**CS 317 Exam 2**

**SQL**

SELECT \* FROM table\_name

UPDATE table\_name

SET column = value

WHERE condition

INSERT INTO table\_name (columns)

VALUES (values)

DELETE FROM table\_name WHERE condition

SUM, AVG, COUNT, MIN, MAX

SELECT AVG(column) FROM table\_name

WHERE condition

GROUP BY column

HAVING condition

ORDER BY column ASC (or DESC)

**TRANSACTIONS**

Atomic: whole is done or none is done

Consistent: Database constraints

Isolated: Appears as if one process at a time

Durable: Effects of process survive crash

START TRANSACTION;

UPDATE table\_name SET column = value WHERE condition;

ROLLBACK (or COMMIT);

Commit: Transactions effect are permanent

Rollback: Transactions effects are aborted, no changes

Read Only: Transaction will never effect the database

Dirty Data: uncommited data written by another transaction

Dirty Read: read of dirty data

Serializable: cannot read that has been modified by uncommited transactions

Repeatable Read: cannot read or modify data that has been modified by an uncommited transaction

Read Committed: Can read data from committed transactions

Read Uncommited: Can read data from uncommited transactions

**Views**

A relation build on top of tables or views, does not store data

Can merge multiple tables into one without using the space required for that

CREATE VIEW view\_name AS (

sql query

)

DROP VIEW view\_name

**Triggers and Constraints**

An action that is triggered on event of the associated table

delimiter ||

CREATE TRIGGER trigger\_name AFTER INSERT ON table\_name

FOR EACH RO

BEGIN

INSERT INTO table\_name(columns) VALUES (values)

(or other queries)

END;

||

Foreign Key

Lets you reference multiple tables with a single value

**Stored Procedures and Functions**

Functions must return a value

Functions can only have input paramters, whereas procedures can have input and output parameters

delimiter ||

CREATE PROCEDURE procedure\_name

(

IN value TYPE,

IN value TYPE,

OUT value Type

)

BEGIN

DECLARE variable TYPE;

SET variable = value;

queries

IF () THEN

stuff

ELSEIF () THEN

Stuff

END IF;

END

||

LOOP EXAMPLE:

BEGIN

label1: LOOP

stuff

LEAVE label1;

END LOOP

END

WHILE EXAMPLE:

BEGIN

WHILE () DO

stuff

END WHILE

END

REPEAT LOOP EXAMPLE:

BEGIN

REPEAT

stuff

UNTIL (condition)

END REPEAT

END

delimiter ||

CREATE FUNCTION function\_name(input TYPE) RETURNS TYPE

BEGIN

DECLARE var TYPE;

RETURN (var);

END

||

FETCH FROM table\_name INTO var\_names