Non-First Normal Form Data

New material NOT on CSE 180 Final!

Instructor: Shel Finkelstein

Important Notices

- Lecture slides and other class information will be posted on Piazza (not Canvas).
  - Slides are posted under Resources→Lectures
  - Lecture Capture recordings are available to all students under Yuja.
    - That includes classes given over Zoom.
  - There's no Lecture Capture for Lab Sections.
- All Lab Sections (and Office Hours) normally meet In-Person, with rare exceptions announced on Piazza.
  - Some slides for Lab Sections have been posted on Piazza under Resources→Lab Section Notes
- Office Hours for TAs and me are posted in Syllabus and Lecture1, as well as in Piazza notice @7; that post also now includes hours for Group Tutors.
- Some suggestions about use (and non-use) of Generative AI systems such as ChatGPT were posted on Piazza in @41, with specific discussion about the Movies tables and the four Avatar queries in Lecture 4.

Important Notices

- The Fifth Gradiance Assignment, CSE 180 Winter 2024 #5, on Design Theory, was assigned on Friday, March 8.
  - It is due by 11:59pm on **Friday, March 15**, the last day of class.
- Explanation of two difficult Gradiance questions on transactions was posted on Piazza on Sunday, March 2 in @122.
- Answers for two of the "Practice" Relational Algebra queries (in magenta) were posted on Piazza on Sunday, March 2 in @126.
- Winter 2024 Student Experience of Teaching Surveys - SETs opened on Monday, March 4.
  - SETs close on Sunday March 17 at 11:59pm.
  - Instructors are not able to identify individual responses.
  - Constructive responses help improve future courses.

Important Notices

- Lab4 was due on **Tuesday, March 12 by 11:59pm**.
  - Our Lab4 solution was posted on Piazza on Wednesday, March 13.
- Lab Sections, Office Hours and Tutoring continue throughout the last week of classes, even though Lab Assignments are complete.
  - Topics may include Lab4 solution, Gradiance #5, the W23 CSE 180 Final, and recent Lecture material.
- Lab Section exchange during last week of classes:
  - Utkarsh will teach Nayan's 01C Lab Section, Wed 3:30PM - 5:05PM on 3/13.
  - Nayan will teach Utkarsh's 01D Lab Section, Thu 1:30PM to 3:05PM on 3/14.
- Plan for lecture on last day of classes, Friday, March 15
  - Will complete Lecture 14 (OLAP), if it wasn't completed on Wednesday, March 13.
  - Will go over the two "Practice" Relational Algebra queries (Lecture 9, in magenta) that were posted on Piazza in @126.
  - Will review discussion of BCNF and 3NF from Lecture 13.
  - May begin Lecture 15 (Non-First Normal Form), but new material in Lecture 15 will not be included on the CSE 180 Final.

Important Notices: Final

- CSE 180 Final is on **Wednesday, March 20, 8:00-11:00am, and it will be given in-person in our classroom, except for students who receive Remote Exam permission.**
  - **No** early/late Exams. **No** make-up Exams. **No** devices (except for Remote students).
  - 3 hours, extended for DRC students, covering the entire quarter.
    - All DRC students should have recently received email about the final.
    - Final will be harder than the Midterm.
  - **You may bring one double-sided 8.5 x 11 sheet to the Final, with anything that you want written or printed on it that you can read unassisted) …**
    - (Okay, you may bring two sheets with writing only on one side of each sheet.)
    - **But you must not use any other material or receive any help during the Final, whether you're in the classroom, remote, or in a DRC room**
    - As the Syllabus emphasizes, Academic Integrity violations have serious consequences!
  - We will assign seats as you enter the room. You may not choose your own seat.
- CSE 180 Final from Winter 2023 was posted on Piazza under Resources→Exams on Sunday, March 3.
  - Solution to that Final was posted on Piazza on Sunday, March 10 … but take it yourself first, rather than just reading the solution.

Important Notices: In-Person Final

Final includes a Multiple Choice Section and a Longer Answers Section.

- In-Person students should bring a Red Scantron sheet (ParSCORE form number f-1712) sold at Bookstore for about 25 cents, and #2 pencils for Multiple Choice Section.
- You'll answer Multiple Choice Section on your Scantron sheet, entering your name, your student ID, and which version ("Test Form Letter") of the Multiple Choice Section you're answering.
  - Write your Multiple Choice answers on the Scantron sheet using a #2 pencil.
  - Ink and #3 pencils don't work.
- You'll answer Longer Answers Section on the Long Answers Section, using either ink or #2 pencil (as on the Midterm).
- Be sure to answer all questions readably!!
- Do not hand in your 8.5 x 11 sheet or your Multiple Choice Section. Just hand in your Scantron Sheet and your Longer Answers Section.
  - But show us your Multiple Choice Section, so that we can check that you filled in the Test Form Letter (A, B, C or D) for that Section on your Scantron Sheet.

Important Notices: Remote Final

CSE 180 Final **will be given in-person in our classroom, except for students who receive Remote Exam permission.**

- If you need to take the Final Remotely, send me an email whose subject is "Taking the CSE 180 Final Remotely" by Tuesday, March 19 at 6:00pm that includes your strong justification for that request.
  - Only students whose requests are approved may take the Final Remotely.
    - If you don't receive a response from me by 9:00pm, please send email again!
- I'll send instructions to you for taking the Final before 8:00am on Wednesday, March 20 which include a Zoom link
  - You must be on Zoom while you are taking the Final.
    - Please make sure that your face is visible on Zoom video, but that your microphone is muted.
    - Please do not have headphones on during the Final.
    - If there are any bugs on the Final, they will be conveyed to all Remote students via Zoom Chat.
    - If you have a question during the Final, please send it to me (just to me) directly using Zoom Chat, not by speaking.
  - You'll have to complete the exam by 11:00am, just like all other students …
  - … except for DRC students, who will receive extra time, whether they take the exam In-Person (in a different room) or Remotely.
  - You won't need a red Scantron Sheet if you're taking the Final Exam Remotely.

A Word to the Unwise

- This is a tough class for some students since it involves a combination of theory and practice.
  - The second half of CSE 180 is much harder than the first half of the course.
- Students who regularly attend Lectures and Lab Sections (and Office Hours and Tutoring) often do well; students who don't regularly attend often do poorly.
  - We don't take attendance; you're responsible for your own choices.
- After the course ends, your course grade will be determined by your scores on Exams, Labs and Gradiance, as described on the “Course Evaluation” and “Grading” Slides in the Syllabus.
  - You won't be able to do any additional work to improve your grade.

Semi-Structured Data Models

- In the Relational Database Management System, a schema must be defined *before* data can be stored.
  - **Schema First!**
  - Rigid schema is known to the optimizer and query processor.
  - The rigid schema is exploited to derive efficient plans to access and update data.
- In a Semi-Structured data model (e.g., XML and JSON), the schema need not be fully defined prior to "data creation".
  - **Data First!**
  - Flexible data model: "Unexpected" attributes may appear if needed!
  - Semi-structured data is sometimes called "self-describing".
    - Actually, the applications need to figure out what to do with the data.
  - Semi-Structured Data Models tend to be Hierarchical.
    - **XML and JSON support non-First Normal Form data!**

First Normal Form (1NF)

- A relation schema is in *first normal form (1NF)* if the type of every attribute is atomic.
  - **No arrays, no sets, no sequences, no records/structures.**
- Very basic requirement of the relational data model.
  - Not based on FDs.
  - **Every other Normal Form that we'll discuss assumes 1NF.**

First Normal Form example:

Employees(ssn: char(9), name: string, age: int)

- All our relational examples so far have been in 1NF.

Example of Non-First Normal Form:

Employees(ssn: char(9), name: Record[firstname: string, lastname: string], age: int, children: Set(string))

XML/JSON and Relational DBMS

- Every major relational DBMS can store and query XML and JSON data, together with relational data.
  - Standards have been defined for rich capabilities.
  - PostgreSQL can store XML and JSON types.
- Some NOSQL systems ("Not Only SQL") focus on JSON.
  - XML seemed important initially, but JSON which is simpler and more efficient, is much more popular, for both messaging and data storage.
  - And there are other standard non-first normal form data formats in both row-format and column-format.

What is JSON?

JSON is…

- A lightweight text based data-interchange format
- Completely language independent
- Based on a subset of the JavaScript Programming Language
- Easy to understand, manipulate and generate

JSON is NOT…

- Overly Complex
- A “document” format
- A markup language
- A programming language

Why use JSON?

- Straightforward syntax
- Easy to create and manipulate
- Can be natively parsed in JavaScript using **eval()**
- Supported by all major JavaScript frameworks
- Supported by most backend technologies

JSON Object Syntax

- Unordered sets of name/value pairs
- Begins with **{** (left brace)
- Ends with **}** (right brace)
- Each name is followed by **:** (colon)
- Name/value pairs are separated by **,** (comma)

JSON Example

```
var employeeData = {
  "employee_id": 1234567,
  "name": "Jeff Fox",
  "hire_date": "1/1/2013",
  "location": "Norwalk, CT",
  "consultant": false
};
```

Arrays in JSON

- An ordered collection of values
- Begins with [(left bracket)
- Ends with] (right bracket)
- Name/value pairs are separated by , (comma)

JSON Array Example

```
var employeeData = {
  "employee_id": 1236937,
  "name": "Jeff Fox",
  "hire_date": "1/1/2013",
  "location": "Norwalk, CT",
  "consultant": false,
   "random_nums": [ 24,65,12,94 ]
};
```

JSON Data Types: Strings

- Sequence of zero or more Unicode characters
- Wrapped in "double quotes"
- Backslash escapement

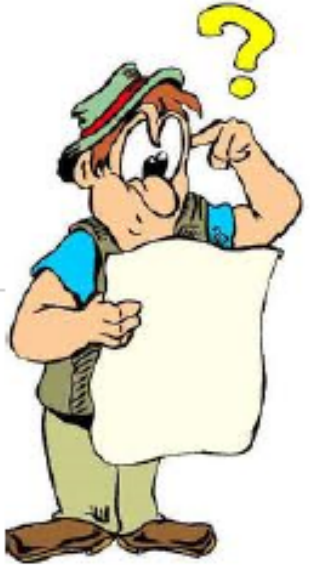
JSON Data Types: Numbers

- Integer
- Real
- Scientific
- No octal or hex
- Use **null** where JavaScript would use NaN (Not a Number)

Let's end with an example JSON

```
{     "firstName": "John",
      "lastName": "Smith",
      "age": 25,
      "address":     {
             "streetAddress": "21 2nd Street",
             "city": "New York",
             "state": "NY",
             "postalCode": "10021"
             },
      "phoneNumber": [
                 {
                 "type": "home",
                 "number": "212 555-1234"
                 },
                 {
                 "type": "fax",
                 "number": "646 555-4567"
                 }
             ]
}
```

The same Example in XML



```
<Object>
<Property><Key>firstName</Key> <String>John</String></Property>
<Property><Key>lastName</Key> <String>Smith</String></Property>
<Property><Key>age</Key> <Number>25</Number></Property>
<Property><Key>address</Key> <Object> <Property><Key>streetAddress</Key>
<String>21 2nd Street</String></Property>
<Property><Key>city</Key> <String>New York</String></Property>
<Property><Key>state</Key> <String>NY</String></Property>
<Property><Key>postalCode</Key> <String>10021</String></Property>
</Object>
</Property> <Property><Key>phoneNumber</Key>
<Array> <Object> <Property><Key>type</Key> <String>home</String></Property>
<Property><Key>number</Key> <String>212 555-1234</String></Property></Object>
<Object>
<Property><Key>type</Key> <String>fax</String></Property> <Property><Key>number</
Key> <String>646 555-4567</String></Property> </Object> </Array>
</Property>
</Object>
```

JSON in PostgreSQL

Other Database Systems have similar capabilities.

- PostgreSQL manual:
  9.16. JSON Functions and Operators

- PostgreSQL/JSON Tutorial (easier to read)

Two PostgreSQL Data Types for Storing JSON

- JSON – store an exact copy of the JSON text
  - Slower, simpler
- JSONB – store the JSON data in binary format
  - Faster, more complex

CREATE and INSERT with JSON

```
CREATE TABLE products (
   productID PRIMARY KEY,
   name VARCHAR(255) NOT NULL,
   properties JSON );

INSERT INTO products VALUES
  ( 17, 'Ink Fusion T-Shirt’,
     '{"color": "white", "size": ["S","M","L","XL"]}')
```

SELECT with JSON

```
SELECT productID, name, properties
FROM products;
```

```
productID |      name                |      properties
--------- ----+----------------------+-------------------------------------------------------------
    17        | Ink Fusion T-Shirt     | {"size": ["S", "M", "L", "XL"], "color": "white"}
    26        | ThreadVerse T-Shirt  | {"size": ["S", "M", "L",], "color": "black"}
    35        | Dynamo T-Shirt         | {"size": ["L", "XL"], "color": "blue"}
```

SELECT on JSON Object Field

```
SELECT productID, name,
        properties ->> 'color’ AS theColor
FROM products
WHERE properties ->> 'color' IN ('black', 'white');
```

```
productID |       name                |       theColor
--------- ----+----------------------+----------------------
       17       | Ink Fusion T-Shirt      |   white
       26       | ThreadVerse T-Shirt  !   black
```

Other Data Serialization Approaches

Related (but not identical) capabilities, all Open Source.

- Protocol Buffers (Protobuf)
  "Protocol buffers are a language-neutral, platform-neutral extensible mechanism for serializing structured data."
- Apache Thrift
  "Framework, for scalable cross-language services development."
- Apache Avro
  "A data serialization system."