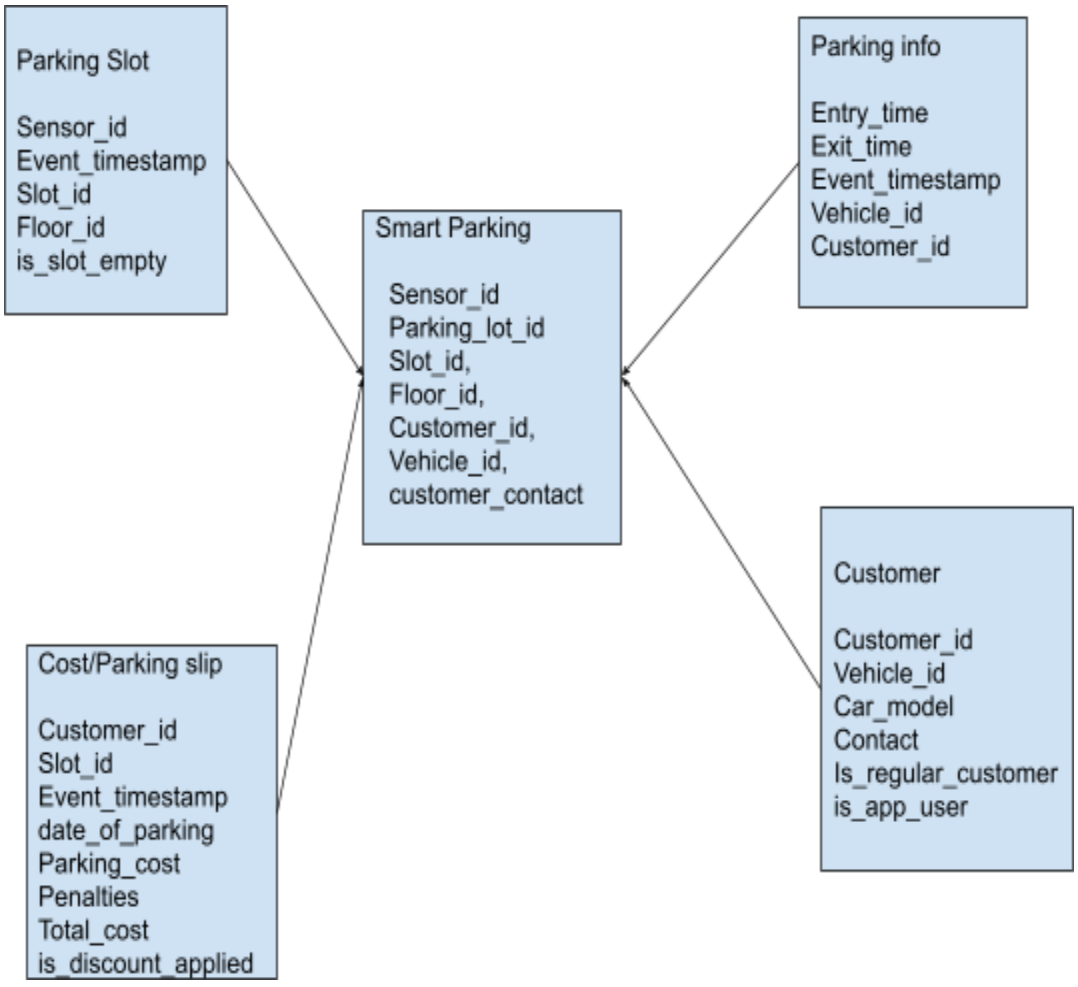
1. The major tracking events would be four different tables:

    - Parking slot - This would provide the sensors real-time streaming data in a period every second that would tell us if the parking slots are filled or empty. It will send the streaming data to backend tables that will be used for prediction model.
    - Customer/Client - As soon as clients scan the barcode it will send the customer data to backend tables that would include the information of client like customer_id, vehicle_id, customer_contact etc.
    - Parking Information - This would provide the details about the entry & exit timestamp of a vehicle & the parking slot information about the vehicle.
    - Cost/parking slip - This would provide the information about the parking slip and costs, penalties paid by customers

    Below is the data modeling of the above scenario which uses star schema.

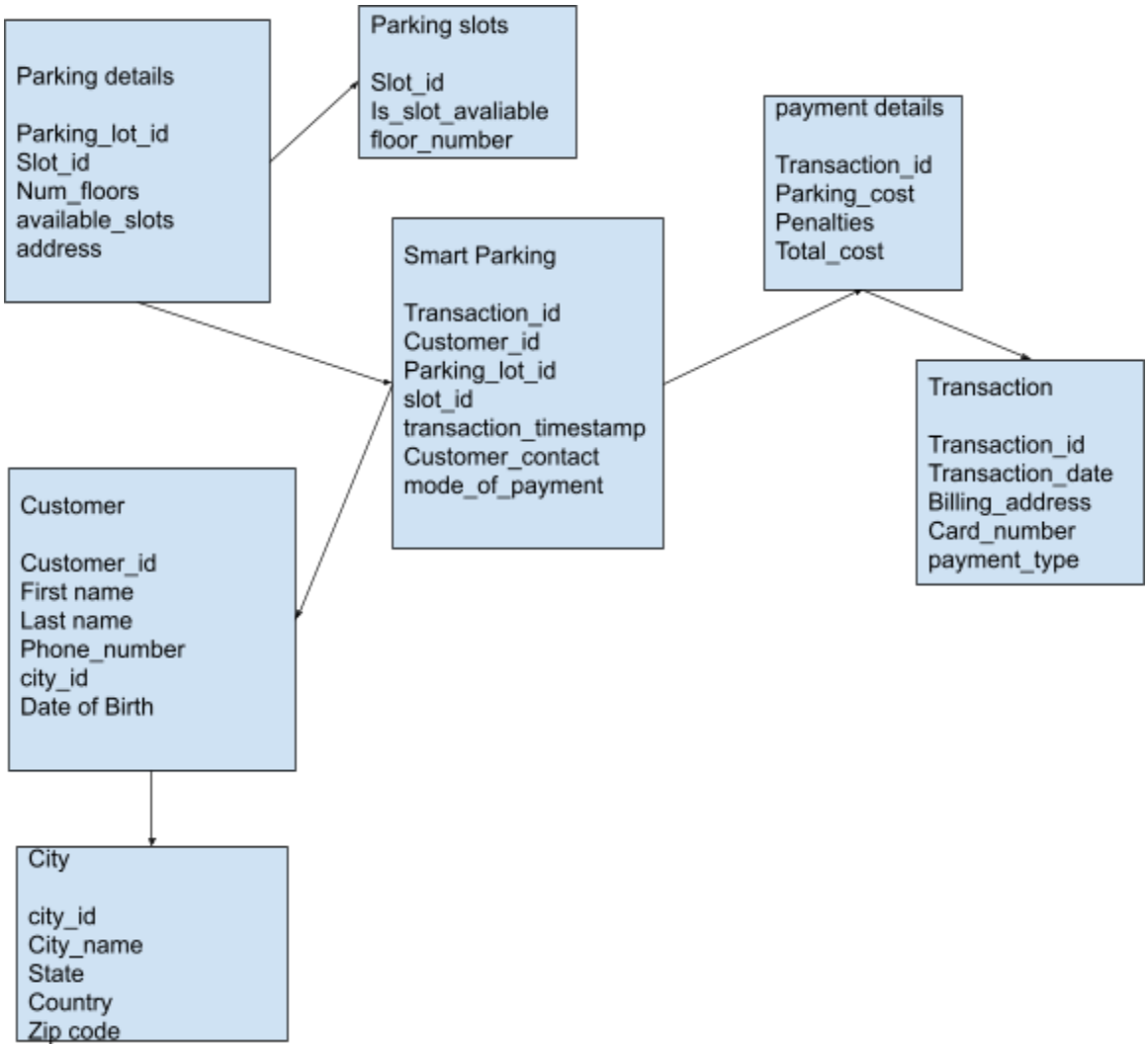


For the above event analytics the technologies that can be used in the process are

- Python requests/SQL queries - if the data is exposed using APIs or SQL queries if the data is fetched from the tables in a database.
- Kafka - to create a message queues for real-time streaming data from sensors
- Spark/Beam - to analyze the streaming data for analytics purpose
- BigQuery/Redshift - A fast and efficient scalable analytics data warehouse

2. The backend system would be using the microservice architecture where each component of the app would acts as a service and there is a service layer called API Gateway that synchronize the services and aggregates the data between different services in the application. The operation system would be mostly storing the transactional data like payments, costs etc

   The model of operational system is defined below and it will follow the snowflake schema



For backend systems and operational systems below technologies can be used

- Frameworks like Django, Flasks for backend
- REST APIs
- Relational Data warehouse like Postgres, Oracle

3. We can combine the two architecture of operational systems and analytics architecture by creating a reliable fault-tolerant ETL (Extract Transform Load) pipelines and combining the transactional data with analytical data in distributed and scalable databases which enables faster querying of the tables and then the combined data can be used for generating reports and machine learning modelling.The historic data can be used for creating the prediction model for the Smart parking systems.

4. The development lifecycle can be done in an agile fashion, where the team delivers the small but consumable increments. The development cycle would be done in sprints of two weeks with one scrum master and product. The requirement for a project or feature would be gathered from Stakeholders and then tickets will be created in the sprint and will be saved in Backlog. With each sprint the features, bug fixes will be delivered.

   The test and deployment automation would be done using CI/CD pipelines in Gitlab. There will be two separate environments in the system architecture
   - Staging
   - Production

The developer can create a local branch, commit changes and merge into staging branch once he tests his code and changes locally. Then the CI/CD pipeline will trigger the build and test your code as well as data before deploying to staging environment. And the pipelines would be tested in Staging before moving to production.

The similar process would be followed for Production, with a slight difference that merge to production would be approved by lead developer or product manager as the changes will be reflected in live product.