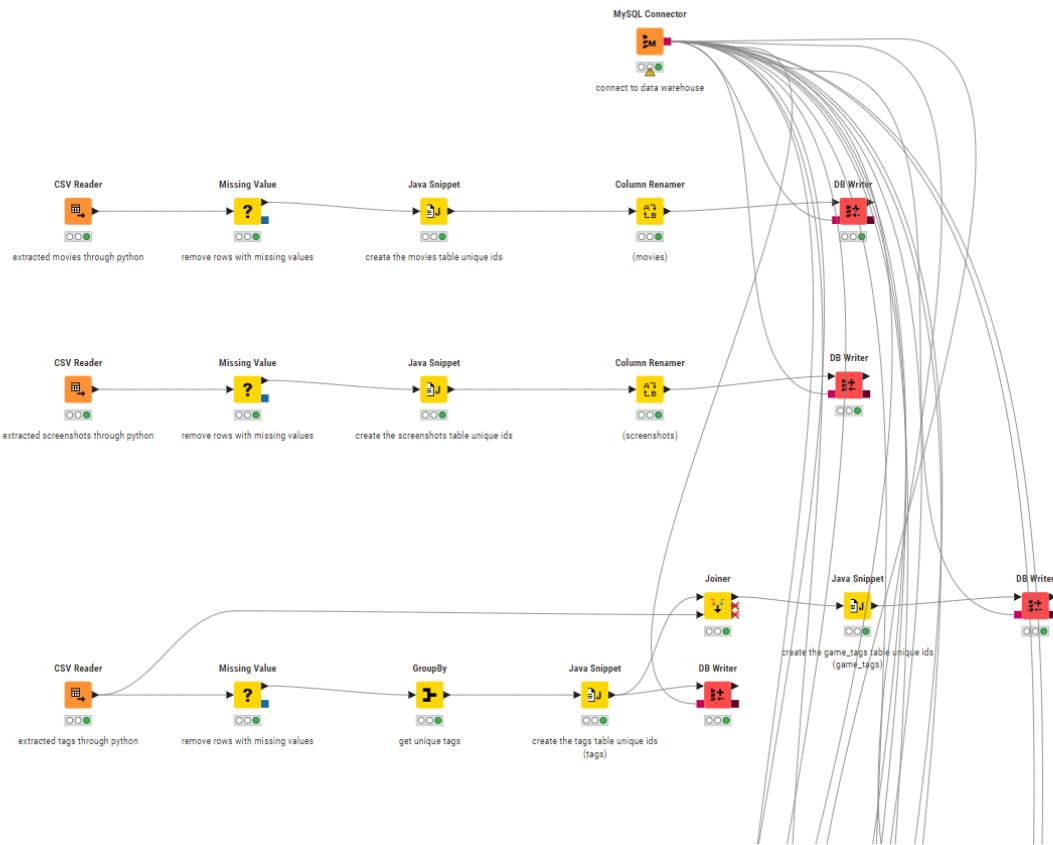
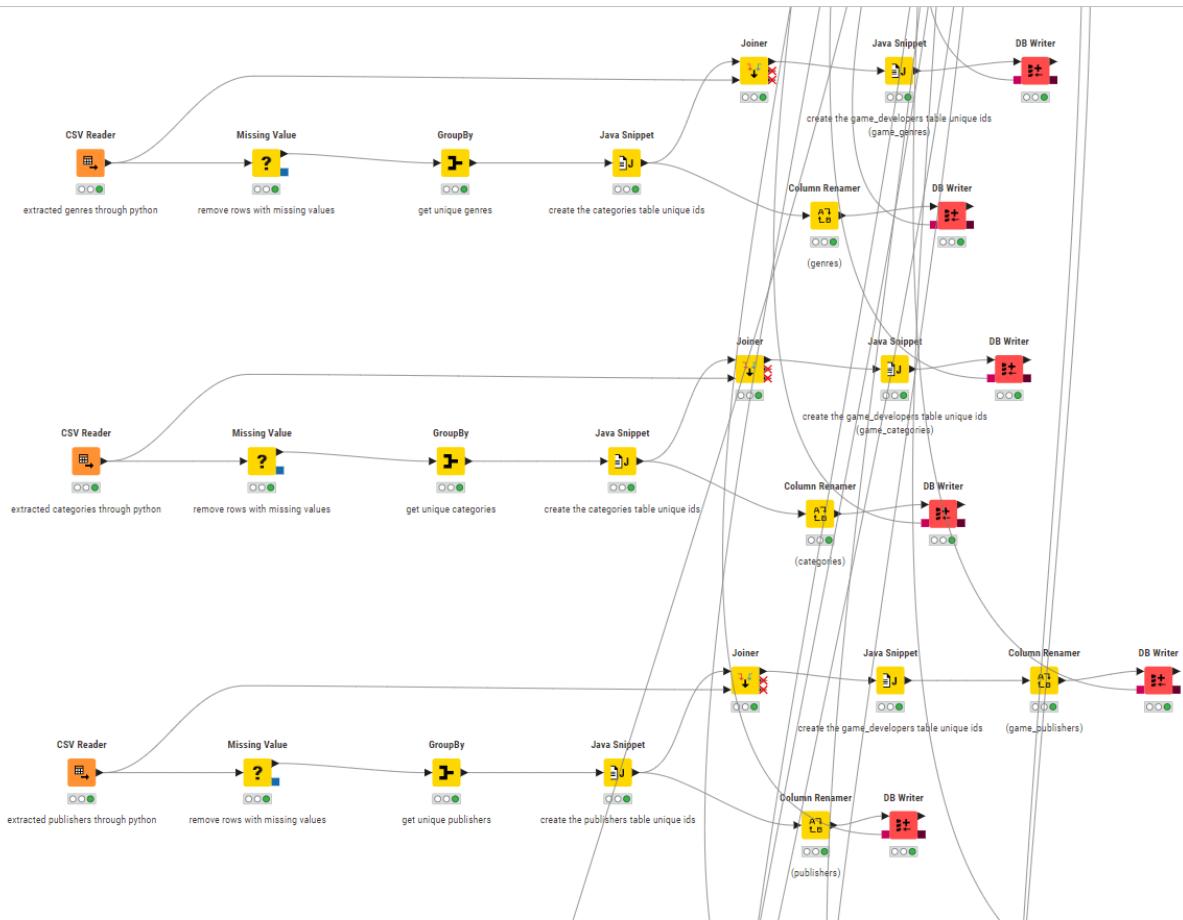
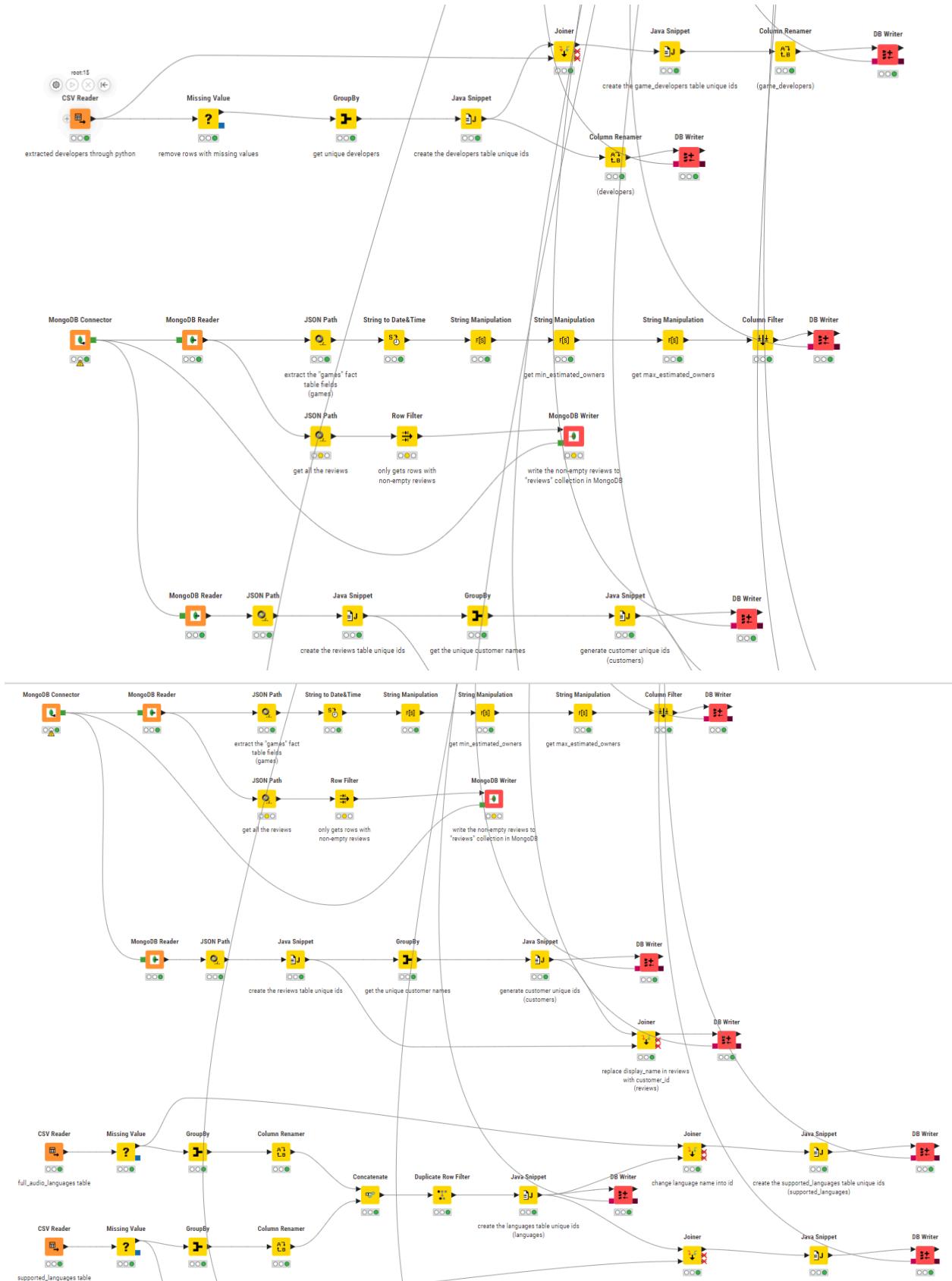
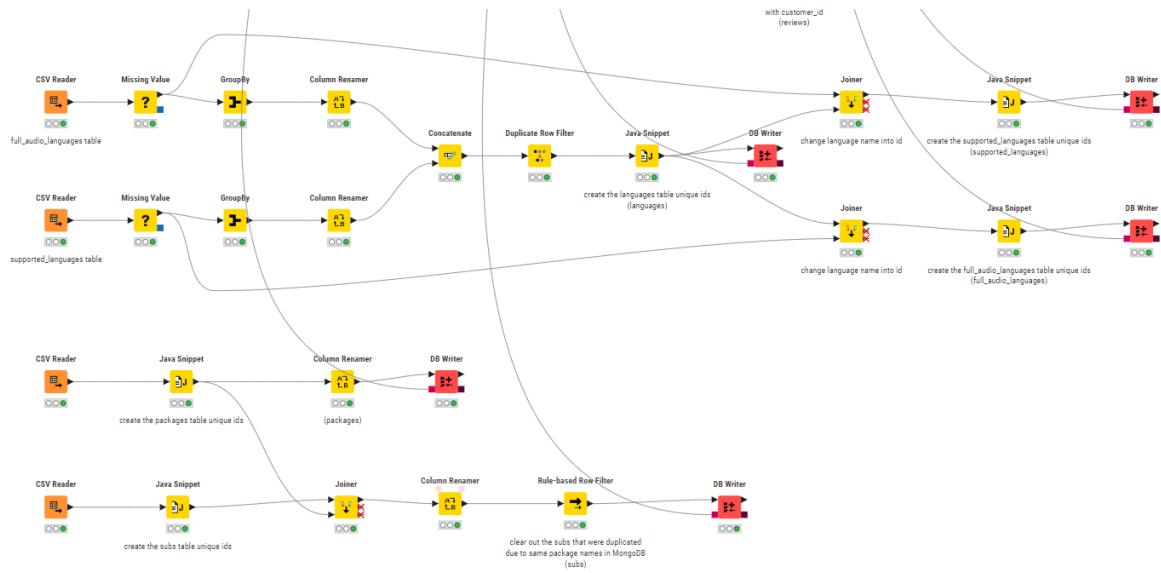


KNIME Workflow:









Python Supplementary Code:

```
▷ v      import pandas as pd
      import json
      import re
      import html
      import numpy as np
[23]      + Code + Markdown Python
      ↴ Restructure the JSON file to allow for importing into MongoDB

      with open('games.json', encoding='utf-8') as f:
          data = json.load(f)

          records = []
          for game_id, details in data.items():
              details['game_id'] = game_id
              records.append(details)

          df = pd.DataFrame(records)

          df.to_json('restructured_data.json', orient='records', lines=True)

          print("Restructured data saved to 'restructured_data.json'")

[3] ... Restructured data saved to 'restructured_data.json'
```

Extract Each Individual Review, their Rating if available, and Customer Display Name

```
▷ v      with open('stadvdb_mco1.reviews.json', 'r', encoding='utf-8') as f:
      data = json.load(f)

      review_pattern = r'^(.*?)"\s*([\d\.\./]+)\s*"\s*([\w\&\-.\/"]*)'

      extracted_reviews = []

      for game in data:
          reviews = game.get('reviews', '')

          if reviews:
              matches = re.findall(review_pattern, reviews)

              for match in matches:
                  review_text = match[0]
                  rating = match[1] if match[1] else ''
                  display_name = match[2].strip() if match[2] else ''

                  extracted_reviews.append({
                      "game_id": game['game_id'],
                      "review_text": review_text,
                      "rating": rating,
                      "display_name": display_name
                  })

      df_reviews = pd.DataFrame(extracted_reviews)

      df_reviews.to_csv('extracted_reviews.csv', index=False)

      print(df_reviews.head())

      game_id           review_text rating \
0 1026428 New WW2 Strategy Game Offers A Harrowing Look ...
1 1026428 (...) in execution Warsaw manages to deliver its...
2 1026428 (...) Beautiful hand-painted artwork and turn-ba...
```

Extract Each Supported Language along with their Corresponding Game ID

```
extracted_data = []

with open('restructured_data.json', 'r', encoding='utf-8') as f:
    for line in f:
        try:
            data = json.loads(line.strip())
            game_id = data.get("game_id", None)
            supported_languages = data.get("supported_languages", [])

            extracted_data.append({
                "game_id": game_id,
                "supported_languages": supported_languages
            })
        except json.JSONDecodeError as e:
            print(f"Error decoding JSON: {e}")

df = pd.DataFrame(extracted_data)

expanded_df = df.explode('supported_languages')

print(expanded_df)

expanded_df.to_csv('supported_languages_table.csv', index=False)
```

game_id	supported_languages
0	20200 English
1	655370 English
1	655370 French
1	655370 Italian
1	655370 German
...	...
97406	2593970 Russian
97406	2593970 Portuguese - Brazil
97407	3137150 English
97409	3137150 English

Python

Extract Each Full Audio Language along with their Corresponding Game ID

```
extracted_data = []

with open('restructured_data.json', 'r', encoding='utf-8') as f:
    for line in f:
        try:
            data = json.loads(line.strip())
            game_id = data.get("game_id", None)
            full_audio_languages = data.get("full_audio_languages", [])

            extracted_data.append({
                "game_id": game_id,
                "full_audio_languages": full_audio_languages
            })
        except json.JSONDecodeError as e:
            print(f"Error decoding JSON: {e}")

df = pd.DataFrame(extracted_data)

expanded_df = df.explode('full_audio_languages')

print(expanded_df)

expanded_df.to_csv('full_audio_languages_table.csv', index=False)
```

game_id	full_audio_languages
0	20200 NaN
1	655370 NaN
2	1732930 NaN
3	1355720 NaN

Python

Extract each Developer Array Item and their corresponding Game ID

```
extracted_data = []

with open('restructured_data.json', 'r', encoding='utf-8') as f:
    for line in f:
        try:
            data = json.loads(line.strip())
            game_id = data.get("game_id", None)
            developers = data.get("developers", [])

            extracted_data.append({
                "game_id": game_id,
                "developers": developers
            })
        except json.JSONDecodeError as e:
            print(f"Error decoding JSON: {e}")

df = pd.DataFrame(extracted_data)

expanded_df = df.explode('developers')

print(expanded_df)

expanded_df.to_csv('developers_table.csv', index=False)
```

Python

	game_id	developers
0	20200	Perpetual FX Creative
1	655370	Rusty Moyher
2	1732930	Campião Games
3	1355720	Odd Critter Games
4	1139950	Unusual Games

Extract each Publisher Array Item and their corresponding Game ID

```
extracted_data = []

with open('restructured_data.json', 'r', encoding='utf-8') as f:
    for line in f:
        try:
            data = json.loads(line.strip())
            game_id = data.get("game_id", None)
            publishers = data.get("publishers", [])

            extracted_data.append({
                "game_id": game_id,
                "publishers": publishers
            })
        except json.JSONDecodeError as e:
            print(f"Error decoding JSON: {e}")

df = pd.DataFrame(extracted_data)

expanded_df = df.explode('publishers')

print(expanded_df)

expanded_df.to_csv('publishers_table.csv', index=False)
```

Python

	game_id	publishers
0	20200	Perpetual FX Creative
1	655370	Wild Rooster
2	1732930	Campião Games
3	1355720	Odd Critter Games
4	1139950	Unusual Games

Extract each Categories Array Item and their corresponding Game ID

```
extracted_data = []

with open('restructured_data.json', 'r', encoding='utf-8') as f:
    for line in f:
        try:
            data = json.loads(line.strip())
            game_id = data.get("game_id", None)
            categories = data.get("categories", [])

            extracted_data.append({
                "game_id": game_id,
                "categories": categories
            })
        except json.JSONDecodeError as e:
            print(f'Error decoding JSON: {e}')

df = pd.DataFrame(extracted_data)

expanded_df = df.explode('categories')

print(expanded_df)

expanded_df.to_csv('categories_table.csv', index=False)
```

	game_id	categories
0	20200	Single-player
0	20200	Multi-player
0	20200	Steam Achievements
0	20200	Partial Controller Support
1	655370	Single-player

Extract each Genres Array Item and their corresponding Game ID

```
extracted_data = []

with open('restructured_data.json', 'r', encoding='utf-8') as f:
    for line in f:
        try:
            data = json.loads(line.strip())
            game_id = data.get("game_id", None)
            genres = data.get("genres", [])

            extracted_data.append({
                "game_id": game_id,
                "genres": genres
            })
        except json.JSONDecodeError as e:
            print(f'Error decoding JSON: {e}')

df = pd.DataFrame(extracted_data)

expanded_df = df.explode('genres')

print(expanded_df)

expanded_df.to_csv('genres_table.csv', index=False)
```

	game_id	genres
0	20200	Casual
0	20200	Indie
0	20200	Sports
1	655370	Action
1	655370	Indie

Extract each Tags Array Item and their corresponding Game ID

```
extracted_data = []

with open('restructured_data.json', 'r', encoding='utf-8') as f:
    for line in f:
        try:
            data = json.loads(line.strip())
            game_id = data.get("game_id", None)
            tags = data.get("tags", {})

            if isinstance(tags, dict):
                for tag_name, tag_value in tags.items():
                    extracted_data.append({
                        "game_id": game_id,
                        "tag_name": tag_name,
                        "tag_value": tag_value
                    })
            elif isinstance(tags, list):
                for tag_name in tags:
                    extracted_data.append({
                        "game_id": game_id,
                        "tag_name": tag_name,
                        "tag_value": None
                    })
        except json.JSONDecodeError as e:
            print(f'Error decoding JSON: {e}')

df = pd.DataFrame(extracted_data)

print(df)

df.to_csv('tags_table.csv', index=False)
```

Python

Extract each Screenshots Array Item and their corresponding Game ID

```
extracted_data = []

with open('restructured_data.json', 'r', encoding='utf-8') as f:
    for line in f:
        try:
            data = json.loads(line.strip())
            game_id = data.get("game_id", None)
            screenshots = data.get("screenshots", [])

            extracted_data.append({
                "game_id": game_id,
                "screenshots": screenshots
            })
        except json.JSONDecodeError as e:
            print(f'Error decoding JSON: {e}')

df = pd.DataFrame(extracted_data)

expanded_df = df.explode('screenshots')

print(expanded_df)

expanded_df.to_csv('screenshots_table.csv', index=False)
```

Python

game_id	screenshots
0	20200 https://cdn.akamai.steamstatic.com/steam/apps/...
...	...
97409 3054200 https://shared.akamai.steamstatic.com/store_it...	
97409 3054200 https://shared.akamai.steamstatic.com/store_it...	

Extract each Packages & Subs Array Item and their corresponding Game ID

```
packages_extracted_data = []
subs_extracted_data = []

with open('restructured_data.json', 'r', encoding='utf-8') as f:
    for line in f:
        try:
            data = json.loads(line.strip())

            game_id = data.get("game_id", None)
            packages = data.get("packages", [])

            for package in packages:
                package_title = package.get("title", None)
                package_description = package.get("description", None)

                packages_extracted_data.append({
                    "game_id": game_id,
                    "package_title": package_title,
                    "package_description": package_description
                })

            subs = package.get("subs", [])

            for sub in subs:
                sub_text = sub.get("text", None)
                sub_description = sub.get("description", None)
                sub_price = sub.get("price", None)

                subs_extracted_data.append({
                    "game_id": game_id,
                    "package_title": package_title,
                    "sub_text": sub_text,
                    "sub_description": sub_description,
                    "sub_price": sub_price
                })
        except json.JSONDecodeError as e:
            print(f"Error decoding JSON: {e}")

packages_df = pd.DataFrame(packages_extracted_data)
subs_df = pd.DataFrame(subs_extracted_data)

print("Packages Table:")
print(packages_df)
print("\nSubs Table:")
print(subs_df)

packages_df.to_csv('packages_table.csv', index=False)
subs_df.to_csv('subs_table.csv', index=False)
```

Python

Packages Table:			
	game_id	package_title	package_description
0	20200	Buy Galactic Bowling	
1	655370	Buy Train Bandit	
2	1732930	Buy Jolt Project	
3	1355720	Buy Henosis™	
4	1659180	Buy TD Worlds	