Cognitive Tooling

Programming Challenge

Code and Data

2025-05-01

This challenge is designed to simulate real-world tasks that an Enterprise IT software engineer will face.

Developers write code, but the managers that you work for expect outcomes in a business context. Almost always, that involves doing something useful to a system or with data that comes from a business process. Customer Service, Sales, Invoicing, Billing, Bug-Tracking, etc. The code written here is always in the context of handling data – transforming it, moving it, extracting insights from it.

This challenge is to extract and transform data. ETLs are ubiquitous patterns and there are many tools and approaches. The challenge is designed to reveal developer’s choices, their decision-making and provide them an opportunity to defend those choices.

Imagine a database. In fact, you won’t need to imagine it at all since I’ll provide one. We’ll use the [Chinook open-source example database](https://github.com/lerocha/chinook-database) for this exercise. It’s an ordering system for music tracks. The schema for this database (Data Model) is on the github Readme. If you want to install it locally, it runs just fine with sqlite3 or MySQL, and there are install instructions on [Database Guide](https://database.guide/2-sample-databases-sqlite/). This is for reference; you will not need it for this challenge.

Databases have tables that are related to each other in various ways. The focus here is to look at a many-to-one relationship that the Invoice and InvoiceLine tables have. Each Invoice has 1:M InvoiceLines. Each InvoiceLine knows the Invoice it belongs to. (Every child knows its parent) This structure is revealed when the InvoiceLine table has a foreign key to Invoice (FK InvoiceLine.InvoiceId).

It's common for analysists to write SQL queries like this:

**SELECT**

*i*.InvoiceId,

*i*.CustomerId,

*i*.InvoiceDate,

*il*.TrackId,

*il*.Quantity,

*il*.UnitPrice

**FROM**

Invoice *i* **INNER** **JOIN**

InvoiceLine *il* **ON** *il*.InvoiceId = *i*.InvoiceId

;

And have that exported as a CSV file. Then they hand that over to some other team ***that doesn’t have access to the database.*** You work on this second team. You must process the CSV file so that it fits into the next set of business tools you do have access to.

**Your Challenge**.

You have this CSV file as input, the output of SQL like above. You need to transform that CSV file into a JSON file, for the next stage of processing. (Many things like APIs accept only JSON as inputs.)

The schema of JSON you must deliver is an array of objects (Invoices) and each object will have an array of child objects (InvoiceLines) as a property. Like this:

[

{

"InvoiceId":"",

"CustomerId":"",

"InvoiceDate":"",

"InvoiceLines": [

{

"TrackId":"",

"Quantity":"",

"UnitPrice":""

},

{

"TrackId":"",

"Quantity":"",

"UnitPrice":""

}

]

}

]

You may write any program you like to accomplish this, so long as the input is transformed into a file shaped like the required output and validates as JSON.

Next:

How much revenue did Chinook sell in November 2021? Determine this ONLY from working with the JSON you produced above. How might you quickly extract such an insight?