**Documentation for Job Jack Assessment:**

**Basic Steps broken down (High level overview):**

In the below table I have broken down the full project into a set of high-level steps that’s I need to take in order to complete the assessment.

|  |  |
| --- | --- |
| Step 1 | Setup Visual studio project   * Create a new directory for the project * Use ‘npm init’ or ‘yarn init’ to initialize a new node.js project |
| Step 2 | Connect to database provided |
| Step 3 | Install any dependencies (like PostgreSQL Driver and any other libraries needed) |
| Step 4 | Create the API for the ETL.   * Setup an Express.js server to handle API requests * Create a route that will trigger the ETL progress |
| Step 5 | Extract and transform data.   * Use the PostgreSQL driver to connect to the db. * Fetch data from the jack\_location table * Aggregate the location density information (eg. count the number of locations in a certain area) |
| Step 6 | Load data into jack\_location\_density table.   * Use the postgresql driver to insert the aggregated data into the jack\_location\_density table |
| Step 7 | Predict sign-ups   * Use the data from the jack\_location table or the jack\_location\_density table) along with sing up date to build a predictive model. * The prediction should estimate the number of sign-ups by January 2025 |
| Step 8 | Write clean and scalable code.   * Organize your code into functions and modules for clarity * Use meaningful variable and function names * Comment code to explain complex logic * Optimize code for performance, considering both small and large datasets. |
| Step 9 | Create a github repository to place submission in. |

**The above steps drilled down:**

|  |  |  |
| --- | --- | --- |
| **Step Overview** | **Description** | **Images/Text references** |
| Create a Folder | I created a new folder to act as a project directory in VS Code. |  |
| Initialize a node.js project | I ran the “npm init” command in my terminal In vs code and I was given an error. Only then realizing then that I probably don’t have have the node package manager installed. I installed Node.JS and then installed extra tools for npm. |  |
| Setup npm project and installations | I then needed to install express js and setup my npm project through powershell as I was having access issues it seems. |  |
| Create well defined folders to keep my codebase structured and maintainable | I then started creating folders and files for things I may need just to get the structured layout created to make it maintainable.   * Root Folder is where I will leave things like app.js/server.js for configuration and main server files etc * Routes folder is where I will place each route handler file to correspond with a specific api route. Making it easier to manage different routes. |  |
| Install the postgres driver | I now needed to install the postgres sql driver before getting to writing my etl logic and connecting |  |
| Document all the code shit |  |  |
|  |  |  |
|  |  |  |
| Create aggregation rules  Installed pgAdmin4 | I wanted to create an example of my aggregation to see what I would need in my BI tool to be able to do my forecasting.  This will be the aggregated data I will need to create my predictions and make some visualizations based on area etc.  Query Used:  select  count(sign\_up\_date) as signupscount,  cast(sign\_up\_date as date) as shorterdate,  full\_location,  suburb,  city,  province  from public.jack\_location  group by full\_location,  cast(sign\_up\_date as date),  suburb,  city,  province |  |
|  | I then attempted to read all my fields into the destination table only to find that there wasn’t a table setup so I needed to create that. For the sake of trying to get my process to read the data in I just manually created the table but usually I would attempt to put a drop and create table into the ETL so that it repopulates the table each time it runs or I would incrementally populate in the ETL itself. Because I am not familiar with Node I left this step out in the interest of time. |  |
| Aggregate and load data | I had to create a few extra lines of code to make sure that my count was catered for and that there was some kind of granularity in my data by including the location along with the suburb etc. The idea is that we never want to just pull the “answer” into a BI tool, we want a set of data that we can drill down in to but at the same time not have such a big load that it affects performance and do our calculations in SQL and not in the BI tool. |  |
| Run node | I got my ETL process to run successfully after the above code. |  |
| Check Output | I then had a look at the output in my table and for some reason I didn’t have any data that was read correctly into my destination table that I created. |  |
| Create a Template | I usually create a aesthetic design template that is easy to use and carry through a BI report for quick design purposes. I have made a lot of specialized colour driven design elements to match Job Jacks colour theme. Things like hovering over buttons makes them green (including the hamburger button on the top left for the pop ups.) Green hover overs wont be seen in the screenshots but please test them yourself out of interest! Please also note if you click on the bar graph provinces, it will drill down on the map on the right. |  |
| Created Forecasting Graph in Power BI after import | For the purpose of saving time I exported the data so I could put it into my tool of choice and start working on the forecasting values | The top graph creates a low end and top end forecast for assumed possible lowest and highest values.    You can drill down to see the values on this graph with the three dots on the top right as seen below. |

**Traditional breakdown of the process:**

Please find below a traditional methodology of how this would be done and how the steps would be broken down using alternative traditional data methods. Keep in mind that for the below comparison I have chosen to use SSIS as I have the most experience with that so the write up would be quicker to do, however we could have also explored using cloud services with Azure and using tools like Azures blob containers along with Azure Data Factory to do the transformations and alternatively we could have used AWS with S3 buckets as storage and AWS Glu and a few other tools to do transformations. Please note that for the below I would have just used a relational database in Azure along with SSIS as an output reporting DB. I would also make sure that AT ALL TIMES I employ the Kimball methodology as I believe that is the most effective way of making sure your data estate is ready in each step of the way with dimensional modelling.

|  |  |
| --- | --- |
| Step 1 | Create a new SSIS project using SQL Server Data Tools (SSDT) |
| Step 2 | Add connection managers.   * Add a Postgres database connection manager to connect to the source database * Add an Azure SQL database connection manager for the destination |
| Step 3 | Create a data flow Task. Within the Control Flow tab, add a Data Flow task, which will contain all ETL logic. |
| Step 4 | Extract data from Postgres inside a data flow task using the OLE DB source component to extract data. |
| Step 5 | Transform the data using various SSIS transformations like data conversions and conditional splits etc to clean, aggregate and manipulate as needed.  I would consider a dimensional model at this stage (Kimball methodology) to make sure that I am using dimensions and facts to build my data solution as accurately as possible. The steps would go, create raw data tables with SSIS as the first set of tables, create dimensions and facts to house transactional data along with “data about data.” I would then create a “service layer” where I would use things like materialized views or aggregated tables that would purely be built for analysis and reporting purposes. This will ensure that data is at its most granular level and also at its most summarized level so that speed and big/small data is all compensated for. |
| Step 6 | Load the data into Azure SQL Database by creating a OLE DB Destination component. Azure does allow for free trial accounts so this could be done for the first year I believe before paying unless very large storage is required. |
| Step 7 | Deploy the SSIS package to a SQL Server instance that is created in the Azure environment |
| Step 8 | Use SQL Server Agent to schedule the execution of the SSIS package at specific intervals/times |
| Step 9 | Connect to the Azure SQL DB from Power BI |
| Step 10 | Create visuals and your Data story in Power BI. In our case visualize the requirements by showing the predicted number of sign ups by January 2025 and the trend for each month leading up etc. |