**A Comprehensive Hospital and Patient Database Management System**

|  |  |  |
| --- | --- | --- |
| **Palak Tater**  **Muma College of Business**  **University of South Florida**  **Tampa, FL, USA**  [**palaktater@mail.usf.edu**](mailto:palaktater@mail.usf.edu) | **Shruti Sridharan**  **Muma College of Business**  **University of South Florida**  **Tampa, FL, USA**  [**shrutis@mail.usf.edu**](mailto:palaktater@mail.usf.edu) | **Varsha Sharma**  **Muma College of Business**  **University of South Florida**  **Tampa, FL, USA**  [**varshasharma@mail.usf.edu**](mailto:varshasharma@mail.usf.edu) |

1. **Abstract**

Technology is becoming increasingly important in the health care industry, as recent technological advancements have had positive impact on health care administrations, treatments and more. As technology brings better treatment options for patients and more efficiency to health care facilities of all sizes, IT jobs in the health care industry are booming. Similarly, IT managers will be carrying more responsibilities in the health care industry; not only will it be required of IT professionals to know and understand the latest technological advancements, IT management will also be tasked with making key business decisions as they apply to technology. Health care facilities have begun using health management analytics to prevent readmissions, and this will continue at a growing pace. Health care management professionals have come to the realization that there is a strong connection between population health management and readmissions reduction. As a result, health care providers are focusing on avertable readmissions. By keeping the number of readmissions down, costs and patient waiting times are reduced, while doctor and health care professional time is freed to care for additional patients.

1. **Keywords**

*Database system, hospital management, entity-relationship, relational model, SQL based queries*

1. **Introduction**

Hospitals are an essential part of our lives, providing best medical facilities to people suffering from various ailments which may be due to change in climatic conditions, increased workloads, emotional trauma- stress etc. It is necessary for the hospitals to keep track of its day to day activities and records of its patients, doctors, nurses.

Currently, different departments in the healthcare center have their own separated systems leading to the lack of communications and the inefficient data sharing. For example, the finance department uses simple EXCEL spreadsheets to record the paycheck information of the employees which is inconvenient to retrieve and update employees’ information; in the clinic department, the doctors have to write down the prescriptions for the patients and keep paper documents, and also do not have any information about the patients’ insurance plans; the medicine department has to keep the prescription and inventory records on their own computer system. Keeping track of all the activities and their records on paper is very cumbersome and also an error. Prone process. While each system serves a distinctive purpose, there is no coordinating, assimilating and representing of data. The systems may have duplicate data which is a waste of space. The different systems also may have different application programs which cause incompatible files.

Due to these disadvantages of the current system, a healthcare management system is proposed. Healthcare management system is a database management system (DBMS), which is based on computer networks, using the advanced database technology to construct, maintain, and manipulate various kinds of data in a database system (DBS). The main aim of this project is to provide a paperless hospital and provide excellent data security at every level of user system interaction considering healthcare data being extremely confidential.

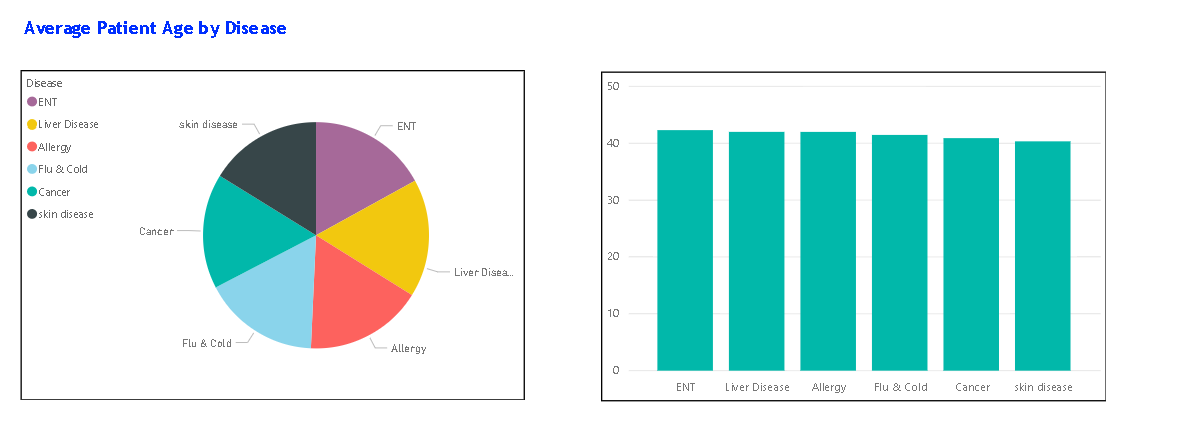
1. **Extraction of Information from Hospital management**

We created a database which consists of 6 tables namely*, Doctor\_table, Patient\_table, Insurance\_table, Paycheck\_table, Prescription\_table, Diagnosis ID.*

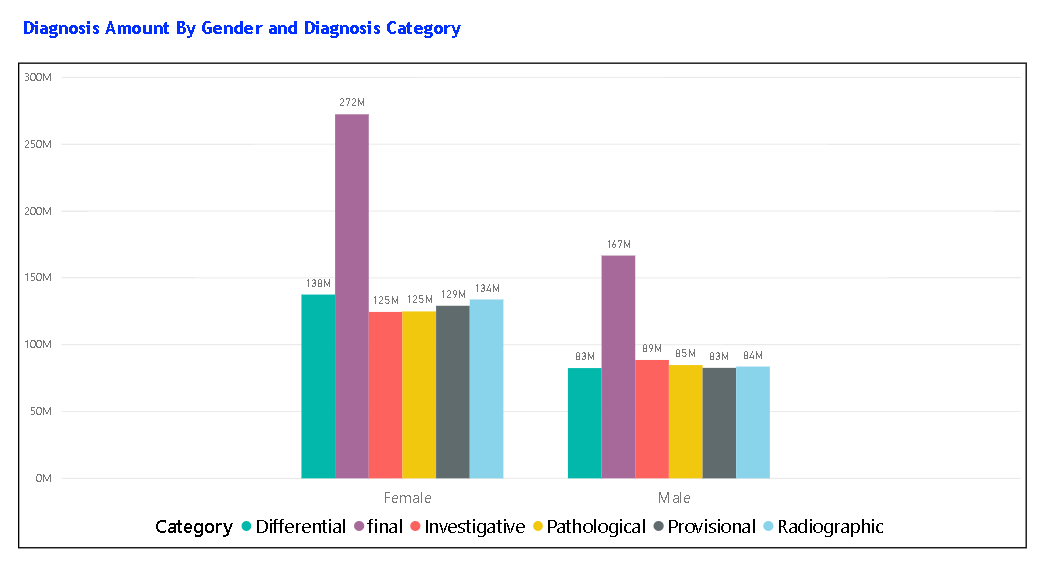
This database has been created using <https://www.onlinedatagenerator.com/>

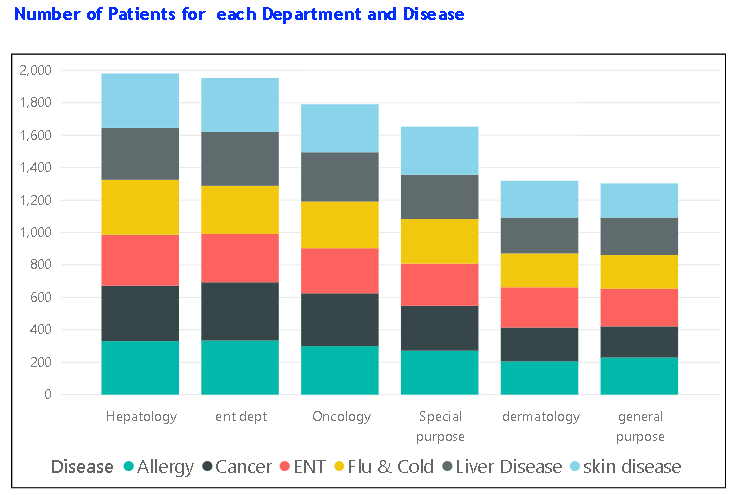
Online Data Generator is a free tool meant to help developers and testers to generate test data for software application.

Let us look at the visualization of our dataset to infer a few things. We have used Power BI to analyze the data

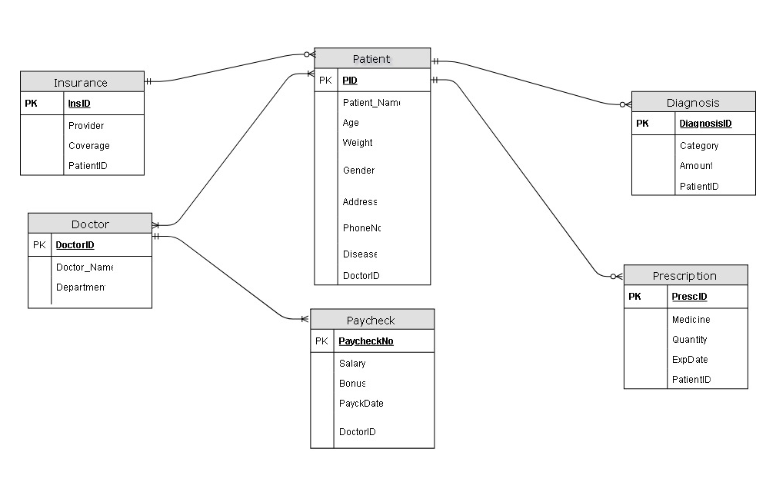


In our dataset, we have used the rage of age from 0 to 50 years of age.





1. **Database design**



Assumptions made in the hospital and Patient Management system:

* The number of doctors available in the hospital are restricted to 100
* Every patient has been given a prescription
* Every patient is assumed to have insurance from a listed insurance provider.
* Hospital accepts insurance from 9 insurance provider companies which are UnitedHealthcare, WellPoint Inc. Group, Kaiser foundation, Humana, Aetna, HCSC, Cigna, Highmark and Coventry
* There were 7 types of diagnosis performed in the hospital. They were Provisional, Differential, Investigative, Pathological, Radiographic and Final Diagnosis.
* The salary ranges of doctors range from $200000 to $400000.
* The Bonus of doctors range from $0 - $20000. We have not included the criteria for allotting bonuses to doctors.
* The expiry date of all medicines prescribed range between Dec1,2018 to Dec31,2018.
* The seven departments that the doctors belonged to were from the following list: hepatology, Oncology, Dermatology, Special Purpose, ENT dept, General Purpose.

The above diagram is the design we designed using Draw.io tool. The design that we obtained from this tool was based on the assumptions that is explained above. All the possibilities and exceptions aren’t handled in the data that are present in the database. But we took care of all the possible relationships that could ever occur and represented it above in the Entity

### **5.1 Capacity Planning:**

Capacity Planning involves planning of the resources as per the business requirements, resources as in not just the human resources but memory, CPU, storage space etc. It also involves predicting the future business needs and carefully planning.

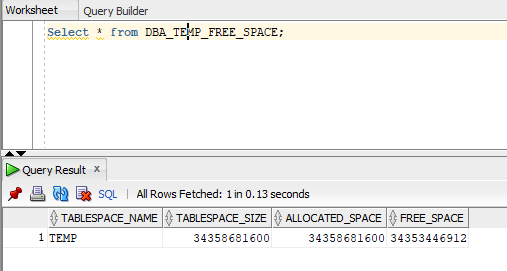


Table PATIENT, DOCTOR is expected to grow in size on a monthly basis. The number of records in it currently is around 10000. As the hospital expands, the number pf doctors and patients are going to be increasing. Hence the concept of range partitioning is to be done with relevant indexes. In our case, the data obtained had very minimal records and hence we did not do a partition on this table. This is one of the anticipated effects and could help improve the performance of the retrieval.

* 1. **Entity Relationship Model (ERM)**

The ER diagram can view the entities- Doctor, Patient, Insurance, Paycheck, Prescription and Diagnosis. Among these entities, relationships exist which connect all the entities in the diagram using primary and foreign key. We also investigate of size of each element and found the string length and defined the data types which otherwise would have significantly consumed lot of space give the number of entities the data possessed. We also had to deal with data duplication and this was successfully countered with Microsoft Excel using functions like VLOOKUP, PIVOT Tables etc., The data was converted to single excel file for every table and employed the direct method of Oracle SQL to upload data into specific tables.

The excel files that we have included are attached below:



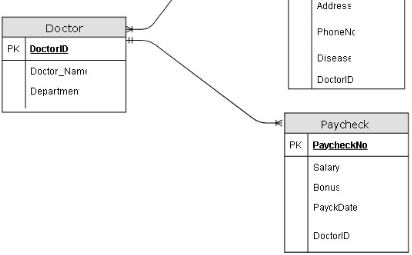
Tables to connect the database we created as follows,

|  |  |  |  |
| --- | --- | --- | --- |
| **Patient Table**  Primary Key **Patient ID**  Colum Name **PID**  Foreign Key **DoctID**  Column Name **DOCTORID** |  |  |  |
| **Doctor Table** |  |  |  |
| **PayCheck Table**  Primary Key **PAYCHKNO**  Colum Name **PAYCHECKNO**  Foreign Key **DID**  Column Name **DOCTORID** |  |  |  |
| **Prescription Table**  Primary Key **PrescID**  Colum Name **PRESCID**  Foreign Key **PatID**  Column Name **PATIENTID** |  |  |  |
| **Insurance Table**  Primary Key **InsID**  Colum Name **INSID**  Foreign Key **PID1**  Column Name **PATIENTID** |  |  |  |
| **Diagnosis Table**  Primary Key Diag **ID**  Colum Name **DIAGNOSISID**  Foreign Key **PID2**  Column Name **PATIENTID** |  |  |  |

* 1. **Cardinalities**
     1. **Binary Relationships**

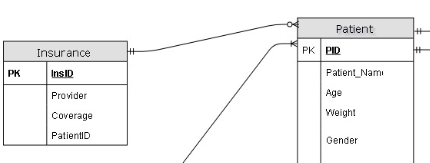
**Doctor-Paycheck** - One to Many Mandatory

* each doctor must have one or more Paychecks



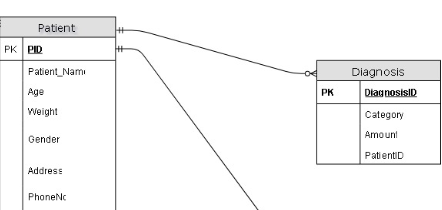
**Patient – Insurance**- One Mandatory to Many optional

* each Patient must have one Insurance coverage
* one Insurance provider can have more than one patient



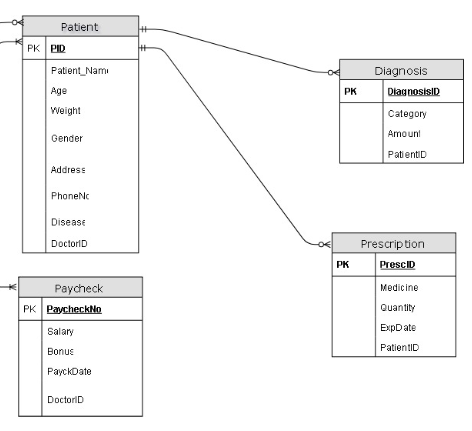
**Patient-Diagnosis** - One Mandatory to Many Optional

* Each Patient can have one or more Diagnosis
* Each Diagnosis ID must have one Patient



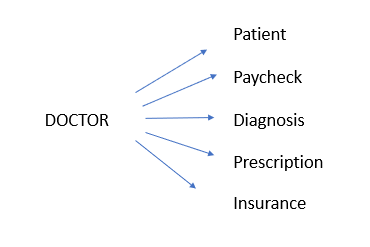
**Patient-Prescription** - One Mandatory to Many Optional

* Each Patient can have one or more Prescriptions
* Each Prescription ID must have one Patient



* + 1. **Ternary Relationships**

**Doctor-Patient** - Many to Many Mandatory



* 1. **Relational Model**

A **relational database** is a collection of tables. – Each table has a unique name. – Each table consists of multiple rows. – Each row is a set of values that by definition are related to each other in some way; these values conform to the attributes or columns of the table.

The entire ER diagram can be converted to a Relational Model (relational tables). The attribute(s) of a relation which serve as a primary key of the table are underlined. Those attribute(s) which represent foreign keys are indicated as fk underneath.

* ***Doctor\_table*** *(DoctorID, Doctorname, Department)*

* ***Patient\_table*** *(Patient ID, PatientName, PatientAge, Weight, Gender, Address, PhoneNum, Disease, DoctorID)*

***Pk Fk***

* ***Insurance\_table*** *(InsuranceID, Insuranceprovider, Coverage (dollars), Patient ID)*

***Pk Fk***

* ***Paycheck\_table*** *(PaycheckNo, salary, Bonus, PaycheckDate, DoctorID)*

***Pk Fk***

* ***Prescription\_table*** *(PresciptionID, Quantity, Expiry Date, Medicine, Patient ID)*

***Pk Fk***

* ***Diagnosis ID*** *(Diagnosis ID, Amount, Category, Patient ID)*

***Pk Fk***

* 1. **Implementation in SQL server**

SQL server is a modern software where we can store huge amount of information via a database. In fact, we have implemented our database system in SQL server. In this server we can store data easily, retrieve data speedily and execute the queries conveniently by SQL query language. We can create relations easily by code and manually (drag and drop method like MS. Access).

1. **Usage of the System**

We can easily retrieve various data based on our demands using SQL Queries. Sample data has been entered in the relational tables. Some useful queries that can be imposed on our designed system and can be retrieved that speedily from the existing manual database

* 1. **Query 1**

**Display Patient Name, Patient Age, Patient Gender, Patient Disease, Doctor Name and Diagnosis Category**

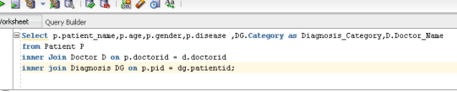
Select p.patient\_name,p.age,p.gender,p.disease,

DG.Category as Diagnosis\_Category,D.Doctor\_Name

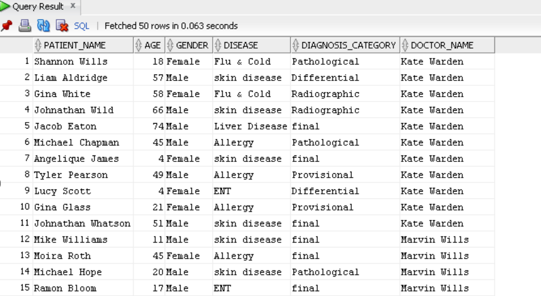
from Patient P

inner Join Doctor D on p.doctorid = d.doctorid

inner join Diagnosis DG on p.pid = dg.patientid;



**OUTPUT:**



* 1. **Query 2**

Display each Patient’s name along with the minimum, maximum, and average Diagnosis Amount. Exclude any patient who has diagnosed for one time only. Show the Patient with the highest Diagnosis Amount first.

select patient.patient\_name,

sum(diagnosis.amount)as DIAG\_sum,

min(diagnosis.amount) as minimum\_DIAG,

max(diagnosis.amount) as maximum\_DIAG,

avg(diagnosis.amount) as average\_DIAG

from patient join diagnosis

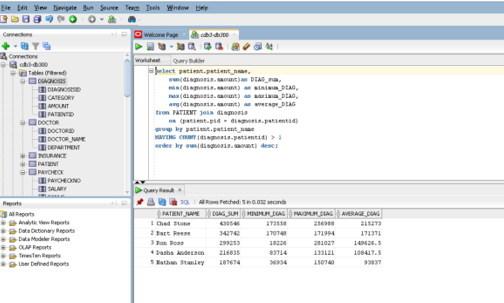
on (patient.pid = diagnosis.patientid)

group by patient.patient\_name

HAVING COUNT (diagnosis.patientid) > 1

order by sum(diagnosis.amount) desc;

**OUTPUT:**



**Query 3**

Find all Insurance provider taken by more than 1000 patients.

select insurance.provider, COUNT(Patient.pid) as Total\_Patients

from insurance join patient

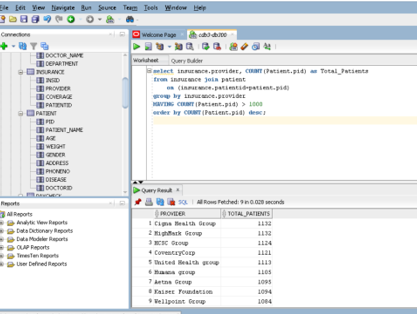
on (insurance.patientid=patient.pid)

group by insurance.provider

HAVING COUNT(Patient.pid) > 1000

order by COUNT(Patient.pid) desc;

**OUTPUT:**



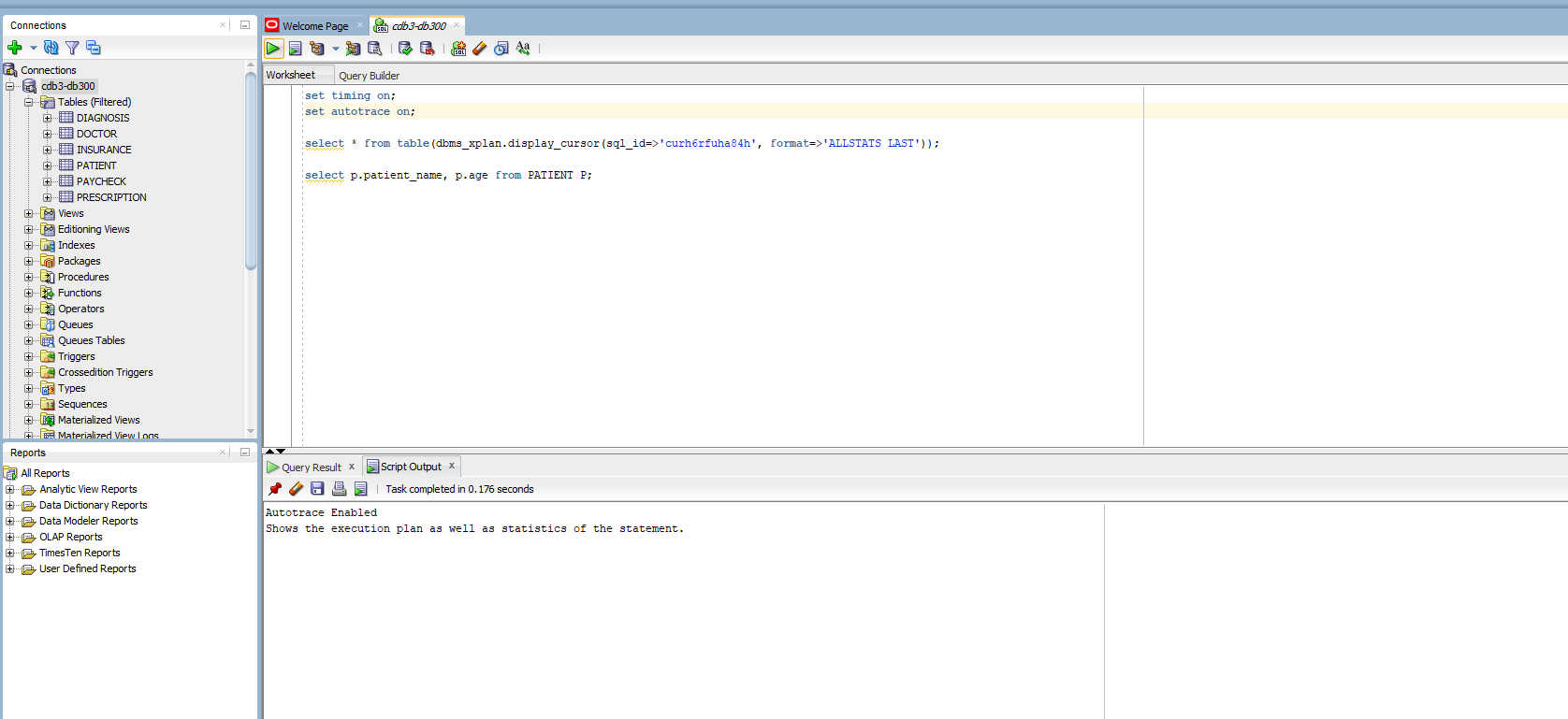
1. **Future Scope:**

**The following are the tasks we are working on to make this database a better version:**

* **Parallelism**

The amount of data stored in databases has been growing exponentially over the recent past and thus there is a need for faster processing of data to meet business requirements. Parallelism is a key component for large scale data processing. Parallel execution uses multiple processes to accomplish a single task. The more effectively the database can leverage all hardware resources - multiple CPUs, multiple IO channels, multiple storage units, and multiple nodes in a cluster - the more efficiently queries and other database operations will be processed. We will be considering a table which has around 10,000 records to test the concept of parallelism and the performance associated with it.

We ran a query to select first and last names and examined the Execution Plan.



Furthermore, we will increase the degree of parallelism and see how the execution plan works.

* **Performance Tuning:**

Performance tuning refers to a set of activities for optimizing and homogenizing the performance of a database. It is the design of the database files, selection of DBMS application and configuration of the database environment. It maximizes the use of system resources to perform work as efficiently and rapidly as possible. In order to identify and correct performance problems; effective data collection and analysis is very critical. There are several tools which are used for gathering information regarding monitoring database performance, diagnosing issues and tuning applications.

We will perform indexing operations on our database to see how the queries work.

### **Database Scripting:**

### Database Administration deals with the functioning and the maintenance of the Database Management Systems. These are generally carried out through professionals called as Database Administrators (DBA). The responsibilities of a DBA include: Performance monitoring, Database tuning, Implementing and Maintaining Database Security, Evaluation of the Database features, Database Recovery, File storage management, Capacity Planning etc. This section will discuss a few DBA scripts which helps the DBA in performing some of the above activities.

Some of the sample DBA scripts that we intend on running are:

* Script displaying all the active sessions of the database.
* Scripts to retrieve all system parameters and their descriptions.
* Script to see the background processes running currently.
* To check if any time-consuming queries are running on the database.

1. **Conclusion**

Our database contains all the information needed to be maintained in a hospital. As we have computerized the entire system via a database, the maintenance is very convenient and efficient and retrieval of data according to demand is speedy. Our designed system is a good and useful implementation. We can further improvise it by enhancing its security. An initiative has also been taken to use Microsoft Visual Studio 2008 and the programming language C# for developing user friendly interfaces to the current database system. That way a software has been developed which is used to interface with the SQL Server and hence data accessed, retrieved and searched for far better in a more efficient and convenient form.