Week 4: Database Design

Quiz
Q1: C
Q2: A
Q3: B
Q4: A
Q5: B

Anomalies

- **Insert Anomaly** - an anomaly caused by inserting entries that depend on other entries which may not exist yet (ex. inserting a new resident living in a certain state, without having that state in the database yet)
- **Update Anomaly** - an anomaly caused by updating information which may affect the correctness of other data (ex. A resident's local address is changed from Houston to Chicago, but their state of residence remains in Texas)
- **Delete Anomaly** - an anomaly caused by deleting entries that causes removal of other information (ex. a relation containing a resident's state may remove an entire state from the database if the last resident living in a certain state is dropped)

Normalization Theory

- **First Normal Form (1NF)** - all fields are in scalar form (atomic)

```
TABLE Hotel
 id | name            | amenities
----+----------------+-----------
1   | Hilton          | WiFi, Cable, Food
```

    - *Ex. 'Hotels' is not in first normal form because of the 'amenities' field*

```
TABLE Hotel
 id | name            | amenities
----+----------------+-----------
1   | Hilton          | WiFi
2   | Hilton          | Cable
3   | Hilton          | Food
```

    - *Ex. Now that we've decomposed 'amenities', it is.*
- **Second Normal Form (2NF)** - All fields are functionally dependent on the primary key
    - **Functional Dependency** - the quality of a set of data such that if a table agrees that a field A determines another field B, then all corresponding

values in A will result in the same value in B (ex. A city *should* functionally depend on the state, because every table that stores a city should agree that it is from the same state)

```
TABLE Food
 id | food_name | random_expression
----+-----------+---------------------------------------
1   | Spaghetti | y = Ax + b
2   | Turkey    | (x - y)(x + y) = x^2 + y^2
```

  - *Ex. 'random_expression' has absolutely nothing to do with the primary key for 'Food'*

```
TABLE Food
 id | food_name |
----+-----------+
1   | Spaghetti |
2   | Turkey    |

TABLE Expression
 id | expression
----+----------------
1   | y = Ax + b
2   | (x - y)(x + y) = x^2 + y^2
```

  - *Ex. We've separated them into another group of tables now*

- **Third Normal Form (3NF)** - There are no fields that are functionally dependent on other non-key attributes

```
TABLE Student
 id | f_name    | mom_name          | mom_relationship_status
----+-----------+-------------------+------------------------
1   | Jason     | Martha Jones      | married
2   | Robert    | Sarah Palin       | its_complicated
3   | Nora      | Martha Jones      | married
```

  - *Ex. 'Students' is not in third normal form because mom_relationship_status is functionally dependent on mom_name.*

```
TABLE Student
 id | f_name    | mom_id
----+-----------+--------
1   | Jason     | 1
2   | Robert    | 2
3   | Nora      | 1

TABLE Mom
```

```
 id | mom_name         | mom_relationship_status
----+-----------------+------------------------
1   | Martha Jones    | married
2   | Sarah Palin     | its_complicated
```

- *Ex. Now that we've resolved the functional dependency, it is.*

<u>More SQL</u>

- **CREATE TABLE AS SELECT** - Creates a table based off of the nested SELECT query statement, with the returned columns being the only fields in the new table.

```
CREATE TABLE Student AS SELECT id, name FROM People WHERE
role = 'student';
```

  - *Ex.* Creates a table 'Student' that contains all the records from People where their role is a student, with only their id and their name.

- **INSERT INTO** - Inserts values into a table based on certain values

```
INSERT INTO Student (id, name) VALUES (36, 'Jason');
```

  - *Ex.* Inserts a new student with id 36, named Jason into the Student table.
  - This function can also handle nested queries by typing **INSERT INTO** Table (field1, field2, …) **SELECT** …

- **DELETE** - Deletes fields from a table via a query.

```
DELETE FROM Student WHERE name = 'Jason'
```

  - *Ex.* Removes all students with the name 'Jason' from the table Student

- **ALTER TABLE DROP/ADD COLUMN** - Removes all of a certain field from a table, or adds a number of columns to a table.

```
ALTER TABLE Student DROP COLUMN mom_id;
```

  - *Ex.* Removes the column mom_id and all of the data of each record from the table Student.

- **UPDATE** - Updates a table. Pretty broad, but here's a specific use case:

```
UPDATE Student SET last_name = mom_last_name;
```

  - *Ex.* Sets all the students' last names to their mother's.