INPUT TYPE="submit" VALUE="Return">
   </td>
  </tr>
 </TABLE>
 </CFFORM>
 <br>
 <TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
  <tr>
   <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
  </tr>
 </TABLE>
<br><br>
<CFELSE>
 <cfquery DATASOURCE="VEGA" NAME="qry_empjob4">
  UPDATE emp_jobs SET dt_end = (to_date('#dt_start#','MM/DD/YYYY')-1)
    WHERE emplid = '#emplid#'
          AND dt_end = to_date('12/31/9999', 'MM/DD/YYYY');
</cfquery>
<cfquery DATASOURCE="VEGA" NAME="qry_empjob5">
  INSERT INTO emp_jobs (emplid, num_job, dt_start, dt_end,
                 emp_lst_updt, dt_lst_updt)
                 VALUES ('#emplid#', '#num_job#',
                      to_date('#dt_start#','MM/DD/YYYY'), '31-DEC-9999',
                      '#session.v_userEMPLID#', SYSDATE);
</cfquery>
<cfquery DATASOURCE="VEGA" NAME="qry_empjob6">
  UPDATE emp_job_sals SET dt_end = (to_date('#dt_start#','MM/DD/YYYY')-1),
     emp_lst_updt = '#session.v_userEMPLID#',
                 dt_lst_updt = SYSDATE
    WHERE emplid = '#emplid#'
          AND dt_end = to_date('12/31/9999', 'MM/DD/YYYY');
</cfquery>
<cfquery DATASOURCE="VEGA" NAME="qry_empjob7">
  INSERT INTO emp_job_sals (emplid, num_job, ann_salary, dt_start,
                  dt_end, emp_lst_updt, dt_lst_updt)
    VALUES ('#emplid#', '#num_job#', #ann_salary#,
                to_date('#dt_start#','MM/DD/YYYY'),
                           to_date('12/31/9999', 'MM/DD/YYYY'),
                           '#session.v_userEMPLID#', SYSDATE);
</cfquery>
<cfoutput>
Employee Number <B>#emplid#</B> has been assigned the job of
<B>#nm_job#</B>, job number <B>#num_job#</B>, at an annual salary of
<B>#ann_salary#</B>, beginning on <B>#dt_start#</B>.
</cfoutput>
<br><br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
<br><br>
</CFIF>
<br><br>
</body></html>
```

EMPJOB_ADD4.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfquery DATASOURCE="VEGA" NAME="qry_empjob1">
  UPDATE emp_jobs SET dt_end = (to_date('#dt_start#','MM/DD/YYYY')-1)
    WHERE emplid = '#emplid#'
          AND dt_end = to_date('12/31/9999', 'MM/DD/YYYY');
</cfquery>
<cfquery DATASOURCE="VEGA" NAME="qry_empjob2">
  INSERT INTO emp_jobs (emplid, num_job, dt_start, dt_end,
                emp_lst_updt, dt_lst_updt)
                VALUES ('#emplid#', '#num_job#',
                    to_date('#dt_start#','MM/DD/YYYY'), '31-DEC-9999',
                    '#session.v_userEMPLID#', SYSDATE);
</cfquery>
<cfquery DATASOURCE="VEGA" NAME="qry_empjob3">
  UPDATE emp_job_sals SET dt_end = (to_date('#dt_start#','MM/DD/YYYY')-1),
     emp_lst_updt = '#session.v_userEMPLID#',
                dt_lst_updt = SYSDATE
    WHERE emplid = '#emplid#'
          AND dt_end = to_date('12/31/9999', 'MM/DD/YYYY');
</cfquery>
<cfquery DATASOURCE="VEGA" NAME="qry_empjob4">
  INSERT INTO emp_job_sals (emplid, num_job, ann_salary, dt_start,
                 dt_end, emp_lst_updt, dt_lst_updt)
    VALUES ('#emplid#', '#num_job#', #ann_salary#,
              to_date('#dt_start#','MM/DD/YYYY'),
                        to_date('12/31/9999', 'MM/DD/YYYY'),
                        '#session.v_userEMPLID#', SYSDATE);
</cfquery>
<html>
        <title>HR-Web Employee Position Management</title>
        <body>
<H2><CENTER>HR-Web<br>
        Employee Position Management</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
```

```
<FONT SIZE=3>
<cfoutput>
Employee Number <B>#emplid#</B> has been assigned the job of
<B>#nm_job#</B>, job number <B>#num_job#</B>, at an annual salary of
<B>#ann_salary#</B>, beginning on <B>#dt_start#</B>.
</cfoutput>
<br><br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
<br><br>
</body>
</html>
```

ERROR.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<html>
<head>
	<title>HR Web Security Error</title>
</head>

<body>
<br><br>
<H2><CENTER><U>Error Accessing HR Web Application</U></center></h2>
<P>
You have attempted to access a part of HR Web for which you
do not have proper security.</P>
<P>To access this page you will need to log in to HR Web
at the main login page. <A HREF="login.cfm">Click Here</a>
to return to the main login page.

</body>
</html>
```

JOB_ADD.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<CFQUERY DATASOURCE="VEGA" NAME="qry_dept1">
SELECT nm_dept, num_dept FROM DEPARTMENTS
  ORDER BY 1
</CFQUERY>
<html>
<head>
        <title>HR-Web Add New Job</title>
</head>
<body>
<H2><CENTER>HR-Web<br>
        Add New Job</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CENTER>Use this screen to insert information about a new job.<br> Then
        enter information in each textbox and click 'Add Job' to insert
        the record.
</CENTER>
<br><br>
<FORM ACTION="job_add2.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="40%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Job Name:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="nm_job"
                SIZE="40" MAXLENGTH="40"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Job Description:</td>
   <td width="50%" align="center"><TEXTAREA NAME="descr"
                COLS="38" ROWS="5" MAXLENGTH="250"></TEXTAREA></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Department:</td>
   <td width="50%" align="center">
    <select name="Dept_Dropdown1" size="7">
    <cfloop QUERY="qry_dept1">
            <cfoutput>
              <option value="#qry_dept1.num_dept#">
                    #nm_dept#, #num_dept#</option>
            </cfoutput>
```

```
      </cfloop>
      </select>
    </td>
  </tr>
   <tr>
           <td width="50%" align="center"><INPUT TYPE="SUBMIT"
                   VALUE="Add Job"></td>
           <td width="50%" align="center"><INPUT TYPE="RESET"
                   VALUE="  Clear  "></td>
   </tr>
   </TABLE>
   <br><br>
   <TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
  <tr>
    <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
  </tr>
</TABLE>
</form>
<br><br><br>
</body>
</html>
```

JOB_ADD2.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfset v_dept = #form.Dept_Dropdown1#>
<CFQUERY datasource="vega">
INSERT INTO JOBS(num_job, nm_job, descr, num_dept, dt_created,
                                        dt_lst_updt)
 VALUES (seq_num_job.NextVal, '#nm_job#', '#descr#', '#v_dept#',
      SYSDATE, SYSDATE)
</CFQUERY>
<html>
<CFOUTPUT>
<title>HR-Web Job #nm_job# Inserted</title>
</cfoutput>
<body>
<H2><CENTER>HR-Web<br>
        Job Insert Confirmation</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<P><CENTER>
<CFOUTPUT> Job <B>#nm_job#</B> has been added.
</cfoutput></CENTER>
</P>
<br>
Click below to insert or update another job, or use the link below
to return to the main menu.
<FORM ACTION="job_add.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Insert
        a new job</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT SIZE=30
        VALUE=" Insert "></INPUT></td>
 </tr>
</TABLE>
</FORM>
<FORM ACTION="job_upd.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Update an existing
        job</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT
VALUE="Update"></INPUT></td>
 </tr>
</TABLE>
</FORM>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
<br><br>
</body>
</html>
```

JOB_UPD.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfquery DATASOURCE="VEGA" NAME="jobupdate1">
SELECT nm_job, descr, num_job from JOBS
        ORDER BY 1</cfquery>
<html>
<head>
        <title>HR Web Job Update Query</title>
</head>
<body>
<H2><CENTER>HR-Web<br>
        Update Employees</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
Select an job from the box below to update, and click the Select button.
<br>
<CFFORM ACTION=job_upd2.cfm METHOD="post">
<select name="JobsUpdate_Dropdown" size="7">
        <option value=" "></option>
<cfloop QUERY="jobupdate1">
        <cfoutput>
        <option value="#jobupdate1.num_job#">
                #nm_job#</option>
        </cfoutput>
</cfloop>
</select>
<P>
<INPUT TYPE="submit" VALUE="Submit">
</CFFORM>
</body>
</html>
```

JOB_UPD2.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfset v_updateJobNum = #form.JobsUpdate_dropdown#>
<CFQUERY DATASOURCE="VEGA" NAME="JOBUPDATE_QUERY">
        SELECT nm_job, descr, num_job
         FROM jobs
         WHERE num_job = #v_updateJobNum#</cfquery>
<html>
<head>
        <title>HR Web Job Update Screen</title>
</head>
<body>
<H2><CENTER>HR-Web<br>
        Update Employees</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CFOUTPUT QUERY="JOBUPDATE_QUERY">
<FORM ACTION="job_upd3.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="40%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Job Name:</td>
   <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="nm_job"
                VALUE="#Trim(nm_job)#" "SIZE="45" MAXLENGTH="40"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Description:</td>
   <td width="50%" align="center"><textarea cols="38" rows="5"
           name="descr" maxlength="250" value="#descr#">#descr#
                </textarea></td>
 </tr>
 <tr>
        <td width="50%" align="center"><INPUT TYPE="SUBMIT"
                VALUE="Update Job"></td>
        <td width="50%" align="center"><INPUT TYPE="RESET"
                VALUE="  Clear  "></td>
 </tr>
 </TABLE>
 <INPUT TYPE="HIDDEN" NAME="num_job" VALUE="#num_job#">
 <br><br>
 <TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
<tr>
```

```
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</form>
</cfoutput>
</body>
</html>
```

JOB_UPD3.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<CFQUERY DATASOURCE="VEGA">
UPDATE jobs
        SET nm_job='#nm_job#',
          descr='#descr#'
        WHERE num_job='#num_job#'
</CFQUERY>
<html>
<CFOUTPUT>
        <title>HR-Web Job #nm_job# Updated</title>
</cfoutput>
<body>
<H2><CENTER>HR-Web<br>
        Update Jobs</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CFOUTPUT> Job <B>#nm_job#</B> has been updated.
</cfoutput></CENTER>
</P>
<br>
Click below to insert or update another job, or use the link below
to return to the main menu.

<FORM ACTION="job_add.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Insert
        a new job</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT SIZE=30
        VALUE=" Insert "></INPUT></td>
 </tr>
</TABLE>
</FORM>
<FORM ACTION="job_upd.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Update an existing
        job</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT
VALUE="Update"></INPUT></td>
```

```
 </tr>
</TABLE>
</FORM>
<br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</body>
</html>
```

JOBS_DATA.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
        <title>HR-Web Jobs Data Insert Screen</title>
</head>
<body>
<H2><CENTER>HR-Web<br>
        Add New Job</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CENTER>Use this screen to manage jobs and related data. Select
an option below to insert or update a specific job and/or its
pertinent information.</CENTER>
<br><br>
<FORM ACTION="job_add.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Insert
        a new job</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT SIZE=30
        VALUE=" Insert "></INPUT></td>
 </tr>
</TABLE>
</FORM>
<FORM ACTION="job_upd.cfm" METHOD="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Update an existing
        job</td>
   <td width="50%" align="center"><INPUT TYPE=SUBMIT
VALUE="Update"></INPUT></td>
 </tr>
</TABLE>
</FORM>
<br>
<br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
  <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
```

```
</TABLE>
</body></html>
```

LOGIN.CFM

```
<CFSET session.v_login_flag = 0>
<HTML>
<TITLE>HR-Web Login</TITLE>
<BODY>
<H2><CENTER>HR-Web Login Page</CENTER></H2>
<br>
<hr align=CENTER width= "400" height= "5"></hr>
<CENTER>Welcome to the Human Resources Managment Login screen.</CENTER>
<FORM ACTION="main.cfm" method="POST">
<TABLE border="0" cellpadding="5" cellspacing="5" width="50%" align="center">
 <tr bgcolor="CCCC99">
   <td width="100%" colspan="2" align="center">Please enter your User ID and Password in the
boxes below</td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">User ID:</td>
   <td width="50%" align="center"><INPUT TYPE=TEXT SIZE=15
NAME="useremplid"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Password:</td>
   <td width="50%" align="center"><INPUT TYPE=PASSWORD SIZE=15
NAME="userlogin"></INPUT></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Click To Login:</td>
   <td width="50%" align="center"><INPUT TYPE="submit" VALUE="Login"></td>
 </tr>
 <tr>
   <td width="50%" bgcolor="CCCC99" align="center">Click To Clear Fields:</td>
   <td width="50%" align="center"><INPUT TYPE="reset" VALUE="Clear"></td>
 </tr>
</table>
<br>
</FORM>
</BODY>
</HTML>
```

MAIN.CFM

```
<CFSET v_EMPLID = "">
<CFSET v_LOGIN = "">
<CFIF session.v_login_flag EQ 0>
        <CFSET session.v_userEMPLID = #useremplid#>
        <CFSET session.v_userLOGIN = #userlogin#>
</cfif>
<HTML>
<TITLE>HR-Web Main Menu</TITLE>
<BODY>
<H2><CENTER>HR-Web Main Menu</H2>

<!--- QUERY EMPLOYEES_DTL TO DETERMINE IF LOGIN ID AND PASSWORD ARE
OK
         (ONLY AFTER INITIAL LOGIN)--->
<cfif session.v_login_flag EQ 0>
        <cfquery datasource="vega" name="login">
        SELECT emplid, login, fl_admin
                from employees_dtl
        WHERE emplid = '#useremplid#'</cfquery>
<CFSET session.v_fl_admin = #login.fl_admin#>
        <!--- SET VARIABLES TO VALUES SUPPLIED BY USER. IF NO MATCH, SEND
                  USER TO 'ERROR.CFM' TO START LOGIN PROCESS AGAIN --->
        <CFOUTPUT QUERY="login">
                <CFSET v_EMPLID = #emplid#>
                <CFSET v_LOGIN = #login#>
                <CFSET v_fl_admin = #fl_admin#></cfoutput>
        <CFIF v_emplid NEQ #session.v_userEMPLID# OR
            v_LOGIN NEQ #session.v_userLOGIN#>
                   <CFLOCATION URL="error.cfm"></cfif>

        <!--- QUERY EMPLOYEES TABLE TO GET USERS FIRST AND LAST NAMES TO
DISPLAY
            ON EACH PAGE --->
        <cfquery datasource="vega" name="user">
        SELECT nm_emp_first, nm_emp_last from employees
        where emplid = '#useremplid#'</cfquery>
        <cfset session.v_nm_emp_first = #user.nm_emp_first#>
        <cfset session.v_nm_emp_last = #user.nm_emp_last#>
</cfif>
<H3>Please select one of the options below</CENTER></H3>
<TABLE width=100%>
 <tr>
  <td ALIGN=LEFT>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#
  </td>
  <td ALIGN=RIGHT>
  <A HREF=logout.cfm><U>Logout</U></a>
  </td>
</tr>
</TABLE>
</FONT>
</cfoutput>

<hr align=CENTER width= "700" height= "5"></hr>
<FONT SIZE="3">
<br>

<!--- THIS IS THE TABLE CONTAINING LINKS TO OTHER PAGES AND A BRIEF
DESCRIPTION OF THEIR FUNCTION --->
<TABLE border="0" cellpadding="3" cellspacing="3" width="80%" align="center">
 <tr bgcolor="CCCC99">
  <td width="100%" colspan="2" align="center">Data Management</td>
 </tr>
 <tr>
  <td width="40%" align="left"><A HREF="emp_data.cfm">Insert / Update Employees</td>
  <td width="60%" align="left">Maintain data on employees and related
                    information.</td>
 </tr>
 <tr>
  <td width="40%" align="left"><A HREF="jobs_data.cfm">Insert / Update Jobs</td>
  <td width="60%" align="left">Insert and update jobs and position-related
                    information.</td>
 </tr>
 <tr>
  <td width="40%" align="left"><A HREF="comps_data.htm">Insert / Update
Competencies</td>
  <td width="60%" align="left">Create or modify information regarding competencies and
                    related data.</td>
 </tr>
 <tr>
  <td width="40%" align="left"><A HREF="dept_data.cfm">Insert / Update Departments</td>
  <td width="60%" align="left">Create or modify information regarding department names
                    and numbers.</td>
 </tr>
 <tr>
  <td width="40%" align="left"><A HREF="empjob_add.cfm">Employee Position
Management</td>
  <td width="60%" align="left">Assign employees to positions within the company.</td>
 </tr>
</table>
<br>
<TABLE border="0" cellpadding="3" cellspacing="3" width="80%" align="center">
 <tr bgcolor="3366CC">
  <td width="100%" colspan="2" align="center">Query Menu</td>
  </tr>
  <tr>
    <td width="40%" align="left"><A HREF="emp_query.cfm">Employee Position and Salary
History</td>
    <td width="60%" align="left">View an employee's position history by date, as well as salary
                       for each position assignment over time.</td>
  </tr>
  <tr>
    <td width="40%" align="left"><A HREF="dept_salaries.cfm">Current Salaries per
department</td>
    <td width="60%" align="left">View current salary information for each employee within
                       a selected department.</td>
  </tr>
  <tr>
    <td width="40%" align="left"><A HREF="fake.htm">Employee Competency Progress
Evaluation</td>
    <td width="60%" align="left">Examine the specific competencies developed by a given
                       employee over their employement history.</td>
  </tr>
  <tr>
    <td width="40%" align="left"><A HREF="temporal_org_chart1.cfm">Temporal
Organizational Charting</td>
    <td width="60%" align="left">View a departmental roster with salaries for a given
department at a
                       specified point in time.</td>
  </tr>
</table>
<br>
<br>
<cfif #session.v_fl_admin# EQ 'Y'>
        <TABLE border="0" cellpadding="3" cellspacing="3" width="80%" align="center">
  <tr bgcolor="3366CC">
    <td width="100%" colspan="2" align="center">Administration Menu</td>
  </tr>
  <tr>
    <td width="40%" align="left"><A HREF="admin.cfm">Application Admininstration</td>
    <td width="60%" align="left">Perform administration functions for database and
                  application (authorized support staff only)</td>
  </tr>
        </TABLE>
</cfif>
<br><br>
<CENTER><FONT SIZE=1><I>Last Update: April 10th, 2001</I></FONT></CENTER>
<CFSET session.v_login_flag = 1>
</BODY>
</HTML>
```

TEMPORAL_ORG_CHART1.CFM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfquery DATASOURCE="VEGA" NAME="qry_dept1">
SELECT num_dept, nm_dept from DEPARTMENTS
        ORDER BY 2;
</cfquery>
<html>
<title>HR-Web Temporal Department Staffing</title>
<body>
<H2><CENTER>HR-Web<br>
        Departmental Roster Query</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<CENTER>
Use this form to select a department for which you would like to see a
staffing chart for a certain date. You can enter any date in the format
specified below. After providing a department and date simply click the
'Submit' button below to view the staff in that department.
</CENTER>
<CFFORM ACTION=temporal_org_chart2.cfm METHOD="post">
<TABLE width="60%" align="center">
<tr bgcolor="3366CC">
 <td width="100%" colspan="2" align="center"><FONT color="white">
  <B>Click on a department from the box on the left, then enter a
            date in the box at right in 'MM/DD/YYYY' format.</B></font>
 </td>
</tr>
<tr bgcolor="CCCC99">
 <td width="100%" align="center">Departments
 </td>
 <td width="100%" align="center">Enter Date Here
 </td>
</tr>
<tr>
 <td width="50%" align="center">
 <select name="Department_Dropdown" size="7">
        <option value=" "></option>
<cfloop QUERY="qry_dept1">
        <cfoutput>
        <option value="#qry_dept1.num_dept#">
                #nm_dept#, #num_dept#</option>
```

```
          </cfoutput>
</cfloop>
</select>
 </td>
 <td width="50%" align="center"><INPUT TYPE="TEXT" NAME="dt_query"
                    SIZE="20" MAXLENGTH="20"></INPUT></td>
</tr>
<tr>
  <td width="100%" colspan="2" align="center">
          <INPUT TYPE="submit" VALUE="Submit">
  </td>
</tr>
</TABLE>
</CFFORM>
</body>
</html>
```

TEMPORAL_ORG_CHART2.CFM

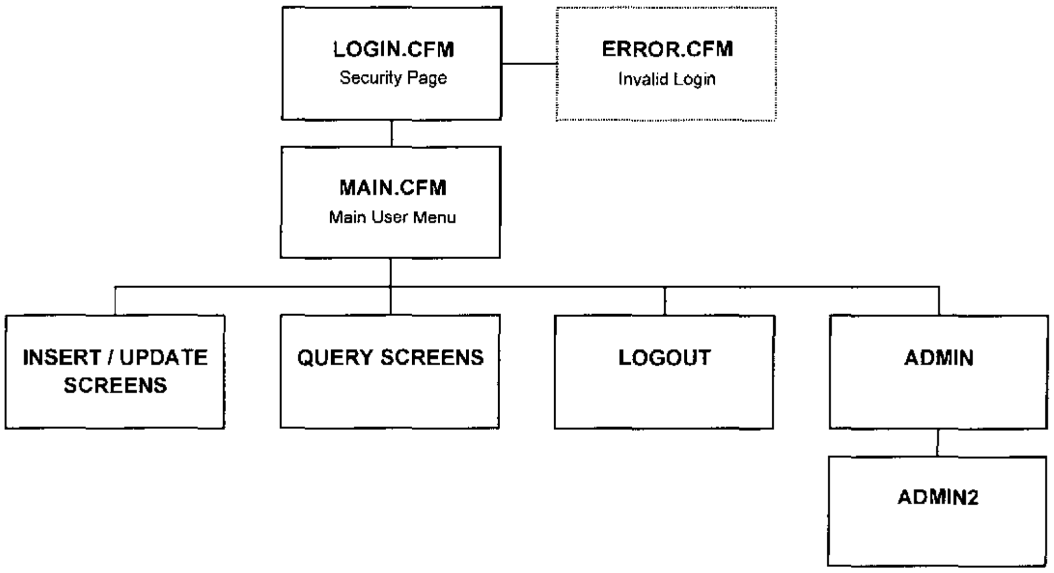
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<cfset v_dt_query = #dt_query#>
<cfset v_dept_num = #form.Department_Dropdown#>
<CFQUERY DATASOURCE="VEGA" NAME="qry_dept_name">
SELECT nm_dept from DEPARTMENTS
WHERE num_dept = #v_dept_num#;
</CFQUERY>
<CFQUERY DATASOURCE="VEGA" NAME="qry_temp_dept1">
SELECT a.emplid, a.nm_emp_last, a.nm_emp_first, e.ann_salary, d.num_job
FROM employees a, jobs b, departments c, emp_jobs d, emp_job_sals e
WHERE c.num_dept = #v_dept_num#
 AND d.dt_start < to_date('#v_dt_query#', 'MM/DD/YYYY')
 AND d.dt_end   > to_date('#v_dt_query#', 'MM/DD/YYYY')
 AND e.dt_start < to_date('#v_dt_query#', 'MM/DD/YYYY')
 AND e.dt_end   > to_date('#v_dt_query#', 'MM/DD/YYYY')
 AND c.num_dept = b.num_dept
 AND b.num_job = e.num_job
 AND d.num_job = e.num_job
 AND a.emplid = d.emplid
 AND a.emplid = e.emplid;
</CFQUERY>
<html>
<head>
        <title>HR-Web</title>
</head>
<body>
<H2><CENTER>HR-Web<br>
        Departmental Roster Query</CENTER></H2>
<br>
<CFOUTPUT>
<RIGHT><FONT SIZE="2">Current User: <FONT SIZE="2" COLOR="BLUE"></RIGHT>
#session.v_nm_emp_first# #session.v_nm_emp_last#</FONT>
</CFOUTPUT>
<br>
<hr align=CENTER width= "700" height= "5"></hr>
<br>
<FONT SIZE=3>
<TABLE border="1" cellpadding="1" cellspacing="1" width="80%" align="center">
 <tr bgcolor="CCCC99">
   <cfoutput query="qry_dept_name">
   <td width="100%" align="center" colspan="4">Staff Roster for
       <FONT COLOR="BLUE">#nm_dept#</FONT> on
                        <FONT COLOR="BLUE">#v_dt_query#</FONT></td>
        </cfoutput>
</tr>
<tr>
   <td width="30%" align="center">Employee ID</td>
```

```
    <td width="20%" align="center">Name</td>
    <td width="25%" align="center">Annual Salary</td>
    <td width="25%" align="center">Job Number</td>
  </tr>
<cfoutput query="qry_temp_dept1">
 <tr bgcolor="#IIf(CurrentRow mod 2 is 1, DE("CCCC99"), DE("White"))#">
  <td align="CENTER"><font size="-1">#emplid#</font></td>
  <td align="CENTER"><font size="-1">#nm_emp_last#, #nm_emp_first#</font></td>
  <td align="CENTER"><font size="-1">$#numberformat(ann_salary)#</font></td>
  <td align="CENTER"><font size="-1">#num_job#</font></td>
  </tr>
</cfoutput>
</TABLE>
<br><br>
<TABLE border="0" cellpadding="0" cellspacing="0" width="40%" align="center">
 <tr>
   <td width="100%" align="center"><A HREF="main.cfm">Return to Main Menu</A></td>
 </tr>
</TABLE>
</body>
</html>
```

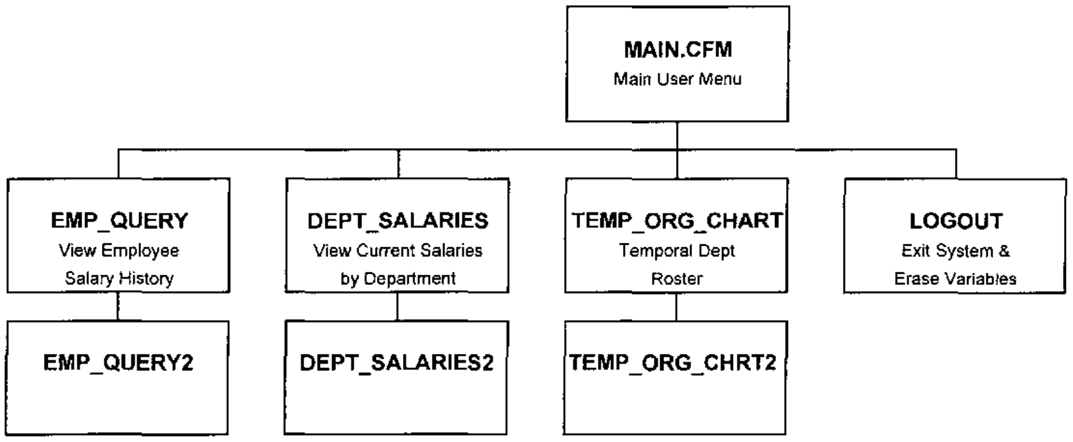
Appendix B

Hierarchy of Web Pages

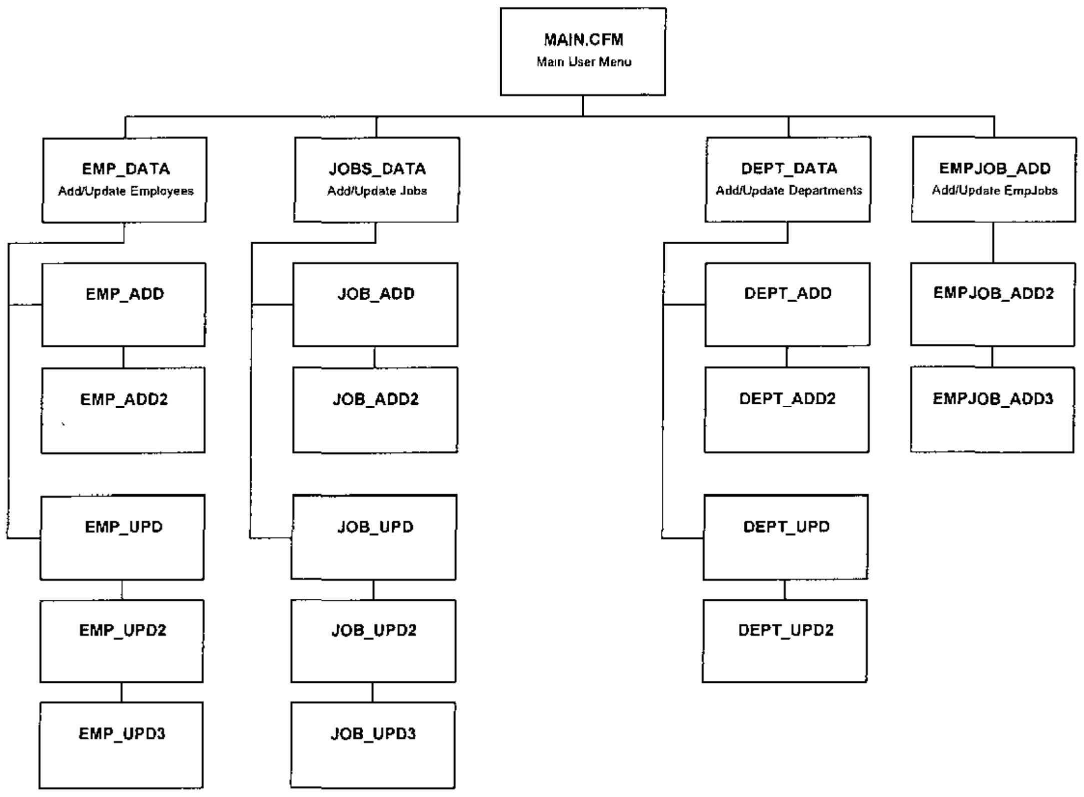
Main Menu



Main Menu to Query Screens



Main Menu to Insert / Update Screens



Appendix C

Database Table Creation Script

```
CREATE TABLE DEPARTMENTS (
NUM_DEPT VARCHAR2(6),
NM_DEPT  VARCHAR2(20) PRIMARY KEY);

CREATE TABLE EMPLOYEES (
EMPLID      VARCHAR2(12) PRIMARY KEY,
NM_EMP_LAST  VARCHAR2(20),
NM_EMP_FIRST VARCHAR2(20),
NO_JOB      VARCHAR2(5),
DT_HIRE     DATE,
DT_TERM      DATE);

CREATE TABLE EMPLOYEES_DTL (
EMPLID   varchar2(12),
LOGIN    varchar2(10),
FL_ADMIN varchar2(1));

ALTER TABLE EMPLOYEES_DTL ADD FOREIGN KEY (EMPLID)
REFERENCES EMPLOYEES.EMPLID;

CREATE TABLE EMP_JOBS (
EMPLID      VARCHAR2(12),
NUM_JOB      VARCHAR2(6),
DT_START    DATE,
DT_END      DATE,
EMP_LST_UPDT VARCHAR2(12),
DT_LST_UPDT  DATE);

ALTER TABLE EMP_JOBS
PRIMARY KEY (EMPLID, NUM_JOB, DT_START, DT_END);

CREATE TABLE EMP_JOB_SALS (
EMPLID VARCHAR2(12),
NUM_JOB  VARCHAR2(6),
ANN_SALARY NUMBER,
DT_START  DATE,
DT_END  DATE,
EMP_LST_UPDT  VARCHAR2(12),
DT_LST_UPDT DATE);

ALTER TABLE EMP_JOB_SALS
PRIMARY KEY (EMPLID, NUM_JOB, ANN_SALARY, DT_START, DT_END);

CREATE TABLE JOBS (
NUM_JOB     VARCHAR2(6) PRIMARY KEY,
NM_JOB      VARCHAR2(40),
DESC        VARCHAR2(250),
LEVEL_JOB   VARCHAR2(2),
RPTS_TO     VARCHAR2(6),
NUM_DEPT    VARCHAR2(6),
DT_CREATED  DATE,
DT_LST_UPDT DATE);

CREATE SEQUENCE SEQ_NUM_JOB INCREMENT BY 1
START WITH 1001;
```

VITA

Joe Brooke is an Oracle database designer and administrator, implementing and supporting various applications on a variety of operating platforms. In addition to research in the field of temporal data modeling, Joe's interests include WAN architecture and computer-based training. Joe is currently employed by Modis Professional Services as a database administrator.

An instrument-rated pilot, Joe devotes much of his spare time to flying. He has flown throughout much of the southeastern United States. He lives in Jacksonville, Florida with his wife Elizabeth and son Joe.