Relational Database Design and Implementation for a Project: Employee Management System

In this project, I will present the tasks performed to design a relational database for a company that deals with multiple projects, employees, and job codes. The tasks included creating a database schema, establishing relationships between tables, inserting sample data, and designing queries to search for specific information.

I have gained practical experience in designing and creating a database for a hypothetical company, as well as performing various tasks related to data manipulation and retrieval using Microsoft SQL Server.

Primary Key: JOB_CODE in JOB Table
Foreign Key: JOB_CODE in EMPLOYEE Table
Foreign Key: ASSIGN_JOB in ASSIGNMENT Table where it does not meet the third normal form

Primary Key: JOB_CHG_HOUR IN JOB Table
Foreign Key: ASSIGN_CHR_HR in ASSIGNMENT Table where the table does not meet the third normal form

Primary Key: EMP_NUM in EMPLOYEE Table
Foreign Key: EMP_NUM in PROJECT Table
Foreign Key: EMP_NUM in ASSIGNMENT Table where the table does not meet the third normal form

Primary Key: PROJ_NUM in PROJECT Table
Foreign Key: PROJ_NUM in ASSIGNMENT Table where the table does not meet the third normal form

Task 1: Table creation statements:

```
CREATE TABLE JOB (
JOB_CODE INT NOT NULL,
JOB_DESCRIPTION VARCHAR(50),
JOB_CHG_HOUR NUMERIC(6,2),
JOB_LAST_UPDATE DATE,
PRIMARY KEY (JOB_CODE)
);

CREATE TABLE EMPLOYEE (
EMP_NUM INT NOT NULL,
EMP_LNAME VARCHAR(15),
EMP_FNAME VARCHAR(15),
EMP_HIRE_DATE DATE,
JOB_CODE INT,
PRIMARY KEY (EMP_NUM),
FOREIGN KEY (JOB_CODE) REFERENCES JOB(JOB_CODE)
);

CREATE TABLE PROJECT (
PROJ_NUM INT NOT NULL,
```

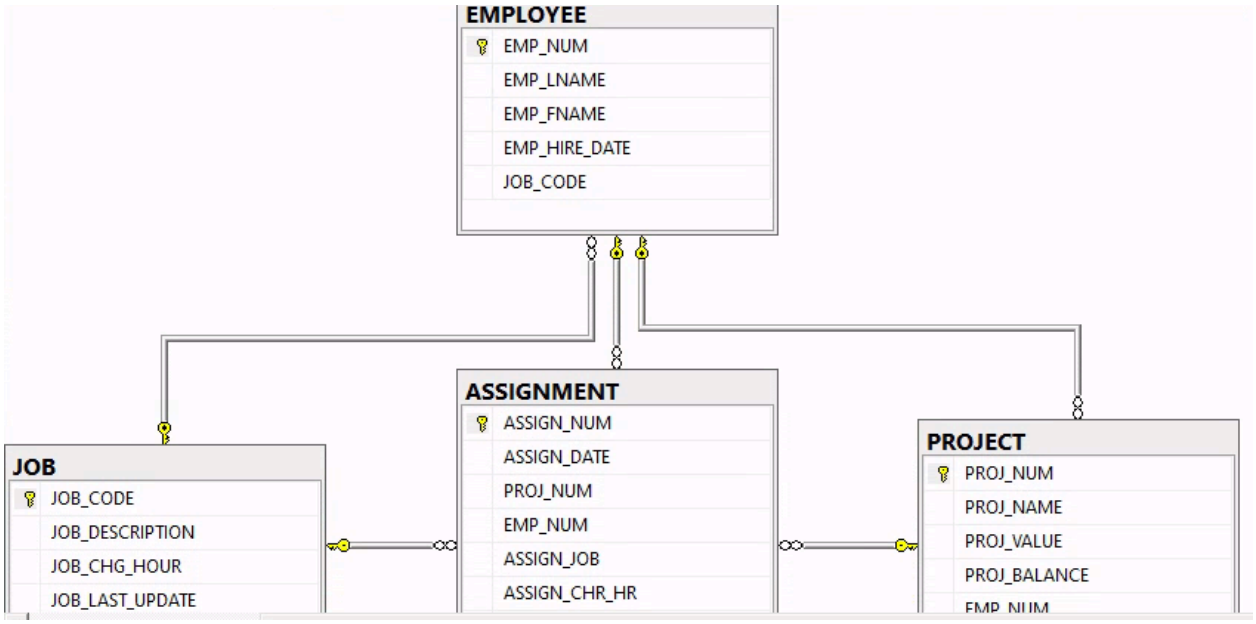
```
PROJ_NAME VARCHAR(30),
PROJ_VALUE NUMERIC(38,2),
PROJ_BALANCE NUMERIC(38,2),
EMP_NUM INT,
PRIMARY KEY (PROJ_NUM),
FOREIGN KEY (EMP_NUM) REFERENCES EMPLOYEE(EMP_NUM)
);

CREATE TABLE ASSIGNMENT (
ASSIGN_NUM INT NOT NULL,
ASSIGN_DATE DATE,
PROJ_NUM INT,
EMP_NUM INT,
ASSIGN_JOB INT,
ASSIGN_CHR_HR NUMERIC(6,2),
ASSIGN_HOURS NUMERIC(5,1),
ASSIGN_CHARGE NUMERIC(6,2),
PRIMARY KEY (ASSIGN_NUM),
FOREIGN KEY (PROJ_NUM) REFERENCES PROJECT(PROJ_NUM),
FOREIGN KEY (EMP_NUM) REFERENCES EMPLOYEE(EMP_NUM),
FOREIGN KEY (ASSIGN_JOB) REFERENCES JOB(JOB_CODE)
);
```

Also used DROP TABLE command to drop PROJECT Table first and then EMPLOYEE Table as mistakes in code were made.

In Task 1, I designed a database schema that includes four tables: EMPLOYEE, PROJECT, JOB, and ASSIGNMENT. The EMPLOYEE table stores information about employees, including their name, employee number, and date of hire. The PROJECT table stores information about the projects, including the project number, project name, and project location. The JOB table stores information about job codes, including the job code and the hourly charge for that job. The ASSIGNMENT table stores information about the assignments of employees to projects, including the assignment number, the assignment date, the project number, the employee number, the job code, the number of hours worked, and the total charge for that assignment. I learned how to create tables and define their attributes, as well as the importance of establishing primary keys for entity integrity control. I also learned how to define relationships between tables using foreign keys for reference integrity control.

Task 2: Created a database diagram that show relationships between tables



Based on the tables and their relationships shown in the database diagram, the following integrity controls could be established:

1. Entity Integrity Control: This ensures that every row in a table is uniquely identifiable, by setting a primary key constraint on the primary key column of each table. In this database, the following primary key constraints have been set:
    - Primary key constraint on JOB_CODE column in JOB table
    - Primary key constraint on EMP_NUM column in EMPLOYEE table
    - Primary key constraint on PROJ_NUM column in PROJECT table
    - Primary key constraint on ASSIGN_NUM column in ASSIGNMENT table
2. Referential Integrity Control: This ensures that the relationships between tables are maintained by enforcing foreign key constraints on the foreign key columns in each table. In this database, the following foreign key constraints have been set:
    - Foreign key constraint on JOB_CODE column in EMPLOYEE table, referencing the JOB_CODE column in JOB table, with ON DELETE and ON UPDATE set to CASCADE. This ensures that when a job is deleted or updated, all employees assigned to that job will also be deleted or updated accordingly.
    - Foreign key constraint on ASSIGN_JOB column in ASSIGNMENT table, referencing the JOB_CODE column in JOB table, with ON DELETE and ON UPDATE set to RESTRICT. This ensures that a job cannot be deleted or updated if there are assignments that reference it.
    - Foreign key constraint on EMP_NUM column in ASSIGNMENT table, referencing the EMP_NUM column in EMPLOYEE table, with ON DELETE and ON UPDATE set to CASCADE. This ensures that when an employee is deleted or updated, all assignments associated with that employee will also be deleted or updated accordingly.
    - Foreign key constraint on PROJ_NUM column in ASSIGNMENT table, referencing the PROJ_NUM column in PROJECT table, with ON DELETE and ON UPDATE set to CASCADE. This ensures that when a project is deleted or updated, all assignments associated with that project will also be deleted or updated accordingly.

3. Domain Integrity Control: This ensures that the data in each column of a table conforms to a specified data type or a range of values, by setting data type constraints or check constraints on each column. In this database, the following domain integrity controls have been established:
    - The data type of ASSIGN_DATE column in ASSIGNMENT table has been set to DATE, to ensure that all records in the table conform to the "Date" data type.

To summarize, the following integrity controls have been established in this database:

- Entity integrity control: primary key constraints on each table
- Referential integrity control: foreign key constraints on each table, with appropriate ON DELETE and ON UPDATE actions
- Domain integrity control: data type constraint on ASSIGN_DATE column in ASSIGNMENT table.

In Task 2, I established relationships between the tables and created constraints for data integrity. Specifically, I established a primary key constraint on the EMPLOYEE, PROJECT, JOB, and ASSIGNMENT tables to ensure unique identification of each record. I also established foreign key constraints between the tables to ensure that data is consistent and accurate. For instance, the foreign key constraint between the EMPLOYEE and ASSIGNMENT tables ensures that an employee can only be assigned to an existing project, and the foreign key constraint between the JOB and ASSIGNMENT tables ensures that a job code can only be assigned if it exists in the JOB table. I further expanded my understanding of database design by creating a diagram to visually represent the relationships between the tables, and by implementing various constraints for integrity control, such as primary key, foreign key, and data type constraints.

Task 3: Insert statements

```
INSERT INTO JOB (JOB_CODE, JOB_DESCRIPTION, JOB_CHG_HOUR, JOB_LAST_UPDATE)
VALUES
('500', 'Programmer', 35.75, '20-Nov-2017'),
('501', 'Systems Analyst', 96.75, '20-Nov-2017'),
('502', 'Database Designer', 125.00, '21-Mar-2018'),
('503', 'Electrical Engineer', 84.50, '20-Nov-2017'),
('504', 'Mechanical Engineer', 67.90, '20-Nov-2017'),
('505', 'Civil Engineer', 55.78, '20-Nov-2017'),
('506', 'Clerical Support', 26.87, '20-Nov-2017'),
('507', 'DSS Analyst', 45.95, '20-Nov-2017'),
('508', 'Applications Designer', 48.10, '21-Mar-2018'),
('509', 'Bio Technician', 34.55, '20-Nov-2017'),
('510', 'General Support', 18.36, '20-Nov-2017')

INSERT INTO EMPLOYEE (EMP_NUM, EMP_LNAME, EMP_FNAME, EMP_INITIAL, EMP_HIREDATE,
JOB_CODE)
VALUES
('101', 'News', 'John', 'G', '08-Nov-2000', '502'),
('102', 'Senior', 'David', 'H', '12-Jul-1989', '501'),
('103', 'Arbough', 'June', 'E', '01-Dec-1996', '500'),
('104', 'Ramoras', 'Anne', 'K', '15-Nov-1987', '501'),
('105', 'Johnston', 'Alice', 'K', '01-Feb-1993', '502'),
('106', 'Smithfield', 'William', NULL, '22-Jun-2004', '500'),
('107', 'Alonzo', 'Maria', 'D', '10-Oct-1993', '500'),
```

('108', 'Washington', 'Ralph', 'B', '22-Aug-1991', '501'),
('109', 'Smith', 'Larry', 'W', '18-Jul-1997', '501'),
('110', 'Olenko', 'Gerald', 'A', '11-Dec-1995', '505'),
('111', 'Wabash', 'Geoff', 'B', '04-Apr-1991', '506'),
('112', 'Smithson', 'Darlene', 'M', '23-Oct-1994', '507'),
('113', 'Joenbrood', 'Delbert', 'K', '15-Nov-1996', '508'),
('114', 'Jones', 'Annelise', NULL, '20-Aug-1993', '508'),
('115', 'Bawangi', 'Travis', 'B', '25-Jan-1992', '501'),
('116', 'Pratt', 'Gerald', 'L', '05-Mar-1997', '510'),
('117', 'Williamson', 'Angie', 'H', '19-Jun-1996', '509'),
('118', 'Frommer', 'James', 'J', '04-Jan-2005', '510');

INSERT INTO PROJECT (PROJ_NUM, PROJ_NAME, PROJ_VALUE, PROJ_BALANCE, EMP_NUM)
VALUES
('15', 'Evergreen', 1453500.00, 1002350.00, '103'),
('18', 'Amber Wave', 3500500.00, 2110346.00, '108'),
('22', 'Rolling Tide', 805000.00, 500345.20, '102'),
('25', 'Star flight', 2650500.00, 2309880.00, '107');

INSERT INTO ASSIGN (ASSIGN_NUM, ASSIGN_DATE, PROJ_NUM, EMP_NUM, ASSIGN_JOB,
ASSIGN_CHR_HR, ASSIGN_HOURS, ASSIGN_CHARGE)
VALUES
('1001', '22-Mar-2018', '18', '103', '500', 35.75, 3.5, 125.13),
('1002', '22-Mar-2018', '22', '117', '509', 34.55, 4.2, 145.11),
('1003', '23-Mar-2018', '18', '117', '509', 34.55, 2.0, 69.10),
('1004', '23-Mar-2018', '18', '103', '500', 35.75, 5.9, 210.93),
('1005', '23-Mar-2018', '25', '108', '501', 96.75, 2.2, 212.85),
('1006', '23-Mar-2018', '22', '104', '501', 96.75, 4.2, 406.35),
('1007', '23-Mar-2018', '25', '113', '508', 48.10, 3.8, 182.78),
('1008', '24-Mar-2018', '18', '103', '500', 35.75, 0.9, 32.18),
('1009', '24-Mar-2018', '15', '115', '501', 96.75, 5.6, 541.80),
('1010', '24-Mar-2018', '15', '117', '509', 34.55, 2.4, 82.92),
('1011', '24-Mar-2018', '25', '105', '502', 125.00, 4.3, 537.50),
('1012', '24-Mar-2018', '18', '108', '501', 96.75, 3.4, 328.95),
('1013', '25-Mar-2018', '25', '115', '501', 96.75, 2.0, 193.50),
('1014', '25-Mar-2018', '22', '104', '501', 96.75, 2.8, 270.90),
('1015', '25-Mar-2018', '15', '103', '500', 35.75, 6.1, 218.08),
('1016', '25-Mar-2018', '22', '105', '502', 125.00, 4.7, 587.50),
('1017', '25-Mar-2018', '18', '117', '509', 34.55, 3.8, 131.29),
('1018', '26-Mar-2018', '25', '117', '509', 34.55, 2.2, 76.01),
('1019', '26-Mar-2018', '25', '104', '501', 96.75, 4.9, 474.08),
('1020', '26-Mar-2018', '15', '101', '502', 125.00, 3.1, 387.50),
('1021', '26-Mar-2018', '22', '108', '501', 96.75, 2.7, 261.23),
('1022', '26-Mar-2018', '22', '115', '501', 96.75, 4.9, 474);

One of my INSERT statements contained an error that I was unable to resolve which led to unsuccessful execution, consequently, the succeeding INSERT statement for other tables was unable to achieve successful execution as well.

INSERT statement for entering my information:

```
INSERT INTO EMPLOYEE (EMP_NUM, EMP_LNAME, EMP_FNAME, EMP_INITIAL,
EMP_HIRE_DATE, JOB_CODE)
VALUES ('119', 'Tazreen', 'Sunehra', 'S', '2023-02-04', '500');
```

I inserted sample data into the tables using SQL insert statements. I was careful not to violate the integrity constraints established in Task 2, such as inserting. I learned how to insert data into the tables while ensuring that the integrity constraints are not violated. I also gained insight into the concept of normalization and how it can be used to reduce data redundancy and improve data integrity.

Task 4:

a. To list employees who were hired before the year 1995:

```
SELECT EMPLOYEE.EMP_NUM, EMPLOYEE.EMP_LNAME, EMPLOYEE.EMP_FNAME,
EMPLOYEE.EMP_INITIAL,  EMPLOYEE.EMP_HIRE_DATE,  EMPLOYEE.JOB_CODE
FROM EMPLOYEE
WHERE EMP_HIRE_DATE < '01-Jan-1995';
```

b. Query to list job code, job description, and employee last name, first name sorted by job description and employee last name:

```
SELECT JOB.JOB_CODE, JOB.JOB_DESCRIPTION, EMPLOYEE.EMP_LNAME,
EMPLOYEE.EMP_FNAME
FROM JOB, EMPLOYEE
WHERE JOB.JOB_CODE = EMPLOYEE.JOB_CODE
ORDER BY JOB_DESCRIPTION, EMP_LNAME;
```

c. Query to list the name of employees who were assigned to the Evergreen project:

```
SELECT EMPLOYEE.EMP_LNAME, EMPLOYEE.EMP_FNAME, EMPLOYEE.EMP_NUM,
PROJECT.EMP_NUM
FROM EMPLOYEE, PROJECT
WHERE EMPLOYEE.EMP_NUM = PROJECT.EMP_NUM
AND ASSIGNMENT.PROJ_NUM = PROJECT.PROJ_NUM
AND PROJECT.PROJ_NAME = 'Evergreen';
```

d. Query to list the project name, the number of employees worked for the project, and the total hours and total charges assigned to each project sorted by project name:

```
SELECT PROJECT.PROJ_NAME, COUNT(ASSIGNMENT.EMP_NUM) AS NUM_EMPLOYEES,
SUM(ASSIGNMENT.ASSIGN_HOURS) AS TOTAL_HOURS, SUM(ASSIGNMENT.ASSIGN_CHARGE)
AS TOTAL_CHARGES
FROM PROJECT, ASSIGNMENT
```

```
WHERE PROJECT.PROJ_NUM = ASSIGNMENT.PROJ_NUM
GROUP BY PROJECT.PROJ_NAME
ORDER BY PROJECT.PROJ_NAME;
```

In Task 4, I designed SQL queries to search for specific information from the database. The queries included listing employees hired before the year 1995, listing job code, job description, and employee last name, first name sorted by job description and employee last name, listing employees assigned to the Evergreen project, and listing project name, the number of employees worked for the project, and the total hours and total charges assigned to each project sorted by project name. I applied my knowledge of SQL to retrieve information from the database using various queries, such as selecting employees hired before a specific year, listing job codes, descriptions, and employee names sorted by job description and last name, and summarizing project information.

In conclusion, I have successfully designed a relational database for a company that deals with multiple projects, employees, and job codes. The database includes four tables with established relationships and constraints to ensure data integrity. I have also inserted sample data and designed queries to search for specific information. Overall, this assignment has provided me with a solid foundation in database design and SQL query language. I have learned how to create a database from scratch, establish relationships between tables, and perform various data manipulation and retrieval tasks using SQL. I believe that these skills will be invaluable for any future work involving databases and data management.