گزارش فاز اول پروژه پایگاه داده

سیستم مدیریت فروش بلیط سفر

حنانه صفرمحمدلو - 40219113 سپهر قارداشی - 40220143

طراحی نهایی ERD

جداول و موجودیت ها:

1. Users -> اطلاعات پایه کاربران سایت اعم از پشتیبانان و کاربران عادی
2. User_contacts -> دخیره تلفن/ایمیل هر کاربر

این موجودیت نسبت به user ضعیف در نظر گرفته شده و از خود شناسه ندارد.
ازین رو هر رکورد نمونه آن با شناسه user و نوع ایمیل یا تلفن شناسایی خواهد شد.
بدین ترتیب هر کاربر می تواند تنها یک تلفن و یک ایمیل داشته باشد و داشتن فقط یکی از آنها الزامی است.

3. Report -> اطلاعات گزارش ثبت شده توسط هر کاربر
4. Reservation -> هر رزرو ایجاد شده با وضعیت و زمان ساخت و انقضای آن
5,6. One-way and two-way reservation ->

این دو موجودیت زیرنوع reservation هستند و برای ذخیره اطلاعات یک سفر یا دو سفر رفت و برگشت تقسیم شده اند. شماره صندلی رزرو شده مربوط به هر تک سفر یا دو سفر نیز، اینجا ذخیره می شود.
فاقد شناسه مستقل اند و شناسه زبرنوع خود را به ارث برده اند.

7. Location_details -> جزئیات یک مکان مبدا یا مقصد که از نوع صفت مرکب میبود را به صورت یک موجودیت جداگانه نگه می دارد تا **فرم نرمال سوم** حفظ شود
8. Trip -> موجودیت نشانگر هر سفر با مبدا، مقصد و زمان حرکت و رسیدن

این موجودیت اطلاعات جامع هر سفر مانند تعداد توقف ها که در انواع بلیط های مختلف تفاوتی ندارند، ذخیره می کند.
همچنین ظرفیت موجود در یک وسیله نقلیه و ظرفیت پر شده آن تا کنون در این جدول قرار می گیرند.

9. Ticket -> موجودیتی ضعیف که از خود شناسه ندارد و با شناسه سفر مربوطه و گروه سنی مربوط به بلیط که در تعیین قیمت آن تاثیر می گذارد، شناخته می شود یعنی primary key آن ترکیبی از trip_id موجودیت trip و age است. زیرا سفر ها تنها با گروه سنی در بلیط ها از هم متمایز می شوند.
هر سفر می تواند چند نوع بلیط با قیمت های متفاوت به فروش برساند که این انواع وابسته به سن هستند.
برای مثال سفر با شناسه 1234 بلیط خردسال و بزرگسال به فروش می رساند.
شناسه این بلیط ها 1234خردسال و 1234 بزرگسال هستند.

10,11, 12. Flight, bus, train ->
زیرنوع های trip هستند که شناسه های trip های مختلف به عنوان شناسه آنها تقسیم شده و هر کدام کلاس سفر یا ویژگی منحصر به وسیله نقلیه سفر را نگه می دارند.

13. Additional_service ->
موجودیت ضعیف به trip که از خود شناسه ندارد و با شماره سفر و نوع سرویس موجود در آن، تشخیص پذیر می شود.
این جدول برای نگهداری امکانات هر سفر مانند اینترنت و پذیرایی تعبیه شده است.

14. Payment -> دارای اطلاعات مربوط به هر تراکنش چه موفق و چه ناموفق با هر رزرو و هر کاربر ارتباط الزامی دارد.

ارتباطات:

User	1:N	User_Contact
User	1:N	Payment
User	1:N	Reservation
User	1:N	Report
One_way_reservation	N:1	Ticket
Two_way_reservation	N:2	Ticket
Trip	1:3	Ticket
Trip	N:1	Location
Trip	1:N	Additional_Service

Enums:

14 تایپ خود ساخته enum در طراحی تعریف شد.

Indexing:

جهت بهبود سرعت جست و جو، صفت هایی که احتمال کوئری زدن روی آنها بیشتر بوده شناسایی و بر روی آنها ایندکس قرار داده شد.

از جمله

- تمام foreign key ها -> زیرا پیوسته در join جدول ها استفاده و جست و جو می شوند.

- ترکیب مبدا و مقصد، تاریخ حرکت سفر ها -> در جست و جو های کاربران بین سفر های موجود بسیار استفاده می شوند.
- نام و نام خانوادگی کاربر و نقش او -> در پیدا کردن اطلاعات سیستمی هر کاربر استفاده می شوند.
- وضعیت گزارش ها -> پیوسته توسط ادمین ها کوئری زده شده و گزارش های بررسی نشده خواسته می شود.

همچنین تمام بلیط ها بر اساس تاریخ حرکت در یک view مرتب شده اند تا به ترتیب به کاربران نمایش داده شوند.

Constraints:

- از constraint check های گوناگون برای بررسی فرمت صحیح ایمیل، تلفن و نام و نام خانوادگی ورودی کاربر استفاده شد.
- از unique constraint تنها برای یکتا بودن ایمیل و تلفن همراه هر کاربر استفاده شد.
- not null بودن اکثر صفات قرار داده شد تا ویژگی های ضروری هر موجودیت حین ایجاد آن تکمیل شوند.
- شرط مثبت بودن مبلغ بلیط قرار داده شد.
- شرط صحیح بودن تاریخ حرکت و تاریخ رسیدن به مبدا به طوری که اولی الزاما پیش از دومی بیاید قرار داده شد.
- مقادیر default برای موجودیت هایی که در ابتدا با مقدار معناداری باید پرشوند قرار داده شد. برای مثال همه بلیط ها در صورت تعیین نشدن، بلیط بزرگسال در نظر گرفته می شوند.

Queries:

از postgresql برای ایجاد جداول اولیه و اعمال محدودیت ها بر روی آنها، و همچنین تعیین شناسه های اصلی و فرعی استفاده شد.

Integrity and 3NF:

- ایجاد جداول user_contacts, locations, additional service در راستای نرمال سازی ER انجام شد. به طوری که در جداول نهایی هیچ ستونی به ستون های دیگر غیر از شناسه وابستگی نداشته باشد.
- برای موجودیت های وابسته مانند موجودیت های ضعیف یا آنهایی که foreign key شان ارتباط قوی با موجودیت دیگر ایجاد کرده، شرایط on delete cascade در نظر گرفته شد تا رکورد های مربوطه را به حضور هم وابسته کند.

  برای مثال، با پاک شدن یک trip از دیتابیس، همه additional service های مربوطه نیز باید پاکسازی شوند و دیگر نقشی در برنامه نخواهند داشت.