**Project Introduction**

***Create a Database for a Taekwondo club***

“**Inquisitive**” is performing a project to automate the necessary operations of ***Kawang Lee Taekwondo School*** (**KLTS**). The offered system is named **Taekwondo Club Enrollment Process**. At present all the operations like student enroll, training, payment, test and tournament at KLTS is done by manually (**file-based system**). The Taekwondo school wishes to substitute **70%** of the manual filling process with a compact computerized database system that will receive records, develop customer services, alleviate operational expenses as well as time and enhance customer allowances. The school has estimated **$50,000** for developing this project and it will take almost six months to establish. The project team “**Inquisitive**” is supervised by Project Manager Mr. **Nafees** from ***B-Tech Computers*** **Sdn Bhd**. Team Inquisitive is consisted with a project manager, one system analyst, one designer and one programmer. B-Tech Computers is integrated with such number of teams like Inquisitive and has been in system development project for the past two decades. That is why our team is so reliable to all.

**Organizational chart for team**

**“Inquisitive”**

**Project Manager**

**Mohammad Nafees**

**Programmer**

**An Nafi**

**Designer**

**Rifath Riaz**

**System Analyst**

**Faisal Khan**

Business Transactions

1. Enroll Student.
2. Register Staff.
3. Register Tournament.
4. Accept Payment.
5. Print Student Invoice.

|  |
| --- |
| ***PAYMENT*** |
| PaymentID {PK}  PaymentType  Amount  studentID {FK}  branch\_no {FK} |

**Conceptual Model**

|  |
| --- |
| ***BRANCH*** |
| Branch No.  Address  City  Street  Zip Code |

|  |
| --- |
| ***STAFF*** |
| Staff ID  Fname  Lname  Sex  Position  Branch\_no {FK} |

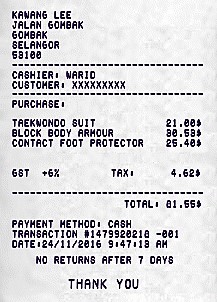
|  |
| --- |
| ***STUDENT*** |
| Student ID {PK}  Fname  Lname  Sex  Age  City  Branch\_no {FK}  staffID {FK} |

|  |
| --- |
| ***LESSON*** |
| CourseID {PK}  staffID {FK}  StudentID {FK} |

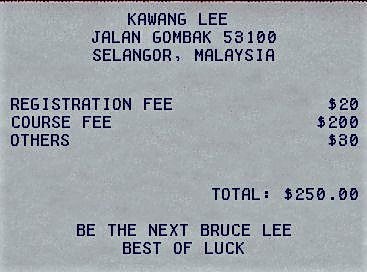
|  |
| --- |
| ***TOURNAMENT*** |
| TournamentID  Entery\_Fee  StudentID {FK} |

**Report Design**

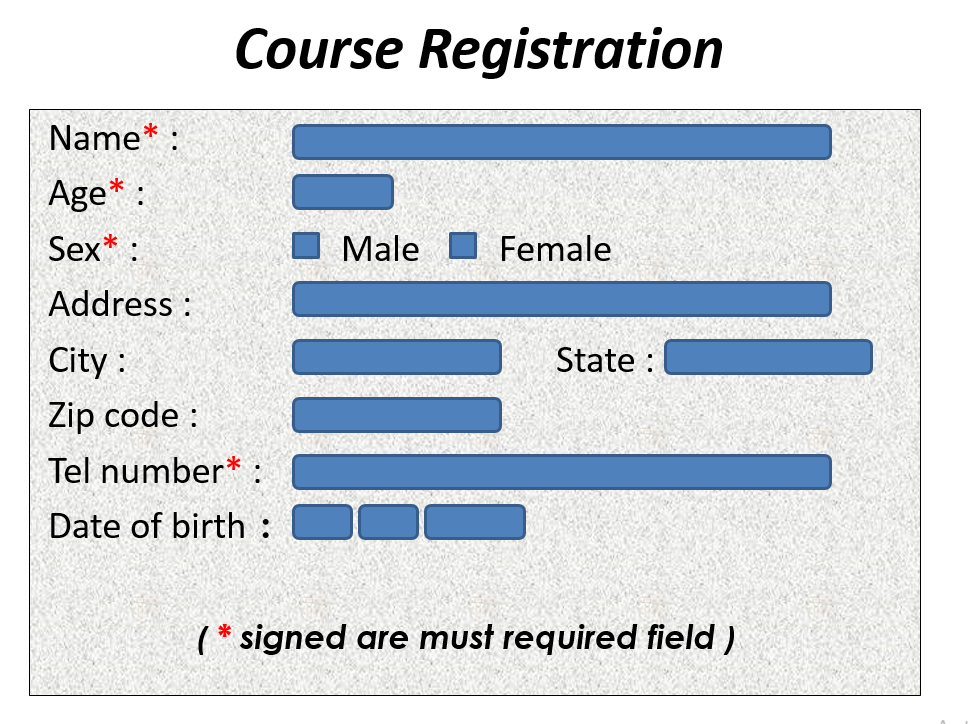
Customer Receipt

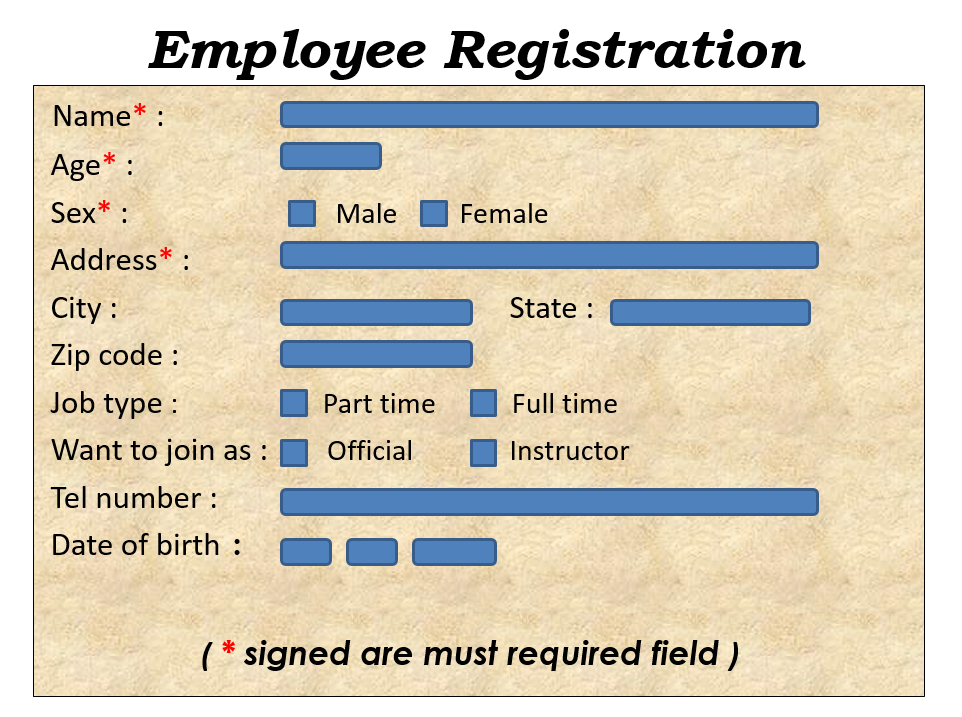


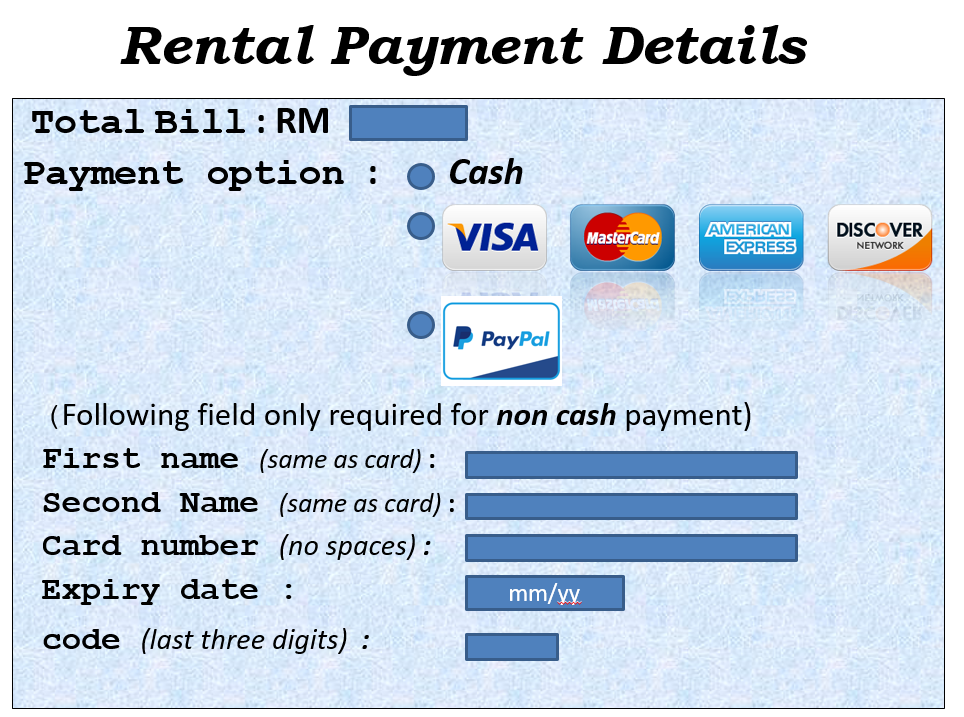
Student Receipt

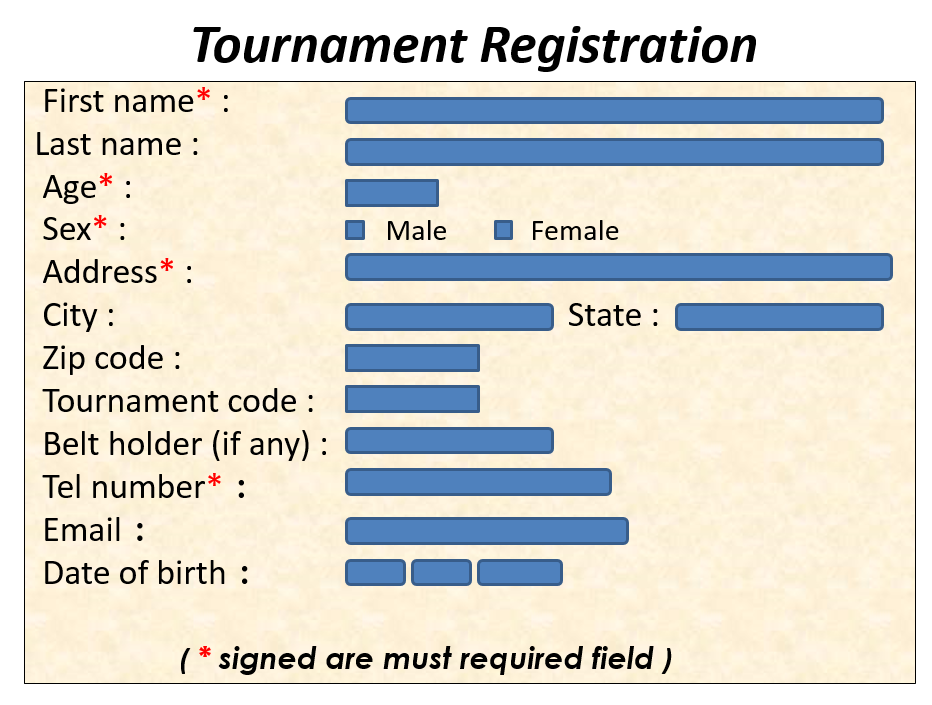


**Screen design**









**Logical Database Design**

**Branch** (Branch\_no, city, street, zipcode)

PK Branch\_no.

**Staff** (StaffID, fname, lname, position, sex, salary, Branch\_no)

PK StaffID

FK Branch\_no reference Branch(Branch\_no).

**Student** (StudentID, name, sex, age, city, Branch\_no, StaffID)

PK StudentID

FK Branch\_no reference Branch(Branch\_no)

FK StaffID reference Staff(StaffID).

**Payment** (PaymentID, PaymentType, Amount, StudentID, Branch\_no,)

PK PaymentID

FK StudentID reference Student(StudentID)

FK Branch\_no reference Branch(Branch\_no)

**Lesson** (CourseID, StudentID, StaffID)

FK StudentID reference Student(StudentID)

FK StaffID reference Staff(StaffID)

**Tournament** (TournamentID, Entry\_fee, StudentID)

FK StudentID reference Student(StudentID)

**Physical Database Design**

**Create table branch**

(

branch\_no varchar (5) not null primary key,

city char (20) not null,

street varchar (25) not null,

zipcode numeric (5)

);

**Create table staff**

(

staffID varchar (5) not null primary key,

fname varchar (10) NOT NULL,

lname varchar (10) NOT NULL,

position varchar (10) NOT NULL,

sex char not null check (sex IN ('M','F')),

salary int not null,

branch\_no varchar (5) not null,

foreign key (branch\_no) references branch(branch\_no) );

**Create table Student**

(

StudentID varchar (8) NOT NULL PRIMARY KEY,

Name char (14) not null,

sex char not null check (sex IN ('M','F')),

age int not null,

city char (20) not null,

branch\_no varchar (5) not null,

staffID varchar (5) not null,

foreign key (branch\_no) references branch(branch\_no),

foreign key (staffID) references staff(staffID)

);

**Create table lesson**

(

courseID varchar (5) not null,

StudentID varchar (8) NOT NULL,

staffID varchar (5) not null,

foreign key (StudentID) references student(StudentID),

foreign key (staffID) references staff(staffID),

);

**Create table payment**

(

paymentID varchar (5) not null primary key,

paymentType varchar (20) not null,

amount int NOT NULL,

StudentID varchar (8) NOT NULL,

branch\_no varchar (5) not null,

foreign key (StudentID) references student(StudentID),

foreign key (branch\_no) references branch(branch\_no)

);

**create table tournament**

(

tournamentID varchar (5) not null,

Entry\_Fee int not null,

StudentID varchar (8) NULL,

foreign key (StudentID) references student(StudentID)

);

**Queries**

1. **Basic Queries**
2. Insert record for **Branch** which branch\_no is ‘B001’.

insert into Branch values ( 'B001' , 'Kuala Lumpur' , 'Jalan Bukit' , 50100 ) ;

1. Change **Staff** with staffID ‘S102’ salary 1200 to 1400.

Update staff

Set salary=1400

Where staffID='S102';

1. List staffID, position, salary from **Staff**.

select staffId, position, salary

from staff;

1. Produce a list of per day wage for all the staffs, showing the staff number, the first and last name and the salary details.

select staffId, fname, lname, salary/30 As Wage\_perDAY

from staff;

1. Delete all details that relate to Student number 530.

Delete from Student

Where StudentID = 530;

**Operational Queries**

1. How many tournaments have entry fee more than 100 Rm?

Select Count (Distinct tournamentID) As NumTournament

from tournament

where Entry\_Fee > 100;

1. Find total numbers of managers and the sum of their salaries.

select COUNT(staffID) As Total\_staff, SUM(salary) As Total\_salary from staff

where position='Manager';

1. Find the minimum, maximum and average staff salary.

select COUNT(staffID) As Total\_staff, SUM(salary) As Total\_salary from staff

where position='Manager';

1. List payment details arranged in descending order of amount.

select paymentID, paymentType, amount from payment

Order by amount DESC;

1. List all the details of students who are from Kuala Lumpur and Johor Bahru arranged in descending order of StudentID.

select \* from Student

where city IN ( 'Kuala Lumpur' , 'Johor Bahru' )

Order by StudentID DESC;

**Strategic Queries**

1. List studentID, name, age and the amount they paid.

Select s.studentID, name, age, amount

from Student s, payment p

Where s.StudentID = p.StudentID ;

1. List staff details and total number of courses teaches by staffs.

select s.staffID , fname, lname, sex, COUNT (\*) As Total\_Course

from Staff s, lesson l

Where s.staffID = l.staffID

Group by s.staffID, fname, lname, sex ;

1. Total salary given to staffs and details based on branch.

select b.branch\_no,city,zipcode, SUM(salary)As Given\_salary

from branch b, staff s

where b.branch\_no=s.branch\_no

Group by b.branch\_no,city,zipcode

Order by b.branch\_no DESC.

1. Total amount earned from students and details based on branch.

select b.branch\_no,city,zipcode, SUM(amount)As Earn\_amount

from branch b, payment p

where b.branch\_no=p.branch\_no

Group by b.branch\_no,city,zipcode

Order by b.branch\_no ASC;

1. Find all staff whose salary is larger than the salary of at least a staff at branch B004.

select staffID, fname, lname, sex, salary

from staff

where salary > some ( select salary

from staff

Where branch\_no='B004' ) ;

**User Views**

1. Top Manager View

:: Create a view of sum of salary of the staffs and sum of the payments by students per branch ::

create view Topmanager (branch\_no, Staff\_salary, student\_pay)

As Select s.branch\_no, SUM (s.salary), SUM (p.amount)

from staff s, payment p

Where s.branch\_no=p.branch\_no

Group by s.branch\_no;

1. Operational Management View

:: Create a view of total number of students and staffs per course ::

create view Courses

AS SELECT courseID, COUNT(StudentID) As Total\_Student, COUNT(staffID) As Total\_staff

From lesson

Group By courseID;

1. Visitors View

:: Create a view of the entry fee of the tournaments ::

create view Visitors

AS SELECT tournamentID, MAX(Entry\_Fee) as Entry\_Fee

From tournament

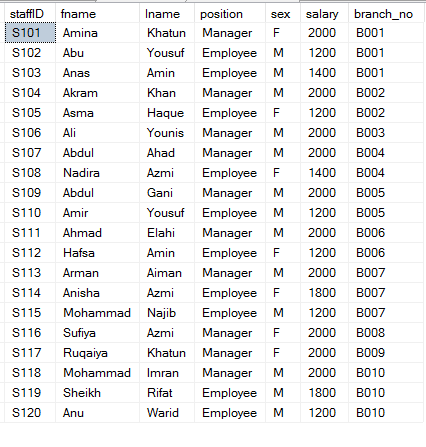
Group by tournamentID;

**Record Listing**

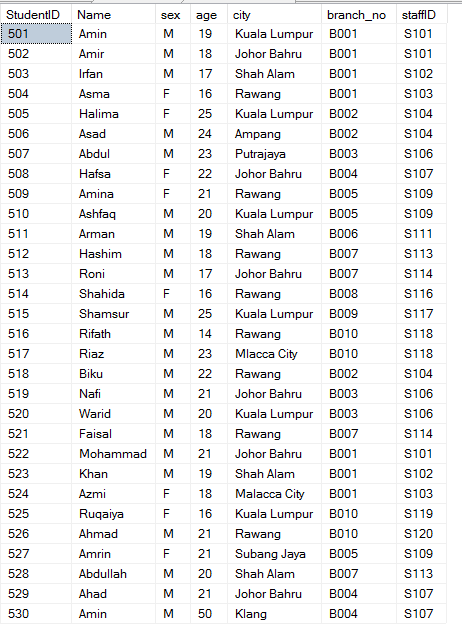
**Branch List**

****

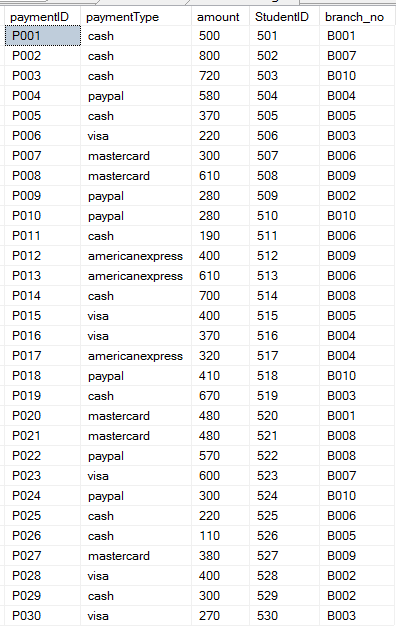
**Staff List**



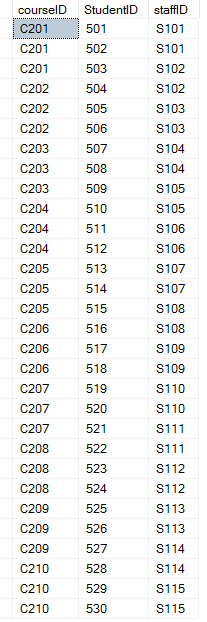
**Student List**



**Payment List**



**Lesson LIST**



**Tournament LIST**

