Database Management Using SQL

CIS 331 – Final Project

Aaron McGuire

December 12th, 2010

.

CIS 331 – Database Management Using SQL
Guidelines for Final Project – *Revised on 11/16/09*

Each student must complete an independent Final Project in which a relational database is described and diagramed according to the concepts learned in class. The student will then create tables and a variety of queries using MySQL, and document all work. The database must be done independently, must be efficient in structure, and must be delivered on time.

This is an official assessment project; the core requirements listed below cannot be changed. The instructor may <u>add</u> additional items. The due dates of each phase and the grading system are determined by the instructor.

Phase I: Proposal and Design (25%)

Each student must identify an application or select a topic for his or her database project by the appointed due date. The chosen application should be different from any of the three case studies provided by the textbook. The following are minimum items that must be included in Phase I. The instructor and student should confer with each other regarding the feasibility of the student's plan so that corrections can be made before proceeding.

- A typed, one-page report that describes the business or personal environment, the specific subject matter of the database, the purpose of each table in the database, the useful queries that can be produced from the database, and the advantages that the database has for the owner or operator. Focus on overall information requirements.

- An E-R diagram that appropriately documents the design of the database. The diagram must contain the following items.
    - An appropriate number of related entities/tables (four or more recommended) in third normal form (3NF)
    - Typical attributes of each entity
    - Properly defined primary keys and foreign keys
    - Appropriate data types and lengths
    - Relationships between tables – include both a one-to-many relationship and a many-to-many relationship

A sample E-R diagram is attached to this document. (See Sample #1)

Phase II: Basic Database Development and Construction (40%)

Create the required tables and insert records into the database using MySQL. The following must be included:

- Create and print the script MySQL code for each table.
- Enter a minimum of ten rows/records into each individual table and print a listing of those rows/records. Remember that there should be variety in the content of the sample data in each table to insure that all required queries and tasks can be demonstrated. Good sample data is important in testing the integrity of the overall database design.
- Table design must exhibit functional dependency and must adhere to the rules of third normal form (3NF).
- Relationships between tables must be accurately defined and established.

A sample of how to document the data is attached to this document. (See Sample #2)

Phase III: Query Development and Results (25%)

Each student will create a variety of queries, using both individual and multiple tables to demonstrate each of the following:

- Three queries should demonstrate the ability to use operators such as "And", "Or", and "Not".
- Two queries should demonstrate the use of computed columns.
- Three queries should demonstrate the use of aggregate functions.
- Four of the queries should demonstrate the joining of two tables.
- Demonstrate the ability to sort data.
- Demonstrate the ability to change table structure
- Demonstrate the ability to delete, append, and update data.

Refer to Sample #3 for examples of how to document queries.

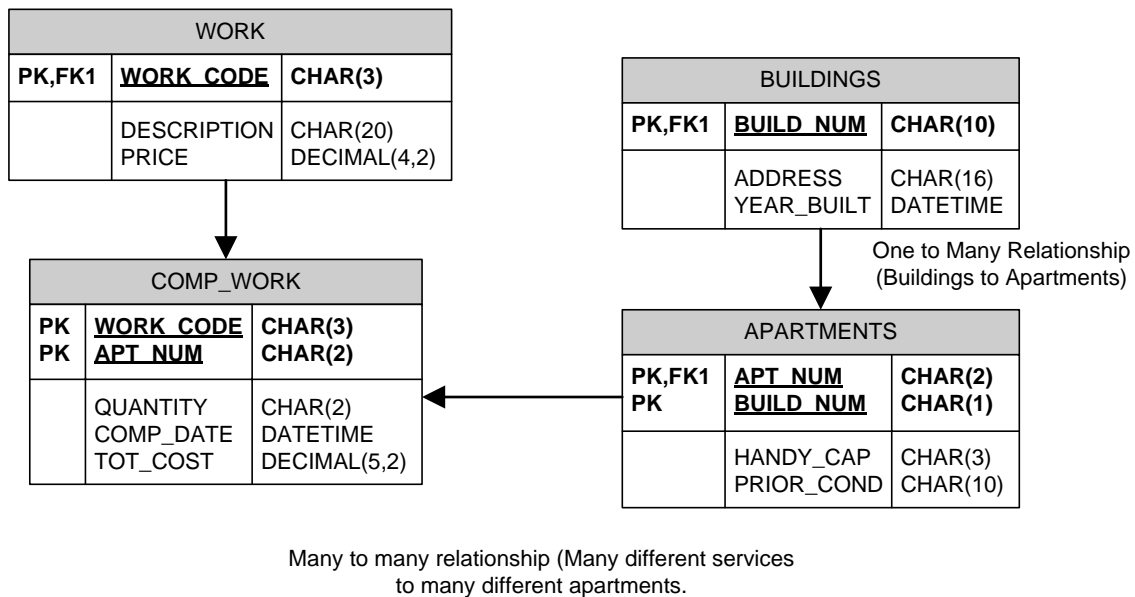**Phase IV: Professional Techniques for Documentation and Completion** (10%)

When finished, the overall project should exhibit the following features:

- Efficiency - database is useful and reusable.

- Independent Work. This is not a team project. Collaboration with the instructor and other students is allowed so long as the concepts from the textbook and class lectures are independently integrated into the project. Direct copying of materials from another person or resource such as the textbook would be considered a violation of the honor code.

- Each aspect of every project phase should be professionally documented including descriptions and explanations of the work that was completed.

- Incorporate the use of professional tools and techniques:
    - Define and utilize views to improve user access
    - Include indexes to improve retrieval efficiency
    - Assign foreign keys to help maintain referential integrity
    - Utilize both the EXPLAIN command and the PROCEDURE ANALYSE() FUNCTION to evaluate query performance
    - Initiate integrity support features to establish and enforce integrity constraints

- Professional Organization and Presentation: all documentation should be collected and neatly organized into a professional looking folder or binder.

- Prompt Delivery: each phase of the project should be delivered on time.

CIS331 Final Project
Phase I
Proposal and Design

Design:

Aaron McGuire
MySQL Final Project
Phase I - CIS 331

WORK		
PK,FK1	WORK_CODE	CHAR(3)
	DESCRIPTION PRICE	CHAR(20) DECIMAL(4,2)

BUILDINGS		
PK,FK1	BUILD_NUM	CHAR(10)
	ADDRESS YEAR_BUILT	CHAR(16) DATETIME

One to Many Relationship
(Buildings to Apartments)

COMP_WORK		
PK PK	WORK_CODE APT_NUM	CHAR(3) CHAR(2)
	QUANTITY COMP_DATE TOT_COST	CHAR(2) DATETIME DECIMAL(5,2)

APARTMENTS		
PK,FK1 PK	APT_NUM BUILD_NUM	CHAR(2) CHAR(1)
	HANDY_CAP PRIOR_COND	CHAR(3) CHAR(10)

Many to many relationship (Many different services
to many different apartments.

Proposal:

For the last 4 years, I have been the painter of Northside Pines Apartments. There I repair and paint apartments before they are leased to new tenants. Sometimes this only requires a touch-up, other times it requires much more work, including drywall repair, painting ceilings and walls, and other need repairs. There are three buildings on the property, and each building contains 6 apartments. They are all 2 bedroom units, and two of them are handicapped accessible.

I am creating a database that will help me keep track of important details about this work, and about each apartment. For instance, it will allow me to keep track of the condition each apartment was left in by the previous tenant, the work required to improve these conditions, and the total amount I charged for the jobs. It can tell me whether or not they are handicap accessible, and the date the buildings were built.

There will be four tables in this database, and they are described as the following:

- *BUILDINGS*: This table assigns a building number to each building, and stores the address and the year each building was built.

- *APARTMENTS*: This table stores the entire apartment numbers according to each building, whether or not they are handicapped accessible, as well as what condition the apartment was left in by the previous tenant.

- *WORK*: This table assigns a work code to the services that I provide, along with the cost and description of these services.

- *COMP_WORK*: This table stores what work I last did in each apartment, the quantity of each services (for example, 2 coats of paint on the walls or ceilings, or more than one 'other repair'". It also stores the date the work was completed, and the total cost of the work listed.

There are endless queries that that can be produced from the database that would be extremely useful. Some of those would include listing the apartments whose ceilings have been painted, or which apartments have had drywall repairs in it. How much it cost to redo the apartment after a tenant moved out On what date was an apartment last worked on, and how much did the work cost? The apartment numbers of the apartments that were only touched up in the last year. This database would benefit me, as well as the owners of Northside Pines Apartments.

Tables

WORK Table

```
CREATE TABLE WORK
(WORK_CODE CHAR(3) NOT NULL,
DESCRIPTION CHAR(20) NOT NULL,
PRICE DECIMAL(4,2) NOT NULL,
PRIMARY KEY(WORK_CODE) );
```

BUILDINGS Table

```
CREATE TABLE BUILDINGS
(BUILD_NUM CHAR(1) NOT NULL,
ADDRESS CHAR(16) NOT NULL,
DATE_BUILT DATE,
PRIMARY KEY(BUILD_NUM) );
```

COMP_WORK Tabe

```
CREATE TABLE COMP_WORK
(WORK_CODE CHAR(3) NOT NULL,
APT_NUM CHAR(2) NOT NULL,
QUANTITY CHAR(2) NOT NULL,
COMP_DATE DATE,
TOT_COST DECIMAL(5,2) NOT NULL,
PRIMARY KEY(WORK_CODE, APT_NUM) );
```

APARTMENTS Table

```
CREATE TABLE APARTMENTS
(APT_NUM CHAR(3) NOT NULL,
BUILD_NUM CHAR(1) NOT NULL,
HANDI_CAP CHAR(3) NOT NULL,
PRIOR_COND CHAR(10),
PRIMARY KEY(APT_NUM, BUILD_NUM) );
```

WORK Table

```
INSERT INTO WORK
VALUES
('TCH','Touch Up', '100.00');
INSERT INTO WORK
VALUES
('DRY','Repair Drywall', '50.00');
INSERT INTO WORK
VALUES
('CEL','Paint Ceilings','150.00');
INSERT INTO WORK
VALUES
('WAL','Paint Walls', '250.00');
INSERT INTO WORK
VALUES
('OTH','Other Repairs','25.00');
```

DESCRIBE WORK;

Field	Type	Null	Key	Default	Extra
WORK_CODE	char(3)	NO	PRI	(null)	(null)
DESCRIPTION	char(20)	NO	(null)	(null)	(null)
PRICE	decimal(6,2)	NO	(null)	(null)	(null)

SELECT * FROM WORK;

WORK_CODE	DESCRIPTION	PRICE
TCH	Touch Up	100.00
DRY	Repair Drywall	50.00
CEL	Paint Ceilings	150.00
WAL	Paint Walls	250.00
OTH	Other Repairs	25.00

BUILDINGS Table

INSERT INTO BUILDINGS
VALUES
('1','1261 Witham Road','1989-01-20');

INSERT INTO BUILDINGS
VALUES
('2','1259 Witham Road','1989-09-24');

INSERT INTO BUILDINGS
VALUES
('3','1257 Witham Road','1990-03-05');

DESCRIBE BUILDINGS;

Field	Type	Null	Key	Default	Extra
BUILD_NUM	char(1)	NO	PRI	(null)	(null)
ADDRESS	char(16)	NO	(null)	(null)	(null)
DATE_BUILT	date	YES	(null)		(null)

SELECT * FROM BUILDINGS;

BUILD_NUM	ADDRESS	DATE_BUILT
1	1261 Witham Road	1989-01-20
2	1259 Witham Road	1989-09-24
3	1257 Witham Road	1990-03-05

APARTMENTS Table

INSERT INTO APARTMENTS
VALUES
('1A','1','YES','Average');
INSERT INTO APARTMENTS
VALUES
('1B','1','YES','Bad');
INSERT INTO APARTMENTS
VALUES
('1C','1','NO','Average');
INSERT INTO APARTMENTS
VALUES
('1D','1','NO','Great');
INSERT INTO APARTMENTS
VALUES
('1E','1','NO','Great');
INSERT INTO APARTMENTS
VALUES
('1F','1','NO','Bad');
INSERT INTO APARTMENTS
VALUES
('2A','2','YES','Bad');
INSERT INTO APARTMENTS
VALUES
('2B','2','YES','Great');
INSERT INTO APARTMENTS
VALUES
('2C','2','NO','Average');
INSERT INTO APARTMENTS
VALUES
('2D','2','NO','Average');
INSERT INTO APARTMENTS
VALUES
('2E','2','NO','Average');
INSERT INTO APARTMENTS
VALUES
('2F','2','NO','Bad');
INSERT INTO APARTMENTS
VALUES
('3A','3','YES','Bad');
INSERT INTO APARTMENTS
VALUES
('3B','3','YES','Average');
INSERT INTO APARTMENTS
VALUES
('3C','3','NO','Bad');
INSERT INTO APARTMENTS
VALUES
('3D','3','NO','Average');
INSERT INTO APARTMENTS
VALUES
('3E','3','NO','Great');
INSERT INTO APARTMENTS
VALUES
('3F','3','NO','Great');

DESCRIBE APARTMENTS;

Field	Type	Null	Key	Default	Extra
APT_NUM	char(3)	NO	PRI	(null)	(null)
BUILD_NUM	char(1)	NO	PRI	(null)	(null)
HANDI_CAP	char(3)	NO	(null)	(null)	(null)
PRIOR_COND	char(10)	YES	(null)		(null)

SELECT * FROM APARTMENTS;

APT_NUM	BUILD_NUM	HANDI_CAP	PRIOR_COND
1A	1	YES	Average
1B	1	YES	Bad
1C	1	NO	Average
1D	1	NO	Great
1E	1	NO	Great
1F	1	NO	Bad
2A	2	YES	Bad
2B	2	YES	Great
2C	2	NO	Average
2D	2	NO	Average
2E	2	NO	Average
2F	2	NO	Bad
3A	3	YES	Bad
3B	3	YES	Average
3C	3	NO	Bad
3D	3	NO	Average
3E	3	NO	Great
3F	3	NO	Great

COMP_WORK Table

```
INSERT INTO COMP_WORK
VALUES
('TCH','1A','1','2010-1-2','100.00');
INSERT INTO COMP_WORK
VALUES
('DRY','1B','2','2010-1-10','100.00');
INSERT INTO COMP_WORK
VALUES
('CEL','1B','2','2010-1-10','200.00');
INSERT INTO COMP_WORK
VALUES
('WAL','1B','2','2010-1-10','500.00');
INSERT INTO COMP_WORK
VALUES
('WAL','1C','1','2010-1-14','250.00');
INSERT INTO COMP_WORK
VALUES
('TCH','1D','1','2010-1-17','100.00');
INSERT INTO COMP_WORK
VALUES
('TCH','1E','1','2010-1-17','100.00');
INSERT INTO COMP_WORK
VALUES
('CEL','1F','1','2010-1-25','150.00');
INSERT INTO COMP_WORK
VALUES
('WAL','1F','2','2010-1-25','500.00');
INSERT INTO COMP_WORK
VALUES
('DRY','2A','1','2010-1-29','50.00');
INSERT INTO COMP_WORK
VALUES
('CEL','2A','1','2010-1-29','150.00');
INSERT INTO COMP_WORK
VALUES
('WAL','2A','2','2010-1-29','500.00');
INSERT INTO COMP_WORK
VALUES
('OTH','2A','2','2010-1-29','50.00');
INSERT INTO COMP_WORK
VALUES
('TCH','2B','1','2010-2-2','100.00');
INSERT INTO COMP_WORK
VALUES
('WAL','2C','1','2010-2-12','250.00');
INSERT INTO COMP_WORK
VALUES
('WAL','2D','1','2010-2-15','250.00');
```

```
INSERT INTO COMP_WORK
VALUES
('WAL','2E','1','2010-2-17','250.00');
INSERT INTO COMP_WORK
VALUES
('DRY','2F','2','2010-2-25','100.00');
INSERT INTO COMP_WORK
VALUES
('CEL','2F','2','2010-2-25','300.00');
INSERT INTO COMP_WORK
VALUES
('WAL','2F','2','2010-2-25','500.00');
INSERT INTO COMP_WORK
VALUES
('OTH','2F','1','2010-2-25','25.00');
INSERT INTO COMP_WORK
VALUES
('DRY','3A','1','2010-3-15','50.00');
INSERT INTO COMP_WORK
VALUES
('CEL','3A','1','2010-3-15','150.00');
INSERT INTO COMP_WORK
VALUES
('WAL','3A','2','2010-3-15','500.00');
INSERT INTO COMP_WORK
VALUES
('WAL','3B','1','2010-3-19','250.00');
INSERT INTO COMP_WORK
VALUES
('DRY','3C','2','2010-3-22','100.00');
INSERT INTO COMP_WORK
VALUES
('CEL','3C','1','2010-3-22','150.00');
INSERT INTO COMP_WORK
VALUES
('WAL','3C','2','2010-3-22','500.00');
INSERT INTO COMP_WORK
VALUES
('WAL','3D','1','2010-3-25','250.00');
INSERT INTO COMP_WORK
VALUES
('OTH','3D','1','2010-3-25','25.00');
INSERT INTO COMP_WORK
VALUES
('TCH','3E','1','2010-3-29','100.00');
INSERT INTO COMP_WORK
VALUES
('TCH','3F','1','2010-3-30','100.00');
```

DESCRIBE COMP_WORK;

Field	Type	Null	Key	Default	Extra
WORK_CODE	char(3)	NO	PRI	(null)	(null)
APT_NUM	char(2)	NO	PRI	(null)	(null)
QUANTITY	char(2)	NO	(null)	(null)	(null)
COMP_DATE	date	YES	(null)		(null)
TOT_COST	decimal(7,2)	NO	(null)	(null)	(null)

SELECT * FROM COMP_WORK;

WORK_CODE	APT_NUM	QUANTITY	COMP_DATE	TOT_COST
TCH	1A	1	2010-01-02	100.00
DRY	1B	2	2010-01-10	100.00
CEL	1B	2	2010-01-10	200.00
WAL	1B	2	2010-01-10	500.00
WAL	1C	1	2010-01-14	250.00
TCH	1D	1	2010-01-17	100.00
TCH	1E	1	2010-01-17	100.00
CEL	1F	1	2010-01-25	150.00
WAL	1F	2	2010-01-25	500.00
DRY	2A	1	2010-01-29	50.00
CEL	2A	1	2010-01-29	150.00
WAL	2A	2	2010-01-29	500.00
OTH	2A	2	2010-01-29	50.00
TCH	2B	1	2010-02-02	100.00
WAL	2C	1	2010-02-12	250.00
WAL	2D	1	2010-02-15	250.00
WAL	2E	1	2010-02-17	250.00
DRY	2F	2	2010-02-25	100.00
CEL	2F	2	2010-02-25	300.00
WAL	2F	2	2010-02-25	500.00
OTH	2F	1	2010-02-25	25.00
DRY	3A	1	2010-03-15	50.00
CEL	3A	1	2010-03-15	150.00
WAL	3A	2	2010-03-15	500.00
WAL	3B	1	2010-03-19	250.00
DRY	3C	2	2010-03-22	100.00
CEL	3C	1	2010-03-22	150.00
WAL	3C	2	2010-03-22	500.00
WAL	3D	1	2010-03-25	250.00
OTH	3D	1	2010-03-25	25.00
TCH	3E	1	2010-03-29	100.00
TCH	3F	1	2010-03-30	100.00

Queries

Question: *Provide a list of handicap accessible apartments whose last known condition was listed as great.*

Answer:
SELECT BUILDINGS.BUILD_NUM, APT_NUM, ADDRESS
FROM APARTMENTS, BUILDINGS
WHERE BUILDINGS.BUILD_NUM = APARTMENTS.BUILD_NUM
AND HANDI_CAP = "Yes"
AND PRIOR_COND = "Great";

BUILD_NUM	APT_NUM	ADDRESS
2	2B	1259 Witham Road

Question: *How much did it cost to repair apartment 2A, and when was it completed?*

Answer:
SELECT APT_NUM, COMP_DATE, SUM(TOT_COST)
FROM COMP_WORK
WHERE APT_NUM = "2A"
GROUP BY APT_NUM;

APT_NUM	COMP_DATE	SUM(TOT_COST)
2A	2010-01-29	750.00

Question: *List the building number, apartment number, and address for each apartment that needed 2 coats of paint on the walls. Sort them by building numbers.*

Answer:
SELECT BUILDINGS.BUILD_NUM, APARTMENTS.APT_NUM, ADDRESS
FROM COMP_WORK, BUILDINGS, APARTMENTS
WHERE BUILDINGS.BUILD_NUM = APARTMENTS.BUILD_NUM
AND APARTMENTS.APT_NUM = COMP_WORK.APT_NUM
AND WORK_CODE = "WAL"
AND QUANTITY = "2"
ORDER BY BUILD_NUM;

BUILD_NUM	APT_NUM	ADDRESS
1	1B	1261 Witham Road
1	1F	1261 Witham Road
2	2A	1259 Witham Road
2	2F	1259 Witham Road
3	3C	1257 Witham Road
3	3A	1257 Witham Road

Question: *List the building number, apartment number, descriptive work done, for each handicapped accessible apartment.*

Answer: SELECT BUILDINGS.BUILD_NUM, APARTMENTS.APT_NUM, DESCRIPTION
FROM BUILDINGS, APARTMENTS, WORK, COMP_WORK
WHERE BUILDINGS.BUILD_NUM = APARTMENTS.BUILD_NUM
AND APARTMENTS.APT_NUM = COMP_WORK.APT_NUM
AND COMP_WORK.WORK_CODE = WORK.WORK_CODE
AND HANDI_CAP = "Yes"
ORDER BY APT_NUM;

BUILD_NUM	APT_NUM	DESCRIPTION
1	1A	Touch Up
1	1B	Paint Ceilings
1	1B	Paint Walls
1	1B	Repair Drywall
2	2A	Paint Ceilings
2	2A	Paint Walls
2	2A	Other Repairs
2	2A	Repair Drywall
2	2B	Touch Up
3	3A	Repair Drywall
3	3A	Paint Ceilings
3	3A	Paint Walls
3	3B	Paint Walls

Question: *Show the building and corresponding apartment number whose last known condition was recorded as "Bad". Sort the results by apartment number.*

Answer: SELECT BUILD_NUM, APT_NUM
FROM APARTMENTS
WHERE PRIOR_COND = "Bad"
ORDER BY APT_NUM;

BUILD_NUM	APT_NUM
1	1B
1	1F
2	2A
2	2F
3	3A
3	3C

Question: *What is the total cost of repairs for every apartment? List the building and apartment number, the address, and the total cost of repairs for each apartment.*

Answer: SELECT BUILDINGS.BUILD_NUM, APARTMENTS.APT_NUM, ADDRESS, SUM(TOT_COST)
FROM APARTMENTS, BUILDINGS, COMP_WORK
WHERE BUILDINGS.BUILD_NUM = APARTMENTS.BUILD_NUM
AND APARTMENTS.APT_NUM = COMP_WORK.APT_NUM
GROUP BY APT_NUM;

BUILD_NUM	APT_NUM	ADDRESS	SUM(TOT_COST)
1	1A	1261 Witham Road	100.00
1	1B	1261 Witham Road	800.00
1	1C	1261 Witham Road	250.00
1	1D	1261 Witham Road	100.00
1	1E	1261 Witham Road	100.00
1	1F	1261 Witham Road	650.00
2	2A	1259 Witham Road	750.00
2	2B	1259 Witham Road	100.00
2	2C	1259 Witham Road	250.00
2	2D	1259 Witham Road	250.00
2	2E	1259 Witham Road	250.00
2	2F	1259 Witham Road	925.00
3	3A	1257 Witham Road	700.00
3	3B	1257 Witham Road	250.00
3	3C	1257 Witham Road	750.00
3	3D	1257 Witham Road	275.00
3	3E	1257 Witham Road	100.00
3	3F	1257 Witham Road	100.00

Question: *How many apartments are there total?*

Answer: SELECT COUNT(APT_NUM) AS "Number of Apartments"
FROM APARTMENTS;

Number of Apartments
18

Question: *What was the cost of repairs for apartment 3A? List the building and apartment numbers, the apartment address, and list the total cost as "Cost of Repairs".*

Answer: SELECT BUILDINGS.BUILD_NUM, APARTMENTS.APT_NUM, ADDRESS, SUM(TOT_COST) AS "Cost of Repairs"
FROM APARTMENTS, BUILDINGS, COMP_WORK WHERE
BUILDINGS.BUILD_NUM = APARTMENTS.BUILD_NUM AND
APARTMENTS.APT_NUM = COMP_WORK.APT_NUM AND
APARTMENTS.APT_NUM = "3A"
GROUP BY APT_NUM;

BUILD_NUM	APT_NUM	ADDRESS	Cost of Repairs
3	3A	1257 Witham Road	700.00

Question: *Add a column to the COMP_WORK table to show whether or not the invoice has been paid.*

Answer: ALTER TABLE COMP_WORK
ADD INVOICE_PAID CHAR(3);

DESCRIBE COMP_WORK;

Field	Type	Null	Key	Default	Extra
WORK_CODE	char(3)	NO	PRI	(null)	(null)
APT_NUM	char(2)	NO	PRI	(null)	(null)
QUANTITY	char(2)	NO	(null)	(null)	(null)
COMP_DATE	date	YES	(null)		(null)
TOT_COST	decimal(7,2)	NO	(null)	(null)	(null)
INVOICE_PAID	char(3)	YES	(null)		(null)

Question: *Add the necessary data to the INVOICE_PAID column of the COMP_WORK table.*

Answer:

```
UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
WHERE APT_NUM = "1A";

UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
WHERE APT_NUM = "1B";

UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
WHERE APT_NUM = "1C";

UPDATE COMP_WORK
SET INVOICE_PAID = "No"
WHERE APT_NUM = "1D";

UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
WHERE APT_NUM = "1E";
```

```
UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
WHERE APT_NUM = "2A";

UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
WHERE APT_NUM = "2B";

UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
WHERE APT_NUM = "2C";

UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
WHERE APT_NUM = "2D";

UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
```

```
UPDATE COMP_WORK
SET INVOICE_PAID = "No"
WHERE APT_NUM = "3A";

UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
WHERE APT_NUM = "3B";

UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
WHERE APT_NUM = "3C";

UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
WHERE APT_NUM = "3D";

UPDATE COMP_WORK
SET INVOICE_PAID =
"Yes"
WHERE APT_NUM = "3E";
```

SELECT * FROM COMP_WORK GROUP BY APT_NUM;

WORK_CODE	APT_NUM	QUANTITY	COMP_DATE	TOT_COST	INVOICE_PAID
TCH	1A	1	2010-01-02	100.00	Yes
DRY	1B	2	2010-01-10	100.00	Yes
WAL	1C	1	2010-01-14	250.00	Yes
TCH	1D	1	2010-01-17	100.00	No
TCH	1E	1	2010-01-17	100.00	Yes
CEL	1F	1	2010-01-25	150.00	Yes
DRY	2A	1	2010-01-29	50.00	Yes
TCH	2B	1	2010-02-02	100.00	Yes
WAL	2C	1	2010-02-12	250.00	Yes
WAL	2D	1	2010-02-15	250.00	Yes
WAL	2E	1	2010-02-17	250.00	Yes
DRY	2F	2	2010-02-25	100.00	No
DRY	3A	1	2010-03-15	50.00	No
WAL	3B	1	2010-03-19	250.00	Yes
DRY	3C	2	2010-03-22	100.00	Yes
WAL	3D	1	2010-03-25	250.00	Yes
TCH	3E	1	2010-03-29	100.00	Yes
TCH	3F	1	2010-03-30	100.00	No

CIS331 Final Project
Phase III
Query Development and Results

Question: *The invoice was paid for Apartment Number 3A. Please update the database to reflect these changes.*

Answer: UPDATE COMP_WORK
SET INVOICE_PAID = "Yes"
WHERE APT_NUM = "3A";

SELECT * FROM COMP_WORK
GROUP BY APT_NUM;

WORK_CODE	APT_NUM	QUANTITY	COMP_DATE	TOT_COST	INVOICE_PAID
TCH	1A	1	2010-01-02	100.00	Yes
DRY	1B	2	2010-01-10	100.00	Yes
WAL	1C	1	2010-01-14	250.00	Yes
TCH	1D	1	2010-01-17	100.00	No
TCH	1E	1	2010-01-17	100.00	Yes
CEL	1F	1	2010-01-25	150.00	Yes
DRY	2A	1	2010-01-29	50.00	Yes
TCH	2B	1	2010-02-02	100.00	Yes
WAL	2C	1	2010-02-12	250.00	Yes
WAL	2D	1	2010-02-15	250.00	Yes
WAL	2E	1	2010-02-17	250.00	Yes
DRY	2F	2	2010-02-25	100.00	No
DRY	3A	1	2010-03-15	50.00	**Yes**
WAL	3B	1	2010-03-19	250.00	Yes
DRY	3C	2	2010-03-22	100.00	Yes
WAL	3D	1	2010-03-25	250.00	Yes
TCH	3E	1	2010-03-29	100.00	Yes
TCH	3F	1	2010-03-30	100.00	No

Question: *Remove the INVOICE_PAID column from the COMP_WORK table.*

Answer: ALTER TABLE COMP_WORK
DROP COLUMN INVOICE_PAID;

DESCRIBE COMP_WORK;

Field	Type	Null	Key	Default	Extra
WORK_CODE	char(3)	NO	PRI	(null)	(null)
APT_NUM	char(2)	NO	PRI	(null)	(null)
QUANTITY	char(2)	NO	(null)	(null)	(null)
COMP_DATE	date	YES	(null)		(null)
TOT_COST	decimal(7,2)	NO	(null)	(null)	(null)

Views

The following view was created to make it easier to show the apartment number, building number, and address of all of the handicap accessible apartments.

```
CREATE VIEW HANDICAP_ACCESSIBLE AS
SELECT BUILDINGS.BUILD_NUM, APARTMENTS.APT_NUM, ADDRESS
FROM APARTMENTS, BUILDINGS
WHERE BUILDINGS.BUILD_NUM = APARTMENTS.BUILD_NUM
AND HANDI_CAP = 'Yes';
```

DESCRIBE HANDICAP_ACCESSIBLE;

Field	Type	Null	Key	Default	Extra
BUILD_NUM	char(1)	NO	(null)	(null)	(null)
APT_NUM	char(3)	NO	(null)	(null)	(null)
ADDRESS	char(16)	NO	(null)	(null)	(null)

SELECT * FROM HANDICAP_ACCESSIBLE;

BUILD_NUM	APT_NUM	ADDRESS
1	1A	1261 Witham Road
1	1B	1261 Witham Road
2	2A	1259 Witham Road
2	2B	1259 Witham Road
3	3A	1257 Witham Road
3	3B	1257 Witham Road

The following view was created to make it easier to show the total cost of repairs for each apartment. It renames the columns as "BNUM, ANUM, ADDR, and AMNT".

```
CREATE VIEW TOTAL_COST (BNUM, ANUM, ADDR, AMNT) AS
SELECT BUILDINGS.BUILD_NUM, APARTMENTS.APT_NUM, ADDRESS, SUM(TOT_COST)
FROM APARTMENTS, BUILDINGS, COMP_WORK
WHERE BUILDINGS.BUILD_NUM = APARTMENTS.BUILD_NUM
AND APARTMENTS.APT_NUM = COMP_WORK.APT_NUM
GROUP BY APT_NUM;
```

DESCRIBE TOTAL_COST;

	Type	Null	Key	Default	Extra
BNUM	char(1)	NO	(null)	(null)	(null)
ANUM	char(3)	NO	(null)	(null)	(null)
ADDR	char(16)	NO	(null)	(null)	(null)
AMNT	decimal(29,2)	YES	(null)		(null)

SELECT * FROM TOTAL_COST;

BNUM	ANUM	ADDR	AMNT
1	1A	1261 Witham Road	100.00
1	1B	1261 Witham Road	800.00
1	1C	1261 Witham Road	250.00
1	1D	1261 Witham Road	100.00
1	1E	1261 Witham Road	100.00
1	1F	1261 Witham Road	650.00
2	2A	1259 Witham Road	750.00
2	2B	1259 Witham Road	100.00
2	2C	1259 Witham Road	250.00
2	2D	1259 Witham Road	250.00
2	2E	1259 Witham Road	250.00
2	2F	1259 Witham Road	925.00
3	3A	1257 Witham Road	700.00
3	3B	1257 Witham Road	250.00
3	3C	1257 Witham Road	750.00
3	3D	1257 Witham Road	275.00
3	3E	1257 Witham Road	100.00
3	3F	1257 Witham Road	100.00

Indexes and Foreign Keys

The following indexes were created to improve retrieval efficiency:

```
CREATE INDEX BLDADR ON BUILDINGS(BUILD_NUM, ADDRESS);

CREATE INDEX DATCOST ON COMP_WORK(APT_NUM, COMP_DATE, TOT_COST);

CREATE INDEX WORKDESC ON WORK(WORK_CODE, DESCRIPTION);

CREATE INDEX PREVCOND ON APARTMENTS(APT_NUM, PRIOR_COND);
```

The following foreign keys were assigned to help maintain referential integrity:

```
ALTER TABLE COMP_WORK
ADD FOREIGN KEY (WORK_CODE) REFERENCES WORK(WORK_CODE);

ALTER TABLE COMP_WORK
ADD FOREIGN KEY (APT_NUM) REFERENCES APARTMENTS(APT_NUM);

ALTER TABLE APARTMENTS
ADD FOREIGN KEY (BUILD_NUM) REFERENCES COMP_WORK(BUILD_NUM);
```

The EXPLAIN Command

Using the EXPLAIN command to evaluate query performance:

EXPLAIN SELECT BUILD_NUM, ADDRESS FROM BUILDINGS;

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	BUILDINGS	index		BLDADR	17		3	Using index

EXPLAIN SELECT BUILDINGS.BUILD_NUM, APARTMENTS.APT_NUM, ADDRESS, SUM(TOT_COST) FROM APARTMENTS, BUILDINGS, COMP_WORK WHERE BUILDINGS.BUILD_NUM = APARTMENTS.BUILD_NUM AND APARTMENTS.APT_NUM = COMP_WORK.APT_NUM GROUP BY APT_NUM;

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	BUILDINGS	index	PRIMARY,BLDADR	BLDADR	17		3	Using index; Using temporary; Using filesort
1	SIMPLE	APARTMENTS	ref	PRIMARY,PREVCOND,BUILD_NUM	BUILD_NUM	1	amcgui01_db.BUILDINGS.BUILD_NUM	3	(null)
1	SIMPLE	COMP_WORK	ref	DATCOST	DATCOST	2	amcgui01_db.APARTMENTS.APT_NUM	4	Using where; Using index

The PROCEDURE ANALYSE() Command

Using the PROCEDURE ANALYSE() command to evaluate query performance:

SELECT DESCRIPTION FROM WORK PROCEDURE ANALYSE();

Field_name	Min_value	Max_value	Min_length	Max_length	Empties_or_zeros	Nulls	Avg_value_or_avg_length	Std	Optimal_fieldtype
amcgui01_db. WORK.DESCRI PTION	Other Repairs	Touch Up	8	14	0	0	12.0000		ENUM('Other Repairs','Paint Ceilings','Paint Walls','Repair Drywall','Touch Up') NOT NULL

SELECT PRIOR_COND FROM APARTMENTS PROCEDURE ANALYSE();

Field_name	Min_value	Max_value	Min_length	Max_length	Empties_or_zeros	Nulls	Avg_value_or_avg_length	Std	Optimal_fieldtype
amcgui01_db.APARTMENTS.PRIOR_COND	Average	Great	3	7	0	0	5.1111		ENUM('Average','Bad','Great') NOT NULL