Week 5 Tutoring

CSE 180

GROUP BY Warnings

- If SELECT clause has aggregates AGGOP, then c1, c2, …, cm must come from the list of grouping attributes
- Aggregates can't appear in WHERE clauses.
- **The non-aggregate columns in the SELECT clause must come from the attributes in the GROUP BY clause.**

Nulls

NULLs are ignored in any aggregation.

- They do not contribute to the SUM, AVG, COUNT, MIN, MAX of an attribute.
- COUNT(*) = number of tuples in a relation (even if some columns are null)
- COUNT(A) is the number of tuples with non-null values for attribute A

SUM, AVG, MIN, MAX on an empty result (no tuples) is NULL.

COUNT of an empty result is 0.

GROUP BY does not ignore NULLs

HAVING

the HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions.

Query Notation

```
SELECT [DISTINCT] c1, c2, ..., cm AGGOP(...)
FROM R1, R2, ..., Rn
[WHERE condition]
[GROUP BY <list grouping attributes>]
[HAVING condition]
[ORDER BY < list of attributes [ASC | DESC]>]
```

Warning!!

- HAVING cannot exist without GROUP BY
- Aggregates cannot appear in WHERE clause unless in a legal subquery
  - Watch out for this in TRUE/FALSE questions

```
SELECT MIN(S.age)

FROM Sailors S

WHERE S.age > MIN(S.age);
```

Database Modification Statements

- Inserting
- Deleting
- Updating

VS Database Definition (CREATE, DROP, ALTER)

Modification changes the state of the database, does not return tuples

INSERT INTO

INSERT INTO R(A1, ..., An) VALUES (v1, ..., vn);

- If a new row is inserted into a table and a value is specified for an attribute, then that attribute will receive that value

DEFAULT

- If a new row is inserted into a table and there is no value specified for an attribute, give it a default

NULL

- If a new row is inserted into a table and there is no value specified AND the value can be NULL, it will be NULL

Otherwise, it will raise an error

INSERT cont.

Subquery and constants

In Lecture 6 example, we insert the constant 2018 :

```
INSERT INTO Movies(movieTitle, movieYear, length, studioName)
        SELECT dm.movieTitle, 2018, dm.length, 'Disney'
        FROM Disney2018Movies dm;
```

DELETE

Removes a tuple from a table if it follows the condition

DELETE FROM R WHERE <condition>;

- Without the WHERE clause, will delete <u>all</u> instances from R

UPDATE

Update or change the tuples in R that follow the WHERE condition

```
UPDATE R
        SET <new value assignments>
        WHERE <condition>;
```

UPDATE cont.

```
UPDATE Tickets t
SET seatNum = newt.seatNum, paid = FALSE
FROM NewTickets newt
WHERE t.ticketID = newt.ticketID
    AND t.custID = newt.custID
    AND t.airlineID = newt.airlineID
    AND t.flightNum = newt.flightNum;
```

You may use tuple variables in the UPDATE, FROM and WHERE clauses of an UPDATE statement ... but attributes that are SET in the updated table must appear without the tuple variable.

- I.e. seatNum and paid do not have the tuple variable (t)

Semantics

Database modification statements are completely evaluated on the old state of the database

- Yes, it is deterministic

One-Statement-At-a-Time

- SQL statements posed to the database system were executed one at a time, retrieving data or changing the data in the database.

ACID

Atomicity: all or nothing; all changes in a transaction are performed or none

Consistency: domain of attribute must have certain restrictions and constraints

Isolation: whether or not multiple transactions trip over each other

Durability: after a transaction completes, changes to data persist