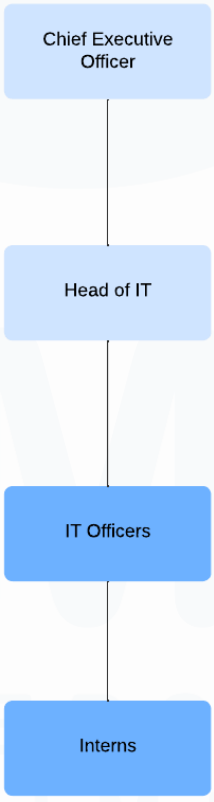
BAB III
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Mahasiswa magang ditempatkan diposisi *Data Enginer* yang berada dinaungan Departemen IT yang dipimpin oleh Head of IT yang terdiri dari beberapa divisi dari masing-masing bagian perusahaan. Team Leader dari setiap bagian divisi memberikan perkenalan atau *training* kepada tim, terutama mahasiswa yang sedang melaksanakan magang di perusaan Thinkspedia Digital Solusindo tentang apa yang sedang dikerjakan oleh perusahaan dan apa yang sedang direncanakan untuk kedepannya. Berikut merupakan bagan dari alur koordinasi.



Gambar 3. 1 Alur Koordinasi Perusahaan Thinkspedia

Sistem koordinasi pada perusahaan biasanya dimulai dari Head of IT yang mendapatkan konfirmasi berkaitan tentang permintaan dari klien yang diterima oleh seorang *Project Manager* dari perusahaan, yang nantinya akan didiskusikan dengan masing-masing *team leader* dari setiap divisi yang kemudian akan didiskusikan kembali dengan skala yang lebih fokus sesuai dengan divisi masing-masing dan berakhir dengan diskusi antar team termasuk mahasiswa magang untuk membahas permintaan klien.

Gambar 3. 2 Alur Kerja Magang Perusahaan Thinkspedia

3.2 Tugas dan Uraian Kerja Magang

Minggu Ke-	Kegiatan	Periode
1	Pengenalan Perusahaan, Jobdesk, dan Plan serta Project yang sedang berjalan.	18 Maret – 25 Maret 2024
2	Melakukan Eksplorasi pada Data Perusahaan serta Mempelajari Peran dan Fungsi dari Data Perusahaan	25 Maret – 1 April 2024
3	Melakukan Data Managing berdasarkan instruksi yang diberikan perusahaan yang sesuai dengan jobdesc yang telah diberikan melalui Studio 3T(MongoDB).	1 April – 8 April 2024
4	Membuat Aggregation & Indexing pada Data perusahaan sesuai dengan atribut pada data yang dimiliki perusahaan.	8 April – 15 April 2024
5	Melakukan Scrapping Data dengan tujuan sebagai referensi perbaikan dan memanipulasi serta memodifikasi data dari perusahaan sesuai dengan ketentuan perusahaan melalui Phyton dan Studio3T (MongoDB).	15 April – 22 April 2024
6	Mempelajari Geographical Information Systems melalui training yang disediakan oleh perusahaan.	22 April – 6 Mei 2024
7	Menggunakan QGIS untuk menciptakan beberapa data Geographical.	6 Mei – 20 Mei 2024

Tabel 3. 1 Tabel Tugas dan Uraian Kerja Magang

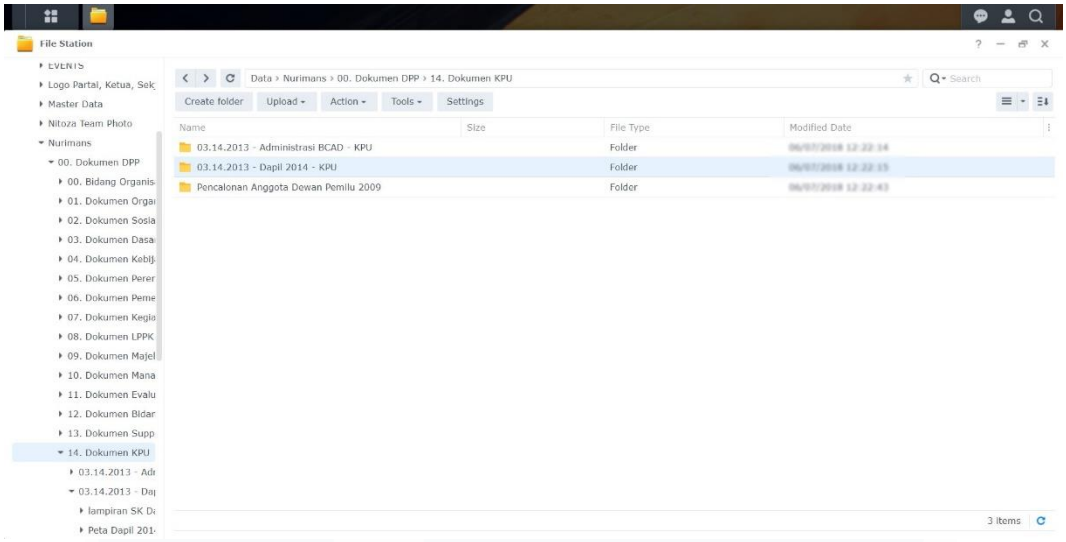
3.2.1 Pengenalan Perusahaan, Jobdesk, dan Plan serta Project yang sedang berjalan

Dalam minggu pertama ini, mahasiswa akan diperkenalkan dengan lingkungan perusahaan, yang meliputi jobdesk yang diberikan sesuai dengan surat magang yang sebelumnya diterima oleh mahasiswa. Selain itu, mahasiswa juga akan diperkenalkan dengan rencana perusahaan ke depannya, seperti beberapa rencana atau project yang ingin dilakukan oleh perusahaan. Di minggu pertama ini, mahasiswa juga akan dikenalkan dengan beberapa tools yang digunakan oleh perusahaan. Selain itu, perusahaan juga terbuka jika mahasiswa ingin menggunakan tools lain sebagai opsi tambahan jika tools yang diperkenalkan tidak dapat memproses hal yang ingin mahasiswa lakukan dalam pekerjaannya.

Mahasiswa juga akan diperkenalkan dengan beberapa project yang sedang berjalan, lalu berdiskusi untuk menentukan pekerjaan atau hal yang bisa dilakukan ke depannya terhadap project yang sedang berjalan tersebut. Prosesnya meliputi pengenalan lingkungan perusahaan kepada mahasiswa, penggunaan tools perusahaan dan opsi tambahan, serta diskusi mengenai project yang sedang berjalan. Proses ini membantu mahasiswa untuk memahami lingkungan perusahaan, mengerti dan siap menjalankan jobdesk yang diberikan, mengenal dan dapat menggunakan tools perusahaan serta tools tambahan jika diperlukan, serta memiliki rencana kerja yang jelas untuk berkontribusi terhadap project yang sedang berjalan.

3.2.2 Melakukan Eksplorasi pada Data Perusahaan serta Mempelajari Peran dan Fungsi dari Data Perusahaan

Dalam tahap ini, mahasiswa melakukan eksplorasi pada data perusahaan, data-data terkait adalah data politik tentang pemilu yang dimiliki sejak tahun 2014 oleh perusahaan, dalam hal ini mahasiswa mencoba untuk melakukan eksplorasi pada data tersebut.



Gambar 3. 1 *Cloud Online* Perusahaan Thinkspedia

Gambar 3.3 diatas merupakan salah satu cloud online yang dimiliki oleh perusahaan, yang dimana cloud tersebut menyimpan beberapa data politik yang tentunya tidak bisa dibagikan ke publik. Cloud tersebut memiliki data penting yang tersedia sejak tahun 2014 dan beberapa data tersebut dapat dieksplorasi oleh mahasiswa sebagai pengenalan dan juga pembelajaran dalam kegiatan magang ini.

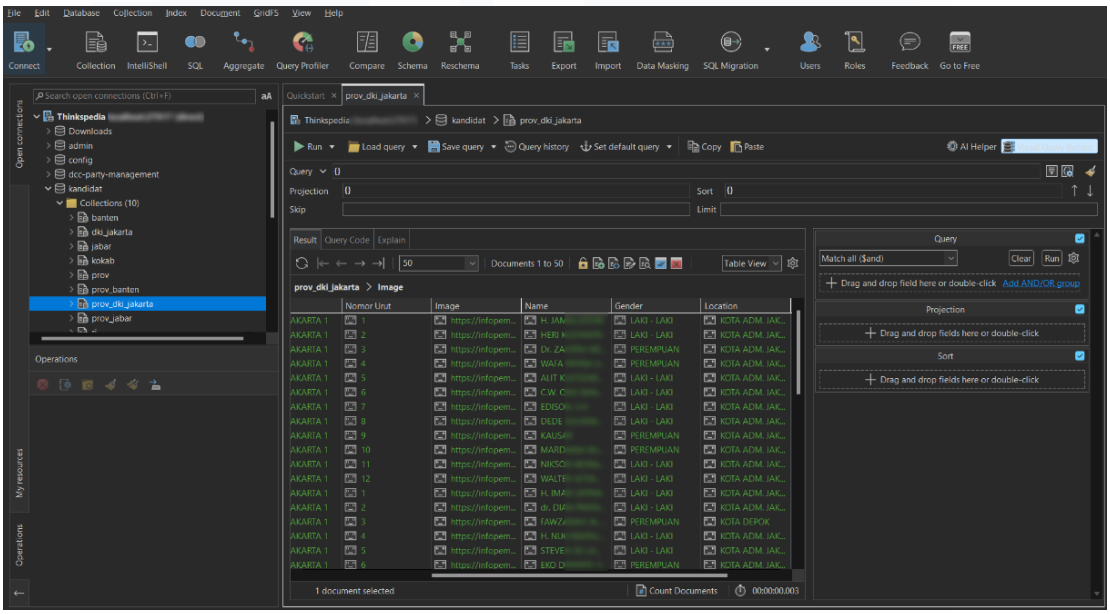
REKAPITULASI JUMLAH KURSI
ANGGOTA DEWAN PERWAKILAN RAKYAT DAERAH PROVINSI
DALAM PEMILIHAN UMUM TAHUN 2014

No	PROVINSI	JUMLAH DAPIL	JUMLAH KAB/KOTA	PEMILIHAN UMUM ANGGOTA DPRD PROVINSI TAHUN 2014												JUMLAH KURSI
				DP I	DP II	DP III	DP IV	DP V	DP VI	DP VII	DP VIII	DP IX	DP X	DP XI	DP XII	
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	ACEH															
2	SUMATERA UTARA															
3	SUMATERA BARAT															
4	RIAU															
5	JAMBI															
6	SUMATERA SELATAN															
7	BENGKULU															
8	LAMPUNG															
9	BANGKA BELITUNG															
10	KEPULAUAN RIAU															
11	DKI JAKARTA															
12	JAWA BARAT															
13	JAWA TENGAH															
14	DI YOGYAKARTA															
15	JAWA TIMUR															
16	BANTEN															
17	BALI															
18	NUSA TENGGARA BARAT															
19	NUSA TENGGARA TIMUR															
20	KALIMANTAN BARAT															

Komisi Pemilihan Umum | Hal 1

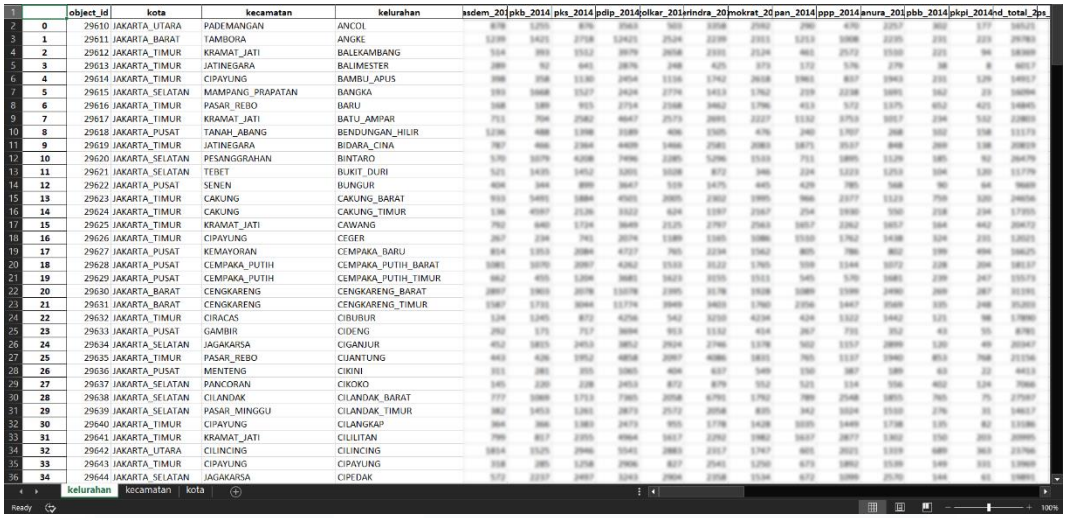
Gambar 3. 2 Contoh Data Perusahaan

Gambar 3.4 merupakan isi salah satu dari data yang dimiliki perusahaan, yaitu data rekapitalisasi untuk jumlah kursi anggota DPRD dalam pemilu tahun 2014, mahasiswa melakukan eksplorasi dengan beberapa data yang dimiliki perusahaan untuk menentukan langkah apa yang tepat sebagai *Data Engineer* pada data-data tersebut.



Gambar 3. 3 Proses Eksplorasi Data

Gambar 3.5 merupakan bagaimana mahasiswa melakukan eksplorasi pada data yang dimiliki perusahaan melalui database server milik perusahaan, sebagai contoh, diatas merupakan kandidat dari dapil provinsi DKI Jakarta yang berdasarkan lokasi tempat para kandidat dapil.



	object_id	kota	kecamatan	kelurahan	asdem_20	pkb_2014	pks_2014	pdip_2014	golkar_201	erindra_20	mokrat_20	pan_2014	ppp_2014	anura_201	pbb_2014	pkpi_2014	nd_total_2p
2	0	29610	JAKARTA_UTARA	PADEMANGAN	878	1255	876	3563	503	3358	2592	290	470	2257	402	177	16521
3	1	29611	JAKARTA_BARAT	TAMBORA	1239	1421	2758	12421	2524	2239	2311	1213	1008	2235	231	223	29783
4	2	29612	JAKARTA_TIMUR	KRAMAT_JATI	514	393	1512	3979	2658	2331	2124	461	2572	1510	221	94	18369
5	3	29613	JAKARTA_TIMUR	JATINEGARA	289	92	641	2876	248	425	373	172	576	279	58	8	6057
6	4	29614	JAKARTA_TIMUR	CIPAYUNG	998	958	1130	2454	1116	1742	2618	1961	837	1943	231	129	14917
7	5	29615	JAKARTA_SELATAN	MAMPANG_PRAPATAN	193	1668	1527	2424	2774	1413	1762	259	2238	1691	162	23	16094
8	6	29616	JAKARTA_TIMUR	PASAR_REBO	168	189	915	2714	2168	3462	1796	413	572	1375	652	421	14845
9	7	29617	JAKARTA_TIMUR	KRAMAT_JATI	711	704	2582	4647	2573	2691	2227	1132	3753	1017	234	532	22803
10	8	29618	JAKARTA_PUSAT	TANAH_ABANG	1236	488	1398	3189	406	1505	476	240	1707	268	102	158	11173
11	9	29619	JAKARTA_TIMUR	JATINEGARA	787	466	2364	4409	1466	2581	2083	1871	3537	848	269	138	20819
12	10	29620	JAKARTA_SELATAN	PESANGGRAHAN	570	1079	4208	7496	2285	5296	1533	711	1895	1129	185	92	26479
13	11	29621	JAKARTA_SELATAN	TEBET	521	1435	1452	3201	1028	872	346	224	1223	1253	104	120	11779
14	12	29622	JAKARTA_PUSAT	SENEN	404	344	899	3647	519	1475	445	429	785	568	90	64	9669
15	13	29623	JAKARTA_TIMUR	CAKUNG	933	5491	1884	4501	2005	2302	1995	966	2377	1123	759	320	24656
16	14	29624	JAKARTA_TIMUR	CAKUNG_TIMUR	136	4597	2126	3122	624	1197	2167	254	1930	550	218	234	17355
17	15	29625	JAKARTA_TIMUR	KRAMAT_JATI	752	640	1724	3649	2125	2797	2563	1657	2262	1657	164	442	20472
18	16	29626	JAKARTA_TIMUR	CIPAYUNG	267	234	741	2074	1189	1165	1086	1510	1762	1438	124	231	12021
19	17	29627	JAKARTA_PUSAT	KEMAYORAN	814	1353	2084	4727	765	2234	1562	805	786	802	199	494	16625
20	18	29628	JAKARTA_PUSAT	CEMPAKA_PUTIH	1081	1070	2097	4262	1533	3122	1765	559	1144	1072	228	204	18137
21	19	29629	JAKARTA_PUSAT	CEMPAKA_PUTIH	662	455	1204	3681	1623	3155	1511	545	570	1681	239	247	15571
22	20	29630	JAKARTA_BARAT	CENGKARENG	2897	1903	2078	11078	2395	3178	1928	1089	1599	2490	269	287	31191
23	21	29631	JAKARTA_BARAT	CENGKARENG	1587	1731	3044	11774	3949	3403	1760	2356	1447	3569	335	248	35203
24	22	29632	JAKARTA_TIMUR	CIRACAS	124	1245	872	4256	542	3233	4234	424	1322	1442	121	98	17890
25	23	29633	JAKARTA_PUSAT	GAMBIR	292	171	717	3694	953	1132	414	267	731	352	63	55	8781
26	24	29634	JAKARTA_SELATAN	JAGAKARSA	452	1815	2453	3852	2924	2746	1378	502	1317	2899	120	49	20347
27	25	29635	JAKARTA_TIMUR	PASAR_REBO	443	426	1952	4858	2097	4086	1821	765	1137	1940	853	768	21156
28	26	29636	JAKARTA_PUSAT	MENTENG	311	281	355	1065	404	637	549	150	387	189	63	22	4413
29	27	29637	JAKARTA_SELATAN	PANCORAN	145	220	228	2453	872	879	552	521	114	556	402	124	7066
30	28	29638	JAKARTA_SELATAN	CILANDAK	777	1069	1713	7365	2058	6791	1792	789	2548	1855	765	75	27597
31	29	29639	JAKARTA_SELATAN	PASAR_MINGGU	382	1453	1261	2873	2572	2058	835	342	1024	1510	276	31	14617
32	30	29640	JAKARTA_TIMUR	CIPAYUNG	364	366	1383	2473	955	1778	1428	1035	1449	1738	135	82	13186
33	31	29641	JAKARTA_TIMUR	KRAMAT_JATI	799	817	2355	4964	1617	2292	1982	1637	2877	1302	150	203	20995
34	32	29642	JAKARTA_UTARA	CILINCING	1814	1525	2946	5541	2883	2317	1747	601	2021	1319	689	363	23766
35	33	29643	JAKARTA_TIMUR	CIPAYUNG	318	285	1258	2906	827	2541	1250	673	1892	1539	149	331	13969
36	34	29644	JAKARTA_SELATAN	JAGAKARSA	572	2237	2497	3243	2904	2358	1534	672	1099	2570	144	61	19851

Gambar 3. 4 Contoh File Milik Perusahaan

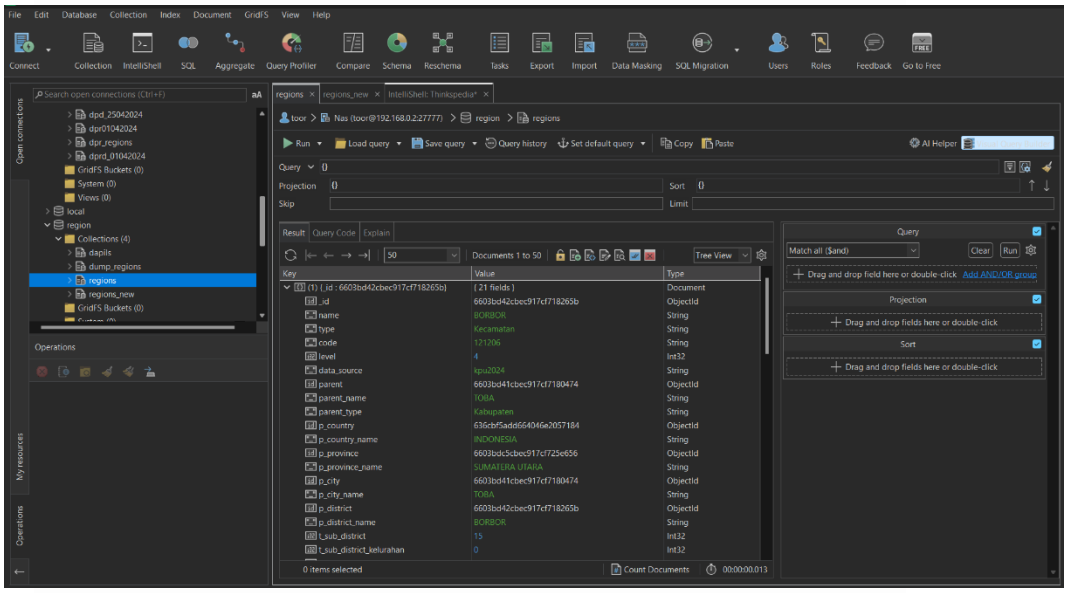
File Excel di atas merupakan salah satu file milik perusahaan yang membahas data perhitungan suara yang diperoleh oleh berbagai partai pada pemilu tahun 2014 lalu. Dalam file ini, terdapat informasi yang sangat detail mengenai jumlah suara yang diterima di berbagai wilayah, mulai dari tingkat kelurahan, kecamatan, hingga kota. Data ini mencakup berbagai indikator penting seperti jumlah pemilih, jumlah suara sah, dan distribusi suara ke setiap partai.

Mahasiswa dapat melakukan eksplorasi data untuk mengenal lebih dekat tentang data-data tersebut yang nantinya akan diolah mahasiswa sebagai Data Engineer. Eksplorasi ini akan melibatkan berbagai teknik analisis data, seperti pembersihan data, visualisasi data, dan penggunaan algoritma statistik untuk mengidentifikasi tren dan pola dalam data pemilu. Melalui proses ini, mahasiswa tidak hanya akan memperoleh pemahaman yang lebih baik tentang cara kerja pemilu dan distribusi suara, tetapi juga akan mengembangkan keterampilan teknis yang diperlukan untuk berkarir dalam bidang data engineering.

Selain itu, data ini dapat digunakan untuk melakukan berbagai analisis lanjutan, seperti mengidentifikasi faktor-faktor yang mempengaruhi hasil pemilu di berbagai wilayah, membandingkan kinerja partai politik dari satu daerah ke daerah lain, dan memprediksi hasil pemilu di masa depan berdasarkan tren historis. Dengan demikian, file ini tidak hanya merupakan sumber informasi penting bagi perusahaan, tetapi juga merupakan alat pembelajaran yang berharga bagi mahasiswa yang ingin memahami dan menguasai analisis data dalam konteks yang nyata.

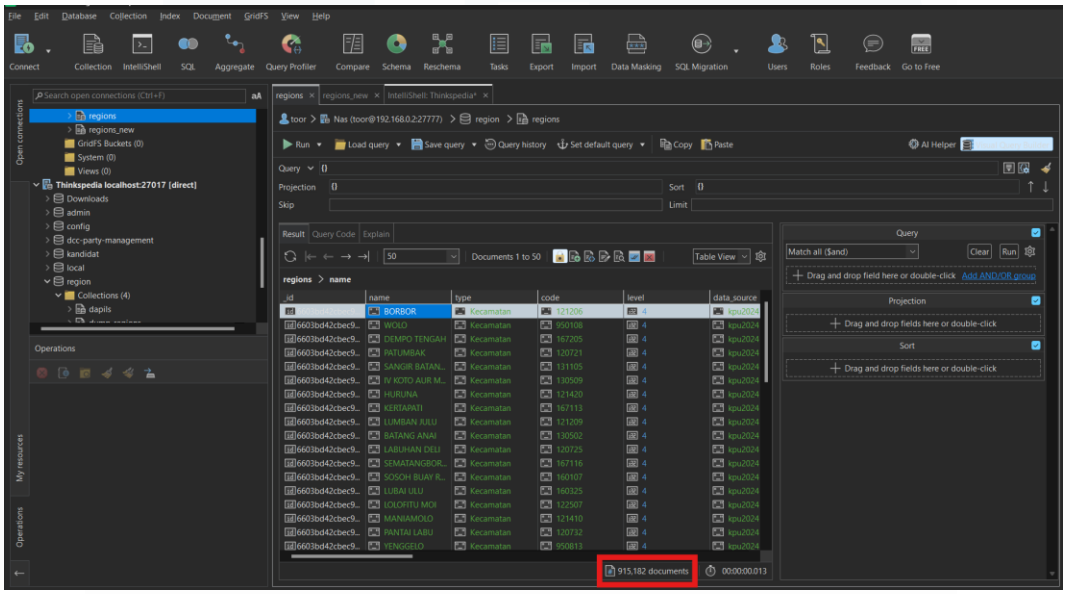
3.2.3 Melakukan Data Managing berdasarkan instruksi yang diberikan perusahaan yang sesuai dengan jobdesc yang telah diberikan melalui Studio 3T(MongoDB).

Dalam tahap ini, mahasiswa melakukan *Data Migration* atau pemindahan data dari satu database ke database lain dengan tujuan sebagai Konsolidasi Data, dengan tujuan untuk menciptakan lingkungan dimana sebuah data terpusat dan dapat dengan mudah diakses, sehingga meningkatkan efisiensi pengelolaan dan analisis data.



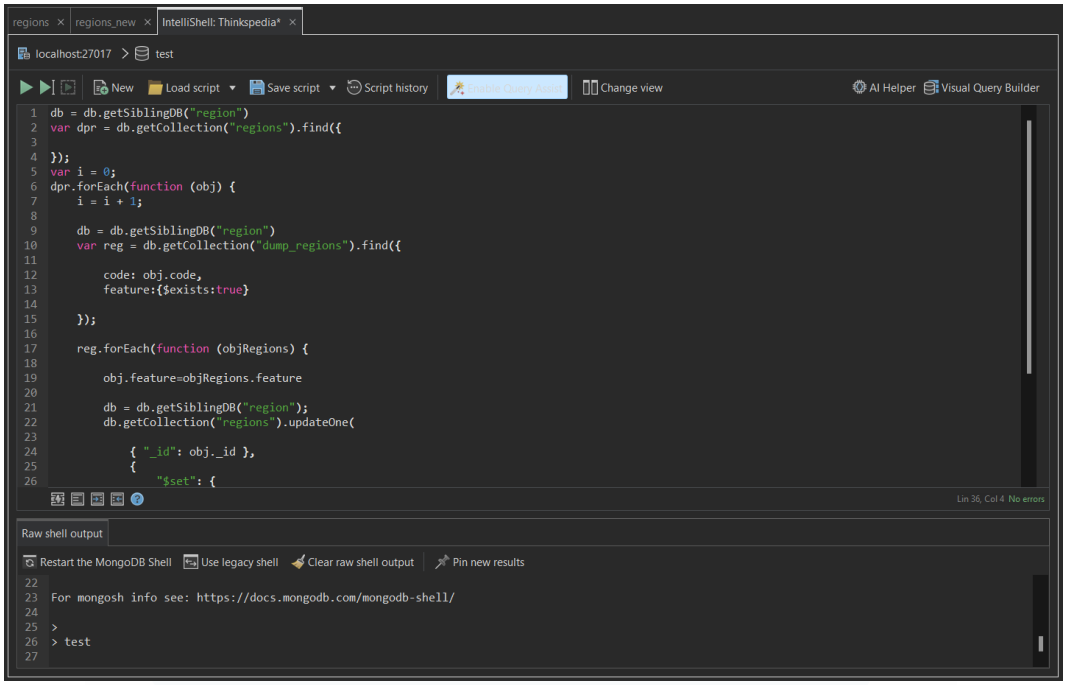
Gambar 3. 5 *Data Collection 'regions'*

Gambar 3.7 merupakan salah satu contoh collection ‘regions’ yang dimana berisikan data-data berupa field dengan berbagai fungsinya, dalam tools Studio3T tersebut terutama ‘regions’ ada beberapa field key yang memiliki peran penting sebagai data tipe parent yang memiliki peran penting dalam database.



Gambar 3. 6 *Tableview* Koleksi *‘regions’*

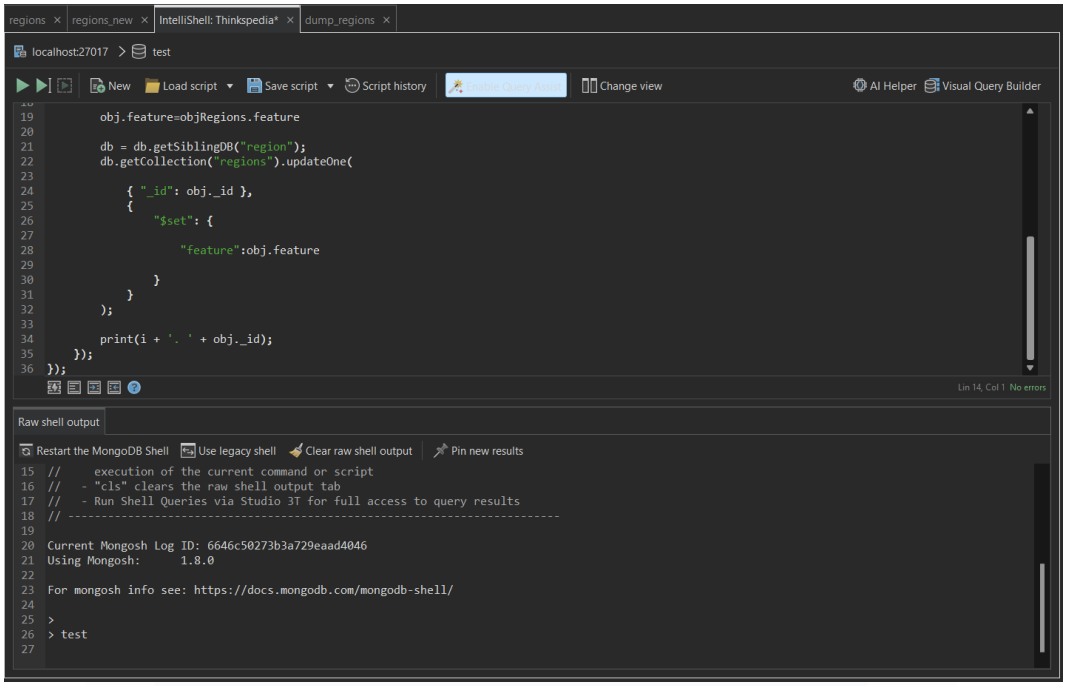
Berikut adalah *tableview* dari tampilan database collection *‘regions’*, dapat dilihat bahwa data tersebut memiliki total 915,182 documents, yang dimana total dari data tersebut merupakan bagian dari data regions yang memiliki atribut dan aspek yang berbeda-beda, serta objectID yang berbeda juga.



Gambar 3. 7 Skrip *Update* pada koleksi '*regions'*

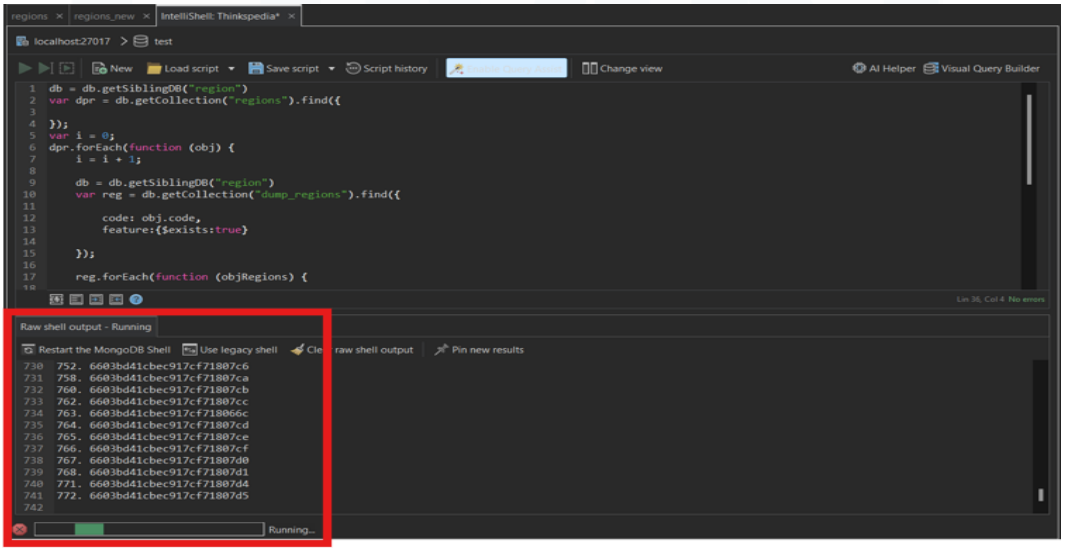
Gambar 3.9 merupakan skrip MongoDB yang digunakan untuk memperbarui data dalam koleksi '*regions'* berdasarkan data dari koleksi '*dump_regions'*. Skrip ini pertama-tama melakukan inisiasi koneksi ke database '*regions'* dan mengambil semua dokumen dari koleksi '*regions'*, yang disimpan dalam variabel *'dpr'*. Variabel *'i'* diinisialisasi ke nol untuk melacak *looping* atau pengulangan . Dalam setiap *looping* melalui dokumen *'dpr'* menggunakan forEach, nilai *'i'* ditingkatkan atau ditambahkan untuk menghitung jumlah dokumen yang diproses.

Selanjutnya, skrip melakukan inisiasi kembali koneksi ke database region dan mencari dokumen yang cocok di koleksi '*dump_regions'* berdasarkan code dari dokumen '*regions'* saat ini, serta memastikan bahwa field *feature* ada. Jika dokumen yang cocok ditemukan dalam '*dump_regions'*, field *feature* dari dokumen tersebut disalin ke dokumen saat ini dalam '*regions'*.



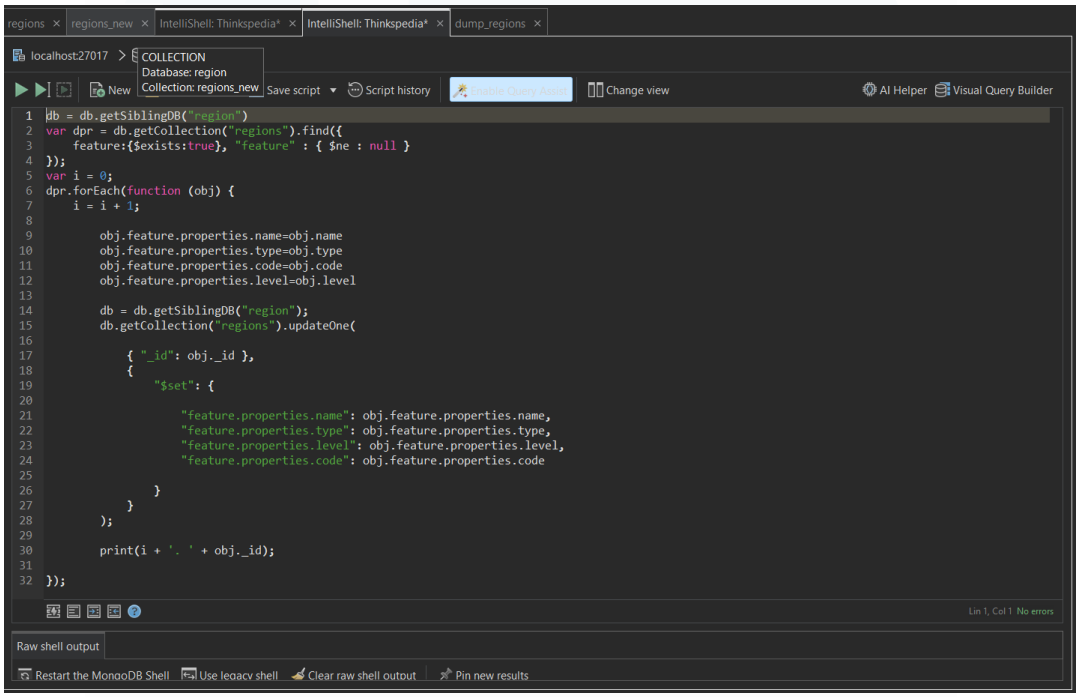
Gambar 3. 8 Lanjutan Skrip pada koleksi *'regions'*

Setelah itu, skrip memperbarui dokumen dalam koleksi '*regions'* dengan field *feature* yang baru menggunakan metode *updateOne*. Pembaruan dilakukan berdasarkan *_id* dokumen untuk memastikan dokumen yang tepat diupdate. Terakhir, skrip mencetak nomor *looping* dan *_id* dari dokumen yang diperbarui ke output untuk memberikan umpan balik tentang proses yang berjalan.



Gambar 3. 9 *Output* dari Skrip *update*

Dengan demikian, skrip ini secara efisien memperbarui field *feature* dalam koleksi '*regions'* berdasarkan data dari koleksi '*dump_regions'*, memastikan bahwa setiap dokumen dalam '*regions'* yang memiliki code yang sama dengan dokumen di '*dump_regions'* akan memiliki field feature yang diupdate dengan benar.



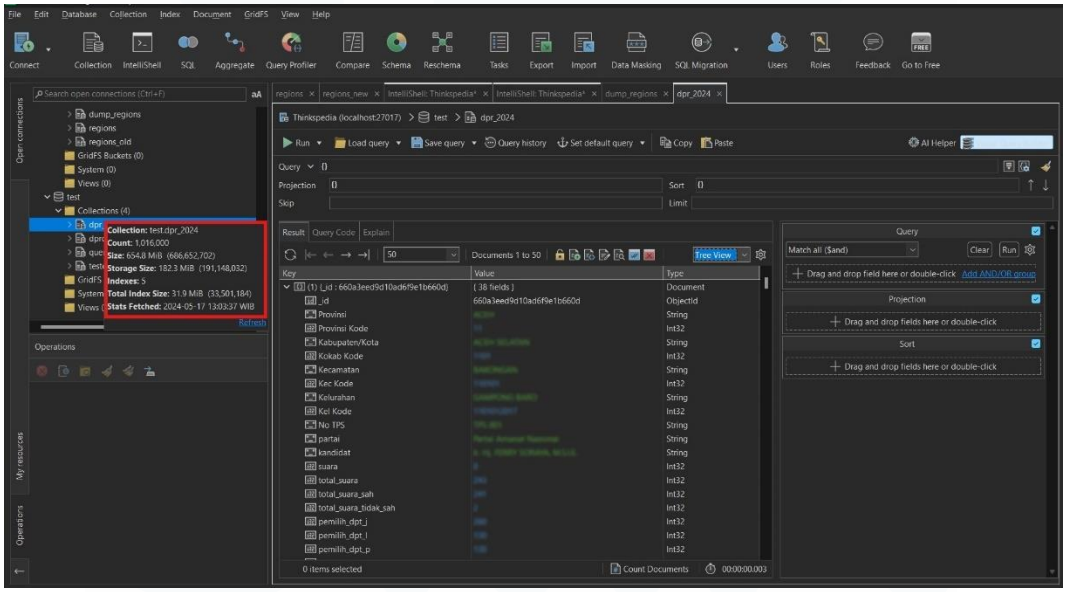
Gambar 3. 10 *Output* dari Skrip *update*

Selanjutnya, kode di atas adalah skrip MongoDB yang digunakan untuk memperbarui field '*feature.properties'* dalam dokumen pada koleksi '*regions*'. Pertama, melakukan inisiasi dengan menghubungkan ke database region, dan semua dokumen yang memiliki field '*feature*' yang ada dan tidak null diambil dari koleksi '*regions*', hasilnya disimpan dalam variabel *'dpr'*. Variabel *'i'* diinisialisasi ke nol untuk menghitung jumlah dokumen yang diproses.

Kemudian, skrip melakukan *looping* setiap dokumen dalam *'dpr'* menggunakan fungsi *'forEach'*, di mana setiap kali sebuah dokumen diproses, nilai *'i'* ditingkatkan. Dalam setiap *looping*, field '*properties*' dalam '*feature*' diperbarui dengan nilai dari field '*name'*, '*type'*, '*code'*,

dan ‘*level’* dari dokumen yang sedang diproses. Setelah field ‘*properties’* diperbarui, dokumen dalam koleksi ‘*regions*' diupdate dengan nilai-nilai baru menggunakan metode ‘*updateOne*’, yang memperbarui dokumen yang sesuai berdasarkan ‘*_id’* dokumen tersebut.

Terakhir, skrip mencetak nomor dari *looping* dan *‘_id’* dari dokumen yang diperbarui ke output untuk memberikan konfirmasi bahwa dokumen tersebut telah diproses. Dengan demikian, skrip ini memastikan bahwa field *‘feature.properties’* dalam setiap dokumen pada koleksi *‘regions’* diperbarui dengan informasi yang relevan dari field ‘*name’*, ‘*type’*, ‘*code’*, dan ‘*level’* dari dokumen yang sama, menjaga konsistensi dan keutuhan data dalam koleksi ‘*regions*'.



Gambar 3. 11 Koleksi Data *dpr_2024*

Selanjutnya merupakan data dengan judul dpr_2024 yang berisikan tentang data terkait suara dari pemilih calon anggota DPR dari beberapa daerah. Data ini mencakup berbagai informasi penting seperti Provinsi, Kabupaten, Kecamatan, serta Kelurahan dari tiap pemilih yang terdata.

Setiap entri dalam dataset ini memuat rincian yang lengkap mengenai wilayah administratif tempat suara tersebut dihitung.

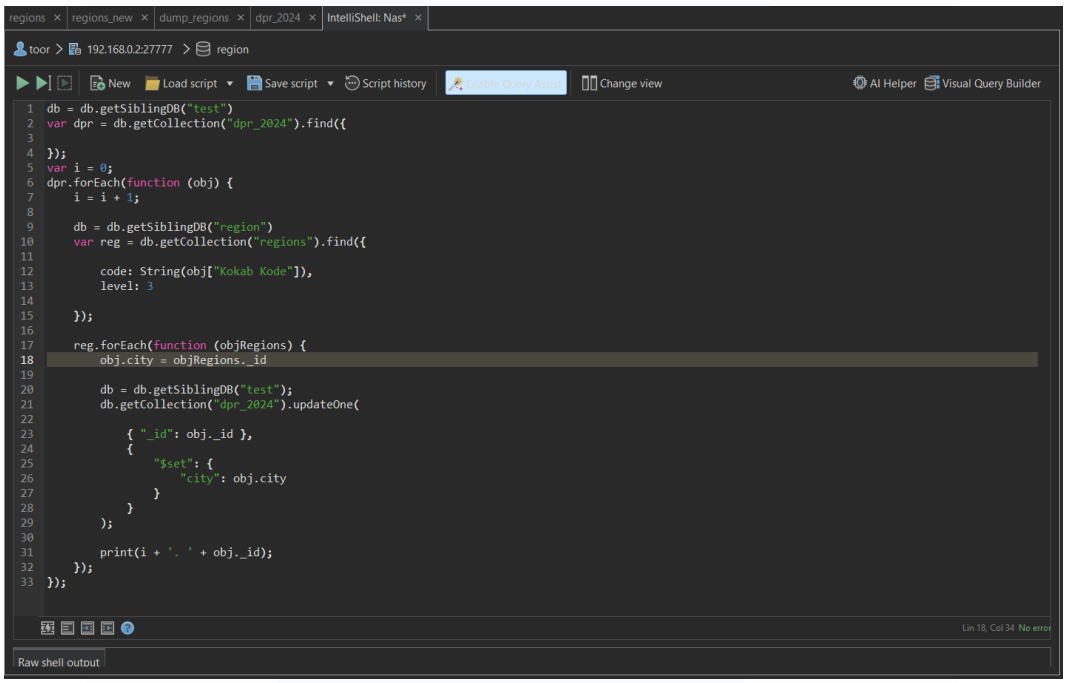
Selain informasi wilayah, data ini juga mencakup berbagai metrik lainnya yang relevan dengan pemilihan umum, seperti jumlah suara yang sah dan tidak sah, serta distribusi suara ke masing-masing calon anggota DPR (Caleg) yang berpartisipasi dalam pemilu tahun 2024. Dengan struktur data yang komprehensif, dpr_2024 memungkinkan analisis mendalam terhadap pola pemungutan suara di berbagai daerah, membantu dalam memahami preferensi pemilih di tingkat lokal hingga nasional.

Data ini juga dilengkapi dengan identifier unik untuk setiap dokumen, yang memungkinkan pelacakan dan pengelolaan data yang efisien. Informasi seperti kode provinsi, kabupaten/kota, dan kecamatan membantu dalam memetakan distribusi suara dengan presisi tinggi. Adanya detail mengenai kelurahan mempermudah analisis mikro terhadap data pemilih, sehingga setiap perubahan atau pola yang muncul di tingkat paling bawah dapat terdeteksi dengan mudah.

Selain itu, data dpr_2024 juga menyimpan informasi tentang waktu pengambilan data, yang memberikan konteks temporal terhadap perubahan yang mungkin terjadi selama periode pemilihan. Dengan menggunakan data ini, analisis longitudinal dapat dilakukan untuk mengidentifikasi tren dan perubahan dalam preferensi pemilih dari satu pemilu ke pemilu berikutnya.

Secara keseluruhan, dataset dpr_2024 adalah sumber informasi yang sangat berharga untuk berbagai analisis statistik dan politik, memungkinkan para peneliti, analis data, dan pembuat kebijakan untuk mendapatkan wawasan mendalam tentang dinamika pemilihan umum di Indonesia pada tahun 2024. Dengan memanfaatkan dataset ini, berbagai

strategi dapat dirumuskan untuk meningkatkan partisipasi pemilih dan memperbaiki sistem pemilihan secara keseluruhan.

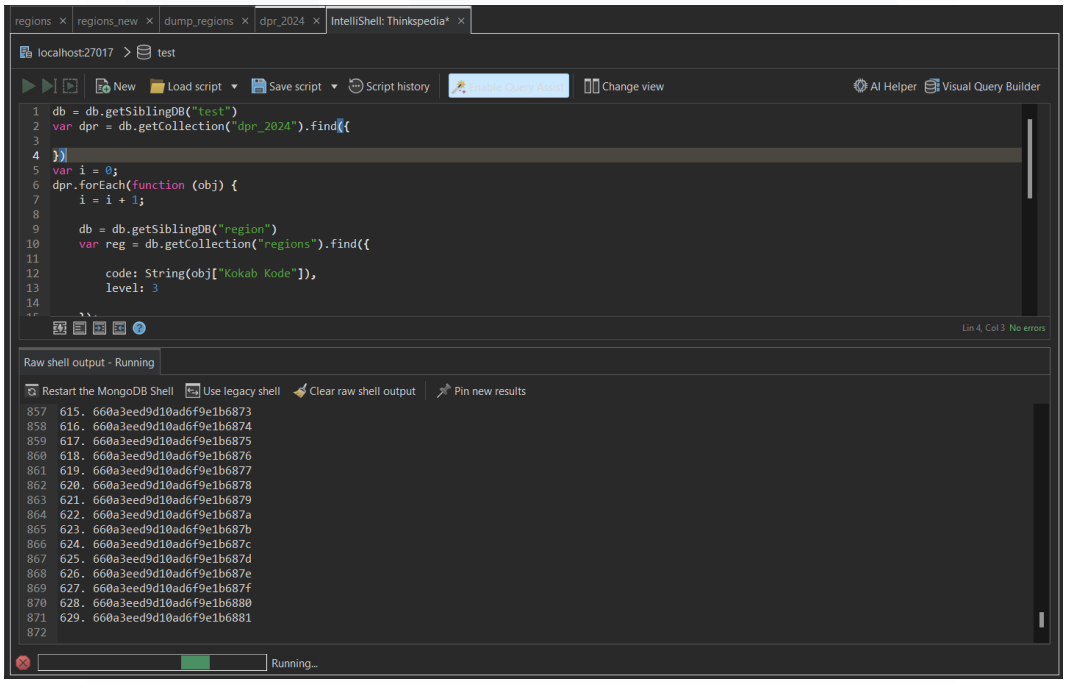


Gambar 3. 12 Skrip untuk *update field 'city'*

Kode di atas adalah skrip MongoDB yang digunakan untuk memperbarui field *'city'* dalam dokumen pada koleksi dpr_2024 dengan data dari koleksi *'regions'*. Skrip dimulai dengan menghubungkan ke database 'test' dan mengambil semua dokumen dari koleksi *'dpr_2024'*, yang disimpan dalam variabel *'dpr'*. Sebuah variabel penghitung *'i'* diinisialisasi ke nol untuk melacak jumlah dokumen yang diproses. Dalam setiap *looping* melalui dokumen dpr, nilai *'i'* ditingkatkan untuk mencatat looping tersebut.

Selanjutnya, skrip menghubungkan ke database *'region'* dan mencari dokumen yang cocok di koleksi *'regions'* berdasarkan code (yang diambil dari field *'Kokab Kode'* pada dokumen *'dpr_2024'*) dan level yang bernilai 3. Jika ditemukan dokumen yang cocok dalam *'regions'*, field *'city'* pada dokumen dpr_2024 diperbarui dengan *'_id'* dari dokumen yang sesuai dalam *'regions'*. Dokumen dalam

‘dpr_2024’ kemudian diupdate dengan value atau nilai *‘city’* yang baru menggunakan metode *‘updateOne’*, yang menggunakan *‘_id’* dokumen untuk memastikan dokumen yang tepat diperbarui. Setelah pembaruan, skrip mencetak nomor looping dan *‘_id’* dari dokumen yang diperbarui untuk memberikan konfirmasi.

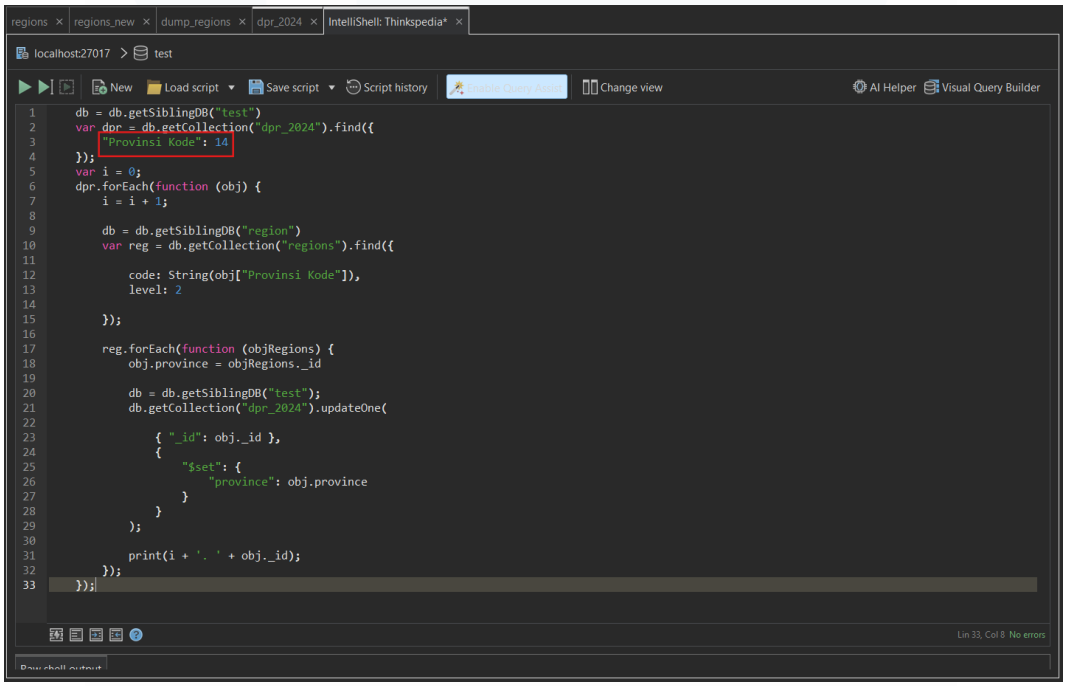


Gambar 3. 13 *Output* berjalan untuk *update field ‘city’*

Dengan demikian, skrip ini memastikan bahwa setiap dokumen dalam koleksi *‘dpr_2024’* yang memiliki *‘Kokab Kode’* sesuai dengan code dalam koleksi ‘regions’ akan memiliki field *‘city’* yang diperbarui dengan benar. Proses ini bertujuan untuk menjaga konsistensi dan keakuratan data antara dua koleksi dalam database yang berbeda. Hal ini sangat penting untuk memastikan bahwa informasi yang terintegrasi dan diperbarui dengan tepat dalam sistem basis data perusahaan. Dengan adanya pembaruan ini, setiap entri dalam *‘dpr_2024’* tidak hanya mencerminkan data suara pemilih secara akurat tetapi juga menghubungkan data tersebut dengan wilayah administratif yang tepat, memungkinkan analisis yang lebih terperinci dan berbasis lokasi.

Konsistensi data ini juga mempermudah proses pelaporan dan analisis, sehingga berbagai departemen dalam perusahaan dapat mengambil keputusan yang lebih tepat berdasarkan data yang akurat dan terbaru. Proses integrasi ini juga penting dalam mengidentifikasi dan mengurangi kesalahan data yang mungkin timbul dari ketidakcocokan informasi antara dua koleksi, yang pada gilirannya meningkatkan integritas keseluruhan dari sistem basis data perusahaan.

Dengan sistem yang terintegrasi dengan baik, perusahaan dapat memastikan bahwa semua data terkait pemilih dan pemilu tersedia secara real-time, memberikan dasar yang kuat untuk analisis strategis dan operasional yang lebih efektif.

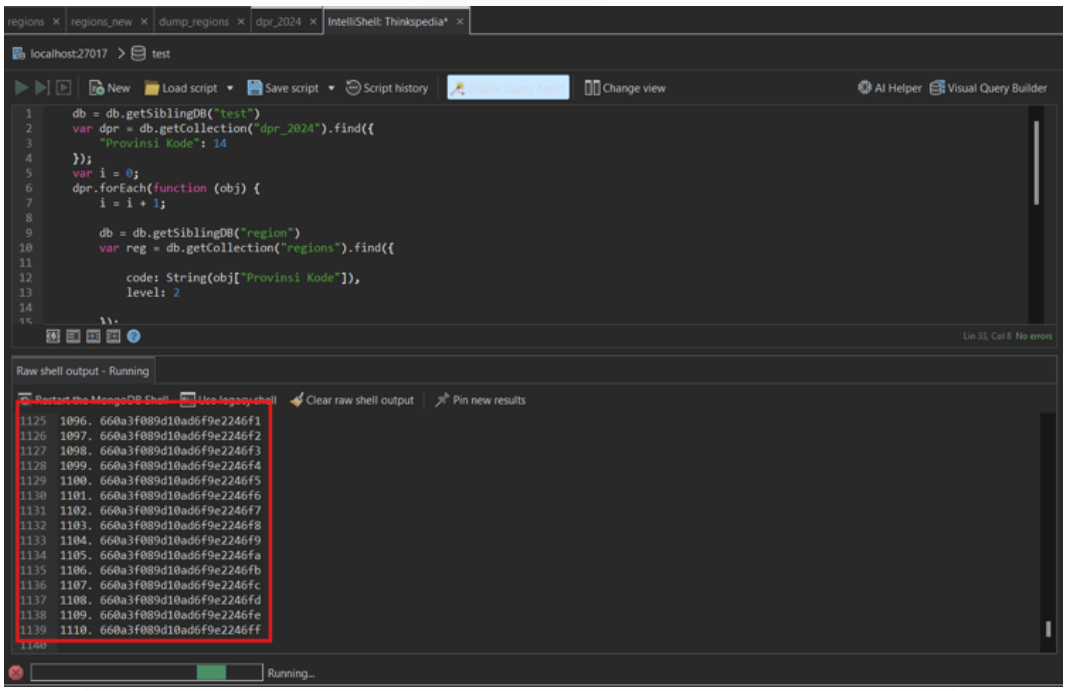


Gambar 3. 14 Skrip untuk update *field 'province'*

Kode di atas adalah skrip MongoDB yang digunakan untuk memperbarui field province dalam dokumen pada koleksi *'dpr_2024'* berdasarkan data dari koleksi 'regions'. Skrip dimulai dengan menghubungkan ke database *'test'* dan mengambil semua dokumen dari koleksi *'dpr_2024'* yang memiliki **Provinsi Kode bernilai 14**.

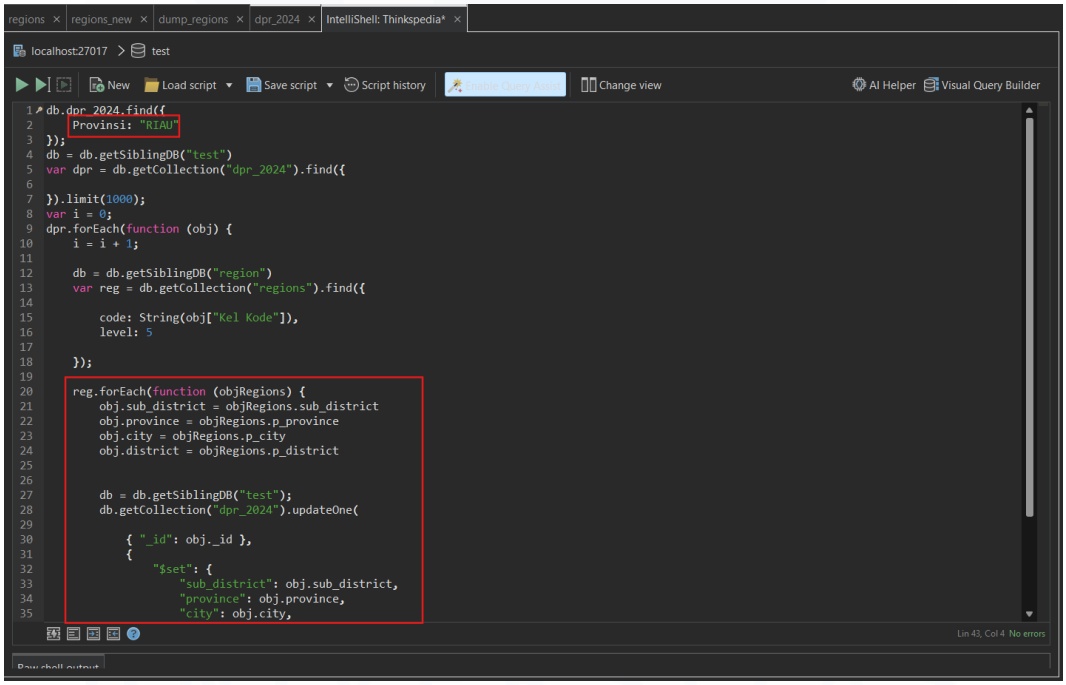
Dokumen-dokumen ini disimpan dalam variabel *'dpr'*. Variabel penghitung *'i'* diinisialisasi ke nol untuk melacak jumlah dokumen yang diproses. Selanjutnya, skrip melakukan *looping* setiap dokumen dalam *'dpr'* menggunakan fungsi *'forEach'*, di mana setiap kali sebuah dokumen diproses, nilai 'i' ditingkatkan.

Untuk setiap dokumen dalam *'dpr_2024'*, skrip menghubungkan ke database 'region' dan mencari dokumen yang cocok di koleksi *'regions'* berdasarkan code (yang diambil dari field *'Provinsi Kode'* pada dokumen *'dpr_2024'*) dan level yang bernilai 2. Jika ditemukan dokumen yang cocok dalam regions, field province pada dokumen *'dpr_2024'* diperbarui dengan *'_id'* dari dokumen yang sesuai dalam regions. Setelah pembaruan, dokumen dalam *'dpr_2024'* diupdate dengan nilai province yang baru menggunakan metode *'updateOne'*, yang memastikan bahwa dokumen yang tepat diperbarui berdasarkan *'_id'*.



Gambar 3. 15 *Output* berjalan untuk *update field 'province'*

Terakhir, skrip mencetak nomor looping dan *'_id'* dari dokumen yang diperbarui untuk memberikan konfirmasi bahwa dokumen tersebut telah diproses. Dengan demikian, skrip ini memastikan bahwa setiap dokumen dalam koleksi *'dpr_2024'* yang memiliki *'Provinsi Kode'* sesuai dengan code dalam regions akan memiliki field *'province'* yang diperbarui dengan benar, menjaga konsistensi dan keakuratan data antara dua koleksi dalam database yang berbeda. Proses ini sangat penting untuk memastikan bahwa informasi yang terintegrasi dan diperbarui dalam sistem basis data perusahaan tetap tepat dan akurat. Dengan sistem yang terintegrasi dengan baik, perusahaan dapat memastikan bahwa semua data terkait pemilih dan pemilu tersedia secara real-time, memberikan dasar yang kuat untuk analisis strategis dan operasional yang lebih efektif.



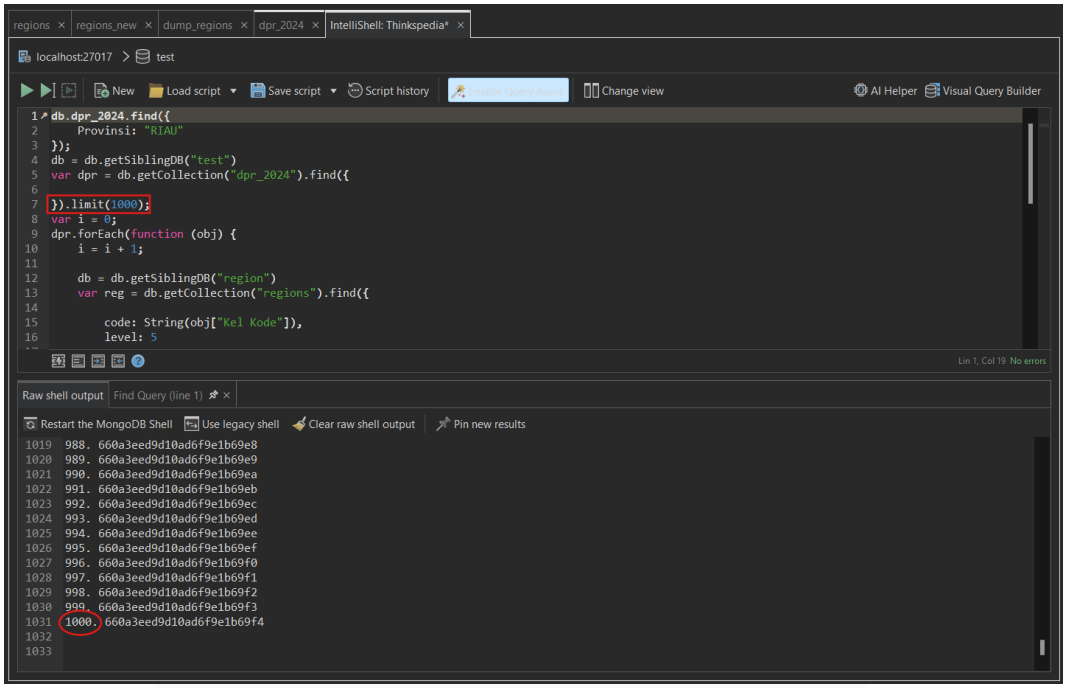
Gambar 3. 16 Skrip untuk *update field 'city'*

Kode di atas adalah skrip MongoDB yang digunakan untuk memperbarui beberapa field dalam dokumen pada koleksi *'dpr_2024'* berdasarkan data dari koleksi *'regions'*. Skrip ini dimulai dengan menghubungkan ke database *'test'* dan mengambil hingga **1000**

dokumen dari koleksi *'dpr_2024'*, yang kemudian disimpan dalam variabel 'dpr'. Langkah pertama ini memastikan bahwa data yang akan diolah terbatas, memungkinkan proses pembaruan yang lebih efisien dan terkendali. Variabel penghitung *'i'* diinisialisasi ke nol untuk melacak jumlah dokumen yang diproses selama *looping*.

Selanjutnya, skrip melakukan *looping* pada setiap dokumen dalam *'dpr'* menggunakan fungsi *'forEach'*. Setiap kali sebuah dokumen diproses, nilai *'i'* ditingkatkan satu untuk mencatat jumlah *looping* yang sedang berlangsung. Langkah ini penting untuk memantau kemajuan dan memastikan bahwa setiap dokumen diolah secara individual dan terpisah. Dalam setiap *looping*, skrip menghubungkan kembali ke database region untuk mencari dokumen yang cocok di koleksi *'regions'*. Pencarian ini didasarkan pada code yang diambil dari field *'Kel Kode'* pada dokumen *'dpr_2024'*, dan level yang bernilai 5. Kriteria pencarian ini memastikan bahwa dokumen yang relevan dan spesifik diidentifikasi dalam koleksi *'regions'*.

Jika ditemukan dokumen yang cocok dalam *'regions'*, skrip memperbarui beberapa field dalam dokumen *'dpr_2024'*. Field yang diperbarui meliputi *'sub_district'*, *'province'*, *'city'*, dan *'district'*, dengan nilai-nilai yang sesuai dari dokumen *'regions'*. Misalnya, *'obj.sub_district'* diperbarui dengan nilai *'objRegions.sub_district'*, *'obj.province'* diperbarui dengan nilai *'objRegions.p_province'*, dan seterusnya. Langkah ini memastikan bahwa data dalam *'dpr_2024'* mencerminkan informasi terkini dan akurat dari koleksi regions.

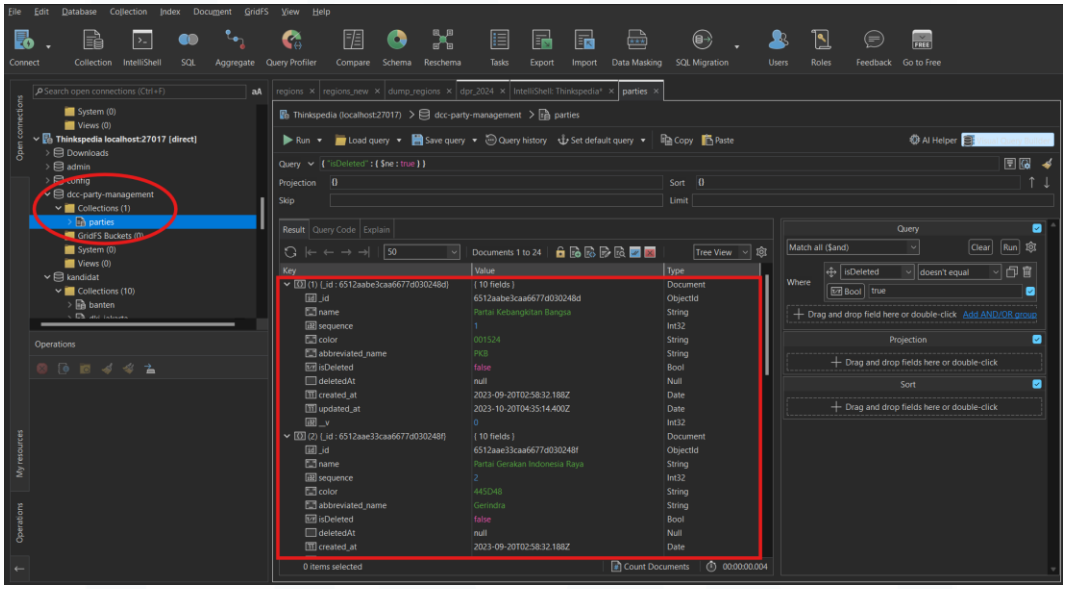


Gambar 3. 17 Berhasil menampilkan 1000 *output*

Setelah memperbarui field dalam objek dokumen *'dpr_2024'*, skrip kemudian menghubungkan kembali ke database *'test'* dan memperbarui dokumen tersebut dalam koleksi *'dpr_2024'* menggunakan metode 'updateOne'. Metode ini menggunakan *'_id'* dokumen untuk memastikan bahwa dokumen yang tepat diperbarui. Perintah pembaruan ini menetapkan field *'sub_district'*, *'province'*, *'city'*, dan *'district'* dengan nilai-nilai baru yang telah diperoleh dari koleksi *'regions'*. Setelah pembaruan berhasil dilakukan, skrip mencetak nomor *looping* dan *'_id'* dari dokumen yang diperbarui untuk memberikan konfirmasi bahwa dokumen tersebut telah diproses. Langkah ini memberikan umpan balik langsung dan membantu dalam memverifikasi bahwa pembaruan telah dilakukan dengan benar.

Dengan demikian, skrip ini memastikan bahwa setiap dokumen dalam koleksi *'dpr_2024'* yang memiliki *'Kel Kode'* sesuai dengan code dalam *'regions'* akan memiliki field *'sub_district'*, *'province'*, *'city'*, dan *'district'* yang diperbarui dengan benar. Proses ini sangat

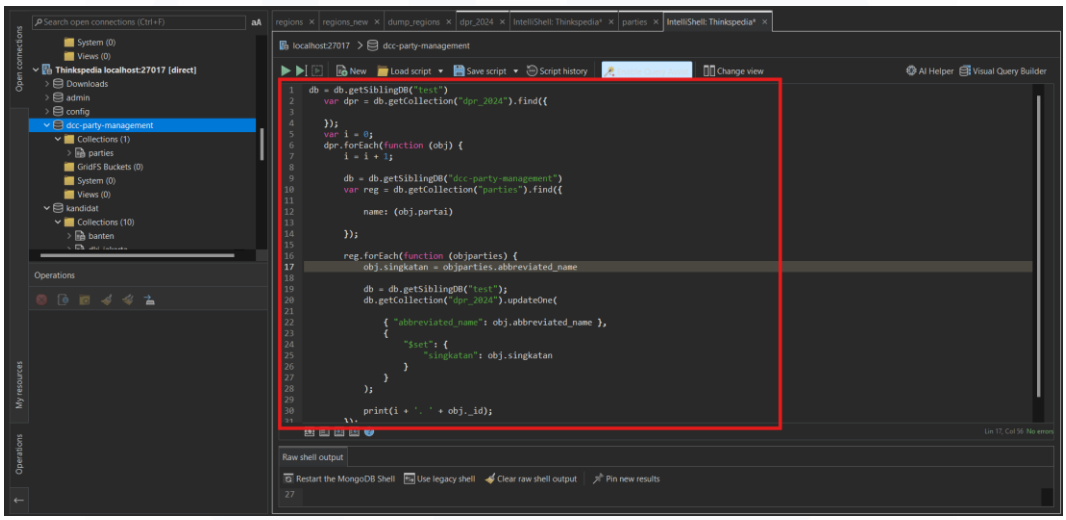
penting untuk menjaga konsistensi dan keakuratan data antara dua koleksi dalam database yang berbeda. Dengan memastikan bahwa informasi yang terintegrasi dan diperbarui dalam sistem basis data perusahaan tetap tepat dan akurat, skrip ini berkontribusi pada pengelolaan data yang lebih efisien dan dapat diandalkan. Proses pembaruan data ini juga mendukung berbagai analisis dan pelaporan yang lebih baik, memungkinkan perusahaan untuk mengambil keputusan berdasarkan data yang akurat dan up-to-date.



Gambar 3. 18 Tampilan untuk data koleksi *'parties'*

Gambar 3.20 menunjukkan bagian dari database *'dcc-party-management'*, yang dikembangkan untuk mengelola dan menyimpan data tentang berbagai partai politik di Indonesia. Dalam tampilan ini, dapat dilihat di koleksi *'parties'* yang diorganisir dalam bentuk *documents* yang mencakup urutan dan beberapa atribut penting lainnya dari masing-masing partai. Setiap entri dalam database ini dilengkapi dengan informasi detail seperti ID partai, nama, serta beberapa parameter lain yang relevan, yang ditujukan untuk mendukung manajemen data secara efisien. Database ini dirancang khusus untuk memudahkan pengguna dalam mengakses dan mengolah data partai-

partai politik yang berkontribusi dalam penyusunan data pemilu atau data politik lainnya yang sedang dikerjakan oleh perusahaan, sehingga memastikan bahwa data yang dihasilkan akurat dan terpercaya untuk kebutuhan analisis dan strategi politik.



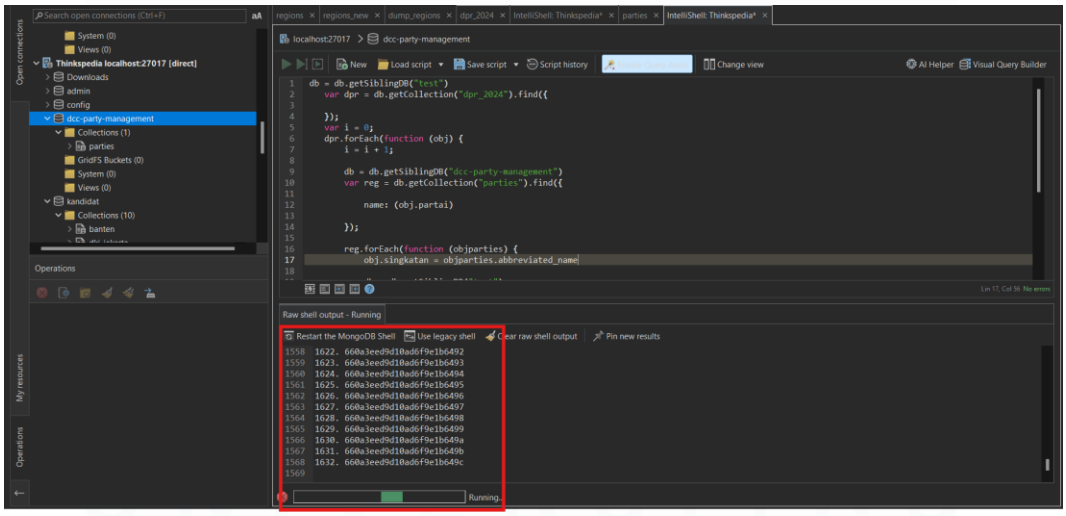
Gambar 3. 19 Skrip untuk mengoptimalkan dan memperbarui data

Gambar 3.21 diatas menampilkan kode untuk mengoptimalkan dan memperbarui data secara efisien antara dua koleksi dalam database yang berbeda. Kode ini berfungsi khusus untuk menyinkronkan dan memperbarui informasi tentang partai politik di antara koleksi *'dpr_2024'* dalam database test dan koleksi parties dalam database *'dcc-party-management'*. Tujuan utama skrip ini adalah untuk memastikan bahwa data mengenai singkatan nama partai politik selalu terkini dan konsisten di semua entri relevan.

Skrip diawali dengan proses koneksi ke database test dan mengambil semua entri dari koleksi *'dpr_2024'*. Setiap dokumen dari koleksi ini disimpan sementara dalam variabel *'dpr'*. Skrip juga menginisialisasi variabel penghitung *'i'* ke nilai nol. Variabel ini berperan penting untuk melacak jumlah dokumen yang telah diproses, yang membantu dalam pemantauan kemajuan skrip secara real-time.

Selanjutnya, menggunakan metode *'forEach'*, skrip melaksanakan *looping* atas setiap dokumen yang ditarik dari koleksi *'dpr_2024'*. Pada setiap tahap *looping*, variabel *'i'* ditingkatkan, menandakan proses satu dokumen telah selesai. Dalam setiap *looping*, skrip mengganti koneksi ke database *'dcc-party-management'*. Di sini, skrip mencari dokumen yang cocok dalam koleksi parties berdasarkan nama partai yang diambil dari dokumen yang sedang diproses dalam *'dpr_2024'*.

Jika ditemukan dokumen yang cocok, skrip kemudian mengambil nama singkatan dari dokumen tersebut dan menyimpan informasi ini dalam field 'singkatan' pada dokumen asal di *'dpr_2024'*. Setelah data 'singkatan' diperoleh, skrip kembali ke database *'test'* dan melakukan pembaruan pada dokumen asal menggunakan metode *'updateOne'*. Metode ini secara spesifik memperbarui field *'singkatan'* berdasarkan *'abbreviated_name'* yang baru diperoleh, memastikan bahwa hanya dokumen yang tepat yang diperbarui, menghindari kesalahan pembaruan yang tidak diinginkan.

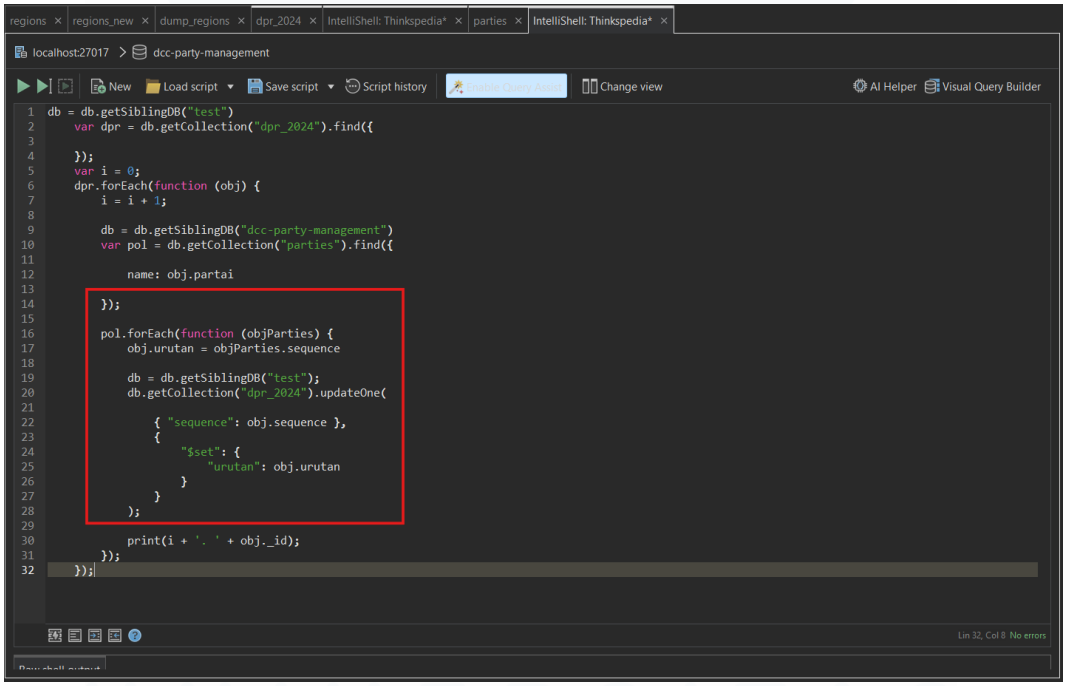


Gambar 3. 20 Code yang berjalan menghasilkan *output*

Proses pembaruan ini juga diiringi dengan pencetakan output yang mencantumkan nomor *looping* dan *'_id'* dari dokumen yang telah diperbarui, memberikan umpan balik visual yang membantu dalam

verifikasi dan audit proses. Output ini esensial untuk memastikan bahwa setiap langkah dalam proses pembaruan dapat dilacak dan diverifikasi oleh pengguna, memudahkan pengelolaan data dan *troubleshooting* jika diperlukan.

Dengan mengimplementasikan skrip ini, pengelola data legislatif dan partai politik dapat memastikan bahwa informasi mereka selalu akurat dan terkini. Skrip ini tidak hanya meningkatkan efisiensi dalam pengelolaan data tetapi juga mendukung integritas data dalam aplikasi yang sangat bergantung pada keakuratan informasi untuk analisis dan pengambilan keputusan. Skrip seperti ini adalah bagian integral dari strategi manajemen data yang efektif, memastikan konsistensi dan keandalan informasi di seluruh sistem database organisasi.



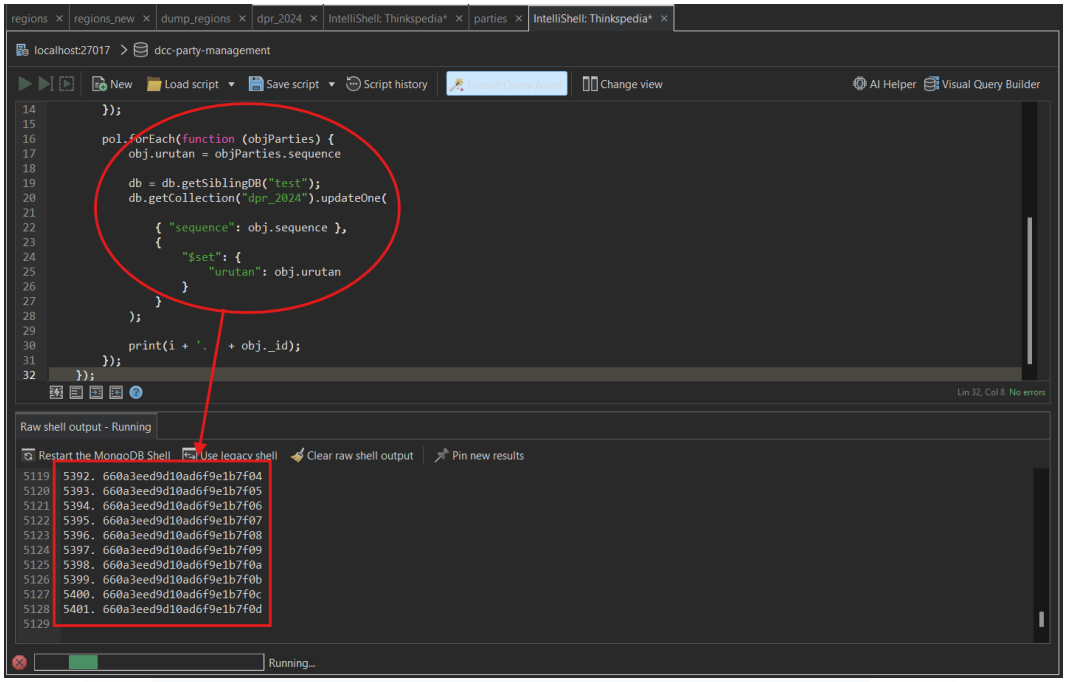
Gambar 3. 21 Skrip untuk menyinkronkan dan memperbarui *'urutan'*

Kode di atas adalah skrip yang dirancang untuk mengoptimalkan dan memperbarui data antara dua koleksi dalam database yang berbeda. Khususnya, kode ini berfungsi untuk menyinkronkan dan memperbarui urutan partai politik antara koleksi *'dpr_2024'* dalam database *'test'*

dan koleksi *'partai'* dalam database *'dcc-party-management'*. Data tentang urutan partai politik harus selalu akurat dan konsisten di setiap entri yang relevan, menurut tujuan utama skrip ini.

Proses koneksi ke database *'test'* dan pengumpulan semua entri dari koleksi *'dpr_2024'* adalah langkah awal dari skrip. Variabel *'dpr'* disimpan sementara dalam setiap dokumen koleksi ini. Skrip juga menginisialisasi variabel penghitung *'i'* ke nilai nol. Untuk memantau jumlah dokumen yang telah diproses secara real-time, variabel ini sangat penting.

Selanjutnya, skrip melakukan iterasi pada setiap dokumen yang ditarik dari koleksi *'dpr_2024'* dengan metode *'forEach'*. Pada setiap tahap *looping*, variabel 'i' diinkremen, yang menunjukkan bahwa proses untuk satu dokumen telah selesai. Skrip mengganti koneksi ke database *'dcc-party-management'* setiap kali melakukan langkah ini. Di sini, skrip mencari dokumen koleksi partai yang sesuai berdasarkan nama partai, yang diambil dari dokumen yang sedang diproses dalam *'dpr_2024'*.



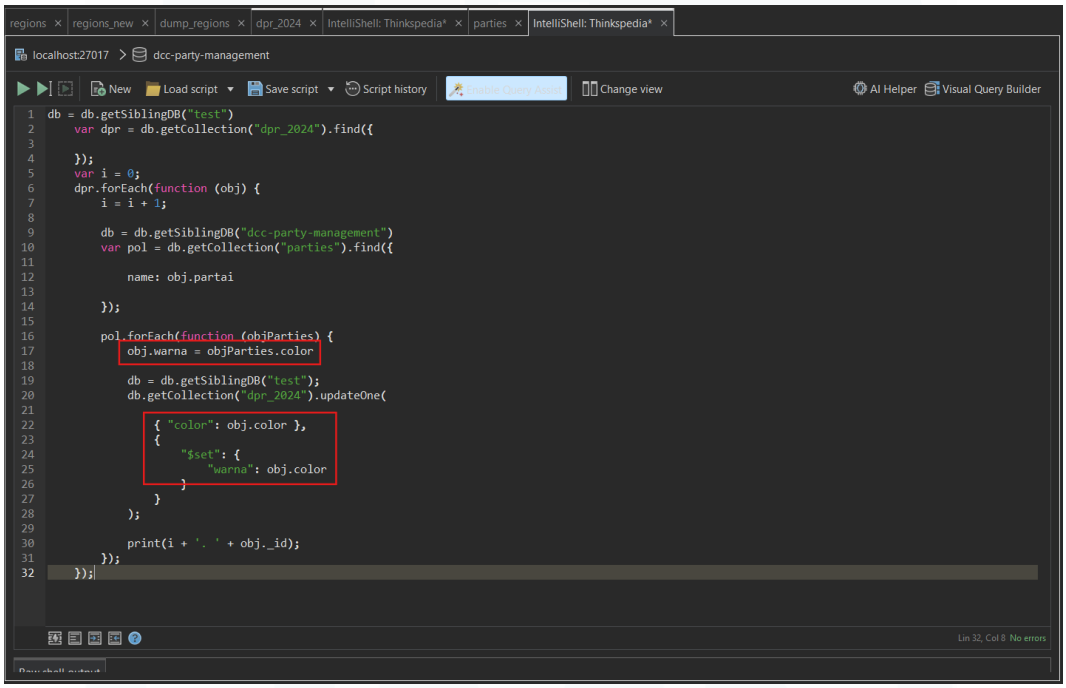
Gambar 3. 22 Skrip yang dijalankan berhasil

Jika dokumen yang dimaksud ditemukan, skrip kemudian mengambil urutan dari dokumen tersebut dan menyimpan informasi ini dalam field urutan pada dokumen awal di *'dpr_2024'*. Setelah data 'urutan' diperoleh, skrip kembali ke database uji dan menggunakan metode *'updateOne'* untuk memperbarui dokumen awal, memastikan hanya dokumen yang tepat yang diperbarui.

Selain itu, pembaruan diikuti dengan pencetakan output yang mencantumkan nomor *looping* dan *'_id'* dari dokumen yang telah diperbarui, ini memberikan umpan balik visual yang membantu dalam verifikasi dan audit proses. Output ini sangat penting untuk memastikan bahwa pengguna dapat melacak dan memverifikasi setiap langkah dalam proses pembaruan ini memudahkan pengelolaan data dan penanganan masalah jika diperlukan.

Pengelola data legislatif dan partai politik dapat memastikan bahwa informasi mereka selalu akurat dan terkini dengan menerapkan skrip ini. Skrip ini tidak hanya meningkatkan efisiensi pengelolaan data tetapi

juga memastikan integritas data dalam aplikasi yang sangat bergantung pada keakuratan informasi untuk analisis dan pengambilan keputusan. Skrip seperti ini adalah bagian penting dari strategi manajemen data yang efektif, yang memastikan konsistensi dan keandalan informasi di seluruh sistem database organisasi.



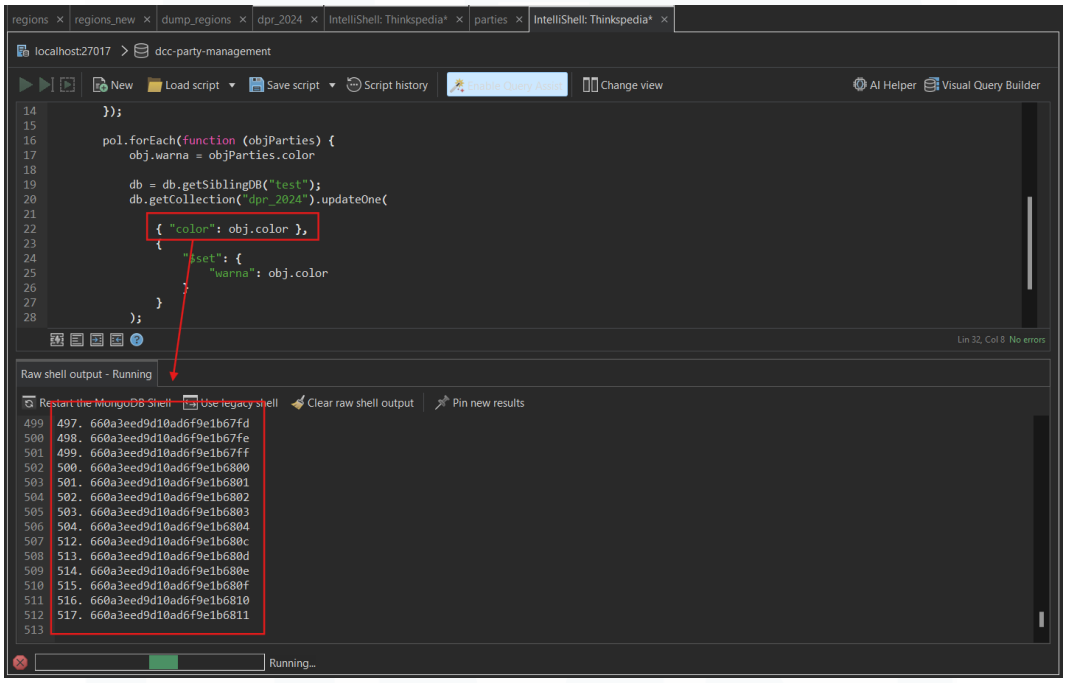
Gambar 3. 23 Skrip untuk menyinkronkan dan memperbarui *'warna'*

Kode tersebut dimulai dengan melakukan koneksi ke database test. Ini memastikan bahwa operasi berikutnya dilakukan dalam database test. Selanjutnya, semua dokumen dari koleksi *'dpr_2024'* diambil menggunakan dan disimpan dalam variabel *'dpr'*. Untuk melacak jumlah *looping*, variabel *'i'* diinisialisasi dengan nilai 0.

Kemudian, kode ini melakukan *looping* pada setiap dokumen dalam koleksi *'dpr_2024'* menggunakan metode *'forEach'*. Setiap kali *looping* dilakukan, variabel *'i'* ditambah satu untuk menjaga penghitung *looping*. Dalam setiap *looping*, konteks database diubah ke *'dcc-party-*

management' untuk mengakses *'koleksi parties'* yang ada di database *'dcc-party-management'*.

Setelah mengubah konteks, kode mengambil dokumen dari koleksi *'parties'* yang memiliki nama partai yang sama dengan nilai *'obj.partai'* dari dokumen *'dpr_2024'* saat ini. Dokumen yang cocok disimpan dalam variabel *'pol'*.

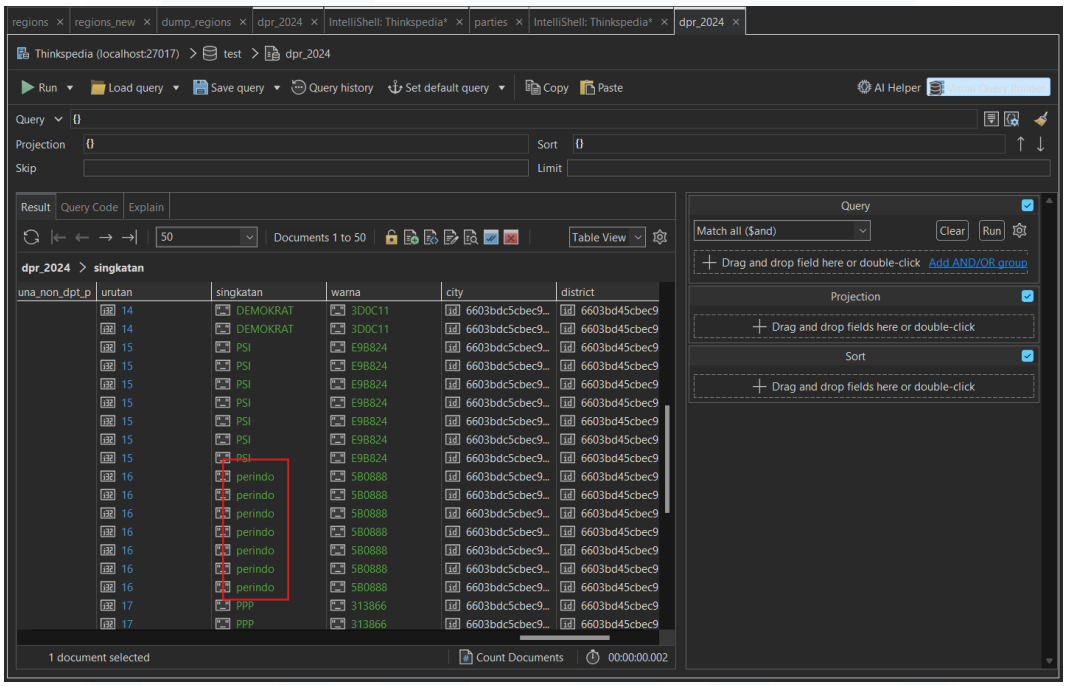


Gambar 3. 24 Skrip yang dijalankan berhasil

Setiap dokumen hasil pencarian dalam pol kemudian *looping* lagi menggunakan *'forEach'*. Dalam *looping* ini, field *'warna'* pada dokumen *'dpr_2024'* diperbarui dengan nilai *'warna'* (color) dari dokumen *'parties'*. Kode ini mengembalikan konteks ke database *'test'* untuk memperbarui dokumen di koleksi *'dpr_2024'*.

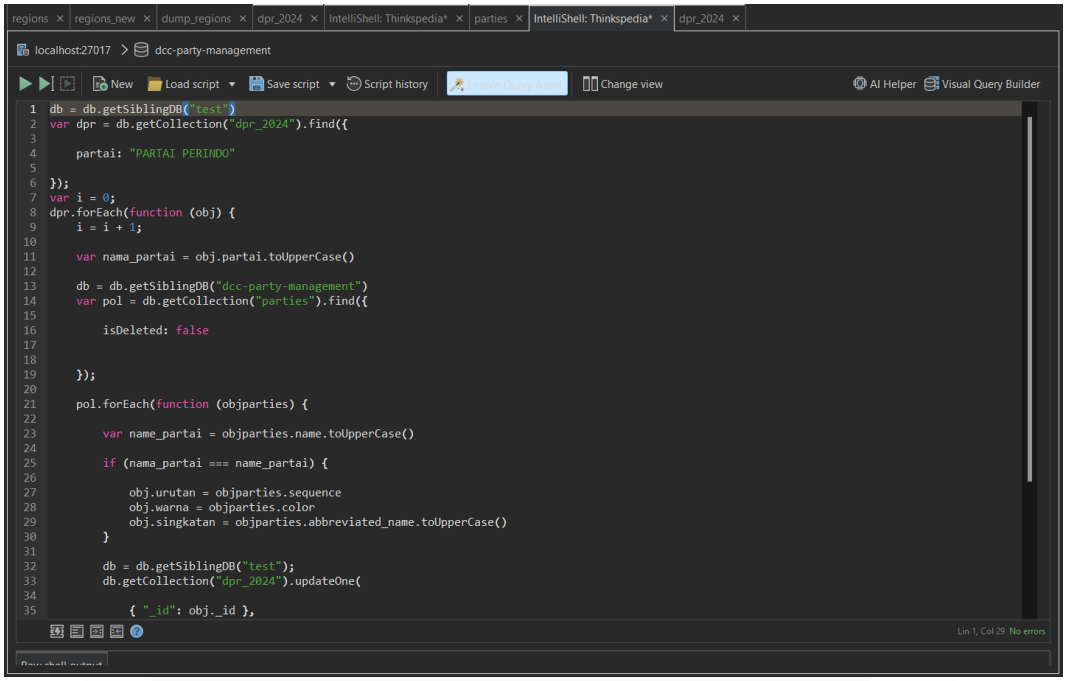
Dalam konteks database *'test'*, dokumen dalam koleksi *'dpr_2024'* diperbarui menggunakan untuk memastikan bahwa field warna dalam dokumen *'dpr_2024'* diisi dengan nilai warna yang diambil dari dokumen *'parties'*.

Terakhir, kode mencetak ID dari setiap dokumen yang diperbarui untuk membantu melacak dokumen mana yang telah diproses dan memperbarui informasi warna partainya. Secara keseluruhan, kode ini berguna untuk menggabungkan informasi warna partai dari koleksi *'parties'* ke dalam koleksi *'dpr_2024'* berdasarkan nama partai yang cocok.

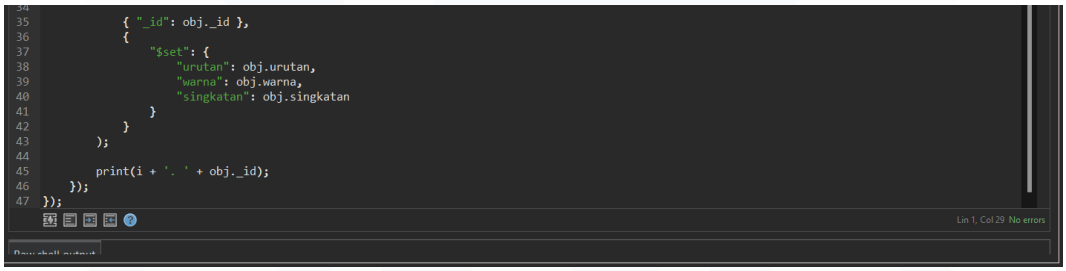


Gambar 3. 25 Tampilan singkatan 'perindo' sebelum di*update*

Kode yang akan digunakan selanjutnya adalah skrip yang dirancang untuk mengelola dan memperbarui data antara dua database yang berbeda. Awalnya, skrip mengatur koneksi ke database yang bernama *'test'* dan melakukan query untuk mendapatkan semua dokumen dari koleksi *'dpr_2024'* yang mengandung entri dengan field partai bertuliskan "PARTAI PERINDO". Proses ini merupakan langkah pertama dalam menyiapkan data yang akan disinkronkan dan diperbarui berdasarkan informasi dari sumber lain.

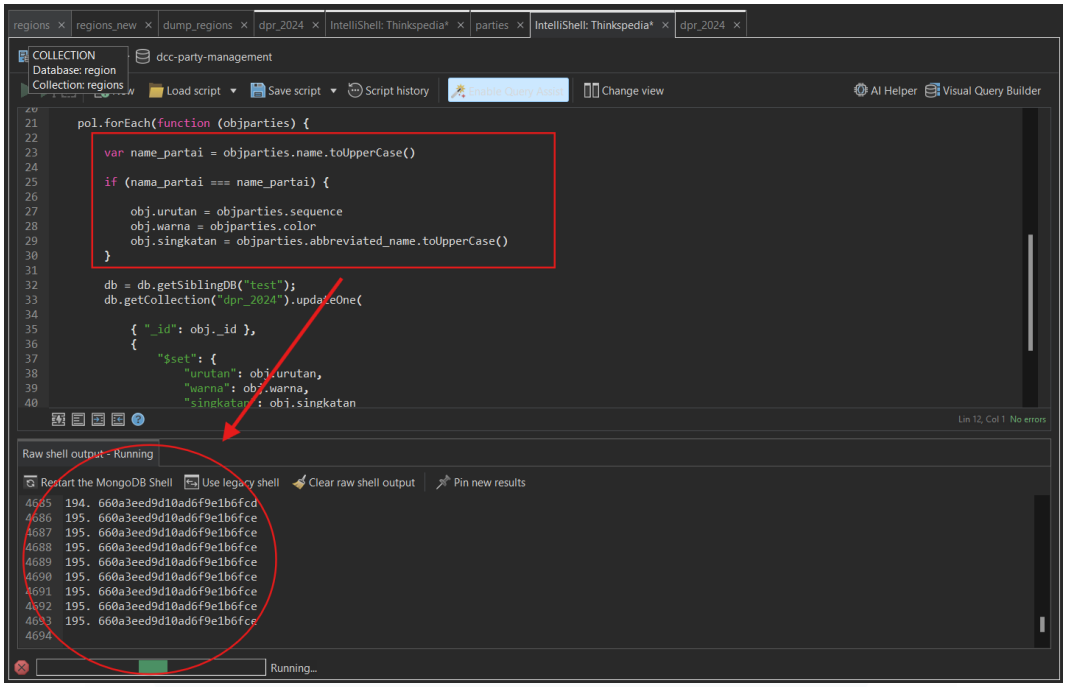


Gambar 3. 26 Skrip yang digunakan untuk *update*



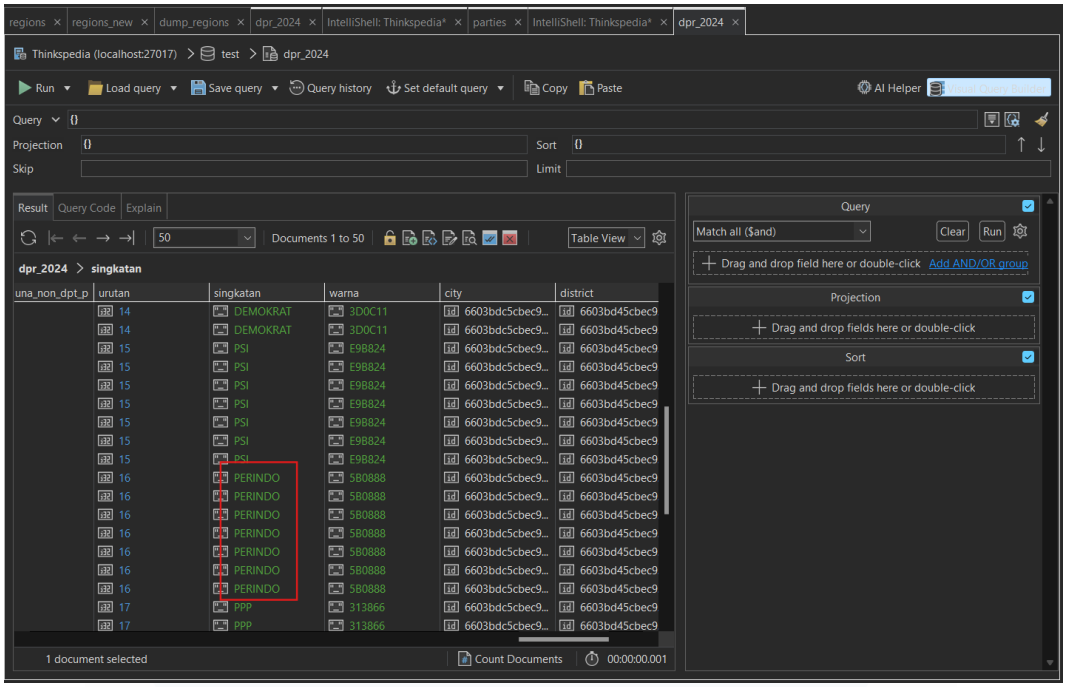
Gambar 3. 27 Lanjutan skrip yang digunakan untuk *update*

Setelah mendapatkan dokumen yang relevan dari koleksi *'dpr_2024'*, skrip melanjutkan untuk melakukan *looping* pada setiap dokumen tersebut. Dalam setiap *looping*, skrip mencatat dan mengubah nama partai ke bentuk huruf besar untuk memfasilitasi proses pencocokan yang tidak sensitif terhadap kapitalisasi. Langkah ini krusial karena memungkinkan penyesuaian data yang akan dilakukan secara akurat, memastikan bahwa setiap perbandingan nama partai antara dua koleksi data adalah konsisten dan dapat diandalkan.



Gambar 3. 28 Skrip yang dijalankan berhasil menghasilkan *output*

Dalam langkah berikutnya, skrip mengganti koneksi database ke *'dcc-party-management'*, dimana skrip ini kembali melakukan query, kali ini pada koleksi *'parties'*. Skrip mencari dokumen yang belum dihapus dan memproses setiap satu dari mereka untuk membandingkan nama partai dengan yang telah disiapkan sebelumnya dari database *'test'*. Tujuan dari proses ini adalah untuk menemukan kesamaan antara entri di kedua database dan menyinkronkan informasi tambahan seperti urutan, warna, dan singkatan partai.



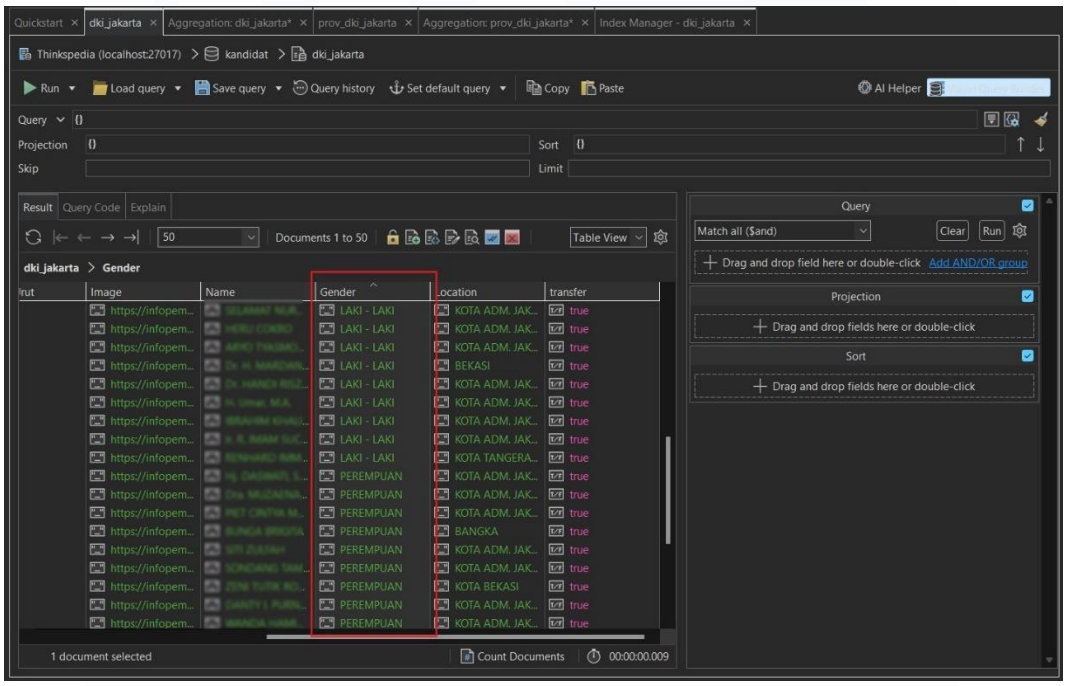
Gambar 3. 29 Tampilan ‘PERINDO’ setelah *update*

Apabila ditemukan kecocokan antara nama partai di kedua koleksi, skrip kemudian akan menyalin atribut-atribut seperti urutan, warna, dan nama singkat dari koleksi parties ke dokumen yang sesuai di koleksi *‘dpr_2024’*. Hal ini dilakukan melalui operasi update, di mana dokumen yang ada di database test diperbarui dengan informasi baru yang sesuai. Proses ini tidak hanya memperbarui data tetapi juga memastikan integritas dan konsistensi informasi antara koleksi yang berinteraksi dalam dua database yang berbeda.

Terakhir, setelah semua perubahan telah diterapkan, skrip mencetak ID dari setiap dokumen yang telah diperbarui bersama dengan urutan pemrosesannya. Ini memberikan umpan balik langsung ke pengguna tentang hasil dari operasi yang dilakukan, memungkinkan verifikasi manual jika diperlukan. Seluruh proses ini menunjukkan penggunaan MongoDB yang efektif dalam pengelolaan dan pemeliharaan data yang konsisten dan terintegrasi di berbagai koleksi dan database, sangat penting dalam banyak aplikasi bisnis dan

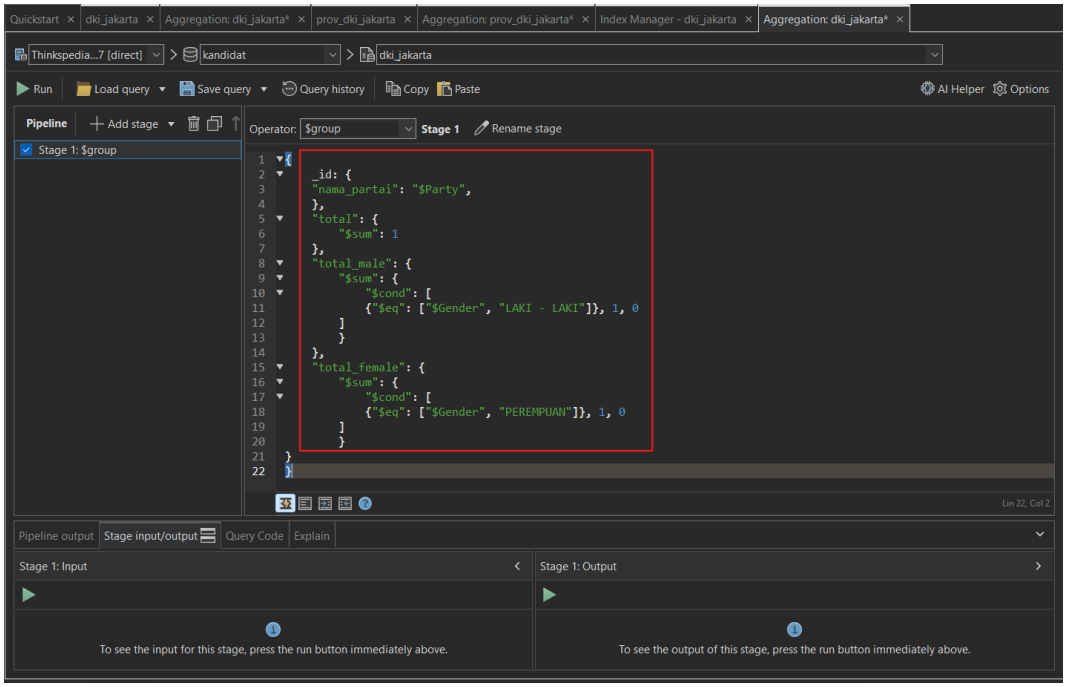
organisasi yang mengandalkan data terstruktur untuk operasional dan analisis keputusan.

3.2.4 Membuat Aggregation & Indexing pada Data perusahaan sesuai dengan atribut pada data yang dimiliki perusahaan.



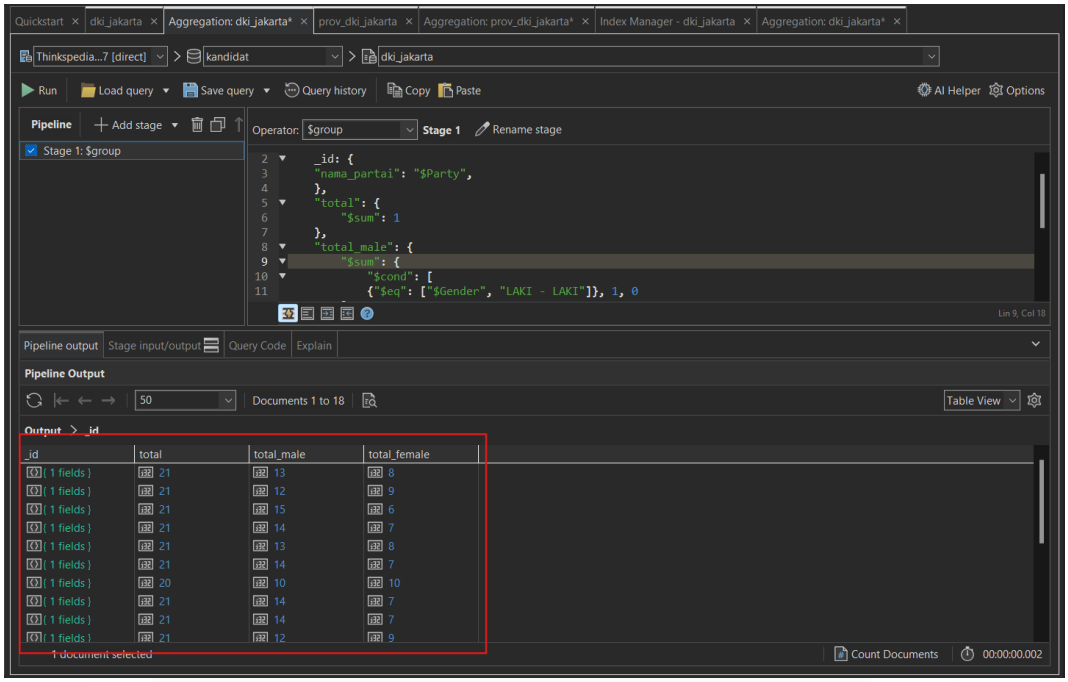
Gambar 3. 30 Tampilan koleksi *dpt_jakarta* pada bagian *'gender'*

Gambar 3.32 di atas menampilkan tampilan dari koleksi *"dki_jakarta"* yang berisi informasi tentang beberapa nama Calon Legislatif Daftar Pemilih Tetap (Caleg DPT) di Provinsi Jakarta. Koleksi ini dilengkapi dengan beberapa parameter penting, salah satunya adalah kolom 'Gender' yang memuat dua jenis kelamin para calon legislatif, yaitu Laki-laki dan Perempuan. Data ini sangat berguna untuk analisis statistik dan memahami distribusi gender di antara calon legislatif, yang dapat memberikan wawasan penting mengenai inklusivitas dan kesetaraan gender dalam konteks politik di Jakarta. Informasi semacam ini juga memungkinkan pengguna untuk melakukan penyaringan dan pencarian data secara lebih spesifik, memudahkan dalam mengevaluasi dan memahami dinamika politik lokal berdasarkan representasi gender.



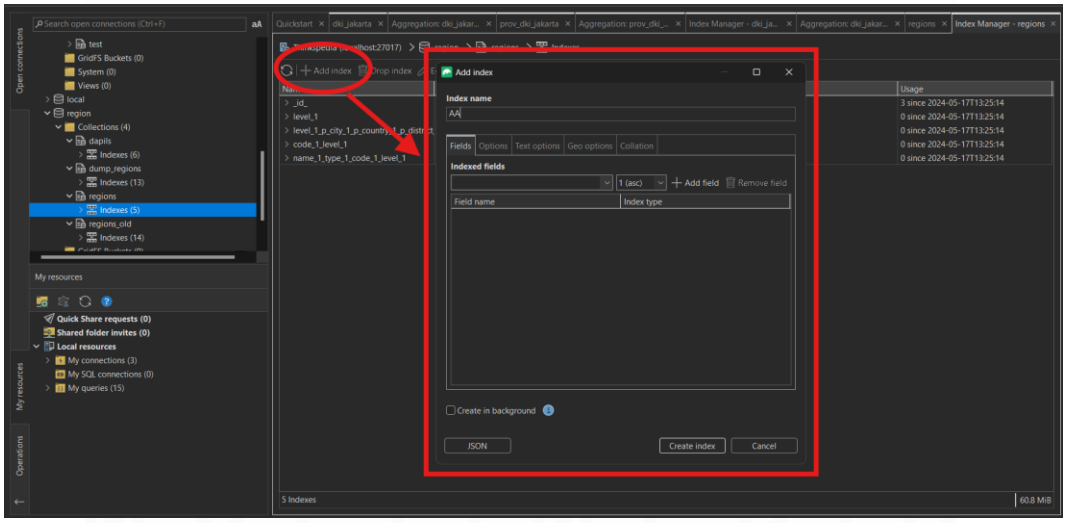
Gambar 3. 31 Skrip yang digunakan untuk melakukan *aggregate*

Kode tersebut adalah bagian dari pipeline agregasi dalam MongoDB yang digunakan untuk mengelompokkan dan menghitung data berdasarkan field *'Party'*. Kode ini mengelompokkan dokumen berdasarkan nama partai dan menghitung jumlah total dokumen, jumlah total individu laki-laki, dan jumlah total individu perempuan dalam setiap grup. Untuk setiap dokumen, ia menambahkan satu ke total umum. Untuk menentukan gender, ia menggunakan kondisi; jika field *'Gender'* sama dengan "LAKI - LAKI", ia menambahkan satu ke total laki-laki, dan jika sama dengan "PEREMPUAN", ia menambahkan satu ke total perempuan. Hasilnya adalah kumpulan data yang menggambarkan distribusi gender di setiap partai politik.



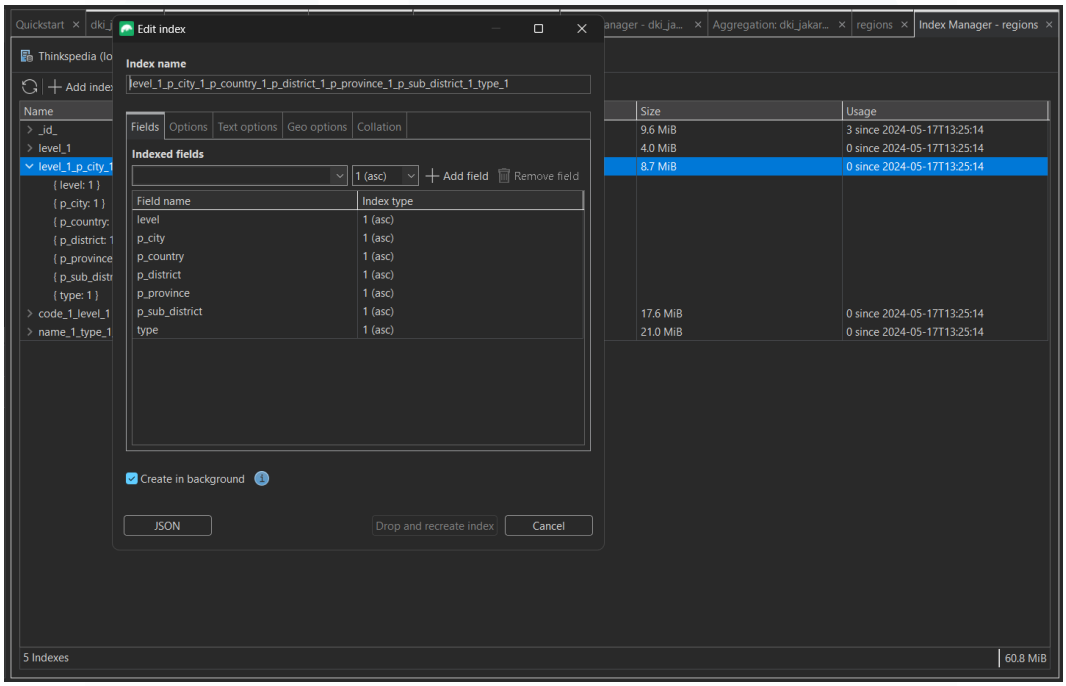
Gambar 3. 32 Hasil output dari *aggregate*

Berikut adalah hasil output dari kode tersebut, dapat dilihat disini bahwa hasil dari *Aggregation* yang telah dibuat sebelumnya dapat menampilkan total dari caleg tiap parta berdasarkan jenis kelamin dari para caleg tersebut.



Gambar 3. 33 Pembuatan *index* pada *software* Studio3T

Pada gambar 3.35 diatas, menunjukan salah satu fitur dari tools Studio3T yang dimana penggunanya mampu untuk membuat *Indexing* tanpa melalui code. Hal ini bukan berarti tidak bisa namun Studio3T memberikan kemudahan dengan fitur ini, dapat dilihat diatas adalah tab untuk mendambahkan index untuk setiap koleksi pada database.



Gambar 3. 34 Index yang digunakan mahasiswa

Diatas merupakan salah satu Index yang dibuat pada koleksi *'regions'*. Fields yang diindeks adalah *'level'*, *'p_city'*, *'p_country'*, *'p_district'*, *'p_province'*, *'p_sub_district'*, dan *'type'*, semuanya diurutkan secara *ascending* (*asc*).

Index ini dibuat untuk mempercepat pencarian dan query pada kombinasi field tersebut, sehingga akses dan pengambilan data yang melibatkan kombinasi field tersebut menjadi lebih efisien. Dengan adanya index ini, database dapat dengan cepat menemukan data yang relevan berdasarkan urutan field yang diindeks, mengurangi waktu yang dibutuhkan untuk pencarian dibandingkan dengan pencarian tanpa index.

3.2.5 Melakukan Scrapping Data dengan tujuan sebagai referensi perbaikan dan memanipulasi serta memodifikasi data dari perusahaan sesuai dengan ketentuan perusahaan.

Proses ini dimulai dengan pengumpulan data, di mana skrip Python digunakan untuk mengakses dan mengumpulkan data dari berbagai sumber online yang relevan. Penting untuk memastikan bahwa data yang diambil sesuai dengan ketentuan perusahaan dan legalitas yang berlaku untuk menghindari masalah hukum dan memastikan integritas data.

Setelah data dikumpulkan, langkah berikutnya adalah pembersihan dan pemrosesan data. Data mentah sering kali mengandung duplikasi, kesalahan, dan inkonsistensi yang perlu dihilangkan. Proses ini melibatkan penggunaan Python untuk membersihkan dan melakukan transformasi data sesuai dengan kebutuhan analisis lebih lanjut. Tujuannya adalah untuk menghasilkan data yang bersih dan siap untuk dianalisis.

Penyimpanan data dilakukan dengan menggunakan Studio3T untuk MongoDB. Data yang telah diproses disimpan ke dalam database MongoDB dengan struktur yang optimal untuk memastikan akses dan penggunaan yang efisien. Studio3T digunakan untuk memanipulasi dan memodifikasi data sesuai kebutuhan analisis dan pelaporan. Pembaruan dan perubahan data dilakukan secara berkala untuk memastikan data selalu up-to-date.

Analisis data dilakukan dengan menggunakan teknik analisis untuk mengidentifikasi tren dan pola yang dapat digunakan untuk perbaikan strategi perusahaan. Hasil analisis ini diolah menjadi laporan dan visualisasi data yang mendukung pengambilan keputusan berbasis data. Implementasi hasil analisis data ke dalam strategi perusahaan dilakukan untuk memastikan bahwa strategi yang dijalankan berdasarkan data yang akurat dan relevan.

Terakhir, evaluasi berkala dilakukan untuk memastikan efektivitas dan efisiensi proses scraping dan analisis data. Dengan metode ini, perusahaan dapat memastikan bahwa data yang diperoleh tidak hanya akurat tetapi juga relevan dan dapat diandalkan untuk keperluan perbaikan dan pengembangan strategi perusahaan. Proses ini membantu perusahaan dalam mengoptimalkan penggunaan data untuk mendukung keputusan strategis dan operasional.

Index ini dibuat untuk mempercepat pencarian dan query pada kombinasi field tersebut, sehingga akses dan pengambilan data yang melibatkan kombinasi field tersebut menjadi lebih efisien. Dengan adanya index ini, database dapat dengan cepat menemukan data yang relevan berdasarkan urutan field yang diindeks, mengurangi waktu yang dibutuhkan untuk pencarian dibandingkan dengan pencarian tanpa index.

```python
import requests
from bs4 import BeautifulSoup
import pandas as pd

# URL dari halaman web yang berisi data DPT Jakarta
url = "https://kpu.go.id/index.php/pages/detail/2020/12/Daftar_Pemilih_Tetap_DPT_Provinsi_DKI_Jakarta"

# Melakukan permintaan GET ke halaman web
response = requests.get(url)
if response.status_code == 200:
    print("Berhasil mengakses halaman web")
else:
    print("Gagal mengakses halaman web")
    exit()

# Parsing konten halaman web menggunakan BeautifulSoup
soup = BeautifulSoup(response.content, "html.parser")

# Mencari tabel yang berisi data DPT
table = soup.find("table")

# Menyimpan data ke dalam list
data = []
for row in table.find_all("tr")[1:]:
    cols = row.find_all("td")
    cols = [col.text.strip() for col in cols]
    data.append(cols)

# Membuat DataFrame dari data yang dikumpulkan
df = pd.DataFrame(data, columns=["No", "Kecamatan", "Jumlah DPT", "Laki-laki", "Perempuan"])

# Menyimpan data ke file CSV
df.to_csv("dpt_jakarta.csv", index=False)
print("Data berhasil disimpan ke dpt_jakarta.csv")

# Menampilkan DataFrame
print(df)
```

Gambar 3. 35 Skrip yang digunakan untuk melakukan *scrapping data*

Kode Python di atas digunakan untuk melakukan web scraping data suara DPT (Daftar Pemilih Tetap) dari Jakarta dari sebuah situs

web, seperti situs Komisi Pemilihan Umum (KPU). Pertama, kode ini mengimpor tiga library penting: requests untuk mengakses halaman web, BeautifulSoup untuk memparsing HTML, dan pandas untuk mengelola dan menyimpan data dalam bentuk DataFrame.

Selanjutnya, kode ini mengakses halaman web yang berisi data DPT Jakarta menggunakan *requests.get().* Setelah memastikan bahwa permintaan berhasil dengan memeriksa status kodenya, konten halaman web kemudian diparsing menggunakan *BeautifulSoup*. Kode ini mencari elemen tabel yang berisi data DPT dengan menggunakan metode *soup.find("table")*

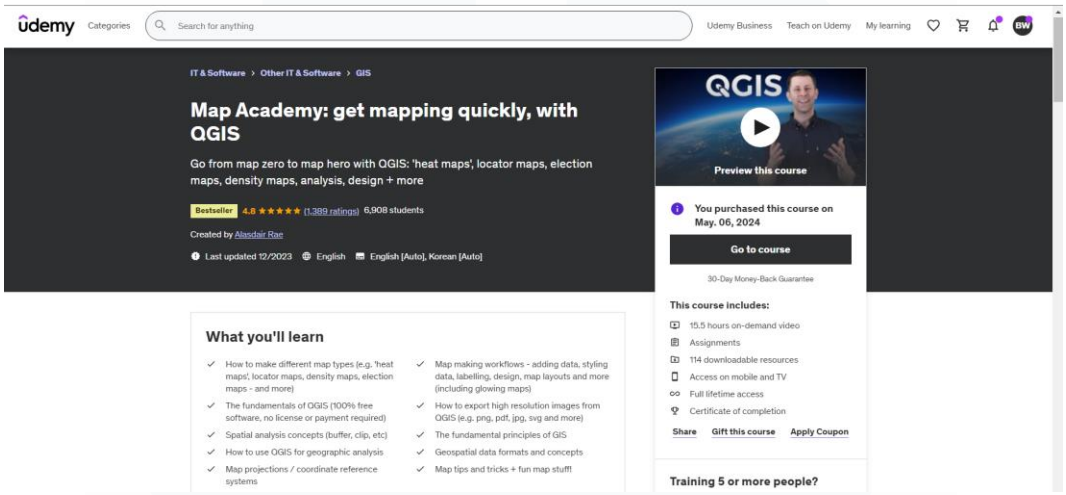
Setelah menemukan tabel yang relevan, kode ini mengekstrak data dari setiap baris tabel (kecuali header) dan menyimpannya dalam list data. Data ini kemudian diubah menjadi DataFrame menggunakan library pandas, dengan kolom yang sesuai dengan header tabel *("No", "Kecamatan", "Jumlah DPT", "Laki-laki", "Perempuan")*. DataFrame ini kemudian disimpan ke dalam file CSV bernama *dpt_jakarta.csv* menggunakan metode *to_csv().* Akhirnya, kode ini menampilkan DataFrame untuk memastikan bahwa data telah diambil dan disimpan dengan benar.

Proses ini membantu mengotomatisasi pengumpulan data suara DPT Jakarta dari sumber web, memastikan data disimpan dengan rapi dan dapat diakses untuk analisis lebih lanjut.

3.2.6 Mempelajari Geographical Information Systems melalui training yang disediakan oleh perusahaan.

Sebagai bagian dari pengembangan profesional dan untuk mendukung tugas-tugas yang berhubungan dengan analisis dan visualisasi data geografis, perusahaan memberikan serangkaian pelatihan dalam bidang *Geographical Information Systems* (GIS) melalui platform Udemy. Perusahaan menyediakan tiga kursus utama

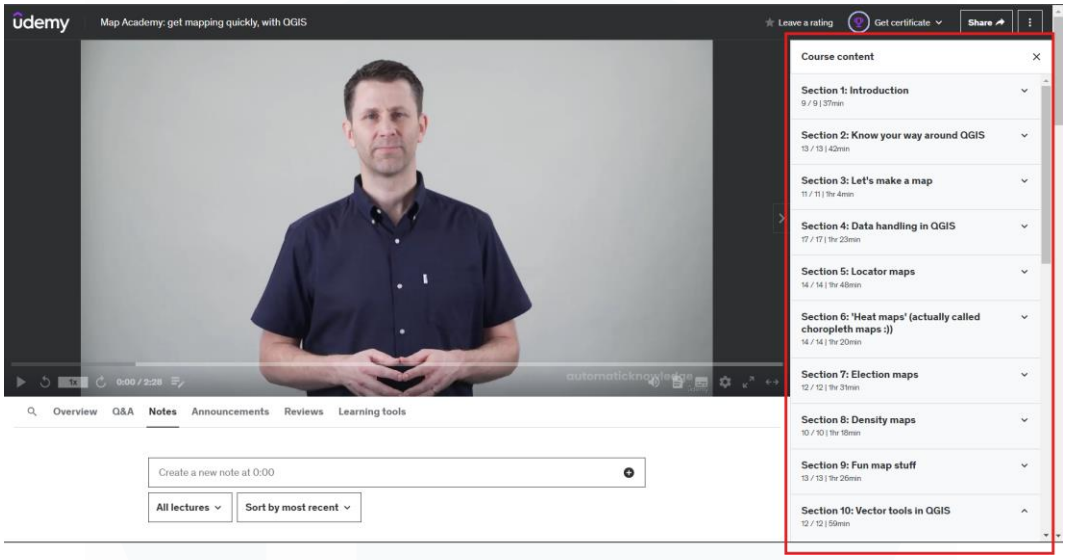
yang sangat komprehensif, yang fokus pada penguasaan QGIS dan Leaflet serta penerapannya dalam proyek GeoDjango.



Gambar 3. 36 Tampilan dari kursus *Map Academy: Get Mapping Quickly, with QGIS*

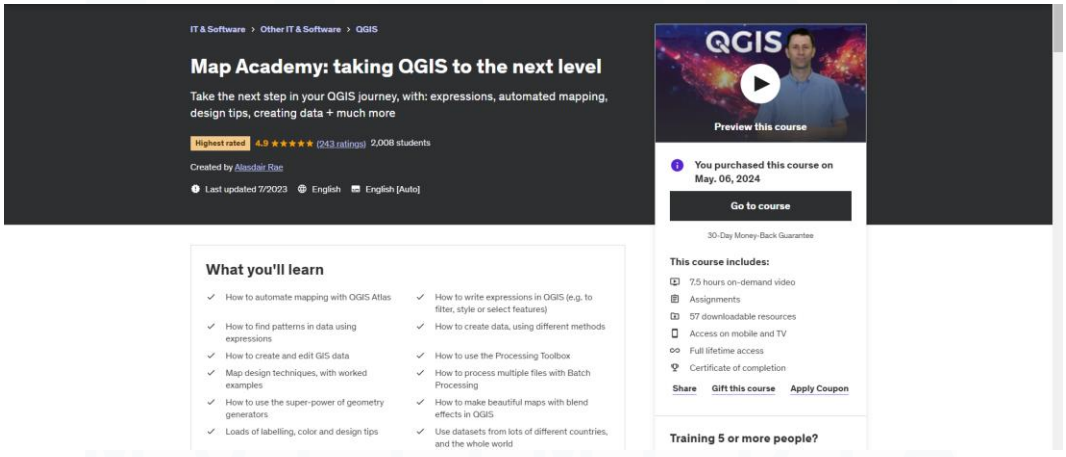
Kursus pertama adalah "*Map Academy: Get Mapping Quickly, with QGIS*". Kursus ini dirancang untuk memberikan pemahaman dasar yang kuat tentang QGIS, sebuah perangkat lunak *open-source* yang sangat populer untuk pemetaan dan analisis geografis. Dalam kursus ini, berbagai konsep dasar yang esensial dalam GIS dipelajari, termasuk cara mengimpor dan mengelola data spasial, serta teknik-teknik dasar dalam pembuatan peta.

Pada awal kursus, antarmuka QGIS dan berbagai alat yang tersedia akan diperkenalkan dan diajarkan cara mengimpor berbagai jenis data spasial, baik itu vektor maupun raster, dan bagaimana cara mengatur data tersebut dalam *layer* yang dapat diolah dan dianalisis lebih lanjut. Selain itu, kursus ini juga mencakup cara mengedit data spasial, seperti menambahkan, menghapus, atau memodifikasi fitur-fitur pada peta.



Gambar 3. 37 Sesi materi pada kursus *Map Academy: Get Mapping Quickly, with QGIS*

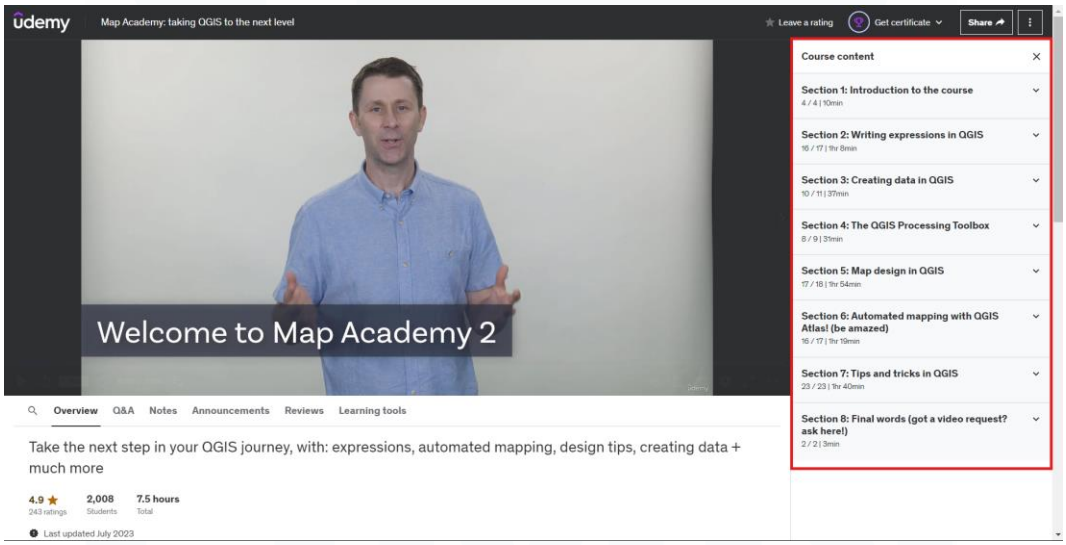
Salah satu bagian yang paling menarik dari kursus ini adalah bagaimana cara membuat peta yang menarik dan informatif. Belajar berbagai teknik visualisasi data, seperti penggunaan simbol, warna, dan label untuk meningkatkan keterbacaan peta. Selain itu, diajarkan juga cara menambahkan elemen-elemen peta seperti legenda, skala, dan judul untuk membuat peta lebih profesional dan mudah dipahami.



Gambar 3. 38 Tampilan dari kursus *Map Academy: Taking QGIS to the Next Level*

Setelah menyelesaikan kursus dasar, dilanjut ke kursus kedua yaitu *"Map Academy: Taking QGIS to the Next Level"*. Kursus ini dirancang untuk memperdalam pengetahuan dan keterampilan dalam menggunakan QGIS, dengan fokus pada fitur-fitur lanjutan dan aplikasi praktis dalam analisis spasial yang lebih kompleks.

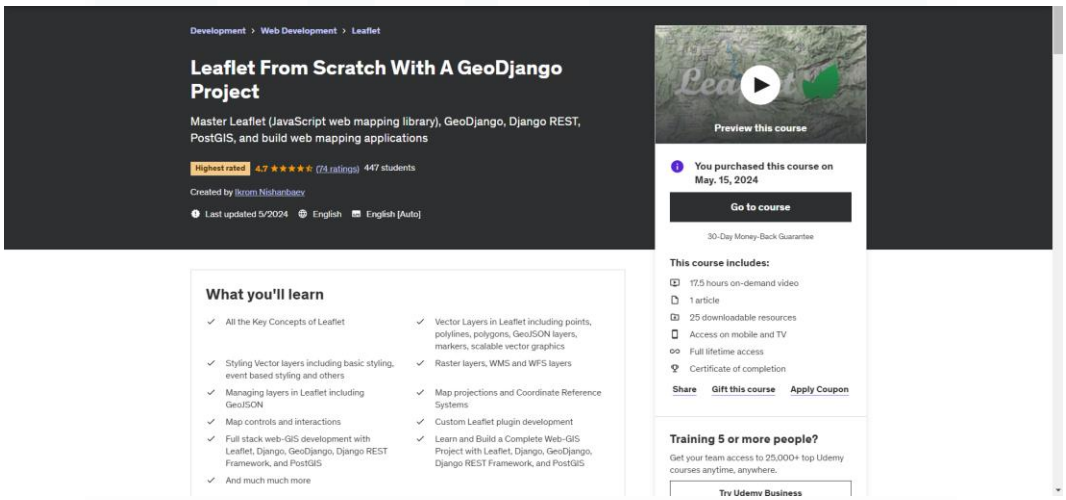
Di kursus ini, berbagai teknik analisis spasial yang lebih canggih akan dipelajari. Salah satu topik utama adalah analisis *overlay*, yaitu bagaimana cara menggabungkan berbagai *layer* data spasial untuk mengidentifikasi hubungan dan pola yang tersembunyi. Misalnya, cara menggunakan alat-alat seperti *intersect*, *union*, dan *difference* untuk menggabungkan atau membandingkan berbagai *layer* data.



Gambar 3. 39 Sesi materi pada kursus *Map Academy: Taking QGIS to the Next Level*

Selain itu, kursus ini juga mencakup penggunaan *plugin* QGIS untuk memperluas fungsionalitas perangkat lunak. Plugin adalah alat tambahan yang dapat diinstal untuk menambah kemampuan QGIS, seperti analisis statistik, visualisasi data 3D, dan integrasi dengan sumber data eksternal. Salah satu plugin yang sangat bermanfaat yang dipelajari adalah "*QGIS Processing Toolbox*", yang menyediakan berbagai alat analisis spasial yang sangat kuat dan fleksibel.

Salah satu proyek praktis yang dikerjakan selama kursus ini adalah pembuatan peta kepadatan populasi berdasarkan data sensus. Teknik interpolasi akan digunakanuntuk menghasilkan peta kepadatan populasi yang menggambarkan distribusi populasi secara lebih detail. Proyek ini memberikan wawasan tentang bagaimana analisis spasial dapat digunakan untuk memahami dinamika populasi dan mendukung pengambilan keputusan dalam perencanaan kota dan kebijakan publik.

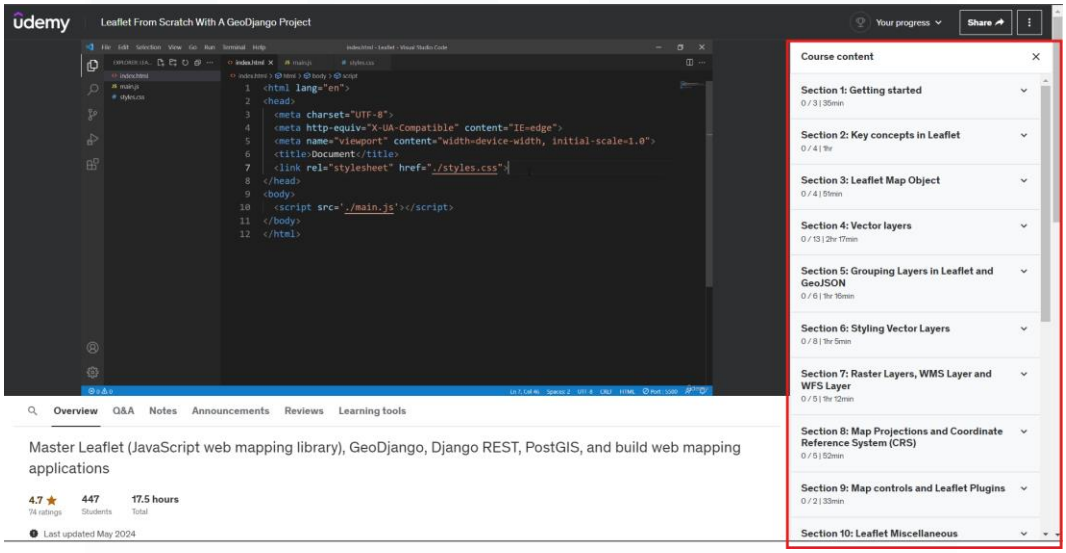


Gambar 3. 40 Tampilan dari kursus *Leaflet From Scratch With A GeoDjango Project*

Kursus terakhir adalah "*Leaflet From Scratch With A GeoDjango Project*". Kursus ini fokus pada Leaflet, sebuah pustaka JavaScript untuk membuat peta interaktif, serta penerapannya dalam proyek GeoDjango. Kursus ini sangat berharga karena menggabungkan pemahaman teknis tentang Leaflet dengan aplikasi praktis dalam pengembangan web menggunakan GeoDjango.

Di awal kursus, dasar-dasar Leaflet akan diperkenalkan, termasuk cara membuat peta interaktif dari awal dengan menambahkan berbagai elemen peta, seperti *marker*, *pop-up*, dan *layer*, yang membuat peta menjadi lebih dinamis dan interaktif. Selain itu, diajarkan juga cara mengintegrasikan data geografis ke dalam peta, sehingga pengguna dapat berinteraksi dengan data tersebut secara langsung.

Salah satu aspek penting yang dibahas dalam kursus ini adalah cara mengintegrasikan Leaflet dengan Django, sebuah framework web berbasis Python. Cara mengatur proyek Django juga diajarkan, membuat model data untuk menyimpan data geografis, dan membangun API untuk mengakses data tersebut dari aplikasi web. Selain itu, juga mempelajari cara menghubungkan Leaflet dengan API GeoDjango, sehingga peta interaktif dapat menampilkan data yang disimpan dalam basis data Django.



Gambar 3. 41 Sesi materi pada kursus *Map Academy: Taking QGIS to the Next Level*

Proyek akhir dari kursus ini adalah membangun sebuah aplikasi web yang menampilkan peta interaktif dengan data real-time. Pembuatan sebuah aplikasi GeoDjango yang menampilkan data cuaca dari berbagai lokasi, dengan peta interaktif yang memungkinkan pengguna untuk melihat informasi cuaca secara detail dengan mengklik marker pada peta. Proyek ini memberikan pemahaman yang mendalam tentang bagaimana menggabungkan GIS dengan pengembangan web untuk menciptakan aplikasi yang informatif dan interaktif.

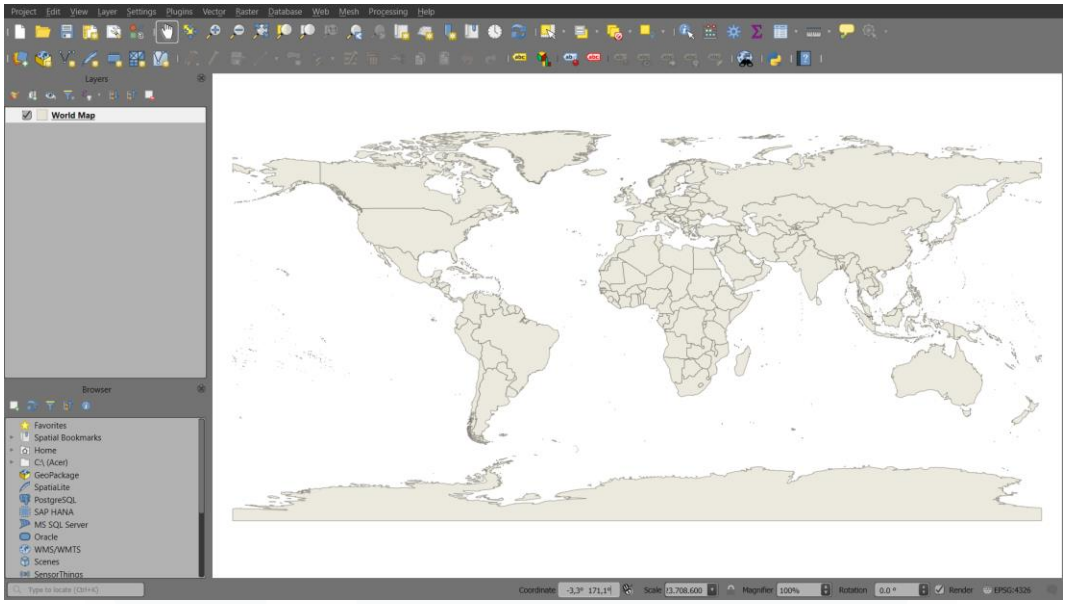
Mengikuti serangkaian kursus ini telah memberikan keterampilan yang sangat berharga dalam bidang Geographical Information Systems (GIS). Penguasaan QGIS dan Leaflet, serta penerapannya dalam proyek

GeoDjango, memungkinkan untuk mengolah, menganalisis, dan memvisualisasikan data geografis dengan cara yang lebih efektif dan inovatif. Pengalaman ini tidak hanya meningkatkan kompetensi teknis, tetapi juga membuka peluang baru dalam pengembangan proyek berbasis GIS di masa depan.

Pelatihan ini juga memberikan wawasan tentang bagaimana GIS dapat diterapkan dalam berbagai bidang, seperti perencanaan kota, analisis lingkungan, dan pengembangan aplikasi web. Dengan keterampilan yang diperoleh untuk merasa lebih siap untuk menghadapi tantangan dalam pekerjaan dan memberikan kontribusi yang lebih besar dalam proyek-proyek yang berhubungan dengan GIS.

3.2.7 Menggunakan QGIS untuk menciptakan beberapa data Geographical.

QGIS (Quantum GIS) adalah sistem informasi geografis (GIS) yang gratis dan sumber terbuka, memungkinkan pengguna untuk membuat, mengedit, memvisualisasikan, menganalisis, dan mempublikasikan informasi geospasial. Antarmuka pengguna QGIS terdiri dari beberapa komponen utama yang memudahkan pengguna dalam mengelola data geospasial.



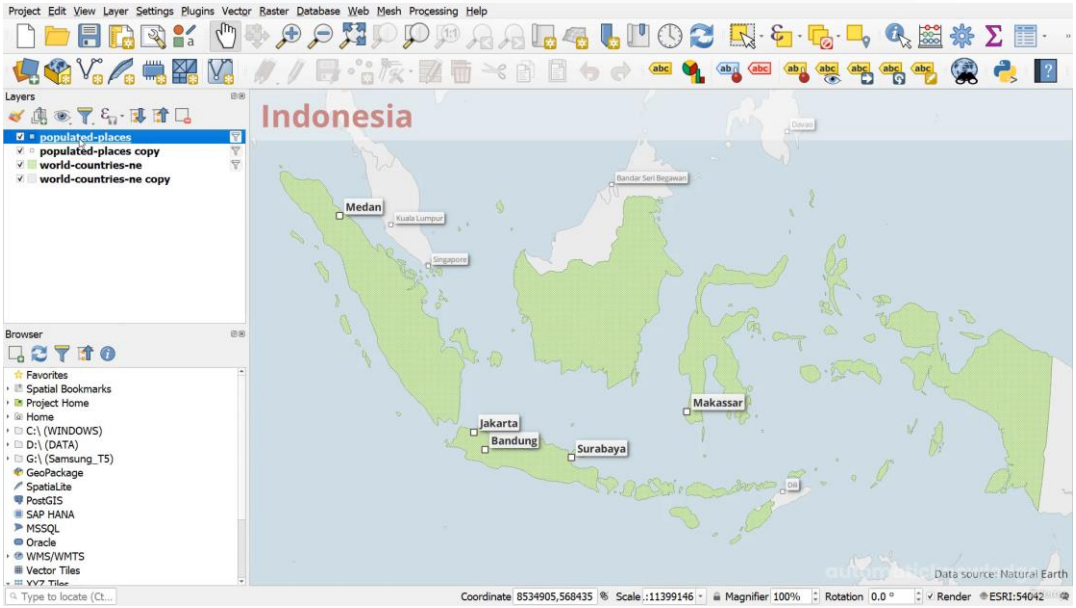
Gambar 3. 42 Tampilan *interface* dari QGIS

Di bagian atas layar terdapat *Bar Menu*, yang menyediakan akses ke berbagai fungsi dan alat QGIS melalui menu seperti *Project*, *Edit*, *View*, *Layer*, *Settings*, *Plugins*, *Vector*, *Raster*, *Database*, *Web*, *Mesh*, *Processing*, dan *Help*. Di bawah *bar menu*, terdapat *Toolbars* yang menawarkan akses cepat ke alat dan fungsi yang sering digunakan, termasuk alat untuk mengelola *layer*, navigasi peta, pengeditan, dan lainnya.

Bagian tengah layar adalah *Map Canvas*, tempat *layer* peta yang dimuat ditampilkan. Pada tangkapan layar ini, terlihat *layer* peta dunia. Di sisi kiri layar, terdapat *Layers Panel* yang menunjukkan daftar semua *layer* yang saat ini dimuat dalam proyek QGIS. Pengguna dapat mengelola visibilitas, urutan, dan properti dari *layer* ini.

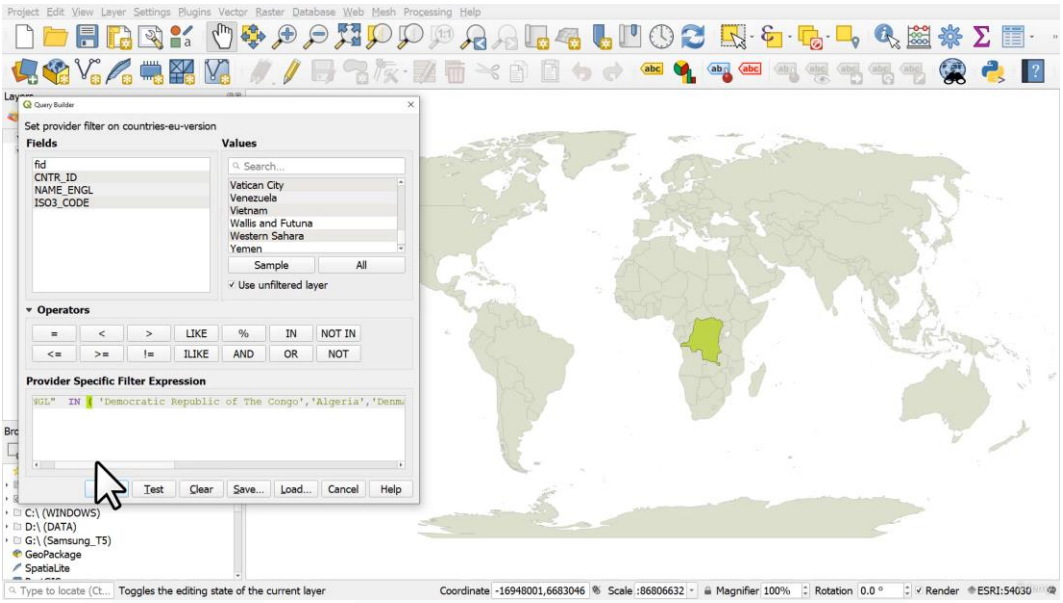
Selain itu, di sisi kiri, di bawah panel *layer*, terdapat *Browser Panel* yang memungkinkan pengguna menjelajahi file sistem, basis data, dan layanan web untuk menambahkan data ke proyek. Di bagian bawah layar, terdapat *Status Bar* yang menampilkan informasi koordinat, skala peta, magnifier, rotasi peta, dan sistem referensi koordinat (CRS) yang digunakan.

Antarmuka QGIS dirancang agar fleksibel dan dapat disesuaikan dengan kebutuhan pengguna, memungkinkan akses cepat ke alat dan fungsi yang paling sering digunakan dalam pengelolaan data geospasial. Dengan antarmuka yang intuitif dan kaya fitur, QGIS memungkinkan pengguna dari berbagai tingkat keahlian untuk bekerja secara efektif dengan data geospasial.



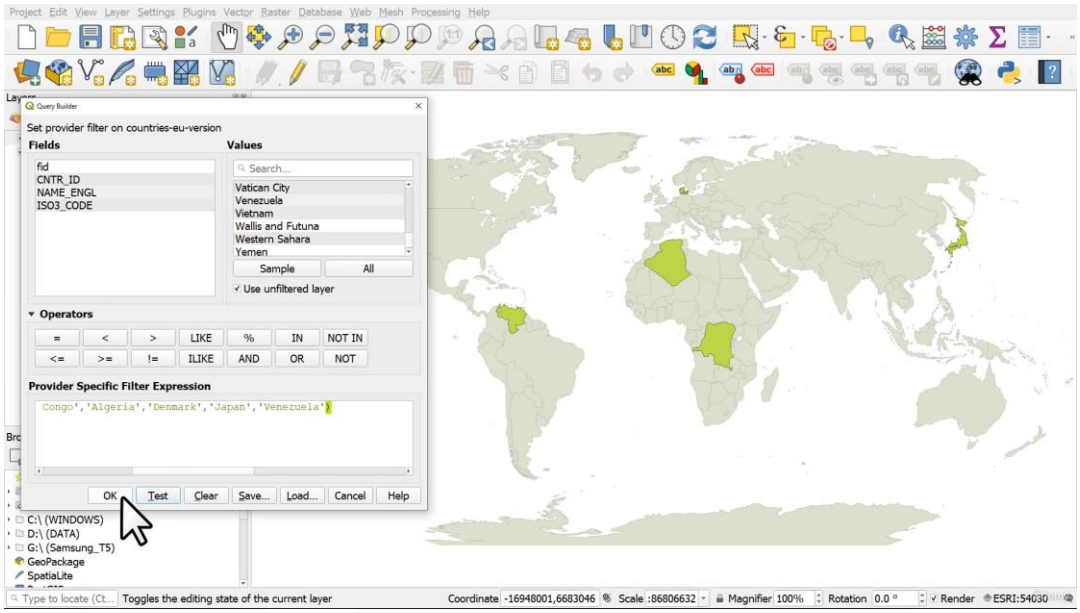
Gambar 3. 43 Tampilan *interface* dari QGIS

Pada gambar 3.45, Terdapat beberapa *layer* yang dimuat dalam proyek ini, yaitu *'populated-places'*, *'populated-places copy'*, *'world-countries-ne'*, dan *'world-countries-ne copy'*. *Layer 'populated-places'* kemungkinan berisi informasi mengenai lokasi-lokasi yang berpenduduk, sementara *'world-countries-ne'* menunjukkan batas-batas negara di seluruh dunia. Pada kanvas peta, wilayah Indonesia ditampilkan dengan beberapa kota besar yang diberi label, seperti Medan, Jakarta, Bandung, Surabaya, dan Makassar. Selain itu, label kota-kota lain di sekitar wilayah Indonesia, seperti Kuala Lumpur dan Singapura, juga ditampilkan, menunjukkan bahwa peta ini mencakup data tambahan dari wilayah sekitarnya.



Gambar 3. 44 Tampilan map sebelum di *filter*

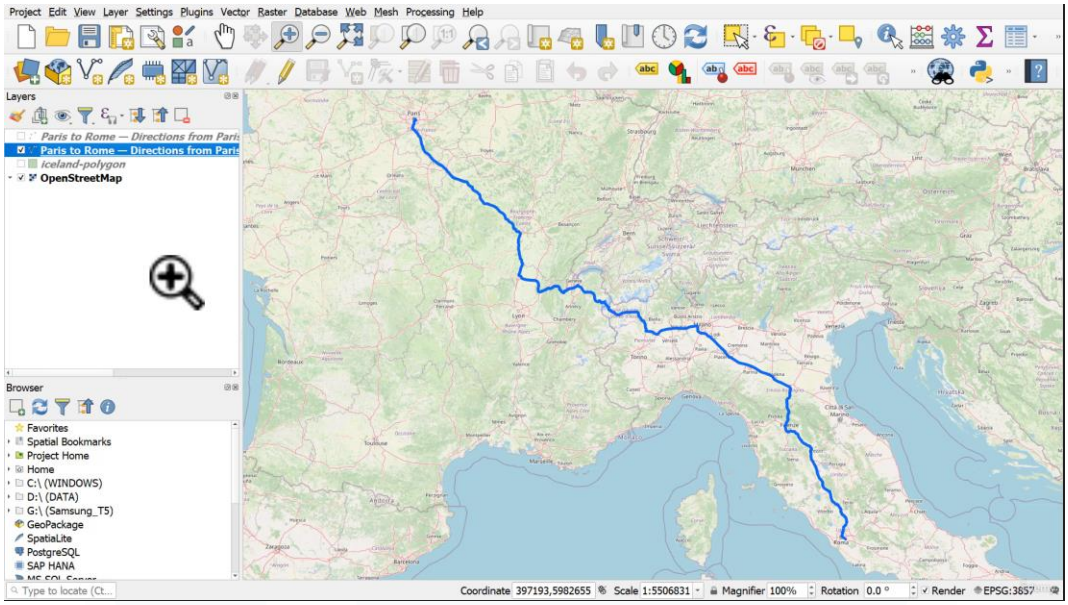
Pada gambar 3.46, jendela *Layer Query Builder* terbuka dengan *layer* countries-eu-version dipilih. Dalam jendela ini, terdapat berbagai field (kolom) yang dapat digunakan untuk menyusun filter, seperti *'CNTR_ID'*, *'NAME_ENGL', dan 'ISO3_CODE'*. Di bagian bawah, kotak ekspresi filter menunjukkan bahwa hanya negara "Democratic Republic of the Congo", "Algeria", dan "Denmark" yang dipilih untuk ditampilkan. Pada peta, hanya wilayah "Democratic Republic of the Congo" yang disorot, menunjukkan bahwa filter awal baru mencakup negara tersebut. Singapura, juga ditampilkan, menunjukkan bahwa peta ini mencakup data tambahan dari wilayah sekitarnya.



Gambar 3. 45 Tampilan map sesudah di filter

Pada gambar 3.47, ekspresi filter diubah untuk memasukkan beberapa negara tambahan: "Congo", "Algeria", "Denmark", "Japan", dan "Venezuela". Ekspresi filter tersebut terlihat di kotak *Provider Specific Filter Expression* sebagai: *"NAME" IN ('Democratic Republic of The Congo', 'Algeria', 'Denmark', 'Japan', 'Venezuela')*

Setelah mengklik tombol *Test* dan OK, peta memperbarui tampilan dengan menyoroti wilayah dari negara-negara yang ditentukan dalam filter baru. Sekarang, beberapa negara tambahan, termasuk Jepang dan Venezuela, ditampilkan dalam peta.



Gambar 3. 46 *Google Maps Route* ke QGIS

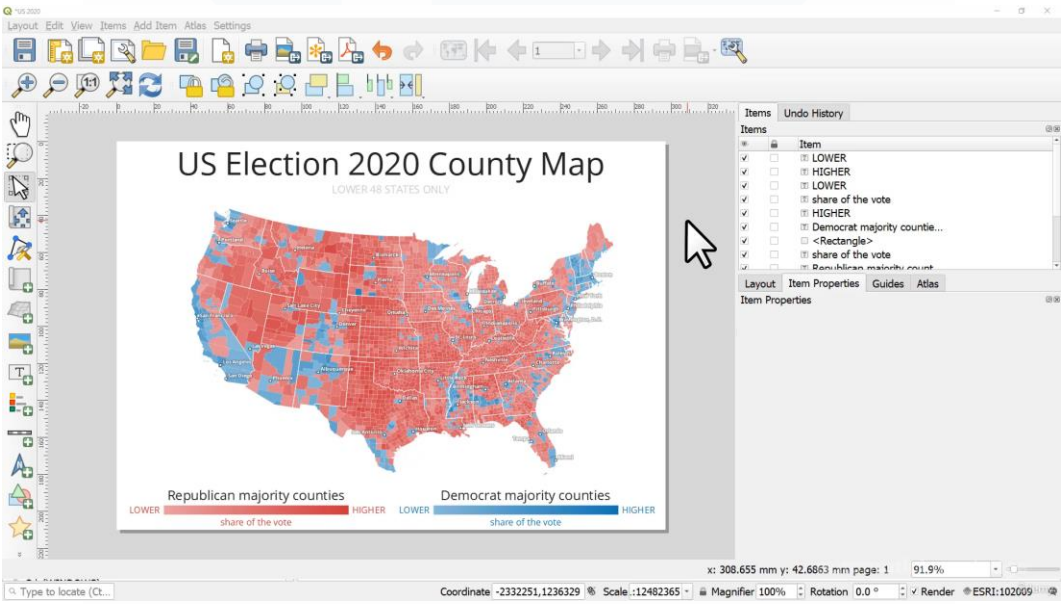
Gambar 3.48 menunjukkan penggunaan QGIS untuk memvisualisasikan rute yang diimpor dari Google Maps. Proyek ini menampilkan rute perjalanan dari Paris ke Roma, digambarkan sebagai garis biru yang membentang dari Paris, melalui beberapa negara Eropa, dan berakhir di Roma.

Pada panel *layer*, terdapat beberapa *layer* data yang dimuat, termasuk Paris to Rome — Directions from Paris, yang berisi data rute perjalanan dari Paris ke Roma, serta *OpenStreetMap* sebagai *layer* peta dasar yang digunakan untuk memberikan konteks geografis pada rute yang diimpor. *Layer* Paris to Rome dipilih dan terlihat pada kanvas peta, menunjukkan rute perjalanan spesifik tersebut.

Kanvas peta menampilkan peta dasar dari *OpenStreetMap* dengan rute perjalanan yang diimpor dari Google Maps, di mana garis biru menunjukkan jalur perjalanan dari Paris ke Roma, melewati beberapa negara seperti Prancis, Swiss, dan Italia. Panel penjelajah menyediakan akses ke berbagai sumber data yang tersimpan di komputer, termasuk drive lokal dan koneksi basis data seperti PostgreSQL dan SAP HANA, memungkinkan penambahan data baru ke proyek QGIS dengan mudah.

Di bagian bawah layar, bar status memberikan informasi penting seperti koordinat kursor saat ini, skala peta, *magnifier*, rotasi peta, dan sistem referensi koordinat (CRS) yang digunakan, yaitu EPSG:3857.

Proses *impor rute* dari Google Maps ke QGIS melibatkan ekspor rute dari Google Maps dalam format yang dapat diterima oleh QGIS, seperti KML atau GPX. File yang diekspor kemudian diimpor ke dalam QGIS dan ditampilkan sebagai *layer* baru. *Layer* peta dasar, dalam hal ini *OpenStreetMap*, digunakan untuk memberikan konteks geografis yang lebih baik pada rute yang diimpor. Dengan menggunakan QGIS, rute perjalanan dari Google Maps dapat divisualisasikan dan dianalisis lebih lanjut dengan alat-alat yang tersedia dalam perangkat lunak GIS ini, memungkinkan pemanfaatan data rute untuk berbagai keperluan seperti perencanaan perjalanan, analisis spasial, dan presentasi peta.



Gambar 3. 47 *Election Map* pada QGIS

Gambar 3.49 tersebut menunjukkan penggunaan QGIS untuk membuat peta hasil Pemilihan Umum Amerika Serikat tahun 2020 berdasarkan county. Peta ini menampilkan distribusi suara dengan warna merah untuk county yang mayoritas memilih kandidat dari Partai Republik dan warna biru untuk county yang mayoritas memilih

kandidat dari Partai Demokrat. Pada bagian atas peta, terdapat judul "*US Election 2020 County Map*," yang menunjukkan fokus peta pada hasil pemilihan di 48 negara bagian utama.

Panel *Items* di sebelah kanan layar menunjukkan elemen-elemen yang disertakan dalam peta, seperti layer yang menampilkan country mayoritas Republik dan Demokrat, serta skala warna yang menggambarkan tingkat persentase suara. *Panel Layout* di sebelah kiri atas layar menunjukkan berbagai alat yang digunakan untuk mengedit dan menyusun elemen-elemen peta, termasuk alat untuk menambahkan teks, legenda, dan skala peta.

Proyek peta ini sangat relevan dengan pekerjaan perusahaan yang bergerak di bidang pemasaran digital dan pengelolaan informasi. Dengan menggunakan QGIS, pembuatan peta interaktif dan informatif dapat meningkatkan kemampuan analisis data geografis dan demografis yang penting untuk merancang strategi pemasaran yang efektif. Kemampuan untuk memvisualisasikan data geografis dengan cara yang informatif dan menarik sangat penting untuk membuat keputusan bisnis yang berdasarkan data.

Kemampuan untuk mengolah dan memvisualisasikan data dengan alat seperti QGIS memberikan keunggulan kompetitif dalam mengembangkan strategi pemasaran yang lebih terarah dan efektif. Dengan demikian, materi mengenai penggunaan QGIS untuk pembuatan peta sangat membantu dalam pekerjaan sehari-hari perusahaan, terutama dalam pengelolaan informasi dan dokumentasi serta dalam merancang dan mengeksekusi strategi pemasaran yang berbasis data.

3.3 Kendala yang Ditemukan

Bagian ini berisi kendala dan kesulitan yang ditemukan selama proses kerja magang.

1. Keterbatasan *device* atau perangkat yang digunakan oleh mahasiswa, perangkat yang digunakan mahasiswa tidak cukup kuat untuk mengelola jumlah data yang masif, sehingga mahasiswa harus menghabiskan waktu yang lebih lama untuk menggunakan data tersebut.
2. Tools yang digunakan cukup banyak, Beberapa tools yang digunakan seperti Studio3T, Visual Studio Code, Looker, Jupyter Notebook, Javascript, MongoDB, dan beberapa tools lainnya. sehingga mahasiswa merasa sulit untuk menggunakan beberapa tools tersebut dan juga beberapa tools tersebut memiliki kegunaan atau fungsi yang berbeda-beda.
3. Pergantian Supervisor pada aktivitas magang, Perusahaan mengganti supervisor yang bertanggung jawab atas mahasiswa yang sedang melakukan magang dikarenakan kontrak yang dimiliki supervisor dengan lama dengan perusahaan sudah mencapai akhir sehingga membuat progress mahasiswa menjadi lebih lambat karena hal tersebut.

3.4 Solusi atas Kendala yang Ditemukan

Bagian ini berisi solusi atas kendala yang ditemukan selama proses kerja magang

1. Perusahaan memberikan fasilitas berupa perangkat yang sudah disediakan kantor yang dapat membantu mahasiswa untuk melanjutkan pekerjaan dengan bantuan fasilitas tersebut.
2. Tools tersebut dipelajari oleh mahasiswa secara berkala yang memungkinkan mahasiswa dapat mempelajari tools tersebut dengan lebih teliti melalui beberapa artikel atau situs serta website yang membahas tools-tools tersebut.

3. Perusahaan dengan jeda waktu yang singkat, berhasil menemukan supervisor pengganti yang akhirnya dapat membantu mahasiswa untuk melakukan aktivitas magang dengan lancar dengan berdiskusi satu sama lain sekaligus melakukan brainstroming bersama.