HOSPITAL DATABASE DESIGN & IMPLEMENTATION

FINAL PROJECT FOR ADVANCED DATABASE MANAGEMENT SYSTEM - GROUP 7

Team Members

Adhithyan Rangarajan

Ananya Srivastava

Geeta Matta

Naveen Ponnaganti

Sai Yarra

Uday Reddy

SUMMARY

Topic Area	Description	Group Members	Weight
Database Design	This part should include a logical database design (for the relational model), using normalization to control redundancy and integrity constraints for data quality.	Adhithyan Rangarajan Ananya Srivastava Sai Yarra Naveen Ponnaganti Geeta Matta Uday Reddy	20%
Query Writing	This part is another chance to write SQL queries, explore transactions, and even do some database programming for stored procedures.	Adhithyan Rangarajan Ananya Srivastava Sai Yarra Naveen Ponnaganti	30%
Performance Tuning	In this section, you can capitalize and extend your prior experiments with indexing, optimizer modes, partitioning, parallel execution and any other techniques you want to further explore.	Adhithyan Rangarajan Ananya Srivastava Sai Yarra Naveen Ponnaganti	20%
DBA Scripts & Data Visualization	Here you are free to explore any other topics of interest. Suggestions include DBA scripts, database security, interface design, data visualization, data mining, and NoSQL databases.	Adhithyan Rangarajan Ananya Srivastava Sai Yarra Naveen Ponnaganti	30%

PURPOSE

The purpose of this document is to explain in detail the design process involved in creating and designing a database for the hospital management system. This document describes all the attributes and entities involved in the database. The end goal is to design a database for a hospital in such a way that there is seamless access to any required form of data for a particular patient/disease/treatment or any business use case no matter how complex in order to ensure fastest possible data retrieval & instant insights from the available data by leveraging preloaded intelligent analytics & data visualization.

NARRATIVE

A hospital management system, in general, is extremely complicated and sensitive, most common issues faced include -not have the required masking or access restrictions to prevent employees of one department from accessing records/ reports of another department, non-integrated data systems between Inpatient history, external medication received, next plan of action, digital record storage and access, pharmacy data, finance & billings, emergency records, pathology lab reports etc. We aim to design a database which provides seamless integration and at the same time has all the required access restrictions to ensure smoothest data transfer, loading, backup and retrieval. We plan to add up more business intelligent decision-making powerful visualizations by using existing data which can help the doctors and all medical teams & departments involved. We will perform query writing and perform analytics to filter out results and draw conclusions. Later, we will do performance tuning to optimize our queries so that requests can be completed efficiently.

We researched multiple online sources as well as healthcare professionals and doctors who have been in the profession for 20+ years as well as experts to understand their deepest issues from a data retrieval standpoint as an end user whilst working for a hospital/healthcare service industry.

Based on the information that we collected, we built our own dataset on the most important variables to solve the problems based on the inputs received. We also spoke to USF folks who have interned at the Tampa General Hospital as well as other healthcare companies to understand more about what's being done with data and where the real technical challenges are. Based on the inputs received, we have tried to come up with queries which provide some insights into business use cases such as finding patients who have the same blood group in a zip code space, finding total **inventory cost, available number of active beds, emergency services offering hospitals in geographical zone etc. –** patient side business emergency use cases have also been considered to come up with the queries. We came up with below Entity Relationship Diagram in the next section after entities and attributes, which depicts attributes of entities and relations between different entities, as part of designing the project. Entities will be considered as tables, attributes as columns and tuples as rows during implementation of project.

ENTITIES IDENTIFIED TO BE TRACKED

- HOSPITAL_DATA
- DOCTOR_DATA
- PATIENT_DATA
- BILLS_FINANCE
- HOSPITALINVENTORY_LINK
- INSURANCE
- LAB_REPORTS
- PHARMACY_DATA
- INPATIENT_VISIT_HISTORY
- NURSE_DATA
- HOSP_PATIENT_LINK
- HOSPITAL_INVENTORY
- HOSPITAL_PHARMACY_LINK

ENTITIES WITH ATTRIBUTES NESTED

HOSPITAL_DATA
- HOSP_ID
- CAPACITY
- HOSP_NAME
- HOSP_ADDRESS
- CITY
- STATE
- COUNTY
- ZIP_CODE
- BEDS_AVAILABLE
- DOCTORS_COUNT
- HOSPITAL_TYPE

DOCTOR_DATA
- DOCTOR_ID
- HOSP_ID
- DOC_NAME
- DOC_PHONE
- DOC_EMAIL_ID
- DOC_ADDRESS

PATIENT_DATA
- PATIENT_ID

- NURSE_ID
- WEIGHT
- BMI
- MAIN_DISEASE
- HEIGHT
- PHONE
- EMAIL_ID
- ADDRESS
- BLOOD_GROUP
- SEX
- AGE
- ADMIT_ID
- DEFICIENCY

BILLS_FINANCE
- BILLS_ID
- CASH
- CHEQUE
- AMOUNT_TOTAL
- AMOUNT_DUE
- AMOUNT_PAID
- DUE_DATE
- PATIENT_ID
- HOSP_ID

HOSPITALINVENTORY_LINK
- HIL_ID
- HOSP_ID
- INV_ID
- INV_COUNT

INSURANCE
- INSURANCE_ID
- HOSPITAL_INSURANCE_TIE_UP
- CORPORATE_INSURANCE
- PERSONAL_INSURANCE
- PATIENT_ID

LAB_REPORTS
- BRANCH_ID
- BRANCH_NAME
- PATIENT_ID
- ACCESS_SENSITIVITY
- DOWNLOAD_METRICS
- REPORT_SUMMARY

PHARMACY_DATA

- MEDICINE_ID
- PAYMENT_METHODS
- PURCHASE_AMOUNT
- MEDICINE_NAME
- DOSAGE

INPATIENT_VISIT_HISTORY

- DATE
- HANDWRITTEN_TEXTNOTES
- PRESCRIPTION_HISTORY
- MEDICATION_PRESCRIBED
- NEXT_CHECKUPDATE
- IP_NOTES
- INPATIENT_ID
- PATIENT_ID
- DOCTOR_ID

NURSE_DATA

- NURSE_ID
- PATIENT_ID
- NURSE_NAME

HOSP_PATIENT_LINK

- LINK_ID
- PATIENT_ID
- HOSP_ID

HOSPITAL_PHARMACY_LINK

- HOSP_ID
- MEDICINE_ID
- MEDICINE_COUNT

HOSPITAL_ INVENTORY

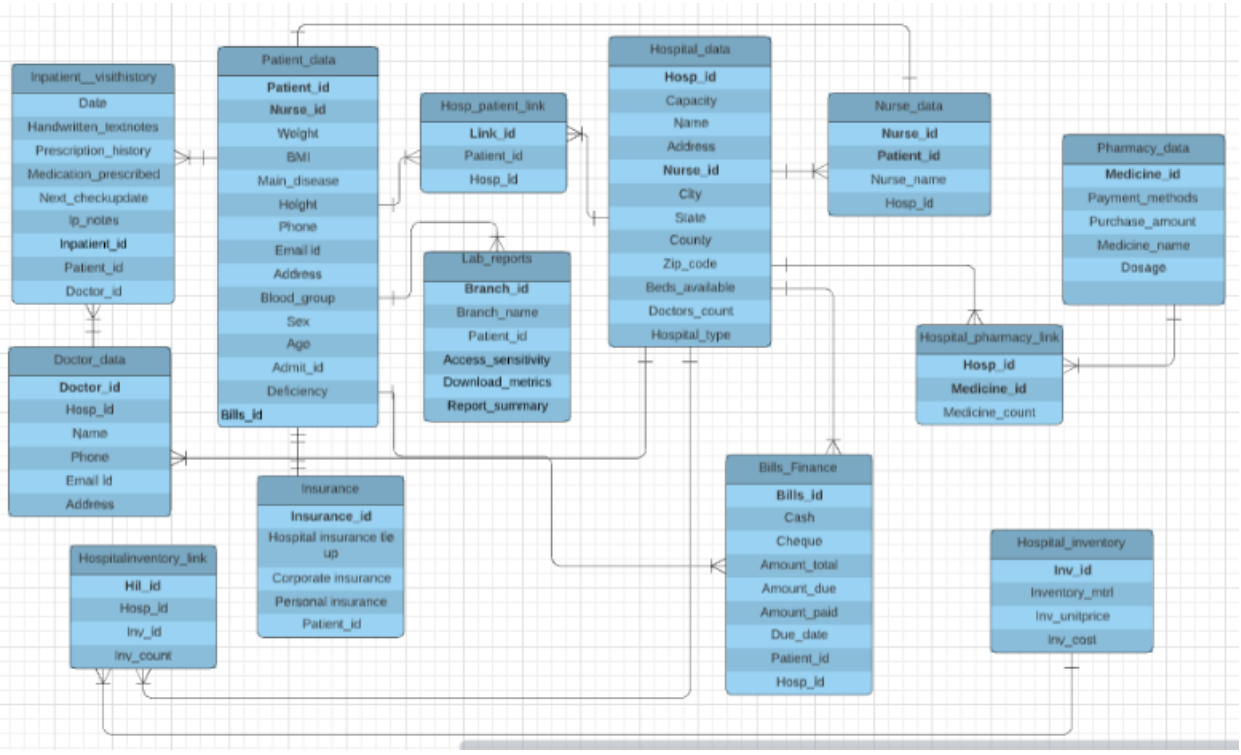
- INV_ID
- INVENTORY_MTRL
- INV_UNITPRICE
- INV_COST

BUSINESS RULES

- A hospital can have many doctors, nurses and patients.
- A patient can have multiple lab reports, bills and visit history.
- A patient can be assigned to only one nurse but one nurse can treat multiple patients.
- Each hospital will generate many bills based on the patient ID.
- Each hospital will have multiple medicines in the pharmacy and same medicine can be found in various hospitals.
- Each hospital will have multiple inventory materials in the inventory and same inventory material can be found in various hospitals.
- Each patient can have only one insurance associated with him identified by insurance ID.

ENTITY RELATIONSHIP DIAGRAM REPRESENTING DATABASE DESIGN

Proposed database design



Implemented design in Oracle SQL-

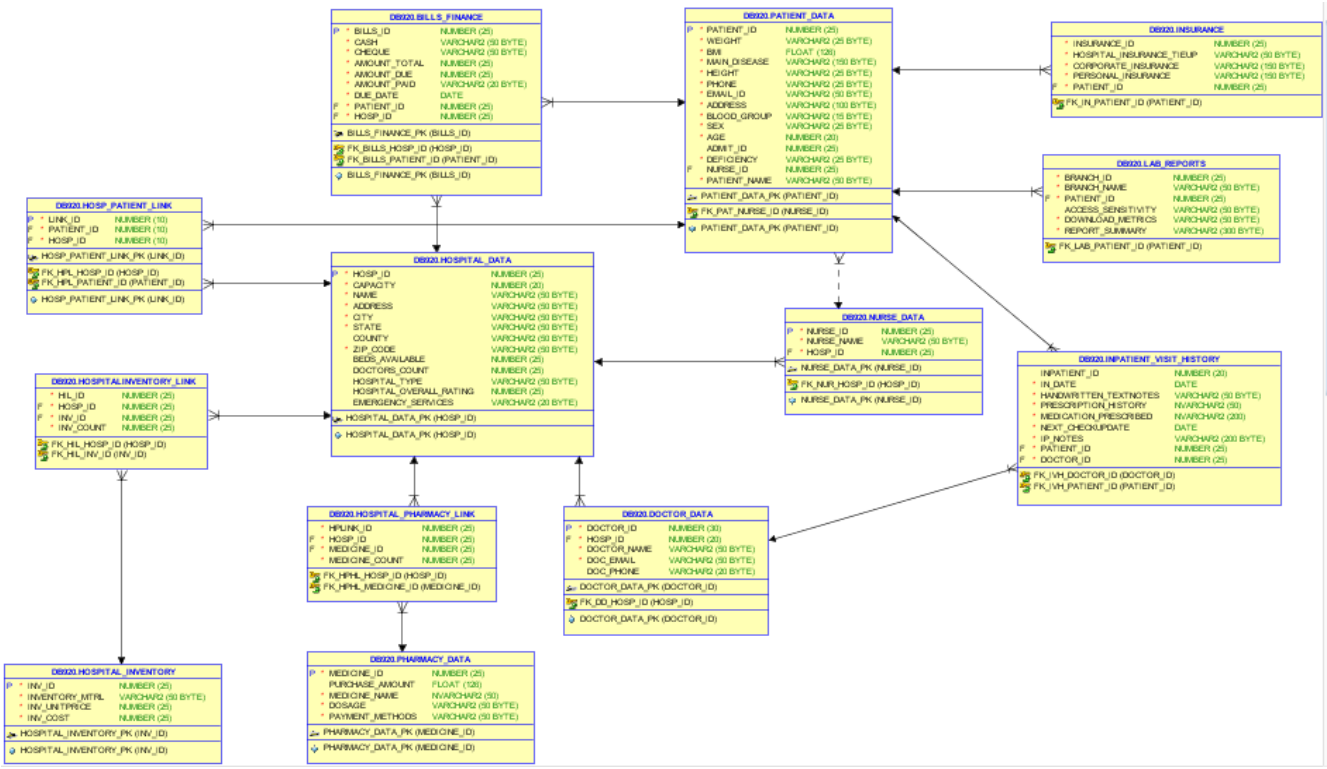
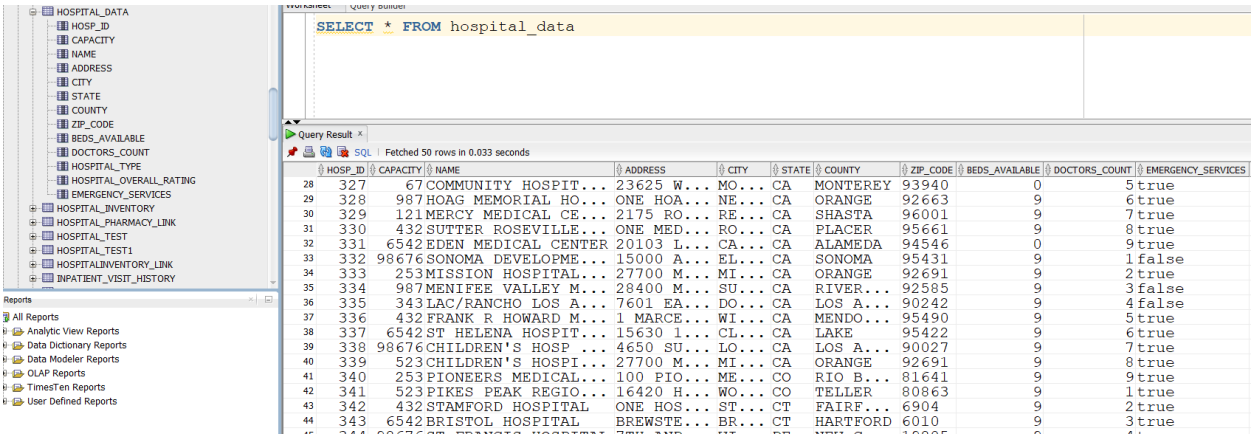


TABLE VIEWS

Hospital_data: This table collects all details about each hospital such as hospital ID(primary key),name, capacity,beds available.



	HOSP_ID	CAPACITY	NAME	ADDRESS	CITY	STATE	COUNTY	ZIP_CODE	BEDS_AVAILABLE	DOCTORS_COUNT	EMERGENCY_SERVICES
28	327	67	COMMUNITY HOSPIT...	23625 W...	MO...	CA	MONTEREY	93940	0	5	true
29	328	987	HOAG MEMORIAL HO...	ONE HOA...	NE...	CA	ORANGE	92663	9	6	true
30	329	121	MERCY MEDICAL CE...	2175 RO...	RE...	CA	SHASTA	96001	9	7	true
31	330	432	SUTTER ROSEVILLE...	ONE MED...	RO...	CA	PLACER	95661	9	8	true
32	331	6542	EDEN MEDICAL CENTER	20103 L...	CA...	CA	ALAMEDA	94546	0	9	true
33	332	98676	SONOMA DEVELOPME...	15000 A...	EL...	CA	SONOMA	95431	9	1	false
34	333	253	MISSION HOSPITAL...	27700 M...	MI...	CA	ORANGE	92691	9	2	true
35	334	987	MENIFEE VALLEY M...	28400 M...	SU...	CA	RIVER...	92585	9	3	false
36	335	343	LAC/RANCHO LOS A...	7601 EA...	DO...	CA	LOS A...	90242	9	4	false
37	336	432	FRANK R HOWARD M...	1 MARCE...	WI...	CA	MENDO...	95490	9	5	true
38	337	6542	ST HELENA HOSPIT...	15630 1...	CL...	CA	LAKE	95422	9	6	true
39	338	98676	CHILDREN'S HOSP ...	4650 SU...	LO...	CA	LOS A...	90027	9	7	true
40	339	523	CHILDREN'S HOSPI...	27700 M...	MI...	CA	ORANGE	92691	9	8	true
41	340	253	PIONEERS MEDICAL...	100 PIO...	ME...	CO	RIO B...	81641	9	9	true
42	341	523	PIKES PEAK REGIO...	16420 H...	WO...	CO	TELLER	80863	9	1	true
43	342	432	STAMFORD HOSPITAL	ONE HOS...	ST...	CT	FAIRF...	6904	9	2	true
44	343	6542	BRISTOL HOSPITAL	BREWSTE...	BR...	CT	HARTFORD	6010	9	3	true

Inpatient__visithistory: This table contains the details of patient visiting the hospital such as Patient ID (Primary Key), In_Date, Medicine Prescribed, In patient Notes.

```
SELECT * FROM inpatient_visit_history
```

Query Result | SQL | Fetched 50 rows in 0.034 seconds

	INPATIENT_ID	IN_DATE	HANDWRITTEN_TEXTNOTES	PRESCRIPTION_HISTORY	MEDICATION_PRESCRIBED	NEXT_CHECKUPDATE	IP_NOTES	PATIENT_ID	DOCTOR_ID
1	1044	13-FEB-10	Critical. Suggested to better...	Humira	Clarithrom...	01-AUG-23	Cystic ...	23642	57240
2	1045	14-FEB-10	Next Visit in one week	Augmentin	Clindamycin	01-SEP-23	Loose b...	23641	57250
3	1046	15-FEB-10	Further Medication required	Benadryl	Clobazam	01-OCT-23	Other s...	23640	57260
4	1047	16-FEB-10	Patient is doing well. Rest f...	Hydroxychlo...	Clofarabine	01-NOV-23	Strain ...	23639	57270
5	1048	17-FEB-10	Critical. Suggested to better...	Methotrexate	Codeine	01-DEC-23	Unspeci...	23638	57280
6	1049	18-FEB-10	Next Visit in one week	Tylenol	Crizanlizumab	01-JAN-24	Angiody...	23637	57290
7	1050	19-FEB-10	Further Medication required	Viagra	Crizotinib	01-FEB-24	Anterio...	23636	57300
8	1051	20-FEB-10	Next Visit in one week	Onpattro	Cyclobenza...	01-MAR-24	Lacerat...	23635	57310
9	1052	21-FEB-10	Further Medication required	ibuprofen	Cyclophosp...	01-APR-24	Other s...	23634	57320
10	1053	22-FEB-10	Patient is doing well. Rest f...	Brilinta	Cyclosporine	01-MAY-24	Salter ...	23633	57330
11	1054	23-FEB-10	Critical. Suggested to better...	Cialis	Cyprohepta...	01-JUN-24	Other o...	23632	57340

Lab_reports: This table contains the details of lab reports of the patient visiting the hospital. It has attributes like Branch_id (Primary Key), Branch Name, Patient_ID.

```
SELECT * FROM lab_reports
```

Query Result | SQL | Fetched 50 rows in 0.04 seconds

	BRANCH_ID	BRANCH_NAME	PATIENT_ID	ACCESS_SENSITIVITY	DOWNLOAD_METRICS	REPORT_SUMMARY
1	26	Infectious Diseases	23440	NO	243	Age: 68yrsSex: MaleCC: Self HarmHR: 56 regularBP:
2	12	General Surgery	23439	YES	244	Age: 83yrsSex: FemaleCC: Chest PainHR: 143 regular
3	23	Endocrinology	23438	YES	245	Age: 45yrsSex: FemaleCC: Difficulty WalkingHR: 12
4	10	Emergency Medicine	23437	YES	246	Age: 19yrsSex: MaleCC: Chest PainHR: 46 regularBP
5	8	Ear Nose Throat Hea...	23436	YES	247	Age: 45yrsSex: FemaleCC: Blanching RashHR: 50 irr
6	26	Dermatology And Cos...	23435	YES	248	Age: 46yrsSex: MaleCC: Difficulty WalkingHR: 46 r
7	6	Critical Care Medicine	23434	YES	249	Age: 19yrsSex: FemaleCC: Self HarmHR: 104 regular
8	29	Corneal Transplant	23433	NO	250	Age: 16yrsSex: FemaleCC: Abdominal PainHR: 56 req
9	18	Clinical Haematolog...	23432	NO	251	Age: 16yrsSex: MaleCC: Self HarmHR: 123 regularBP:

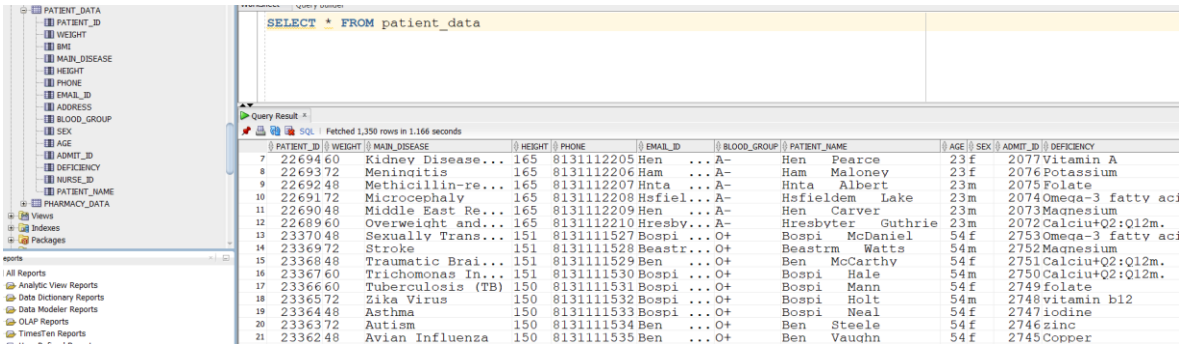
Doctor_data: This table contains the details of doctors such as Doctor_ID, Name, Hospital_ID, mobile number and Address.

```
SELECT * FROM doctor_data
```

Query Result | SQL | Fetched 850 rows in 0.52 seconds

	DOCTOR_ID	HOSP_ID	DOCTOR_NAME	DOC_PHONE	DOC_EMAIL
223	59030	522	Monroe Mynatt	9876543432	Sop...
224	59040	523	Coreen Sutherlin	9876543433	Sop...
225	59050	524	Deana Speed	9876543434	Sop...
226	59060	525	Booker Lemen	9876543435	Sop...
227	59070	526	Tomeka Flom	9876543436	Sop...
228	59080	527	Carletta Hurdle	9876543437	Sop...
229	59090	528	Awilda Guynes	9876543438	Sof...
230	59100	529	Jamee Wootton	9876543439	Sof...
231	59110	530	Violette Moudy	9876543440	Sof...
232	59120	531	Aleen Kinnear	9876543441	Sof...
233	59130	532	Rosaria Mathes	9876543442	Sky...
234	59140	533	Irene Copen	9876543443	Sky...
235	59150	534	Tracey Weldon	9876543444	Sky...
236	59160	535	Dayle Chenard	9876543445	Sky...
237	59170	536	Dara Turlington	9876543446	Sim...
238	59180	537	Lieselotte Leask	9876543447	Sil...
239	59190	538	Katina Brumett	9876543448	Sie...
240	59200	539	Elizabeth Correa	9876543449	Sie...
241	59210	540	Apryl Rosenthal	9876543450	Sie...
242	59220	541	Daniela Apolinar	9876543451	Sie...
243	59230	542	Chastity Lentz	9876543452	Sid...
244	59240	543	Emelina Liddell	9876543453	Sid...
245	59250	544	Sana Urbano	9876543454	Sid...
246	59260	545	Elenora Sprankle	9876543455	Sid...

Patient_data: This table collects all details about each patient such as patientID (primary key),weight,height,BMI, disease etc.



```
SELECT * FROM patient_data
```

PATIENT_ID	WEIGHT	MAIN_DISEASE	HEIGHT	PHONE	EMAIL_ID	...	BLOOD_GROUP	PATIENT_NAME	AGE	SEX	ADMIT_ID	DEFICIENCY
22694	60	Kidney Disease...	165	8131112205	Hen	...	A-	Hen Pearce	23	f	2077	Vitamin A
22693	72	Meningitis	165	8131112206	Ham	...	A-	Ham Maloney	23	f	2076	Potassium
22692	48	Methicillin-re...	165	8131112207	Hnta	...	A-	Hnta Albert	23	m	2075	Folate
22691	72	Microcephaly	165	8131112208	Hsfiel	...	A-	Hsfieldem Lake	23	m	2074	Omega-3 fatty aci
22690	48	Middle East Re...	165	8131112209	Hen	...	A-	Hen Carver	23	m	2073	Magnesium
22689	60	Overweight and...	165	8131112210	Hresby	...	A-	Hresbyter Guthrie	23	m	2072	Calciu+Q2:Q12m.
23370	48	Sexually Trans...	151	8131111527	Bospi	...	O+	Bospi McDaniel	54	f	2753	Omega-3 fatty aci
23369	72	Stroke	151	8131111528	Beastr	...	O+	Beastrm Watts	54	m	2752	Magnesium
23368	48	Traumatic Brai...	151	8131111529	Ben	...	O+	Ben McCarthy	54	f	2751	Calciu+Q2:Q12m.
23367	60	Trichomonas In...	151	8131111530	Bospi	...	O+	Bospi Hale	54	m	2750	Calciu+Q2:Q12m.
23366	60	Tuberculosis (TB)	150	8131111531	Bospi	...	O+	Bospi Mann	54	f	2749	folate
23365	72	Zika Virus	150	8131111532	Bospi	...	O+	Bospi Holt	54	m	2748	vitamin b12
23364	48	Asthma	150	8131111533	Bospi	...	O+	Bospi Neal	54	f	2747	iodine
23363	72	Autism	150	8131111534	Ben	...	O+	Ben Steele	54	f	2746	zinc
23362	48	Avian Influenza	150	8131111535	Ben	...	O+	Ben Vaughn	54	f	2745	Copper

Hosp_patient_link: This table is the intermediate table that links Hospital_data table and Patient_data table based on link_id,hospitalID and patientID.

```
SELECT * FROM hosp_patient_link
```

Fetched 50 rows in 0.038 seconds

LINK_ID	PATIENT_ID	HOSP_ID
1	23688	300
2	23684	301
3	23683	302
4	23682	303
5	23681	304
6	23680	305
7	23679	306
8	23678	307
9	23677	308
10	23676	309
11	23675	310
12	23674	311
13	23673	312

Nurse_data: This table collects details about each nurse such as nurseID (primary key), nurse name.

```
SELECT * FROM NURSE_DATA
```

Fetched 50 rows in 0.045 seconds

NURSE_ID	NURSE_NAME	HOSP_ID
100	Katelyn	300
101	Janiya	301
102	Liana	302
103	Brylee	303
104	Amira	304
105	Mareli	305
106	Paulina	306
107	Tatum	307
108	Addyson	308
109	Lizbeth	309
110	Katherine	310
111	Kassandra	311
112	Daisy	312

Pharmacy_data: This tables collects details about the medicines such as medicineID(primary key), medicine name, dosage.

```
SELECT * FROM PHARMACY_DATA
```

Query Result

SQL | Fetched 50 rows in 0.037 seconds

	MEDICINE_ID	PURCHASE_AMOUNT	MEDICINE_NAME	DOSAGE	PAYMENT_METHODS
1	11	110	Bacterimuran	100mg	Credit/debit
2	12	210	Bexlestid	75mg	cash
3	13	329	Deslotamine	50mg	cash
4	14	235	Cabosteride Dextropex	100mg	cash
5	15	45	Levominphen Ademfine	650mg	cash
6	16	643	Librabucil Tesphine	600mg	cash
7	17	63	Atracunonide Ablaprodol	100mg	cash
8	18	653	Fosinopur	75mg	cash
9	19	64	Sucderal	50mg	cash
10	20	7543	Novobital	100mg	cash
11	21	346	Ceftasoline	650mg	cash
12	22	6573	Ethatensin	600mg	cash
13	23	432	Albutensin	100mg	cash
14	24	4743	Claprox Clinotecan	75mg	cash

Hospital_pharmacy_link: This table is the intermediate table that links Hospital_data table and pharmacy_data table based on HospitalID and pharmacyID.

```
SELECT * FROM hospital_pharmacy_link
```

Query Result

SQL | Fetched 50 rows in 0.03 seconds

	HPLINK_ID	HOSP_ID	MEDICINE_ID	MEDICINE_COUNT
1	1001	300	11	36
2	1002	301	12	29
3	1003	302	13	63
4	1004	303	14	49
5	1005	304	15	27
6	1006	305	16	50
7	1007	306	17	33
8	1008	307	18	85
9	1009	308	19	35
10	1010	309	20	20
11	1011	310	21	83

Hospital_inventory: This table contains the inventory data such as inventoryID (primary key), inventory material, inventory unit price, inventory cost.

```
SELECT * FROM hospital_inventory
```

Query Result

SQL | Fetched 50 rows in 0.046 seconds

	INV_ID	INVENTORY_MTRL	INV_UNITPRICE	INV_COST
1	1	GLUCOSE	2	2
2	2	VIALS	4	4
3	3	FIRSTAID KITS	7	7
4	4	PACKETS	1	1
5	5	Stretchers	9	9
6	6	Monitors	6	6
7	7	Sterilizers	2	2
8	8	Vaccines	9	9
9	9	Dispensers	9	9
10	10	Anesthesia	1	1
11	11	Defibrillators	8	8
12	12	ECGMachines	1	1
13	13	Surgical lights	7	7

Hospitalinventory_link: This table is the intermediate table that links Hospital_data and Inventory_data based on HillD(primary key),Hosp_id and Inv_id.

```
SELECT * FROM hospitalinventory_link
```

Query Result

SQL | Fetched 50 rows in 0.029 seconds

	HIL_ID	HOSP_ID	INV_ID	INV_COUNT
1	1	300	1	35
2	2	301	2	19
3	3	302	3	23
4	4	303	4	40
5	5	304	5	38
6	6	305	6	27
7	7	306	7	26
8	8	307	8	39
9	9	308	9	20
10	10	309	10	29
11	11	310	11	12
12	12	311	12	17
13	13	312	13	37

Insurance: This table collects the details regarding insurance of patients such as insuranceID(primary key),Corporate Insurance, Personal Insurance.

```
SELECT * FROM insurance
```

Query Result

SQL | Fetched 50 rows in 0.029 seconds

	INSURANCE_ID	HOSPITAL_IN...	CORPORATE_INSURANCE	PERSONAL_INSURANCE	PATIENT_ID
1	120	Yes	Blue Cross Blue...	RLI Corp.	23566
2	121	Yes	Wellmark Inc.	Safe Auto Insurance C...	23565
3	122	Yes	Blue Cross Blue...	Safeco	23564
4	123	Yes	Neighborhood He...	Safeway Insurance Group	23563
5	124	No	Independence Bl...	Securian Financial Group	23562
6	125	No	Providence Health	Selective Insurance	23561
7	126	Yes	Health Care Ser...	Sentry Insurance	23560
8	127	Yes	Medical Mutual ...	Shelter Insurance	23559
9	128	Yes	Noridian Mutual...	Society Insurance	23558
10	129	No	Blue Cross Blue...	Southern Aid and Insu...	23557
11	130	No	Mulberry Health	SquareTrade	23556
12	131	Yes	Molina Health C...	Standard Insurance Co...	23555
13	132	Yes	Blue Cross Blue...	State Auto Insurance ...	23554
14	133	Yes	Anthem Inc.	State Farm Insurance	23553

Bills_Finance: This table collects all details regarding the bills and payment history of patients such as BillsID (primary key),amount_total,amount_due, amount_paid.

```
SELECT * FROM bills_finance
```

Query Result

SQL | Fetched 50 rows in 0.044 seconds

	BILLS_ID	CASH	CHEQUE	AMOUNT_TOTAL	AMOUNT_DUE	AMOUNT_PAID	DUE_DATE	PATIENT_ID	HOSP_ID
1	1234	Yes	Yes	5000	1703	Yes	03-FEB-21	23688	300
2	1235	No	No	5000	1541	Yes	03-FEB-21	23684	301
3	1236	Yes	Yes	5000	1758	Yes	03-FEB-21	23683	302
4	1237	No	No	5000	1711	Yes	03-FEB-21	23682	303
5	1238	Yes	Yes	5000	1706	Yes	03-FEB-21	23681	304
6	1239	No	No	5000	1736	Yes	03-FEB-21	23680	305
7	1240	Yes	Yes	6500	1562	Yes	03-FEB-21	23679	306
8	1241	No	No	5000	1534	Yes	03-FEB-21	23678	307
9	1242	Yes	Yes	6500	1716	Yes	03-FEB-21	23677	308
10	1243	No	No	5000	1684	Yes	03-FEB-21	23676	309
11	1244	Yes	Yes	6500	1638	Yes	03-FEB-21	23675	310
12	1245	No	No	5000	1718	Yes	03-FEB-21	23674	311
13	1246	Yes	Yes	6500	1616	Yes	03-FEB-21	23673	312

DATA SYNTHESIS

The data for the project has been synthesized ma l n l y u s l ng Microsoft Excel. Some of the prominent functions that were used in Excel include,

- VLOOKUP
- INDEX
- RAND and
- RANDBETWEEN

The tabulation below provides a summary of the data housed in the tables,

Table Name	Columns	Number of constraints	Number of Records
DOCTOR_DATA	5	2	2000
HOSPITAL_DATA	4	2	2000
PATIENT_DATA	4	3	2000
BILLS_FINANCE	4	4	2000
PHARMACY_DATA	3	3	2000
HOSPITAL_INVENTORY	2	3	2000

DATA INTEGRITY

Data Integrity refers to the consistency and maintenance of the data through the life cycle of the database. In a database, data integrity can be ensured through the implementation of Integrity Constraints in a table. Integrity constraints help apply business rules to the database tables. The constraints can either be at a column level or a table level. Some of the most common constraints are,

- NOT NULL – Prevents a column from having a NULL value.
- PRIMARY KEY – Uniquely identifies each row or record in table.
- FOREIGN KEY – Uniquely identifies a column that references a PRIMARY KEY in another table.
- UNIQUE – Prevents a column from having duplicate values.
- CHECK – Checks for values that satisfy a specific condition as defined by the user.

Listed below are the constraints that were created for our database development project along with their purpose-

```
CREATE TABLESPACE students
--  WARNING: Tablespace has no data files defined
 LOGGING ONLINE EXTENT MANAGEMENT LOCAL AUTOALLOCATE FLASHBACK ON;

CREATE USER db920 IDENTIFIED BY account UNLOCK
;

-- predefined type, no DDL - MDSYS.SDO_GEOMETRY

-- predefined type, no DDL - XMLTYPE

CREATE TABLE db920.bills_finance (
    bills_id      NUMBER(25) NOT NULL,
    cash          VARCHAR2(50 BYTE) NOT NULL,
    cheque        VARCHAR2(50 BYTE) NOT NULL,
    amount_total  NUMBER(25) NOT NULL,
    amount_due    NUMBER(25) NOT NULL,
    amount_paid   VARCHAR2(20 BYTE) NOT NULL,
    due_date      DATE NOT NULL,
    patient_id    NUMBER(25) NOT NULL,
    hosp_id       NUMBER(25) NOT NULL
)
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
```

```
   STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
   DEFAULT );

CREATE UNIQUE INDEX db920.bills_finance_pk ON
   db920.bills_finance (
      bills_id
   ASC )
      TABLESPACE students PCTFREE 10
         STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1
MAXEXTENTS 2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
         DEFAULT )
      LOGGING;

ALTER TABLE db920.bills_finance
   ADD CONSTRAINT bills_finance_pk PRIMARY KEY ( bills_id )
      USING INDEX db920.bills_finance_pk;

CREATE TABLE db920.doctor_data (
   doctor_id    NUMBER(30) NOT NULL,
   hosp_id      NUMBER(20) NOT NULL,
   doctor_name  VARCHAR2(50 BYTE) NOT NULL,
   doc_email    VARCHAR2(50 BYTE) NOT NULL,
   doc_phone    VARCHAR2(20 BYTE)
)
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
   STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
   DEFAULT );

CREATE UNIQUE INDEX db920.doctor_data_pk ON
   db920.doctor_data (
      doctor_id
   ASC )
      TABLESPACE students PCTFREE 10
         STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1
MAXEXTENTS 2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
         DEFAULT )
      LOGGING;

ALTER TABLE db920.doctor_data
   ADD CONSTRAINT doctor_data_pk PRIMARY KEY ( doctor_id )
      USING INDEX db920.doctor_data_pk;

CREATE TABLE db920.hosp_patient_link (
   link_id     NUMBER(10) NOT NULL,
   patient_id  NUMBER(10) NOT NULL,
   hosp_id     NUMBER(10) NOT NULL
```

```
)
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
  STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
  DEFAULT );

CREATE UNIQUE INDEX db920.hosp_patient_link_pk ON
  db920.hosp_patient_link (
    link_id
  ASC )
    TABLESPACE students PCTFREE 10
      STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1
MAXEXTENTS 2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
      DEFAULT )
    LOGGING;

ALTER TABLE db920.hosp_patient_link
  ADD CONSTRAINT hosp_patient_link_pk PRIMARY KEY ( link_id )
    USING INDEX db920.hosp_patient_link_pk;

CREATE TABLE db920.hospital_data (
  hosp_id              NUMBER(25) NOT NULL,
  capacity             NUMBER(20) NOT NULL,
  name                 VARCHAR2(50 BYTE) NOT NULL,
  address              VARCHAR2(50 BYTE) NOT NULL,
  city                 VARCHAR2(50 BYTE) NOT NULL,
  state                VARCHAR2(50 BYTE) NOT NULL,
  county               VARCHAR2(50 BYTE),
  zip_code             VARCHAR2(50 BYTE) NOT NULL,
  beds_available       NUMBER(25),
  doctors_count        NUMBER(25),
  hospital_type        VARCHAR2(50 BYTE),
  hospital_overall_rating  NUMBER(25),
  emergency_services   VARCHAR2(20 BYTE)
)
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
  STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
  DEFAULT );

CREATE UNIQUE INDEX db920.hospital_data_pk ON
  db920.hospital_data (
    hosp_id
  ASC )
    TABLESPACE students PCTFREE 10
      STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1
MAXEXTENTS 2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
      DEFAULT )
```

```
    LOGGING;

ALTER TABLE db920.hospital_data
  ADD CONSTRAINT hospital_data_pk PRIMARY KEY ( hosp_id )
    USING INDEX db920.hospital_data_pk;

CREATE TABLE db920.hospital_inventory (
  inv_id        NUMBER(25) NOT NULL,
  inventory_mtrl  VARCHAR2(50 BYTE) NOT NULL,
  inv_unitprice   NUMBER(25) NOT NULL,
  inv_cost       NUMBER(25) NOT NULL
)
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
  STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
  DEFAULT );

CREATE UNIQUE INDEX db920.hospital_inventory_pk ON
  db920.hospital_inventory (
    inv_id
  ASC )
    TABLESPACE students PCTFREE 10
      STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1
MAXEXTENTS 2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
      DEFAULT )
    LOGGING;

ALTER TABLE db920.hospital_inventory
  ADD CONSTRAINT hospital_inventory_pk PRIMARY KEY ( inv_id )
    USING INDEX db920.hospital_inventory_pk;

CREATE TABLE db920.hospital_pharmacy_link (
  hplink_id      NUMBER(25) NOT NULL,
  hosp_id        NUMBER(25) NOT NULL,
  medicine_id    NUMBER(25) NOT NULL,
  medicine_count  NUMBER(25) NOT NULL
)
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
  STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
  DEFAULT );

CREATE TABLE db920.hospitalinventory_link (
  hil_id    NUMBER(25) NOT NULL,
  hosp_id   NUMBER(25) NOT NULL,
  inv_id    NUMBER(25) NOT NULL,
  inv_count  NUMBER(25) NOT NULL
)
```

```
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
   STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
   DEFAULT );
```

```
CREATE TABLE db920.inpatient_visit_history (
   inpatient_id          NUMBER(20),
   in_date               DATE NOT NULL,
   handwritten_textnotes  VARCHAR2(50 BYTE) NOT NULL,
   prescription_history   NVARCHAR2(50) NOT NULL,
   medication_prescribed  NVARCHAR2(200) NOT NULL,
   next_checkupdate       DATE NOT NULL,
   ip_notes               VARCHAR2(200 BYTE) NOT NULL,
   patient_id             NUMBER(25) NOT NULL,
   doctor_id              NUMBER(25) NOT NULL
)
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
   STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
   DEFAULT );
```

```
CREATE TABLE db920.insurance (
   insurance_id             NUMBER(25) NOT NULL,
   hospital_insurance_tieup  VARCHAR2(50 BYTE) NOT NULL,
   corporate_insurance       VARCHAR2(150 BYTE) NOT NULL,
   personal_insurance        VARCHAR2(150 BYTE) NOT NULL,
   patient_id                NUMBER(25) NOT NULL
)
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
   STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
   DEFAULT );
```

```
CREATE TABLE db920.lab_reports (
   branch_id          NUMBER(25) NOT NULL,
   branch_name        VARCHAR2(50 BYTE) NOT NULL,
   patient_id         NUMBER(25) NOT NULL,
   access_sensitivity  VARCHAR2(50 BYTE),
   download_metrics    VARCHAR2(50 BYTE) NOT NULL,
   report_summary      VARCHAR2(300 BYTE) NOT NULL
)
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
   STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
   DEFAULT );
```

```
CREATE TABLE db920.nurse_data (
   nurse_id    NUMBER(25) NOT NULL,
```

```
   nurse_name  VARCHAR2(50 BYTE) NOT NULL,
   hosp_id     NUMBER(25) NOT NULL
)
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
   STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
   DEFAULT );

CREATE UNIQUE INDEX db920.nurse_data_pk ON
   db920.nurse_data (
      nurse_id
   ASC )
      TABLESPACE students PCTFREE 10
         STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1
MAXEXTENTS 2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
         DEFAULT )
      LOGGING;

ALTER TABLE db920.nurse_data
   ADD CONSTRAINT nurse_data_pk PRIMARY KEY ( nurse_id )
      USING INDEX db920.nurse_data_pk;

CREATE TABLE db920.patient_data (
   patient_id    NUMBER(25) NOT NULL,
   weight        VARCHAR2(25 BYTE) NOT NULL,
   bmi           FLOAT(126) NOT NULL,
   main_disease  VARCHAR2(150 BYTE) NOT NULL,
   height        VARCHAR2(25 BYTE) NOT NULL,
   phone         VARCHAR2(25 BYTE) NOT NULL,
   email_id      VARCHAR2(50 BYTE) NOT NULL,
   address       VARCHAR2(100 BYTE) NOT NULL,
   blood_group   VARCHAR2(15 BYTE) NOT NULL,
   sex           VARCHAR2(25 BYTE) NOT NULL,
   age           NUMBER(20) NOT NULL,
   admit_id      NUMBER(25),
   deficiency    VARCHAR2(25 BYTE) NOT NULL,
   nurse_id      NUMBER(25),
   patient_name  VARCHAR2(50 BYTE) NOT NULL
)
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
   STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
   DEFAULT );

CREATE UNIQUE INDEX db920.patient_data_pk ON
   db920.patient_data (
      patient_id
   ASC )
```

```
    TABLESPACE students PCTFREE 10
      STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1
MAXEXTENTS 2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
        DEFAULT )
    LOGGING;
```

ALTER TABLE db920.patient_data

```
  ADD CONSTRAINT patient_data_pk PRIMARY KEY ( patient_id )
    USING INDEX db920.patient_data_pk;
```

CREATE TABLE db920.pharmacy_data (

```
  medicine_id      NUMBER(25) NOT NULL,
  purchase_amount  FLOAT(126),
  medicine_name    NVARCHAR2(50) NOT NULL,
  dosage           VARCHAR2(50 BYTE) NOT NULL,
  payment_methods  VARCHAR2(50 BYTE) NOT NULL
)
PCTFREE 10 PCTUSED 40 TABLESPACE students LOGGING
  STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1 MAXEXTENTS
2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
  DEFAULT );
```

CREATE UNIQUE INDEX db920.pharmacy_data_pk ON

```
  db920.pharmacy_data (
    medicine_id
  ASC )
    TABLESPACE students PCTFREE 10
      STORAGE ( INITIAL 65536 NEXT 1048576 PCTINCREASE 0 MINEXTENTS 1
MAXEXTENTS 2147483645 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL
        DEFAULT )
    LOGGING;
```

ALTER TABLE db920.pharmacy_data

```
  ADD CONSTRAINT pharmacy_data_pk PRIMARY KEY ( medicine_id )
    USING INDEX db920.pharmacy_data_pk;
```

ALTER TABLE db920.bills_finance

```
  ADD CONSTRAINT fk_bills_hosp_id FOREIGN KEY ( hosp_id )
    REFERENCES db920.hospital_data ( hosp_id )
  NOT DEFERRABLE;
```

ALTER TABLE db920.bills_finance

```
  ADD CONSTRAINT fk_bills_patient_id FOREIGN KEY ( patient_id )
    REFERENCES db920.patient_data ( patient_id )
  NOT DEFERRABLE;
```

ALTER TABLE db920.doctor_data

```
  ADD CONSTRAINT fk_dd_hosp_id FOREIGN KEY ( hosp_id )
```

```
    REFERENCES db920.hospital_data ( hosp_id )
  NOT DEFERRABLE;

ALTER TABLE db920.hospitalinventory_link
  ADD CONSTRAINT fk_hil_hosp_id FOREIGN KEY ( hosp_id )
    REFERENCES db920.hospital_data ( hosp_id )
  NOT DEFERRABLE;

ALTER TABLE db920.hospitalinventory_link
  ADD CONSTRAINT fk_hil_inv_id FOREIGN KEY ( inv_id )
    REFERENCES db920.hospital_inventory ( inv_id )
  NOT DEFERRABLE;

ALTER TABLE db920.hospital_pharmacy_link
  ADD CONSTRAINT fk_hphl_hosp_id FOREIGN KEY ( hosp_id )
    REFERENCES db920.hospital_data ( hosp_id )
  NOT DEFERRABLE;

ALTER TABLE db920.hospital_pharmacy_link
  ADD CONSTRAINT fk_hphl_medicine_id FOREIGN KEY ( medicine_id )
    REFERENCES db920.pharmacy_data ( medicine_id )
  NOT DEFERRABLE;

ALTER TABLE db920.hosp_patient_link
  ADD CONSTRAINT fk_hpl_hosp_id FOREIGN KEY ( hosp_id )
    REFERENCES db920.hospital_data ( hosp_id )
  NOT DEFERRABLE;

ALTER TABLE db920.hosp_patient_link
  ADD CONSTRAINT fk_hpl_patient_id FOREIGN KEY ( patient_id )
    REFERENCES db920.patient_data ( patient_id )
  NOT DEFERRABLE;

ALTER TABLE db920.insurance
  ADD CONSTRAINT fk_in_patient_id FOREIGN KEY ( patient_id )
    REFERENCES db920.patient_data ( patient_id )
  NOT DEFERRABLE;

ALTER TABLE db920.inpatient_visit_history
  ADD CONSTRAINT fk_ivh_doctor_id FOREIGN KEY ( doctor_id )
    REFERENCES db920.doctor_data ( doctor_id )
  NOT DEFERRABLE;

ALTER TABLE db920.inpatient_visit_history
  ADD CONSTRAINT fk_ivh_patient_id FOREIGN KEY ( patient_id )
    REFERENCES db920.patient_data ( patient_id )
  NOT DEFERRABLE;
```

ALTER TABLE db920.lab_reports
ADD CONSTRAINT fk_lab_patient_id FOREIGN KEY (patient_id)
REFERENCES db920.patient_data (patient_id)
NOT DEFERRABLE;

ALTER TABLE db920.nurse_data
ADD CONSTRAINT fk_nur_hosp_id FOREIGN KEY (hosp_id)
REFERENCES db920.hospital_data (hosp_id)
NOT DEFERRABLE;

ALTER TABLE db920.patient_data
ADD CONSTRAINT fk_pat_nurse_id FOREIGN KEY (nurse_id)
REFERENCES db920.nurse_data (nurse_id)
NOT DEFERRABLE;

```
-- Oracle SQL Developer Data Modeler Summary Report:
--
-- CREATE TABLE                            13
-- CREATE INDEX                             8
-- ALTER TABLE                             23
-- CREATE VIEW                              0
-- ALTER VIEW                               0
-- CREATE PACKAGE                           0
-- CREATE PACKAGE BODY                      0
-- CREATE PROCEDURE                         0
-- CREATE FUNCTION                          0
-- CREATE TRIGGER                           0
-- ALTER TRIGGER                            0
-- CREATE COLLECTION TYPE                   0
-- CREATE STRUCTURED TYPE                   0
-- CREATE STRUCTURED TYPE BODY              0
-- CREATE CLUSTER                           0
-- CREATE CONTEXT                           0
-- CREATE DATABASE                          0
-- CREATE DIMENSION                         0
-- CREATE DIRECTORY                         0
-- CREATE DISK GROUP                        0
-- CREATE ROLE                              0
-- CREATE ROLLBACK SEGMENT                  0
-- CREATE SEQUENCE                          0
-- CREATE MATERIALIZED VIEW                 0
-- CREATE MATERIALIZED VIEW LOG             0
-- CREATE SYNONYM                           0
-- CREATE TABLESPACE                        1
-- CREATE USER                              1
--
-- DROP TABLESPACE                          0
-- DROP DATABASE                            0
```

```
--
-- REDACTION POLICY                          0
-- TSDP POLICY                               0
--
-- ORDS DROP SCHEMA                          0
-- ORDS ENABLE SCHEMA                        0
-- ORDS ENABLE OBJECT                        0
--
-- ERRORS                                    0
-- WARNINGS                                  1
```

QUERY WRITING

1. **Fetch list of patients with their total amount & the due amount which is pending.**

```
SELECT bills_Id, patient_id, patient_name, SUM(amount_total), SUM(amount_due)
FROM patient_data
        INNER JOIN Bills_Finance
        Using (patient_id)
GROUP BY bills_Id, patient_id, patient_name;
```

Script Output × Query Result ×

SQL | Fetched 50 rows in 0.043 seconds

	BILLS_ID	PATIENT_ID	PATIENT_NAME	SUM(AMOUNT_TOTAL)	SUM(AMOUNT_DUE)
1	1242	23677	Aosephqf Taylor	6500	1716
2	1257	23662	Aen Walker	5000	1781
3	1259	23660	Aavid's Northof Allen	5000	1745
4	1267	23652	Aen Baker	5000	1752
5	1276	23643	Aospi Mitchell	6500	1763
6	1296	23623	AwesternMf Torres	5000	1799
7	1303	23616	A Kelly	6500	1777
8	1320	23599	Aul Barnes	6500	1683
9	1325	23594	Aea Fisher	5000	1657
10	1327	23592	Am Reves	5000	1502
11	1339	23580	Aen Jordan	5000	1679
12	1342	23577	Aen Griffin	5000	1579
13	1347	23572	Aea Bryant	5000	1674
14	1351	23568	Aen Harrison	5000	1779
15	1356	23563	AJnf Ramos	6500	1678
16	1361	23558	AAV Wells	6500	1692
17	1367	23552	A Gordon	6500	1700

2. **Display list the patients above a certain age; grouped on the basis of their location with their name and contact details.**

```
Select count(p.patient_id) AS No_Of_Patients, p.patient_name, p.phone, h.zip_code
From patient_data p
INNER JOIN Hosp_patient_link hl
ON (p.patient_id=hl.patient_id)
INNER JOIN Hospital_data h
ON (hl.Hosp_id=h.Hosp_id)
WHERE (age>20)
GROUP BY zip_code,p.patient_id, p.patient_name, p.phone;
```

Script Output | Query Result

SQL | Fetched 50 rows in 0.072 seconds

	NO_OF_PATIENTS	PATIENT_NAME	PHONE	ZIP_CODE
1	1	Ae Moore	8131111221	86504
2	1	Aazle Rodriquez	8131111222	86045
3	1	Aen Young	8131111239	92663
4	1	Aen Baker	8131111244	92691
5	1	Aospi Green	8131111245	92585
6	1	Aeba Roberts	8131111251	81641
7	1	Aen Parker	8131111259	32605
8	1	Aen Cook	8131111264	99574
9	1	Aospi Morgan	8131111268	30474
10	1	Aour Ward	8131111279	62002
11	1	Anlf Richardson	8131111285	61036
12	1	Aospi Myers	8131111292	47250
13	1	Adencewf Kim	8131111299	67029
14	1	Aospi Cole	8131111317	4769
15	1	Aen Griffin	8131111319	4915

3. **Display the list of hospitals which provide emergency services with a minimum capacity value where hospital's overall rating is greater than 3 out of 5.**

```
Select * from Hospital_data where (Capacity > 100)
        AND (Emergency_services = 'true')
        AND (Hospital_overall_rating > 3)
        AND (state = 'FL');
```

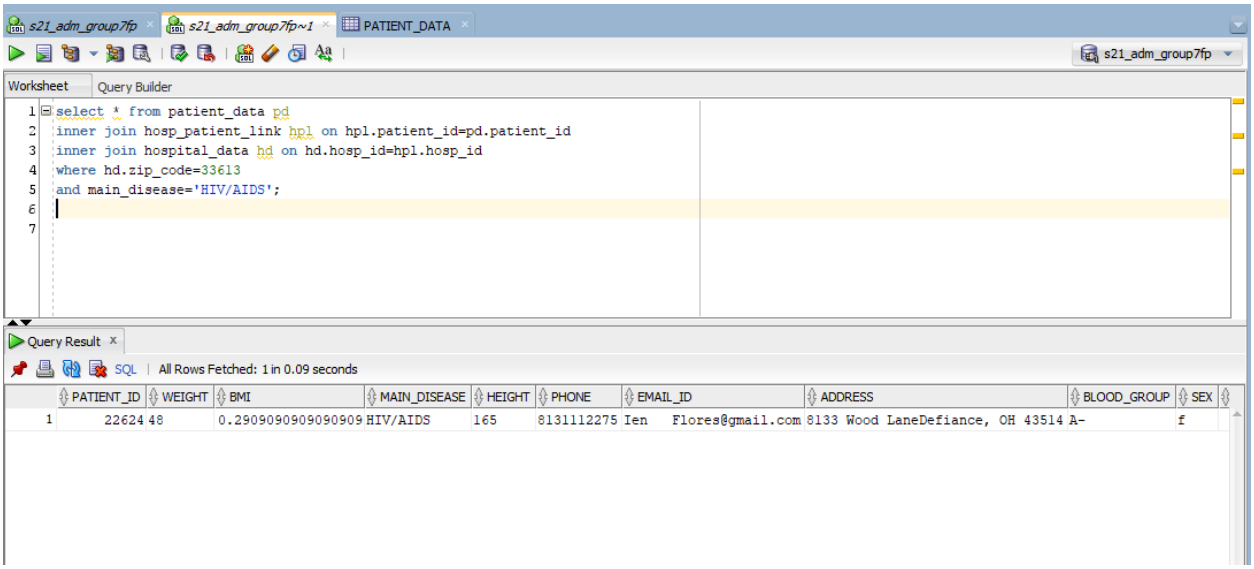
Script Output | Query Result

SQL | All Rows Fetched: 21 in 0.031 seconds

	HOSP_ID	CAPACITY	NAME	ADDRESS	CITY	STATE	COUNTY
1	1503	11560	BROWARD HEALTH CORAL SPRINGS	3000 CORAL HILLS DR	CORAL SPRINGS	FL	BROWARD
2	1505	11580	ST CLOUD REGIONAL MEDICAL CENTER	2906 17TH STREET	SAINT CLOUD	FL	OSCEOLA
3	1510	11630	PALMETTO GENERAL HOSPITAL	2001 W 68TH ST	HIALEAH	FL	MIAMI-DADE
4	1520	11730	REGIONAL GENERAL HOSPITAL WILLISTON	125 SW 7TH ST	WILLISTON	FL	LEVY
5	1538	11910	JUPITER MEDICAL CENTER	1210 S OLD DIXIE HWY	JUPITER	FL	PALM BEACH
6	1547	12000	MEMORIAL HOSPITAL JACKSONVILLE	3625 UNIVERSITY BLVD S	JACKSONVILLE	FL	DUVAL
7	1558	12110	BAYFRONT HEALTH - ST PETERSBURG	701 6TH ST S	SAINT PETERSBURG	FL	PINELLAS
8	1560	12130	OCALA REGIONAL MEDICAL CENTER	1431 SW 1ST AVE	OCALA	FL	MARION
9	1568	12210	SARASOTA MEMORIAL HOSPITAL	1700 S TAMIAMI TRL	SARASOTA	FL	SARASOTA
10	1580	12330	UF HEALTH JACKSONVILLE	655 W 8TH ST	JACKSONVILLE	FL	DUVAL
11	1581	12340	GEORGE E WEEMS MEMORIAL HOSPITAL	135 AVE G	APALACHICOLA	FL	FRANKLIN
12	1587	12400	HIGHLANDS REGIONAL MEDICAL CENTER	3600 S HIGHLANDS AVE	SEBRING	FL	HIGHLANDS
13	1589	12420	FLORIDA HOSPITAL NEW SMYRNA	401 PALMETTO ST	NEW SMYRNA BEACH	FL	VOLUSIA
14	1595	12480	BAPTIST HOSPITAL OF MIAMI	8900 N KENDALL DR	MIAMI	FL	MIAMI-DADE
15	1601	12540	BROWARD HEALTH NORTH	201 E SAMPLE RD	POMPANO BEACH	FL	BROWARD
16	1612	12650	FLORIDA HOSPITAL ZEPHYRHILLS	7050 GALL BLVD	ZEPHYRHILLS	FL	PASCO
17	1616	12690	SOUTH FLORIDA BAPTIST HOSPITAL	301 N ALEXANDER ST	PLANT CITY	FL	HILLSBOROUGH
18	1628	12810	FLORIDA HOSPITAL WAUCHULA	533 CARLTON ST	WAUCHULA	FL	HARDEE
19	1630	12830	SHANDS LIVE OAK REGIONAL MEDICAL CENTER	1100 SW 11TH ST	LIVE OAK	FL	SUWANNEE

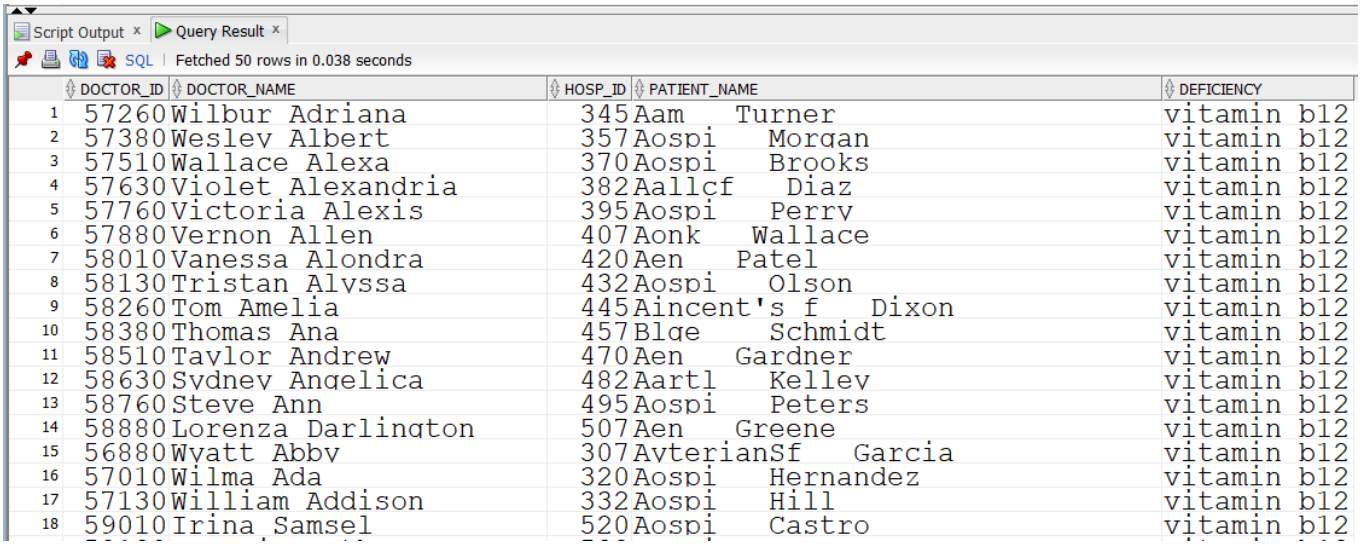
4. **Display list of patients who suffer from a particular disease in the same location.**

```
SELECT * from patient_data pd
inner join hosp_patient_link hpl on hpl.patient_id=pd.patient_id
inner join hospital_data hd on hd.hosp_id=hpl.hosp_id
where hd.zip_code=33613
and main_disease='HIV/AIDS';
```



5. **Display names of doctors who treated patients with vitamin deficiency**

```
Select d.doctor_id,d.doctor_name,d.hosp_id,p.patient_name,p.deficiency

From doctor_data d

INNER JOIN Inpatient_visit_history i

ON (d.doctor_id=i.doctor_id)

INNER JOIN Patient_data p

ON (i.patient_id=p.patient_id)

WHERE deficiency LIKE 'vitamin%';
```



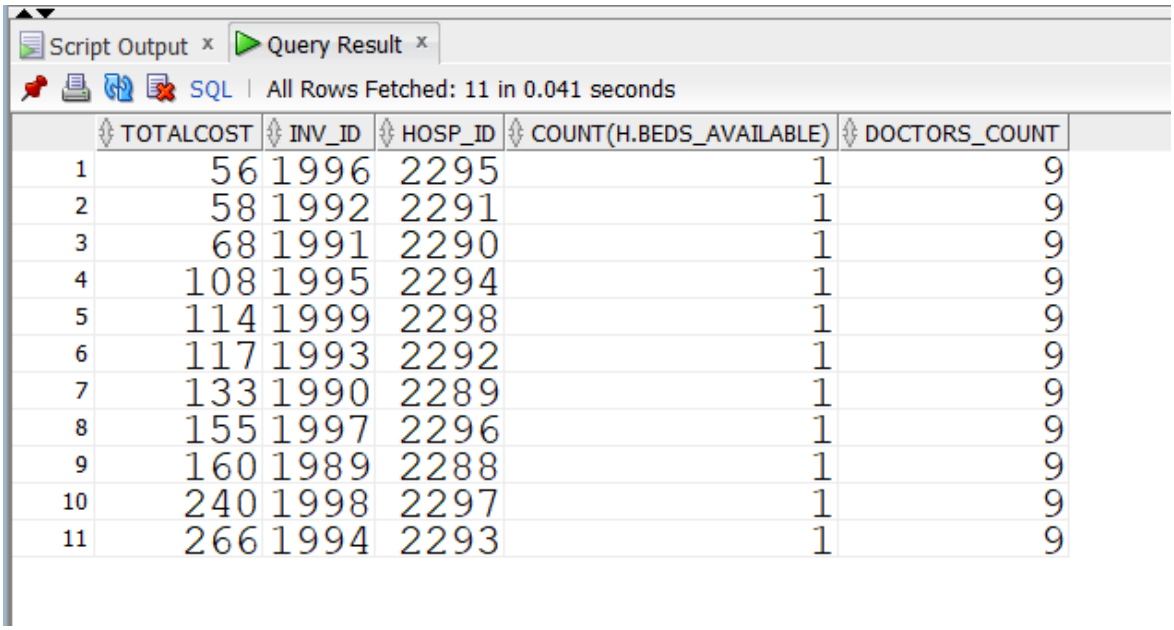
Script Output × Query Result ×

SQL | Fetched 50 rows in 0.038 seconds

	DOCTOR_ID	DOCTOR_NAME	HOSP_ID	PATIENT_NAME	DEFICIENCY
1	57260	Wilbur Adriana	345	Aam Turner	vitamin b12
2	57380	Wesley Albert	357	Aospi Morgan	vitamin b12
3	57510	Wallace Alexa	370	Aospi Brooks	vitamin b12
4	57630	Violet Alexandria	382	Aallcf Diaz	vitamin b12
5	57760	Victoria Alexis	395	Aospi Perry	vitamin b12
6	57880	Vernon Allen	407	Aonk Wallace	vitamin b12
7	58010	Vanessa Alondra	420	Aen Patel	vitamin b12
8	58130	Tristan Alyssa	432	Aospi Olson	vitamin b12
9	58260	Tom Amelia	445	Aincent's f Dixon	vitamin b12
10	58380	Thomas Ana	457	Blge Schmidt	vitamin b12
11	58510	Taylor Andrew	470	Aen Gardner	vitamin b12
12	58630	Sydney Angelica	482	Aartl Kelley	vitamin b12
13	58760	Steve Ann	495	Aospi Peters	vitamin b12
14	58880	Lorenza Darlington	507	Aen Greene	vitamin b12
15	56880	Wyatt Abby	307	AyterianSf Garcia	vitamin b12
16	57010	Wilma Ada	320	Aospi Hernandez	vitamin b12
17	57130	William Addison	332	Aospi Hill	vitamin b12
18	59010	Irina Samsel	520	Aospi Castro	vitamin b12

6. **Find the total cost of inventory of a particular hospital with a minimum of 3 doctors and number of beds are above 1990.**

```
Select SUM(hl.inv_count * i.inv_unitprice) AS totalcost,
i.inv_id, hl.hosp_id,
count(h.beds_available),
h.doctors_count
    From hospital_inventory i
    INNER JOIN hospitalinventory_link hl
    ON (i.inv_id = hl.inv_id)

    INNER JOIN hospital_data h

    ON (hl.hosp_id = h.hosp_id)

    Where h.beds_available > 1990 AND h.doctors_count >3

    GROUP BY h.beds_available, i.inv_id, hl.hosp_id, h.doctors_count

    Order by totalcost;
```



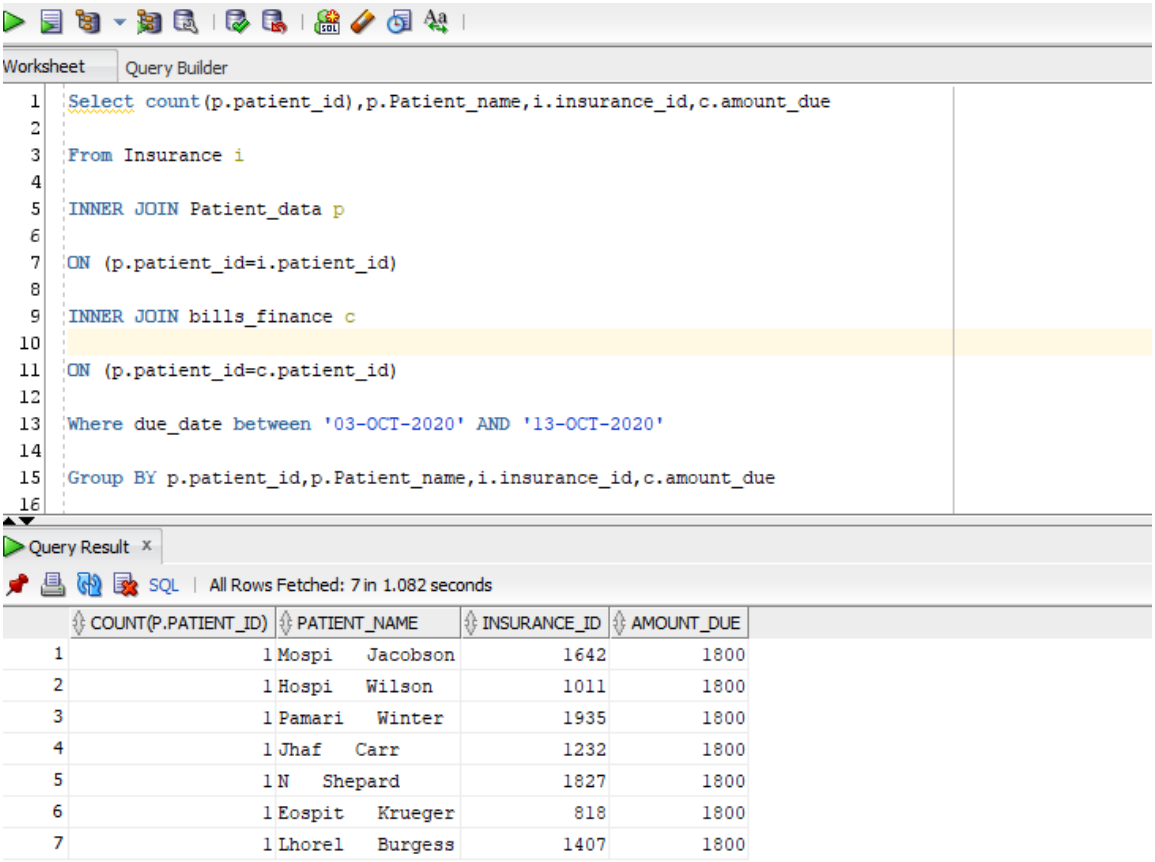
Script Output | Query Result

SQL | All Rows Fetched: 11 in 0.041 seconds

	TOTALCOST	INV_ID	HOSP_ID	COUNT(H.BEDS_AVAILABLE)	DOCTORS_COUNT
1	56	1996	2295	1	9
2	58	1992	2291	1	9
3	68	1991	2290	1	9
4	108	1995	2294	1	9
5	114	1999	2298	1	9
6	117	1993	2292	1	9
7	133	1990	2289	1	9
8	155	1997	2296	1	9
9	160	1989	2288	1	9
10	240	1998	2297	1	9
11	266	1994	2293	1	9

7. **Display list of patients and their associated insurance id who owe at least $1800 and the due date is within a particular date range.**

```
Select count(p.patient_id),p.Patient_name,i.insurance_id,c.amount_due

From Insurance i

INNER JOIN Patient_data p

ON (p.patient_id=i.patient_id)

INNER JOIN bills_finance c

ON (p.patient_id=c.patient_id)

Where due_date between '03-OCT-2020' AND '13-OCT-2020'

Group BY p.patient_id,p.Patient_name,i.insurance_id,c.amount_due

HAVING c.amount_due>1799;
```



Worksheet | Query Builder

```
1  Select count(p.patient_id),p.Patient_name,i.insurance_id,c.amount_due
2
3  From Insurance i
4
5  INNER JOIN Patient_data p
6
7  ON (p.patient_id=i.patient_id)
8
9  INNER JOIN bills_finance c
10
11 ON (p.patient_id=c.patient_id)
12
13 Where due_date between '03-OCT-2020' AND '13-OCT-2020'
14
15 Group BY p.patient_id,p.Patient_name,i.insurance_id,c.amount_due
16
```

Query Result | SQL | All Rows Fetched: 7 in 1.082 seconds

	COUNT(P.PATIENT_ID)	PATIENT_NAME	INSURANCE_ID	AMOUNT_DUE
1	1	Mospi Jacobson	1642	1800
2	1	Hospi Wilson	1011	1800
3	1	Pamari Winter	1935	1800
4	1	Jhaf Carr	1232	1800
5	1	N Shepard	1827	1800
6	1	Eospit Krueger	818	1800
7	1	Lhorel Burgess	1407	1800

PERFORMANCE TUNING

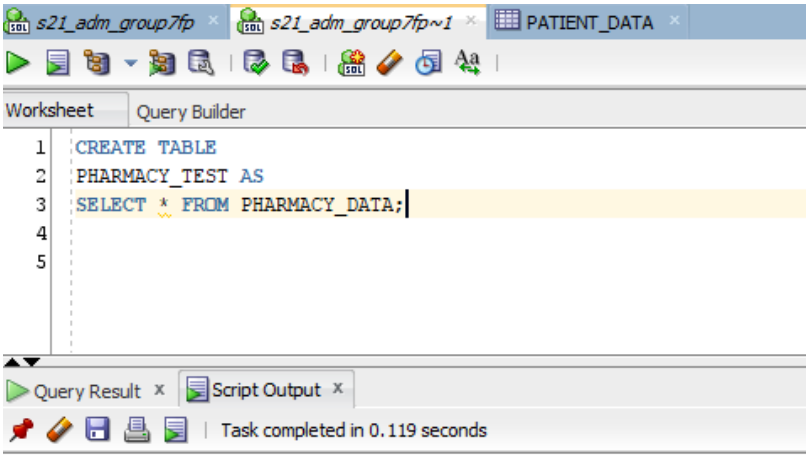
A. INDEXING

1. Create duplicate table PHARMACY_TEST for PHARMACY_DATA table:

CREATE TABLE

PHARMACY_TEST AS

SELECT * FROM PHARMACY_DATA



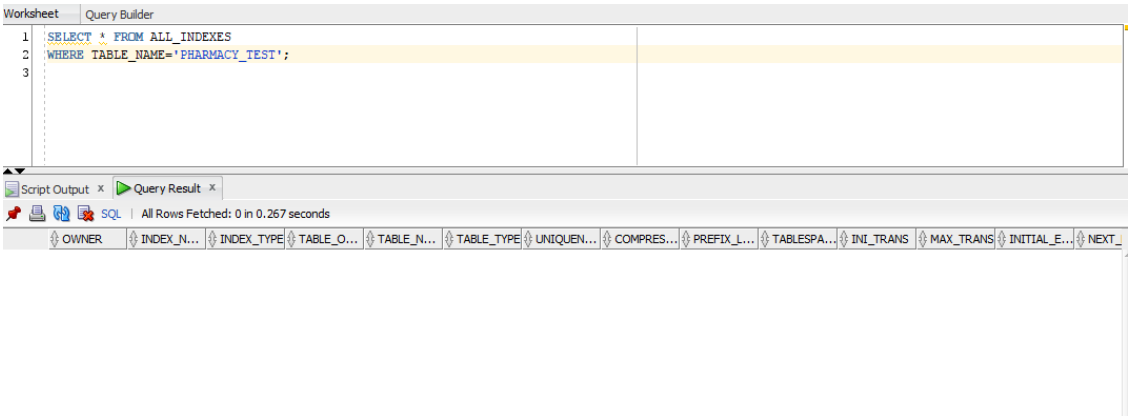

```
Table PHARMACY_TEST created.
```

2. Checking the index on PHARMACY_TEST table

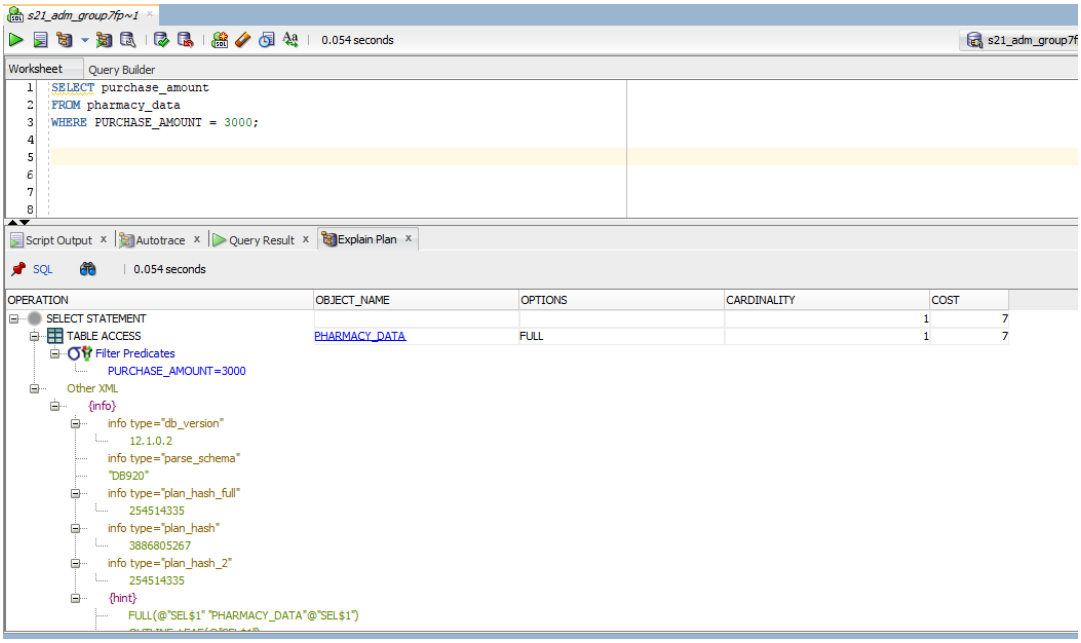
NO results as there are no indexes

SELECT * FROM ALL_INDEXES

WHERE TABLE_NAME='PHARMACY_TEST';



3.Performance of the query without the index.



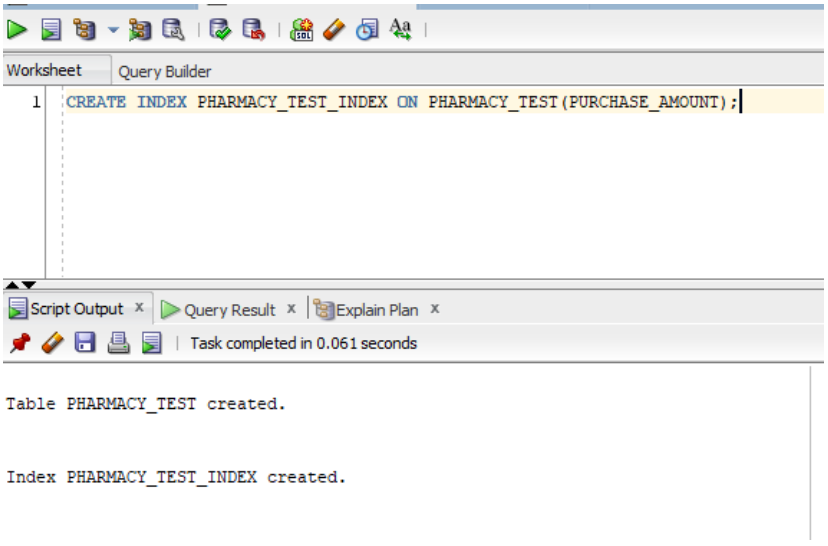
4.CREATING AN INDEX ON COLUMN PURCHASE_AMOUNT

CREATE INDEX PHARMACY_TEST_INDEX ON PHARMACY_TEST(PURCHASE_AMOUNT);

SELECT *

FROM ALL_INDEXES

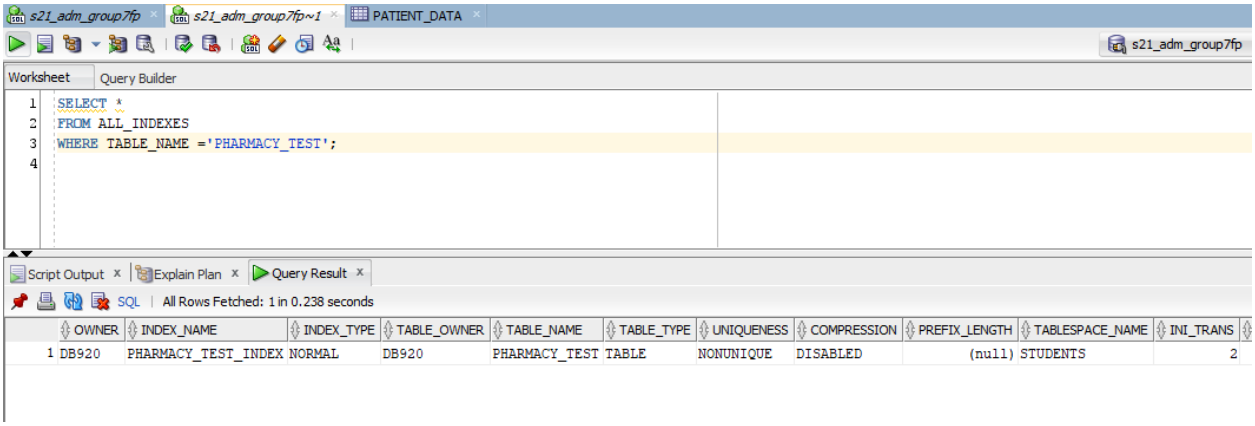
WHERE TABLE_NAME ='PHARMACY_TEST';



5.CHECKING THE INDEX AGAIN ON PHARMACY_TEST.

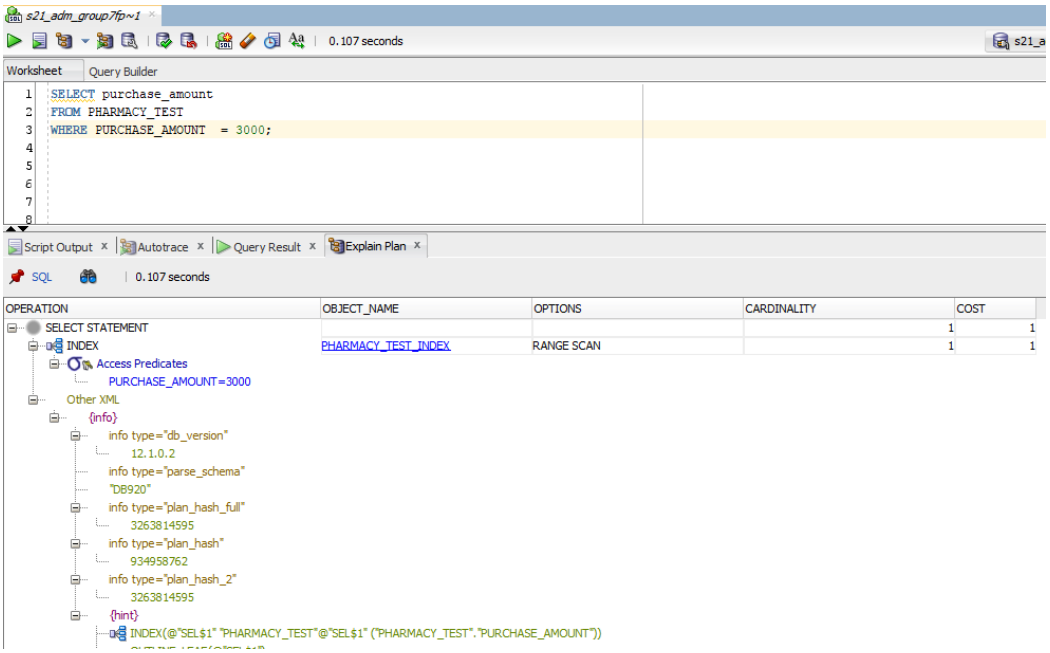
From the query result, we can see a record of the index-

```
SELECT *
FROM ALL_INDEXES
WHERE TABLE_NAME ='PHARMACY_TEST';
```



6.PERFORMANCE OF THE QUERY AFTER CREATING THE INDEX

```
SELECT purchase_amount
FROM PHARMACY_TEST
WHERE PURCHASE_AMOUNT  = 3000;
```



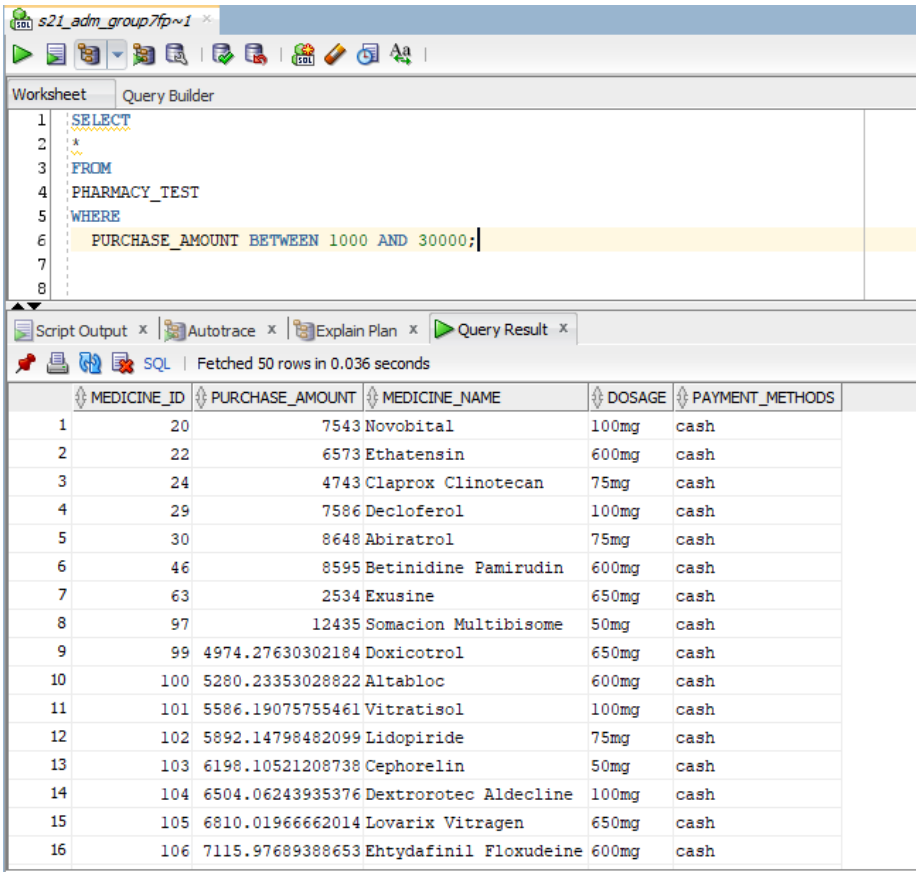
From above results, we can clearly see that Indexing helps in optimizing query execution time as the **cost decreased from 7 to 1. This will be much significant when we have huge data.**

B. EXECUTION PLAN

This plan shows execution of the SELECT statement. The table PHARMACY_TEST is being accessed by using a full table scan.

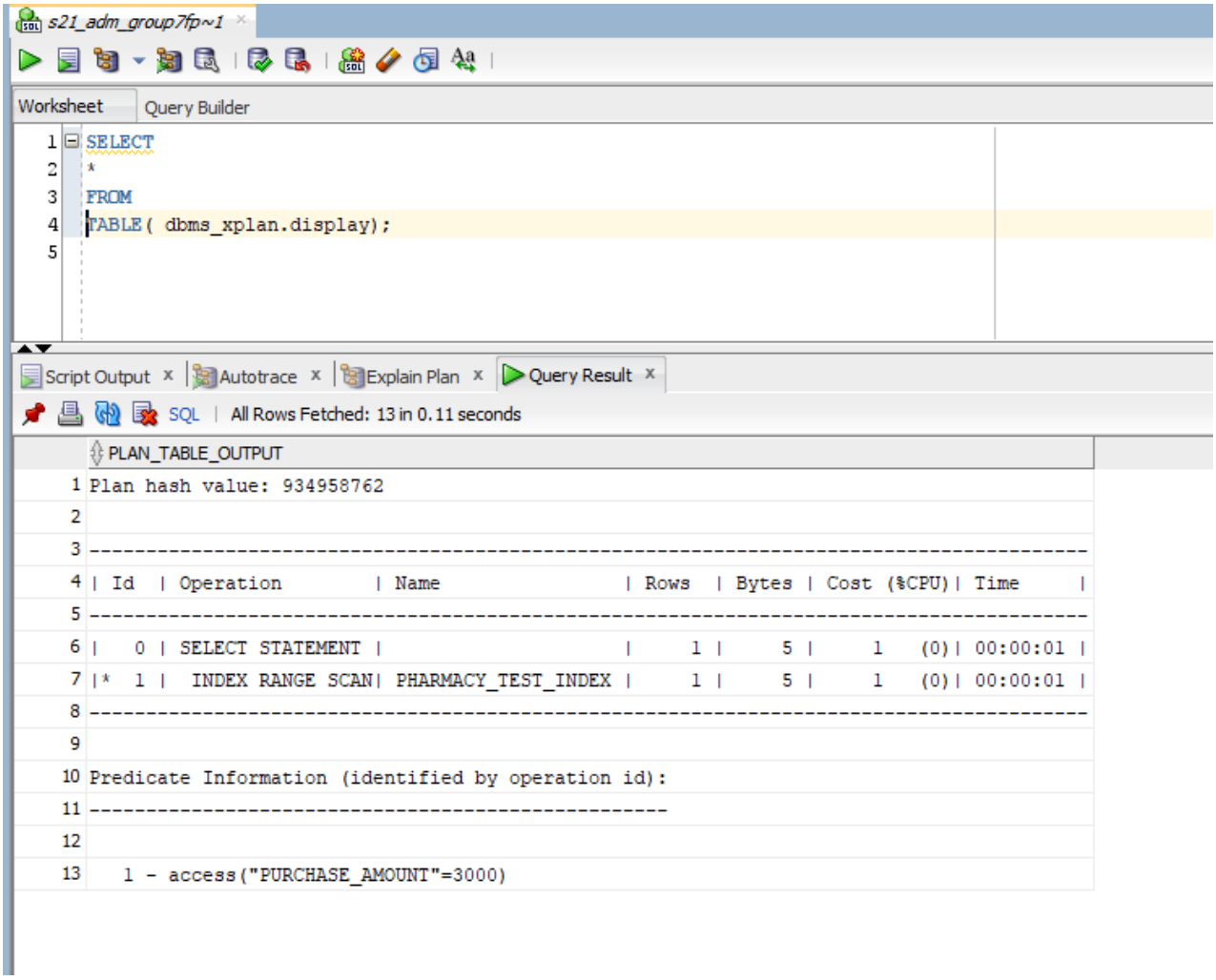
- Every row in the table PHARMACY_TEST is accessed.
- For every row, the WHERE clause criteria is evaluated.
- The SELECT statement returns the rows meeting the WHERE clause criteria.

```
EXPLAIN PLAN
FOR
SELECT
*
FROM
PHARMACY_TEST
WHERE
 PURCHASE_AMOUNT BETWEEN 1000 AND 30000;
```



	MEDICINE_ID	PURCHASE_AMOUNT	MEDICINE_NAME	DOSAGE	PAYMENT_METHODS
1	20	7543	Novobital	100mg	cash
2	22	6573	Ethatensin	600mg	cash
3	24	4743	Claprox Clinotecan	75mg	cash
4	29	7586	Decloferol	100mg	cash
5	30	8648	Abiratrol	75mg	cash
6	46	8595	Betinidine Pamirudin	600mg	cash
7	63	2534	Exusine	650mg	cash
8	97	12435	Somacion Multibisome	50mg	cash
9	99	4974.27630302184	Doxicotrol	650mg	cash
10	100	5280.23353028822	Altabloc	600mg	cash
11	101	5586.19075755461	Vitratisol	100mg	cash
12	102	5892.14798482099	Lidopiride	75mg	cash
13	103	6198.10521208738	Cephorelin	50mg	cash
14	104	6504.06243935376	Dextrorotec Aldecline	100mg	cash
15	105	6810.01966662014	Lovarix Vitragen	650mg	cash
16	106	7115.97689388653	Ehtydafinil Floxudeine	600mg	cash

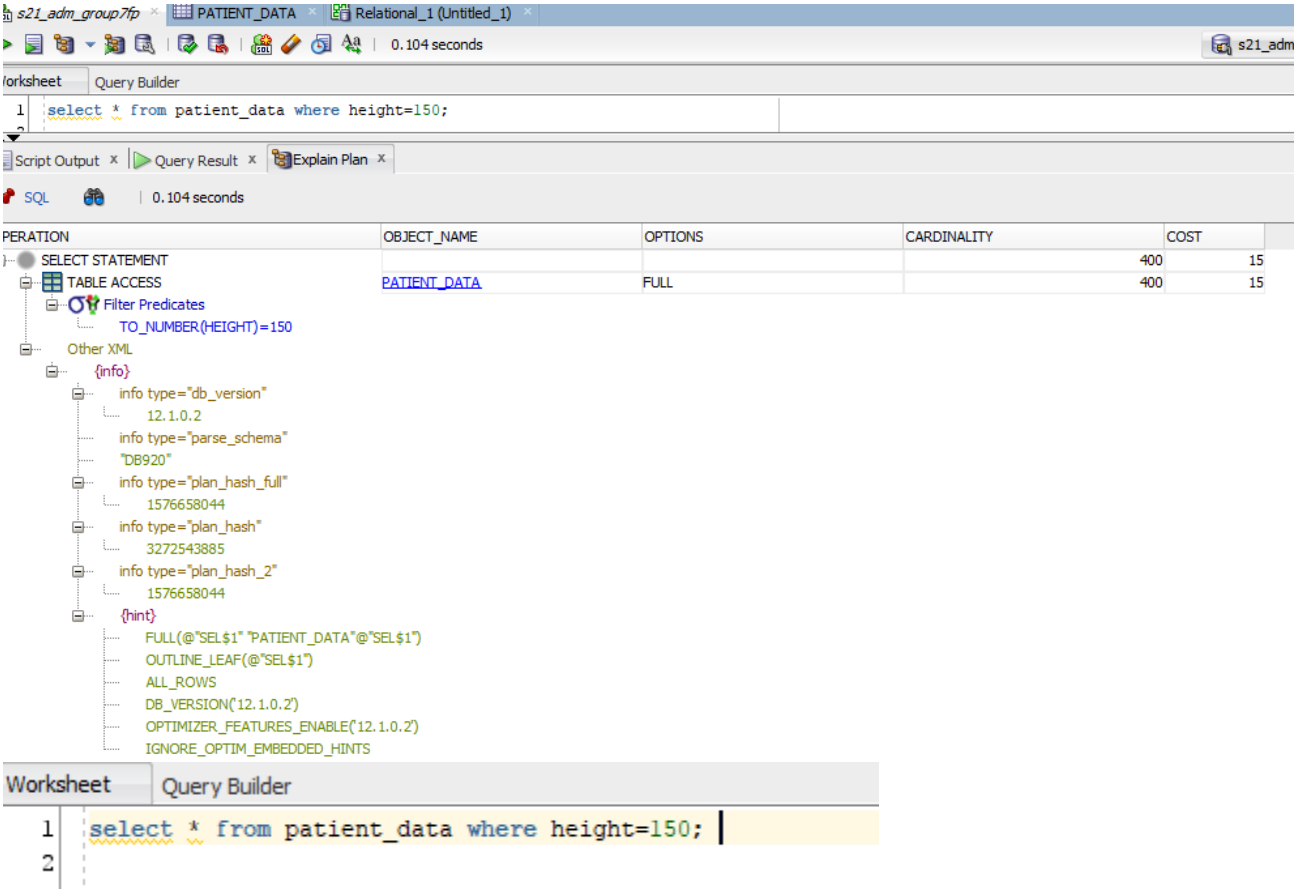
```
SELECT
*
FROM
TABLE( dbms_xplan.display);
```



Here we can find the execution plan where it specifies the data size, no. of rows, cost, CPU Utilization and the indexes used.

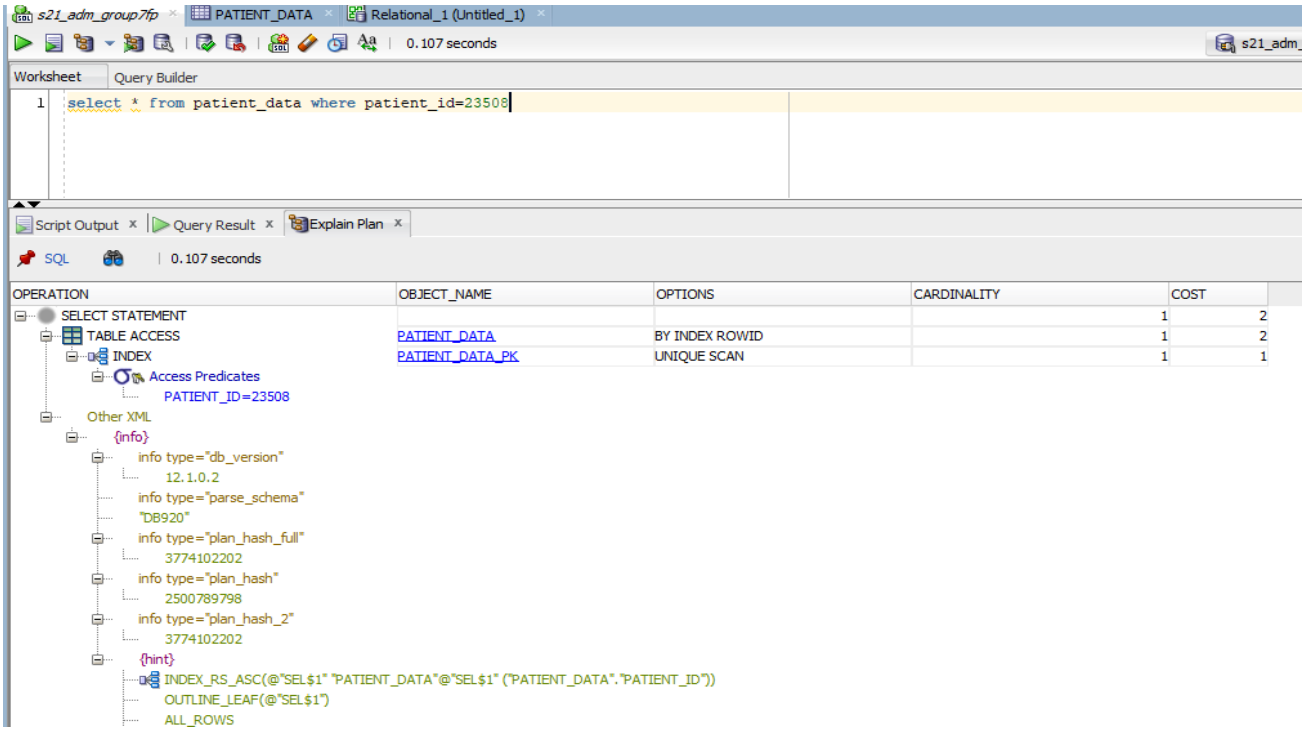
C.POINT QUERIES

Point queries are used for selecting small set of data which return small number of rows.
Query:



Here we are retrieving patients who have a height of 150. We can see that the cost of the execution is 15 as there is no index and the query went for full table scan.

- When we retrieve data using the column **patient_id** we can see that the cost of execution has decreased significantly as there is an index by default in the primary key column **patient_id.**



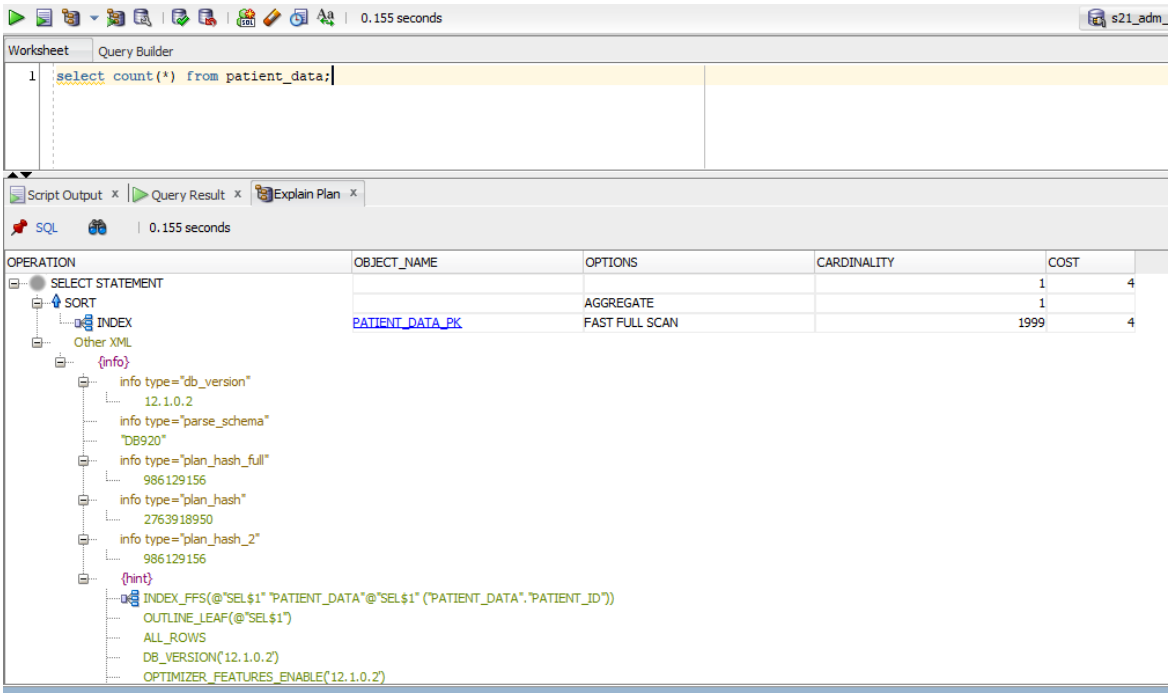
Inference: In the above scenario we can see the performance jump/cost reduction in point queries where index is being used.

D. PARELLEL DATABASES

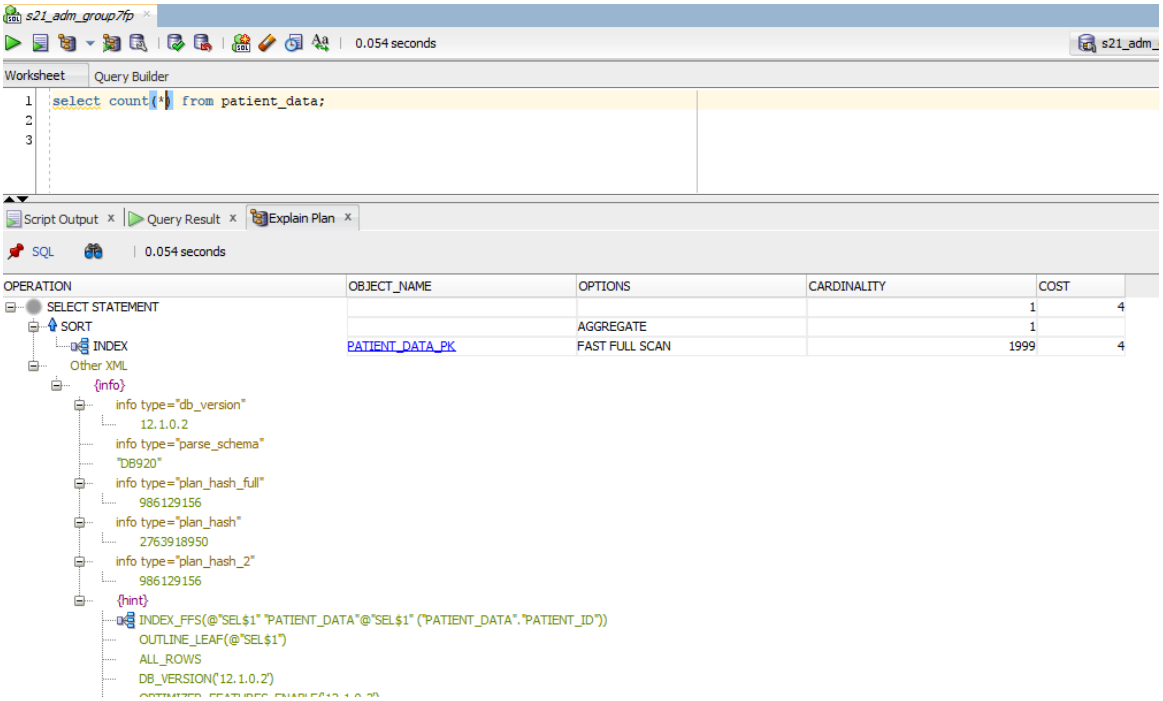
Basic Parallel Execution

Parallel execution is very useful in utilizing parallel processing to execute complex queries, queries that have huge rows in their outputs. Parallel execution can help us reduce the cost and query time involved in running these large-scale queries and help us optimize of querying.
The examples below will show the cost difference between using parallel execution and without using parallel execution.

- Finding the cost for finding the total number of rows in the **patient_data** table without using parallel execution.



- With parallel execution.



- In our scenario we can see that the query optimizer chooses not to use parallel execution as the data size is too less to use parallel execution and the cost of using parallel execution outweighs its benefits.

DBA SCRIPTS

1.

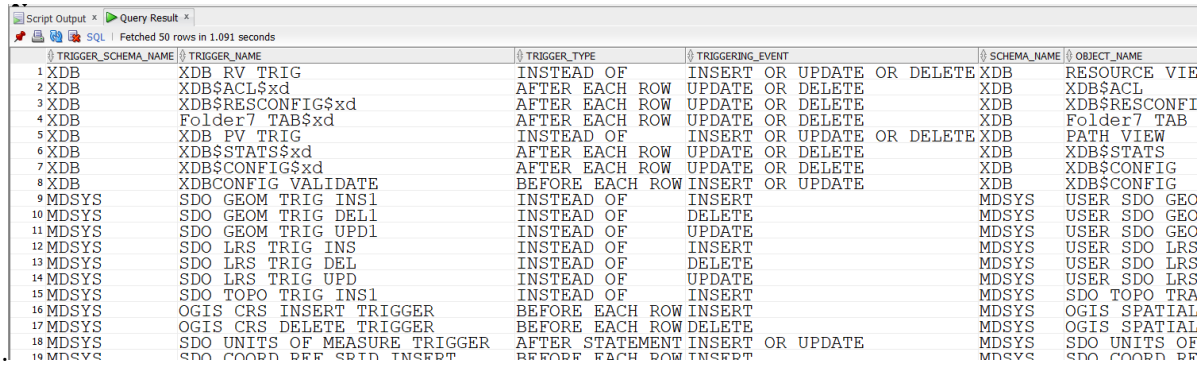
-- File Name : dba/monitoring/table_triggers.sql

-- Author : Ananya Shrivastava

-- Description : Lists the triggers for the specified table.

-- Call Syntax : @table_triggers (schema) (table_name)

```
SELECT owner as trigger_schema_name,
trigger_name, trigger_type, triggering_event,
table_owner as schema_name,
table_name as object_name,
base_object_type as object_type, status,
trigger_body as script
        from sys.all_triggers;
```



Script Output | Query Result

SQL | Fetched 50 rows in 1.091 seconds

	TRIGGER_SCHEMA_NAME	TRIGGER_NAME	TRIGGER_TYPE	TRIGGERING_EVENT	SCHEMA_NAME	OBJECT_NAME
1	XDB	XDB_RV_TRIG	INSTEAD OF	INSERT OR UPDATE OR DELETE	XDB	RESOURCE_VIE
2	XDB	XDBACLxd	AFTER EACH ROW	UPDATE OR DELETE	XDB	XDB$ACL
3	XDB	XDB$RESCONFIG$xd	AFTER EACH ROW	UPDATE OR DELETE	XDB	XDB$RESCONFI
4	XDB	Folder7_TAB$xd	AFTER EACH ROW	UPDATE OR DELETE	XDB	Folder7_TAB
5	XDB	XDB_PV_TRIG	INSTEAD OF	INSERT OR UPDATE OR DELETE	XDB	PATH_VIEW
6	XDB	XDB$STATS$xd	AFTER EACH ROW	UPDATE OR DELETE	XDB	XDB$STATS
7	XDB	XDB$CONFIG$xd	AFTER EACH ROW	UPDATE OR DELETE	XDB	XDB$CONFIG
8	XDB	XDBCONFIG_VALIDATE	BEFORE EACH ROW	INSERT OR UPDATE	XDB	XDB$CONFIG
9	MDSYS	SDO_GEOM_TRIG_INS1	INSTEAD OF	INSERT	MDSYS	USER_SDO_GEO
10	MDSYS	SDO_GEOM_TRIG_DEL1	INSTEAD OF	DELETE	MDSYS	USER_SDO_GEO
11	MDSYS	SDO_GEOM_TRIG_UPD1	INSTEAD OF	UPDATE	MDSYS	USER_SDO_GEO
12	MDSYS	SDO_LRS_TRIG_INS	INSTEAD OF	INSERT	MDSYS	USER_SDO_LRS
13	MDSYS	SDO_LRS_TRIG_DEL	INSTEAD OF	DELETE	MDSYS	USER_SDO_LRS
14	MDSYS	SDO_LRS_TRIG_UPD	INSTEAD OF	UPDATE	MDSYS	USER_SDO_LRS
15	MDSYS	SDO_TOPO_TRIG_INS1	INSTEAD OF	INSERT	MDSYS	SDO_TOPO_TRA
16	MDSYS	OGIS_CRS_INSERT_TRIGGER	BEFORE EACH ROW	INSERT	MDSYS	OGIS_SPATIAL
17	MDSYS	OGIS_CRS_DELETE_TRIGGER	BEFORE EACH ROW	DELETE	MDSYS	OGIS_SPATIAL
18	MDSYS	SDO_UNITS_OF_MEASURE_TRIGGER	AFTER STATEMENT	INSERT OR UPDATE	MDSYS	SDO_UNITS_OF
19	MDSYS	SDO_COORD_REF_SRID_INSERT	BEFORE EACH ROW	INSERT	MDSYS	SDO_COORD_RE

2.

-- File Name : monitoring/user_objects.sql

-- Author : Ananya Shrivastava & Naveen Ponnaganti

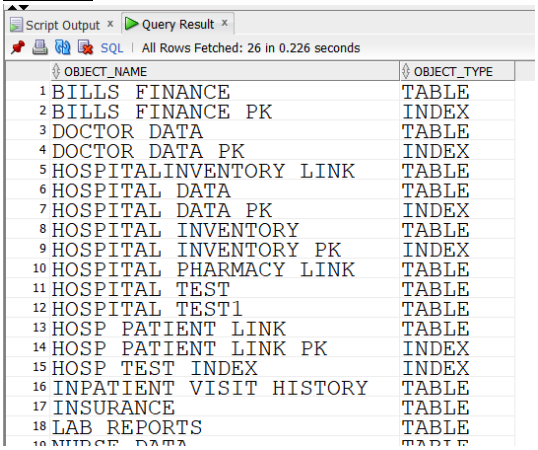
-- Description : Displays the objects owned by the current user.

SELECT object_name, object_type FROM user_objects

ORDER BY 1, 2;

Description: The above sql which access user_objects and displays object name and object type for current user.

Result:


Script Output | Query Result | SQL | All Rows Fetched: 26 in 0.226 seconds

	OBJECT_NAME	OBJECT_TYPE
1	BILLS_FINANCE	TABLE
2	BILLS_FINANCE_PK	INDEX
3	DOCTOR_DATA	TABLE
4	DOCTOR_DATA_PK	INDEX
5	HOSPITALINVENTORY_LINK	TABLE
6	HOSPITAL_DATA	TABLE
7	HOSPITAL_DATA_PK	INDEX
8	HOSPITAL_INVENTORY	TABLE
9	HOSPITAL_INVENTORY_PK	INDEX
10	HOSPITAL_PHARMACY_LINK	TABLE
11	HOSPITAL_TEST	TABLE
12	HOSPITAL_TEST1	TABLE
13	HOSP_PATIENT_LINK	TABLE
14	HOSP_PATIENT_LINK_PK	INDEX
15	HOSP_TEST_INDEX	INDEX
16	INPATIENT_VISIT_HISTORY	TABLE
17	INSURANCE	TABLE
18	LAB_REPORTS	TABLE

3.

-- File Name : dba/monitoring/system_privs.sql

-- Author : Ananya Shrivastava

-- Description : Displays users granted the specified system privilege.

-- Requirements : Access to the DBA views.

-- Call Syntax : @system_privs ("sys-priv")

SELECT privilege, grantee, admin_option

FROM dba_sys_privs

WHERE privilege LIKE UPPER('%&1%') ORDER BY privilege, grantee;

Description: The above sql which access dba_sys_privs and displays the users granted the specified system privilege.

Result:

Script Output × Query Result ×

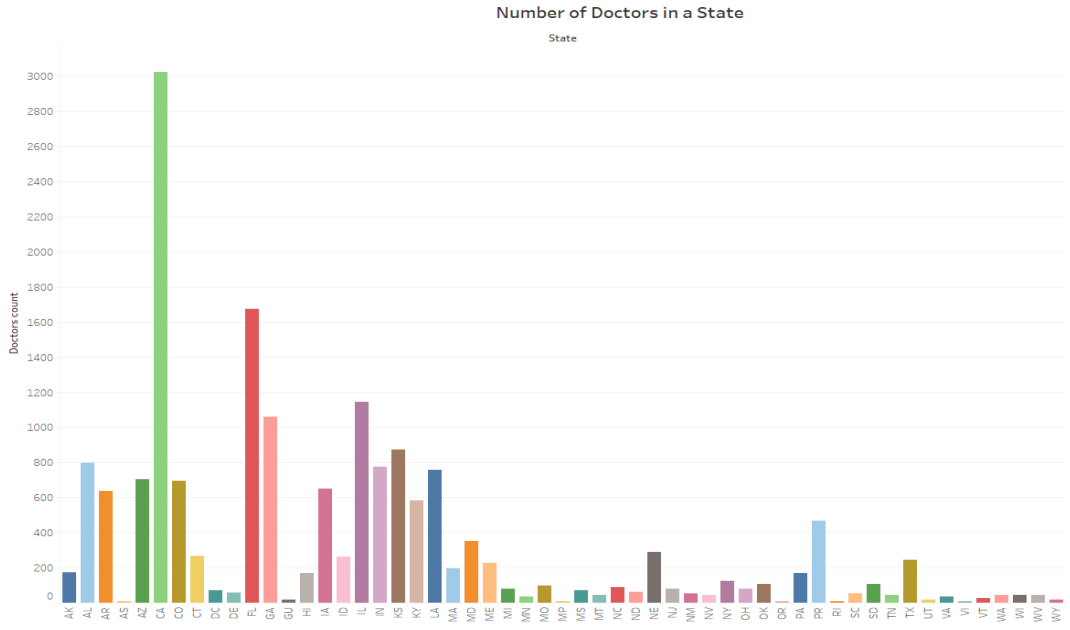
SQL | Fetched 100 rows in 0.104 seconds

	PRIVILEGE	GRANTEE	ADMIN_OPTION
1	ADMINISTER ANY SQL TUNING SET	DBA	NO
2	ADMINISTER ANY SQL TUNING SET	EM EXPRESS ALL	NO
3	ADMINISTER ANY SQL TUNING SET	SYS	NO
4	ADMINISTER DATABASE TRIGGER	DBA	NO
5	ADMINISTER DATABASE TRIGGER	DVSYS	NO
6	ADMINISTER DATABASE TRIGGER	DV OWNER	NO
7	ADMINISTER DATABASE TRIGGER	IMP FULL DATABASE	NO
8	ADMINISTER DATABASE TRIGGER	LBACSYS	NO
9	ADMINISTER DATABASE TRIGGER	MDSYS	NO
10	ADMINISTER DATABASE TRIGGER	RECOVERY CATALOG OWNER	NO
11	ADMINISTER DATABASE TRIGGER	SYS	NO
12	ADMINISTER DATABASE TRIGGER	WMSYS	NO
13	ADMINISTER KEY MANAGEMENT	SYSKM	YES
14	ADMINISTER RESOURCE MANAGER	APPQOSSYS	NO
15	ADMINISTER RESOURCE MANAGER	DBA	NO
16	ADMINISTER RESOURCE MANAGER	EXP FULL DATABASE	NO
17	ADMINISTER RESOURCE MANAGER	IMP FULL DATABASE	NO
18	ADMINISTER RESOURCE MANAGER	SYS	NO
19	ADMINISTER SQL MANAGEMENT OBJECT	DBA	NO
20	ADMINISTER SQL MANAGEMENT OBJECT	EM EXPRESS ALL	NO

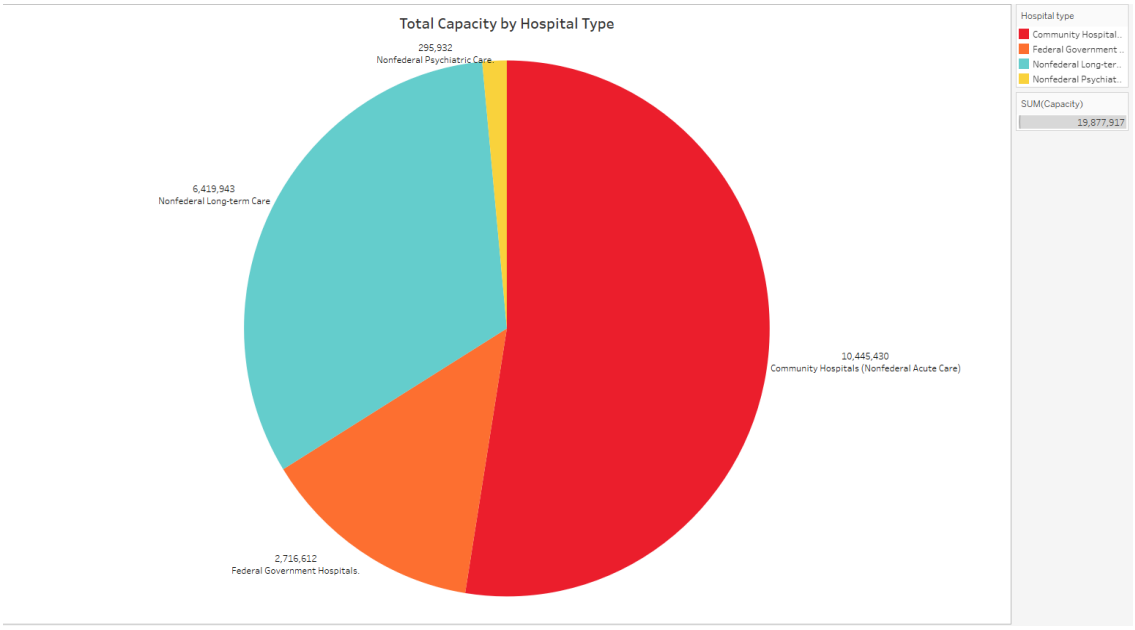
DATA VISUALIZATION

Using Tableau, we have created the following insightful Data Visualizations for our hospital database in order to show the metrics and numbers in a beautiful and clear way.

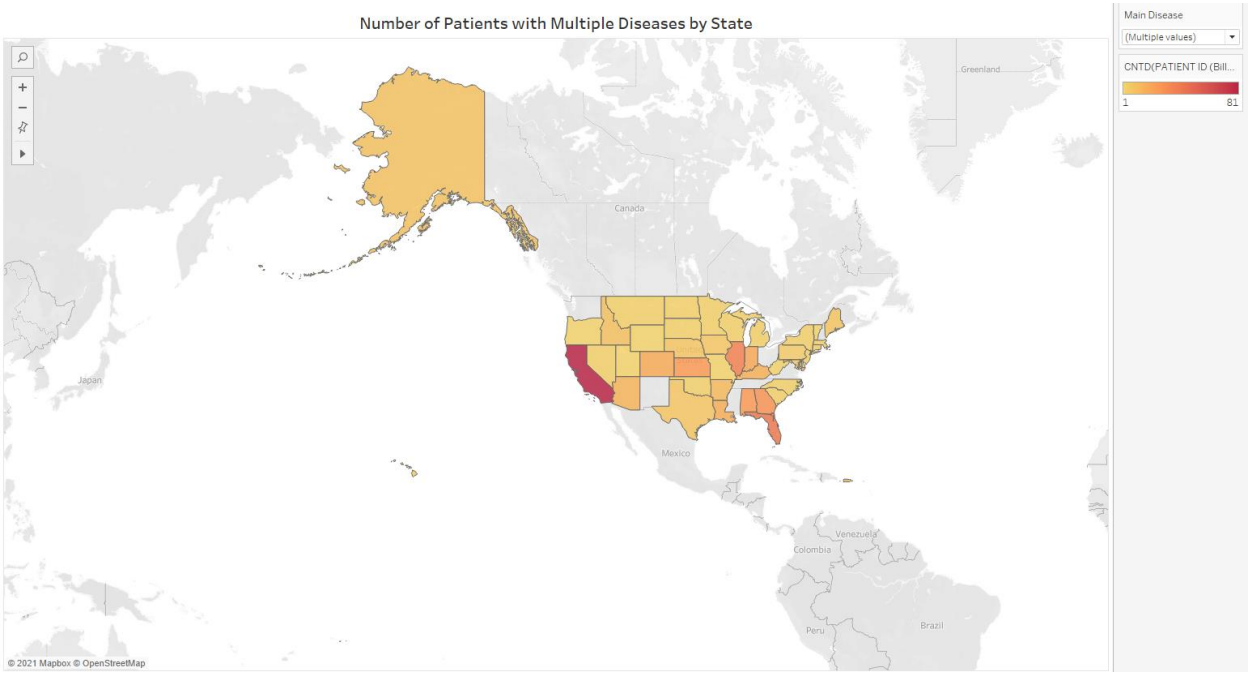
- The numbers of active doctors in a particular state in the US. We clearly see that California and Florida lead the way.



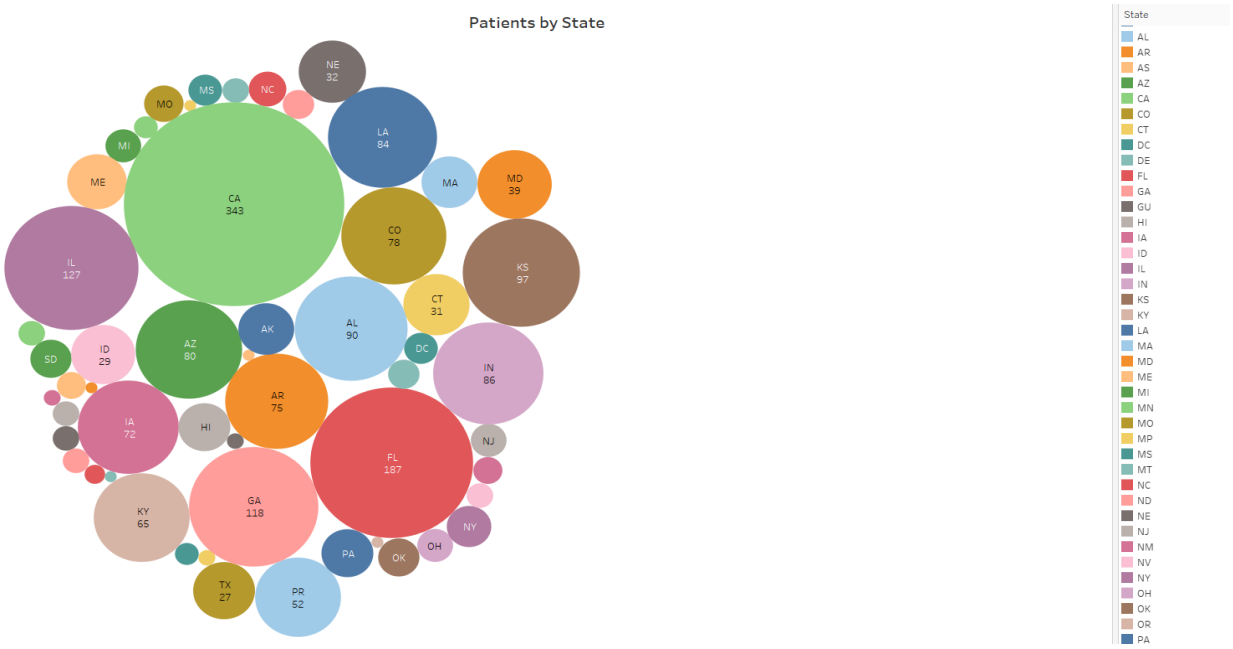
- The total available capacity of the hospitals grouped based on the hospital type countrywide.

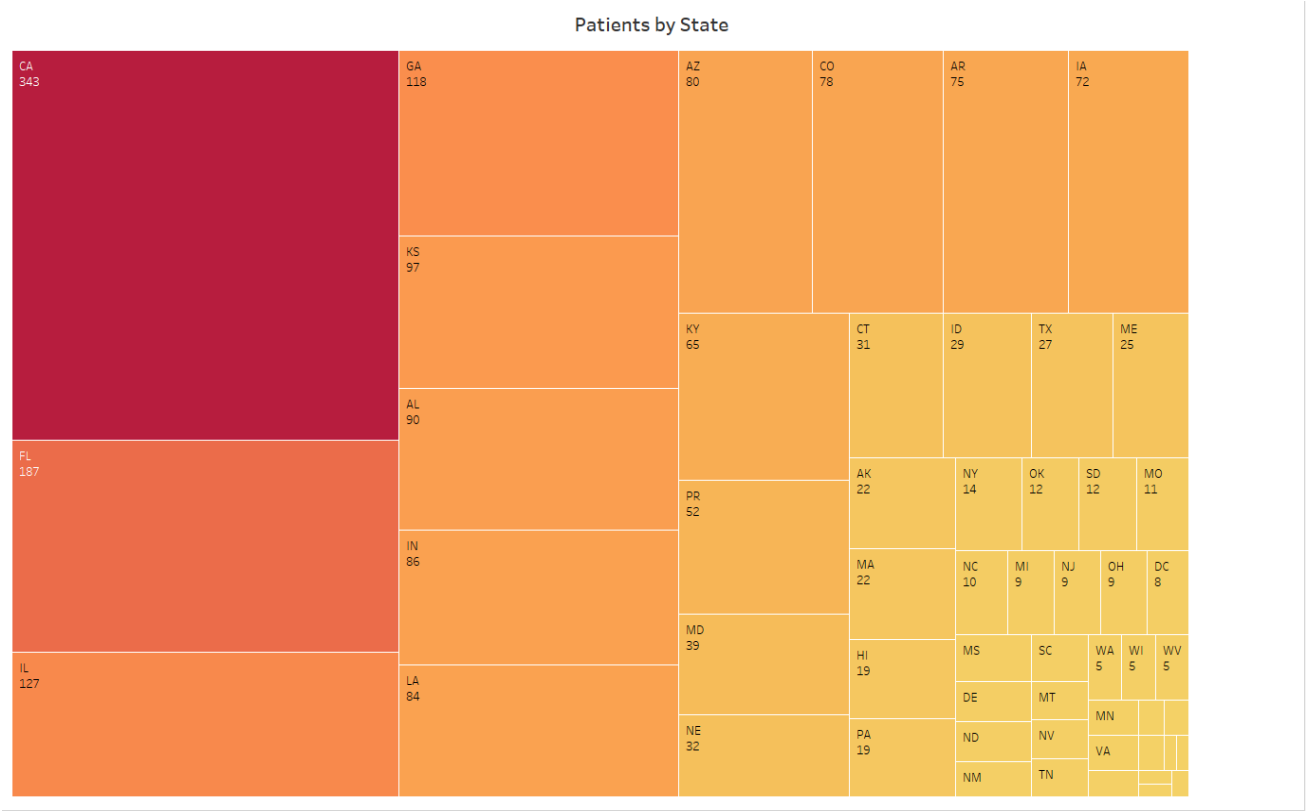


- A heatmap of the number of patients with multiple diseases by state in the US.

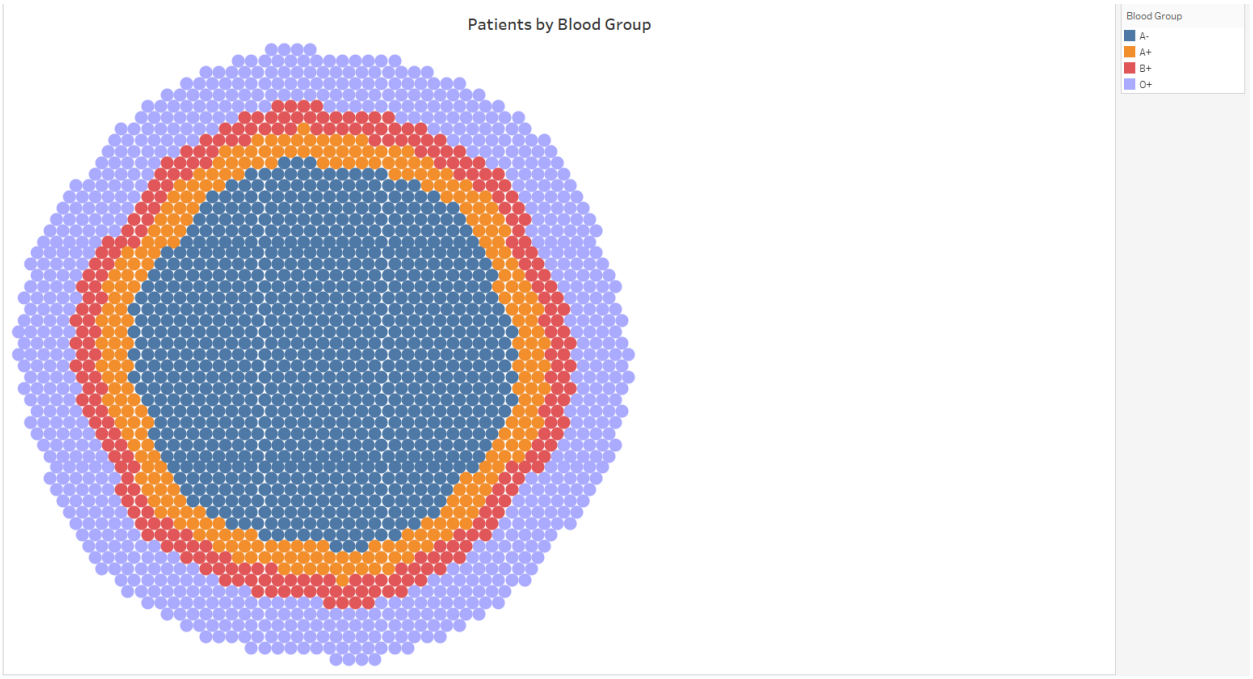


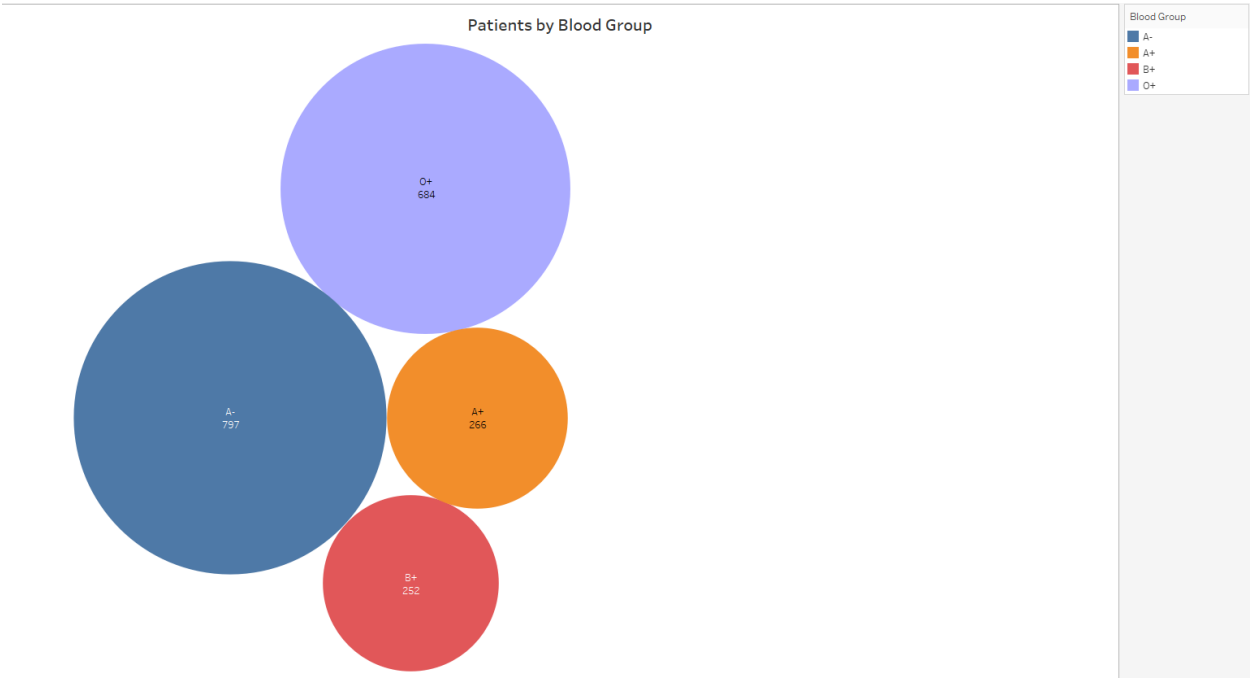
- The below two charts present a clear picture of the number of patients in our database at a state level.



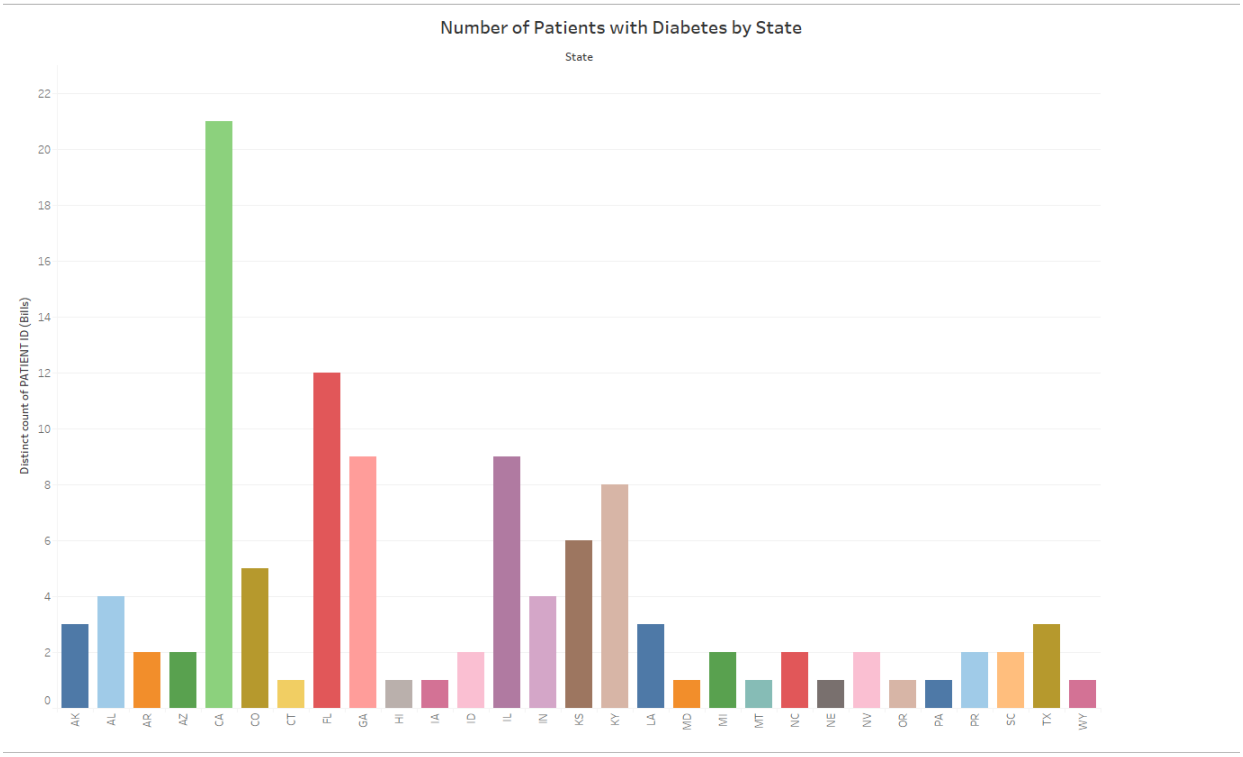


- The below two visualizations show the number of patients sorted on the basis of their blood group.





- Diabetes is one of the most widespread diseases in the US. The below graphic presents the number of patients with diabetes at a state level.



- The total inventory cost for each hospital calculated based on the inventory volume and the unit price for the material.