# Data Integration Engineer

First and foremost, we are glad you are here! Thank you for taking the time to check out Ursa Health and apply for our role. Prior to starting the puzzle, we want to note that at this time, Ursa Health is unable to offer visa sponsorship for this position. We look forward to hearing from candidates who are authorized to work in the United States without sponsorship now or in the future.

*A few notes about this challenge: (**1**) You shouldn't have to spend more than a couple hours with it. There's no time limit, but you're busy and we don't want to waste your time. (**2**) As a modern engineer you might be tempted to copy-paste the entire thing into ChatGPT. Feel free to do so! Be forewarned though that ChatGPT won't do a good job without a fair amount of careful guidance. Moreover, a key part of the interview process will be to defend, reconsider, and elaborate on your answers in the puzzle, and it will be embarrassing for everyone if you've faked your way through it with AI without gaining a real understanding of the contours of the challenge. (**3**) There isn't one correct answer to these questions, so don't tie yourself in a knot looking for any one world-beating solution. There are many defensible answers. We're more interested in your thought process, and the way you can explain your answers during the in-person interview. (4) Some of the questions ask you to write SQL, but we're not trying to test your encyclopedic knowledge of SQL – the solutions shouldn't require advanced SQL beyond window function and an optional CTE or two. You can answer the questions using whatever dialect of SQL you're most comfortable with.*

©2025 by Ursa Health LLC. All rights reserved.

Before we get into the challenge itself, there are three bits of subject matter background to go over. For those of you who have worked in health care before, much of this will likely not be new, but it will still be worth reading to clarify the nomenclature used throughout the challenge.

First, this challenge involves medical claims data. Under our definition, medical claims are the financial records created by health plans that reflect the payment for (and various other salient details of) health care services or products rendered by an individual provider (like a doctor) or a health care facility (like a hospital). For the purposes of this challenge, consider medical claims to have a simple hierarchical structure: a "header", reflecting information like the identity of the patient, the total amount paid to the provider, and the identity of the provider to be paid; and one or more subordinate "service line items", reflecting information about the distinct services rendered to the patient, such as the billing code associated with each service and the date each service was rendered. In health care, claims data are often the only data available and so they are a major focus of our work. The "header" and the "service line items" are often, as in the challenge below, joined together into a single denormalized table.

Second, claims are often amended over time, and the administrative systems that health plans use to manage claims typically use "transactions" to process these changes. That is, when something about a claim needs to be changed – e.g., the paid amount associated with a service line item increased or decreased, or the entire claim denied, etc. – it isn't simply edited and the old version overwritten; rather one or more transactions are recorded in the system reflecting the desired change. Therefore, to reach an understanding of the claim's "final action" status – that is, the claim's current state, net of all updates made over the lifetime of the claim up to present day – one must apply all the changes embodied by these transactions, in sequence.

Third, health care data are messy, and bad stuff happens. A claim might be edited in a non-standard way in the original administrative system; the process by which the health plan generates the data feed it sends you might be buggy; a flat file might be errantly loaded twice; etc. When it comes to working with health care data, data workers should expect the unexpected, and not every data problem will have a perfect solution that works 100% of the time.

OK, here's the challenge:

**Part I (Question 1)**

You've just received an extract of medical claims data from a new health plan and are trying to make sense of it. Your understanding is that this dataset is transactional – that is, each record in the file represents a transaction, not the final action status of the record.

Several months of these medical claims data files have been loaded into a database table (raw_claim_trx) and you're looking through it. Knowing that the data are transactional, one of the first things you often look for is a column describing the type of transaction – e.g., "reversal" transactions (which undo a previous transaction) might be represented with an "R", and "adjustment" transactions (which update the claim with new information, typically following a reversal) might be represented with an "A". But no such column seems to exist here.

©2025 by Ursa Health LLC. All rights reserved.

However, you do see certain claim numbers – e.g., 290E2942842R1 – that stick out, because they're two characters longer than all the other claim numbers onscreen. The majority of the records have only eleven digits, and the only two-digit "suffixes" you see on the thirteen-digit claim numbers are R[x] and A[x], which might mean something like "the [x]th reversal" and "the [x]th adjustment".

So you search for records whose claim_num value starts with 290E2942842:

| member_num | servdt | proc_cd | claim_num | line_num | trxdt | paid_amt |
|---|---|---|---|---|---|---|
| 345983 | 20210215 | G0463 | 290E2942842 | 1 | 20210302 | 334.00 |
| 345983 | 20210215 | G0463 | 290E2942842R1 | 1 | 20210516 | -334.00 |

This makes sense! It's a little weird that the data source is appending a 2-character substring representing the type of transaction to the end of the claim identifier and calling that the "claim number", but it's clear enough what's happening on this claim: there was an original claim (with claim number 290E2942842), with one line item, for one procedure (with code G0463) provided on a single day (February 15, 2021) and originally paid in a couple of weeks (on or around the transaction date of March 2, 2021); the plan decided later that it wasn't going to pay for it, and reversed it (effective May 16, 2021). Maybe the claim was a clerical error, or maybe they discovered that the patient's primary insurance had already paid for it, we're not going to know. But we can just add up the paid amounts on the two transactions to yield the "final action" net paid amount, which is $0.

You look for another one of these two-extra-digit claim numbers, and find one with claim_num value of 2239G230928A1, and so you search for records whose claim_num starts with 2239G230928 and get the following:

| member_num | servdt | proc_cd | claim_num | line_num | trxdt | paid_amt |
|---|---|---|---|---|---|---|
| 993478 | 20210425 | 36415 | 2239G230928 | 1 | 20210523 | 47.25 |
| 993478 | 20210425 | 85025 | 2239G230928 | 2 | 20210523 | 22.92 |
| 993478 | 20210425 | 84443 | 2239G230928 | 3 | 20210523 | 112.31 |
| 993478 | 20210425 | 80053 | 2239G230928 | 4 | 20210523 | 84.22 |
| 993478 | 20210425 | 80053 | 2239G230928 | 5 | 20210523 | 84.22 |
| 993478 | 20210425 | 36415 | 2239G230928R1 | 1 | 20210712 | -47.25 |
| 993478 | 20210425 | 36415 | 2239G230928R1 | 1 | 20210712 | -47.25 |
| 993478 | 20210425 | 36415 | 2239G230928R1 | 1 | 20210712 | -47.25 |
| 993478 | 20210425 | 85025 | 2239G230928R1 | 2 | 20210712 | -22.92 |
| 993478 | 20210425 | 85025 | 2239G230928R1 | 2 | 20210712 | -22.92 |
| 993478 | 20210425 | 85025 | 2239G230928R1 | 2 | 20210712 | -22.92 |
| 993478 | 20210425 | 84443 | 2239G230928R1 | 3 | 20210712 | -112.31 |
| 993478 | 20210425 | 84443 | 2239G230928R1 | 3 | 20210712 | -112.31 |
| 993478 | 20210425 | 84443 | 2239G230928R1 | 3 | 20210712 | -112.31 |
| 993478 | 20210425 | 80053 | 2239G230928R1 | 4 | 20210712 | -84.22 |
| 993478 | 20210425 | 80053 | 2239G230928R1 | 4 | 20210712 | -84.22 |

©2025 by Ursa Health LLC. All rights reserved.

| | | | | | | |
|---|---|---|---|---|---|---|
| 993478 | 20210425 | 80053 | 2239G230928R1 | 4 | 20210712 | -84.22 |
| 993478 | 20210425 | 80053 | 2239G230928R1 | 5 | 20210712 | -84.22 |
| 993478 | 20210425 | 80053 | 2239G230928R1 | 5 | 20210712 | -84.22 |
| 993478 | 20210425 | 80053 | 2239G230928R1 | 5 | 20210712 | -84.22 |
| 993478 | 20210425 | 36415 | 2239G230928A1 | 1 | 20210712 | 47.25 |
| 993478 | 20210425 | 36415 | 2239G230928A1 | 1 | 20210712 | 47.25 |
| 993478 | 20210425 | 36415 | 2239G230928A1 | 1 | 20210712 | 47.25 |
| 993478 | 20210425 | 85025 | 2239G230928A1 | 2 | 20210712 | 22.92 |
| 993478 | 20210425 | 85025 | 2239G230928A1 | 2 | 20210712 | 22.92 |
| 993478 | 20210425 | 85025 | 2239G230928A1 | 2 | 20210712 | 22.92 |
| 993478 | 20210425 | 84443 | 2239G230928A1 | 3 | 20210712 | 112.31 |
| 993478 | 20210425 | 84443 | 2239G230928A1 | 3 | 20210712 | 112.31 |
| 993478 | 20210425 | 84443 | 2239G230928A1 | 3 | 20210712 | 112.31 |
| 993478 | 20210425 | 80053 | 2239G230928A1 | 4 | 20210712 | 84.22 |
| 993478 | 20210425 | 80053 | 2239G230928A1 | 4 | 20210712 | 84.22 |
| 993478 | 20210425 | 80053 | 2239G230928A1 | 4 | 20210712 | 84.22 |

Generally, this looks like a case where the original claim was amended via a reversal (to back out the original payments) and then was adjusted to apply the updated details. But it also looks like some of the records for this claim have duplicates, which will need to be dealt with.

*Question 1: What do you think the "final action" paid amount for this claim is?*

**Part II (Question 2)**

Next, let's write a query that will produce the correct final action results for the full dataset.

First, we need to do something to about the duplicative data, which we've confirmed appears throughout the dataset (so we'll probably have to assume it may happen in future files as well). One general approach to deal with duplicative transactional claims data is to identify (or construct, if necessary) a Transaction ID, and then keep only one record per Transaction ID.

After removing the duplicates, we want to further collapse the (now-de-duplicated) transactional data down to a "final action" grain size of one record per claim service line item. To do this we will want to identify (or construct) a Claim Service Line Item ID, and then use that to aggregate the transactional data associated with each line item.

Lastly, we will want to identify (or construct) a Claim Header ID, which we might later use to generate aggregate (header-level) financial figures from the service line items on the claim.

*Question 2: Based on what you've seen so far, write a SELECT query against the raw_claim_trx table that returns one record per claim service line item, and includes columns for (1) Transaction ID; (2) Claim*

©2025 by Ursa Health LLC. All rights reserved.

*Service Line Item ID; (3) Claim Header ID; and (4) Service Line Item Paid Amount (with the final action paid amount for that service line item).*

**Part III (Questions 3 - 5)**

Bad news! You just found a bunch of records that seem to be missing line numbers. For example:

| member_num | servdt | proc_cd | claim_num | line_num | trxdt | paid_amt |
|---|---|---|---|---|---|---|
| 223725 | 20210502 | 99212 | 4835Y583761 | | 20210523 | 55.23 |
| 223725 | 20210502 | 87426 | 4835Y583761 | | 20210523 | 82.21 |
| 223725 | 20220112 | 99213 | 7352E795832 | | 20220131 | 76.65 |
| 223725 | 20220112 | 90658 | 7352E795832 | | 20220131 | 40.00 |
| 223725 | 20220112 | 69210 | 7352E795832 | | 20220131 | 24.49 |
| 223725 | 20220112 | 99213 | 7352E795832R1 | | 20220325 | -76.65 |
| 223725 | 20220112 | 90658 | 7352E795832R1 | | 20220325 | -40.00 |
| 223725 | 20220112 | 69210 | 7352E795832R1 | | 20220325 | -24.49 |
| 223725 | 20220112 | 99212 | 7352E795832A1 | | 20220325 | 58.88 |
| 223725 | 20220112 | 90658 | 7352E795832A1 | | 20220325 | 40.00 |
| 223725 | 20220112 | 69210 | 7352E795832A1 | | 20220325 | 24.49 |

(For records like this, consider the line_num column to contain NULL.)

This sort of heterogeneity within a single extract isn't all that uncommon. For example, the single file you receive may actually be populated by two different administrative systems with different original structure and behavior but shoehorned into a single extract format.

*Question 3: You still need to generate the three identifiers described earlier (i.e., Transaction ID, Claim Service Line Item ID, and Claim Header ID). How would you generate those values for rows with missing line numbers?*

*Question 4: Generally speaking, to what extent, if any, is the integrity of your resulting dataset harmed by the absence of line number values? (For example, are there any scenarios where you could imagine the absence of line numbers would lead to your logic generating inaccurate final action paid amount values?)*

*Question 5: A solution to deal with the missing line numbers might be more straightforward if we could prove that a single claim was always uniformly either missing all line numbers or not missing any line numbers at all. Write a SQL query that checks this.*

**Part IV (Question 6 - 7)**

©2025 by Ursa Health LLC. All rights reserved.

Let's say that you are able to confirm with this query that claims can always be sorted cleanly into the two categories, i.e., claims either have all transactions with populated line_num values or have all transactions with a line_num value of NULL.

Furthermore, after looking closely at the data and running some additional diagnostic queries you determine that the duplicate record problem is limited to claims with populated line numbers, i.e., that transactions with a missing line number are never duplicated, and you feel confident that they will not be duplicated in future received files. (OK, this isn't very realistic, because you could never be sure that a future file wouldn't have duplicate transactions with missing line numbers, but it's just a simplifying assumption we're making for the purpose of the next question.)

*Question 6: Update your earlier Question 2 query to produce the desired results (as outlined in Question 2) given your new understanding of the source data (i.e., that there are claims with missing line numbers).*

*Question 7: Will you now or in the future require sponsorship for employment visa status (e.g., H-1B visa)? YES -or- NO*

*If yes, do you currently have a H-1B visa? Please explain.*

©2025 by Ursa Health LLC. All rights reserved.